Universidad EAFIT

Doctoral Publication

Compendium of Publications on:

Differential Operators on Manifolds for CAD CAM CAE and Computer Graphics

Supervisor
Prof. Oscar E. Ruiz Salguero,
Universidad EAFIT
Co-supervisor:
Dr. Jorge L. Posada Velásquez,
Vicomtech

Doctoral Student:
Daniel Mejia Parra

Jury:
Prof. Carlos A. Cadavid Moreno,
Universidad EAFIT
Dr. Jairo R. Sánchez Tapia,
Vicomtech

Dissertation
Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering in the College of Engineering of the Universidad EAFIT

UNIVERSIDAD EAFIT
COLLEGE OF ENGINEERING
DOCTORAL PROGRAM IN ENGINEERING
MEDELLIN, COLOMBIA
MAY 2020

Dedication

Dedicated to my beloved parents Mauricio and Liliam, and my sister Mariana, for their strong support and sacrifice all these years.

Acknowledgments

I want to deeply thank my PhD advisors, Prof. Oscar Ruiz and Dr. Jorge Posada for their support through this whole doctoral process, their patience, trust, understanding, compromise, and their never-ending efforts in challenging and motivating me.

I also want to thank the Directors of Vicomtech, Dr. Jorge Posada and Prof. Julián Flórez for giving me the opportunity and trust to join their valuable research team. Thanks to all Vicomtech's staff, specially Dr. Aitor Moreno, Dr. Jairo Sánchez, Iñigo Barandiaran, Ander Arbelaiz, and all the Industry and Advanced Manufacturing members, who strongly supported this process in terms of academic, professional and personal growth.

I am genuinely thankful to Universidad EAFIT and Vicomtech as institutions for their trust and support to my master and doctoral studies.

I am grateful to my colleagues at the CAD CAM CAE Laboratory for their support and partnership. Special thanks to Profs. Carlos Cadavid and Jorge Lalinde for their support and advice. In addition, special thanks to my colleague Santiago Moreno for his help in assembling and formatting this Thesis.

I want to sincerely thank the Jury members, Prof. Oscar Ruiz, Dr. Jorge Posada, Prof. Carlos Cadavid and Dr Jairo Sánchez for their time and effort evaluating this Thesis, and for their valuable comments. Many thanks to all the doctoral team staff. Without them, the results of this research would not have been possible.

Finally, I am wholeheartedly thankful to my family for their love and constant support. In addition, special thanks to my friends Andoni, Sheyla, Alex E. and Alex L., who cheered and supported me in very tough times.

Contents

Part I
Additional Documentation

I-A

PhD Defense Flyer

UNIVERSIDAD EAFIT®

The College of Engineering of

EAFIT University

announces the

final examination of

Daniel Mejia Parra

for the degree of

Doctor of Philosophy in Engineering

Wednesday, May 20, 2020
at 1700h (Central European Time, GMT+2)
1000h (Colombia Time, GMT-5)
https://global.gotomeeting.com/join/875776541
Colombia / Spain

COLOMBIA

Major field of Study
Computational Geometry

Dissertation
Differential Operators on Manifolds for CAD CAM CAE and Computer Graphics

Dissertation committee chairperson
Prof. Dr. Eng. Oscar Ruiz Salguero
Universidad EAFIT, Colombia

Directors of dissertation research
Prof. Dr. Eng. Oscar Ruiz Salguero
Universidad EAFIT, Colombia

Dr. Eng. Jorge Posada
Vicomtech, Spain

Examining committee
Prof. Dr. Math. Carlos A. Cadavid Moreno
Universidad EAFIT, Medellín, Colombia

Dr. Eng. Jairo R. Sánchez Tapia
Vicomtech, Donostia-San Sebastián, Spain

President of the Board of the Doctoral Program in Engineering
Dean Ricardo Taborda Ríos

This examination is open to the public

ABSTRACT

This Doctoral Thesis develops novel articulations of Differential Operators on Manifolds for applications on Computer Aided Design, Manufacture and Computer Graphics, as follows:

(1) Mesh Parameterization and Segmentation. Development and application of Laplace-Beltrami, Hessian, Geodesic and Curvature operators for topology and geometry – driven segmentations and parameterizations of 2-manifold triangular meshes. Applications in Reverse Engineering, Manufacturing and Medicine.

(2) Computing of Laser-driven Temperature Maps in thin plates. Spectral domain - based analytic solutions of the transient, non-homogeneous heat equation for simulation of temperature maps in multi-laser heated thin plates, modeled as 2-manifolds plus thickness.

(3) Real-time estimation of dimensional compliance of hot out-of-forge workpieces. A Special Orthogonal SO(3) transformation between 2-manifolds is found, which enables a distance operator between 2-manifolds in R^3 (or m-manifolds in R^n). This process instruments the real-time assessment of dimensional compliance of hot workpieces, in the factory floor shop.

(4) Slicing or Level-Set computation for 2-manifold triangular meshes in Additive Manufacturing. Development of a classification of non-degenerate (i.e. non-singular Hessian) and degenerate (i.e. singular Hessian) critical points of non-Morse functions on 2-manifold objects, followed by computation of level sets for Additive Manufacturing.

Most of the aforementioned contributions have been screened and accepted by the international scientific community (and published). Non-published material corresponds to confidential developments which are commercially exploited by the sponsors and therefore banned from dissemination.

VITAE

Doctoral Student Daniel Mejia Parra
Daniel Mejia (Medellín, Colombia) obtained the B.Sc. degree in Engineering Mathematics (2013) from U. EAFIT (Colombia). He joined the CAD CAM CAE Laboratory at U. EAFIT in 2014, obtaining the M.Sc in Engineering (2016) under supervision of Prof. O. Ruiz. In 2016, he started the Ph.D. program in Computational Geometry at CAD CAM CAE Lab. During his M.Sc. and Ph.D. programs, Daniel executed a 48-month Internship in Vicomtech, Spain, which allowed additional trainings in Europe, and a 12-month post-internship doctoral stage at U. EAFIT, fully sponsored by Vicomtech and U. EAFIT.

Doctoral Supervisor Prof. Dr. Eng. Oscar Ruiz Salguero
Prof. Oscar Ruiz (Tunja, Colombia) earned B.Sc. degrees in Mechanical Engineering and Computer Science (1983, 1987) at Los Andes University, Colombia, a M.Sc. degree (1991) and a Ph.D. (1995) from the Mechanical Eng. Dept. of University of Illinois at Urbana- Champaign, USA. He has been Visiting Researcher at Ford Motor Co. (USA), Fraunhofer Inst. for Computer Graphics (Germany), University of Vigo (Spain), Max Planck Inst. for Informatik (Germany) and Purdue University (USA). Prof. Ruiz is a Colciencias Senior Investigator and coordinates the CAD CAM CAE Laboratory at U. EAFIT. His research and teaching interests are Computational Geometry applied on Design, Manufacturing, Medicine, etc.

Doctoral Co-Supervisor Dr. Eng. Jorge L. Posada Velásquez
Dr. Jorge Posada is the Scientific and Associate Director of Vicomtech (Spain) since 2001, and a member of the BRTA (Basque Research and Technology Alliance). He holds a Ph.D. in Engineering and Computer Science from the Technische Universitat Darmstadt (Germany), and an Executive MBA from IE Business School. He is the president of the Board of GraphicsVision.ai. Author of over 90 scientific publications. Jorge Posada is a member of the Editorial Boards in IJIDeM (Springer), Multimedia Tool and Applications (Springer), Sensors (MDPI), and the Applied Sciences Journal (MDPI).

I-B

PhD Defense Approval

FINAL EXAM OF DISSERTATION

DOCTORAL PROGRAM IN ENGINEERING

EAFIT UNIVERSITY

Doctoral Student:	**Daniel Mejia Parra** **Student code: 201620003125**
Thesis Supervisors:	**Prof. Dr. Oscar Ruiz Salguero (U. EAFIT)** **Dr. Ing. Jorge Posada Velasquez (Vicomtech)**
Topic of Examination:	**Differential Operators on Manifolds for CAD CAM CAE and Computer Graphics**
Room:	**Tele-conference:** **https://global.gotomeeting.com/join/875776541**
Date:	**20-May-2020**
Time:	**1700h (Central European Time, GMT+2)** **1000h (Colombia Time, GMT-5)**
Jury:	**Prof. Dr. Carlos Cadavid Moreno** **Dr. Eng. Jairo R. Sánchez Tapia** **Dr. Eng. Jorge Posada Velásquez** **Prof. Dr. Oscar Ruiz Salguero**

The Jury considers that the Doctoral Student has PASSED (PASSED/FAILED) this Final Examination, faced by the Doctoral Student in the mentioned place and date.

Juror
Prof. Dr. Carlos Cadavid M.

Juror
Dr. Eng. Jairo Sanchez T.

Juror and
Thesis Co-Supervisor
Dr. Eng. Jorge Posada V.

Juror and
Thesis Supervisor
Prof. Dr. Oscar Ruiz S.

Witnesses:

Prof. Dr. Edgar Alexander Ossa H.
Chair of the Examination Committee
U. EAFIT, COLOMBIA

President of the Doctoral Board
Dean Dr. Eng. Ricardo Taborda Ríos

Part II
Introduction

II-A

Goal of the Final Examination

Under the regulations of the Doctoral Program in Engineering at U. EAFIT, the purpose of the Final Examination is to assess the thesis work of the doctoral student, which should reflect the capacity of the student to: (I) conduct high-quality scientific research, (II) contribute to the state of the art, and (III) articulate in novel manners the existing knowledge to advance in the formulation and solution of theoretic and practical problems in the Engineering domain.

The Final Exam assesses these aspects:

1. The academic trajectory undertaken and opportunities profited by the doctoral student during the doctoral studies, in terms of (a) Doctoral Courses, (b) Special Trainings, (c) Attendance to Specialized Forums and Industries (d) Equipment, Software, accessory materials, (e) Funding Proceedings, (f) Special Advisors, etc.

2. The thematically connected results of the research of the student and the doctoral team, and the endorsement of the international scientific community to these results, in the form of ranked publications.

The Jury approves or reproves the thesis work of the doctoral student.

II-B

Organization of this Document

The remaining of this document is organized as follows:

- Part III: Academic Trajectory. This part reports the following aspects of the doctoral process:
  (a) List of Publications and Co-authors
  (b) Doctoral Courses
  (c) Special Trainings
  (d) Attendance to Specialized Forums
  (e) Special Advisors
  (f) Projects
  (g) Distinctions
- Part IV: Research Results. This part provides:
  (a) An overview of the domains in which the doctoral investigation has been conducted.
  (b) The compendium of publications generated in each of the investigated domains.
- Part V: General Conclusions.

Part III
Academic Trajectory

III-A

Academic History

III-A.1

Summary

1. In December of 2013, Daniel Mejia obtained his bachelor degree in Engineering Mathematics for Universidad EAFIT.
2. In January of 2014, Daniel joined the CAD CAM CAE Laboratory of Universidad EAFIT and began his Master in Science studies in Engineering under the supervision of Prof. Dr. Eng. Oscar Ruiz.
3. From July 2015 to June 2016, Daniel undertook a research internship at the Industry and Advanced Manufacture department of Vicomtech (Spain), under the supervision of Dr. Jorge Posada.
4. In June of 2016, Daniel obtained his M.Sc. Degree in Engineering from Universidad EAFIT.
5. In July of 2016, Daniel started his Doctoral studies in Engineering at Universidad EAFIT under the supervision of Prof. Dr. Eng. Oscar Ruiz (U. EAFIT, Colombia) and Dr. Eng. Jorge Posada (Vicomtech, Spain).
6. From July 2016 to July 2019, Daniel continued his research internship at the Industry and Advanced Manufacture department of Vicomtech (Spain) as part of his Doctoral Thesis.
7. From July 2015 to June 2020, Daniel received collaborative funding as part of a joint sponsorship provided by Universidad EAFIT and Vicomtech for his M.Sc. and Ph.D. studies.

In the framework of the collaborative program between EAFIT and Vicomtech, the student and his doctoral support team have achieved several publications, formalizing the doctoral work of the student. The mentioned doctoral support team is composed by the doctoral supervisors of the student and several researchers from EAFIT and Vicomtech.

III-A.2

List of Publications

Table III-A.2.1: List of published articles of the doctoral support team.

Item	Bibliographic Information	Type / Status	Indexing / Qualification
1	Daniel Mejia-Parra, Ander Arbelaiz, Oscar Ruiz-Salguero, Juan Lalinde-Pulido, Aitor Moreno and Jorge Posada. Fast simulation of laser heating processes on thin metal plates with FFT using CPU/GPU hardware. *Applied Sciences*, **2020**, 10(9), pp. 3281:1-3281:25. DOI: 10.3390/app10093281.	Journal Article / Published	ISI (Q2), SCOPUS(Q1), Publindex(A1)
2	Daniel Mejia-Parra, Ander Arbelaiz, Oscar Ruiz-Salguero, Aitor Moreno and Jorge Posada (**2020**). DST and FFT - based algorithms for laser heating simulation on thin metal plates in GPU. In *Mathematics And Computers in Science & Engineering (MACISE 2020)*. January 18-20, Madrid, Spain.	International Conference / Published.	Indexing Pending
3	Daniel Mejia-Parra, Jairo R. Sánchez, Jorge Posada, Oscar Ruiz-Salguero and Carlos Cadavid. Quasi-isometric mesh parameterization using heat-based geodesics and Poisson surface fills. *Mathematics*, **2019**, 7(8), 753. DOI: 10.3390/math7080753.	Journal Article / Published	ISI(Q1), SCOPUS(Q2), Publindex(A1)
4	Daniel Mejia-Parra, Juan Lalinde-Pulido, Jairo R. Sánchez, Oscar Ruiz-Salguero and Jorge Posada (**2019**). Perfect Spatial Hashing for point-cloud-to-mesh registration. In *Spanish Computer Graphics Conference (CEIG 2019)*. June 26-28, Donostia - San Sebastián, Spain. ISBN: 978-3-03868-093-2. DOI: 10.2312/ceig.20191202.	International Conference / Published	EUROGRAPHICS Digital Library

Continued on next page

Table III-A.2.1 – *Continued from previous page*

Item	Bibliographic Information	Type / Status	Indexing / Qualification
5	Daniel Mejia-Parra, Aitor Moreno, Jorge Posada, Oscar Ruiz-Salguero, Iñigo Barandiaran, Juan Carlos Poza and Raúl Chopitea. Frequency-domain analytic method for efficient thermal simulation under curved trajectories laser heating. *Mathematics and Computers in Simulation*, **2019**, 166. DOI: 10.1016/j.matcom.2019.05.006.	Journal Article / Published	ISI(Q2), SCOPUS(Q2), Publindex(A1)
6	Daniel Mejia-Parra, Jairo R. Sánchez, Oscar Ruiz-Salguero, Marcos Alonso, Alberto Izaguirre, Erik Gil, Jorge Palomar and Jorge Posada. In-line dimensional inspection of warm-die forged revolution workpieces Using 3D Mesh Reconstruction. *Applied Sciences*, **2019**, 9(6), pp. 1069:1-1069:21. DOI: 10.3390/app9061069.	Journal Article / Published	ISI (Q2), SCOPUS(Q1), Publindex(A1)
7	Daniel Mejia-Parra, Diego Montoya-Zapata, Ander Arbelaiz, Aitor Moreno, Jorge Posada and Oscar Ruiz-Salguero. Fast analytic simulation for multi-laser heating of sheet metal in GPU. *Materials*, **2018**, 11, pp. 2078:1-2078:19. DOI: 10.3390/ma11112078.	Journal Article / Published	ISI(Q2), SCOPUS(Q1), Publindex(A2)
8	Daniel Mejia, Oscar Ruiz-Salguero, Carlos Cadavid, Jairo R. Sánchez, Jorge Posada and Diego A. Acosta (**2018**). Mesh segmentation driven by bijective parameterization. In *Proceedings of the 12th International Symposium on Tools and Methods of Competitive Engineering (TMCE 2018)*. May 7-11, Las Palmas de Gran Canaria, Spain. ISBN: 978-94-6186-910-4. **Award Best Senior Paper Contribution**	International Conference / Published	Award Best Senior Paper Contribution
9	Daniel Mejia, Oscar Ruiz-Salguero, Jairo R. Sánchez, Jorge Posada, Aitor Moreno and Carlos A. Cadavid. Hybrid geometry / topology based mesh segmentation for reverse engineering. *Computers & Graphics*, **2018**, 73, pp. 47-58. DOI: 10.1016/j.cag.2018.03.004.	Journal Article / Published	ISI(Q3), SCOPUS(Q1), Publindex(A1)
10	Daniel Mejia, Aitor Moreno, Ander Arbelaiz, Jorge Posada, Oscar Ruiz-Salguero and Raúl Chopitea. Accelerated thermal simulation for three-dimensional interactive optimization of computer numeric control sheet metal laser cutting. *Journal of Manufacturing Science and Engineering*, **2018**, 140, pp. 031006:1-031006:9. DOI: 10.1115/1.4038207.	Journal Article / Published	ISI(Q2), SCOPUS(Q1), Publindex(A1)

Continued on next page

Table III-A.2.1 – *Continued from previous page*

Item	**Bibliographic Information**	**Type / Status**	**Indexing / Qualification**
11	Daniel Mejia, Diego A. Acosta and Oscar Ruiz-Salguero. Weighted area/angle distortion minimization for mesh parameterization. *Engineering Computations*, **2017**, 34 (6), pp. 1874-7895. DOI: 10.1108/EC-02-2016-0072.	Journal Article / Published	ISI(Q3), SCOPUS(Q2), Publindex(A1)
12	Daniel Mejia, Jairo R. Sánchez, Álvaro Segura, Oscar Ruiz-Salguero, Jorge Posada and Carlos Cadavid (**2017**). Mesh segmentation and texture mapping for dimensional inspection in Web3D. In *Proceedings of the 22nd International Conference on 3D Web Technology (Web3D ´17)*. June 05-07, Brisbane, Australia. ISBN: 978-1-4503-4955-0. DOI: 10.1145/3055624.3075954.	International Conference / Published	SCOPUS, dblp, Semantic Scholar

III-A.2.1 List of Co-authors

Aside from the Ph.D. Supervisors, this compendium of publications has the following co-authors.

Table III-A.2.2: Co-authors of this compendium of publications.

Name	**Affiliation**
Diego A. Acosta	Grupo de Desarrollo y Diseño de Procesos, Universidad EAFIT
Ander Arbelaiz	Industry and Advanced Manufacturing, Vicomtech
Marcos Alonso	Computational Intelligence Group, University of the Basque Country UPV/EHU
Iñigo Barandiaran	Industry and Advanced Manufacturing, Vicomtech
Carlos Cadavid	Departamento de Matemáticas Aplicadas, Universidad EAFIT
Raúl Chopitea	Lantek R&D, Lantek Sheet Metal Solutions
Erik Gil	GKN Driveline Legazpi S.A.
Alberto Izaguirre	CIS & Electronics Department, University of Mondragon
Juan G. Lalinde	High Performance Computing Facility APOLO, Universidad EAFIT
Diego Montoya	Laboratorio de CAD CAM CAE, Universidad EAFIT
Aitor Moreno	Industry and Advanced Manufacturing, Vicomtech
Jorge Palomar	GKN Driveline Legazpi S.A.
Juan C. Poza	Lantek R&D, Lantek Sheet Metal Solutions
Jairo R. Sánchez	Industry and Advanced Manufacturing, Vicomtech
Álvaro Segura	Industry and Advanced Manufacturing, Vicomtech

III-A.3

Doctoral Courses

III-A.3.1 Preparatory Courses

According to the regulations of U. EAFIT, the courses that prepare the student to perform his doctoral thesis are taken during the M.Sc. and Ph.D. programs. The preparatory courses that the student took are presented in Table III-A.3.1:

Table III-A.3.1: M.Sc.(2014-1 to 2016-1) and Ph.D. (2016-2 to 2019-1) Preparatory Courses.

Course	**Semester**
Dimensional Reduction	2014-1
Advanced Continuum Mechanics	2014-2
Computational Geometry	2014-2
Introduction to the Boundary Element Method	2014-2
Computer Aided Geometric Design I	2015-1
Introduction to the Finite Element Method	2015-1
Optimization Techniques	2015-1
Advanced Data Structures and Algorithms	2015-2
Numerical Solutions of Differential Equations	2016-1
Underlying Mathematics for CAD CAM	2016-2
Non-linear Finite Element Method	2017-1
Graph Algorithms	2017-2
Modern Concepts of Programming in Engineering	2018-1
Parallel Computing	2018-2
Study and Application of General Programming using the Graphics Processor	2019-1

III-A.3.2 Qualifying Exams

Starting the second year of the doctoral program, the student prepared, took and approved the doctoral qualifying exams that are reported in Table III-A.3.2:

Table III-A.3.2: Qualifying Exams.

Exam	**Date**	**Examiner**
Numerical Computations	June 2017	Prof. Dr. Eng. Juan D. Jaramillo, U. EAFIT
Graph Algorithms	November 2017	Dr. Eng. Hugo Álvarez, Vicomtech

III-A.3.3 Preliminary Exam of Dissertation

During the fourth year of the doctoral studies, the student prepared, presented and approved the Preliminary Exam of Dissertation. The Preliminary Examination assessed: (a) the academic trajectory undertaken and opportunities profited by the doctoral student during the first 36 months of the doctoral studies, (b) the thematically connected results of the research of the student and the doctoral team in the form of ranked publications, and (c) the closure research activities and goals of the doctoral student and supporting team for the remaining 12 months (approx.).

On 24-09-2019 the Jury decided to permit the doctoral student to continue the academic and research activities, in order to prepare and perfect the materials, goals, publications, etc. for the Final Examination.

III-A.4

Special Trainings

As part of the doctoral formation, the student has undertaken the trainings presented in Table III-A.4.1:

Table III-A.4.1: Special Trainings.

Topic	**Entity-Context**	**Date**	**Supervisor**
Interactive Simulation of Sheet Metal Laser cutting	Lantek Sheet Metal Solutions (Vitoria-Gasteiz, Spain), Vicomtech (San Sebastián, Spain)	Nov. 2015	Eng. Raúl Chopitea, Eng. Juan C. Poza
Agile & SCRUM Methodologies	Agilar (Madrid, Spain), Vicomtech (San Sebastián, Spain)	Jun. 2017	Eng. Jose R. Díaz
GIT and GITLAB Formation	Agilar (Madrid, Spain), Vicomtech (San Sebastián, Spain)	Nov. 2017	Eng. Juan B. Barrera
Dimensional Inspection of Warm-forged Revolution Work-pieces in the Automotive Industry	GKN Driveline (Legazpi, Spain), Vicomtech (San Sebastián, Spain)	Jul. 2018 - Feb. 2019	Eng. Jorge Palomar
Computational Geometry Techniques in Medical (Dentistry) Applications	BTI Biotechnology Institute (Vitoria, Spain), Vicomtech (San Sebastián, Spain)	Sep. 2018 - Jun. 2019	Eng. Luis Saracho
Geomagic Freeform 2019	Improto 3D (Medellín, Colombia), U. EAFIT (Medellín, Colombia)	Jul. 2019	Andres M. Quiceno

III-A.5

Attendance to Specialized Forums

During the doctoral formation, the student has attended to the following specialized forums:

III-A.5.1 Scientific Conferences

1. TMCE 2018: Twelfth International Symposium on Tools and Methods of Competitive Engineering, May 7-11, 2018, Las Palmas de Gran Canaria, Spain. **Award Best Senior Paper Contribution.**
2. CEIG 2018: Spanish Computer Graphics Conference, June 27-29, 2018, Madrid, Spain.

III-A.5.2 Professional Forums

1. FORJA: Jornada Análisis FEM de Uniones Mecánicas. CAE Innovación - Grupo AyS (Analysis & Simulation). September 2016. Amorebieta-Etxano, Spain.
2. FORJA: Jornada Análisis de Problemática Común en la Industria de Forja. CAE Innovación - Grupo AyS (Analysis & Simulation). September 2016. Bergara, Spain.
3. Master Internship: Vicomtech. Industry and Advanced Manufacturing. July 2015 - June 2016. Donostia - San Sebastián, Spain.
4. Doctoral Internship: Vicomtech. Industry and Advanced Manufacturing. July 2016 - June 2019. Donostia - San Sebastián, Spain.

III-A.6

Special Advisors

In addition to the support provided by several professors and researchers from EAFIT and Vicomtech, the student was advised by the specialists presented in Table III-A.6.1.

Table III-A.6.1: Special Advisors.

Name	**Role**	**Entity**	**Topic**
Prof. Dr. Eng. Diego A. Acosta	Scientific Coordinator and Advisor	Design and Development of Processes and Products Research Group (DDP) EAFIT, Colombia.	Optimization Techniques. Operations Research. Sensitivity Analysis.
Prof. Dr. Carlos A. Cadavid	Scientific Coordinator and Advisor	Mathematics and Applications, Department of Mathematical Sciences, EAFIT, Colombia	Differential Operators. Topology. Spectral Analysis. Graph Theory. Morse Theory.
Prof. Dr. Juan G. Lalinde	Scientific Coordinator and Advisor	High Performance Computing Facility APOLO, EAFIT, Colombia.	Parallel Computing (Architecture / Technique). Data Structures. High Performance Programming.
Dr. Eng. Aitor Moreno	Senior Researcher. Head of Interactive Laser Simulation (BEROSIM) and Medicine (FERULAS3D) projects.	Industry and Advanced Manufacturing, Vicomtech, Spain.	Thermal Simulation of Sheet Laser Cutting. Mesh Segmentation. Interactive Simulation.
Dr. Eng. Jairo R. Sánchez	Senior Researcher. Head of Geometry Processing Library (GEOMLIB) and Dimensional Assessment (HOTINSPECT) projects.	Industry and Advanced Manufacturing, Vicomtech, Spain.	Mesh Registration. Dimensional Inspection. Computational Geometry Algorithms.

III-A.7

Projects

During the student internship at Vicomtech, the student has participated in projects related to the areas of Computational Geometry, CAD CAM CAE, Industry / Manufacturing and Medicine:

1. BEROSIM: Development of an interactive 3D simulator of CNC sheet metal laser cutting operations. Fast / interactive simulation of geometric and thermal behavious of the sheet metal caused by the high input energies from the cutting laser.

2. HOTINSPECT: Development of dimensional inspection software tools for in-line inspection of warm-forged work-pieces, common in the automotive industry. Application of Computer Vision and Computational Geometry techniques (such as mesh registration) for dimensional assessment of manufactured parts.

3. GEOMLIB: Development of different Computational Geometry data structures (e.g. Perfect Spatial Hashing) and algorithms (e.g. Mesh Parameterization / Segmentation / Deformation / Registration). Contribution to the growth of a Computational Geometry library for point cloud and mesh processing.

4. FERULAS3D: Applications of Computational Geometry in medicine (confidential).

III-A.8

Distinctions

The following table presents the distinctions and acknowledgements that the student and the doctoral team achieved during the doctoral process:

Table III-A.8.1: Distinctions during the Doctoral process

Distinction	**Context**	**Observations**
Scholarship. EAFIT M.Sc. and Ph.D.	EAFIT University (2014-2 to 2020-1)	Renewed each semester upon student meeting academic performance and publication requirements.
Sponsorship. Vicomtech M.Sc. and Ph.D.	Vicomtech/EAFIT Agreement (2015-2 to 2019-1)	Renewed each year upon student meeting academic performance, publications and industry projects participation requirements.
Exceptional 12-month Sponsorhip. Vicomtech Ph.D.	Vicomtech (2019-2 to 2020-1)	Additional 12-month funding exceptional to the Vicomtech/EAFIT agreement (outside of the original 48-month funding). The student performs away from the research center (at U. EAFIT).
Award: Best Senior Paper Contribution	International Symposium on Tools and Methods of Competitive Engineering (TMCE 2018)	May 7-11, Las Palmas de Gran Canaria, Spain.

TMCE 2018

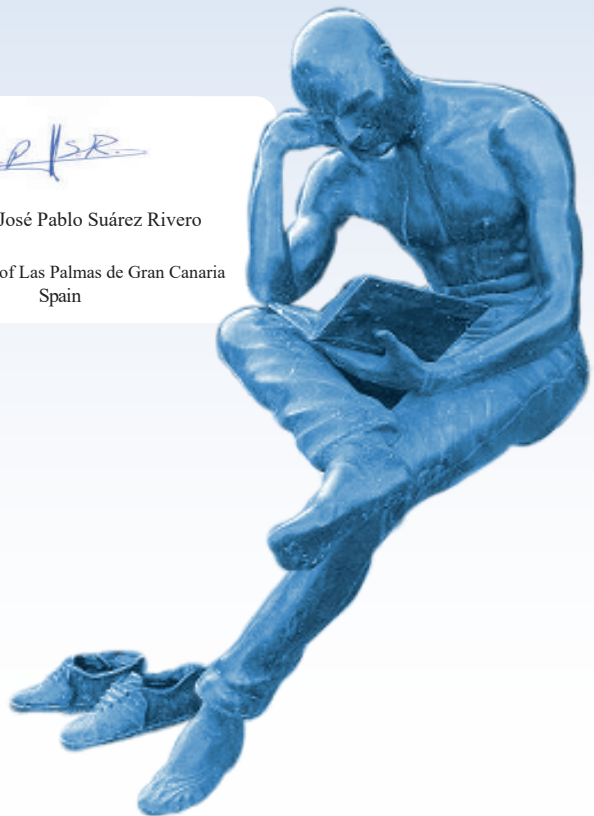
Tools and Methods of Competitive Engineering

Best Senior Contribution Award

Mesh Segmentation Driven By Bijective Parameterization

D. Mejia **O. Ruiz-Salguero** **C. Cadavid**
J.R. Sánchez **J. Posada** **D. Acosta**

The Organizing Committee of the International Tools and Methods of Competitive Engineering Symposia has the pleasure to recognize your outstanding contribution to the past events as well as to the TMCE 2018 Symposium.



Prof. Dr. Imre Horváth
co-chairman
Delft University of Technology
the Netherlands

Prof. Dr. José Pablo Suárez Rivero
University of Las Palmas de Gran Canaria
Spain

Part IV
Research Results

IV-A

Context

This Doctoral Thesis develops novel articulations of Differential Operators on Manifolds and other Computational Geometry techniques for applications on Computer Aided Design and Manufacture. The Mathematical and Computer Science concepts applied in this Thesis are presented in Table IV-A.

Mathematical / Computer Science Topic	**Area of Application**
Coordinate Frames	Mesh Parameterization, Mesh Registration, Dimensional Assessment.
Affine Transformations (area / angle / distance - preserving, rigid)	Mesh Parameterization, Mesh Registration, Dimensional Assessment, 2-Manifold Level Sets.
Quaternion Algebra	Mesh Registration.
Homeomorphic Maps	Mesh Parameterization.
Dimensionality Reduction	Mesh Parameterization.
Curvature Operators (min, max, mean, Gaussian, dihedral angle)	Mesh Segmentation
Differential Geometry Operators (Tangent derivative, Laplace-Beltrami, Tangent Hessian)	Mesh Parameterization, Mesh Segmentation, Laser Cutting Simulation, 2-Manifold Level Sets.
Morse Theory	2-Manifold Level Sets
Spatial Partition Techniques (e.g. Octrees, Perfect Spatial Hashing)	Mesh Registration, Dimensional Assessment.
Non-linear Optimization	Mesh Parameterization, Mesh Registration
Finite Element Method (FEM)	Mesh Parameterization, Mesh Segmentation, Laser Cutting Simulation.
Spectral Analysis / Generalized Singular Value Decomposition (SVD)	Mesh Parameterization, Mesh Segmentation, Laser Cutting Simulation.
Parallel Programming Techniques	Laser Cutting Simulation, Mesh Registration.

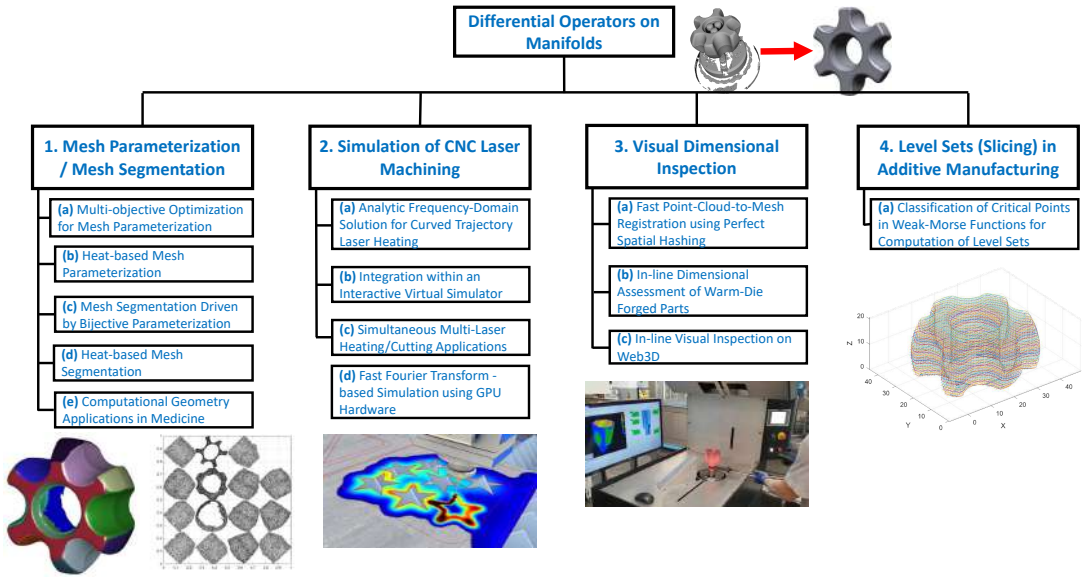


Figure 1: Overview of the problems addressed in this Thesis, their domain of application and location within the Differential Operators on Manifolds framework.

Fig. 1 presents the problems addressed in this thesis, their domain of application, and where they take place within the Differential Operators framework. A brief description of each domain is discussed below:

1. **Mesh Parameterization / Mesh Segmentation**

   (a) Development of a multi-objective non-linear optimization algorithm for the minimization of a distortion-based operator. Such an operator measures the angle/area distortion of a (sought) parameterization $\phi : M \to \mathbb{R}^2$.

   (b) Application of the divergence and Laplace-Beltrami operators for the computation of mesh temperature maps and geodesic fields. The computed geodesics are used to retrieve an isometric (distance-preserving) parameterization of M.

   (c) Development of an iterative algorithm which computes a segmentation of a mesh M based on non-bijective parameterizations of M. These parameterizations are computed from the spectrum (eigenvectors) of a tangent Hessian operator.

   (d) Development of a mesh segmentation algorithm based on the computation of temperature maps on M with temperature constrains. These constraints are set based on curvature fields. Applications of the segmentation/parameterization algorithm include Reverse Engineering and Manufacturing.

   (e) Development and implementation of Computational Geometry algorithms for Medical Applications (confidential).

2. **Simulation of CNC Laser Machining**

(a) Development of a spectral-based analytic solution to the transient, non-homogeneous heat transfer equation in rectangular 2-manifold plates. Laser beams acting on the plate are modelled in time using 1-manifold trajectories.

(b) Integration of the above method within an interactive virtual simulator for visual assessment of the plate temperature and cut geometry. The developed simulator is relevant for the assessment and optimization of CNC laser machining programs.

(c) Extension of the former analytic solution to include the effects of simultaneous (and asynchronous) laser beams, acting on the plate surface. Applications of the developed method include simulation and assessment of multi-beam CNC laser machining programs.

(d) Implementation and optimization of the single- and multi- beam simulation approaches using Fast Fourier Transform algorithms in GPU hardware. Applications include real-time / highly interactive simulation.

3. **Visual Dimensional Inspection**

(a) Implementation of a Perfect Spatial Hashing data structure for the computation of a rigid transformation $SO(3)$ between 2-manifolds embedded in $\mathbb{R}^3$. A distance operator is minimized to compute the named transformation, which "*registers*" an input free mesh against a reference (immovable) mesh.

(b) Implementation of the above registration algorithm for the real-time assessment of dimensional compliance of warm-forged workpieces. Using computer vision, the quality of each forged part is evaluated directly in the factory manufacturing line.

(c) Implementation of heat-based mesh segmentation and Hessian-based mesh parameterization for the application of dimensional deviation fields as texture maps. The computed texture maps enable the deployment of real-time dimensional assessment reports under the Web3D framework.

4. **Level Sets (Slicing) in Additive Manufacturing**

(a) Development of a classification of non-degenerate (i.e. non-singular Hessian) and degenerate (i.e. singular Hessian) critical points of weak-Morse functions on 2-manifold objects. Such classification enables the development of an algorithm for the computation of level sets of 2-manifold triangular meshes. The computed level sets (slices) are relevant in the design of Additive Manufacturing programs.

The description of the specific problems and contributions performed in the Mesh Parameterization / Segmentation, Simulation of CNC Laser Machining, Visual Dimensional Inspection and Level Sets in Additive Manufacturing domains are presented in the subsequent Parts, respectively.

IV-B

Summary of Contributions

This thesis develops and applies techniques from the area of Computational Geometry and Differential Operators on Manifolds to different domains of CAD CAM CAE, Computer Graphics, Industry and Manufacture. The specific domains that are investigated and the contributions performed in each of such domains are the following:

Table 1: Applied domains and main contributions of this thesis.

Applied Domain	**Contribution**
Parameterization / Segmentation of 2-Manifold Triangular Meshes	(1) An angle / area preserving mesh parameterization algorithm based on multi-objective non-linear optimization. (2) A distance-preserving mesh parameterization algorithm based on geodesic fields and heat transfer analysis. (3) A mesh segmentation algorithm driven by non-bijective Hessian parameterizations. (4) A geometry / topology - based mesh segmentation algorithm extracted from temperature fields. (5) Applications of Mesh segmentation / parameterization algorithms in Reverse Engineering, Manufacture and Medicine (Dentistry).
Thermal Simulation of CNC Laser Machining	(1) A spectral-based analytic solution to the curved trajectory laser beam heating problem in rectangular 2-manifold plates. (2) An interactive 3D virtual simulator for metal plate laser cutting that includes geometric and thermal effects. (3) An algorithm for simultaneous application of several laser beams on the plate surface (multi-laser plate heating / cutting). (4) An implementation into GPU hardware using FFT methods for fast interactive laser heating simulation.

Continued on next page

Table 1 – *Continued from previous page*

Applied Domain	**Contribution**
In-line Dimensional Assessment of Forged Parts	(1) An implementation of a Perfect Spatial Hash data structure for fast point-cloud-to-mesh registration of large datasets. (2) An algorithm for in-line dimensional assessment of warm-die forged parts using mesh reconstruction, registration and computer vision. (3) An implementation of mesh segmentation / parameterization algorithms for in-line Web3D dimensional assessment.
Level Sets (Slicing) in Additive Manufacturing	A classification algorithm of non-degenerate and degenerate critical points of weak-Morse functions for the computation of mesh level sets.

IV-C

Parameterization / Segmentation of 2-Manifold Triangular Meshes

IV-C.1

Weighted Area/Angle Distortion Minimization for Mesh Parameterization

Daniel Mejia-Parra[1,2], Diego A. Acosta[1] and Oscar E. Ruiz[1]

[1] Laboratory of CAD CAM CAE, Universidad EAFIT, Colombia

[2] Vicomtech, Spain

Citation

Daniel Mejia, Diego A. Acosta and Oscar Ruiz-Salguero. Weighted area/angle distortion minimization for mesh parameterization. *Engineering Computations*, **2017**, 34 (6), pp. 1874-7895. DOI: 10.1108/EC-02-2016-0072.

Indexing: ISI (Q3), SCOPUS (Q2), Publindex (A1)

Abstract

Mesh Parameterization is relevant for reverse engineering, tool path planning, etc. Current Mesh Parameterization methods usually seek to minimize area, angle and/or distance distortion in the parameterization. Some of these algorithms fix the parameterization boundary inducing high distortions or require an initial valid parameterization. This article presents a free boundary method which does not require a valid initial parameterization and produces locally bijective parameterizations. Our method parameterizes each triangle of a triangular mesh and then maps it to the parameter space by a nonlinear function F, built as a weighted function of area and shape distortion, which is then minimized by the Levenberg-Marquardt algorithm. Complexity analysis shows asymptotically linear behavior in the number of mesh nodes and a sensitivity analysis shows fine-tuning the weighting parameter turns globally non-bijective parameterizations into bijective ones in specific cases. Our test runs show parameterizations with no triangle flips.

Keywords: Reverse Engineering, Mesh Parameterization, Nonlinear Optimization, Levenberg-Marquardt, Complexity Analysis, Sensitivity Analysis.

Glossary

LM:	Levenberg-Marquardt.
I_k:	Identity matrix of degree k.
$\mathcal{O}(f(n))$	Computational complexity of an algorithm being asymptotic to $f(n)$, with n being the measuring unit. In this article, $\mathcal{O}()$ refers to *time* complexity.
M:	Triangular mesh (with boundary) of a 2-manifold embedded in $\mathbb{R}^3$, composed by the set of triangles $T = \{t_1, t_2, \cdots, t_m\}$ with vertex set $X = \{x_1, x_2, \cdots, x_n\}$ ($X \subset \mathbb{R}^3$).
$\phi(x)$:	Parameterization of M which is a piecewise affine mapping $\{\phi_1, \phi_2, \cdots, \phi_m\}$ with $\phi_i = \psi_i \circ \mathcal{R}_i$. $\phi : M \to \mathbb{R}^2$ is an homeomorphism.
U:	Set of points $U = \{u_1, u_2, \cdots, u_n\}$ corresponding to the image of X under the mapping ϕ (i.e., $u_i = \phi(x_i)$).
$\mathcal{R}_i$:	Rigid transformation which maps the triangle t_i to the plane XY and its center of mass $\overline{x}_i$ to the origin.
R^i:	Image of the vertices $[x_{i_1}, x_{i_2}, x_{i_3}]$ of the i-th triangle of M under the mapping $\mathcal{R}_i$.
Q^i:	Right pseudo-inverse of R^i (i.e., $R^i Q^i = I_2$).
ψ_i:	An affine mapping $\mathbb{R}^2 \to \mathbb{R}^2$ which maps R^i to its final place in the parameterization (ϕ). $\psi_i(\varepsilon) = A^i \varepsilon + c_i$.
A^i:	Jacobian matrix of ψ_i.
D^i_{area}:	Area distortion of the triangle t_i under the mapping ϕ_i defined as $D^i_{area} = \big(\det(A^i) - 1\big)^2$.
D^i_{angle}:	Angle distortion of the triangle t_i under the mapping ϕ_i defined as $D^i_{angle} = \big(A^i_{11} - A^i_{22}\big)^2 + \big(A^i_{12} + A^i_{21}\big)^2$.
$F(U)$:	Function $\mathbb{R}^{2n} \to \mathbb{R}$ to be minimized which sums the weighted area and angle distortion of the triangles in M under the mapping $U = \phi(X)$.

F^*:	Value at which the penalty function F is a local minimum.
α:	Parameter $0 \leq \alpha \leq 1$ which weights area distortion (α) against angle distortion ($1-\alpha$) in the penalty function F.
∇:	Gradient operator.
$\mathcal{H}$:	Hessian operator.
λ:	Damping parameter of the LM algorithm.
ε:	Tolerance parameter of the LM algorithm.
S_p^f:	Relative sensitivity of a penalty function f with respect to a p parameter.

IV-C.1.1 Introduction

In CAD CAM CAE, it is usual to have a triangular mesh $M \subset \mathbb{R}^3$ as a result of the segmentation of a larger triangular mesh. M is an open 2-manifold with low curvature (i.e., M is near-developable). Therefore, M admits a 2-variable parameterization which is an homeomorphism between M and $\mathbb{R}^2$.

Mesh Parameterization consists of finding a mapping $\phi : M \to \mathbb{R}^2$ such that: 1) ϕ and ϕ^{-1} are continuous (i.e., connectivity of the triangles is preserved after the mapping) and 2) ϕ is bijective (i.e., triangles do not overlap after the mapping). ϕ is an homeomorphism and the image of M under ϕ is a parameterization of M. In addition, local preservation of properties (e.g., angle, area, dimensions, colinearity, etc.) is pursued but rarely achieved in parameterizations of actual engineering M meshes.

Mesh Parameterization is relevant in areas such as reverse engineering, tool path planning, feature detection, re-design, etc. This article proposes an algorithm for computing ϕ by minimizing a penalty function F which measures a weighted area and shape distortion. Different results may be obtained for the same mesh by changing the weighting parameter α in F. Therefore, fine-tuning of this parameter allows in some cases to recover globally bijective parameterizations.

The remainder of this article is structured as follows: Section IV-C.1.2 reviews the relevant literature. Section IV-C.1.3 describes the implemented methodology. Section IV-C.1.4 presents and discusses the results of the test runs. Section IV-C.1.5 concludes the paper and introduces opportunities for future work.

IV-C.1.2 Literature review

Mesh Parameterization is usually achieved by posing an optimization problem where some kind of distortion measure is minimized in the parameter space. Linear-gradient Mesh Parameterization algorithms seek to reduce to the solution of a single linear system of equations. Several algorithms of this type pose a minimization problem where the boundary is fixed in the parameterization [1–3]. However, fixing the boundary introduces high area and angle distortions in most application cases. The Virtual Boundary algorithm partially overcomes this problem by introducing an artificial boundary connected to the real boundary. Therefore, the real boundary is allowed to move in the parameter space since the algorithm constrains only the virtual boundary. However, the shape of the virtual boundary affects the result and may introduce distortions. Algorithms such as MIPS (Most Isometric ParameterizationS) [4], LSCF (Least Squares Conformal Maps) [5], Linear ABF and ASAP (As Similar As Possible) [6], are linear-gradient algorithms that do not require fixing the boundary. Though, these algorithms only seek conformal parameterizations.

A nonlinear-gradient algorithm arises when the problem is formulated such that the gradient of the function to be minimized is nonlinear. Nonlinear-gradient algorithms usually offer better results than their linear-gradient counterparts at the expense of more computational cost. In addition, global optima cannot be guaranteed which can lead the algorithm to get stuck in a bad parameterization. To avoid getting stuck in such bad parameterization, most nonlinear-gradient algorithms require an initial valid parameterization (i.e., with no triangle flips). A linear-gradient parameterization algorithm is used to compute the initial parameterization overcoming the aforementioned problem in some cases as described in Natural Conformal Maps [1], Quasi-Harmonic Maps [7], ARAP (As Rigid As Possible) algorithm [6] and the Constrained Parameterization on Parallel Platforms algorithm [8]. The Circle Patterns [9], Curvature Prescription [10] and Conformal Equivalence [11] algorithms transfer the gaussian curvature of the whole mesh to a selected set of nodes known as *cone singularities*. However, these approaches require a careful selection of these *cone singularities* which is a non-trivial task usually done manually.

The ABF (Angle-Based Flattening) algorithm [12] is a nonlinear-gradient algorithm which poses an optimization problem in terms of the angles of the mesh triangles in order to find a conformal parameterization. However, the ABF algorithm is computationally expensive making it unusable for large datasets. The ABF++ algorithm [13] and the Linear ABF [14] propose a variation of the original ABF algorithm which potentially improves the computation time at the cost of some global distortion.

Dimensionality Reduction Techniques have been also successfully applied to Mesh Parameterization. These techniques perform a parameterization of a manifold M using the information about the graph of the mesh rather than its triangular structure. In [15] a variation to the Isomap algorithm for triangular meshes is proposed. The problem with Isomap lies in that estimation of geodesics can be computationally expensive and it may present problems for non-convex domains. In [16], Laplacian Eigenmaps and HLLE (Hessian Locally Linear Embedding) are used for mesh parameterization which overcomes the discussed shortcomings of Isomap. However, none of these two algorithms can guarantee preservation of shape or areas which might be a requisite in most applications.

The problem of globally bijective parameterizations cannot be guaranteed in most cases (specially for free boundary methods). The recently proposed Bijective Parameterizations with Free Boundaries algorithm [17] overcomes this problem by posing a nonlinear optimization problem with barrier functions which do not allow boundary overlapping. However, this algorithm becomes highly expensive when evaluating boundary overlaps. In addition, as most nonlinear-gradient parameterization algorithms it requires an initial valid parameterization.

IV-C.1.2.1 Conclusions of the literature review

As dicussed earlier, current Mesh Parameterization algorithms may present one or more of the following disadvantages: local overlaps (triangle flips), non-area preservation in case of conformal maps, requirement of an initial valid parameterization to avoid local minima and non-bijective mappings, etc. In this article we propose a parameterization algorithm where each triangle is mapped individually to the XY plane by a rigid transformation and then a penalty function F measuring distortion in the global parameterization is minimized. Our algorithm however, does not require an initial valid parameterization as opposed to most nonlinear-gradient algorithms and produces no triangle flips. A $0 \leq \alpha \leq 1$ parameter weighs area distortion against $1 - \alpha$ which weighs shape distortion. However, the weighting scheme proposed in this manuscript restricts the

α parameter by a linearly convex combination which potentially avoids numerical instabilities that may arise as opposed to unbounded weighting parameters as in [1, 6].

In order to minimize F we implement the Levenberg-Marquardt (LM) algorithm. LM is a gradient descent method with high convergence rate allowing to evade the computation of an initial valid parameterization which is a requirement in most nonlinear-gradient algorithms. The test runs show that no local overlaps (triangle flips) occur in the resulting parameterization and an adequate fine-tuning of the α parameter results in globally bijective parameterizations in specific cases. In addition, a complexity analysis of our algorithm and a sensitivity analysis for the minimized F^* with respect to α is presented. As far as we know, a sensitivity analysis for weighting parameters has not been presented in the Mesh Parameterization literature yet.

IV-C.1.3 Methodology

Consider $M = (X, T)$ a connected 2-manifold in $\mathbb{R}^3$ with border and possibly with holes which is homeomorphic to $\mathbb{R}^2$. Our goal is to find a set of points $U = \{u_1, u_2, \cdots, u_n\} \subset \mathbb{R}^2$ such that u_i is the image of x_i under a homeomorphism $\phi : M \to \mathbb{R}^2$.

We build ϕ as a piecewise affine mapping (i.e., $\phi(x) = \phi_i(x)$ for all $x \in t_i$, with $\phi_i : t_i \subset M \to \mathbb{R}^2$ being an affine transformation) which highly preserves distances (isometric). Since a mapping is isometric if it is both authalic (preserves areas) and conformal (preserves angles and orientation), we estimate ϕ by minimizing the sum over all the triangles of a weighted area and shape distortion.

We propose an algorithm for finding U described below:

1. **Rigid mapping $\mathcal{R}_i : M \to \mathbb{R}^2$:** Find the rigid transformation $\mathcal{R}_i : M \to \mathbb{R}^2$ that maps the triangle t_i to the XY plane and its center of mass $\overline{x}_i$ to the origin. The matrix R^i corresponds therefore to the image of the vertices of the triangle t_i under such mapping.

2. **Affine mapping $\psi_i : \mathbb{R}^2 \to \mathbb{R}^2$:** Since each triangle has been mapped individually to $\mathbb{R}^2$, an affine mapping $\psi_i(\varepsilon) = A^i\varepsilon + c_i$ which maps each R^i to the final parameterization ϕ is constructed. The Jacobian matrix A^i can be computed in terms of R^i and the vertices $[u_{i_1}, u_{i_2}, u_{i_3}]$ of the triangle $\phi(t_i)$. From this construction, $\phi_i = \psi_i \circ \mathcal{R}_i$ is an affine mapping and $\phi = \{\phi_1, \phi_2, \cdots, \phi_m\}$ is a piecewise affine mapping which parameterizes M. The continuity of ϕ is implied in $\psi = \{\psi_1, \psi_2, \cdots, \psi_m\}$ from the connectivity of M, i.e., if t_i and t_j share the edge (x_k, x_l) then ψ_i and ψ_j overlap in the edge (u_k, u_l).

3. **Weighted penalty function $F(U)$:** A penalty function $F : \mathbb{R}^{2n} \to \mathbb{R}$ which penalizes the weighted area and shape distortion of each triangle is constructed in this step. Since $\phi_i = \psi_i \circ \mathcal{R}_i$ is an affine mapping and $\mathcal{R}_i$ is rigid, all the distortion of ϕ_i can be extracted from A^i. An area (D^i_{area}) and shape (D^i_{angle}) distortion is build for each triangle in terms of A^i and a weighted sum of these terms over all the triangles compose the penalty function F.

4. **Parameterization $U = \phi(X)$:** Since ϕ has minimum distortion, U is estimated by minimizing F. Because ∇F is nonlinear, we implement the LM algorithm for this optimization process.

Each one of the steps described is shown in the fig, IV-C.1.1 and discussed in detail in sections IV-C.1.3.1 through IV-C.1.3.4.

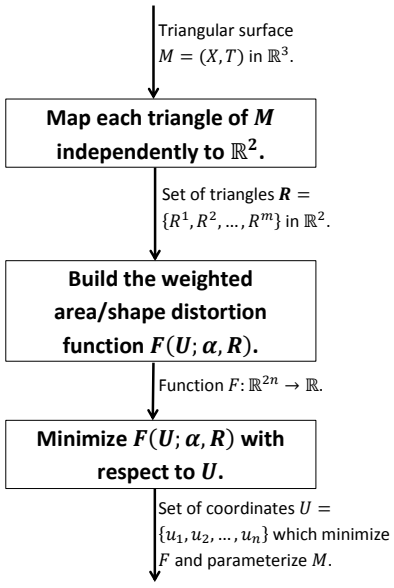


Figure IV-C.1.1: Proposed Mesh Parameterization algorithm.

IV-C.1.3.1 Rigid mapping $\mathcal{R}_i : M \to \mathbb{R}^2$

In order to build the function F, we propose to map each triangle t_i to $\mathbb{R}^2$ individually first. One way to do this is to compute the center of mass $\overline{x}_i$ and the normal vector $\vec{n}_i$ of the triangle t_i. If $B_i = [\vec{v}_1^i, \vec{v}_2^i]$ is an orthonormal basis of the plane with normal $\vec{n}_i$, then $\mathcal{R}_i : \mathbb{R}^3 \to \mathbb{R}^2$ defined as:

$$\mathcal{R}_i(x) = B_i^T(x - \overline{x}_i), \tag{IV-C.1.1}$$

is a projection which maps isometrically the triangle t_i to the plane tangent to t_i (fig IV-C.1.2). Therefore, the matrix R^i corresponds to the image of the current triangle vertices under the map $\mathcal{R}_i$, i.e.:

$$R^i = \mathcal{R}_i([x_{i_1}, x_{i_2}, x_{i_3}]) \tag{IV-C.1.2}$$

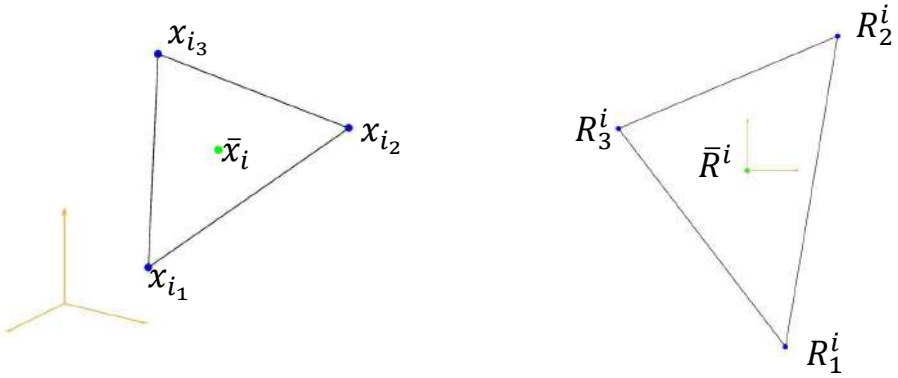
where x_{i_j} is the j-th vertex of t_i.

IV-C.1.3.2 Affine mapping $\psi_i : \mathbb{R}^2 \to \mathbb{R}^2$

ϕ_i is an affine transformation such that $\phi_i = \psi_i \circ \mathcal{R}_i$, where $\psi_i : \mathbb{R}^2 \to \mathbb{R}^2$ is an affine transformation that maps R^i to the global paramaterization U. Therefore:

$$\psi_i(\varepsilon) = A^i\varepsilon + c_i, \tag{IV-C.1.3}$$

where A^i is a 2×2 linear transformation and $c_i = \sum_{j=1}^{3} u_{i_j}$ is a translation term corresponding to the center of mass of the triangle $\phi(t_i)$.



(a) A triangle on the surface (t_i).

(b) Mapping of the triangle t_i to $\mathbb{R}^2$. The mapped triangle R^i is isometric to t_i and it is mean centered.

Figure IV-C.1.2: Mapping of a triangle on M to $\mathbb{R}^2$ by projecting it onto the tangent plane.

Since $\mathcal{R}_i$ is isometric to t_i, the matrix A^i contains all the information about the distortion of the triangle t_i under the mapping ϕ_i. Recalling that $\phi(t_i) = \psi_i(R^i)$, we solve eq. (IV-C.1.3) for A^i:

$$A^i = [u_{i_1} - c_i, u_{i_2} - c_i, u_{i_3} - c_i]Q^i, \tag{IV-C.1.4}$$

where Q^i is the right pseudoinverse of R^i (i.e., $R^iQ^i = I_2$). The preservation of the connectivity of M under ϕ is implied in eq. (IV-C.1.4). Specifically, the set of matrices $\mathbf{A} = \{A^1, A^2, \cdots, A^m\}$ are correlated in the sense that if t_i and t_j share an edge (x_k, x_l), the matrices A^i and A^j share the terms u_k, u_l.

IV-C.1.3.3 Weighted penalty function $F(U)$

Calculating A^i in terms of the parameterization coordinates U allows to evaluate the local authalic and conformal distortion for each triangle under the parameterization ϕ. A transformation is authalic if and only if its Jacobian determinant is ± 1. A consistent orientation is important to avoid local overlaps (triangle flips) which may result in a non-bijective mapping, therefore the minus sign is discarded. Thus we measure the area distortion D^i_{area} on each triangle by setting:

$$D^i_{area} = \big(\det(A^i) - 1\big)^2 \tag{IV-C.1.5}$$

On the other hand, a mapping is conformal if its Jacobian matrix is k times a rotation matrix. Similar to [5], we construct the angle distortion measure D^i_{angle} of each triangle as follows:

$$D^i_{angle} = \big(A^i_{11} - A^i_{22}\big)^2 + \big(A^i_{12} + A^i_{21}\big)^2 \tag{IV-C.1.6}$$

If we define the area and shape distortion of the mapping as the sum of all D^i_{area} and all D^i_{angle} respectively, we can measure the global distortion as a linear combination of the global area and shape distortion:

$$F = \sum_{i=1}^{m} \alpha D^i_{area} + (1 - \alpha) D^i_{angle}, \tag{IV-C.1.7}$$

with $0 \leq \alpha \leq 1$ being a weighting parameter such that F measures only angle distortion if $\alpha \to 0$ and F measures only area distortion if $\alpha \to 1$. F is only dependant of $\mathbf{Q} = \{Q^1, Q^2, \cdots, Q^m\}$ and $U = \{u_1, u_2, \cdots, u_n\}$ which are required to compute A^i as described in eq. (IV-C.1.4) and the weighting parameter α, which is introduced in order to allow some control over the resulting parameterization. Our test runs have shown that tuning the α parameter allows our method to find valid parameterizations for some complex datasets which would fail if such parameter is discarded (e.g., if an isometric mapping of M does not exists but a conformal or authalic does).

IV-C.1.3.4 Parameterization $U = \phi(X)$

In order to find the global parameterization $U = \phi(X)$ it is necessary to minimize the F defined in eq. (IV-C.1.7). Therefore our global parameterization is given by the value of U that solves the following unconstrained problem:

$$F^* = \min_U \left\{ \sum_{i=1}^{m} \alpha D^i_{area} + (1-\alpha) D^i_{angle} \right\} \tag{IV-C.1.8}$$

where F^* stands for the minimum of F. The minimization problem posed in (IV-C.1.8) is relatively easy to solve since F is continuous and the region of search is the convex set $\mathbb{R}^{2n}$ (because it is unconstrained). However, the nonlinear nature of the gradient of F requires a nonlinear method for finding a solution to eq. (IV-C.1.8). We choose LM for minimization which is described below.

IV-C.1.3.5 The Levenberg-Marquardt (LM) algorithm

The LM algorithm is a second-order method for optimization of unconstrained continuous twice differentiable functions. It is applied here to compute F^*. In LM the following iterative scheme is posed [18]:

$$U^{k+1} = U^k - \left(\mathcal{H}[F(U^k)] + \lambda^k I_{2n}\right)^{-1} \nabla F(U^k), \tag{IV-C.1.9}$$

where k is the current iteration, λ^k is the LM damping parameter which is updated iteratively according to the current solution, and $\mathcal{H}[F(U^k)]$ is the Hessian matrix of F defined as $\mathcal{H}_{ij}[F] = \frac{\partial^2 F}{\partial u_i \partial u_j}$ [19].

The gradient of F is estimated analytically. It can be computed by adding the gradient associated to the distortion of each triangle:

$$\nabla F = \sum_{i=1}^{m} \alpha \nabla D^i_{area} + (1-\alpha) \nabla D^i_{angle} \tag{IV-C.1.10}$$

Usually, the Hessian matrix in eq. (IV-C.1.9) is approximated as $\mathcal{H}[F] \approx \nabla F \cdot \nabla F^{\mathrm{T}}$. However, such approximation leads to a dense matrix. An alternative is to compute the Hessian analytically as follows:

$$\mathcal{H}[F] = \sum_{i=1}^{m} \alpha \mathcal{H}[D^i_{area}] + (1-\alpha) \mathcal{H}[D^i_{angle}] \tag{IV-C.1.11}$$

In eq. (IV-C.1.11), the Hessian of each triangle is computed individually and added afterwards to $\mathcal{H}[F]$ resulting in a $2n \times 2n$ sparse matrix where only adjacent points in M have respective nonzero

elements. Therefore, the linear system that arises in eq. (IV-C.1.9) can be handled efficiently by a sparse solver.

In addition, the iterative procedure in eq. (IV-C.1.9) requires an initial parameterization U^0. The formula is then applied until certain criteria is met: 1) the norm of the gradient $\|\nabla F\|$ is lower than ε (where $\varepsilon \in \mathbb{R}$ is a fixed tolerance) or 2) a certain number of iterations have occurred. A global minimum cannot be guaranteed in most nonlinear-gradient algorithms because the penalty function is not convex [20] requiring a careful selection of an adequate U^0. However, our implementation has provided consistent results despite a random initial parameterization U^0 has been used in all test cases, which is superior than most nonlinear-gradient algorithms which require an initial valid parameterization to proceed such as [6, 8, 17].

IV-C.1.3.6 Complexity analysis

The time complexity of our algorithm is discussed in this section. In [21], a complexity analysis of the LM algorithm has been presented which shows that LM iterates $\mathcal{O}(\ln \varepsilon^{-1})$ times. In our algorithm, for a mesh with m triangles and n nodes each LM iteration must perform the following operations:

1. Compute $F(U^k)$, $\nabla F(U^k)$ and $\mathcal{H}[F(U^k)]$.
2. Solve the linear system $\mathcal{H}[F(U^k) + \lambda^k I_{2n}] = \nabla F(U^k)$ as per eq. (IV-C.1.9).

In the first step of the LM iteration, $F(U^k)$, $\nabla F(U^k)$ and $\mathcal{H}[F(U^k)]$ are computed by adding the distortion D^i_{area} and D^i_{angle} and their corresponding derivatives for each individual triangle. The cost of this operation is $\mathcal{O}(m)$. For the linear system in the second step, the matrix $\mathcal{H}[F(U^k)]+\lambda^k I_{2n}$ is a $2n \times 2n$ symmetric sparse matrix whose nonzero elements correspond to adjacent nodes in the mesh as discussed in section IV-C.1.3.5. The solution of this linear system costs $\mathcal{O}(nz)$ (where nz is the number of nonzeros in the matrix $\mathcal{H}[F(U^k)] + \lambda^k I_{2n}$). Due to the Euler characteristic of triangular meshes, $nz \approx 28n$. Hence, the complexity order for the linear system solution becomes $\mathcal{O}(n)$ [22].

Putting it all together, the order of our algorithm is the order of the LM algorithm times the internal loop i.e., $\mathcal{O}(\ln \varepsilon^{-1})(\mathcal{O}(m) + \mathcal{O}(n))$. However, for a fixed ε and assuming (by the Euler characteristic) that $\mathcal{O}(m) = \mathcal{O}(n)$, our algorithm becomes of the order $\mathcal{O}(n)$. Table IV-C.1.2 presents a comparison of computational complexities for several Mesh Parameterization algorithms.

IV-C.1.3.7 Sensitivity analysis

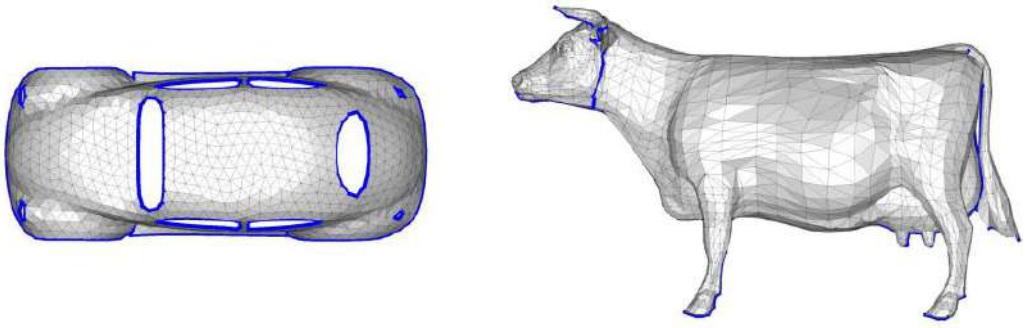
We also perform a sensitivity analysis of the penalty function with respect to the parameter α to compare the resulting paramaterization and the global distortion given the chosen value for α. We therefore estimate numerically the relative sensitivity of F with respect to α as [20]:

$$S^F_\alpha = \frac{\partial \ln F^*}{\partial \ln \alpha} = \frac{\alpha}{F^*}\frac{\partial F^*}{\partial \alpha} \approx \frac{\overline{\alpha}}{\overline{F}^*}\frac{\Delta F^*}{\Delta \alpha} \tag{IV-C.1.12}$$

Eq. (IV-C.1.12) provides an idea of how relatively small changes in the α parameter impact the penalty function F for a given mesh M.

Table IV-C.1.2: Computational time complexity of several Mesh Parameterization algorithms. The complexity analysis of such algorithms considers: a) a fixed ε in all cases [23], and b) solving a sparse linear system for a mesh operator is $\mathcal{O}(n)$ [22]

Algorithm name	**Time complexity**	**Reference**
Intrinsic Parameterizations	$\mathcal{O}(n)$	[1]
Stretch Minimizing Mesh Parameterization	$\mathcal{O}(n)$	[3]
Least Squares Conformal Maps (LSCM)	$\mathcal{O}(n)$	[5]
Linear ABF	$\mathcal{O}(n)$	[14]
As Rigid As Possible (ARAP)	$\mathcal{O}(f(\varepsilon) \cdot n) = \mathcal{O}(n)$	[6]
Boundary Free Composite Approach	$\mathcal{O}(n)$	[7]
Conformal Flattening by Curvature Prescription	$\mathcal{O}(f(\varepsilon) \cdot n) = \mathcal{O}(n)$	[10]
Angle-Based Flattening (ABF)	$\mathcal{O}(n)$	[12]
Bijective Parameterizations with Free Boundaries	$\mathcal{O}(b^2)$ (b: Number of boundary edges)	[17]
Levenberg-Marquard Parameterization	$\mathcal{O}(f(\varepsilon) \cdot n) = \mathcal{O}(n)$	This manuscript

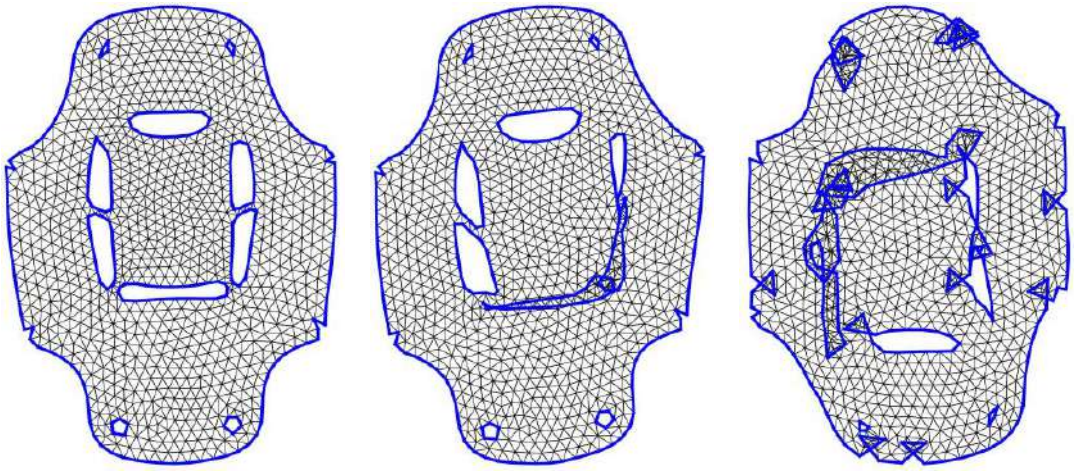


(a) *Beetle* dataset. (b) *Cow* dataset.

Figure IV-C.1.3: Case studies datasets.

IV-C.1.3.8 Datasets

In order to test the proposed algorithm, several triangular meshes have been obtained from public sites in internet. These datasets impose a standard benchmark for Mesh Parameterization and are available from [24]. Due to the high non-developability of some datasets, these have been segmented manually (*Partial-Glove* dataset) while others have an artificial boundary (*Cow*, *Fandisk*, *Foot* and *Bull* datasets) computed by the Seamster algorithm [25] prior to Mesh Parameterization, allowing the algorithm to unfold the surface in the plane.



(a) $\alpha = 0.1$. (b) $\alpha = 0.5$. (c) $\alpha = 0.9$

Figure IV-C.1.4: Resulting parameterization for the *Beetle* dataset for different α values: (a) $\alpha = 0.1$ (quasi-conformal), (b) $\alpha = 0.5$ (conformal and authalic) and (c) $\alpha = 0.9$ (quasi-authalic).

IV-C.1.4 Results and discussion

In this section, two case studies from the literature are presented and analyzed thoroughly. Section IV-C.1.4.1 presents the first case corresponding to the *Beetle* dataset (fig. IV-C.1.3(a)). We show that by tuning adequately the α parameter, a valid parameterization can be achieved. Section IV-C.1.4.2 presents the second case study namely the *Cow* dataset (fig. IV-C.1.3(b)). This case study has presented several problems and though a nearly-valid parameterization is achieved with our algorithm, global overlaps cannot be helped. Finally, section IV-C.1.4.3 presents and discusses a summary of the results of our parameterization algorithm applied to other datasets.

IV-C.1.4.1 *Beetle* dataset results

Fig. IV-C.1.4 presents the resulting parameterization U for the *Beetle* dataset with different values of α. Setting $\alpha = 0.1$ results in a valid parameterization with low shape distortion as seen in fig. IV-C.1.4(a). This is not the case for $\alpha = 0.5$ (fig. IV-C.1.4(b)), where global overlaps occur as some area preservation is demanded to the algorithm (triangle flips do not happen). A highly authalic mapping ($\alpha = 0.9$) results in a parameterization with higher shape distortion and low area distortion (fig. IV-C.1.4(c)). Despite no triangle flips occur, the boundary and non-adjacent triangles overlap resulting in a non-bijective parameterization. The mapped texture in fig. IV-C.1.5 shows how the shape is highly preserved as squares attain its form through the bijective mapping for $\alpha = 0.1$. Similar results for this dataset have been presented by other authors [6, 15].

Recalling that the implemented algorithm converges to the same solution despite the initial (possibly non-valid) parameterization, fig. IV-C.1.6 presents the initial, intermediate and final stages for $\alpha = 0.1$ of the LM for different initial parameterizations: i) an initial parameterization U^0_{Isomap} computed by the Isomap algorithm [26] (fig. IV-C.1.6(a)), ii) an initial parameterization

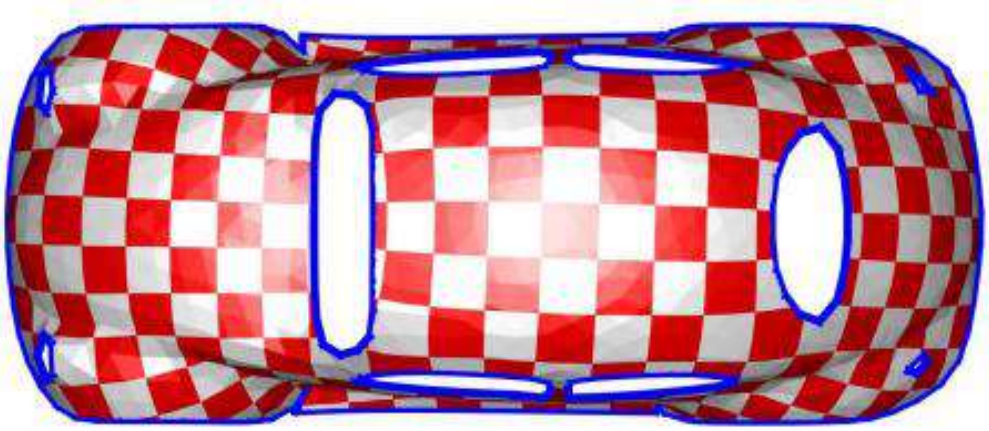


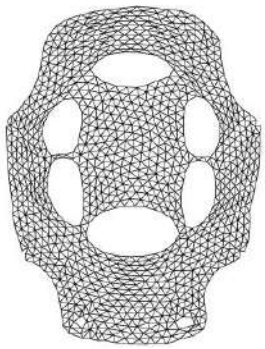
Figure IV-C.1.5: Texture map for the *Beetle* dataset ($\alpha = 0$).

U^0_{LapEig} computed by the Laplacian Eigenmaps algorithm [27] (fig. IV-C.1.6(b)) and iii) a randomly generated (non-bijective) initial parameterization U^0_{Rand} (fig. IV-C.1.6(c)). The respective intermediate steps (figs. IV-C.1.6(d), IV-C.1.6(e) and IV-C.1.6(f)) show how the surface is unfolded in each case and fig. IV-C.1.6(g) presents the resulting parameterization for all the cases illustrating the consistency of the algorithm (even for the random non-valid initial parameterization U_{Rand}) as discussed in section IV-C.1.3.5.

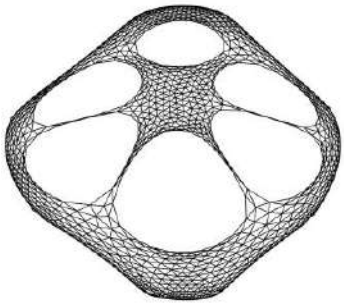
Using the random initial solution of fig. IV-C.1.6(c), fig. IV-C.1.7(a) presents the evolution of the penalty function F for different α values. Higher values of α take more iterations before converging. Also, though F^* reaches a lower value for $\alpha = 1$ than for $\alpha = 0.5$, this is not guarantee of a better result as seen in figs. IV-C.1.4(b) and IV-C.1.4(c). Fig. IV-C.1.7(b) plots the relative sensitivity of F^* with respect to α as per eq. (IV-C.1.12). F^* becomes highly sensitive to α for values greater than 0.6 making F more stable for lower values of α. Despite this stability, the resulting parameterization may vary and become a non-valid result as seen in figs. IV-C.1.4(a) and IV-C.1.4(b).

IV-C.1.4.2 *Cow* dataset results

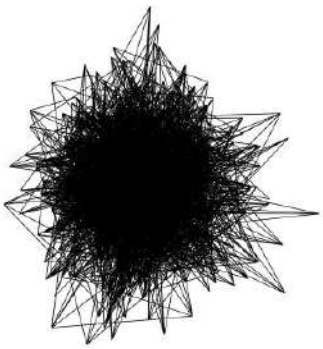
For a random initial parameterization U^0_{Rand}, fig. IV-C.1.8 presents the resulting parameterization U for the *Cow* dataset with different α values. Setting $\alpha = 0.1$ results in a non-valid parameterization where the head of the *Cow* overlaps its body (fig. IV-C.1.8(a)). For $\alpha = 0.01$, the head no longer overlaps the body in the resulting parameterization (fig. IV-C.1.8(b)). However, the resulting parameterization is still non-bijective. Seeking a purely conformal parameterization ($\alpha = 0$) results in a high distorted mapping where the head and the legs present a high area distortion (figs. IV-C.1.8(c) and IV-C.1.9). Zooming into the head and tail of the *Cow* it is clear that the boundary self-intersects and the parameterization is not bijective (figs. IV-C.1.8(d) and IV-C.1.8(e)). It is important to emphasize that none of the discussed results above present triangle flips despite the map not being bijective. Our results are in concordance with other authors [6, 10, 13, 14] where non-bijective parameterizations (e.g., global overlaps) have been also reported for the *Cow* dataset. Finally, higher values of α were tested resulting in worse parameterizations.



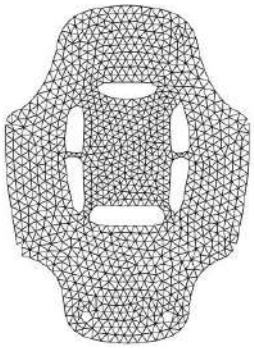
(a) Isomap initial parameterization U^0_{Isomap}.



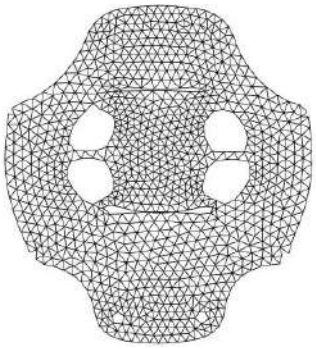
(b) Laplacian Eigenmaps initial parameterization U^0_{LapEig}.



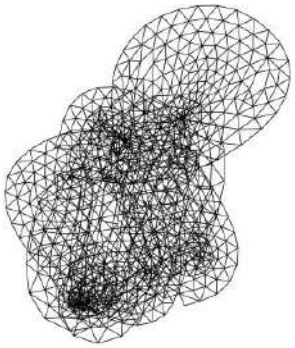
(c) Random (non-bijective) initial parameterization U^0_{Rand}.



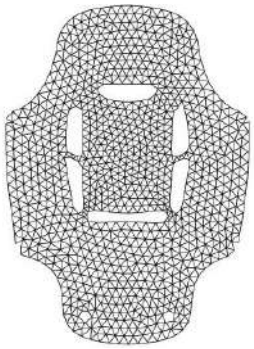
(d) Intermediate solution for U^0_{Isomap}.



(e) Intermediate solution for U^0_{LapEig}.

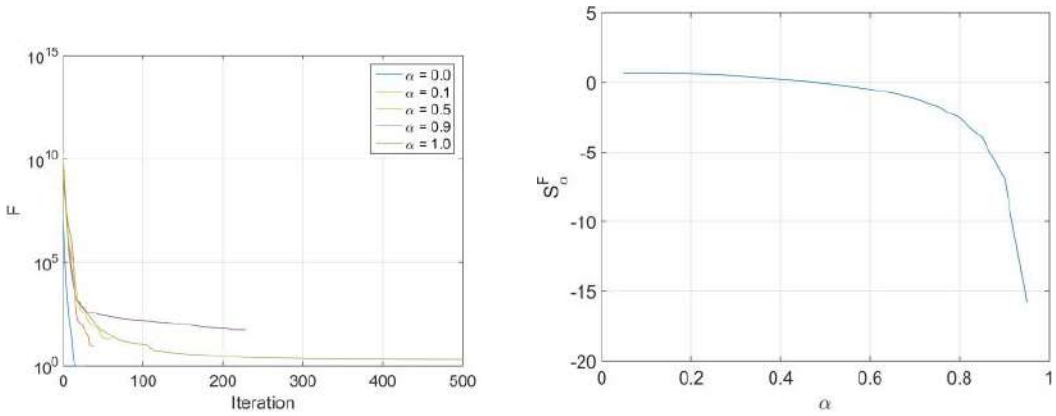


(f) Intermediate solution for U^0_{Rand}.



(g) Final LM parameterization for: i) U^0_{Isomap}, ii) U^0_{LapEig} and iii) U^0_{Rand}.

Figure IV-C.1.6: Initial, intermediate and final stages of LM optimization for the *Beetle* dataset using different initial parameterizations: i) Isomap U^0_{Isomap}, ii) Laplacian Eigenmaps U^0_{LapEig} and iii) random parameterization U^0_{Rand}. α is set to 0.1.



(a) Evolution of F for different α values.

(b) Relative sensitivity S^F_α.

Figure IV-C.1.7: *Beetle* dataset. Sensitivity analysis of F with respect to α.

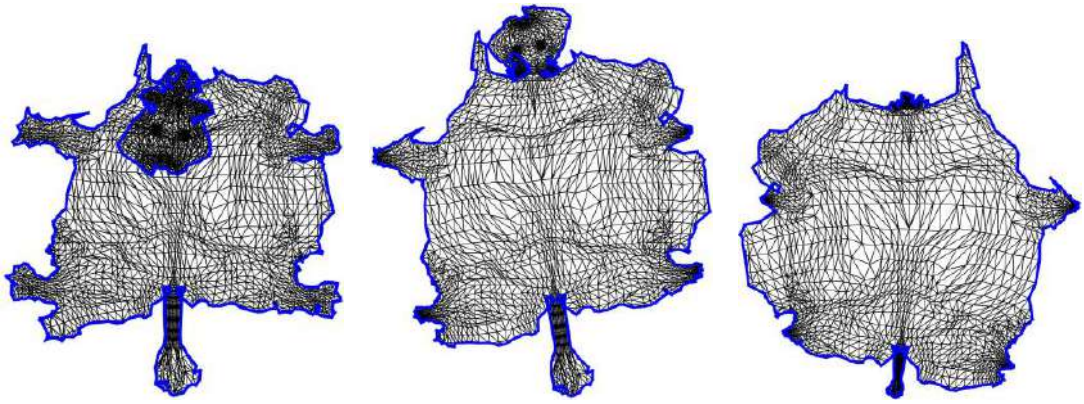
Table IV-C.1.3: Appraisal of the algorithm results for the test datasets.

Dataset	**n**	$\boldsymbol{\alpha}$	$\|\nabla \mathbf{F}\|$	**Result**
Beetle	988	0.1	2×10^{-12}	Fig. IV-C.1.5
Cow	3195	0	4×10^{-11}	Fig. IV-C.1.9
Sliced-Glove	985	0.5	7×10^{-11}	Fig. IV-C.1.11(a)
Fandisk	6699	0.5	4×10^{-11}	Fig. IV-C.1.11(b)
Foot	10211	0.5	6×10^{-11}	Fig. IV-C.1.11(c)
Bull	17918	0.1	4×10^{-9}	Fig. IV-C.1.11(d)

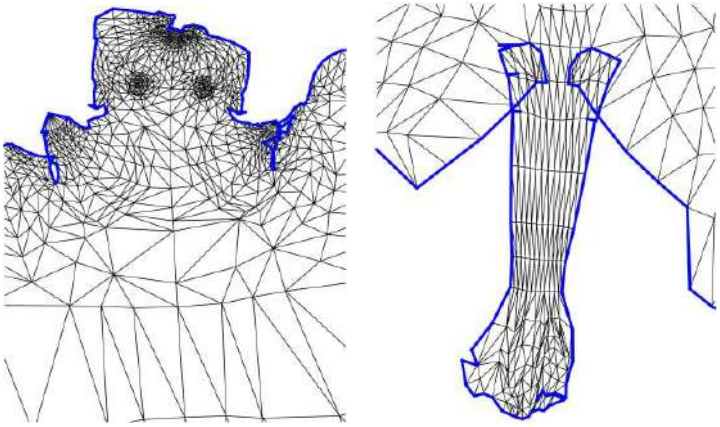
Fig. IV-C.1.10 presents a sensitivity analysis of F with respect to α for the *Cow* dataset. Again, higher values of α makes the algorithm to require more iterations to converge (fig. IV-C.1.10(a)). The relative sensitivity of F with respect to α (fig. IV-C.1.10(b)) shows how F becomes highly sensitive to α for values higher than 0.6. However, this analysis must always be complemented by the resulting parameterization (fig. IV-C.1.8).

IV-C.1.4.3 Results for other datasets

Fig. IV-C.1.11 presents the parameterization results of the proposed algorithm for the *Sliced-Glove*, *Fandisk*, *Foot* and *Bull* datasets. With an $\alpha = 0.5$ the algorithm converged in all the presented cases to valid parameterizations (except for the *Bull* parameterization which is not bijective) which competes against most Mesh Parameterization algorithms which have presented similar results qualitatively for the same datasets [6, 7, 10, 13, 14] as well as similar (asymptotic) time performance as illustrated in table IV-C.1.2. Table IV-C.1.3 presents an appraisal of the algorithm applied to the test datasets. A random initial parameterization U^0_{Rand} is used and in all the test cases, $\|\nabla F^*\| < 10^{-8}$ and the lowest eigenvalue of the Hessian is nonnegative indicating that a local minimum has been reached and the algorithm has converged. All the resulting parameterizations are locally bijective (no triangle flips).



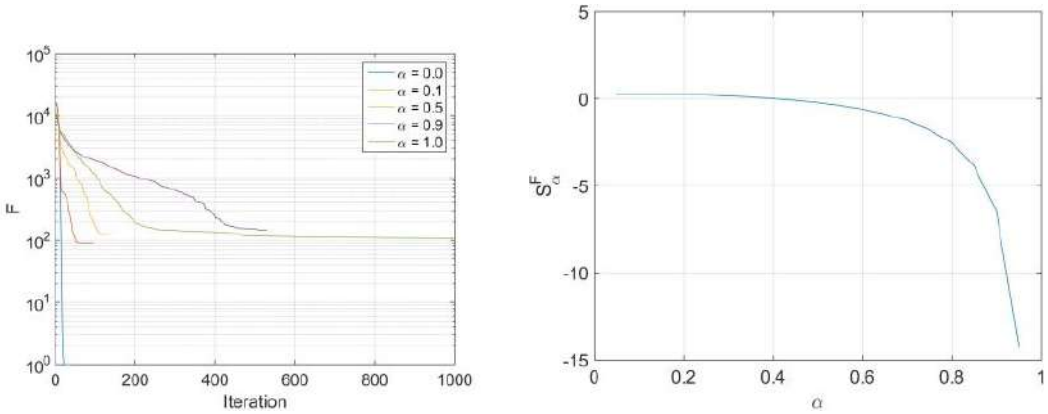
(a) *Cow* parameterization ($\alpha = 0.1$). (b) *Cow* parameterization ($\alpha = 0.01$). (c) *Cow* parameterization ($\alpha = 0$).



(d) *Cow* parameterization ($\alpha = 0$). Zoom into head. (e) *Cow* parameterization ($\alpha = 0$). Zoom into tail.

Figure IV-C.1.8: Parameterization results for the *Cow* dataset.

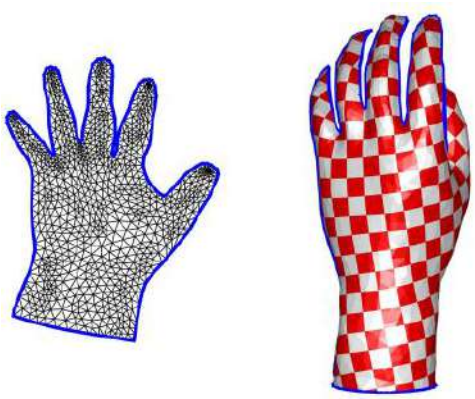
Figure IV-C.1.9: Texture map for the *Cow* dataset ($\alpha = 0$). Area distortion is more noticeable in the head, legs and tail.



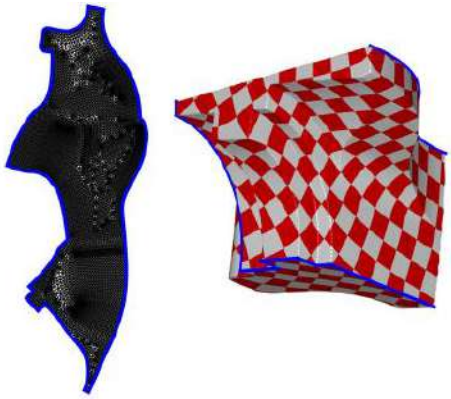
(a) Evolution of F for different α values.

(b) Relative sensitivity S^F_α.

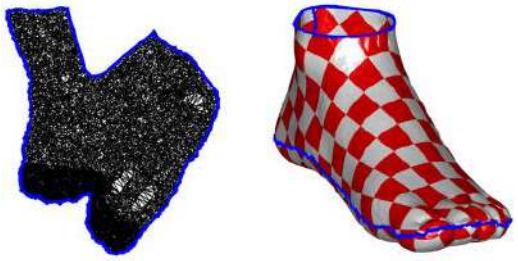
Figure IV-C.1.10: *Cow* dataset. Sensitivity analysis of F with respect to α.



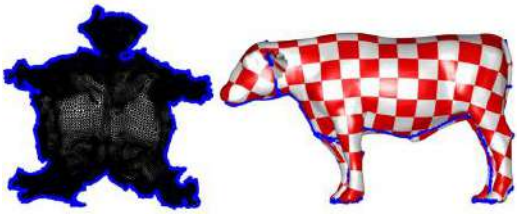
(a) *Sliced-Glove* bijective parameterization and texture map.



(b) *Fandisk* bijective parameterization and texture map.



(c) *Foot* bijective parameterization and texture map.



(d) *Bull* non-bijective parameterization and texture map.

Figure IV-C.1.11: Parameterization results for several datasets and their corresponding texture map. All meshes are 2-manifolds with border.

IV-C.1.5 Conclusions

This article presents an algorithm for parameterizing a triangular mesh M of an open 2-manifold embedded in $\mathbb{R}^3$. The proposed algorithm consists of mapping each triangle individually to the XY plane by a rigid transformation $\mathcal{R}$ and then mapping it to the global parameterization ϕ by an affine mapping ψ. The parameterization $U = \phi(X)$ is obtained by minimizing the weighted area and angle distortion of ψ with the LM algorithm. The complexity analysis of our algorithm showed asymptotic linear behavior $\mathcal{O}(n)$ which makes our method comparable to most mesh parameterization that are also asymptotically linear in time as illustrated in table IV-C.1.2. Our algorithm presents the advantage over other nonlinear-gradient parameterization algorithms of not requiring an initial valid parameterization.

A weighting parameter α is introduced in the penalty distortion function which allows tuning by the user to favour area against angle preservation turning a non-bijective parameterization into a bijective one in specific cases. Our sensitivity analysis shows that higher values for α make the penalty function more unstable and the algorithm slower. However, the sensitivity analysis does not evidence the quality of the parameterization and a small change in α may turn a bijective mapping into a non-bijective or vice-versa.

In general, the proposed algorithm converged presenting correct results across the datasets, rendering low distortion, non-overlapping and valid parameterizations except for highly non-developable datasets (i.e., *Cow* and *Bull* datasets).

IV-C.1.5.1 Ongoing work

Segmentation of large meshes into smaller ones increases the probability of finding bijective individual parameterizations for the smaller ones. Therefore, it is of interest to explore mesh segmentation as a necessary step for mesh parameterization.

Also, bijectivity of the resulting parameterization can be broken by two facts: 1) local overlaps (triangle flips) and 2) global overlaps. The first one has been already addressed in this article. However, global bijectivity is a non-trivial constraint which increases the computational complexity of the algorithm to $\mathcal{O}(n^2)$ as shown in [17]. Therefore, further work is required on this aspect.

IV-C.2

Quasi-Isometric Mesh Parameterization Using Heat-Based Geodesics and Poisson Surface Fills

Daniel Mejia-Parra[1,2], Jairo R. Sánchez[2], Jorge Posada[2], Oscar Ruiz-Salguero[1], Carlos Cadavid[3]

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, K49 #7 Sur-50, Medellín (Colombia)

[2] Vicomtech, Mikeletegi Pasealekua, 57, Donostia - San Sebastián (Spain)

[3] Matemáticas y Aplicaciones, Departamento de Ciencias Matemáticas, Universidad EAFIT, K49 #7 Sur-50, Medellín (Colombia)

an Open Access Journal by MDPI

Citation

Daniel Mejia-Parra, Jairo R. Sánchez, Jorge Posada, Oscar Ruiz-Salguero and Carlos Cadavid. Quasi-isometric mesh parameterization using heat-based geodesics and Poisson surface fills. *Mathematics*, **2019**, 7(8), 753. DOI: 10.3390/math7080753.

Indexing: ISI (Q1), SCOPUS (Q2), Publindex (A1)

Abstract

In the context of CAD, CAM, CAE, and reverse engineering, the problem of mesh parameterization is a central process. Mesh parameterization implies the computation of a bijective map ϕ from the original mesh $M \in \mathbb{R}^3$ to the planar domain $\phi(M) \in \mathbb{R}^2$. The mapping may preserve angles, areas, or distances. Distance-preserving parameterizations (i.e., isometries) are obviously attractive. However, geodesic-based isometries present limitations when the mesh has concave or disconnected boundary (i.e., holes). Recent advances in computing geodesic maps using the heat equation in 2-manifolds motivate us to revisit mesh parameterization with geodesic maps. We devise a Poisson surface underlying, extending, and filling the holes of the mesh M. We compute a near-isometric mapping for quasi-developable meshes by using geodesic maps based on heat propagation. Our method: (1) Precomputes a set of temperature maps (heat kernels) on the mesh; (2) estimates the geodesic distances along the piecewise linear surface by using the temperature maps; and (3) uses multidimensional scaling (MDS) to acquire the 2D coordinates that minimize the difference between geodesic distances on M and Euclidean distances on $\mathbb{R}^2$. This novel heat-geodesic parameterization is successfully tested with several concave and/or punctured surfaces, obtaining bijective low-distortion parameterizations. Failures are registered in nonsegmented, highly nondevelopable meshes (such as seam meshes). These cases are the goal of future endeavors.

Keywords: Mesh Parameterization, Geodesic Maps, Heat Transfer Analysis, Poisson Fills.

Glossary

MDS:	Multidimensional scaling.
M:	Triangular mesh $M = (X, \mathcal{T})$ of a connected 2-manifold with border (and possibly holes), embedded in $\mathbb{R}^3$. M is represented as a set of points $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^3$ and a set of oriented triangles $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$.
M^*:	Triangular mesh $M^* \subset \mathbb{R}^3$ of a connected 2-manifold with border (but without holes). M^* is an extension of M ($M \subset M^*$).
$\Delta t, T$:	Time step size Δt and total simulation time T parameters defined for the heat transfer simulations on M.
ϕ:	Continuous and bijective function (homeomorphism) $\phi : M \to \mathbb{R}^2$ that maps M to a planar region in $\mathbb{R}^2$. In this manuscript, ϕ is a nearly-isometric map (i.e., highly preserves distances).
g:	Geodesic distance function $g : M \times M \to \mathbb{R}^+$ defined on M. $g_{ij} = g(x_i, x_j)$.
δ_{x_i}	Dirac delta (temperature) distribution $\delta_{x_i} : M \to [0, \infty]$ associated to the source point x_i, such that the temperature at x_i is infinite and 0 everywhere else.
u_i:	Heat kernel function $u_i : M \times (0, T] \to \mathbb{R}$ associated to vertex $x_i \in M$. $u_i(x, t)$ is the temperature at the point x, t, due to an initial infinite-heat point-source δ_{x_i}.
χ:	Continuous function $\chi : \mathbb{R}^3 \to$ that indicates if a point $p \in \mathbb{R}^3$ is "inside" ($\chi(p) = 1$), "outside" ($\chi(p) = 0$), or "in-between" ($\chi(p) = \frac{1}{n}\sum_{x \in M} \chi(x)$) a solid defined by M^*.
$\vec{H}_i$:	Vector field $\vec{H}_i : M \to \mathbb{R}^3$ defined on M. $\vec{H}_i(x, t) = -\frac{\nabla u_i(x,t)}{\|\nabla u_i(x,t)\|}$ is the normalized heat flux for the heat kernel u_i. $\|\vec{H}_i(x, t)\| = 1$.

$\vec{N}$: Vector field $\vec{N} : \mathbb{R}^3 \to \mathbb{R}^3$ defined on $\mathbb{R}^3$. For every $x \in M$, $\vec{N}(x)$ is the normal vector to the surface M at x. For any $x \notin M$, $\vec{N}(x) = 0$.

Φ: Discrete parameterization $\Phi = \{\phi_1, \phi_2, \ldots \phi_n\} \subset \mathbb{R}^2$ of M such that $\phi_i = \phi(x_i)$.

L, B: $n \times n$ Laplace–Beltrami L and mass B matrices, respectively. L and B are the sparse and symmetric matrices that approximate the Laplace–Beltrami operator on the triangular mesh M.

$U^{(t)}$: $n \times n$ matrix with the discrete heat kernels maps associated to each vertex in M, i.e., $U_{ij}^{(t)} = u_i(x_j, t)$.

$K^{(t)}$: $n \times n$ matrix of the divergence fields of all heat kernels $U^{(t)}$. $K_{ij}^{(t)} = \nabla \cdot \vec{H}_i(x_j, t)$.

G, G^2: $n \times n$ matrices of geodesic and squared geodesic distances, respectively, defined on M. $G_{ij} = g(x_i, x_j)$, $G_{ij}^2 = g(x_i, x_j)^2$.

C: $n \times n$ symmetric matrix whose entries contain the mean centered squared geodesic distances in M, i.e., $C_{ij} = -\frac{1}{2}[g_{ij}^2 - \frac{1}{n}(\sum_{kl} g_{kl}^2)]$. C is semidefinite positive.

I_n, J_n: $n \times n$ identity I_n and all-ones J_n matrices.

λ_1, V_1: Largest (positive) eigenvalue λ_1 of the semidefinite positive matrix C and its corresponding $n \times 1$ eigenvector V_1. $\sqrt{\lambda_1}V_1$ are the discrete u-coordinates of the parameterization $\Phi \subset \mathbb{R}^2$.

λ_2, V_2: Second largest (positive) eigenvalue λ_2 of the semidefinite positive matrix C and its corresponding $n \times 1$ eigenvector V_2. $\sqrt{\lambda_2}V_2$ are the discrete v-coordinates of the parameterization $\Phi \subset \mathbb{R}^2$.

IV-C.2.1 Introduction

Mesh parameterization is the process by which a piecewise linear surface (i.e. triangular mesh) M is mapped with the least possible distortion onto a planar ($\mathbb{R}^2$) region, via a bijective continous function $\phi : M \to \mathbb{R}^2$. The mesh M is supposed to be a connected 2-manifold with border (and possibly holes).

Mesh parameterization is central in tool path generation for surface machining, texture mapping, thermo-forming of thin layers (metal, leather, plastic, etc.), reverse engineering, finite element remeshing, facial expressions, morphing, etc.

A geodesic curve between two points of a continuous surface is the shortest path *within the surface* joining the two points. The length of such a path is the geodesic (shortest) distance, embedded in the surface, between those two points.

Given any two points $x_i, x_j \in M$, ideal parameterizations of such a surface seek to map them to $\phi(x_i), \phi(x_j) \in \mathbb{R}^2$ so that the geodesic distance between x_i and x_j in M matches the Euclidean 2D distance between $\phi(x_i)$ and $\phi(x_j)$ in $\mathbb{R}^2$. In the rare occasions in which this goal is possible, M is called a developable surface and ϕ is an isometric map. When the distortion in such distances is small, one qualifies M as a quasi-developable surface. This case is sufficiently frequent, since large triangular meshes can be segmented with a goal being that the resulting sub-meshes are quasi-developable or developable.

It is not convenient, when the mesh has holes or concavities in its boundary, to parameterize the mesh via geodesics. The reason is that mesh points being close neighborhs in the surface may be far apart via geodesic curves due to mesh gaps or concavities.

IV-C.2.2 Literature Review

Mesh parameterization algorithms can be classified depending on the type of distortion being minimized, as follows: (a) Area-preserving (authalic) algorithms; (b) angle-preserving (conformal) algorithms; and (c) distance-preserving (isometric) algorithms. The remainder of this section discusses a summary of recent mesh parameterization algorithms already present in the literature (Detailed surveys on mesh parameterization algorithms are presented in [28–30]).

IV-C.2.2.1 Area-Preserving Mesh Parameterization

Area-preserving (authalic) parameterization algorithms rely on the minimization of an area preserving continuous cost function. Zou et al. [31] solved a Lie advection problem on the mesh M. The gradient of the scalar Lie advection field was then added to an initial parameterization ϕ_0 of M, resulting in an authalic parameterization. Zhao et al. [32] solved an optimal mass transport problem from the mesh M to its parameterization $\phi(M)$. The optimal mass transport poses a partial differential equation in which the parameterization $\phi(x)$ locally preserves the area at every point $x \in M$. Since most optimal transport methods only parameterize meshes with a connected boundary (i.e., without holes), Su et al. [33] introduced additional boundary conditions to allow authalic parameterizations of meshes with more complex topologies.

IV-C.2.2.2 Angle-Preserving Mesh Parameterization

Angle-preserving (conformal) optimization aims to minimize the parameterization angle distortion. Since this objective can be achieved by collapsing all triangles to a single point, these algorithms rely on constraining the parameterized boundary to a region in $\mathbb{R}^2$. Disk geometries are usually used in this context [34, 35] however, other geometries such as squared domains have also been proposed [36, 37]. The imposed boundary restrictions in these constrained optimization algorithms induce additional distortion in the resulting parameterization.

Free boundary algorithms allow unrestricted boundary parameterizations, producing less distorted maps. Sawhney and Crane [38] presented an algorithm in which the mesh boundary is mapped to $\mathbb{R}^2$ according to its shape. The parameterized boundary is then used as a constraint to produce a boundary-free parameterization. Starting from a disk parameterization, Bright et al. [39] performed nonlinear optimization while unconstraining the boundary edges of the parameterized mesh. The resulting parameterization allows the (initially mapped to disk) boundary to move freely in the parameter space. Smith and Schaefer [17] presented a mesh parameterization method which introduced a barrier function in its optimization process. The introduced barrier function penalizes nonadjacent triangle overlaps, which guarantees global bijectiveness in the resulting parameterization.

Dimensionality reduction is a superset of mesh parameterization, in which a d-manifold embedded in a higher dimensional space $\mathbb{R}^D$, is parameterized to its corresponding $\mathbb{R}^d$ domain. As a consequence, these algorithms have been applied successfully in mesh parameterization applications. Such algorithms include Laplacian Eigenmaps [16] and Hessian Locally Linear Embedding [40].

IV-C.2.2.3 Distance-Preserving Mesh Parameterization

By definition, a distance-preserving (isometric) mapping is a function that simultaneously preserves areas and angles. Mejia et al. [41] presented a nonlinear minimization algorithm for area-angle (isometry) preservation. The minimization function is a linear combination of area and angle distortion, and the weighting parameters for each distortion term are adjusted by the user. The authors pointed out that the algorithm performs better when the angle-preserving term is preponderant over the area-preserving one. Similarly, Yu et al. [42] used polar factorization to introduce area-angle preserving mappings, in which area distortion increases as angle distortion decreases. ARAP (As Rigid As Possible) algorithms divide the parameterization into two optimization steps—local parameterization and global parameterization—performing these steps iteratively one after another until convergence [6, 43, 44]. These algorithms produce different bijective parameterizations, but since the weighting parameters are nonoptimized (as they are user-defined), the resulting parameterization is rarely the optimal distance-preserving map.

Ruiz et al. [16] used a dimensionality-reduction geodesic-based algorithm (Isomap) for the computation of isometric parameterization of quasi-developable meshes. However, in addition to the classic nonconvex parameterization problems, such algorithms estimate geodesics using shortest-path graph algorithms which introduce additional distortions in the resulting parameterization. Li et al. [45] presented a geodesic approximation approach in which cutting planes are intersected with the mesh to estimate geodesic paths. This approach solves the problem of nonconvex surfaces and distortion errors induced by mesh graph approximation. However, the method is limited to geodesic curves embedded in $\mathbb{R}^2$ (i.e., the cutting plane).

IV-C.2.2.4 Conclusions of the Literature Review

Most of the distance-preserving parameterization algorithms rely on weighting angle vs. area distortion. Such a weighting is defined by the user and drastically changes the resulting parameterization, not providing the optimal isometric mapping. Geodesic-based parameterization algorithms solve this problem by directly minimizing the distance distortion. However, current geodesic-based algorithms rely on shortest-path graph algorithms for geodesics estimation, introducing unnecessary distortion in the resulting parameterization. In addition, estimation of geodesics fails when the surface is nonconvex (such as surfaces with holes and boundary concavities).

To overcome these problems, this manuscript presents a novel heat-geodesic mesh parameterization algorithm. Our algorithm computes a set of temperature maps (heat kernels) on the mesh M, which are then used to retrieve the set of point-to-point geodesic distances on the discrete mesh. Afterwards, a near-isometric parameterization is obtained by minimizing the difference between the parameterization Euclidean distances and their corresponding geodesics. Since our method relies on finite element mesh discretization to estimate the temperature maps and geodesics, our geodesics estimation is unaffected by mesh-graph topology (as opposed to shortest-path graph algorithms). To overcome the nonconvexity problem, our algorithm uses Poisson surface reconstruction [46], in which the surface holes and boundary concavities are temporarily filled for parameterization. The resulting parameterization for the Poisson reconstructed surface is trimmed with the original boundary of M, producing a trimmed surface. The implementation and integration of these different techniques provide a novel geodesic-based mesh parameterization algorithm which is (1) unaffected by mesh holes and/or concavities and (2) less sensitive to mesh graph topology.

IV-C.2.3 Methodology

Consider $M = (X, \mathcal{T})$ (with point set $X = \{x_1, x_2, \cdots, x_n\}$ and triangle set $\mathcal{T} = \{t_1, t_2, \cdots, t_m\}$), a connected 2-manifold with border (and possibly holes) embedded in $\mathbb{R}^3$. The problem of mesh parameterization consists of finding a set of points $\Phi = \{\phi_1, \phi_2, \cdots, \phi_n\} \subset \mathbb{R}^2$ such that ϕ_i is the image of $x_i \in M$ under the image of a homeomorphism $\phi : M \to \mathbb{R}^2$ (i.e. $\phi_i = \phi(x_i)$). The function ϕ must satisfy the following conditions:

1. **Continuity:** If $t_i \in \mathcal{T}$ and $t_j \in \mathcal{T}$ $(t_i \neq t_j)$ are adjacent triangles in M, then $\phi(t_i)$ and $\phi(t_j)$ are adjacent in $\phi(M)$.
2. **Local bijectiveness:** All mapped triangles $\phi(M)$ share the same orientation in $\mathbb{R}^2$.
3. **Global bijectiveness:** Triangles in $\phi(M)$ do not overlap each other. This can happen even if all triangles share the same orientation as illustrated in [17].

In addition, define $g : M \times M \to \mathbb{R}^+$ as the geodesic distance function in M. If $g(x_i, x_j) = \|\phi_i - \phi_j\|$ $(x_i, x_j \in M)$, then ϕ is an isometric mapping (i.e. ϕ preserves geodesic distances) and M is a developable surface.

As most of the surfaces are not developable in practice, we aim to find the discrete mapping Φ that minimizes the difference between these two distances as follows:

$$\begin{aligned} &\min_{\Phi} \sum_{i=1}^{n} \sum_{j=1}^{n} \left\| \|\phi_i - \phi_j\| - g(x_i, x_j) \right\|^2 \\ &\text{s.t.} \\ &\sum_{i=1}^{n} \phi_i = \mathbf{0} \end{aligned} \tag{IV-C.2.1}$$

where the restriction $\sum_{i=1}^{n} \phi_i = \mathbf{0}$ indicates that the parameterization is mean centered, i.e., the center of mass of the parameterization points is $\mathbf{0} \in \mathbb{R}^2$. Such a restriction is introduced to obviate all the possible translations of the same solution.

IV-C.2.3.1 Algorithm Overview

Our mesh parameterization algorithm aims to retrieve a parameterization $\phi(M)$ which highly preserves the geodesic distances of M as Euclidean distances. In order to estimate the geodesic distances g in M, the heat-based algorithm the heat-based geodesics algorithm presented in [47] is implemented. Afterwards, we use classic Multi-Dimensional Scaling to retrieve the 2D coordinates of the parameterization ϕ from the computed geodesic distances. In the case that M presents holes or concavities, our algorithm applies Poisson surface reconstruction [46] and computes a parameterization on a trimmed surface instead. A summary of the algorithm is presented in Fig. IV-C.2.1.

The remainder of this section details the steps to solve Eq. (IV-C.2.1), and the details of our mesh parameterization algorithm. The algorithm has been implemented in MATLAB®[48], except for the Poisson Surface Fills which have been implemented in C++ using the PCL library [49]. Figures including triangular meshes, scalar fields, vector fields, and 2D parameterizations have been produced in MATLAB®. Figures including texture maps on the surface have been produced using MeshLab[50].

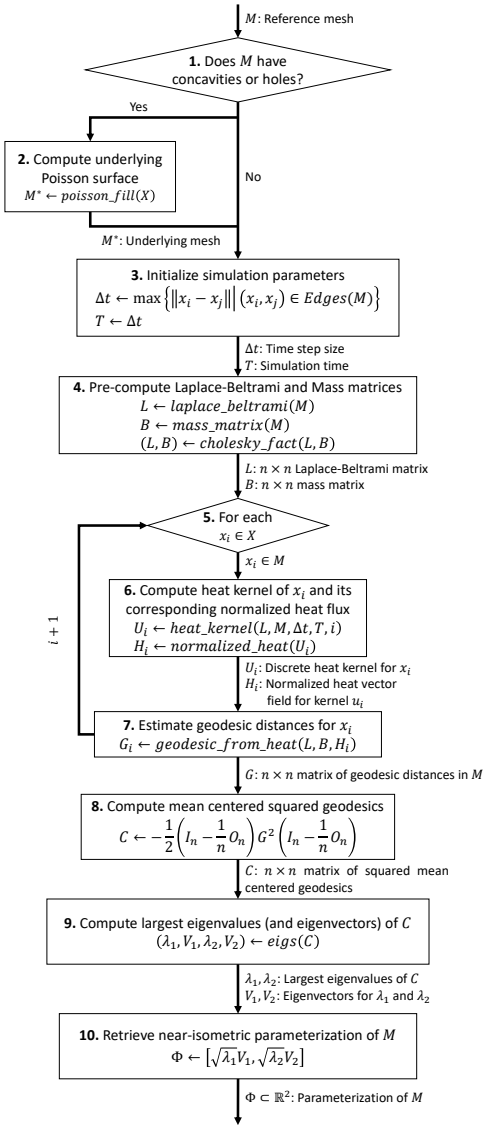


Figure IV-C.2.1: Scheme of our heat-geodesic mesh parametertization algorithm

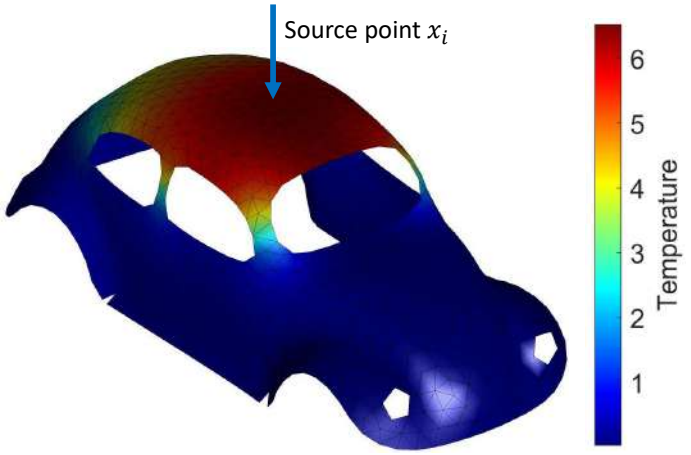


Figure IV-C.2.2: Heat kernel $u_i(x,t)$ for the vertex source x_i $(t > 0)$. Heat dissipates from x_i.

IV-C.2.3.2 Mesh Heat Kernels

A heat kernel of a point $x_i \in M$ is a function $u_i : M \times (0,T] \to \mathbb{R}$ that satisfies the following partial differential equation [51]:

$$\begin{aligned} \frac{\partial u_i(x,t)}{\partial t} + \Delta u_i(x,t) &= 0, \qquad x \in M, t \in (0,T] \\ \left.\frac{\partial u(x,t)}{\partial n}\right|_{\partial M} &= 0 \\ u_i(x,0) &= \delta_{x_i}(x) \end{aligned} \tag{IV-C.2.2}$$

where Δ is the Laplace-Beltrami operator, u_i is the temperature distribution (heat kernel) associated to the source point x_i, $x \in M, t \in (0,T]$ are the spatial and time coordinates , respectively, and $T > 0$ is the simulation time. The term $\left.\frac{\partial u(x,t)}{\partial n}\right|_{\partial M} = 0$ corresponds to the Neumann boundary condition (no boundary heat flux). Finally, the term $u_i(x,0) = \delta_{x_i}(x)$ corresponds to Dirac delta initial conditions, i.e.:

$$\begin{aligned} \delta_{x_i}(x) &= \begin{cases} \infty & \text{if } x = x_i, \\ 0 & \text{otherwise} \end{cases} \\ \int_M \delta_{x_i} &= 1 \end{aligned} \tag{IV-C.2.3}$$

The above initial conditions dictate initial infinite temperature at vertex x_i and 0 everywhere else. After some time $t > 0$ has passed, heat dissipates from x_i as illustrated in Fig. IV-C.2.2.

Eq. (IV-C.2.2) can be solved using a Finite Element discretization scheme, as follows:

$$(\Delta t L + B) U_i^{(t+\Delta t)} = B U_i^{(t)} \tag{IV-C.2.4}$$

where Δt is the time step, $U_i^{(t)} = \{u_{i1}^{(t)}, u_{i2}^{(t)}, \cdots, u_{in}^{(t)}\}$ is the vector of temperatures values ($u_{ij}^{(t)} = u_i(x_j,t)$), and L and B are the $n \times n$ Laplace-Beltrami and mass (sparse and symmetric) matrices,

respectively. For a given edge $e_{ij} = (x_i, x_j)$, the Laplace-Beltrami matrix L is defined as follows:

$$L_{ij} = \begin{cases} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2}, & e_{ij} \in Edges(M) \\ -\sum_{k \in E_i^*} L_{ik}, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{IV-C.2.5}$$

where $\alpha_{ij}, \beta_{ij} \in (0, \pi)$ are the two opposite angles to the edge e_{ij}, and $S_i^* = \{k | e_{ik} \in Edges(M)\}$ is the index set of all incident edges to the vertex $x_i \in X$. The entries L_{ij} of the matrix L are known as cotangent weights [52].

Similarly, the mass matrix B is defined as follows:

$$B_{ij} = \begin{cases} \frac{|t_1| + |t_2|}{12}, & t_1, t_2 \in \mathcal{T} \text{ adjacent triangles to } e_{ij} \in Edges(M) \\ \sum_{k \in S_i^*} B_{ik}, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{IV-C.2.6}$$

where $t_1, t_2 \in \mathcal{T}$ are the pair of triangles adjacent to the edge e_{ij}, and $|t_l|$ is the area of the triangle t_l $(l = 1, 2)$.

For each vertex $x_i \in M$, Eq. (IV-C.2.4) is solved for U_i using a sparse Cholesky linear solver. It is worth noting that for every x_i and $t \in (0, T]$, the matrices $\Delta t L$ and B are the same. As a consequence, these matrices are pre-factored only once using Cholesky factorization, which speeds up the computation of the heat kernels.

Finally, the simulation parameters $\Delta t, T$ are chosen according to [47]:

$$\begin{aligned} \Delta t &= \{ \|x_i - x_j\| \, | \, (x_i, x_j) \in Edges(M) \} \\ T &= \Delta t \end{aligned} \tag{IV-C.2.7}$$

with Δt computed as the magnitude of the largest edge in M, and T equals to Δt. These values have provided better results in our parameterization experiments than other values.

IV-C.2.3.3 Heat-based Geodesic Distance

The vector field $-\nabla u_i(x, t)$ (∇: gradient operator on M) describes the heat flux on M for the respective heat kernel u_i. Define the normalized heat flux vector field H_i as follows:

$$\vec{H}_i(x, t) = -\frac{\nabla u_i(x, t)}{\|\nabla u_i(x, t)\|} \tag{IV-C.2.8}$$

It is worth noting that the magnitude of the vector field $\vec{H}_i$ is 1 everywhere ($\|\vec{H}_i(x, t)\| = 1$). In addition, as illustrated in Fig. IV-C.2.3, the temperature contours are perpendicular to the geodesic paths from $x_i \in M$, and the corresponding vector field points in the same direction that such paths.

The geodesic field $g(x_i, x_j)$ satisfies the following differential equation [47]:

$$\Delta g(x_i, x) = \lim_{T \to 0} \nabla \cdot \vec{H}_i(x, T) \tag{IV-C.2.9}$$

where $\nabla \cdot \vec{H}_i(x, T)$ is the divergence field of the normalized heat flux. Similar to Eq. (IV-C.2.2), Eq. (IV-C.2.9) is discretized using the same Finite Element scheme, as follows:

$$LG_i = \lim_{T \to 0} BK_i^{(T)} \tag{IV-C.2.10}$$

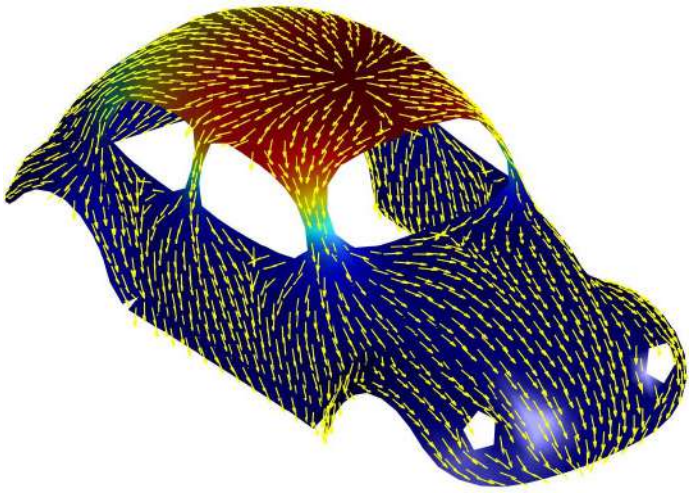


Figure IV-C.2.3: Normalized heat flux field $\vec{H}_i(x,t)$. The vector field is normalized and it points in the direction of the geodesic paths from $x_i \in M$.

where $G_i = \{g_{i1}, g_{i2}, \cdots, g_{in}\}$ is the vector of geodesic distances $g_{ij} = g(x_i, x_j)$, and $K_i^{(T)} = \{k_{i1}^{(T)}, k_{i2}^{(T)}, \cdots, k_{in}^{(T)}\}$ is the divergence field of the normalized gradient $k_{ij}^{(T)} = \nabla \cdot \vec{H}_i(x_j, T)$ [47]. Fig. IV-C.2.4 plots the estimated geodesic distance field $g(x_i, x)$ for the vertex x_i.

IV-C.2.3.4 Multi-Dimensional Scaling (MDS)

After the geodesic field $g_{ij} = g(x_i, x_j)$ has been estimated on M, the minimization problem in Eq. (IV-C.2.1) can be solved. Classic Multi-Dimensional Scaling poses an equivalent minimization problem [53]:

$$\min_{\Phi} \sum_{ij} \left[\frac{1}{n} \left(\sum_{kl} g_{kl}^2 \right) - g_{ij}^2 - 2\phi_i \cdot \phi_j \right]^2 \tag{IV-C.2.11}$$

Let C be the symmetric, semidefinite positive matrix whose entries contain the mean centered squared geodesics (i.e. $C_{ij} = -\frac{1}{2}[g_{ij}^2 - \frac{1}{n}(\sum_{kl} g_{kl}^2)]$). In matrix form, C is equivalent to:

$$C = -\frac{1}{2} \left(I_n - \frac{1}{n} J_n \right) G^2 \left(I_n - \frac{1}{n} J_n \right) \tag{IV-C.2.12}$$

where I_n, J_n are the $n \times n$ identity and all-ones matrices, respectively, and G^2 is the $n \times n$ symmetric matrix whose entries contain the squared geodesic distances in M, i.e., $G_{ij}^2 = g_{ij}^2$. Then, Eq. (IV-C.2.11) becomes:

$$\min_{\Phi} \|C - \Phi\Phi^T\|_F^2 \tag{IV-C.2.13}$$

with $\|A\|_F^2 = \sum_{ij} A_{ij}^2$ the (squared) Frobenius norm of A.

Finally, Eq. (IV-C.2.13) can be solved by an eigendecomposition of C as follows [53]: let λ_1 and λ_2 be the largest positive eigenvalues of C, with respective $n \times 1$ eigenvectors V_1 and V_2. The

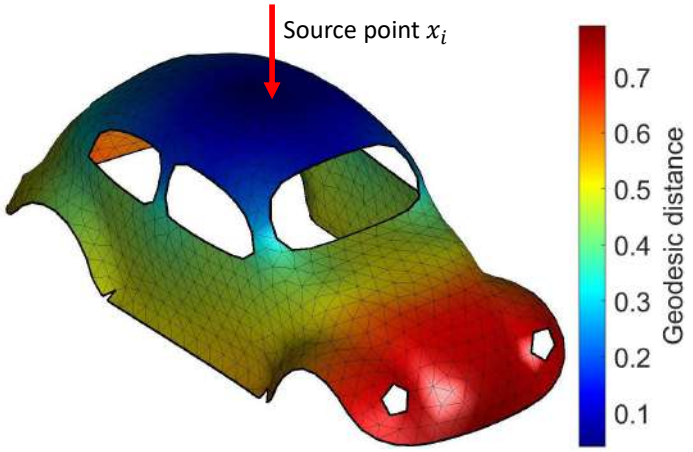


Figure IV-C.2.4: Geodesic field $g(x_i, x)$ for the vertex x_i, computed from its respective heat kernel $u_i(x, T)$

near-isometric parameterization of M becomes:

$$\Phi = [\sqrt{\lambda_1}V_1, \sqrt{\lambda_2}V_2] \tag{IV-C.2.14}$$

where $\sqrt{\lambda_1}V_1$ are the discrete u-coordinates and $\sqrt{\lambda_2}V_2$ are the discrete v-coordinates of the parameterization $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\} \subset \mathbb{R}^2$. Fig. IV-C.2.5 plots the resulting parameterization using MDS on the estimated geodesic fields G.

IV-C.2.3.5 Poisson Mesh Reconstruction

In the case that M is non-convex (i.e. it has holes or concavities), we seek to compute an underlying extending surface M^*. Such a surface contains the points in M, and fills the holes and concavities by extending M in such areas ($M \subset M^*$). As an example, a geodesic path in a non-convex M, circles a given hole (Fig. IV-C.2.6(a)). On the other hand, the same geodesic path in the extended surface M^* crosses through the hole (Fig. IV-C.2.6(b)).

To compute the surface M^*, our parameterization algorithm uses Poisson surface reconstruction [46] from the PCL library [49]. Define $\chi : \mathbb{R}^3 \to \mathbb{R}$ as an indicator function such that $\chi(x) = 1$ if $x \in \mathbb{R}^3$ is "inside" the solid defined by M^* and $\chi(x) = 0$ if x is "outside" such solid. The surface M^* is composed by the points in-between, as follows [46]:

$$M^* = \left\{ p \in \mathbb{R}^3 \middle| \, \chi(p) = \frac{1}{n} \sum_{x \in M} \chi(x) \right\} \tag{IV-C.2.15}$$

The indicator function χ is computed by solving the following partial differential equation in $\mathbb{R}^3$ [46]:

$$\Delta\chi(x) = \nabla \cdot \vec{N}(x) \tag{IV-C.2.16}$$

where Δ and ∇ are the $\mathbb{R}^3$ Euclidean Laplacian and gradient operators, respectively. It is worth noting that this Laplacian and gradient operators are different from the 2-manifold version presented

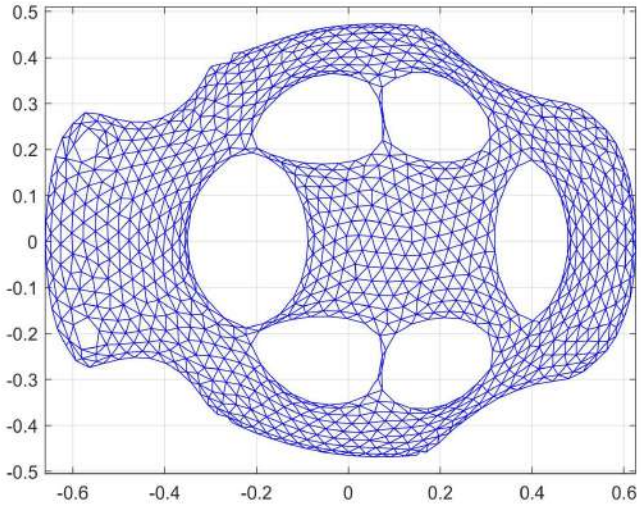
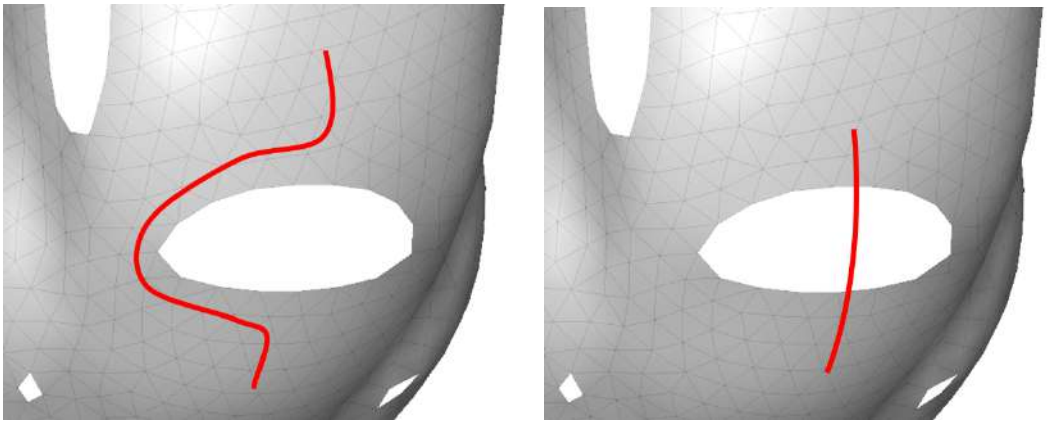


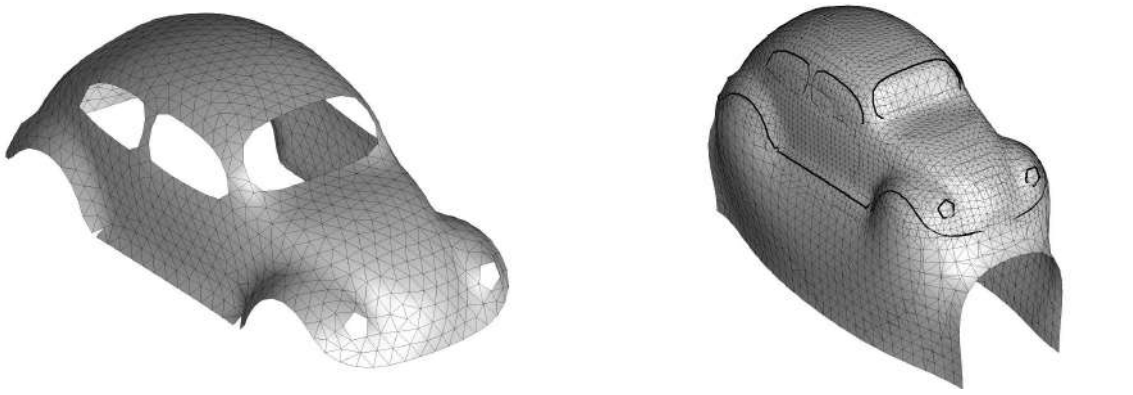
Figure IV-C.2.5: MDS parameterization $\Phi = [\sqrt{\lambda_1}V_1, \sqrt{\lambda_2}V_2]$ from the estimated geodesic distances



(a) Geodesic path on raw mesh M

(b) Geodesic path on M with the help of underlying Poisson surface

Figure IV-C.2.6: Our algorithm computes an underlying Poisson surface M^* to fix the geodesic paths on non-convex mesh M



(a) Original mesh M (b) Poisson surface M^* underlying the raw mesh M^*

Figure IV-C.2.7: Raw mesh M and its underlying Poisson surface approximation M^*

in Sects. IV-C.2.3.2 and IV-C.2.3.3. $\vec{N} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is a vector field in $\mathbb{R}^3$ whose value $\vec{N}(x)$ is the normal vector to the original surface M if $x \in M$, and $\vec{N}(x) = 0$ everywhere else.

To solve Eq. (IV-C.2.16), the PCL library uses a hierarchical 3D spatial discretization and a Finite Differences approach [49].

Fig. IV-C.2.7 plots the Poisson surface filling M^* for a given non-convex mesh M. The resulting geodesic field (Fig. IV-C.2.8) is distributed along the original mesh M and its extents $M^* - M$. The corresponding parameterization of the underlying Poisson surface M^* (Fig. IV-C.2.9(a)) is finally trimmed in order to retrieve the final parameterization Φ of M (Fig. IV-C.2.9(b)). Fig. IV-C.2.10 plots the chessboard texture maps for both parameterization without Poisson filling (Fig. IV-C.2.10(a)) and parameterization with Poisson surface filling (Fig. IV-C.2.10(b)). As illustrated, using Poisson filling reduces parameterization distortions close to mesh holes and boundary concavities.

IV-C.2.4 Results and Discussion

To test our mesh parameterization algorithm we run tests with several parameterizable surfaces. Sect. IV-C.2.4.1 presents a comparison of our mesh parameterization algorithm without Poisson surface filling vs. Poisson surface filling, for quasi-developable non-convex meshes. Sect. IV-C.2.4.2 presents parameterization results for some challenging, strongly non-developable data sets. Finally, Sect. IV-C.2.4.3 presents the application of our parameterization algorithm for the reverse engineering of a scanned cow vertebra.

IV-C.2.4.1 Non-filling vs. Poisson Filling Parameterization

Fig. IV-C.2.11 plots the parameterization results (2D Φ coordinates and 3D texture map) for the Mask data set. Without using Poisson surface reconstruction, the resulting parameterization is bijective with relatively low distortion, except for the eye holes (Fig. IV-C.2.11(a)). However, such

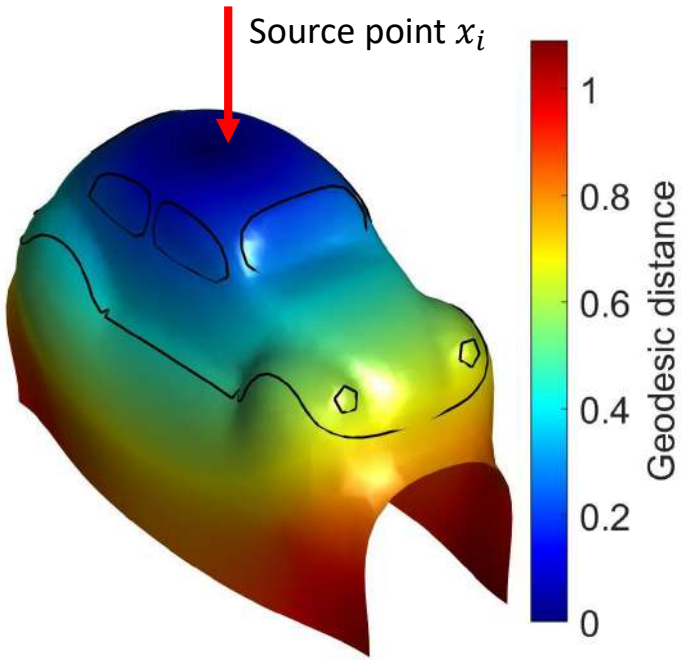
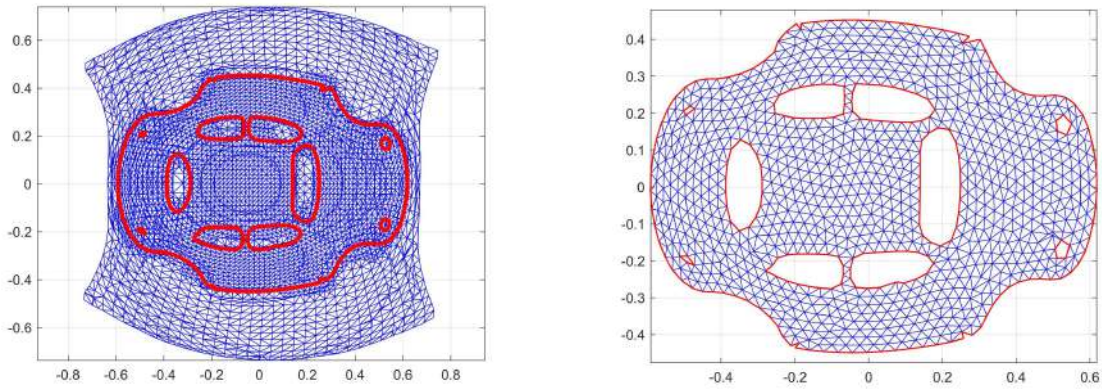


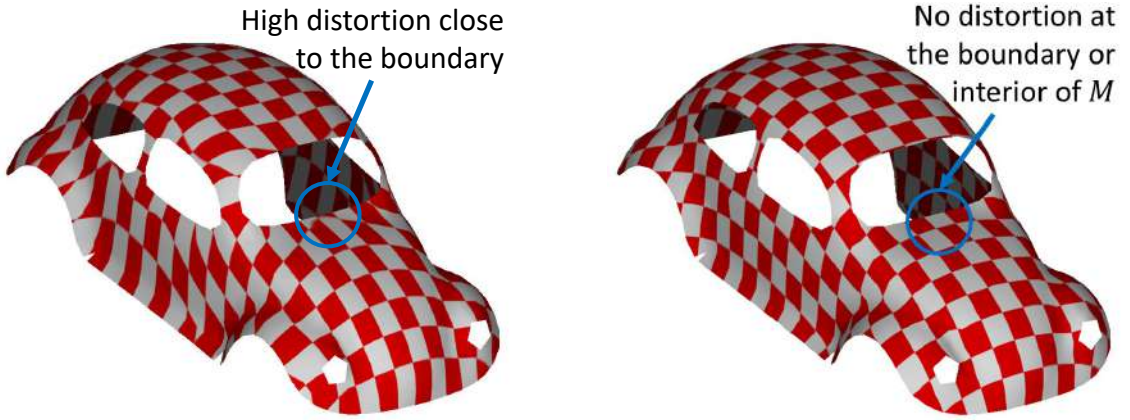
Figure IV-C.2.8: Geodesic distance estimation on the Poisson surface M^* approximating raw mesh M



(a) Mesh parameterization of the Poisson surface M^*

(b) Trimmed parameterization Φ

Figure IV-C.2.9: Parameterization of the Poisson surface M^* and its corresponding trimmed parameterization Φ.



(a) Texture on raw mesh M, distorted at holes and boundary concavities

(b) Texture using underlying Poisson surface M^*, undistorted

Figure IV-C.2.10: Chessboard texture maps from our heat-geodesic based parameterization

a parameterization is improved by applying the Poisson reconstruction, reducing the distortion close to mesh holes and boundary concavities (Fig. IV-C.2.11(b)).

A more extreme case is illustrated in Fig. IV-C.2.12, with the S-trimmed-on-cone dataset from [16]. Since the S shape is trimmed on a cone, M is a fully developable surface.Yet, the heat-geodesic parameterization on M fails to compute a bijective parameterization (Fig. IV-C.2.12(a)). Application of the Poisson (extended underlying surface M^*) filling (Fig. IV-C.2.12(b)) solves this issue, resulting in a non-distorted bijective parameterization. This case illustrates (a) the vulnerability of geodesic-based parameterizations in the presence of mesh holes or concavities at mesh borders, (b) the capacity of the underlying Poisson extended surface to prevent (a), (c) the natural manner in which geodesic curves isometrically parameterize a developable surface.

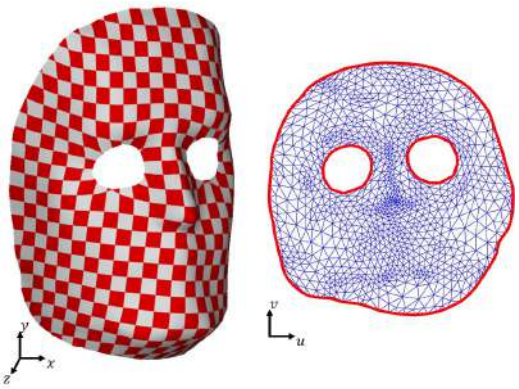
IV-C.2.4.2 Parameterization of Strongly Non-Developable Meshes

In this section, the public data sets Foot, Gargoyle and Cow are parameterized with our heat-based geodesics algorithm. These benchmark datasets contain an artificially introduced border [25], which allows their parameterization. Figs. IV-C.2.13(a) and IV-C.2.13(b) plot the parameterization results for the seam Foot and Gargoyle, respectively. The resulting parameterization is bijective, with some noticeable distortion (e.g. in the Gargoyle head). Fig. IV-C.2.13(c) plots our parameterization results for the seamed Cow, which is not bijective in the head, legs and tail.

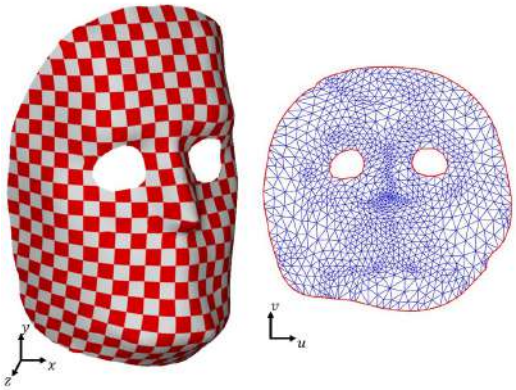
It is worth noting that although parameterizable, these benchmark data sets are strongly non-developable without a proper mesh pre-segmentation. This fact is illustrated in the next section.

IV-C.2.4.3 Reverse Engineering of Cow Vertebra

For this section, the vertebra of a cow is scanned using a 3D optical scanner. The scanned mesh is closed and therefore accepts no (bijective) parameterization. The closed mesh is segmented into quasi-developable meshes using a heat-based segmentation approach [54]. Fig. IV-C.2.14

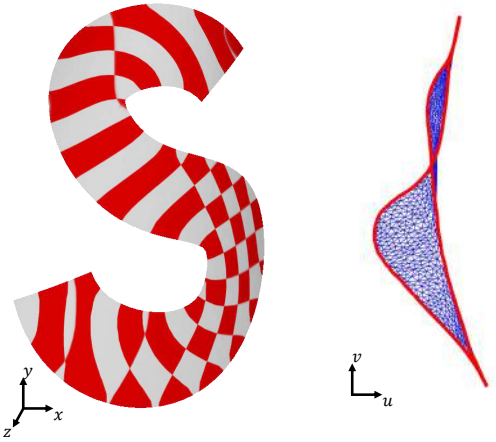


(a) Parameterization of M. 3D texture map (left) and 2D Φ coordinates (right). Distorted at holes and concavities.

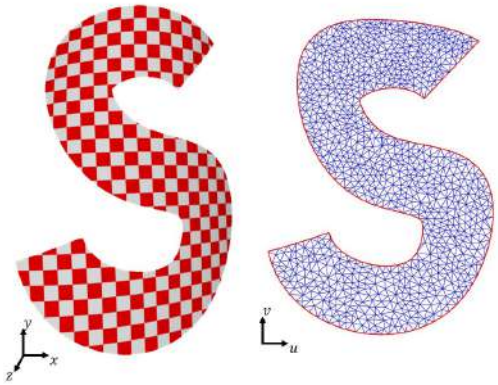


(b) Undistorted parameterization with underlying surface M^*. 3D texture map (left) and 2D Φ coordinates (right).

Figure IV-C.2.11: Data set Mask. Hole-distorted and undistorted heat-geodesic parameterizations.

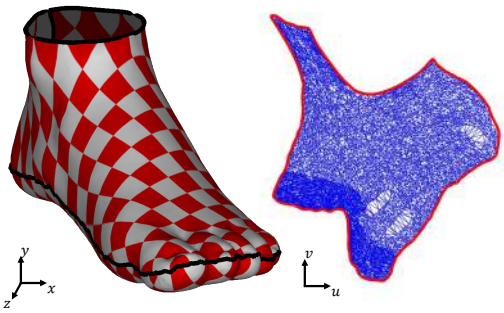


(a) Non-bijective parameterization of M. 3D texture map (left) and 2D Φ coordinates (right).

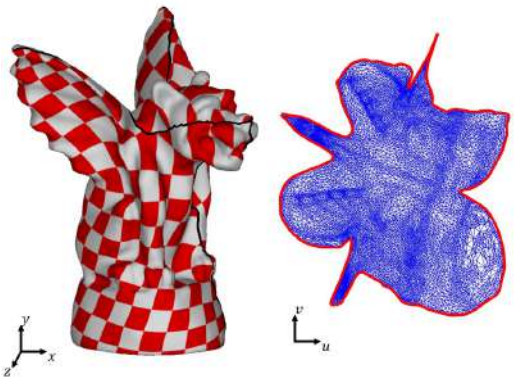


(b) Undistorted and bijective parameterization with underlying surface M^*. 3D texture map (left) and 2D Φ coordinates (right).

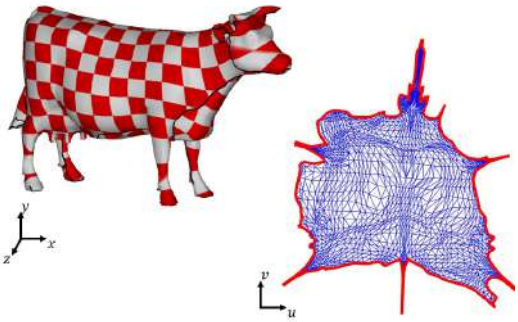
Figure IV-C.2.12: Data set S-trimmed-on-cone. Non-bijective parameterization using raw mesh M. Bijective isometric parameterization using underlying Poisson mesh M^*.



(a) Data set Foot. Bijective parameterization.



(b) Data set Gargoyle. Bijective parameterization.



(c) Data set Cow. Non-bijective parameterization near the legs and tail.

Figure IV-C.2.13: Heat-geodesic parameterization of seam meshes [25]. The strong non-developability of the meshes produces high parameterization distortions and in some cases non-bijectiveness.

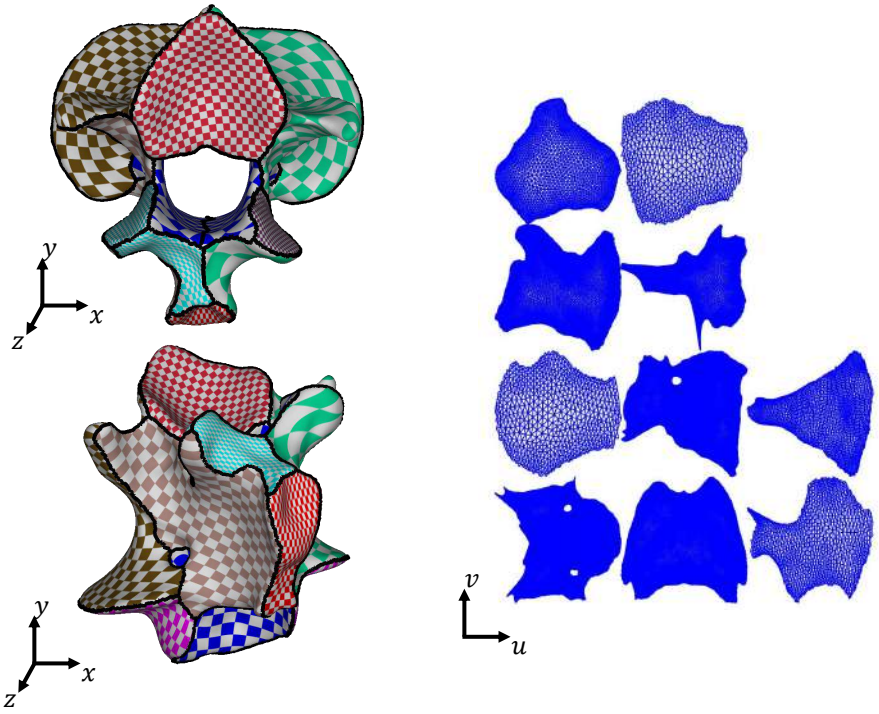


Figure IV-C.2.14: Cow Vertebra data set. Undistorted parameterization using heat-based geodesic maps. Segmentation by Mejia et al [54].

plots the parameterization results for the segmented mesh. Each sub-mesh bijective parameterization presents low distortion, enabling further reverse engineering operations such as NURBs re-parameterization [16], finite element analysis, structural optimization and/or dimensional inspection [54].

IV-C.2.5 Conclusions

This article presents the implementation of a novel application of heat propagation in 2-manifolds used for mesh parameterization. The temperature contours for the heat kernels computed on the mesh are perpendicular at each point to the geodesic curves in the surface. This principle permits to determine geodesic maps in the mesh and in particular vertex-to-vertex geodesic distances. Although Finite Element methods produce better results as the mesh resolution increases (higher polygon count), the geodesics estimation method still produces robust results for low polygon count meshes as each geodesic path traverses across the mesh faces (contrary to graph algorithms which traverse the mesh graph). A quasi-isometric bijective function (i.e. the parameterization) is synthesized, to map the 3D mesh to the parameter (2D) space. This parameterization is near isometric in the sense that geodesic distance on the mesh between any two points on the mesh approximates the Euclidean distance between their images in the parametric space. This approach

obviously limited to meshes which are quasi-developable or developable, since mesh developability is necessary for the existence of an isometric parameterization for it.

Our approach circumvents the weakness of geodesic maps in the presence of mesh interruptions (boundary concavities and mesh holes) by devising an underlying continuous Poisson surface that approximates the input mesh but contains no such interruptions. This underlying surface allows for geodesic maps to be computed on it, which are also valid in the input mesh. In this manner, the parameterization computed for the Poisson surface is valid for the input mesh. Finally, the boundaries of the input mesh are explicitly marked on the parameterization to obtain a *trimmed surface* or *FACE* in the Boundary Representation jargon.

Future work is required in these aspects: (a) to eliminate redundant computation that is present in the construction of heat-based geodesic maps, (b) to use failures in the bijectiveness of the computed parameterizations to force mesh segmentation, when the input mesh is strongly non-developable.

IV-C.3

Mesh Segmentation Driven By Bijective Parameterization

Daniel Mejia[1,2], Oscar Ruiz-Salguero[1], Carlos Cadavid[1], Jairo R. Sánchez[2], Jorge Posada[2], Aitor Moreno[2], Diego A. Acosta[3]

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, Medellín, Colombia.

[2] Vicomtech, Donostia / San Sebastián, Spain.

[3] Grupo de Desarrollo y Diseño de Procesos, Universidad EAFIT, Medellín, Colombia.

Citation

Daniel Mejia, Oscar Ruiz-Salguero, Carlos Cadavid, Jairo R. Sánchez, Jorge Posada and Diego A. Acosta (**2018**). Mesh segmentation driven by bijective parameterization. In *Proceedings of the 12th International Symposium on Tools and Methods of Competitive Engineering (TMCE 2018)*. May 7-11, Las Palmas de Gran Canaria, Spain. ISBN: 978-94-6186-910-4.

Award Best Senior Paper Contribution TMCE-2018.

TMCE 2018

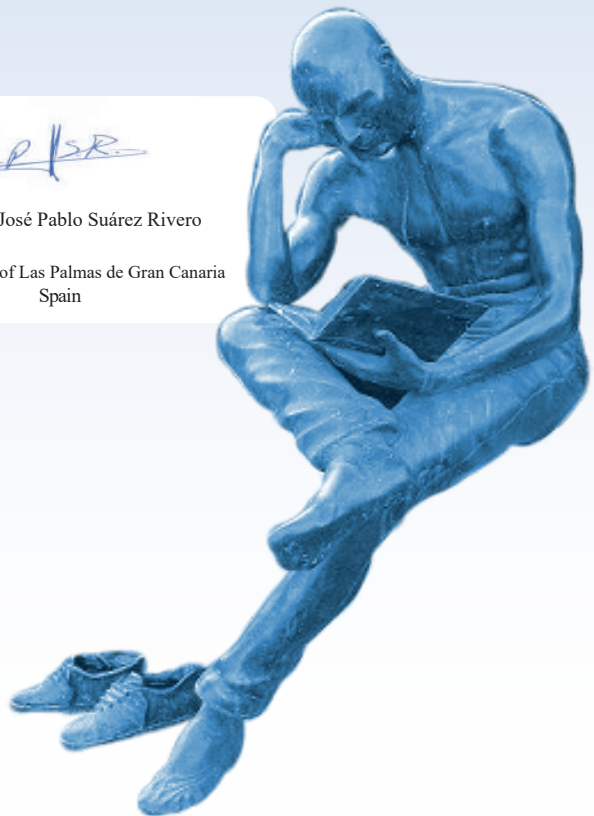
Tools and Methods of Competitive Engineering

Best Senior Contribution Award

Mesh Segmentation Driven By Bijective Parameterization

D. Mejia **O. Ruiz-Salguero** **C. Cadavid**
J.R. Sánchez **J. Posada** **D. Acosta**

The Organizing Committee of the International Tools and Methods of Competitive Engineering Symposia has the pleasure to recognize your outstanding contribution to the past events as well as to the TMCE 2018 Symposium.



Prof. Dr. Imre Horváth
co-chairman
Delft University of Technology
the Netherlands

Prof. Dr. José Pablo Suárez Rivero
University of Las Palmas de Gran Canaria
Spain

Abstract

In Reverse Engineering (RE), mesh segmentation is usually followed by mesh parameterization. A segmentation of a mesh M is not acceptable if the parameterization of its sub-mesh M_i fails. A defined parameterization failure criterion is given by the non-bijective nature of the parameterization $\psi_i : M_i \rightarrow \mathbb{R}^2$. Current mesh segmentation algorithms produce either (1) topology - based partitions which present highly non-developable sub-meshes or (2) geometry - based partitions which present over-segmentation for organic meshes. The algorithm that we present in this manuscript partially overcomes these limitations by using the failure regions of a (Hessian - based) parameterization algorithm as basis for mesh segmentation. Our algorithm produces a hierarchical subdivision of M_i only if the Hessian - based parameterization ψ_i is not bijective, avoiding mesh over-segmentation. As no further subdivision is triggered, the final parameterization is obtained using an angle preserving parameterization algorithm (CONFOP). The decision of using two parameterization algorithms obeys the fact that Hessian parameterization is a Dimensionality Reduction - based algorithm which allows parameterization foldings (ideal for segmentation) while CONFOP parameterization penalizes such foldings (ideal for bijective parameterization). Our interplay of segmentation / parameterization is conservative in subdivision and yet produces parameterizable sub-meshes. The test runs show fully bijective mappings on several landmark datasets. Ongoing work addresses sub-mesh boundary smoothing to avoid jagged boundary FACEs in the Boundary Representation.

Keywords: Mesh Segmentation, Mesh Parameterization, Mesh Hessian, Conformal Mapping, Reverse Engineering

Glossary

RE	Reverse Engineering.
DR	Dimensional Reduction.
CAD	Computer Aided Design.
CAM	Computer Aided Manufacturing.
CAE	Computer Aided Engineering.
CONFOP	CONformal OPtimization algorithm for mesh parameterization.
M	Triangular mesh of a connected 2-manifold embedded in $\mathbb{R}^3$. $M = (X, T)$ is composed by the set of triangles $T = \{t_1, t_2, \ldots, t_m\}$ with vertex set $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^3$.
$\psi_{hessian}$	Hessian - based non-bijective parameterization of M. $\psi_{hessian} : M \rightarrow \mathbb{R}^2$.
M_i	A sub-mesh of the segmentation of M. $\bigcup_{i=1}^{k} M_i = M$, $M_i \cap M_j = \emptyset \, (i \neq j)$.
O^+, O^-	Triangle 2D orientations subsets. O^+, O^- contain all the mesh triangles which have positive and negative orientation, respectively, in the (non-bijective) Hessian parameterization. Therefore, $M = O^+ \cup O^-$.
f	Continuous smooth function $f \in \mathcal{C}^2(M)$ defined on the mesh.

$\mathbf{H_x^{tan}}$	Tangent Hessian of M defined at any point $x \in M$.
$\mathcal{H}$	Hessian functional on M. $\mathcal{H} : \mathcal{C}^2(M) \rightarrow \mathcal{C}(M)$.
$\mathbf{K}$	$n \times n$ matrix which discretizes the $\mathcal{H}$ functional on M. The first two non-constant eigenvectors of $\mathbf{K}$ define the Hessian parameterization $\psi_{hessian}$.
ε	Segmentation user - defined parameter which defines the minimum sub-mesh size.
ψ_i	CONFOP 2D parameterization of M_i. $\psi_i : M_i \rightarrow \mathbb{R}^2$.
$A^{(t_j)}$	2×3 transformation matrix that maps the triangle $t_j \in M_i$ to the plane $\mathbb{R}^2$.
$\mathbf{B}$	An orthonormal basis $\mathbf{B} = [\vec{v}_1^{t_j}, \vec{v}_2^{t_j}]$ for the plane tangent to triangle t_j.

IV-C.3.1 Introduction

In the context of manufacturing, RE is a core part of the Visual Computing field, which is considered a key enabling technology for the implementation of the Industry 4.0 initiative [55]. This technology can be used to digitalize physical prototypes of a product during the design phase to evaluate its behavior using CAD CAM CAE tools.

The general workflow of a RE process begins with the acquisition of the geometry of the physical part in form of a point cloud using a 3D scanner. This geometry is processed to reconstruct the connectivity of the points in the form of a triangle mesh. In this work, we address the segmentation/parameterization problem that finally allows to obtain a representation of the mesh suitable for be processed by CAD CAM CAE tools.

Given a triangular mesh M, the mesh segmentation / parameterization problem is defined as follows:

1. Split M into a set of disjoint and connected sub-meshes $\{M_1, M_2, \ldots, M_k\}$. The computed partition of M must favor the developability of each M_i.

2. For each sub-mesh M_i, compute a bijective parameterization $\psi_i : M_i \rightarrow \mathbb{R}^2$.

In contrast to common state of the art mesh segmentation methods, a natural approach to compute the segmentation of M could be to learn and segment M from non-bijective parameterizations until the computed segmentation performs bijective parameterizations. Following such idea, this manuscript implements an algorithm which computes a Hessian - based non-bijective parameterization $\psi_{hessian}$ of M. Such parameterization produces mesh foldings in the parameter space $\mathbb{R}^2$ which define the new sub-mesh boundaries. This segmentation approach is applied hierarchically on each sub-mesh M_i until the computed segmentation is completely bijective. The final parameterization ψ_i of the developable sub-mesh M_i is computed with an angle preserving mesh parameterization algorithm (CONFOP), which minimizes mapping distortions.

Our algorithm contributes to the mesh segmentation / parameterization state of the art by extracting information of non-bijective parameterizations in order to guide the mesh segmentation. As a consequence, our algorithm (1) produces a fully bijective segmentation of M and, (2) avoids over-segmentation as each sub-mesh is only sub-segmented if a non-bijective request has been triggered.

The remainder of this manuscript is organized as follows: Sect. IV-C.3.2 reviews the relevant literature. Sect. IV-C.3.3 describes the mesh segmentation / parameterization algorithm. Sect. IV-C.3.4 presents and discusses results of the conducted experiments. Sect. IV-C.3.5 concludes the paper and introduces what remains for future work.

IV-C.3.2 Literature review

This section reviews the taxonomy of mesh segmentation / parameterization algorithms. Mesh segmentation algorithms can be classified depending on the surface features used to partition the mesh as follows:

IV-C.3.2.1 Geometry - based Mesh Segmentation

Geometry - based algorithms capture local geometric features on the surface. Features such as the dihedral angles between adjacent triangles [56], the gaussian curvature [57] or the instantaneous speed [58] on the surface, are common descriptors that characterize CAD-like models. A region growing method is then applied to the characterized model in order to produce the segmentation [59].

Since curvature and sharpness indicators on the surface are used to segment the mesh, geometry - based algorithms are able to produce highly developable mesh partitions with low curvatures and non-sharp transitions. However, the local nature of these algorithms make them highly sensitive to noise and to organic (non-CAD) meshes, resulting frequently in high mesh over-segmentation.

IV-C.3.2.2 Topology-based Mesh Segmentation

On the other hand, topology based algorithms take advantage of the mesh connectivity encoded in the spectrum (eigenvalues and eigenvectors) of some mesh operators such as the graph Laplacian [60] or the dual Laplacian [61]. The Laplace-Beltrami operator is a type of mesh Laplacian which has gained a lot of importance in the context of Computational Geometry [62]. The segmentation can be achieved finally by applying a k-means algorithm to the Laplacian eigenvectors [63] or by merging such eigenvectors into a single segmentation field [64]. In order to account for geometric features (in addition to the topologic ones), operators such as the Giaquinta - Hildebrandt incorporate connectivity weights which capture mesh concavities / convexities [65].

Topology - based segmentation algorithms do not have the over-segmentation problem inherent to geometry - based algorithms as the former ones are designed to produce small partitions with large sub-meshes. However, these large sub-meshes are in most cases non-developable. Such a shortcoming is critical in mesh parameterization applications as the segmentation algorithm cannot be directly applied in the segmentation / parameterization context.

IV-C.3.2.3 Mesh Parameterization

Mesh parameterization is a special case of Dimensional Reduction (DR) where a 2-manifold M (triangular mesh) embedded in $\mathbb{R}^3$ must be mapped to the plane through a bijective (no triangle flips nor mesh overlaps) mapping $\psi : M \rightarrow \mathbb{R}^2$. As a consequence, some DR algorithms have been used successfully in mesh parameterization applications. In [15], a distance preserving DR algorithm (Isomap) is used to compute the parameterization of the mesh, which is then used to

apply a 2D texture to the surface. Other DR algorithms such as Laplacian Eigenmaps and Hessian Locally Linear Embedding have been used for mesh parameterization in order to fit parametric surfaces [16, 40].

In addition to bijectiveness, the preservation of geometric properties such as angles (conformal maps), areas (authalic maps) or distances (isometric maps) becomes a requisite in most mesh parameterization applications. Discrete Conformal Maps algorithms [66] apply the Riemann mapping theorem to solve a complex - valued system of equations which produces low angle distortion mappings. Similarly, Angle-Based Flattening (ABF) algorithms [14] solve a minimization problem in the domain of mesh triangle angles. Harmonic map algorithms [39] minimize the Laplace-Beltrami operator in order to compute area preserving parameterizations. Isometric (distance preserving mappings) can be computed by both preservation of angles and areas [43, 67].

Seam-based algorithms compute mesh parameterization of full meshes by introducing an artificial boundary (seam) which partially cuts the mesh allowing developability. The seamster algorithm [25] is a common approach in the mesh parameterization literature which introduces such a boundary in high curvature - low visibility regions of the mesh. However, parameterization of these meshes easily introduces large distortions, and global non-bijectiveness (mesh overlaps) [67]. The recently developed unfolding from mesh failures algorithm [68] introduces a seam based segmentation of the mesh by learning from unfolding failures. However, this algorithm is computationally expensive due to its heuristic nature, rendering it useless for modest-to-large size datasets.

IV-C.3.2.4 Conclusions of the Literature Review

Most mesh segmentation algorithms present shortcomings such as: (1) high mesh over-segmentation in the case of geometry - based algorithms, and (2) non-developability in the case of topology - based algorithms. To overcome these problems, this article presents a parameterization - driven mesh segmentation algorithm. Our algorithm aims to segment the surface using the failure information of a non-bijective parameterization. The algorithm incorporates: (1) a Hessian - based segmentation algorithm which detects sub-mesh boundaries from a non-bijective Hessian parameterization, and (2) an angle preserving (CONFOP) mesh parameterization algorithm [67]. The algorithm produces fully bijective parameterizations and avoids over-segmentation. Test results also show that our algorithm is able to identify mesh protrusions.

IV-C.3.3 Methodology

To compute a bijective parameterization of the triangular mesh M, our algorithm is implemented as follows (Fig. IV-C.3.1):

1. A Hessian - based mesh parameterization algorithm [40] is applied on M.
2. The non-bijective parameterization $\psi_{hessian}(M)$ overlaps in the parameter space ($\mathbb{R}^2$). Each mesh triangle is classified according to its orientation in the plane (positive O^+ and negative O^- orientation).
3. The segmentation $\{M_1, M_2, \ldots, M_k\}$ of M is computed by extracting the connected components of the orientation sets O^+ and O^- (as they are probably non-connected).

4. Some triangles become isolated in the previous step. These triangles are merged into large neighbor sub-meshes.

5. To obtain the (potentially) final parameterization $\psi_i(M_i)$, an angle preservation mesh parameterization algorithm (CONFOP) [67] is applied.

6. If the parameterization ψ_i of the sub-mesh M_i is bijective, then stop the algorithm for the current sub-mesh and proceed with the next one. Otherwise, go to step (1) using mesh M_i as input.

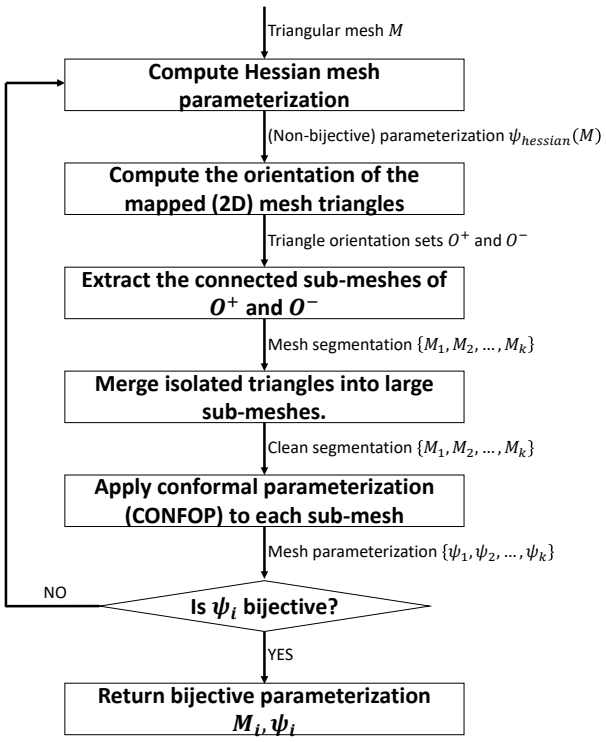


Figure IV-C.3.1: Mesh segmentation/parameterization interplay algorithm.

The previous algorithm uses two different mesh parameterization algorithms: (a) Hessian - based parameterization and (b) angle preserving mesh parameterization (CONFOP). The reason we choose both these algorithms is because Hessian parameterization allows mesh foldings and triangle flips, which are exploited by the segmentation algorithm. On the other hand, CONFOP only permits a single triangle orientation for the full parameterization and minimizes angle distortions, resulting in a higher quality parameterization result.

Additionally, step (6) from the algorithm introduces a hierarchical segmentation where the algorithm requests a subdivision of sub-mesh M_i if its current parameterization is non-bijective. As a consequence, the algorithm produces a conservative segmentation with large sub-meshes, avoiding over-segmentation and vouching bijectivity.

Each step of the algorithm is described in detail in the remainder of this section.

IV-C.3.3.1 Hessian-based Parameterization

A Hessian-based mesh parameterization algorithm [40] is applied on the full triangular mesh M. Since the mesh M is non-developable, such parameterization is non-bijective. However, this non-bijectiveness is exploited in further sections to produce a developable segmentation of M.

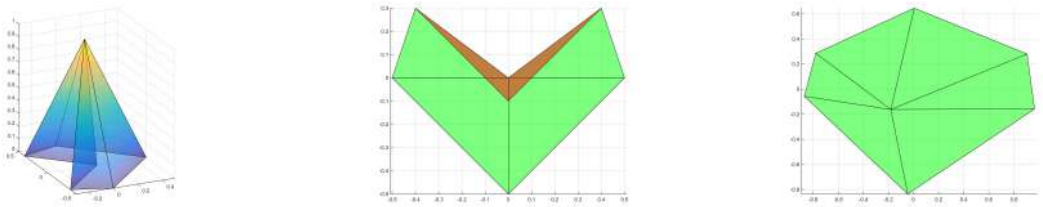
According to [40], a parameterization of M is given by the first 2 non-constant eigenvectors of the Hessian functional $\mathcal{H}$, defined as:

$$\mathcal{H}f = \int_M \|\mathbf{H}^{\mathbf{tan}}_{\mathbf{x}} f\|^2_F \mathrm{dA} \approx \mathbf{f^T K f} \tag{IV-C.3.1}$$

where $\|\cdot\|_F$ is the Frobenius norm, dA is the surface differential, $\mathbf{H}^{\mathbf{tan}}_{\mathbf{x}}$ is the (tangent) Hessian at any point $x \in M_i$, $f \in \mathcal{C}^2(M)$ is a smooth function defined on M, $\mathbf{f} = \{f_1, f_2, \cdots, f_n\}$ are the corresponding discrete values of f at each vertex $x_i \in M$, and $\mathbf{K}$ is the $n \times n$ Hessian estimator matrix. The tangent mesh Hessian $\mathbf{H}^{\mathbf{tan}}_{\mathbf{x}}$ is defined as follows:

$$\mathbf{H}^{\mathbf{tan}}_{\mathbf{x}} f = \begin{bmatrix} \frac{\partial^2 f}{\partial b_1^2} & \frac{\partial^2 f}{\partial b_1 b_2} \\ \frac{\partial^2 f}{\partial b_2 b_1} & \frac{\partial^2 f}{\partial b_2^2} \end{bmatrix} \tag{IV-C.3.2}$$

with $b_1, b_2, \in \mathbb{R}^3$ being an orthonormal basis for the tangent plane at $x \in M$.



(a) High frequency mesh neighborhood.

(b) Non-bijective local parameterization using average normal Hessian (Mejia et al., 2016).

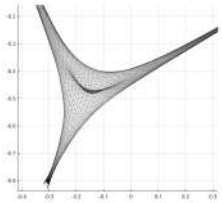
(c) Bijective local parameterization using conformal (CONFOP) approach (Mejia, Acosta, & Ruiz-Salguero, 2017).

Figure IV-C.3.2: Parameterization of high frequency mesh neighborhood. Hessian (failure) and conformal (success) approach.

To estimate this tangent plane, standard Hessian parameterization algorithm uses PCA [69] or the average triangle normals method [40]. However, these methods produce in some special cases locally non-bijective parameterizations (Figs. IV-C.3.2(a),IV-C.3.2(b)), affecting the quality of the global parameterization. Instead, to estimate the tangent plane at x_i our algorithm uses CONFOP [67] to unfold the triangles adjacent to x_i. Such unfolding results in tangent planes which are always bijective as seen in Fig. IV-C.3.2(c). The CONFOP algorithm is detailed in Sect. IV-C.3.3.4.

The matrix $\mathbf{K}$ is semidefinite-positive [40,69]. Therefore, the eigenvalues of $\mathbf{K}$ are larger or equal than zero. The parameterization $\psi_{hessian}(M)$ is extracted by computing the first two eigenvectors of $\mathbf{K}$ with the smallest non-zero eigenvalue. Such eigenvectors correspond to an orthogonal basis for all linear functions (and as a consequence, a basis for all parameterizations) defined on M.

IV-C.3.3.2 Hessian-based Segmentation



(a) (Non-bijective) Hessian parameterization of the full mesh.

(b) Sub-mesh classification sets O^+ (yellow) and O^- (blue) according to Hessian - based orientation.

(c) Connected sub-meshes with common Hessian - based orientation.

Figure IV-C.3.3: Airplane dataset. Hessian - based mesh segmentation.

For the non-developable mesh M, the Hessian-based parameterization algorithm produces a non-bijective parameterization $\psi_{hessian}(M)$ (Fig. IV-C.3.3(a)). As a consequence, the triangles of the non-bijective parameterization present positive and negative orientation on different regions of such parameterization. Classifying each triangle in M with its corresponding orientation in $\psi_{hessian}(M)$ splits the mesh M into two subsets $M = O^+ \cup O^-$ (one for each possible orientation of the triangles) as illustrated in Fig. IV-C.3.3(b).

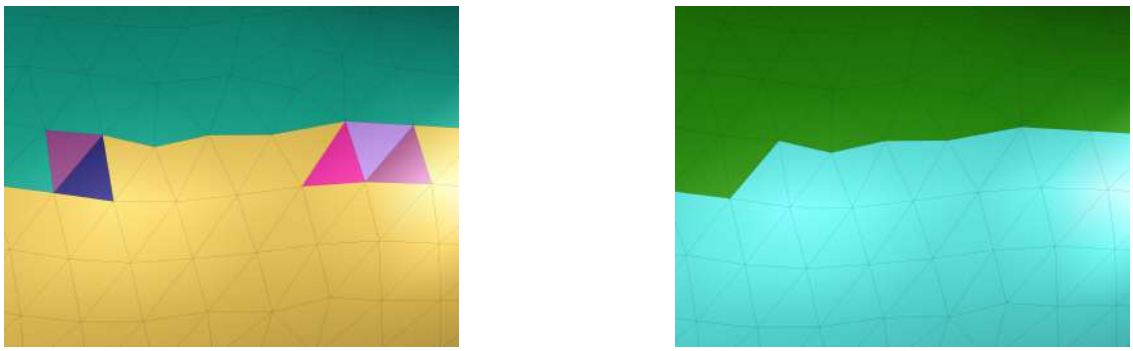
By definition, each connected sub-mesh in O^+ and O^- (Fig. IV-C.3.3(b)) accepts a parameterization. Such a collection of connected sub-meshes is itself a mesh segmentation (Fig. IV-C.3.3(c)).

Figure IV-C.3.4: Airplane dataset. Mesh segmentation using CONFOP parameterization instead of Hessian parameterization.

To compute the mesh parameterization $\psi_{hessian}(M)$, the Hessian-based algorithm aligns the triangles of M in the parameter space, encouraging a consistent orientation between adjacent triangles. However, such algorithm does not prefer a given orientation against the other. Other mesh

parameterization algorithms enforce a unique orientation, resulting in a low quality mesh segmentation of M with smaller highly nonconvex sub-meshes and sub-mesh boundaries with significantly higher frequencies (Fig. IV-C.3.4) than the Hessian-based one (Fig. IV-C.3.3(c)).

IV-C.3.3.3 Process Small Sub-meshes



(a) Isolated triangles on raw segmentation.

(b) Merged segmentation.

Figure IV-C.3.5: Small sub-meshes and isolated triangles are merged into large sub-meshes.

The Hessian-based segmentation algorithm may produce an over-segmentation with isolated triangles and small sub-meshes (Fig. IV-C.3.5(a)). This over-segmentation occurs due to:

1. Several triangles become isolated near sub-mesh boundaries since the Hessian parameterization algorithm tends to collapse triangles close to the boundary to straight lines (e.g. Fig. IV-C.3.6(a)).

2. Small sub-meshes occur in high frequency non-developable zones which are folded by the Hessian parameterization algorithm.

To handle this over-segmentation, we implement a simple algorithm to merge isolated triangles and small sub-meshes into large ones as follows:

1. Small sub-meshes are detected by the algorithm by a user defined parameter ε which defines the minimum sub-mesh size allowed by the algorithm.

2. If a given sub-mesh is smaller than the given threshold, it is merged into an adjacent sub-mesh.

3. If a small sub-mesh (or an isolated triangle) has two or more adjacent sub-meshes, it is merged onto the sub-mesh with which it shares the longest boundary curve.

To improve the quality of the mesh segmentation, changes to geometry and topology could be implemented (such as curve smoothing or edge flips at sub-mesh boundaries). However, this is left for future research.

IV-C.3.3.4 Conformal Mesh Parameterization (CONFOP)

After computing a developable segmentation of M, a mesh parameterization algorithm is applied on each sub-mesh M_i. In order to highly preserve the sub-meshes shape (preserve angle), the CONFOP (conformal optimization) algorithm is implemented. CONFOP is a specialization of the angle/area preserving algorithm presented in [67], where only the optimization term regarding angle preservation is used. In CONFOP, the following minimization problem arises for each sub-mesh M_i:

$$\min_{\Psi} \sum_{t_j \in M_i} \left(A_{11}^{(t_j)} - A_{22}^{(t_j)}\right)^2 + \left(A_{12}^{(t_j)} + A_{21}^{(t_j)}\right)^2 \tag{IV-C.3.3}$$

where $\Psi = \{\psi_1, \psi_2, \ldots, \psi_n\}$ is the mapping of the sub-mesh nodes to the CONFOP parameterization (i.e. $\psi_j = \psi(x_j)$, $\psi : M_i \rightarrow \mathbb{R}^2$), and the matrix $A^{(t_j)}$ is the transformation matrix that maps the triangle $t_j \in M_i$ to the plane (i.e. $\psi(t_j)$) defined as:

$$A^{(t_j)} = \left[\overline{\psi}_1^{(t_j)}, \overline{\psi}_2^{(t_j)}, \overline{\psi}_3^{(t_j)}\right] \cdot pinv\left(\mathbf{B^T}\left[\overline{x}_1^{(t_j)}, \overline{x}_2^{(t_j)}, \overline{x}_3^{(t_j)}\right]\right) \tag{IV-C.3.4}$$

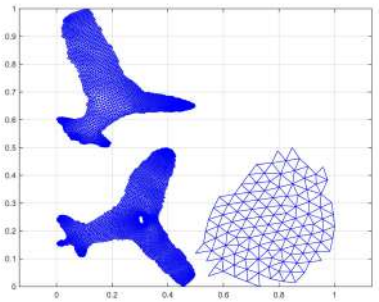
with $\mathbf{B} = [\vec{v}_1^{t_j}, \vec{v}_2^{t_j}]$ an orthonormal basis for the plane tangent to triangle t_j and $\overline{x}_i^{t_j}$, $\overline{\psi}_i^{t_j}$ are the mean centered coordinates of triangle t_j on the mesh and the parameterization, respectively:

$$\begin{aligned} \overline{x}_i^{(t_j)} &= x_i^{(t_j)} - \sum_{k=1}^{3} x_k^{(t_j)}, \\ \overline{\psi}_i^{(t_j)} &= \psi_i^{(t_j)} - \sum_{k=1}^{3} \psi_k^{(t_j)} \end{aligned} \tag{IV-C.3.5}$$

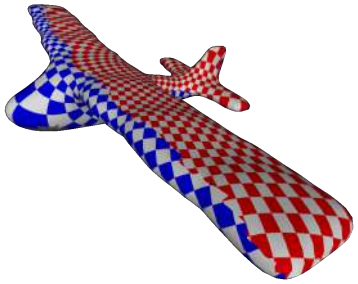
The conformal parameterization Ψ of the sub-mesh M_i is obtained by solving the minimization problem in Eq. (IV-C.3.3). Such problem reduces to the solution of a linear system of equations as opposed to the approach taken in [67], which requires iterative nonlinear Marquardt optimization. This improvement in performance arises due to the fact that our algorithm does not consider the area-preserving term which is highly nonlinear. The CONFOP parameterization produces a bijective mapping for the developable partition of the mesh as presented in Fig. IV-C.3.6(a). The algorithm enforces a unique orientation for the whole mesh parameterization and minimizes angle distortions (as illustrated on the chessboard texture map in Fig. IV-C.3.6(b)).

IV-C.3.4 Results

We test our algorithm with several datasets from the Princeton benchmark database [70]. Fig IV-C.3.7 presents segmentation/parameterization results for the Teddy, Gargoyle, Cow and Armadillo meshes. Our Hessian - based algorithms produces conservative mesh partitions ranging from only two sub-meshes (such as the Teddy and Gargoyle datasets - Figs. IV-C.3.7(a) and IV-C.3.7(d)) to 9 sub-meshes (Armadillo dataset - Fig. IV-C.3.7(j)). Boundary curves do not follow geometry but rather divide the surface into the large, low curvature sub-meshes. In all test cases, CONFOP parameterization achieved fully bijective parameterizations (as seen in Figs. IV-C.3.7(b), IV-C.3.7(e), IV-C.3.7(h) and IV-C.3.7(k)).



(a) Airplane (bijective) mesh parameterization.



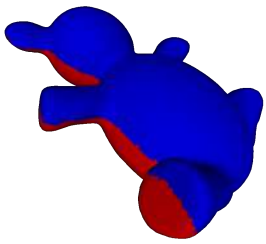
(b) Airplane texture mapping.

Figure IV-C.3.6: Conformal mesh parameterization on the Hessian-based segmentation of the Airplane dataset.

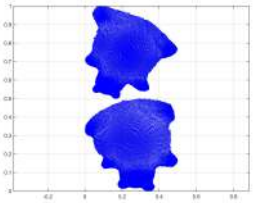
Chessboard texture maps show low shape (angle) distortion for the Teddy and Gargoyle parameterizations (Figs. IV-C.3.7(c) and IV-C.3.7(f)). However, the Cow (Fig. IV-C.3.7(i)) presents high angle distortions at mesh protrusions (such as ears and horns), evidenced by the loss of the chessboard pattern in these zones. Similarly, the Armadillo dataset presents high distortions at mesh protrusions (such as fingers and toes). However, our Hessian - based algorithm partially detects these zones as they are naturally non-developable (Fig. IV-C.3.8(a)). As a consequence, our CONFOP algorithm parameterizes independently some sections of the Armadillo toes in order to guarantee parameterization bijectiveness (Fig. IV-C.3.8(b)).

IV-C.3.5 Conclusions

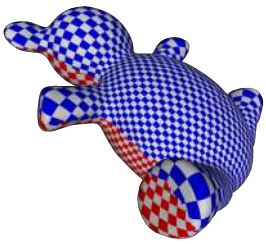
This manuscript presents a parameterization - driven segmentation algorithm for bijective parameterization of triangular meshes. Our algorithm exploits the failure regions of a Hessian - based non-bijective parameterization of the mesh to partition it into developable meshes. Subsequent sub-mesh partitioning is triggered by the algorithm only if the sub-mesh parameterization is non-bijective. As a consequence, our algorithm produces conservative (in partition size) but parameterizable mesh partitions. The final parameterization of each sub-mesh is computed using an angle preserving mesh parameterization algorithm (CONFOP). We choose to use two different mesh parameterization algorithms in our approach because (1) Hessian - based parameterization allows mesh foldings and triangle flips which are the main key for the segmentation algorithm, (2) CONFOP parameterization only permits a single triangle orientation (penalizing triangle flips and mesh foldings) and minimizes angle distortions, leading to higher quality final parameterizations. Our conducted experiments show fully bijective segmentations which are moderate in partition size. The algorithm successfully detects, segments and parameterizes mesh protrusions, which are a common challenge in mesh parameterization.



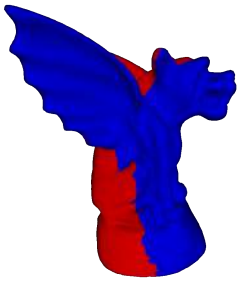
(a) Teddy segmentation (2 sub-meshes).



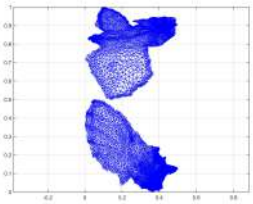
(b) Teddy parameterization (100% bijective).



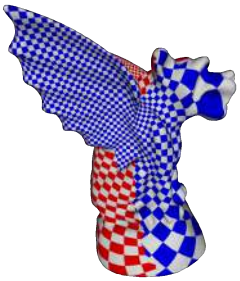
(c) Teddy chessboard texture.



(d) Gargoyle segmentation (2 sub-meshes).

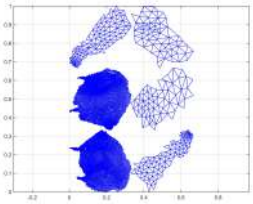


(e) Gargoyle parameterization (100% bijective).



(f) Gargoyle chessboard texture.

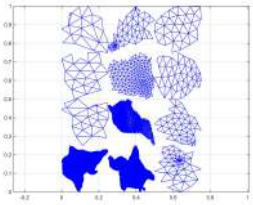
(g) Cow segmentation (6 sub-meshes).



(h) Cow parameterization (100% bijective).

(i) Cow chessboard texture.

(j) Armadillo segmentation (9 sub-meshes).



(k) Armadillo parameterization (100% bijective).

(l) Armadillo chessboard texture.

Figure IV-C.3.7: Algorithm Segmentation/Parameterization results for several datasets.

(a) Hessian - based segmentation.

(b) CONFOP texture mapping.

Figure IV-C.3.8: Armadillo's foot. Segmentation and texture mapping of mesh protrusions (toes).

Mesh segmentation / parameterization is important for the process of RE. A flaw of our algorithm lies in the fact that it produces jagged sub-mesh boundaries in most segmentations, which is sub-optimal for RE surface fitting and Boundary Representation. To overcome this problem, future work addresses: (1) boundary smoothing via geometry operations, (2) topology operations such as edge flips and triangles exchange at sub-mesh boundaries, and (3) mesh refinement in coarse boundary poly-lines.

IV-C.4

Hybrid geometry / topology based mesh segmentation for Reverse Engineering

Daniel Mejia[1,2], Oscar Ruiz-Salguero[1], Jairo R. Sánchez[2], Jorge Posada[2], Aitor Moreno[2], Carlos A. Cadavid[3]

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, Medellín, Colombia.

[2] Vicomtech, Donostia / San Sebastián, Spain.

[3] Matemáticas y Aplicaciones, Departamento de Ciencias Matemáticas, Universidad EAFIT, Medellín, Colombia.

Citation

Daniel Mejia, Oscar Ruiz-Salguero, Jairo R. Sánchez, Jorge Posada, Aitor Moreno and Carlos A. Cadavid. Hybrid geometry / topology based mesh segmentation for reverse engineering. *Computers & Graphics*, **2018**, 73, pp. 47-58. DOI: 10.1016/j.cag.2018.03.004.

Indexing: ISI (Q3), SCOPUS (Q1), Publindex (A1)

Abstract

Mesh segmentation and parameterization are crucial for Reverse Engineering (RE). Bijective parameterizations of the sub-meshes are a sine-qua-non test for segmentation. Current segmentation methods use either (1) topologic or (2) geometric criteria to partition the mesh. Reported topology-based segmentations produce large sub-meshes which reject parameterizations. Geometry-based segmentations are very sensitive to local variations in dihedral angle or curvatures, thus producing an exaggerated large number of small sub-meshes. Although small sub-meshes accept nearly

isometric parameterizations, this significant granulation defeats the intent of synthesizing a usable Boundary Representation (compulsory for RE). In response to these limitations, this article presents an implementation of a hybrid geometry / topology segmentation algorithm for mechanical workpieces. This method locates heat transfer constraints (topological criterion) in low frequency neighborhoods of the mesh (geometric criterion) and solves for the resulting temperature distribution on the mesh. The mesh partition dictated by the temperature scalar map results in large, albeit parameterizable, sub-meshes. Our algorithm is tested with both benchmark repository and physical piece scans data. The experiments are successful, except for the well - known cases of topological cylinders, which require a user - introduced boundary along the cylinder generatrices.

Indexing: Mesh Segmentation, Heat Transfer, Reverse Engineering, CAD/CAM/CAE.

IV-C.4.1 Introduction

In the context of Computer Aided Design / Manufacturing / Engineering (CAD/CAM/CAE) and the emerging Industry 4.0 framework, RE encompasses (re-)design, manufacturing, simulation, etc. [55]. Typical RE processes (Fig. IV-C.4.1): (1) tessellate the point cloud of the scanned model, (2) clean the raw triangular mesh (smoothing, filling, non-manifold repair, decimation, etc.), (3) build the Boundary Representation (B-Rep) of the mesh, (4) segment the mesh, (5) fit the resulting partition with parametric surfaces (analytic and / or freeform surfaces), (6) build the B-Rep of the reconstructed CAD model, and (7) conduct the engineering analysis. RE applications include (but are not limited to) Finite Element Analysis (FEA) [71, 72], Structural Optimization [73, 74] and Dimensional Analysis [75, 76].

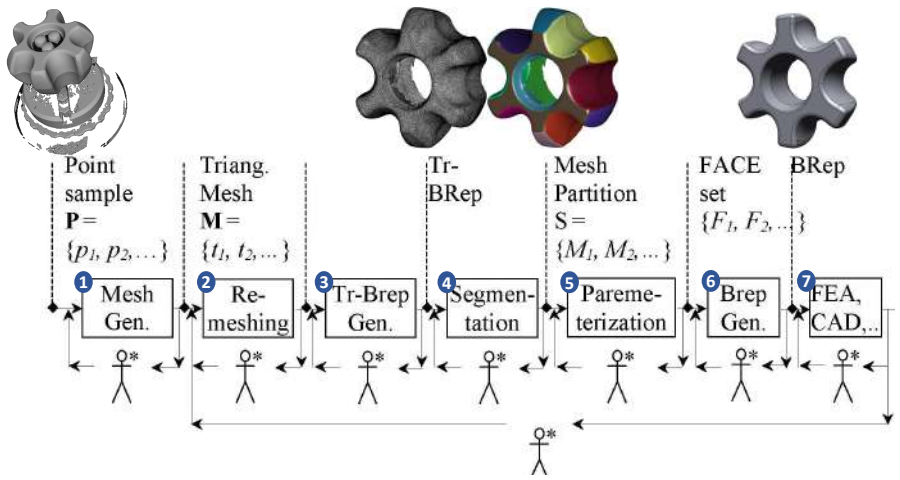


Figure IV-C.4.1: The current RE workflow is user-intensive [77]

Mesh segmentation / parameterization plays a crucial role in RE (steps 4-5) for the adequate geometric modeling of the workpiece. The mesh segmentation / parameterization problem is defined as follows:

Given: A 2-manifold triangular mesh M (or simply, "mesh") embedded in $\mathbb{R}^3$. **Goal:** (i)

to partition (i.e. segment) the triangle set M into a set of disjoint and connected sub-meshes $\{M_1, M_2, ...M_k\}$ which together compose the original mesh, and (ii) to compute a bijective parameterization $\psi_i : M_i \rightarrow \mathbb{R}^2$ for each sub-mesh M_i. The segmentation step (i) must favor the parameterizability of the computed sub-meshes while retaining feature (functional) surfaces of the scanned workpiece.

Mesh segmentation algorithms can be classified depending on the surface features used to divide the mesh:

1. **Geometry-based segmentation** captures locally geometric features of the surface (sharp edges, principal curvatures, surface normals, etc.) and partitions the surface using this information. This type of segmentation is ideal for CAD meshes that present clear sharp transitions between sub-meshes. However, geometric criterion alone applied to noisy or imperfect meshes results in over-segmentation (Fig. IV-C.4.2). If the workpiece is smooth, geometric segmentation produces large and (likely) non-parameterizable sub-meshes.

2. **Topology-based segmentation** relies on the spectra (eigenpairs) of any Laplacian operator computed on the mesh graph. This type of segmentation is common in Computer Graphics applications. However, this segmentation usually results in non-parameterizable sub-meshes (Fig. IV-C.4.3(a)).

3. **Interactive segmentation** is the most common practice by RE software (such as Geomagic® and Polyworks®, Fig. IV-C.4.3(b)). The current state-of-the-art segmentation approaches still demand expensive user interaction in order to achieve suitable segmentations for parameterization and B-Rep reconstruction (Fig. IV-C.4.1).

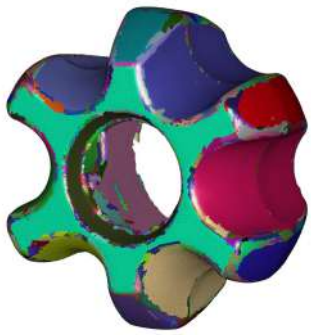
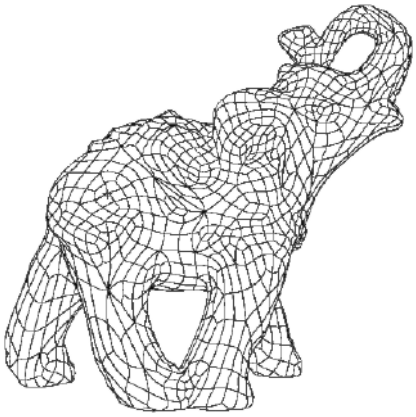


Figure IV-C.4.2: Dihedral segmentation produces over-segmentation due to surface imperfections and surface blends

Having the mesh segmented, the construction of the B-Rep becomes straightforward if a bijective parameterization of each sub-mesh is computed. A trimmed NURBS (Non-Uniform Rational B-Splines) surface can be fitted by Least Squares [78] or Radial Basis Functions (RBFs) [16] to each sub-mesh. The fitted surfaces and their trimming curves can be oriented and related to each other by their adjacency graph in order to produce the reconstructed B-Rep model [79].

(a) Topologic segmentation [77]



(b) Interactive segmentation with Geomagic® for RE [77]

Figure IV-C.4.3: Topology vs. user - based segmentations [77]

This manuscript presents a hybrid mesh segmentation / parameterization algorithm for RE, as follows: (i) A set of heat transfer equations are defined on the mesh. The topology of the mesh is captured by the Laplace-Beltrami operator inherent in the differential equation for heat transfer. (ii) Temperature constraints are imposed on a subset of vertices (mesh seeds), acting as heat sources and sinks. The local geometry of the mesh is captured by choosing the mesh seeds according to a dihedral angle criterion. (iii) To avoid over-segmentation, seeds that produce small sub-meshes are ignored. The temperature fields are used to re-compute the segmentation without these small sub-meshes. (iv) The parameterization of each sub-mesh is thereafter computed by a Hessian-based parameterization [80].

The contribution of this manuscript resides in the mixed topology (temperature) / geometry (dihedral) nature of the segmentation algorithm. Our algorithm not only pursues mesh parameterizability but also a functional partition of scanned mechanical workpieces, without resorting to over-segmentation. The algorithm allows (almost) automatic processing of 3D meshes from scanned workpieces, improving the RE workflow.

The remainder of this manuscript is organized as follows: Sect. IV-C.4.2 reviews the relevant literature. Sect. IV-C.4.3 describes the mesh segmentation algorithm. Sect. IV-C.4.4 discusses the implementation details of the algorithm. Sect. IV-C.4.5 presents and discusses results of the conducted experiments. Sect. IV-C.4.6 concludes the paper and introduces what remains for future work.

IV-C.4.2 Literature review

Mesh segmentation algorithms can be classified depending on the mesh properties used to partition the mesh as follows:

IV-C.4.2.1 Geometry-based segmentation

Geometry-based segmentation approaches compute local geometric properties (e.g. dihedral angle, curvature, frequency, [56, 77]) and use region-growing algorithms to lump property - homogeneous regions (Fig. IV-C.4.2).

Shape recognition algorithms partition the surface by matching analytic shapes to the mesh [57–59]. One of these analytic shapes (plane, sphere, cylinder or cone) is registered to each mesh vertex according to the local geometric information (such as curvature). A clustering or region growing algorithm is finally applied to compute the mesh segmentation.

Geometry-based segmentation algorithms (1) require several post-processing due to over-segmentation, (2) do not favor functional or feature segmentation, and (3) are highly sensitive to noise. Mesh smoothing may be used to reduce noise previous to segmentation [56].

IV-C.4.2.2 Topology-based segmentation

In spectral analysis, a mesh topology operator matrix (e.g. adjacency or Laplacian) is estimated on the mesh graph in order to extract and analyze its spectra (eigenpairs) [81]. A partition of the first non-constant Laplacian (Fiedler) eigenvector reflects a possible segmentation of the mesh [77, 82]. A central pre-condition for spectral methods is the edge length homogeneity through the mesh. To improve the robustness of the spectral segmentation, Refs. [83, 84] segment similar meshes simultaneously by introducing edge correspondences between meshes, while Ref. [85] captures images of the same mesh from different perspectives in order to correlate the mesh edges.

Ref. [86] computes an edge weighted Laplacian which includes information about concave regions. Chosen Laplacian eigenvectors are merged into a single scalar field whose partition segments the mesh. Ref. [65] introduces Secondary Laplacian and Giaquinta-Hildebrant operators which locally capture geometric properties (e.g. principal curvatures), thus allowing to infere 3D concavities / convexities. Ref. [61] computes the spectra of a weighted dual graph Laplacian. The dual Laplacian encodes the topology of the mesh in terms of the connectivity of the triangles (instead of the points connectivity). The weighting scheme incorporates dihedral angles, which improves the sub-mesh definition.

Heat-based algorithms are an alternative approach for topologic segmentation, defining and solving different heat transfer equations on the mesh. The topology of the mesh is captured by the Laplace-Beltrami operator, present in the heat equation. The resulting segmentation is obtained from the computed temperature fields on the mesh. Ref. [87] presents an interactive segmentation algorithm where the user draws lines perpendicular to potential sub-mesh boundaries. The algorithm defines temperature constraints according to these user strokes. The algorithm computes the constrained temperature fields and produces the segmentation based on the temperature contours.

Heat kernels are specific solutions to the heat transfer problems with unique point sources. These heat kernels can be computed by means of the eigenvectors of the Laplace-Beltrami operator [51, 88]. Refs. [89, 90] compute the heat potential (tendency to attract heat) of each mesh point in order to identify crucial heat sources which are then used to compute the heat kernels and the underlying segmentation.

In general, topology-based methods present several shortcomings: (1) they produce large submeshes which are non-parameterizable, and (2) they usually require heavy user interaction in selection of eigenpairs (spectral) or heat sources (heat-based) on the mesh, critical for the quality of the segmentation [91].

IV-C.4.2.3 Mesh segmentation in RE

RE workflow currently requires intensive, costly user input (Fig. IV-C.4.1). Commercial tools include PolyWorks® [73], RapidWorks® [75] and Geomagic® [77]. Refs. [71, 92] apply RE to run FEA on scanned turbine blades. The turbine blades are manually divided into sections prior to digitizing. Ref. [79] uses the dihedral angle and curvature scalar fields on the mesh to segment it, seeking to optimally fit analytic shapes (sphere, cylinder, cone, etc.). Refs. [78, 93] fit freeforms to growing sub-meshes, with Ref. [93] favoring rectangular ones. A common approach to represent an unknown model is to fit rectangular NURBs patches to the whole mesh [93]. These small NURBs patches have the advantage to produce low-distortion parameterizations, even in the case of complex geometries where such parameterization can be optimized to produce the smallest distortion [94,95]. However, such patches usually lack from the functional information of the source CAD model (see Fig. IV-C.4.3(b)).

IV-C.4.2.4 Conclusions of the literature review

Current state-of-the-art segmentation algorithms are not fully suitable for RE applications. Geometry-based segmentation algorithms produce over-segmentation on scanned workpieces due to surface imperfections and surface blends between sub-meshes. On the other hand, topology-based algorithms result in parameterization - hostile segmentations. Therefore, the current RE workflow demands massive user input in order to produce usable B-Reps, requiring between 25-150 hours of interactive work for a single scanned workpiece [73, 77].

To overcome these problems, this article presents an automatic mesh segmentation algorithm for RE: (1) Our algorithm defines several constrained heat transfer problems on the mesh for segmentation. Temperature constraints are located automatically using a dihedral criterion. To avoid over-segmentation, constraints that produce small sub-meshes are removed. Therefore, our algorithm favors sub-meshes parameterizability by capturing local geometric features (dihedral angle) and avoids over-segmentation by capturing topological mesh features (temperature fields). (2) The sub-meshes are parameterized with a Hessian-based parameterization algorithm [80]. Results are presented for meshes collected from a 3D optical scanner and public benchmarks.

IV-C.4.3 Methodology

To compute the segmentation of the mesh M, we extend the heat-based approach presented in Ref. [87], making the segmentation procedure completely automatic (in the sense that it does not require user interaction) as follows (Fig. IV-C.4.4): (1) instead of manually selecting a heat source and a heat sink to split the mesh into two sub-meshes, our segmentation algorithm locates simultaneously the set of all heat sources and sinks S. Such heat sources and sinks (mesh seeds) are located automatically by a dihedral angle criterion which captures the local geometry of the mesh. (2) These heat sources / sinks are used to define a set of heat transfer differential equations on the whole mesh. Therefore, for each heat source $S_i \subset S$, a mesh temperature field u_i is found. (3) The computed temperature fields are compared for each vertex in order to define a unique pre-segmentation of M. (4) Seeds that produce small sub-meshes are removed to avoid over-segmentation, resulting in a new set of temperature fields. (5) Finally, these new temperature fields define the final segmentation of M. In addition, an almost automatic parameterization algorithm proceeds as follows: (6) Artificial boundaries are manually (interactively) introduced only in the

case of cylinder-like sub-meshes to allow their parameterization. (7) The parameterization of each sub-mesh is computed with a Hessian-based parameterization algorithm [80].

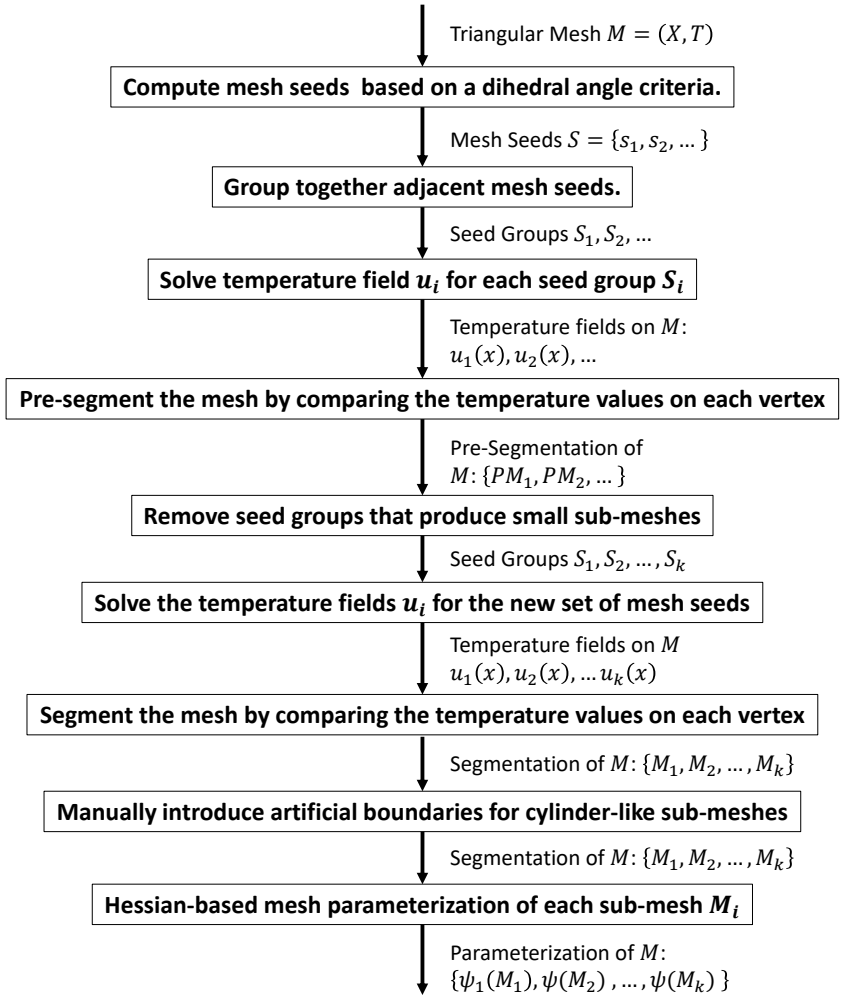


Figure IV-C.4.4: Overall scheme of the segmentation algorithm

IV-C.4.3.1 Automatic placement of mesh seeds

One of the crucial requisites in mesh parameterization resides in the parameterizability of the resulting segmentation. Such parameterizability is hindered by high frequency zones and favored by low frequency zones. Our algorithm locates a set of mesh seeds S in the low frequency neighborhoods of the mesh. These mesh seeds will expand the different sub-meshes of the segmentation by

propagating heat through the whole mesh (discussed in subsequent sections). We identify such low frequency zones by a dihedral angle criterion as follows:

1. Set a dihedral angle threshold $\theta_{threshold} \to 0$
2. For each vertex $x_i \in M$:
   (a) Compute the incident edges $E_i = e_1, e_2, \ldots$ on x_i.
   (b) Compute the dihedral angle θ_j of each incident edge $e_j \in E_i$.
   (c) If $\pi - \theta_j > \theta_{threshold}$ (for any incident edge e_j), then skip the current vertex.
   (d) Else, insert the current vertex x_i in the list of the mesh seeds S.

A vertex is considered as a low frequency vertex if and only if none of its incident edges is sharp (Fig. IV-C.4.5). An edge is sharp (non-planar) if $\pi - \theta_j$ is larger than the dihedral threshold $\theta_{threshold}$. Therefore, $\theta_{threshold} \to 0$ can be seen as the maximum non-coplanarity between two adjacent triangles in a low frequency zone.

(a) Low frequency vertex

(b) High frequency vertex with a sharp incident edge

Figure IV-C.4.5: Examples of low and high frequency vertex for selection of mesh seeds based on a dihedral criterion

Our algorithm ensures that adjacent low frequency mesh vertices lie in the interior of a common sub-mesh by grouping them into a subset of mesh seeds $S_i \subset S$ (Fig. IV-C.4.6(a)).

IV-C.4.3.2 Heat transfer with temperature constraints

The following partial differential equation describes the steady heat transfer phenomenon without heat sources on the mesh M:

$$\Delta u(x) = 0 \tag{IV-C.4.1}$$

where Δ is the Laplace-Beltrami operator and $u(x)$ is the temperature distribution along the surface. We impose a temperature value $(u(S_i) = 1)$ on a subset of mesh seeds (heat sources) $S_i \subset S$, and temperature value $(u(S_j) = 0)$ at the remaining seed sets (heat sinks) $S_j \subset M, i \neq j$. Each subset of sources S_i will define a sub-mesh M_i of the segmentation. Therefore, for each sub-mesh M_i, the

(a) Mesh seeds on low frequency zones grouped by color

(b) Temperature solution for one seed group

Figure IV-C.4.6: Mesh seeds are located at low frequency zones. Each seed group defines a temperature field on the mesh.

following constrained heat problem arises:

$$\begin{aligned} & \Delta u_i(x) = 0 \\ s.t. & \\ & u_i(S_i) = 1, \\ & u_i(S_j) = 0, \qquad i \neq j \end{aligned} \tag{IV-C.4.2}$$

For each heat source S_i, its corresponding temperature field $u_i(x)$ is obtained by propagating the thermal energy through the whole mesh M (Fig. IV-C.4.6(b)). The temperature solution $u_i(x)$ is directly related to the sub-mesh M_i, achieving maximum value ($u = 1$) at the defined heat sources S_i and minimum value ($u = 0$) at the remaining heat sinks $S_j, i \neq j$.

The FEA Method is implemented to estimate Δ numerically. Therefore, Δ is approximated by the FEA matrix $\mathbf{L}$, defined as [52, 96]:

$$\mathbf{L}_{ij} = \begin{cases} \frac{3}{A_i} w_{ij}, & \text{if } (x_i, x_j) \text{ is an edge of } M \\ -\frac{3}{A_i} \sum_{x_k \in N_i} w_{ik}, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{IV-C.4.3}$$

where N_i is the neighborhood of x_i, $w_{ij} = \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2}$ is the cotangent weight of edge (x_i, x_j), α_{ij} and β_{ij} are the angles opposite to edge (x_i, x_j), and A_i is the area of all the triangles incident to vertex x_i. An $n \times n$ linear system of equations $\mathbf{A}U_i = B_i$ arises for each heat source S_i, with:

$$\mathbf{A} = \begin{bmatrix} \mathbf{L} \\ \mathbf{I_{S_i}} \\ \mathbf{I_{S_j}} \end{bmatrix}, \text{ and } B_i = \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix} \tag{IV-C.4.4}$$

where $\mathbf{L}$ is the FEA matrix for all the nodes with unknown temperature in M (i.e. rows associated to mesh sources and sinks are excluded from $\mathbf{L}$), $\mathbf{I_{S_i}}, \mathbf{I_{S_j}}$ are the constraints matrices for the current heat sources S_i and heat sinks S_j, respectively (i.e. entry kl of matrix $\mathbf{I_S}$ is 1 if constraint k fixes the temperature for the heat source / sink x_l, 0 otherwise).

Our algorithm simultaneously solves several heat transfer problems (one for each group of seeds S_i). The matrix $\mathbf{A}$ is common to all of them and it is computed and prefactored once. The linear system defined by Eq. (IV-C.4.4) is then solved using sparse Cholesky factorization, which in most cases can be solved in nearly linear time $\mathcal{O}(n)$ [97, 98]. The following section describes how to combine the different temperature fields to obtain a single segmentation field.

IV-C.4.3.3 Heat-based mesh segmentation

At this point, each vertex x_i in the mesh has an associated set of temperature values $u_1(x_i), u_2(x_i), \cdots$ from the temperature fields generated by each set of mesh seeds $S_1, S_2, \ldots$. The segmentation of M is achieved by computing the maximum temperature value at each vertex and its corresponding seed group. Thus, each sub-mesh M_i is composed by the subset of vertices whose maximum temperature is $u_i(x)$:

$$M_i = \{x_k \in M \mid u_i(x_k) > u_j(x_k), i \neq j\} \tag{IV-C.4.5}$$

This construction guarantees that the set of heat sources S_i belongs to the sub-mesh M_i, assigning low frequency areas to the same sub-mesh. Heat propagates smoothly from these zones to higher frequency zones, defining the sub-mesh boundaries.

IV-C.4.3.4 Discarding small seed groups

In RE, the mesh M presents surface imperfections due to manufacturing imperfections and / or RE pre-processing results (such as data acquisition, surface meshing, mesh filtering, mesh decimation, etc). Such imperfections and mesh noise produce small groups of seeds that lead the heat algorithm to an over-segmentation of the surface. A second heat - based segmentation is then executed excluding noise - originated seeds. An overview of the method follows:

1. Locate the initial heat seeds on low frequency neighborhoods.
2. Find the mesh temperature fields and segment accordingly.
3. Compute the area of each sub-mesh.
4. Given the sub-mesh with the largest area $A_{largest}$, locate all the sub-meshes with an area below $\varepsilon \cdot A_{largest}$ (small sub-meshes).
5. Discard seeds in small sub-meshes.
6. Re-compute the temperature fields with the surviving seeds.
7. Re-compute the segmentation with the new temperature fields.

The area percentage parameter $0 \leq \varepsilon \leq 1$ measures the minimum sub-mesh size (relative to the largest sub-mesh) allowed by the segmentation. Triangles belonging to small sub-meshes in the over-segmentation are appended to the largest sub-meshes by temperature propagation as discussed in Sect. IV-C.4.3.3.

IV-C.4.3.5 Segmentation of cylinder-like sub-meshes

In the well known case of cylinder-like sub-meshes, our segmentation algorithm produces parameterization - hostile surfaces. However, such surfaces can be made parameterizable by manually making a generatrix of the cylinder a mesh boundary (Fig. IV-C.4.7), as follows:

1. The user selects the start and end vertices of the generatrix.
2. A shortest path (Dijkstra) algorithm computes the path that links the start and end vertices.
3. Our algorithm generates a new B-Rep of the sub-mesh introducing the computed trajectory as sub-mesh boundary.

(a) Parameterization - hostile cylinder-like sub-mesh

(b) Parameterizable cylinder-like sub-mesh

Figure IV-C.4.7: To parameterize cylinder-like sub-meshes, an artificial boundary (red) is manually introduced using a cylinder generatrix

Other authors have addressed the problem of fitting closed cylinders using least squares minimization [78, 79]. However, our approach comprises not only standard cylinders but also their topological equivalents (with and without holes).

IV-C.4.3.6 Hessian-based mesh parameterization

To compute the parameterization $\psi \subset \mathbb{R}^2$ of M, a Hessian-based mesh parameterization algorithm [80] is applied on each sub-mesh M_i. This Hessian mesh parameterization algorithm applies the main concepts of Hessian Locally Linear Embedding (HLLE)[69] (a Dimensional Reduction algorithm) on triangular meshes.

According to [69, 80], a parameterization of M_i is given by the first 2 non-constant eigenvectors of the Hessian functional $\mathcal{H}$, defined as:

$$\mathcal{H}f = \int_{M_i} \|\mathbf{H}^{\mathbf{tan}}_{\mathbf{x}} f\|^2_F \mathrm{dA} \approx \mathbf{f^T K f} \tag{IV-C.4.6}$$

where $f \in \mathcal{C}^2(M_i)$ is a smooth function defined on M_i, $\|\cdot\|_F$ is the Frobenius norm, dA is the surface differential, $\mathbf{f} = \{f_1, f_2, \cdots, f_n\}$ are the values of $f(x_j)$ at each vertex $x_j \in M_i$, and $\mathbf{K} =$

$(\mathbf{K}_1 + \mathbf{K}_2 \cdots, \mathbf{K}_n)$ is the discrete Hessian estimator (matrix). This matrix is semidefinite-positive [69,80]. Therefore, the parameterization $\psi_i(M_i)$ is extracted by computing the first two eigenvectors of $\mathbf{K}$ with the smallest non-zero eigenvalue. Such eigenvectors correspond to an orthogonal basis for all linear functions (and as a consequence, a basis for all parameterizations) defined on M_i.

IV-C.4.4 Implementation of the algorithm

To test our algorithm in a real RE context, different engineering pieces have been scanned with an optical 3D scanner (Fig. IV-C.4.8). The RE result for these pieces is used in real engineering contexts. The optical 3D scanner produces point cloud data for each workpiece. Fig. IV-C.4.9 plots the datasets obtained by scanning (a) a knob, (b) a tripod joint, and (c) a rocker arm base. These datasets were user - processed in Geomagic® Design™ to ensure manifold properties (pre-condition for segmentation and parameterization).

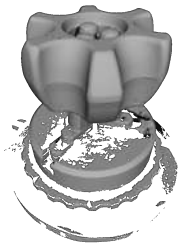
Figure IV-C.4.8: Acquisition of 3D point cloud data through an optical 3D scanner

Fig. IV-C.4.10 plots the resulting meshes after the interactive processing. Large holes have been left in the mesh. These meshes are the actual input for our segmentation algorithm.

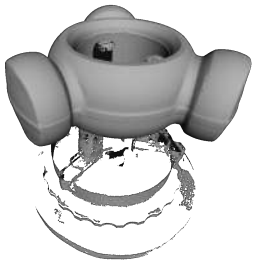
Table IV-C.4.1: Default parameter values for our segmentation algorithm

Parameter	**Value**
$\theta_{threshold}$	$\frac{1}{20}\pi$ radians
ε	5%

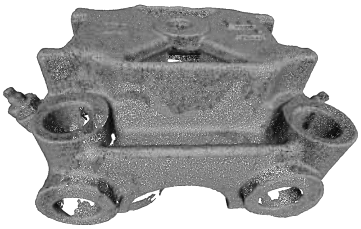
Fig. IV-C.4.11 plots the seed groups processing for the Knob mesh. The initial mesh seeds are computed with a $\theta_{threshold} = \frac{1}{20}\pi$ radians (see Table IV-C.4.1), since this value has shown to consistently capture flat zones in all of our experiments. Several small seed groups arise due to isolated low frequency points inside high frequency zones (Fig. IV-C.4.11(a)). The temperature-based segmentation using these seeds results in an over-segmentation of the surface (Fig. IV-C.4.11(b)). Small (noise-generated) sub-meshes are then discarded by the algorithm (as discussed in Sect. IV-C.4.3.4), no longer receiving heat seeds and therefore being absorbed in natural form by large meshes when the heat algorithm is run again. After applying seeds processing, the remaining



(a) Knob (17.3m points)



(b) Tripod joint (23.6m points)

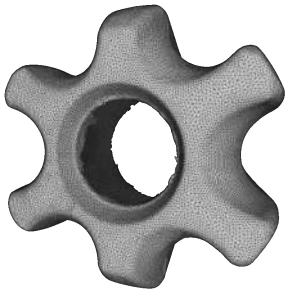


(c) Rocker arm base (3.1m points)

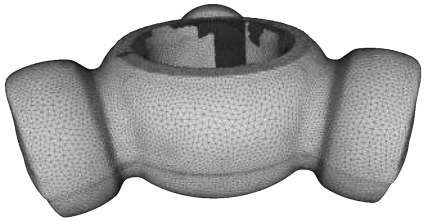
Figure IV-C.4.9: Datasets (point clouds) obtained with an optical scanner

seeds capture the local flat geometry of the mesh and the high frequency seeds dissapear (Fig. IV-C.4.11(c)). The final temperature-based segmentation preserves the geometric properties from the dihedral criterion in low frequency zones while producing a smooth transition between sub-meshes, avoiding mesh over-segmentation (Fig. IV-C.4.11(d)). Fig. IV-C.4.12 plots the distribution of the sub-mesh sizes (sorted by surface area) and the area threshold used to discard small sub-meshes. In all our conducted experiments we choose an area threshold parameter of $\varepsilon = 5\%$ (see Table IV-C.4.1) as we have identified that it consistently differentiates large (albeit parameterizable) sub-meshes (Fig. IV-C.4.11(d)) from small (noise-generated) ones (Fig. IV-C.4.11(b)). The initial segmentation of the knob produces 300 sub-meshes while the final segmentation produces only 15 sub-meshes. Decreasing the value of ε in Fig. IV-C.4.12 would increase the likelihood of over-segmentation. On the other hand, increasing its value could lead the algorithm to merge large sub-meshes and produce non-parameterizable segmentations. The user may, of course, change the cutting value (upon examination of the distribution exemplified in Fig. IV-C.4.12), reinforcing or decreasing the absorption of small sub-meshes into the larger ones.

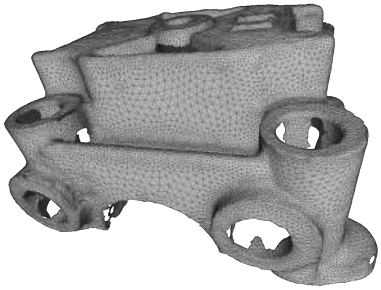
Fig. IV-C.4.13 plots the final segmentation results for each mesh. The computed sub-meshes present low frequencies while sub-mesh boundaries are located in high frequency zones. The segmentation is controlled by the area percentage parameter ε (taken as $\varepsilon = 5\%$ in all our experiments), discarding noise - related sub-meshes as discussed in Sect. IV-C.4.3.4. Our algorithm produces parameterization - friendly segmentations while keeping a relatively low number of sub-meshes (15 sub-meshes for the knob - Fig. IV-C.4.13(a), 13 for the tripod joint - Fig. IV-C.4.13(b), and 27 for the rocker arm base - Fig. IV-C.4.13(c), respectively.).



(a) Knob (57.8 faces)

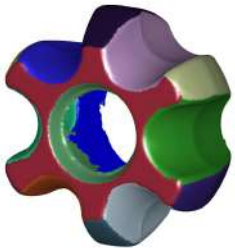


(b) Tripod joint (61.0k faces)

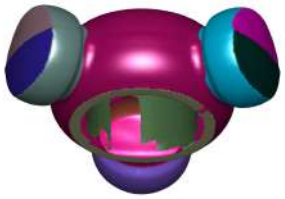


(c) Rocker arm base (36.1k faces)

Figure IV-C.4.10: Input meshes for our segmentation algorithm. These meshes are the result of manual preprocessing with commercial software (Geomagic®).



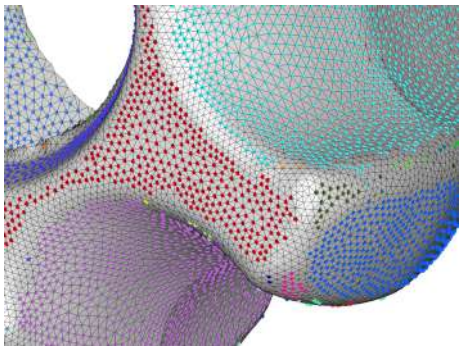
(a) Knob (15 sub-meshes)



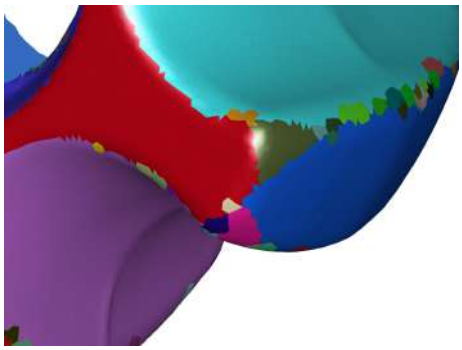
(b) Tripod joint (13 sub-meshes)

(c) Rocker arm base (27 sub-meshes)

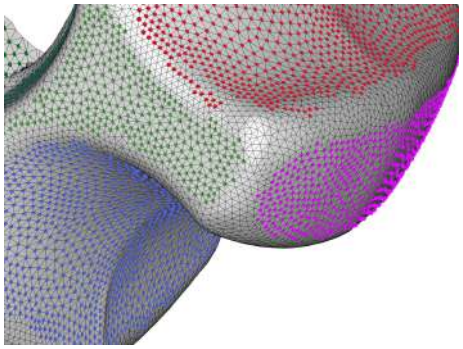
Figure IV-C.4.13: Temperature-based segmentation. The dihedral criterion captures the local mesh geometry while the temperature approach produces smooth transitions between sub-meshes.



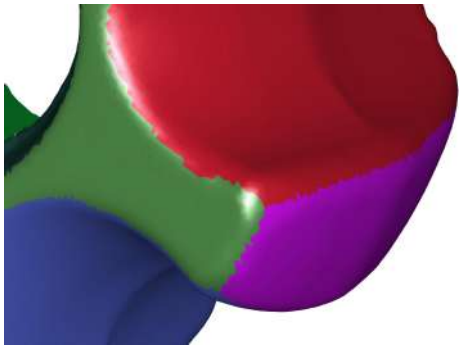
(a) (Raw) Knob seeds



(b) Knob temperature-based segmentation (raw seeds)



(c) (Processed) Knob seeds



(d) Knob temperature-based segmentation (processed seeds)

Figure IV-C.4.11: Discarding mesh seeds. The initial seed groups produce over-segmentation due to surface imperfections (a-b). After discarding small seed groups, such over-segmentation is removed (c-d).

Hessian parameterization is then applied on each sub-mesh. Fig. IV-C.4.14 plots the 2D parameterization of each of the knob sub-meshes. Such parameterization is completely bijective (i.e. no triangle flips nor surface overlaps occur in the parametric space). Fig. IV-C.4.15 plots the chessboard textures applied on the resulting segmentation using the computed Hessian parameterization. The distortion of the chessboard squares represents the distortion of the computed parameterization. In the case of the tripod joint (Fig. IV-C.4.15(b)), artificial boundaries have been introduced manually on the cylinder-like sub-meshes as dicussed in Sect. IV-C.4.3.5. Fig. IV-C.4.15 displays (using chessboard textures) the computed bijective Hessian parameterizations of the sub-meshes. The special segmentation case of topological cylinders (e.g. tripod joint, Fig. IV-C.4.15(b)) currently requires the manual creation of a boundary along a cylinder generatrix (discussion in Sect. IV-C.4.3.5).

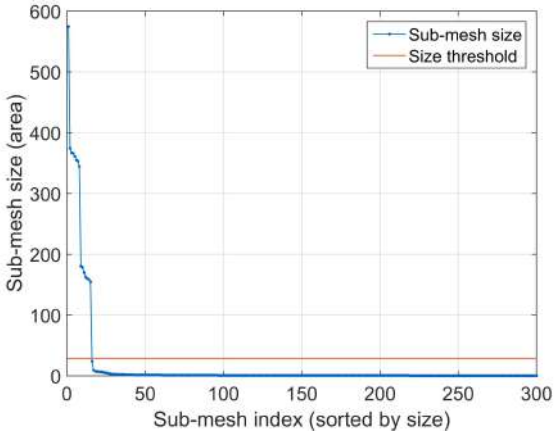


Figure IV-C.4.12: Sub-mesh sizes for the initial segmentation of the knob mesh. The red line plots the area threshold ($\varepsilon = 5\%$) used to discard small sub-meshes from the final segmentation.

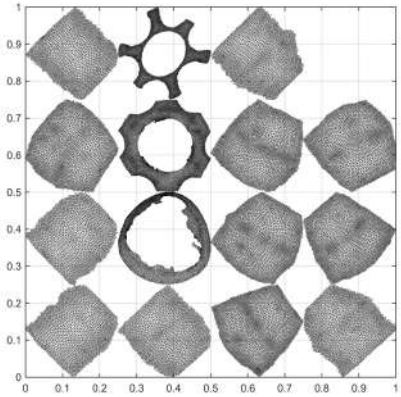
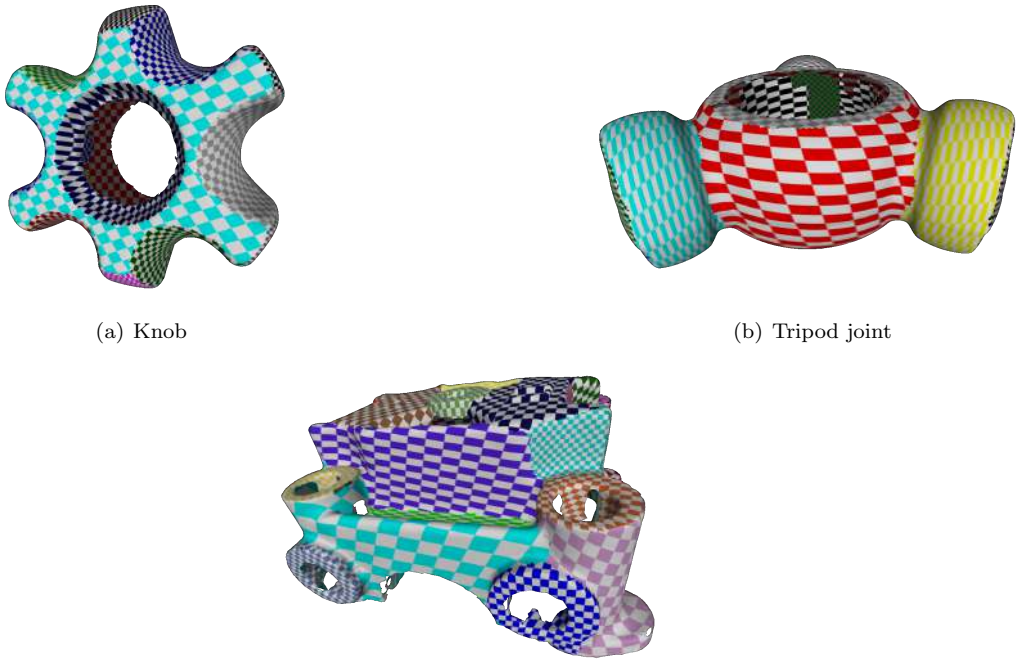


Figure IV-C.4.14: Knob Hessian parameterization

IV-C.4.5 Results and benchmarking

This section presents a comparison of our segmentation algorithm against several state-of-the-art algorithms and commercial CAD software. Sect. IV-C.4.5.1 presents a standard benchmarking using datasets and algorithms from the National Design Repository [99] and the Princeton Benchmark Repository [70], which are standard in the mesh segmentation literature. Afterwards, Sect. IV-C.4.5.2 compares our algorithm against recent algorithms from the literature and some commercial software using our in-house scanned pieces (introduced in Sect. IV-C.4.4).



(a) Knob

(b) Tripod joint

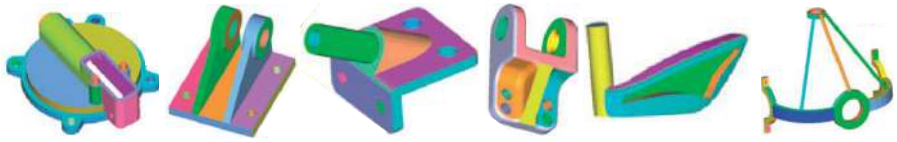
(c) Rocker arm base

Figure IV-C.4.15: Chessboard texture applied on each sub-mesh. The resulting parameterization is bijective for all the scanned models.

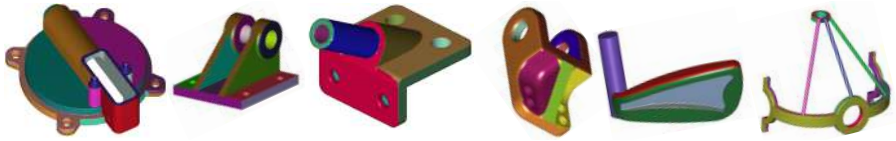
IV-C.4.5.1 Standard benchmarking

Fig. IV-C.4.16 plots our segmentation results compared with the Cross Boundary Brushes algorithm results [87] for some CAD models from the National Design Repository [99]. Both methods use a heat-based approach to capture geometric features of each CAD model. However, Cross Boundary Brushes is completely interactive, requiring user input for each computed sub-mesh. CAD models usually present several geometric features which require moderate segmentation sizes (> 10 sub-meshes). Therefore, interactive user input may become unreliable in such cases. In contrast, our algorithm produces similar segmentation results and parameterizable sub-meshes without requiring any user input.

Fig. IV-C.4.17 plots segmentation results of our algorithm and some automatic algorithms from the Princeton Benchmark [70]. Our algorithm is able to capture the geometric features of the surface for the flange dataset (Fig. IV-C.4.17(d)) while other algorithms struggle to capture such features, grouping different surfaces (such as the cylinders, cones and the plane on the flange orifices) into the same sub-mesh (Figs. IV-C.4.17(a) - IV-C.4.17(c)). As a consequence, our algorithm produces more sub-meshes (21) than the benchmark algorithms (< 10), which in a RE context is preferable to allow easy parameterization of each of the flange sub-meshes (see Table IV-C.4.2). On the other hand, our segmentation of the cup dataset (Fig. IV-C.4.17(d)) results in a similar number of sub-meshes (see Table IV-C.4.2), and it is in agreement with the rest of the benchmarking algorithms



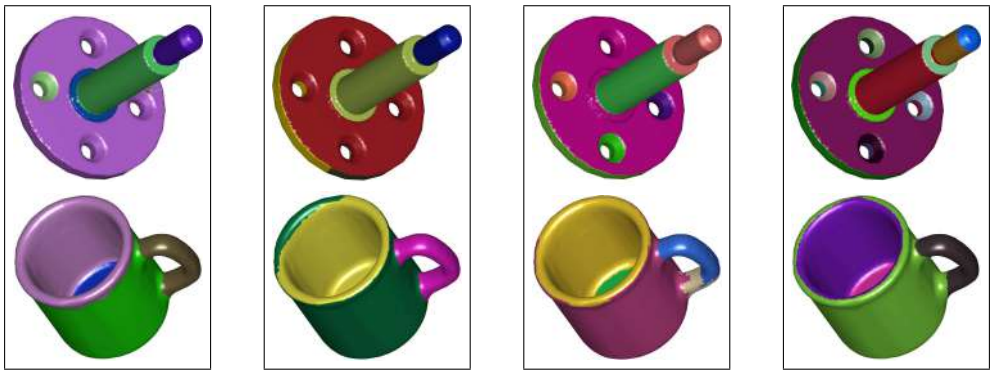
(a) Cross Boundary Brushes [87] (fully interactive)



(b) Our method (automated)

Figure IV-C.4.16: Standard benchmarking. Segmentation results of the Cross Boundary Brushes method [87] (above) vs. our automated method (below). Meshes from the National Design Repository [99].

(Figs. IV-C.4.17(a) - IV-C.4.17(c)), correctly segmenting the cup model into its meaningful parts (Fig. IV-C.4.17(d)).



(a) Random Cuts [100] (b) Random Walks [101] (c) Fitting Primitives [102] (d) Our algorithm

Figure IV-C.4.17: Standard benchmarking. Segmentation results for the flange and cup models. Meshes from the Princeton Benchmark [70].

Table IV-C.4.2: Number of sub-meshes for the segmentation results of the Princeton Benchmark

Algorithm \Dataset	Flange	Cup
Random Cuts [100]	7	4
Random Walks [101]	5	3
Fitting Primitives [102]	8	6
Our (Temp-Geom) algorithm	21	5

Our segmentation algorithm is designed to work on scanned meshes of mechanical pieces. As

a consequence, our algorithm behaves unexpectedly if applied to organic meshes. Fig. IV-C.4.18 illustrates this fact by applying our algorithm to a human mesh. The result is a bad segmentation with features not being characterized by our algorithm (such as head, hands or leg), and also each sub-mesh is non-parameterizable. Despite of the topology (heat-based) component of the algorithm, such a result is mainly due to the dihedral-criterion used to place the temperature seeds on the mesh (see Sect. IV-C.4.3.1). This problem can be addressed by changing the approach to define these seeds, which is left for future work.

Figure IV-C.4.18: Non-parameterizable segmentation of an organic mesh with our algorithm. Human mesh from the Princeton benchmark [70].

IV-C.4.5.2 RE benchmarking

Fig. IV-C.4.19 plots the segmentation results of the scanned mechanical workpieces (introduced in Sect. IV-C.4.4) using state-of-the-art segmentation techniques. Fig. IV-C.4.19(a) plots the segmentation result using our implementation of the Contour Based automatic algorithm [56]. The resulting segmentation captures some of the surface features of the tripod joint and rocker arm meshes. However, sub-mesh boundaries are non-smooth and do not capture the real boundaries of the workpiece surfaces. The number of sub-meshes is relatively low (see Table IV-C.4.3) for each segmented piece, grouping several feature surfaces of the workpiece in the same sub-mesh, which difficults the parameterization step of the RE process. The segmentation of the knob mesh is undesirable in the context of RE.

Figs. IV-C.4.19(b)-IV-C.4.19(c) plot the automatic segmentation results of the scanned workpieces using commercial CAD software. The Autodesk® 3ds Max® result is able to locate the different feature surfaces of the CAD meshes. However, it produces an excessive amount of sub-meshes (> 1000, see Table IV-C.4.3) which are for the most part product of mesh noise and blending surfaces. On the other hand, the Geomagic® Design™ result captures not only feature surfaces but also blending surfaces (which dictate smooth transitions between feature surfaces) while ignoring the mesh noise. Such a result is highly desirable in a RE context to reconstruct the analytic surfaces of the scanned model. However, these blending surfaces can produce over-segmentation at some degree as illustrated in the rocker arm of Fig. IV-C.4.19(c), which has 115 sub-meshes.

Our algorithm solves this problem by merging the blending surfaces into the feature surfaces (Fig. IV-C.4.19(d)), reducing this number to 27 [16] while keeping the segmentation parameterizable.

Table IV-C.4.4 presents the main advantages and disadvantages of all the segmentation algorithms used in this manuscript. Our algorithm provides an automatic alternative to mesh segmentation of mechanical pieces for RE, avoiding over-segmentation even in the presence of blending surfaces and mesh noise (natural to scanning devices and manufacture defects). It is worth to note that in the general context of mesh segmentation, an algorithm is considered to be automatic if it does not require interactive input of the user to compute the result. However, it is very common for automatic algorithms (including ours, see Table IV-C.4.1) to require the use of at least one input parameter (prior to segmentation) which is used by the algorithm to internally perform numerical decisions during the segmentation.

(a) Implementation of Contour Based Segmentation [56]

(b) Automatic Autodesk® 3ds Max® segmentation

(c) Automatic Geomagic® Design™ segmentation

(d) Our algorithm

Figure IV-C.4.19: Comparison of results: (a) state-of-the-art competitor [56], (b-c) commercial tools, (d) our algorithm. Datasets: in-house scanned mechanical pieces.

Table IV-C.4.3: Number of sub-meshes for the segmentation results of our scanned models using state-of-the-art algorithms and commercial software

Algorithm \Dataset	Knob	Tripod Joint	Rocker Arm Base
Contour Based Segmentation [56]	6	9	11
Autodesk® 3ds Max®	6216	5633	11772
Geomagic® Design™	40	26	115
Our (Temp-Geom) algorithm	15	13	27

Table IV-C.4.4: Advantages and disadvantages of each segmentation algorithm

Algorithm	**Advantages**	**Disadvantages**
Cross Boundary Brushes [87]	1. Works on both mechanical and organic meshes 2. Smooth sub-mesh boundaries	1. Non-automatic (requires heavy user interaction)
Random Cuts [100]	1. Avoids over-segmentation 2. Smooth sub-mesh boundaries	1. Non-automatic (requires user interaction) 2. Non-parameterizable sub-meshes
Random Walks [101]	1. Automatic segmentation 2. Works on both mechanical and organic meshes 3. Avoids over-segmentation 4. Smooth sub-mesh boundaries	1. Non-parameterizable sub-meshes
Fitting Primitives [102]	1. Automatic segmentation 2. Smooth sub-mesh boundaries 3. Parameterizable sub-meshes for RE	1. Does not work properly on organic meshes 2. Does not work properly on mechanical meshes composed by several freeform surfaces
Contour Based Segmentation [56]	1. Works on both mechanical and organic meshes 2. Avoids over-segmentation	1. Non-parameterizable sub-meshes 2. Non-smooth sub-mesh boundaries
Autodesk® 3ds Max®	1. Automatic segmentation 2. Parameterizable sub-meshes	1. Does not work properly on organic meshes 2. Over-segmentation
Geomagic® Design™	1. Automatic segmentation 2. Smooth sub-mesh boundaries 3. Parameterizable sub-meshes for RE	1. Does not work properly on organic meshes 2. Over-segmentation on meshes with a lot of small features (such as blending surfaces)
Our (TempGeom) algorithm	1. Automatic segmentation 2. Avoids over-segmentation 3. Smooth sub-mesh boundaries 4. Parameterizable sub-meshes for RE	1. Does not work properly on organic meshes 2. Ignores small feature surfaces

IV-C.4.6 Conclusions and future work

This manuscript presents an algorithm for automatic mesh segmentation of 3D meshes of digitized mechanical pieces for RE applications. The implemented algorithm articulates a dihedral / heat transfer-based segmentation with a Hessian-based parameterization.

Compared to similar approaches, our method improves the RE workflow with an automatic

hybrid geometry / topology approach which segments triangular meshes acquired from scanned mechanical models. The geometric component of the algorithm (i.e. dihedral criterion) favors the parameterizability of the resulting partition. On the other hand, the topologic component (captured by the temperature fields) favors smooth transitions between sub-meshes and avoids over-segmentation. The experiments were conducted on data acquired by a 3D optical scanner and from public repositories, and yet resulted in sets of fully parameterizable sub-meshes.

Ongoing work addresses: (1) Detection of cylinder-like sub-meshes and automatic computation of virtual boundaries as our algorithm currently requires user guidance to parameterize such cases. (2) Design of an alternative method to compute temperature seeds in the mesh to allow the segmentation of organic meshes. (3) Triangle negotiation / splitting between adjacent sub-meshes in order to produce smoother sub-mesh boundary curves, in preparation of cleaner B-Reps. (4) Definition of a consistent topology (SHELL, FACES, LOOPS, EDGES and VERTICES) and geometry (freeform curves and surfaces) which together compose the final B-Rep of the reconstructed model.

IV-C.5

Computational Geometry Techniques in Medical (Dentistry) Applications

Daniel Mejia-Parra[1,2], Aitor Moreno[2], Asier Lazcano[3], Luis Saracho[3], Oscar Ruiz-Salguero[1], Carlos A. Cadavid[1] and Jorge Posada[2]

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, Medellín, Colombia.

[2] Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), San Sebastián, Spain.

[3] BTI Biotechnology Institute, Vitoria, Spain

Context

This work is CONFIDENTIAL at the time of publication of this Thesis. The developments in this area are being commercially exploited by the sponsors, thus currently banning this research from dissemination. For more information, please refer to the College of Engineering, Universidad EAFIT.

IV-D

Thermal Simulation of CNC Laser Machining

IV-D.1

Frequency-domain analytic method for efficient thermal simulation under curved trajectories laser heating

Daniel Mejia-Parra[1,2], Aitor Moreno[2], Jorge Posada[2], Oscar Ruiz-Salguero[1], Iñigo Barandiaran[2], Juan Carlos Poza[3], Raúl Chopitea[3]

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, K49 #7 Sur-50, Medellín (Colombia)

[2] Vicomtech, Mikeletegi Pasealekua, 57, Donostia - San Sebastián (Spain)

[3] Lantek Investigación y Desarrollo, Parque Tecnológico de Alava - Ferdinand Zeppelin 2, 01510 Miñano, Álava (Spain)

Mathematics and Computers in Simulation
Volume 166, December 2019, Pages 177-192

Citation

Daniel Mejia-Parra, Aitor Moreno, Jorge Posada, Oscar Ruiz-Salguero, Iñigo Barandiaran, Juan Carlos Poza and Raúl Chopitea. Frequency-domain analytic method for efficient thermal simulation under curved trajectories laser heating. *Mathematics and Computers in Simulation*, **2019**, 166. DOI: 10.1016/j.matcom.2019.05.006.

Indexing: ISI (Q2), SCOPUS (Q2), Publindex (A1)

Abstract

In the context of Computer Simulation, the problem of heat transfer analysis of thin plate laser heating is relevant for downstream simulations of machining processes. Alternatives to address the

problem include *(i)* numerical methods, which require unaffordable time and storage computing resources even for very small domains, *(ii)* analytical methods, which are less expensive but are limited to simple geometries, straight trajectories and do not account for material non-linearities or convective cooling. This manuscript presents a parallel efficient analytic method to determine, in a thin plate under convective cooling, the transient temperature field resulting from application of a laser spot following a curved trajectory. Convergence of both FEA (Finite Element Analysis) and the analytic approaches for a small planar plate is presented, estimating a maximum relative error for the analytic approach below 3.5% at the laser spot. Measured computing times evidence superior efficiency of the analytic approach w.r.t. FEA. A study case, with the analytic solution, for a large spatial and time domain ($1\,m \times 1\,m$ and $12\,s$ history, respectively) is presented. This case is not tractable with FEA, where domains larger than $0.05\,m \times 0.05\,m$ and $2\,s$ require high amounts of computing time and storage.

Keywords: Heat Transfer, Laser Heating, Analytic Solution, Efficient Simulation, Parallel Computing, Thin Plate.

Glossary

FEA	Finite Element Analysis.
a, b, Δz	Width (m), height (m) and thickness (m) of the plate.
$\mathbf{x}$, t	Spatial $\mathbf{x} = [x, y]$ $([m, m])$ and temporal $t \geq 0$ (s) coordinates.
$u = u(\mathbf{x}, t)$	Temperature distribution along the plate at a given time (K).
ρ	Plate density (kg/m^3).
c_p	Plate specific heat $(J/(kg \cdot K))$.
k	Plate thermal conductivity $(W/(m \cdot K))$.
R	Plate reflectivity i.e., portion of the laser energy that is not absorbed by the plate $(0 \leq R \leq 1)$.
$q = q(u)$	Heat loss due to convection at the thin plate surface (W/m^2).
h	Convection coefficient at the plate surface $(W/(m^2 \cdot K))$.
u_∞	Temperature of the plate surrounding medium (K).
$\mathbf{x}_0 = \mathbf{x}_0(t)$	Laser spot center location at a given time $[x_0(t), y_0(t)]$ $([m, m])$.
$f = f(\mathbf{x}, \mathbf{x}_0)$	Laser power density model (W/m^2). There are four types in this manuscript: circle-shape (f_c), square-shape (f_s), Gaussian (f_g) and Dirac delta (f_d).
P	Laser power (W).
$\vec{v} = [v_x, v_y]$	Laser scanning speed (m/s).
r	Circle-shape laser spot radius (m).
Δx	Square-shape laser spot edge length (m).
σ	Parameter of the Gaussian laser model (m).
$\mathbb{X}_i = \mathbb{X}_i(x)$	i-th Fourier basis function in the x-axis.
$\mathbb{Y}_j = \mathbb{Y}_j(y)$	j-th Fourier basis function in the y-axis.
$\Theta_{ij} = \Theta_{ij}(t)$	ij-th Fourier coefficient for the temperature solution u.
$(\mathbf{\Omega}, \mathbf{X})$	Finite element discretization of the problem. $\mathbf{\Omega} = \{\Omega_1, \Omega_2, \ldots, \Omega_m\}$, $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x_2}, \ldots, \mathbf{x}_n\}$.

Δt	Timestep size for the time discretization in FEA.
$\mathbf{U}^{(t)}$	Nodal values of the temperature for the FEA discretization of the problem. $\mathbf{U}^{(t)} = [u(\mathbf{x}_1, t), u(\mathbf{x}_2, t), \ldots, u(\mathbf{x}_n, t)]^T$.
$\mathbf{F}^{(t)}$	Nodal values of the laser source for the FEA discretization of the problem. $\mathbf{F}^{(t)} = [f(\mathbf{x}_1, t), f(\mathbf{x}_2, t), \ldots, f(\mathbf{x}_n, t)]^T$.

IV-D.1.1 Introduction

Thin plate laser heating is an important manufacturing process in which a high powered laser source (such as a CO_2 or Nd-YAG laser) is applied to locally heat, melt and/or remove the plate material. Applications include metal plate alloying, drilling, forming, bending and cutting.

Numerical computer simulations of laser applications consume large computing resources, even for very small domains. On the other hand, analytic or closed form formulations require much less computer resources, at the price of lower precision and significant restrictions on the application circumstances. However, these analytic solutions become appealing as they may produce economic forecasts of the overall heating process, for specific study cases.

This manuscript presents an efficient analytic solution for the time history of the temperature field of thin rectangular flat plates heated by a constant speed laser spot. Unlike other analytic methods, our solution considers convective energy exchange and piecewise linear curved trajectories. It handles time and space domains sizes significantly larger than the feasible for FEA. Our method uses Fourier coefficients to find a solution in the frequency domain and maps it back to the time-space domain. We compute the solution for timestep t_n and trajectory piece $\mathbf{x_0}(t_n)$ based on timestep t_{n-1} and trajectory piece $\mathbf{x_0}(t_{n-1})$. The presented algorithm enables easy parallelization resulting in further improvement in the overall efficiency for larger space and/or time domains

This article is organized as follows: Sect. IV-D.1.2 reviews the relevant literature. Sect. IV-D.1.3 describes the methodology. Sect. IV-D.1.4 presents and discusses results of the conducted experiments. Sect. IV-D.1.5 concludes the paper. Sect. IV-D.1.6 introduces what remains for future work.

IV-D.1.2 Literature Review

This section discusses the state of the art for the simulation of the laser heating problem. Sect. IV-D.1.2.1 reviews the literature concerning numerical approaches to the problem solution while Sect. IV-D.1.2.2 discusses the analytic approaches. Sect. IV-D.1.2.3 concludes the literature review.

IV-D.1.2.1 Numerical methods for laser heating simulation

FEA is one of the most important numerical tool for thermodynamic analysis of metal plates under laser heating. [103, 104] perform a parametric study on a rectangular plate using FEA in order to measure the impact of laser speed, laser spot radius and laser power on the plate temperature distribution during laser heating. [105] performs and statistically validates the parametric analysis using ANOVA tests. [106, 107] solve a thermal/stress model with FEA in order to study the plate deformations due to the high temperature gradients. [108] compare FEA with trained ANN (Artificial Neural Networks) for predicting thermal stresses in laser cutting of glass sheets.

[109, 110] simulate rectangular cuts with laser using FEA, while [111, 112] perform the same analysis for circular and triangular cuts respectively. The enthalpy method is used to account for non-linearites of the material as well as phase changes that induce material melting. Experimental validation of the estimated temperature is presented using thermocouples.

[113] investigate the laser heating problem using the element birth and death method in order to account for material non-linearities. For accounting material removal in the FEA models, [114–116] incorporate a temperature-threshold approach which removes melted elements from the plate mesh during the simulation.

Aside from FEA, other numerical methods have been used for simulation of laser heating processes. [117–120] use the Finite Differences Method (FDM) for the analysis of laser heating phenomenon while [121, 122] employs a Boundary Element Method (BEM) approach. Recently, the Finite Volume Method (FVM) has been incorporated for the simulation of the laser heating problem [123, 124].

Despite the modeling complexity that can be reached with numerical tools, these approaches are highly sensitive to spatial and time discretizations of the plate [125–127]. Therefore, such approaches are currently unusable in industrial scenarios where fast decisions must be made for large plates and complex laser trajectories.

IV-D.1.2.2 Analytic methods for laser heating simulation

Analytic (or semi-analytic) solutions to the problem have been also proposed in the literature of laser heating simulation. [128] develop an ordinary non-linear differential equation which is then solved numerically for the laser heating problem. [129] solves a 1D laser heating problem for solid-liquid interfaces using the Laplace transform. [130] develop a non-linear analytic model which is solved iteratively to estimate the plate temperature in underwater laser cutting. The model is then validated numerically and statistically [131]. [132] present an analytic solution for the thermal/stress equations by means of Fourier series. [133] presents an analytic solution for the 3D laser heating problem for piecewise linear trajectories by a superposition of fundamental solutions in a semi-infinite domain. Convective heat losses are omitted at the plate surface and the plate is assumed with infinite depth.

Analytic approaches provide computationally faster results than numerical approaches. However, they are very limited in model assumptions [134]. Such limitations include: linear laser trajectories, 1D and 2D rectangular domains, constant material properties and null convection on the plate surface.

IV-D.1.2.3 Conclusions of the literature review

As discussed above, numerical tools are impractical for industrial scenarios [126,127] where decisions must be made on large plate sizes. Current analytic approaches partially overcome this problem by providing fast solutions at the cost of limitations such linear trajectories, no convection at the plate surface and material properties independent of the temperature. However, they only work for linear trajectories on the plate.

This manuscript presents an analytic solution for the 2D laser heating of rectangular thin plates problem. Our algorithm: *(1)* acts recursively in the time domain calculating the Fourier solution for time t_n using the coefficients from timestep t_{n-1}, *(2)* allows parallelization for computing the Fourier coefficients of timestep t_n. Features *(1)* and *(2)* are the basis for the algorithm low computational

Table IV-D.1.2: Comparison of the contributions and drawbacks of our analytic method and some state of the art methodologies.

Ref.	Curved Laser Trajectory	Convection at the Surface	Nonlinear Thermal Properties	Large Domain Study
Our Analytic Method	Yes	Yes	No	Yes
[128]	No	No	No	No
[129]	No	No	No	No
[130]	No	Yes	Yes	No
[132]	No	No	No	No
[133]	Yes	No	No	Yes
FDM [120]	No	Yes	No	No
BEM [122]	No	Yes	No	No
FEA [107]	Yes	Yes	Yes	No
FVM [124]	No	Yes	Yes	No

cost. Our analytic approach covers larger space and time domains than the ones achieved by FEA methods. A study case is presented in order to compare the convergence rate and execution times of the algorithm vs. FEA in a MATLAB implementation.

To illustrate the capabilities of the implemented approach, a study case for a large plate ($1\,m^2$, $12\,s$ history) is presented. Table IV-D.1.2 presents an appraisal of this manuscript contributions versus other approaches in the current literature.

IV-D.1.3 Methodology

This section discusses the methodology for our analytic solution to the laser heating problem and poses a study case. Sect. IV-D.1.3.1 introduces the theoretical model and assumptions for the heat transfer analysis. Sect. IV-D.1.3.2 presents the analytic solution to the problem. Sect. IV-D.1.3.3 discusses about the different laser models. Sect. IV-D.1.3.4 describes the implementation details of the solution. Sect. IV-D.1.3.5 briefly discusses the FEA approach used to validate numerically the analytic solution. Finally, Sect. IV-D.1.3.6 presents a simulation case of study.

IV-D.1.3.1 Heat equation for the thin plate laser heating problem

In the case of metal plates, it is reasonable to ignore heat transfer through the plate thickness (i.e. using a 2D model $\nabla \cdot k\nabla = k\frac{\partial^2}{\partial x^2} + k\frac{\partial^2}{\partial y^2}$) due to the relative size of the plate thickness w.r.t. its width and height ($\Delta z \to 0$) and the high thermal conductivity. According to Ref. [135], heat transfer in a 2D plate subject to a continuous laser source satisfies the following PDE with initial

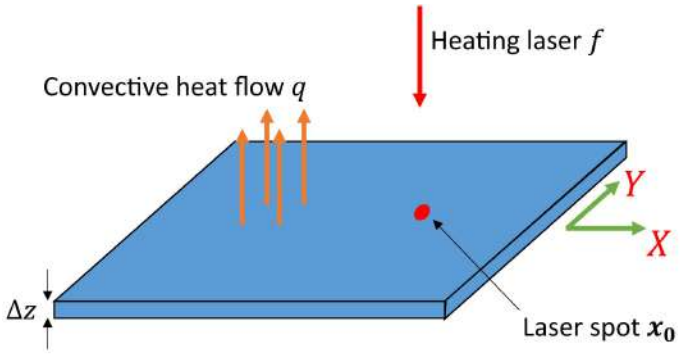


Figure IV-D.1.1: Schematic of the laser heating model. A laser passes an amount of energy f at a plate location $\mathbf{x_0}$ while the plate cools down due to convection q at the surface.

and boundary conditions:

$$\begin{aligned} \rho c_p \frac{\partial u}{\partial t} - \nabla \cdot (k \nabla u) &= \frac{f - q}{\Delta z} \\ q &= h(u - u_\infty) \\ u|_{x=0} = u|_{x=a} &= u|_{y=0} = u|_{y=b} = u_\infty \\ u(\mathbf{x}, t_0) &= u_\infty \end{aligned} \tag{IV-D.1.1}$$

where ρ, c_p and k are the material density, specific heat and thermal conductivity respectively (assumed constant in this manuscript). u is the temperature distribution on the plate. f is the laser surface power density of the laser (discussed in Sect. IV-D.1.3.3) and q is the heat loss due to convection at the plate surface. The plate initial temperature is assumed at constant ambient temperature u_∞ and the 2D borders of the plate are assumed at ambient temperature for the whole simulation. An scheme of the laser heating problem is depicted in Fig. IV-D.1.1.

IV-D.1.3.2 Analytic solution of the problem

Following the same procedure as in [132], the following analytic solution for the temperature distribution can be derived for Eq. (IV-D.1.1):

$$u(\mathbf{x}, t) = u_\infty + \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \Theta_{ij}(t) \mathbb{X}_i(x) \mathbb{Y}_j(y) \tag{IV-D.1.2}$$

with Fourier basis functions:

$$\begin{aligned} \mathbb{X}_i(x) &= \sin \frac{i \pi x}{a} \\ \mathbb{Y}_j(y) &= \sin \frac{j \pi y}{b} \end{aligned} \tag{IV-D.1.3}$$

and their respective Fourier coefficients $\Theta_{ij}(t)$. The value for these coefficients is given below.

IV-D.1.3.2.1 Fourier Coefficients

The closed form of the Fourier coefficients $\Theta_{ij}(t)$ from Eq. (IV-D.1.2) can be derived using separation of variables [132]:

$$\Theta_{ij}(t) = \frac{4}{ab\rho c_p \Delta z} \int_{t_0}^{t} \int_0^b \int_0^a f(\mathbf{x}, \mathbf{x}_0(\tau)) \mathbb{X}_i(x) \mathbb{Y}_i(y) e^{-\omega_{ij}(t-\tau)} dx dy d\tau \tag{IV-D.1.4}$$

where ω_{ij} are the eigenvalues of Eq. (IV-D.1.3) for the current operator (Eq. (IV-D.1.1)) defined as:

$$\omega_{ij} = \frac{k}{\rho c_p} \left(\frac{i^2\pi^2}{a^2} + \frac{j^2\pi^2}{b^2} \right) + \frac{h}{\rho c_p \Delta z} \tag{IV-D.1.5}$$

The curved trajectory $\mathbf{x_0}(t)$ is discretized into a sequence of linear trajectories $\mathbf{x_0}(t) = [\mathbf{x}_0(t_0), \mathbf{x}_0(t_1), \cdots, \mathbf{x}_0(t_n)]$. Therefore, Eq. (IV-D.1.4) becomes:

$$\begin{aligned} \Theta_{ij}(t_n) = \frac{4}{ab\rho c_p \Delta z} \sum_{l=0}^{n} e^{-\omega_{ij}(t_n - t_l)} \\ \int_{t_l - 1}^{t_l} \int_0^b \int_0^a f(\mathbf{x}, \mathbf{x}_0(\tau)) \mathbb{X}_i(x) \mathbb{Y}_i(y) e^{-\omega_{ij}(t_l - \tau)} dx dy d\tau \end{aligned} \tag{IV-D.1.6}$$

Such discretization allows to compute easier the integral term in Eq. (IV-D.1.4) for the nonlinear laser trajectory as a sum of linear laser trajectories. In order to satisfy the initial condition of the differential equation, the Fourier coefficients are initialized to $\Theta_{ij}(t_0) = 0$. Thus, Eq. (IV-D.1.2) becomes $u(\mathbf{x}, t_0) = u_\infty$, satisfying the initial condition presented in Eq. (IV-D.1.1).

IV-D.1.3.2.2 Recursive Fourier Coefficients

Eq. (IV-D.1.6) can be rewritten in recursive form as follows:

$$\begin{aligned} \Theta_{ij}(t_n) =& \Theta_{ij}(t_{n-1}) e^{-\omega_{ij}(t_n - t_{n-1})} + \\ & \frac{4}{ab\rho c_p \Delta z} \int_{t_{n-1}}^{t_n} \int_0^b \int_0^a f(\mathbf{x}, \mathbf{x}_0(\tau)) \mathbb{X}_i(x) \mathbb{Y}_i(y) e^{-\omega_{ij}(t_n - \tau)} dx dy d\tau \end{aligned} \tag{IV-D.1.7}$$

where $\Theta_{ij}(t_{n-1})$ are the Fourier coefficients of a previous timestep solution (recursive term). In the time domain, the term $\Theta_{ij}(t_n)$, for time t_n can be economically solved in recursive manner by using the term $\Theta_{ij}(t_{n-1})$ instead of computing the whole history. Furthermore, since the laser trajectory has been discretized into linear paths, the integral term in Eq. (IV-D.1.7) accounts for a linear laser trajectory at time t_n. Therefore, such integral can be solved easier than using a nonlinear path.

IV-D.1.3.3 Laser source models

Eq. (IV-D.1.7) requires evaluating the following integral for the laser beam source:

$$\int_{t_{n-1}}^{t_n} \int_0^b \int_0^a f(\mathbf{x}, \mathbf{x}_0) \mathbb{X}_i(x) \mathbb{Y}_j(y) e^{-\omega_{ij}(t_n - \tau)} dx dy d\tau \tag{IV-D.1.8}$$

Table IV-D.1.3: Equivalence table between laser model parameters.

Model	Parameter	Value
Circle-shape	r	
Square-shape	Δx	$\Delta x = r\sqrt{\pi}$
Gaussian model	σ	$\sigma = r$
Dirac laser		$r \to 0$

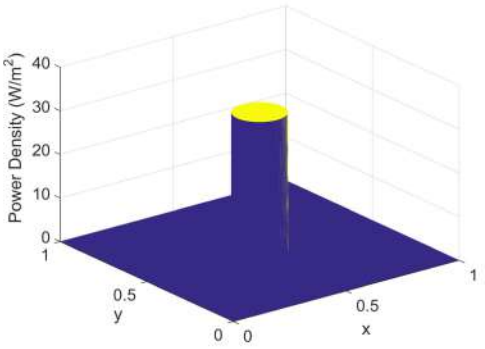
The value of such integral depends on the laser model used. The most common models used are: circle-shape laser model (f_c, Fig. IV-D.1.2(a)), square-shape laser model (f_s, Fig. IV-D.1.2(b)), Gaussian laser model (f_g, Fig. IV-D.1.2(c)) and Dirac delta laser model (f_d). Each of these models are presented below:

$$
\begin{aligned}
f_c(\mathbf{x}, \mathbf{x_0}) &= \begin{cases} \frac{P(1-R)}{\pi r^2}, & \|\mathbf{x} - \mathbf{x}_0\| < r \\ 0, & \text{otherwise} \end{cases} \\
f_s(\mathbf{x}, \mathbf{x_0}) &= \begin{cases} \frac{P(1-R)}{\Delta x^2}, & |x - x_0| < \frac{\Delta x}{2} \quad \wedge \quad |y - y_0| < \frac{\Delta x}{2} \\ 0, & \text{otherwise} \end{cases} \\
f_g(\mathbf{x}, \mathbf{x_0}) &= \frac{P(1-R)}{\pi\sigma^2} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_0\|^2}{\sigma^2}\right) \\
f_d(\mathbf{x}, \mathbf{x_0}) &= \lim_{\sigma \to 0} f_g(\mathbf{x}, \mathbf{x_0})
\end{aligned}
\tag{IV-D.1.9}
$$

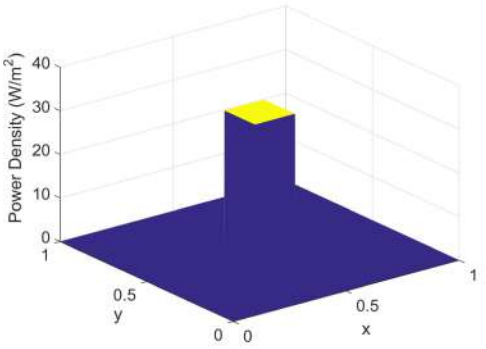
Solution for Eq. (IV-D.1.8) is presented in the Appendix for a squared (f_s) and a Dirac delta (f_d) laser source. For the other two laser models, Table IV-D.1.3 presents an equivalence between laser parameters such that the overall input energy $\int_0^b \int_0^a f(\mathbf{x}, \mathbf{x_0}) dxdy$ and the power density peak $\max_{\mathbf{x}} f(\mathbf{x}, \mathbf{x_0})$ of the laser beam are the same independently of the model. As the laser spot becomes smaller, all the energy input localizes in a smaller area despite the chosen model as illustrated in Fig. IV-D.1.3. Therefore, for relatively small laser spots (w.r.t. the 2D plate size), the heat transfer phenomenon described in Eq. (IV-D.1.1) should behave similarly for all the laser models.

IV-D.1.3.4 Algorithm overview

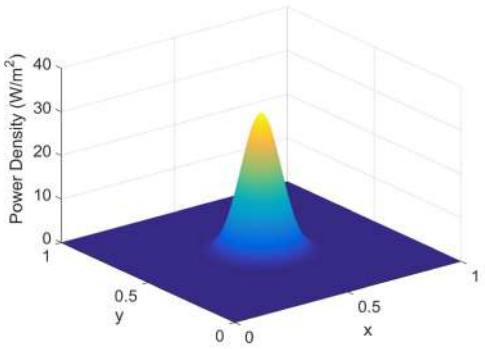
To apply the analytic solution posed in Eq. (IV-D.1.2), the curved laser trajectory $\mathbf{x}_0(t)$ is discretized into a sequence of linear trajectories $\mathbf{x_0}(t) = [\mathbf{x}_0(t_0), \mathbf{x}_0(t_1), \cdots, \mathbf{x}_0(t_n)]$. Such discretization is achieved by uniformly sampling the parametric trajectory such that the timestep remains constant through the whole simulation. Afterwards, Eq. (IV-D.1.7) is applied recursively on the sequence of linear trajectories in order to compute the Fourier coefficients at each timestep. As already discussed in Sect. IV-D.1.3.2, the algorithm is initialized by setting $\Theta_{ij}(t_0) = 0$. Finally, the temperature solution $u(\mathbf{x}, t_l)$ at any timestep t_l can be recovered by applying Eq. (IV-D.1.2). The infinite sum is truncated in order to obtain an approximate solution. Fig. IV-D.1.4 presents an overview of the algorithm.



(a) Circle-shape laser beam model f_c distribution.

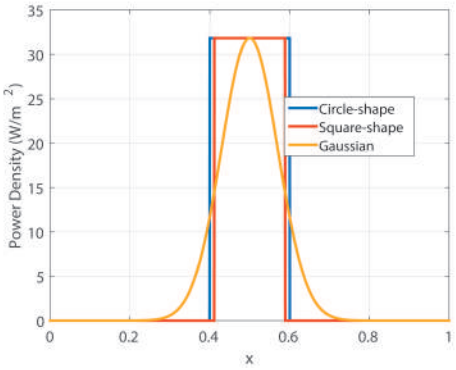


(b) Square-shape laser beam model f_s distribution.

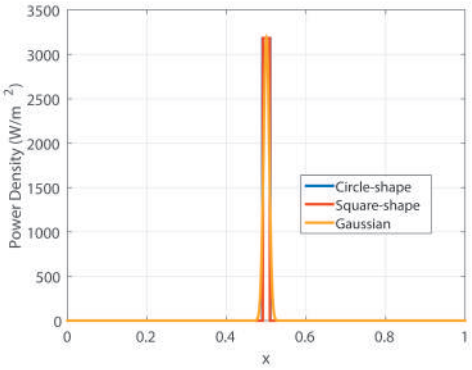


(c) Gaussian laser beam model f_g distribution.

Figure IV-D.1.2: Laser model distribution for the different laser beam models: f_c, f_s and f_g.



(a) Laser power density distribution for a spot radius $r = a/10$.



(b) Laser power density distribution for a spot radius $r = a/100$.

Figure IV-D.1.3: Distribution of the laser power densities for the different laser models along the x-axis using different laser spot sizes.

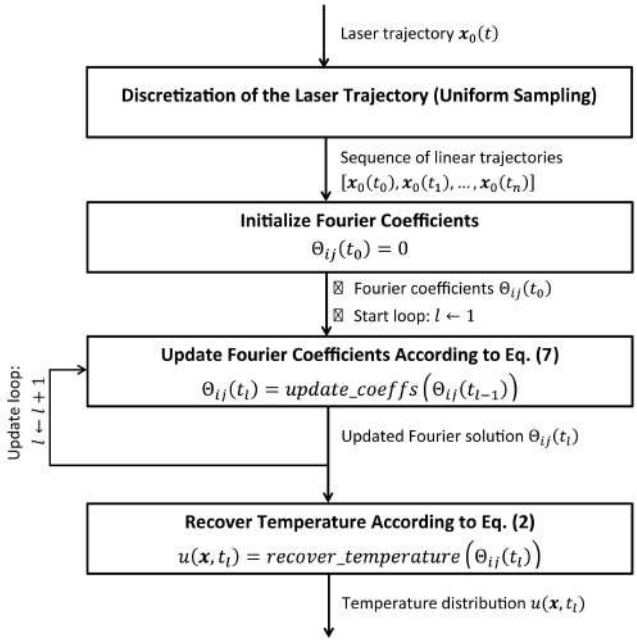


Figure IV-D.1.4: Diagram of the analytic approach algorithm.

Table IV-D.1.4: Physical parameters for simulation of laser heating of an AISI 304 steel plate (Ref. [106]). Natural convection due to surrounding air is considered.

Parameter	Value
ρ	$8030\,kg/m^3$
c_p	$574\,J/(kg \cdot K)$
k	$20\,W/(m \cdot K)$
R	0
h	$20\,W/(m^2 \cdot K)$
u_∞	$300\,K$

IV-D.1.3.5 Numerical Comparison with FEA

In order to validate numerically the implemented approach, FEA is used to simulate the laser heating problem. The FEA linear system of equations that arises for Eq. (IV-D.1.1) is:

$$\left[\left(\frac{\rho c_p}{\Delta t} + \frac{h}{\Delta z}\right)\mathbf{M} + k\mathbf{L}\right]\mathbf{U}^{(t+\Delta t)} = \mathbf{M}\left(\frac{\rho c_p}{\Delta t}\mathbf{U}^{(t)} + \frac{1}{\Delta z}\int_t^{t+\Delta t}\mathbf{F}^{(\tau)}\mathrm{d}\tau + \frac{h}{\Delta z}u_\infty\right) \quad \text{(IV-D.1.10)}$$

where:

$$\begin{aligned}\mathbf{L}_{ij} &= \sum_{\Omega_k \in \mathbf{\Omega}} \int_{\Omega_k} \nabla\phi_i \cdot \nabla\phi_j \mathrm{dA} \\ \mathbf{M}_{ij} &= \sum_{\Omega_k \in \mathbf{\Omega}} \int_{\Omega_k} \phi_i\phi_j \mathrm{dA}\end{aligned} \quad \text{(IV-D.1.11)}$$

are the Laplace-Beltrami (stiffness) and norm (mass) matrices respectively. $\mathbf{\Omega} = \{\Omega_1, \Omega_2, \ldots, \Omega_m\}$, $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x_2}, \ldots, \mathbf{x}_n\}$ is a discretization of the plate into finite elements. $\phi_i = \phi_i(\mathbf{x})$ is the interpolation function associated to the node $\mathbf{x}_i$. $\mathbf{U}^{(t)} = [u(\mathbf{x}_1, t), u(\mathbf{x}_2, t), \ldots, u(\mathbf{x}_n, t)]^T$ and $\mathbf{F}^{(t)} = [f(\mathbf{x}_1, t), f(\mathbf{x}_2, t), \ldots, f(\mathbf{x}_n, t)]^T$ are the nodal values of the temperature and the laser source respectively. Finally, Δt is the timestep size.

To carry out the comparison of our analytic algorithm with FEA, a small study case (which can be solved accurately with FEA) is simulated with both approaches: a $0.01\,m \times 0.01\,m \times 0.001\,m$ AISI 304 steel plate (Table IV-D.1.4) is heated by a $P = 100\,W$, $r = 0.0003\,m$ squared laser source (f_s) which follows the trajectory depicted in Fig. IV-D.1.5 at constant speed $\|\vec{v}\| = 0.1\,m/s$. To discretize the plate, triangular elements are used with linear interpolation (Fig. IV-D.1.6). A timestep $\Delta t = 0.0012\,s$ is chosen for the time discretization.

IV-D.1.3.6 Experimental setup

To test the implemented algorithm, a simulation study case with a relatively large plate is presented (computationally impractical for FEA). A $1\,m \times 1\,m \times 0.001\,m$ AISI 304 steel plate (Table IV-D.1.4) is heated by a $P = 100\,W$ point laser source (f_d) that follows the trajectory depicted in Fig. IV-D.1.5 at constant speed $\|\vec{v}\| = 0.1\,m/s$. The surface of the plate is surrounded by air, which cools the plate by natural convection. Ambient temperature is set at $300\,K$.

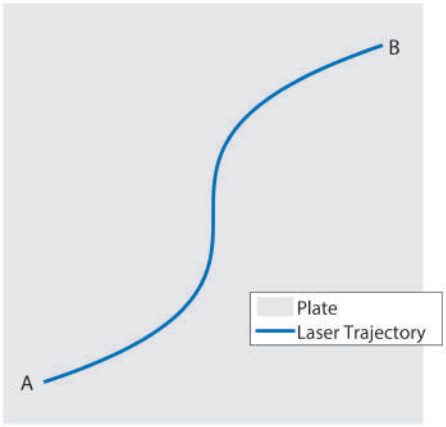


Figure IV-D.1.5: Trajectory of the laser on the plate surface (from A to B).

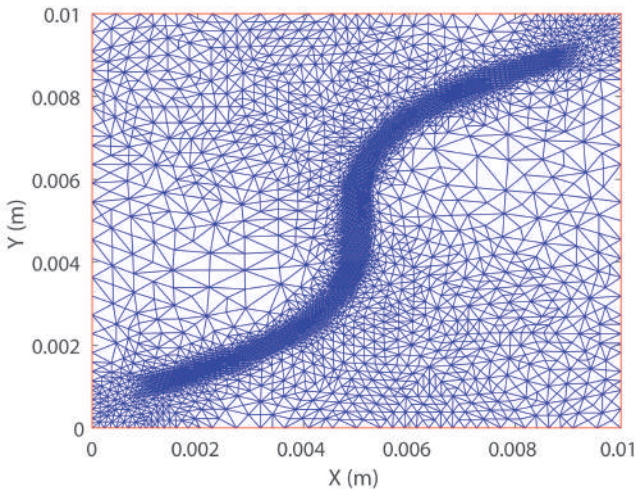


Figure IV-D.1.6: Triangular mesh discretization of the plate for FEA.

IV-D.1.4 Results and Discussion

This section presents and discusses a numerical comparison of the implemented analytic method against FEA for a small plate study case (Sect. IV-D.1.4.1). Sect. IV-D.1.4.2 presents results of our analytic algorithm for a large plate study case, where current FEA becomes impractical computationally. Finally, Sect. IV-D.1.4.3 compares measured execution times for the analytic (serial and parallel implementation) and FEM approaches.

IV-D.1.4.1 Numerical comparison of the analytic solution vs. FEA

This section presents the numerical results of the study case presented and discussed in Sect. IV-D.1.3.5. These results are used to compare the analytic approach with FEA. Fig. IV-D.1.7(a) plots the plate temperature distribution at the end of the simulation ($t = 0.12\,s$) estimated with our analytic approach. For this particular case, Eq. IV-D.1.2 is truncated at 200×200 Fourier terms since: *(1)* the error of the solution does not change significantly with more Fourier terms, and *(2)* such error is tolerable (below 3.5%). Fig. IV-D.1.7(b) plots the FEA temperature at the same simulation time. A timestep of $\Delta t = 0.0012\,s$ is used and the triangular mesh in Fig. IV-D.1.6 is for both FEA and our analytic solution (as per Eq. (IV-D.1.2)). Fig. IV-D.1.7(c) plots the relative error distribution of the analytic temperature considering the FEA solution as reference. A maximum relative error of 3.43% is measured around the laser spot.

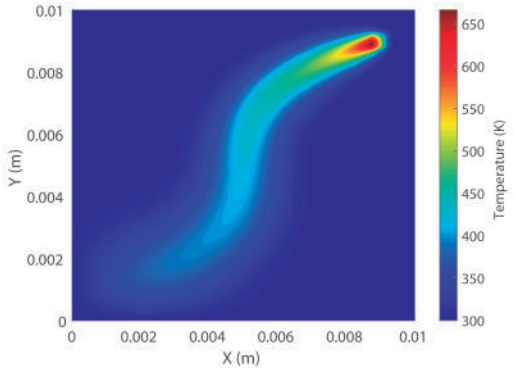
Fig. IV-D.1.8(a) plots the maximum relative error of the analytic solution as a function of the number of Fourier terms in Eq. (IV-D.1.2). The analytic solution stabilizes above 100×100 Fourier terms for the current study case (a relative error of $\approx 3.5\%$). Fig. IV-D.1.8(b) shows, in contrast, that the FEA solution falls below a relative error of 2% for mesh sizes above 10000 nodes. A FEA solution with a high resolution mesh ($> 50K$ mesh nodes) is used as reference temperature in both cases (MATLAB adaptive remeshing used). This relative analysis is only applicable to the described study case, as the convergence of the problem is dependent on the laser spot radius / plate size ratio. Larger plates (or smaller laser spots) require more Fourier terms in the analytic case and larger meshes in the FEA case in order to accurately simulate the physical phenomenon.

IV-D.1.4.2 Experiment results

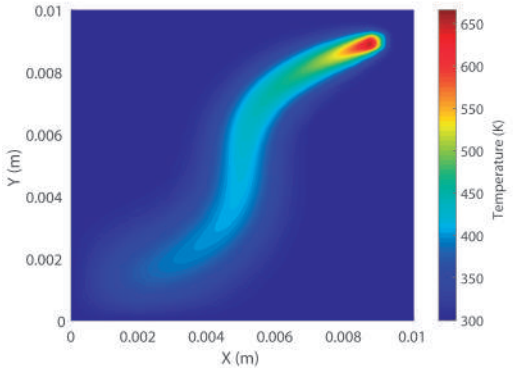
This section presents the results obtained the analytic approach for the large study case ($1\,m \times 1\,m \times 0.001\,m$) in Sect. IV-D.1.3.6. A point source f_d is used to simulate the laser. The Fourier series is set to 2000×2000 terms. Fig. IV-D.1.9 presents the evolution of the temperature field on the plate at different simulation times. Figs. IV-D.1.9(a) and IV-D.1.9(b) plot the temperature at early ($t = 0.6189\,s$) and halfway ($t = 6.1892\,s$) stages respectively while Fig. IV-D.1.9(c) plots the temperature at the end of the laser trajectory ($t = 12.3786\,s$). The zoom near the laser spot exhibits the high spatial resolution captured by our analytic solution.

IV-D.1.4.3 Computing times

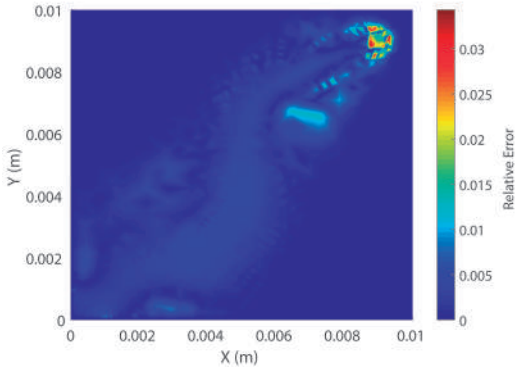
This section compares the analytic vs. FEA computing times for the case in Sect. IV-D.1.3.5. In the FEA case, Eq. (IV-D.1.10) is implemented in MATLAB and solved using a linear solver for sparse matrices (sparse Cholesky factorization library `CHOLMOD`).



(a) Analytic temperature distribution at $t = 0.12\,s$.

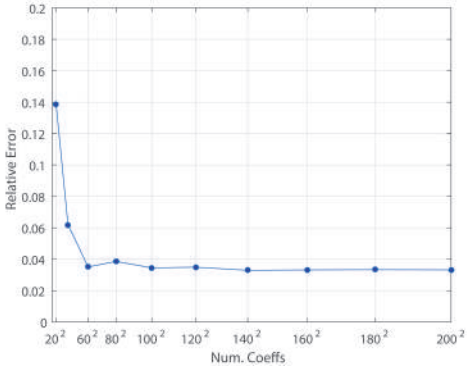


(b) FEA temperature distribution at $t = 0.12\,s$.

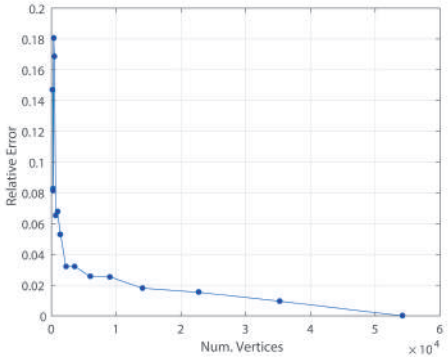


(c) Relative error of the analytic solution vs. FEA. The maximum relative error is 3.43%.

Figure IV-D.1.7: Temperature and error distribution for a small plate ($0.01\,m \times 0.01\,m \times 0.001\,m$) obtained by the analytic and FEA approaches.



(a) Error of our analytic solution as a function of the number of Fourier terms.



(b) Error in the FEA solution as a function of the number of nodes.

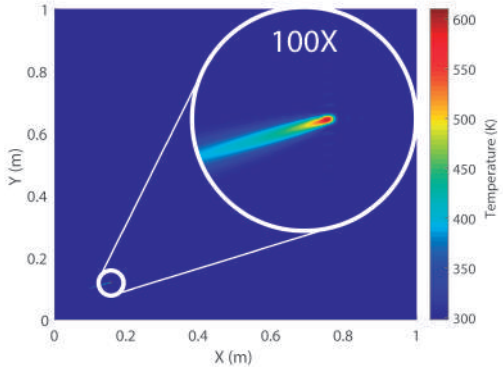
Figure IV-D.1.8: Maximum relative error evolution for the analytic and FEA methods for the study case in Sect. IV-D.1.3.5 (the reference solution is a $54K$ node FEA simulation).

Table IV-D.1.5: Hardware and software specifications of the machine used to run FEA and the analytic simulations in both serial and parallel form.

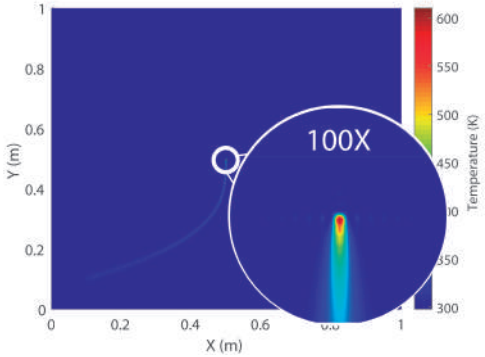
Item	Specifications
Operating System	Microsoft Windows 10 Home Single Language
Processor	Intel©Core™i7-4700HQ CPU @2.40GHz 2394 Mhz
Random Access Memory (RAM)	16.0 GB
Operating System Type	64 bits (x64)
GPU Unit	NVIDIA GeForce GTX 760M
Software	MATLAB R2014b 64-bit (win64), MATLAB Parallel Computing Toolbox

Our analytic solution lends itself for parallel computing. Our algorithm in Sect. IV-D.1.3.4 requires computing each Fourier coefficient $\Theta_{ij}(t)$ (Eq. (IV-D.1.7)) as a sequence of simple operations (such as sums, products and powers) independent from each other. These sequences of operations are independent between Fourier coefficients. In addition, the temperature field from Eq. (IV-D.1.2) describes the temperature at each point $\mathbf{x}$ in the domain as a linear combination of the Fourier basis. Hence, the temperature can be recovered for each point $\mathbf{x}$ in the domain independently of others points. Therefore, we implement the analytic algorithm using both MATLAB basic operations (serial implementation) as well as `gpuArray` operations from the MATLAB Parallel Computing Toolbox (parallel implementation). Table IV-D.1.5 lists the software and hardware specifications of the machine used to run FEA, as well as the analytic algorithm in both serial and parallel form. Such configuration is a low end for numerical computing. This modest demand poses an advantage for our analytic approach.

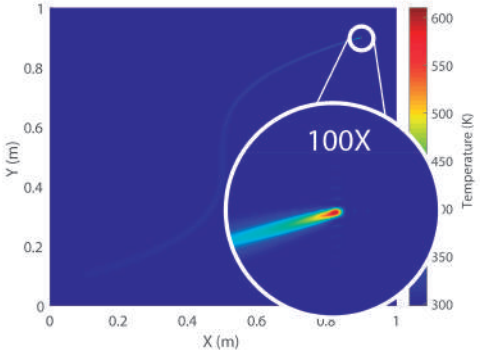
Fig. IV-D.1.10 plots the measured execution times for computing the Fourier coefficients (Eq. (IV-D.1.7)) with the analytic approach as a function of the number of Fourier terms in both serial and parallel implementations for the small plate case. The temperature recovery step of Eq. (IV-D.1.2) is not included in the measured times. The intersection point between the serial and



(a) Plate temperature distribution at $t = 0.6189\,s$.



(b) Plate temperature distribution at $t = 6.1892\,s$.



(c) Plate temperature distribution at $t = 12.3783\,s$.

Figure IV-D.1.9: Simulated temperature distribution for the large plate ($1\,m \times 1\,m \times 0.001\,m$) case study at different timestamps.

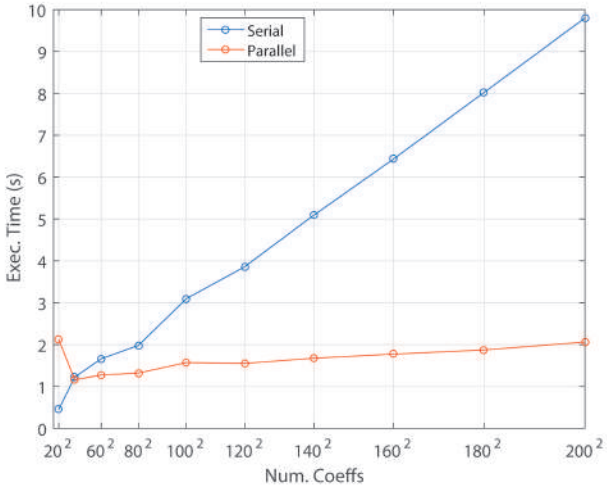


Figure IV-D.1.10: Serial and parallel execution times for computing the analytic Fourier solution (Eq. (IV-D.1.7)) vs. number of Fourier coefficients used for the small plate case study (Sect. IV-D.1.3.5). The temperature recovery step of Eq. (IV-D.1.2) is not considered.

parallel times in the plot is near the 60×60 Fourier coefficients and the the gap between the serial and parallel execution times becomes larger as the number of Fourier coefficients increases. Therefore, the parallel version of the algorithm becomes in fact, significantly faster than the serial one for larger number of coefficients.

Fig. IV-D.1.11 presents the execution times for FEA and the serial analytic algorithm as a function of the number of mesh nodes and the number of Fourier terms (for the analytic case). The measured computation times consider the computation of the Fourier terms at each timestep and the recovery of the temperature (as per Eq. (IV-D.1.2)) at the end of the simulation in the case of the analytic approach. However, the meshing step is not taken into account for measuring analytic or FEA times. Our analytic approach performs significantly faster than FEA as the mesh size increases, even for a large number of Fourier coefficients. Such difference in efficiency becomes crucial as the problem grows to bigger domains where FEA becomes very expensive computationally. For simplicity of the plot, parallel times are not included in Fig. IV-D.1.11. However, our experiments showed that the parallel implementation of the analytic algorithm performs better than the serial one (and therefore, better than FEA).

IV-D.1.5 Conclusions

This paper has presented a parallel efficient analytic solution for the 2D rectangular plate laser heating problem for curved laser trajectories and convection at the plate surface. Our algorithm discretizes the curved laser trajectory as a piecewise linear trajectory with constant speed. The solution for timestep t_n in the trajectory $\mathbf{x_0}(t_n)$ uses the result accumulated till the previous timestep t_{n-1}. Our analytic solution allows to consider convective energy into the balance. Although the assumptions of the mathematical model simplify the laser heating phenomenon (constant material

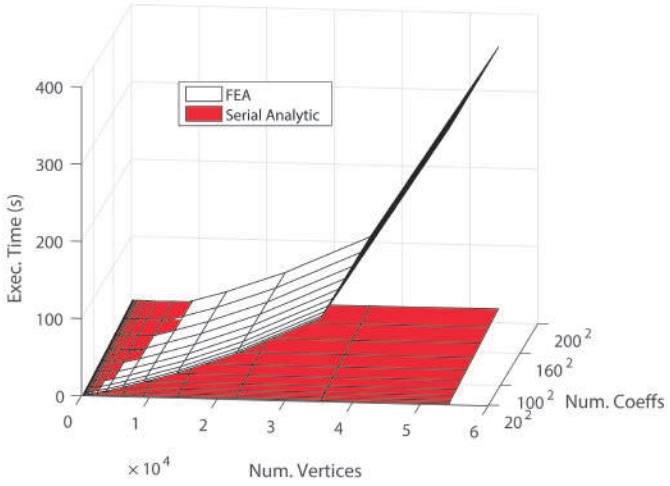


Figure IV-D.1.11: Execution times for computing the temperature solution with FEA and the serial analytic approaches w.r.t. the mesh size and the number of Fourier coefficients used in the analytic approach. Both Fourier computation times (Eq. (IV-D.1.7)) and temperature recovery times (Eq. (IV-D.1.2)) are considered. Meshing times are not considered.

properties, no phase changes and constrained plate and laser geometries), the analytic algorithm provides an efficient solution to problems that are very expensive computationally for current FEA methods.

Computing times of our algorithm are significantly lower than the FEA times for the same problem. Numerical comparison of the analytic method with FEA presents a relative error that reaches a maximum of 3.5% in very localized areas at the laser spot. Results for a $1\,m \times 1\,m \times 0.001\,m$ AISI 304 steel plate during $12.37\,s$ history (intractable with FEA) are presented for our analytic method.

IV-D.1.6 Future Work

Current mathematical description of heat transfer phenomena does not consider all physical factors involved in the laser machining process. Essentially, the methodology presented in this manuscript provides an approximated, but fast solution to the laser heating problem.

Consideration of radiative heat exchange between the laser beam and the metal plate requires the introduction of non-linear terms in the heat transfer equation (Eq. (IV-D.1.1)). As of our knowledge, there is no known analytic solution to such a non-linear formulation, even for the simplest geometries.

The authors remark that the presented method heavily rests on the continuous rectangular thin plate assumption. Domain discontinuities (e.g. holes) require a quite different approach, which surely would include both analytic and numerical methods. The authors seek to address such problems, and solutions, in the near future.

IV-D.2

Accelerated Thermal Simulation for 3D Interactive Optimization of CNC Sheet Metal Laser Cutting

Daniel Mejia[1,2], Aitor Moreno[1], Ander Arbelaiz[1], Jorge Posada[1], Oscar Ruiz-Salguero[2], Raúl Chopitea[3].

[1] Vicomtech-IK4, Donostia-San Sebastián Spain.

[2] CAD/CAM/CAE Laboratory, Universidad EAFIT, Medellin, Colombia.

[3] Lantek Investigación y Desarrollo, Parque Tecnológico de Álava, Miñano (Araba/Álava), Spain.

Citation

Daniel Mejia, Aitor Moreno, Ander Arbelaiz, Jorge Posada, Oscar Ruiz-Salguero and Raúl Chopitea. Accelerated thermal simulation for three-dimensional interactive optimization of computer numeric control sheet metal laser cutting. *Journal of Manufacturing Science and Engineering*, **2018**, 140, pp. 031006:1-031006:9. DOI: 10.1115/1.4038207.

Indexing: ISI (Q2), SCOPUS (Q1), Publindex (A1)

Abstract

In the context of CNC-based (Computer Numeric Control) sheet metal laser cutting, the problem of heat transfer simulation is relevant for the optimization of CNC programs. Current physically-based simulation tools use numeric or analytic algorithms which provide accurate but slow solutions

due to the underlying mathematical description of the model. This manuscript presents: (*1*) an analytic solution to the laser heating problem of rectangular sheet metal for curved laser trajectories and convective cooling, (*2*) a GPU implementation of the analytic solution for fast simulation of the problem, and (*3*) an integration within an interactive environment for the simulation of sheet metal CNC laser cutting. This analytic approach sacrifices the material removal effect of the laser cut in favor of an approximated real-time temperature map on the sheet metal. The articulation of thermal, geometric and graphic feedback in virtual manufacturing environments enables interactive redefinition of the CNC programs for better product quality, lower safety risks, material waste and energy usage among others. The error with respect to FEA in temperature prediction descends as low as 3.5 %.

Keywords: Computational fabrication, Geometric algorithms, Heat transfer, CNC optimization, Fast simulation.

IV-D.2.1 Introduction

Sheet metal cutting is an important technique of metalworking, and is widely used in many industries (automotive, aeronautics, etc). One of the most efficient and advanced technologies for sheet metal cutting is CNC laser cutting (specially for steel and aluminum), a process in which a high-power laser beam is directed through a nozzle to cut the material (melting, burning or vaporizing it) providing high-quality surface finishing, greater accuracy, and quicker production [136].

Interactive 3D simulation of CNC laser cuts has proven to be useful at industrial level allowing a better design of the cutting parameters to optimize the production in many ways [137–139]. The ability to interactively visualize and modify the effect of a certain CNC sequence for laser cutting is a valuable resource for manual and automatic optimization procedures that aim to reduce manufacturing costs. Moreover, laser cutting interactive simulation is a good example of the challenges and opportunities identified in [55].

However, current laser-cutting interactive simulations are in most cases purely geometric. One of the reasons is the fact that accurate physical simulations require massive computing resources (e.g. FEA), incompatible with interactive simulation.

In this work, we present the implementation of a GPU accelerated simulation of sheet metal laser heating/cutting, integrated into an interactive CNC sheet cut environment. This contribution allows novel and more interactive ways of designing and planning the laser cutting processes, in order to improve not only geometric and material waste aspects, but also the thermal effects on the sheet.

The remainder of this manuscript is organized as follows: Section IV-D.2.2 reviews the relevant literature. Section IV-D.2.3 presents the methodology and models used. Section IV-D.2.4 presents and discusses the results. Section IV-D.2.5 presents conclusions and introduces what remains for future work.

IV-D.2.2 Literature Review

IV-D.2.2.1 Thermal Simulation of Laser Cutting

Analytic solutions to the laser heating problem have been developed in order to estimate the temperature on rectangular sheet metal. Solutions that account only one dimension (distance from the laser source to the hole boundary [128] or the sheet depth [129]) have been presented for fast solution of laser drilling problems. As the drilling process dictates, such approaches require the laser to be at a fixed location through the entire simulation.

An analytic solution for the 3D laser heating problem is presented in [132]. The solution uses Fourier series to express the temperature field at any location of the sheet. However, such solution only accounts for a straight line laser trajectory perpendicular to one of the edges of the rectangular sheet. An analytic solution for general piecewise linear laser trajectories is presented in [133]. This approach employs fundamental solutions to compute the temperature on a 3D sheet with infinite depth.

Simulation based on analytic solutions provide the advantage of being computationally efficient at the cost of limited model assumptions such as: linear laser trajectories, rectangular domains, constant material properties and null convection on the sheet surface [134]. On the other hand, numerical approaches provide stronger tools for the simulation of complex physical phenomena. Finite Element Analysis (FEA) is a common numerical tool for the simulation of thermodynamic phenomena. 3D simulations of triangular [112], rectangular [109, 110] and circular cuts [111] have been achieved using nonlinear FEA to analyze how changes in the laser trajectory impact the resulting temperature profile. To account laser ablation, methodologies such as the element birth and death method [113], volume fractions [140], temperature thresholds [114–116] or the enthalpy method [109–112, 141] are coupled to the FEA routines. Other numerical methods include: Finite Differences [117, 118, 120, 142], Boundary Elements [121, 122] and Finite Volumes [123, 124]. All the aforementioned numerical methods present the shortcoming of being computationally expensive for large geometries and complex laser trajectories [40, 125–127, 143], rendering them useless for interactive simulation of industrial scenarios.

IV-D.2.2.2 Virtual Manufacturing Environment to Support Laser Machining Processes

There has not been a lot of effort in the integration of geometric and physical modules for the simulation of laser machining processes. Current state of the art algorithms for thermal analysis of the problem impose computational time constraints that do not allow an interactive workflow between the geometric simulation and the thermal simulation for complex laser trajectories. However, the integration is a relevant research topic for current trends in virtual manufacturing [55] and specifically, for laser path planning optimization [139, 144]. A coupling of a numerical solver inside a Simulated Annealing program has been presented for the optimization of laser cutting trajectories [142, 145]. The computational cost of such integration is high considering that heuristic algorithms have to simulate multiple thermodynamic scenarios. Therefore, the use of analytical temperature solutions have proven to be more usable for laser path optimization regardless of the underlying model simplifications [146].

To overcome the aforementioned problem, this manuscript presents the integration of a fast analytic heat solver with a geometric module for simulation of the laser cutting process on a rect-

angular sheet metal. The heat solver allows curved laser trajectories and considers convection at the sheet surface. The geometric module represents the sheet using the algorithms described in [138, 147]. The geometric model of the cut sheet is texturized with the color mapped temperature field (dynamically computed in the GPU). The implemented approach allows visual interaction at real-time rates for large geometries and complex laser trajectories.

IV-D.2.3 Methodology

This section presents an analytic solution to the laser heating problem on rectangular sheets. The analytic solution enables efficient computation of the temperature for any curved laser trajectory defined on the sheet, allowing convection at the sheet surface. Using the GPU capabilities, the temperature solution is integrated in a geometric module that simulates the CNC machining process and the sheet cutting operation.

IV-D.2.3.1 Heat Equation for the Sheet Laser Heating Problem

In the case of a sheet metal, it is reasonable to ignore heat transfer through the sheet depth (i.e. using a 2D model $\nabla \cdot k\nabla = k\frac{\partial^2}{\partial x^2} + k\frac{\partial^2}{\partial y^2}$) due to the relative size of the sheet depth with respect to its width and height ($\Delta z \to 0$) and the high thermal conductivity. Therefore, the equation that models the temperature $u = u(\mathbf{x}, t)$ distribution on a 2D rectangular sheet subject to a continuous laser source should satisfy the following PDE with initial and boundary conditions [135]:

$$\begin{aligned} \frac{f-q}{\Delta z} &= \rho c_p \frac{\partial u}{\partial t} - \nabla \cdot (k \nabla u) \\ q &= h(u - u_\infty) \\ u|_{x=0} &= u|_{x=a} = u|_{y=0} = u|_{y=b} = u_\infty \\ u(\mathbf{x}, 0) &= u_\infty \end{aligned} \tag{IV-D.2.1}$$

where ρ, c_p and k are the material density, specific heat and thermal conductivity, respectively (assumed constant in this manuscript). u is the temperature distribution on the sheet, q is the heat loss due to convection at the sheet surface and f is the surface power density of a squared laser beam centered at $\mathbf{x_0} = [x_0, y_0]$:

$$f(\mathbf{x}, \mathbf{x_0}) = \begin{cases} \frac{P(1-R)}{\Delta x^2}, & |x - x_0| < \frac{\Delta x}{2} \quad and \\ & |y - y_0| < \frac{\Delta x}{2} \\ 0, & otherwise \end{cases} \tag{IV-D.2.2}$$

with R being the sheet reflectivity and P, Δx being the laser power and laser diameter, respectively. The sheet initial temperature is assumed at constant ambient temperature u_∞ and the 2D borders of the sheet are assumed at ambient temperature for the whole simulation. An scheme of the problem is depicted in Fig. IV-D.2.1.

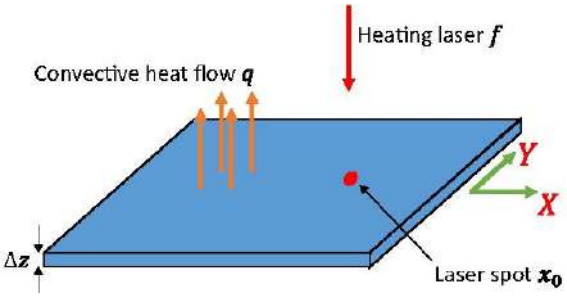


Figure IV-D.2.1: Schematic of the laser heating model. A laser passes an amount of energy f at a sheet location $\mathbf{x_0}$ while the sheet cools down due to convection q at the surface.

IV-D.2.3.2 Analytic Solution

Similar to [132], the following analytic solution for the temperature distribution can be derived for Eq. (IV-D.2.1):

$$u(\mathbf{x},t) = u_\infty + \sum_{i=1}^{\infty}\sum_{j=1}^{\infty} \Theta_{ij}(t)\mathbb{X}_i(x)\mathbb{Y}_j(y) \tag{IV-D.2.3}$$

with Fourier basis functions:

$$\begin{aligned} \mathbb{X}_i(x) &= \sin\frac{i\pi x}{a} \\ \mathbb{Y}_j(y) &= \sin\frac{j\pi y}{b} \end{aligned} \tag{IV-D.2.4}$$

and their respective Fourier coefficients $\Theta_{ij}(t)$. These Fourier coefficients can be derived using separation of variables:

$$\begin{aligned} \Theta_{ij}(t_n) = \frac{4}{ab\rho c_p \Delta z}\sum_{l=0}^{n} e^{-\omega_{ij}(t_n - t_l)} \times \\ \int_{t_l-1}^{t_l}\int_0^b\int_0^a f(\mathbf{x},\mathbf{x}_0(\tau))\mathbb{X}_i(x)\mathbb{Y}_i(y)e^{-\omega_{ij}(t_l-\tau)}dxdyd\tau \end{aligned} \tag{IV-D.2.5}$$

with eigenvalues ω_{ij} defined as:

$$\omega_{ij} = \frac{k}{\rho c_p}\left(\frac{i^2\pi^2}{a^2} + \frac{j^2\pi^2}{b^2}\right) + \frac{h}{\rho c_p \Delta z} \tag{IV-D.2.6}$$

In Eq. (IV-D.2.5), the curved trajectory $\mathbf{x_0}(t)$ has been discretized into a piecewise linear trajectory $\mathbf{x_0}(t) = [\mathbf{x}_0(t_0), \mathbf{x}_0(t_1), \cdots, \mathbf{x}_0(t_n)]$. Between the timestep $(t_{l-1}, t_l]$, the linearized laser subtrajectory $\mathbf{x_0}(t_l)$ is defined as:

$$\mathbf{x_0}(t_l) = \mathbf{v}\cdot(t_l - t_{l-1}) + \mathbf{p} \tag{IV-D.2.7}$$

with instantaneous speed $\mathbf{v} = [v_x, v_y]^{\mathrm{T}}$ and origin of subtrajectory $\mathbf{p} = [p_x, p_y]^{\mathrm{T}}$. Therefore, for each linear subtrajectory, the integral term in Eq. (IV-D.2.5) becomes:

$$\begin{aligned}
&\int_{t_{l-1}}^{t_l} \int_0^b \int_0^a f(\mathbf{x}, \mathbf{x}_0) \mathbb{X}_i(x) \mathbb{Y}_j(y) e^{-\omega_{ij}(t_l - \tau)} dx dy d\tau \\
&\quad = \frac{abP(1-R)}{ij\pi^2 \Delta x^2} \\
&\quad \Bigg[c_1 c_3 \int_0^{\Delta t} e^{-\omega_{ij}(\Delta t - \tau)} \cos\alpha_x \tau \cos\alpha_y \tau d\tau \\
&\quad - c_1 c_4 \int_0^{\Delta t} e^{-\omega_{ij}(\Delta t - \tau)} \cos\alpha_x \tau \sin\alpha_y \tau d\tau \\
&\quad - c_2 c_3 \int_0^{\Delta t} e^{-\omega_{ij}(\Delta t - \tau)} \sin\alpha_x \tau \cos\alpha_y \tau d\tau \\
&\quad + c_2 c_4 \int_0^{\Delta t} e^{-\omega_{ij}(\Delta t - \tau)} \sin\alpha_x \tau \sin\alpha_y \tau d\tau \Bigg]
\end{aligned} \tag{IV-D.2.8}$$

where:

$$\begin{aligned}
c_1 &= \cos\beta_x - \cos\gamma_x, \quad & c_2 &= \sin\beta_x - \sin\gamma_x, \\
c_3 &= \cos\beta_y - \cos\gamma_y, \quad & c_4 &= \sin\beta_y - \sin\gamma_y, \\
\alpha_x &= \frac{i\pi v_x}{a}, \quad & \alpha_y &= \frac{j\pi v_y}{b}, \\
\beta_x &= \frac{i\pi(p_x + \Delta x/2)}{a}, \quad & \beta_y &= \frac{j\pi(p_y + \Delta x/2)}{b}, \\
\gamma_x &= \frac{i\pi(p_x - \Delta x/2)}{a}, \quad & \gamma_y &= \frac{j\pi(p_y - \Delta x/2)}{b} \\
&\Delta t = t_l - t_{l-1}
\end{aligned} \tag{IV-D.2.9}$$

The integral terms in Eq. (IV-D.2.8) can be solved in closed form for any linear subtrajectory, i.e. no numerical integration is required. Therefore, the Fourier solution of the temperature (Eq. (IV-D.2.5)) for a given timestep t_l can be computed directly with the available information of (*i*) the sheet and laser parameters, (*ii*) the current linear subtrajectory parameters, and (*iii*) the Fourier solution of the previous timestep t_{l-1}. The temperature of the sheet can be retrieved from Eq. (IV-D.2.3) after truncating the Fourier solution.

IV-D.2.3.3 Interactive Simulation of Sheet Metal Cutting

In order to estimate the temperature of the sheet for a given subtrajectory, the analytic algorithm requires (as per Eq. (IV-D.2.5)): *(i)* the trajectory of the current machining operation: origin, destination and speed (timing information); and *(ii)* the Fourier coefficients calculated in the previous simulation step. The analytic temperature is estimated by applying the following steps iteratively: *(i)* compute Fourier coefficients for the current subtrajectory, *(ii)* retrieve the temperature from the computed coefficients.

Configurable discretizations of curved laser trajectories are required. A discretization is basically a piecewise linear reparameterization of a curve $\mathbf{x_0}(t) = [\mathbf{x}_0(t_0), \mathbf{x}_0(t_1), \cdots, \mathbf{x}_0(t_n)]$, which permits variable traversal velocity of the tool and regulation of the torch (power and radius of the laser beam).

In the first step, the calculation of Fourier coefficients for each linear subtrajectory requires only the coefficients from the previous subtrajectory (Eq. (IV-D.2.5)). This calculation is fast and independent of the sheet geometry discretization. Contrary to FEA methods, Eq. (IV-D.2.5) does not impose a restriction on the timestep Δt, allowing arbitrarily large timesteps for any linear subtrajectory.

For the second step, the evaluation of the actual temperatures in the sheet metal requires to use Eq. (IV-D.2.3) at discrete sample points of the sheet. A $m \times n$ grid generates such sample points. The temperature evaluated on each sample point is used to create a RGBA color image. A color mapping function is used to obtain a color from a given temperature value.

To simulate the material removal due to laser ablation, the geometric module presented in [138] is used. This geometric module represents the laser trajectory on the sheet as a set of 2D boolean operations between the sheet itself and the contours generated by the laser trajectory. A high-level spatial subdivision algorithm is implemented for such boolean operations in order to increase the computation performance. Moreover, the simulated ablation is purely geometric as the analytic approach does not account for the material removal. This assumption significantly reduces the computational cost of the simulation at the expense of numerical accuracy in the estimated temperature. Such a gain in computational efficiency becomes crucial in industrial applications where immediate albeit approximate results are required for the design of complex CNC processes. Fig. IV-D.2.2 lays out the modules of the CNC laser cut simulator. An additional heat source term might complement Eq. (IV-D.2.1) by filtering the heat propagation through the sheet cuts. Such filtering would allow to simulate heat propagation across the empty space produced by the laser cutting process. However, we have not identified any filter of this kind in the literature. Future work aims to explore this possibility.

IV-D.2.3.4 Algorithm Analysis

Fig. IV-D.2.3 presents the algorithm or pseudocode of the analytic laser cutting simulation. The procedure AnalyticLaserSimulation calls the analytic routines ComputeFourier and ComputeTemperature.

Given the (piecewise linear) laser trajectory $\mathbf{x_0}$, function ComputeFourier applies Eq. (IV-D.2.5) to compute the Fourier coefficients of the solution. To satisfy $u(\mathbf{x}, t_0) = u_\infty$ in Eq. (IV-D.2.3), the Fourier coefficients are initialized by setting $\Theta_{ij}(t_0) = 0$. The function UpdateCoeffs computes the value of a single Fourier coefficient at the current subtrajectory end given its previous value. Since this last operation is achieved in constant time, the time complexity order of ComputeFourier is $\mathcal{O}(num\_subtrajectories \times num\_coeffs)$.

On the other hand, the function ComputeTemperature applies Eq. (IV-D.2.3) to retrieve the temperature distribution from the computed Fourier coefficients. The sheet temperature is initialized at ambient temperature (i.e. $u[k] \leftarrow u_\infty$). The function SampleSheet produces the sheet sampling in linear time $\mathcal{O}(num\_points)$. The function FourierBasis applies Eq. (IV-D.2.4) to compute the Fourier basis for a given coefficient and a given sample point (in constant time). Therefore, the time complexity order of ComputeTemperature is $\mathcal{O}(num\_points \times num\_coeffs)$. The infinite sum of Fourier basis functions has been truncated by num_coeffs in order to obtain

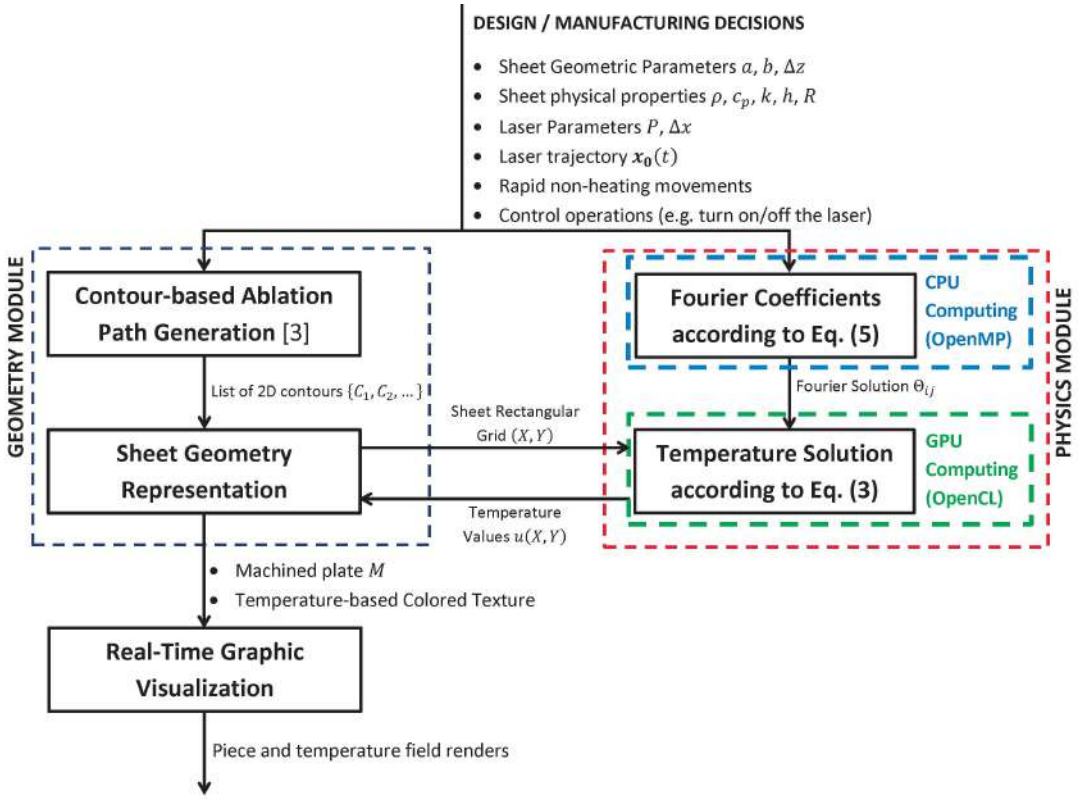


Figure IV-D.2.2: Integration scheme between the physics and geometry modules for interactive simulation of the CNC sheet metal laser cutting

Algorithm 1 Analytic Simulation of Sheet Laser Cutting

```
1:  procedure AnalyticLaserSimulation
2:      Θ ← ComputeFourier(x_0, num_coeffs)
3:      u ← ComputeTemperature(Θ, num_points)
4:  end procedure
5:
6:  function ComputeFourier(x_0, num_coeffs)
7:      Θ.Initialize(num_coeffs)
8:      for x_0(t_l) ∈ x_0 do
9:          for Θ_ij ∈ Θ do
10:             Θ_ij ← UpdateCoeffs(x_0(t_l), Θ_ij)
11:         end for
12:     end for
13:     return Θ
14: end function
15:
16: function ComputeTemperature(Θ, num_points)
17:     num_coeffs ← Θ.Size
18:     u.Initialize(num_points)
19:     x ← SampleSheet(num_points)
20:     for k ∈ {0, 1, ..., num_points − 1} do
21:         for ij ∈ {0, 1, ..., num_coeffs − 1} do
22:             u[k] ← u[k]+
23:                 Θ_ij × FourierBasis_ij(x[k])
24:         end for
25:     end for
26:     return u
27: end function
```

Figure IV-D.2.3: Algorithm for Analytic Simulation of Sheet Laser Cutting

an approximate solution.

The geometry module call (presented in [138]) simulates the laser cuts on the sheet as geometric boolean operations. Using spatial subdivision, the time complexity order of such calling is $\mathcal{O}(ne \times log(ne) + k + z \times log(ne))$ (where z is the number of cut contours, ne is the number of contour edges and k is the number of edge intersections) [138].

As a consequence, the time complexity order of the physical (analytic) module becomes:

$$\begin{aligned} &\mathcal{O}(\textsc{ComputeFourier}) + \mathcal{O}(\textsc{ComputeTemperature}) \\ &= \mathcal{O}(ns \times nc) + \mathcal{O}(np \times nc) \\ &= \mathcal{O}(nc(ns + np)) \end{aligned} \tag{IV-D.2.10}$$

with $nc = num\_coeffs$, $ns = num\_subtrajectories$ and $np = num\_points$.

Eqs. (IV-D.2.3) and (IV-D.2.5) present some interesting properties at the implementation level as follows: *1)* for the current subtrajectory, each Fourier coefficients is independent from the rest of the coefficients and, *2)* the temperature at each sample point is independent of the other points. Therefore, the functions ComputeFourier and ComputeTemperature allow easy parallelization. In the following section we take advantage of such parallelization properties in order to use modern parallel computer architectures such as multi-core computing and GPU.

IV-D.2.4 Results and Discussion

This section presents and discusses the results of the system (Fig. IV-D.2.2) that integrates: *(i)* the physics (analytic) module that calculates the temperature field, and *(ii)* the geometry module for the sheet metal geometry representation. Section IV-D.2.4.1 presents a numerical comparison of our analytic module against a FEA implementation. Section IV-D.2.4.2 presents results of the temperature evaluation using OpenCL. Section IV-D.2.4.3 presents a performance assesment of the analytic module. Section IV-D.2.4.4 discusses the interactivity of our implementation. Section IV-D.2.4.5 discusses the impact of the implemented interactive simulator in the design workflow of CNC-based laser cutting programs.

IV-D.2.4.1 Numerical Comparison with FEA

The FEA linear system for Eq. (IV-D.2.1) is:

$$\begin{aligned} &\left[\left(\frac{\rho c_p}{\Delta t} + \frac{h}{\Delta z}\right)\mathbf{M} + k\mathbf{L}\right]\mathbf{U}^{(t+\Delta t)} = \\ &\mathbf{M}\left(\frac{\rho c_p}{\Delta t}\mathbf{U}^{(t)} + \frac{1}{\Delta z}\int_t^{t+\Delta t}\mathbf{F}^{(\tau)}\mathrm{d}\tau + \frac{h}{\Delta z}u_\infty\right) \end{aligned} \tag{IV-D.2.11}$$

where:

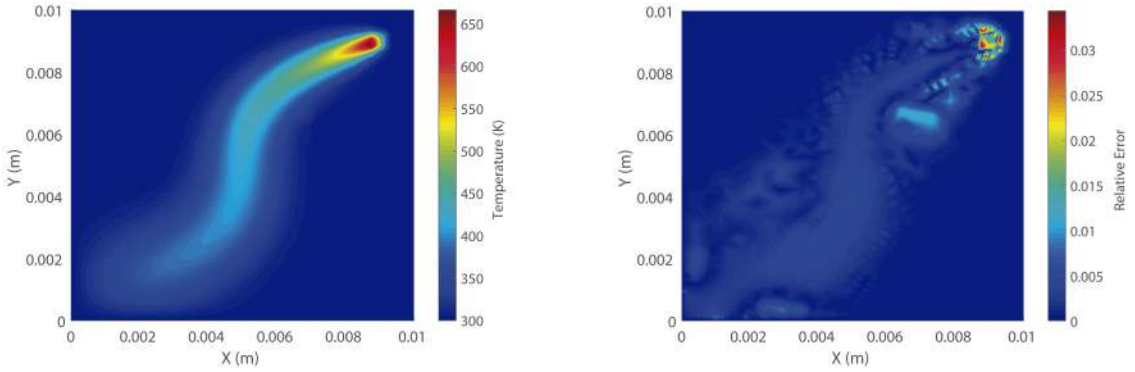
$$\begin{aligned} \mathbf{L}_{ij} &= \sum_{\Omega_k \in \Omega}\int_{\Omega_k} \nabla\phi_i \cdot \nabla\phi_j \mathrm{dA} \\ \mathbf{M}_{ij} &= \sum_{\Omega_k \in \Omega}\int_{\Omega_k} \phi_i\phi_j \mathrm{dA} \end{aligned} \tag{IV-D.2.12}$$

are the stiffness and mass matrices, respectively. $\Omega = \{\Omega_1, \Omega_2, \ldots, \Omega_m\}$ is a discretization of the sheet into finite elements. $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x_2}, \ldots, \mathbf{x}_n\}$ is the discrete set of mesh nodes. $\phi_i = \phi_i(\mathbf{x})$ is the interpolation function associated to the node $\mathbf{x}_i$. $\mathbf{U}^{(t)} = [u(\mathbf{x}_1, t), u(\mathbf{x}_2, t), \ldots, u(\mathbf{x}_n, t)]^T$ and $\mathbf{F}^{(t)} = [f(\mathbf{x}_1, t), f(\mathbf{x}_2, t), \ldots, f(\mathbf{x}_n, t)]^T$ are the nodal values of the temperature and the laser source, respectively. Δt is the simulation timestep.

A case is studied contrasting *(a)* analytic and *(b)* FEA implementations. The case conditions are:

1. Sheet geometry: $0.01\,m \times 0.01\,m \times 0.001\,m$.
2. Material: AISI 304 steel (Table IV-D.2.1).
3. Laser power: $100W$.
4. Laser spot radius: $r = 0.0003\,m$.
5. Traversal laser speed: $0.1\,m/s$.

The laser beam follows an S-shape trajectory. A timestep $\Delta t = 0.0012\,s$ is chosen for the FEA time discretization.



(a) Analytic temperature distribution at $t = 0.12\,s$

(b) Relative error. Analytic vs. FEA solution. Relative error below 3.43%

Figure IV-D.2.4: Analytic temperature and relative error distribution (w.r.t. FEA) for the S-shape laser trajectory

Fig. IV-D.2.4(a) plots the sheet temperature distribution at the end of the analytic simulation ($t = 0.12\,s$) using 100×100 Fourier coefficients. Fig. IV-D.2.4(b) shows the relative error distribution of the analytic model considering the FEA solution as reference. The maximal relative error (3.43%) occurs at the laser spot. Additional experiments increasing the number of Fourier coefficients did not provide any significant improvement in the analytic solution of the current study case.

Domains with sizes above the small one here discussed do not permit termination of the FEA computation. Therefore, a comparison analytic vs. FEA is only possible for exceedingly small domains. Following subsections present study cases with larger domains.

Table IV-D.2.1: Physical parameters for simulation of laser heating of an AISI 304 steel sheet with natural convection at the surface [106]

Parameter	Value
ρ	$8030\,kg/m^3$
c_p	$574\,J/(kg \cdot K)$
k	$20\,W/(m \cdot K)$
R	0
h	$20\,W/(m^2 \cdot K)$
u_∞	$300\,K$

IV-D.2.4.2 Temperature Evaluation Assisted by GPU

The analytic approach presented in section IV-D.2.3, permits a parallel implementation, which this section discusses. As specified in section IV-D.2.3.3, grid sampling is used in order to obtain the temperature at discrete points on the sheet. To illustrate the capabilities of the presented approach, a study case with complex laser trajectory is evaluated on a large (larger than FEA) sheet metal. The following setup is used:

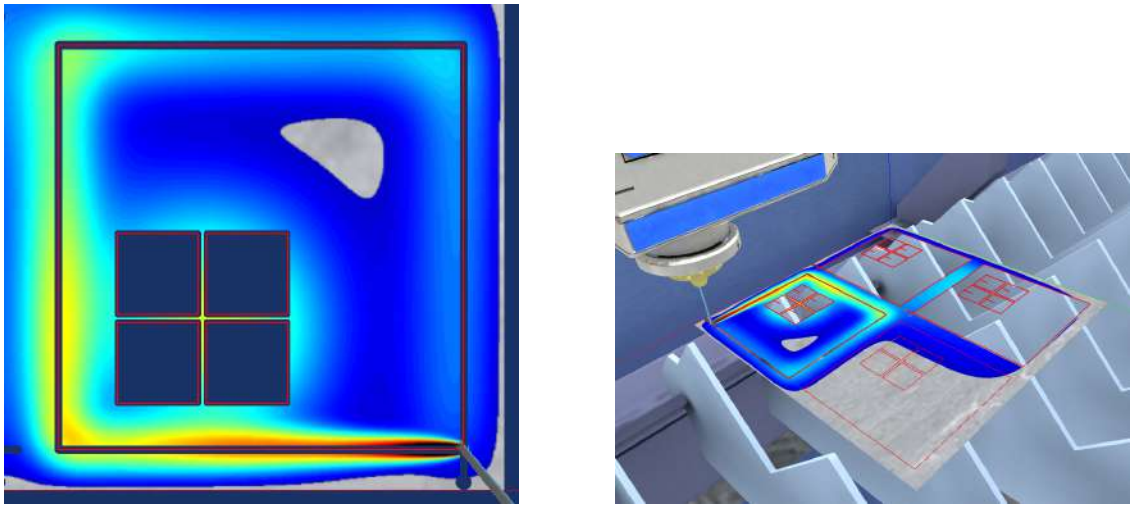
1. Sheet geometry: $0.235\,m \times 0.235\,m \times 0.01\,m$.
2. Material: AISI 304 steel (Table IV-D.2.1).
3. Laser power: $8000W$.
4. Laser spot radius: $r = 0.0001\,m$.

The study presents a custom CAM model to produce a set of nested squares. The resulting CNC program is composed as a combination of cutting and non-cutting instructions:

1. **Machining instructions:** Instructions which move the laser head and cut the sheet. The laser moves at traversal speed $\|\vec{v}\| = 0.028\,m/s$.
2. **Rapid non-cutting movements:** Instructions which move the laser head but do not cut the sheet ($P = 0\,W$). The laser moves at traversal speed $\|\vec{v}\| = 0.183\,m/s$.
3. **Other CNC control instructions:** Sourceless ($P = 0\,W$) and static ($\|\vec{v}\| = 0\,m/s$) instructions (such as powering on and off the laser beam) simulated as delays of $1\,s$.

Table IV-D.2.2: GPU implementation. Execution times (in seconds) for retrieving the temperature at a given timestep t_l, according to Eq. (IV-D.2.3). Computation times of Fourier coefficients are not considered.

Fourier Coeffs.	**Grid Resolution** 64×64	**Grid Resolution** 128×128	**Grid Resolution** 256×256	**Grid Resolution** 512×512	**Grid Resolution** 1024×1024	**Grid Resolution** 2048×2048	**Grid Resolution** 4096×4096
64×64	0.001313	0.002641	0.007977	0.029518	0.115439	0.4318	1.6787
128×128	0.002796	0.008041	0.027954	0.110535	0.414949	1.6098	6.3809
256×256	0.008013	0.027058	0.101933	0.415542	1.591032	6.3067	25.1822
512×512	0.028004	0.101988	0.397375	1.598041	6.281901	25.0937	100.3546
1024×1024	0.105995	0.401026	1.581736	6.327188	25.08374	100.3443	N/A



(a) Temperature texture map

(b) Visualization of the cutting process in the interactive simulator

Figure IV-D.2.5: Simulation of the CNC process integrating the physical (512×512 Fourier coefficients) and the geometric modules (1024×1024 grid points)

As illustrated in section IV-D.2.3.4, the computation time of our analytic algorithm is determined by the number of Fourier coefficients times the number of sample points, which for high Fourier and sampling resolutions may render our algorithm beyond the interactivity needs. However, the evaluation of the sheet temperature is independent for each sampling point (as per Eq. (IV-D.2.3)), thus the temperature distribution can be easily evaluated in parallel. Therefore, the computation time of the temperature becomes determined by the number of coefficients, which ultimately translates to a great amount of memory access operations. Unlike CPUs, current GPUs can evaluate a larger number of sample points in parallel and provide faster and more efficient memory access. In definitive, GPUs are better suited for the temperature evaluation problem. Therefore, a GPGPU implementation of Eq. (3) has been developed using OpenCL [39] as follows: (*1*) The Fourier coefficients are transferred from host memory to the GPU memory. (*2*) A 1D OpenCL kernel is launched to perform the temperature evaluation. The number of threads is determined by the desired number of sample points (resolution), the local work-group size is determined by

Table IV-D.2.3: Modest hardware and software specifications of the machine used to run the the CNC simulations

Item	Specifications
Operating System	Microsoft Windows 10 Pro (64 bits)
Processor	Intel©Core™i5-6500 CPU @3.20GHz
RAM	8.0 GB
GPU	NVIDIA GeForce GTX 960
OpenCL	OpenCL 1.2 CUDA 8.0.0
Compilation Environment	Visual Studio 2013™(x64 with optimization -O2 flags)

the underlying OpenCL implementation in function of the GPU. (*3*) When the kernel computation finalizes the results are transferred from GPU to host memory.

The simulation system is implemented in a Windows 10 platform using C++ with OpenMP support. OpenCL support is provided by the NVIDIA native drivers through the proprietary CUDA API. With a grid sampling of 512×512 and 512×512 coefficients, the temperature evaluation takes about $1.59\,s$ in average (without considering the computation times of the Fourier coefficients) for any given subtrajectory. Increasing the number of coefficients or the number of sampling points increases linearly the temperature evaluation times (see Table IV-D.2.2). Fig. IV-D.2.5 plots the temperature distribution at the middle of the sheet laser cutting simulation, coupling both the analytic and the geometric modules.

The number of coefficients and grid resolution cannot be increased without limit, as the internal GPU memory, registers and available threads are finite. In our test machine (Table IV-D.2.3) increasing the grid resolution and number of coefficients above 2048×2048 and 1024×1024 respectively, reduces significantly our GPU performance, rendering noncomputable results beyond such resolutions (as shown in Table IV-D.2.2 where such combination crashes the GPU computation).

In order to compute the temperature, the Fourier coefficients must be calculated as per Eq. (IV-D.2.5). The independence between Fourier coefficients allows efficient CPU parallelization of this calculation using OpenMP. Fig. IV-D.2.6 plots the computation times of the Fourier coefficients for different Fourier resolutions. Such calculation could be implemented directly in the GPU. However, our experiments show that the CPU times are fast enough to allow the interactivity rate (below 0.15 seconds for the larger resolution).

IV-D.2.4.3 Assessment of Computing Performance

Table IV-D.2.4 presents CPU (with OpenMP support) performance results for the temperature calculation of the test case introduced in section IV-D.2.4.2. Compared to the GPU results (see Table IV-D.2.2), the CPU implementation shows significant performance decrease. As an example, the 256×256 test case (0.102 seconds) runs 150 times faster than the CPU execution (15.877 seconds). As a rule of thumb, we consider interactive simulation if the waiting times for the user lie below 5 seconds. The CPU implementation breaks this rule of thumb for moderate resolutions (up to 128×128 Fourier coefficients and 256×256 grid points) while the GPU allows larger resolutions (up to 256×256 Fourier coefficients and 1024×1024 grid points).

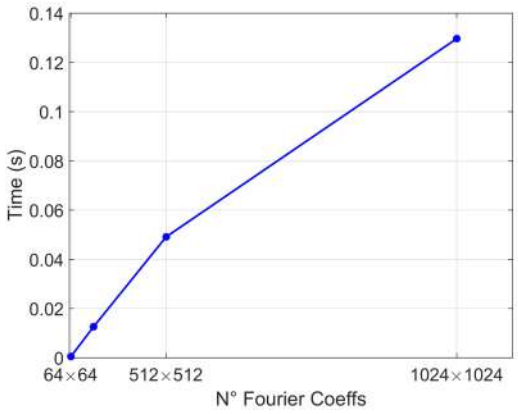


Figure IV-D.2.6: CPU computation times of the Fourier coefficients for a single timestep (as per Eq. (IV-D.2.5))

Table IV-D.2.4: CPU implementation. Execution times (in seconds) for the computation of the temperature at a given timestep t_l, according to Eq. (IV-D.2.3).

Fourier Coeffs.	**Grid Res.** 64×64	**Grid Res.** 128×128	**Grid Res.** 256×256	**Grid Res.** 512×512
64×64	0.0662	0.2630	1.0169	3.9880
128×128	0.2771	1.0788	4.3264	16.8444
256×256	1.1368	4.0004	15.8775	63.9231
512×512	3.9514	15.5803	62.5949	250.4050

In addition, linear FEA for thermal analysis has been implemented in MATLAB with adaptive re-meshing from the MATLAB PDE Toolbox in order to compare the performance between FEA and our analytic (GPU) algorithm. Since FEA simulation becomes unfeasible as the sheet size increases and the laser trajectory becomes more complex [40, 125], the (much smaller) test case presented in section IV-D.2.4.1 is used for such comparison. In our test machine (see Table IV-D.2.3), the FEA computation of the temperature for a sheet discretization of 3586 points and 100 simulation timesteps takes in average 6.3835 seconds (not considering FEA re-meshing time). In contrast, our GPU implementation of the analytic algorithm performs the same trajectory simulation (with 100×100 Fourier coefficients) in 0.3556 seconds (18 times faster).

IV-D.2.4.4 Interactive Simulation of the Laser Cutting Process

Our work efficiently integrates geometry and thermal modules for the simulation of sheet metal CNC laser cutting. We use a contour-based representation to model the geometry of the processed sheet [138, 147], coupled with an analytic solution of the underlying thermal phenomena which generates the corresponding temperature texture. The simulator presents the user a virtual 3D interactive scenario with a fully detailed CNC machine equipped with a virtual laser that will be the target of the machining instructions. In this virtual scene, the sheet metal is rendered with the computed temperature distribution on its surface.

We identify two different use cases in the virtual 3D simulator: *(1)* the user visualizes the machining simulation as a continuous animation to track the overall process and, *(2)* the user inspects a specific timestep of the simulation looking for more detail in the current state of the sheet (e.g. cutting and temperature profile near the laser spot).

In the first scenario, the temperature on the sheet is calculated after each machining movement. Even if some preprocessing must be done, the worst case scenario implies a user request of the whole process as soon as the simulator itself is loaded. The computation process involves the utilization of CPU and GPU resources. The contour-based representation is calculated in the CPU (single-threaded). For each machining instruction, a new updated geometric representation of the sheet metal is generated. At the same time, the Fourier coefficients are calculated in CPU using all its available cores (thanks to OpenMP). Once the geometry and the coefficients are calculated, the temperature field is evaluated in the GPU using a 256×256 grid with 300×300 Fourier coefficients (thanks to OpenCL). This grid resolution is small enough to allow fast and interactive visualization of the machining process (below $0.12\,s.$ as shown in Table IV-D.2.2).

In the second scenario, the simulation is stopped at a given timestep and the user interactively inspects the current state of the machining process. In this scenario, the user might request a more accurate representation of the temperatures on the sheet metal. In such case, the computing time is less penalized in favor of more precise results. Some UI elements enable the user to configure the desired quality of these results. Therefore, if high quality is requested, a 1024×1024 grid sampling is used with 1024×1024 Fourier coefficients. This combination of grid resolution and number of coefficients provide high quality results and it is dependent on the existing GPU hardware. State-of-the-art GPU models might use 2048×2048 or higher grid resolutions within affordable computational time limits (below 5-6 s).

IV-D.2.4.5 Impact in the Design Workflow of CNC Programs

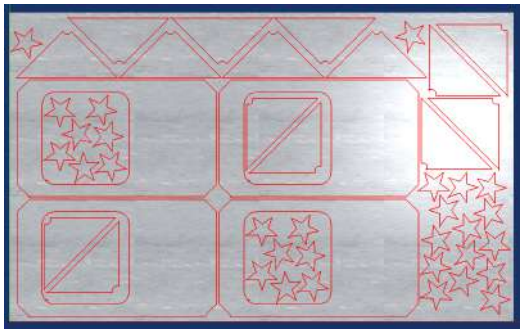
The design process of CNC programs is essentially iterative. The designer receives the list of parts to be produced, their material and any other relevant information. The nesting software produces the arrangement of the parts in any of the available metal sheets in the Manufacturing Execution System (MES) software. This process can be configured by the designer in order to optimize different aspects of the machining process such as: *i)* minimization of the produced scrap, *ii)* minimization of the overall machining time and *iii)* reduction of rapid movements over cut parts (to reduce potential collisions).

As a consequence, professional nesting software provides a variety of options to the designer to fine-tune the produced CNC program. This number of options continuously increases. Therefore, an analysis of different simulated scenarios aids the designer in the parameter selection of the CNC machine for an optimal configuration. As an example, the energy output of the laser must be controlled. The available laser with the highest power is not always the best option to operate on the sheet metal. It might produce better quality cuts with positive economical consequences (such as less machining time), negative economical consequences (such as more energy consumption) or potential side-effects in the quality of the parts near the cut zone. Therefore, the selection of the proper heat source parameters can be optimized with the support of the CNC simulator.

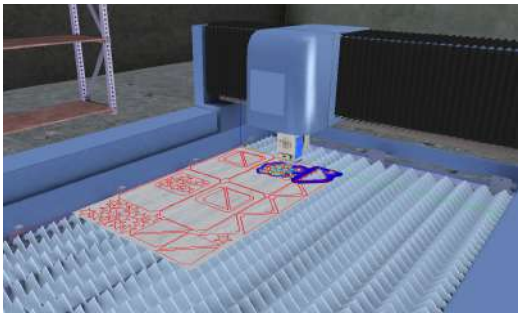
Using high quality simulations with FEA software is precise but computationally expensive [40,125–127,143]. Therefore, the number of test scenarios to be carried out during the design phase becomes limited. On the other hand, utilization of fast simulation software like the one presented in this work, enables the simulation of a considerable number of test scenarios. The design workflow

would be more agile and versatile, providing new optimization opportunities that would led to better CNC programs in terms of quality of the parts, economical benefits, wasted resources and safety of the operators in the factory floor.

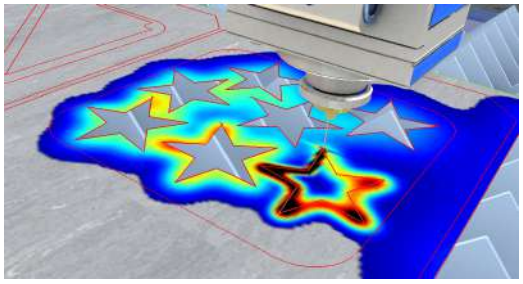
Fig. IV-D.2.7 shows the sheet metal cutting simulator running a real CNC program. The temperature texture on the sheet surface provides useful information to the designer. As an example, heat progation around the recent cut part (star) affects posterior cuts (Fig. IV-D.2.7(c)). Therefore, the designer might consider to modify the arrangement of the star cuts in order to reduce potential rejections of the produced parts in the post-machining inspection.



(a) Nesting of the CNC program



(b) Overall 3D visualization of the CNC simulation



(c) Detailed inspection near a recent cut. Heat affects posterior cuts

Figure IV-D.2.7: The interactive CNC simulator can be used to detect potential problems in the nesting planning due to heat propagation

IV-D.2.5 Conclusions and Future Work

This manuscript implements an integration of a physical and a geometric module for fast and interactive simulation of CNC-based sheet metal laser cutting.

Sections IV-D.2.3.1 and IV-D.2.3.2 presents the analytic solution to the laser heating problem. Although the assumptions of the mathematical model simplify the laser heating phenomenon (constant material properties, no material removal, no phase changes and constrained sheet geometries), the analytic algorithm provides an efficient solution to problems that are very expensive computationally for current FEA methods.

Section IV-D.2.3.3 presents the integration of the geometric and physic modules in the CNC simulator. A grid sampling of the sheet metal is used to calculate the temperature distribution and to create a texture that is overlaid on the geometric representation of the sheet. The goal of the grid sampling is to provide fast and detailed results for visual inspection at an specific timestep using the GPU. A complexity analysis of the analytic approach is presented in section IV-D.2.3.4. The parallelization properties of the mathematical description allow efficient implementation of the algorithm.

A numerical comparison with FEA is presented in section IV-D.2.4.1. With the curved S trajectory path case, the calculated maximum error is 3.43% around the laser spot.

Section IV-D.2.4.2 presents the results for the integrated CNC simulator with the grid sampling technique. The implementation of Equation (IV-D.2.3) in OpenCL allows the utilization of the capabilities of modern GPU hardware to speed up the temperature calculation. Section IV-D.2.4.3 conducts a performance assesment of such GPU implementation against non-GPU analytic implementation and FEA. The GPU implementation performs 150 times faster than the CPU implementation and 18 times faster than linear FEA, allowing our approach to run interactive simulations of complex CNC programs, which are otherwise non-computable in the latter ones.

The analytic model presented in section IV-D.2.3 simplifies by design the thermal phenomena of laser cutting in order to achieve real-time performance. As future work, we aim to (1) introduce nonlinear physical behaviour into the analytical model in the form of nonlinear material properties and material ablation (sheet removal), (2) evaluate the accuracy of the method with experimental data in order to provide error bounds, and (3) use meshing techniques that improve the sheet sampling in order to allow even larger sheet configurations and better temperature resolutions near the laser spot.

Acknowledgements
This work was supported by Vicomtech-IK4, Lantek Business Solutions, EAFIT University and the Basque Government under the GAITEK and HAZITEK programs.

IV-D.3

Fast Analytic Simulation for Multi-Laser Heating of Sheet Metal in GPU

Daniel Mejia-Parra[1,2], Diego Montoya-Zapata[1,2], Ander Arbelaiz[2], Aitor Moreno[2], Jorge Posada[2], Oscar Ruiz-Salguero[1]

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, Medellín, Colombia.

[2] Vicomtech, Donostia / San Sebastián, Spain.

Citation

Daniel Mejia-Parra, Diego Montoya-Zapata, Ander Arbelaiz, Aitor Moreno, Jorge Posada and Oscar Ruiz-Salguero. Fast analytic simulation for multi-laser heating of sheet metal in GPU. *Materials*, **2018**, 11, pp. 2078:1-2078:19. DOI: 10.3390/ma11112078.

Indexing: ISI (Q2), SCOPUS (Q1), Publindex (A2)

Abstract

In the context of multi-beam laser machining, the problem of heat transfer simulation is relevant for thermal-stress analysis and path planning optimization. Currently used methods rely on numerical methods such as Finite Differences or Finite Element Analysis (FEA). These numerical solutions provide precise results at the expense of a high computational cost. To overcome this limitation, this article introduces a fast analytic temperature solution to the multi-beam laser heating problem for any type of laser trajectories and arbitrary sheet sizes. The test runs show that our algorithm produces accurate temperature fields, even for a large number of simultaneous laser beams. The heat transfer module is integrated into an interactive simulation environment for sheet cutting.

Ongoing work addresses thermal stress coupling and efficient models that consider laser ablation.

Keywords: multi-beam laser; heat transfer analysis; fast simulation; GPU; analytic solution.

Glossary

The following abbreviations are used in this manuscript:

FEA / FEM	Finite Element Analysis / Finite Element Method.
GPU	Graphics Processing Unit.
$a, b, \Delta z$	Width, height and thickness of the sheet (m).
$\vec{x}, t$	Spatial $\vec{x} = (x, y) \in [0, a] \times [0, b]$ and temporal $0 \leq t \leq T_f$ (s) coordinates for the simulation.
$u = u(\vec{x}, t)$	Temperature distribution along the sheet at any given time (K).
ρ	Sheet density (kg/m^3).
c_p	Sheet specific heat $(J/(kg\,K))$.
κ	Sheet thermal conductivity $(W/(m\,K))$.
R	Sheet reflectivity $(0 \leq R < 1)$.
$q = q(u)$	Temperature-dependent heat convection on the sheet surface (W/m^2).
u_∞	Ambient temperature K.
h	Natural convection coefficient $(W/(m^2\,K))$.
f_k	Heat input from laser beam k (W/m^2). $k = 1 \dots num\_lasers$.
P_k	Power of laser beam k (W).
r_k	Radius of laser spot k (m).
$\vec{x}_0^k = \vec{x}_0^k(t)$	Laser spot center for laser beam k at time t.
$[t_0^k, t_f^k]$	Simulation time frame in which the laser beam k remains turned on. $0 \leq t_0^k < t_f^k \leq T_f$.
$\vec{v}_k$	Scan speed of laser beam k (m/s).
$F = F(\vec{x}, t)$	Sum of all laser beam heat sources (W/m^2).
$\mathbb{X}_i = \mathbb{X}_i(x)$	Fourier basis function associated to the x coordinate. $i = 1 \dots \infty$.
$\mathbb{Y}_j = \mathbb{Y}_j(y)$	Fourier basis function associated to the y coordinate. $j = 1 \dots \infty$.
$\Theta_{ij} = \Theta_{ij}(t)$	Fourier coefficient associated to basis functions $\mathbb{X}_i$ and $\mathbb{Y}_j$.
$\theta_{ij}^k = \theta_{ij}^k(t)$	Pseudo-Fourier-coefficient associated to the k-th laser beam source.
ω_{ij}	ij-th eigenvalue of the heat operator (Laplacian) on the rectangular sheet.

IV-D.3.1 Introduction

Multi-beam laser heating of sheet metal is a relevant metalworking technique which has arisen interest of researchers in the last years. In contrast to single-beam heating, multi-beam heating provides two main advantages to the former: (1) the ability to process different locations of the sheet simultaneously [148], and (2) control of thermal stress levels by specific multi-beam configurations [149]. Industrial applications of multi-beam heating of sheet metal include laser forming

and bending, laser cutting and additive manufacturing.

Thermal simulation is crucial for temperature and stress analysis of manufactured pieces. An adequate selection of laser parameters and a correct path planning allows to improve the efficiency of the process and minimizes material damage and waste. Current simulation approaches rely on numerical schemes which require fine geometry and time discretizations. Such discretizations imply high computational costs, which limit the application of these approaches in real manufacturing scenarios with large time / space domains and complex laser trajectories.

This manuscript presents a simulation approach for the multi-beam laser heating problem based on an analytic solution to the heat equation on rectangular domains. This analytic solution does not require a mesh nor a fine time discretization to solve the problem. As a consequence, our algorithm is able to run complex simulations with large time / space domains and complex multi-laser trajectories at interactive time rates. Furthermore, each laser beam trajectory is allowed to be independent from the others, with different time-dependent parameters, trajectories and time frames (i.e. each laser beam can be turned on / off independently at any point of the simulation).

The remainder of this manuscript is organized as follows: Section IV-D.3.2 discusses the relevant literature. Section IV-D.3.3 presents the mathematical model and describes the implementation of the proposed algorithm. Section IV-D.3.4 discusses the obtained results for different test cases. Finally, Section IV-D.3.5 presents the conclusions and introduces the future work.

IV-D.3.2 Literature review

IV-D.3.2.1 Multi-beam Single Trajectory vs. Multiple-Trajectory Simultaneous Laser Heating

There are currently two main applications for multi-laser heating in laser machining: (1) single-trajectory multi-laser heating and, (2) multi-trajectory multi-laser heating.

In single-trajectory multi-beam laser heating, a leading laser is followed by a pattern of secondary finishing ones, all in the same trajectory (or with minimum spatial offset). Experimental evidence has shown that such a configuration reduces the thermal stresses produced by the laser beams in laser cutting (compared to single-beam cutting) [150]. Furthermore, specific configurations of the laser beams have shown to reduce the required pressure of assisting gas [151]. Each laser beam may be produced by an independent source [152] or by diffraction of a single beam source [153].

On the other hand, in multi-trajectory heating, each laser beam follows an independent trajectory [148]. Multi-trajectory heating is relevant as it improves the machining times by processing different zones of the sheet at the same time. This manuscript focuses on the simulation of multi-trajectory laser heating.

IV-D.3.2.2 Thermal Simulation for Sheet Metal Laser Heating

The problem of laser beam heating simulation has been widely researched for single-beam applications. Numerical methods are the most common simulation approach. Methods such as the Finite Differences Method (FDM) [117–120] and the Boundary Element Method (BEM) [121, 122] have been used in the literature to study the thermal behaviour of sheet laser heating. However, these methods impose several numerical limitations such as dense rectangular grids in the case of FDM and non-sparse linear systems for BEM, requiring a high amount of computational resources even for simple problems.

The Finite Element Method (FEM) is a standard numerical approach for the simulation of physical problems. Non-linear FEM has been applied to study the thermal / stress behaviour of the single-beam laser heating of rectangular sheets [154–157]. In contrast to FDM and BEM, FEM works by discretizing the domain using different types of meshes which allow fine discretizations (high level of detail) near interest zones (laser spot and trajectory) and coarse resolutions in other zones resulting in less expensive systems of equations. To address laser ablation and material removal, methods such as element birth and death [113], volume fractions [140], temperature thresholds [114, 116, 158], and the enthalpy method [141, 154–157], are coupled to the different numerical schemes.

The Finite Volume Method (FVM) has been recently introduced in the literature for the study of laser ablation and sheet heating [123, 124]. Contrary to previous numerical methods, the FVM allows to accurately model and simulate interactions between the laser beam and the sheet in its physical states: solid, liquid and gas. However, these interactions are highly non-linear and as a consequence, computationally expensive.

Numerical methods provide tools to simulate non-linear interactions, phase changes, laser ablation and material removal. However, they are also computationally expensive, limiting the applications of the algorithms in real manufacturing scenarios with complex laser trajectories and large space / time domains. Analytical methods do not have such limitations, allowing fast simulation of complex problems at the cost of some simplifications of the physical model. Uni-dimensional analytic models have been implemented for the simulation of laser drilling processes for static laser beams [128, 129]. A 3D model for laser heating of sheet metal for straight line trajectories is presented in [132], while 2D models for arbitrary laser trajectories have been presented in [76, 133].

Despite the large amount of literature concerning single-beam laser heating, simulation for multi-beam laser processing has been rarely studied. In [152,159], the FDM is used to analyze the thermal and structural impact of two independent laser beams melting a sheet metal. On the other hand, in [149,160] a thermal / stress analysis of multi-beam laser heating is performed using FEM. These simulations show that multi-beam heating reduce the thermal stresses afflicted to the material. A semi-analytic model for the steady multi-beam laser heating problem is presented in [161]. This pseudo-analytic model is also implemented inside an optimization algorithm which estimates the best laser parameters for a given manufacturing process. The use of analytic solutions is crucial for optimization due to the optimization process being expensive per se, requiring multiple evaluations of the temperature fields with different laser parameters and/or trajectories.

IV-D.3.2.3 Conclusions of the literature review

Contrary to single-beam, multi-beam heating is scarce in the literature. Numerical (as opposed to analytical) methods are computationally expensive. As a consequence, these methods are unable to simulate real world sheet sizes and laser trajectories.

This manuscript offers the implementation of an analytic Fourier-based method to simulate multi-trajectory laser heating. The characteristics of the implemented method are: (a) constant material properties, (b) natural convection, (c) simplification of the sheet into a 2D domain, (d) transient (time-history) temperature, (e) multiple laser head configurations, (f) independent laser trajectories (with independent parameters), (g) arbitrary power on/off time history on each trajectory.

In contrast to numerical methods (such as FEA), the accuracy of our method is not affected by the time and space discretizations, allowing large time steps and arbitrary meshes (such as

triangular meshes, grids, a polyline or a set of sampled points on the sheet) which speeds up the solving process. Finally, the algorithm has been implemented into an interactive laser cutting simulation environment for real time assessment of the laser cutting process in real manufacturing scenarios.

IV-D.3.3 Methodology

IV-D.3.3.1 Heat Transfer Equation for Multi-beam Laser Heating

The temperature distribution $u = u(\vec{x}, t)$ of a 2D rectangular sheet satisfies the following partial differential equation:

$$\begin{aligned} \rho c_p \frac{\partial u}{\partial t} - \nabla \cdot (\kappa \nabla u) &= \frac{F - q}{\Delta z} \\ q &= h(u - u_\infty) \end{aligned} \tag{IV-D.3.1}$$

where ρ, c_p and κ are the material density, specific heat and thermal conductivity, respectively. $\nabla \cdot \nabla = \partial^2/\partial^2 x + \partial^2/\partial^2 y$ is the 2D Laplace operator. Δz is the thickness of the sheet. $F = F(\vec{x}, t)$ is the set of surface heat sources affecting the sheet, and $q = q(u)$ is the temperature-dependent convection on the sheet surface with its respective convection coefficient $h \geq 0$ and ambient temperature $u_\infty \in \mathbb{R}$. For the previous partial differential equation, the following boundary and initial conditions are imposed on the sheet:

$$\begin{aligned} u|_{x=0} = u|_{x=a} = u|_{y=0} = u|_{y=b} &= u_\infty \\ u(\vec{x}, 0) &= u_\infty \end{aligned} \tag{IV-D.3.2}$$

For the multi-beam approach, the set of heat sources F is defined as the sum of the heat inputs of each laser beam f_k:

$$\begin{aligned} F(\vec{x}, t) &= \sum_{k=1}^{num\_lasers} f_k(\vec{x}, t) \\ f_k(\vec{x}, t) &= \begin{cases} \frac{P_k(1-R)}{\pi r_k^2}, & \|\vec{x} - \vec{x}_0^k(t)\|_\infty < \frac{r_k\sqrt{\pi}}{2} \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{IV-D.3.3}$$

where $P_k \geq 0$ is the power of the laser beam, $r_k > 0$ is the radius of the laser spot and $\vec{x}_0^k(t) \in \mathbb{R}^2$ is the center of the laser spot at time t. $0 \leq R < 1$ is the reflectivity of the material. $\|\vec{x}\|_\infty = \max(x, y)$ is the infinity norm in $\mathbb{R}^2$. The above laser model transforms the circular laser spot with radius r_k to its equivalent squared spot with area πr_k^2. We apply such transformation in order to develop the analytic solution in the next section. Fig. IV-D.3.1 presents an scheme of the multi-beam laser heating problem.

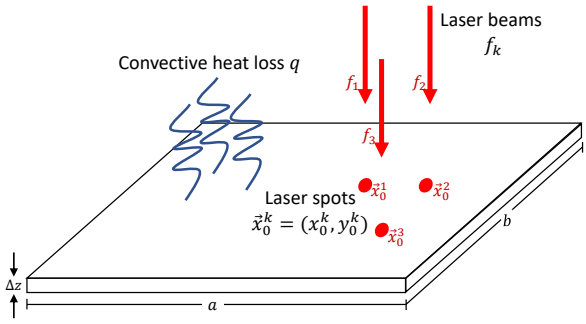


Figure IV-D.3.1: Multi-beam laser heating scheme. The sheet surface is heated by a set of laser beams $f_1, f_2, \ldots$ and cooled down due to natural convection q.

IV-D.3.3.2 Analytic Solution

Following the same procedure as in [162], the temperature u on the rectangular sheet can be expressed as a linear combination of Fourier functions:

$$u(\vec{x}, t) = u_\infty + \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \Theta_{ij}(t) \mathbb{X}_i(x) \mathbb{X}_j(y), \tag{IV-D.3.4}$$

with basis:

$$\begin{aligned} \mathbb{X}_i(x) &= \sin \frac{i\pi x}{a} \\ \mathbb{Y}_i(y) &= \sin \frac{j\pi y}{b} \end{aligned} \tag{IV-D.3.5}$$

Applying separation of variables, the Fourier coefficients Θ_{ij} from Eq. (IV-D.3.4) can be expressed as a sum of the pseudo-coefficients θ_{ij}^k of each independent heat source:

$$\Theta_{ij}(t) = \sum_{k=1}^{num\_lasers} \theta_{ij}^k(t) \tag{IV-D.3.6}$$

where each laser beam f_k defines its respective pseudo-coefficient θ_{ij}^k as follows:

$$\theta_{ij}^k(t) = \theta_{ij}^k(t_0) e^{-\omega_{ij}(t-t_0)} + \frac{4}{ab\rho c_p \Delta z} \int_{t_0}^{t} \int_0^b \int_0^a f_k(\vec{x}, \tau) \mathbb{X}_i(x) \mathbb{Y}_j(y) e^{-\omega_{ij}(t-\tau)} dx dy d\tau \tag{IV-D.3.7}$$

with corresponding Laplacian eigenvalues:

$$\omega_{ij} = \frac{\kappa}{\rho c_p} \left(\frac{i^2\pi^2}{a^2} + \frac{j^2\pi^2}{b^2} \right) + \frac{h}{\rho c_p \Delta z} \tag{IV-D.3.8}$$

Eq. (IV-D.3.7) is written recursively in terms of a previous known solution $\theta_{ij}^k(t_0)$. Therefore, each laser trajectory is discretized as a piece-wise linear trajectory $\vec{x}_0^k = [\vec{x}_0^k(0), \vec{x}_0^k(t_1), \vec{x}_0^k(t_2), \dots]$. Finally, the closed form for the integral term in Eq. (IV-D.3.7) has been already presented in [162].

IV-D.3.3.3 Algorithm

Fig. IV-D.3.2 presents a diagram of the simulation algorithm based on the analytic solution presented in Section IV-D.3.3.2. Each step of the algorithm is discussed in detail below:

1. **Discretize laser trajectories:** As discussed in Sect. IV-D.3.3.2, the laser beam trajectories $\vec{x}_0^k(t)$ are discretized as sequences of piece-wise linear trajectories $[\vec{x}_0^k(0), \vec{x}_0^k(t_1), \dots, \vec{x}_0^k(T_f)]$, as described in [162]. The only requirement for this discretization is the fact that all laser beam trajectories must share the same time discretization, i.e., $t_0, t_1, \dots, T_f$ are the same for all trajectories $\vec{x}_0^k$.

2. **Compute the Laplacian eigenvalues:** The Laplacian eigenvalues of the sheet are computed as per Eq. (IV-D.3.8). Since the eigenvalues ω_{ij} are time-independent, this step is performed before the simulation loop starts.

3. **Initialize time and sheet temperature:** The simulation time is initialized to $t_0 \leftarrow 0$. In order to satisfy the initial temperature condition $u(0) = u_\infty$, the pseudo coefficients are initialized as $\theta_{ij}^k(0) = 0$ (see Eq. (IV-D.3.4)).

4. **Update current time t:** The current simulation time $t = t_{l+1}$ is updated according to the previous time $t_0 = t_l$, in concordance with the discretization of trajectories from step 1.

5. **For each laser beam k:** This inner loop computes the pseudo-coefficients $\theta_{ij}^k(t)$ for each laser beam ($k = 1 \dots num\_lasers$).

6. **QUESTION: Is laser beam k turned on?:** This step allows to simulate asynchronous laser beams by asking at the current time t if the laser is turned on / off. Therefore, each laser beam might have its own internal time frame $[t_0^k, t_f^k]$, different from the simulation time frame $[0, T_f]$.

7. **Set power P_k / Set null power $P_k \leftarrow 0$:** In the previous step, the program checks the state (on / off) of the current laser beam k. The laser is turned on by the simulation by setting its corresponding power input P_k. In the case of the laser being turned off, the simulation simply sets its power to 0.

8. **Compute the pseudo-coefficients for each laser beam:** The pseudo-coefficiens in Eq. (IV-D.3.7) are solved analytically for each laser beam as described in [162]. The number of coefficients computed is truncated to num_coeffs.

9. **QUESTION: $k < num\_lasers$?:** Check if the pseudo-coefficients have been computed for all the laser beams.

10. **Compute the Fourier coefficients as per Eq.** (IV-D.3.6)**:** This step computes the real Fourier coefficents $\Theta_{ij}(t)$ from the pseudo-coefficients $\theta_{ij}^k(t)$ as per Eq. (IV-D.3.6).

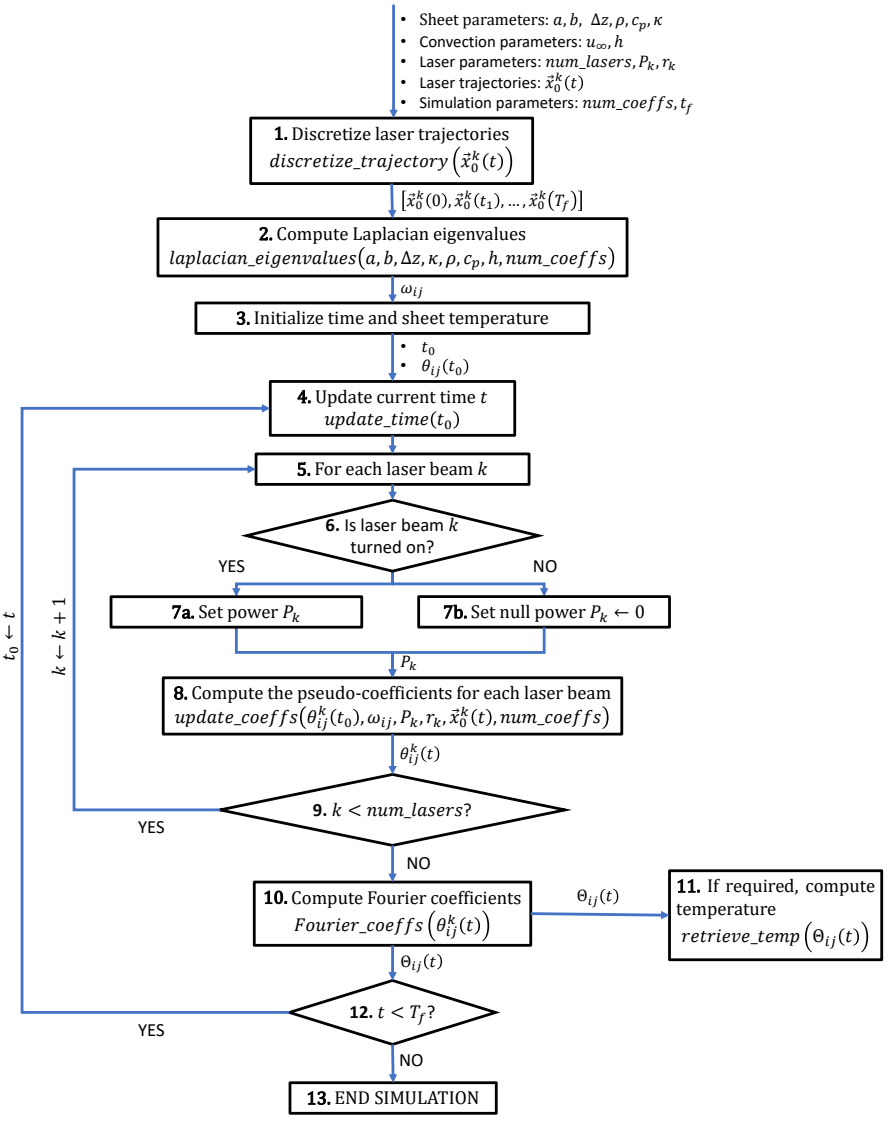


Figure IV-D.3.2: Diagram of the analytic multi-laser simulation algorithm.

11. **If required, compute temperature:** The temperature field is computed on a set of discrete points sampled on the sheet $[(x_0, y_0), (x_1, y_1) \dots]$ as per Eq (IV-D.3.4). This step is optional since the result may be stored in the frequency domain ($\Theta_{ij}(t)$). Therefore, the temperature is made available only when requested by the user, allowing to skip iterations of no interest and improving the performance in the process.

12. $t < T_f$**?:** Check if the simulation has reached the final step.

13. **END SIMULATION**

There are several concepts in the previous algorithm that help to improve its efficiency, crucial for the interactive nature of the simulation:

1. In step 1 of the previous algorithm, the curved laser beam trajectories $\vec{x}_0^k(t)$ are discretized as piece-wise linear ones $[\vec{x}_0^k(0), \vec{x}_0^k(t_1), \dots, \vec{x}_0^k(T_f)]$, which inherently produces the time discretization $[0, t_1, \dots, T_f]$. This time discretization does not affect the numerical accuracy of the temperature solution. Therefore, as opposed to most numerical methods (namely FEA), the time step size $\Delta t = t_{l+1} - t_l$ of our algorithm can be arbitrarily large.

2. Step 6 allows turning on / off any laser beam at any point of the simulation. In addition, the algorithm allows to change any laser parameters at will, resulting in time-dependent parameters $P_k(t)$ (laser power) and $r_k(t)$ (spot radius). For simplicity, this manuscript uses constant parameters.

3. The complete information of the solution is stored in the frequency domain (step 10) and temperature data is computed only when requested. Therefore, the user requests the temperature only at specific times and in specific zones (i.e. at the middle or the end of the simulation). Furthermore, since the space discretization does not affect the solution, any sheet sampling can be used to render the temperature (rectangular grid, triangular mesh, a curve or a single point in the sheet).

4. In step 10, each pseudo-coefficient θ_{ij}^k is independent from the rest of the pseudo-coefficients (Eq. (IV-D.3.6)). Similarly, in step 11, the temperature value u at a given point $\vec{x}$ is independent from the temperature on the rest of the sheet (Eq. (IV-D.3.4)). Therefore, the computation in both of these steps is parallelized.

IV-D.3.4 Results

This section presents the simulation results of our algorithm. For all the simulations, a Fourier discretization of 512×512 coefficients and a spatial (grid) discretization of 512×512 points are used. These resolutions have been chosen as they have shown in our experiments enough accuracy within our desired execution time ranges (see Sections IV-D.3.4.1 and IV-D.3.4.3, respectively).

Table IV-D.3.1 presents the geometric and physical parameters of the sheet used in the simulations. The first study case presents two different laser beams heating the sheet simultaneously. The first laser beam follows a star-shaped trajectory on the sheet while the second laser beam follows a rectangular trajectory. Table IV-D.3.2 presents the parameters of each laser beam. As discussed in Section IV-D.3.3, our algorithm not only enables different parameters for multiple lasers (path, power, speed, spot radius), but it also allows independent time frames for each trajectory.

Table IV-D.3.1: Heat transfer parameters for the simulations (same as in [76, 163]).

PARAMETER	VALUE
Geometry	
a	$0.01\,m$
b	$0.01\,m$
Δz	$0.001\,m$
Material	**AISI 304 Steel**
ρ	$8030\,kg/m^3$
c_p	$574\,J/(kg \cdot K)$
κ	$20\,W/(m \cdot K)$
R	0
Convection Type	**Natural**
h	$20\,W/(m^2 \cdot K)$
u_∞	$300\,K$

Table IV-D.3.2: Parameters for the simultaneous laser heating trajectories of Fig. IV-D.3.3(a)

Parameter	Star trajectory	Square trajectory
t_0^k	$0.00\,s$	$0.06\,s$
t_f^k	$0.16\,s$	$0.13\,s$
P_k	$100\,W$	$200\,W$
r_k	$0.0003\,m$	$0.0005\,m$
$\|\vec{v}_k\|$	$0.1\,\frac{m}{s}$	$0.2\,\frac{m}{s}$

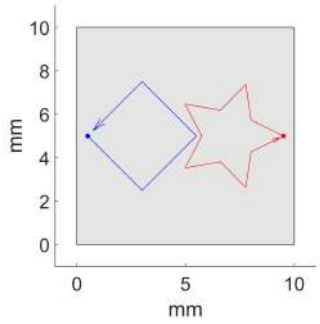
Fig. IV-D.3.3 plots the simulation results for two laser beams heating the sheet surface. Fig. IV-D.3.3(a) shows the star and square laser trajectories planned on the sheet. The laser beam parameters for each trajectory are described in Table IV-D.3.2. At the beginning of the simulation, a unique laser beam (star) heats the surface (Fig. IV-D.3.3(b)). As discussed in Sect. IV-D.3.3.3, our algorithm allows independent time frames for the multiple laser beams. Therefore, the second laser is introduced in the simulation at $t = 0.065\,s$ (Fig IV-D.3.3(c)). Fig. IV-D.3.3(d) plots the temperature when the two lasers are near each other (the trajectories do not intersect). Finally, both laser beam trajectories end at different steps: $t_f^k = 0.13\,s$ for the square trajectory (Fig. IV-D.3.3(e)) and $t_f^k = 0.16\,s$ for the star trajectory (Fig. IV-D.3.3(f)).

IV-D.3.4.1 Comparison with FEA

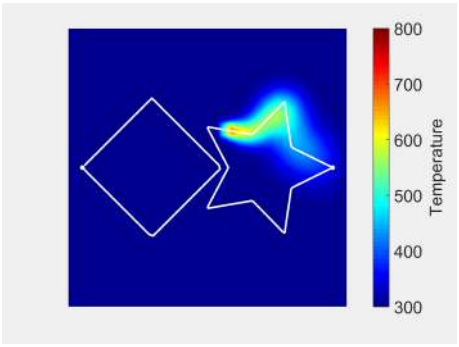
To validate the analytic approach, FEA simulation is performed. The FEA linear system for Eq. (IV-D.3.1) becomes [162]:

$$\left[\left(\frac{\rho c_p}{\Delta t} + \frac{h}{\Delta z}\right)\mathbf{M} + \kappa\mathbf{L}\right]\mathbf{U}^{(t+\Delta t)} = \mathbf{M}\left(\frac{\rho c_p}{\Delta t}\mathbf{U}^{(t)} + \frac{1}{\Delta z}\sum_{k=1}^{num\_lasers}\int_t^{t+\Delta t}\mathbf{F}_k^{(\tau)}d\tau + \frac{h}{\Delta z}u_\infty\right) \tag{IV-D.3.9}$$

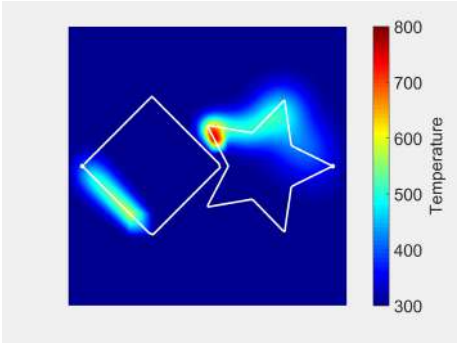
where $\mathbf{U}$ and $\mathbf{F}_k$ are the vectors of temperature and heat sources sampled on the sheet, and $\mathbf{L}$ and $\mathbf{M}$ are the stiffness and mass matrices, respectively [162].



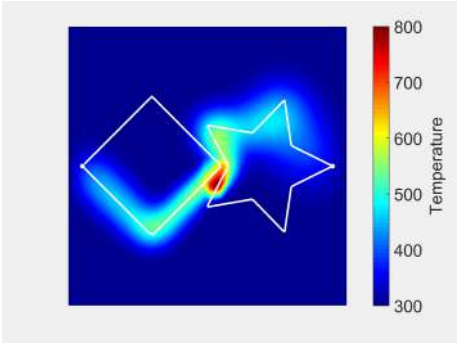
(a) Laser trajectories for the simulation



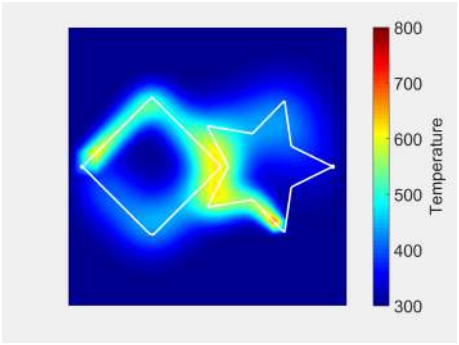
(b) $t = 0.060\,s$. Only Star trajectory occurs.



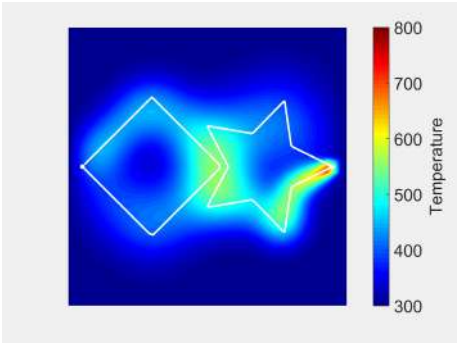
(c) $t = 0.075\,s$. Square trajectory enters the simulation.



(d) $t = 0.094\,s$. Lasers are simultaneously near each other.

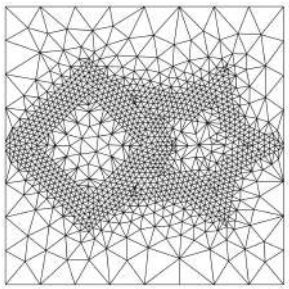


(e) $t = 0.13\,s$. Square trajectory finishes. Star trajectory continues.

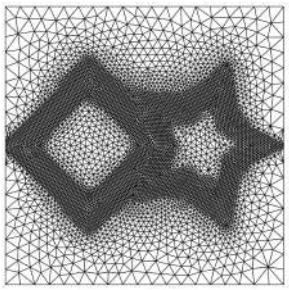


(f) $t = 0.166\,s$. Star trajectory finishes.

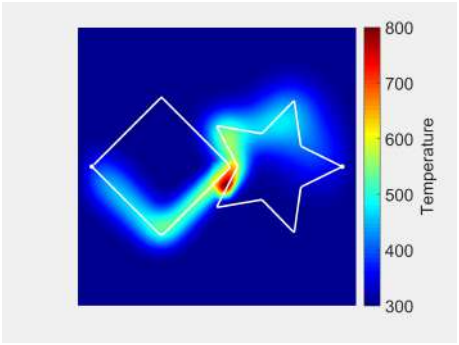
Figure IV-D.3.3: Laser trajectories and sheet temperature history for simultaneous diverse shape laser trajectories.



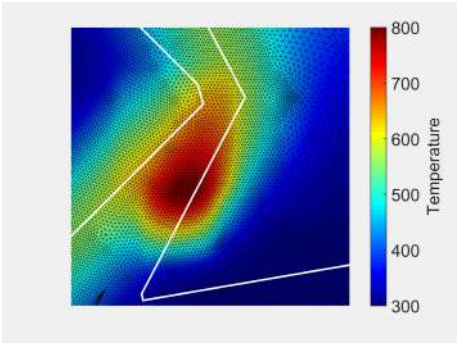
(a) Coarse FEA mesh (2.1k triangles)



(b) Intermediate mesh refinement at laser trajectory (13k triangles)



(c) $t = 0.094\,s$. FEA temperature.



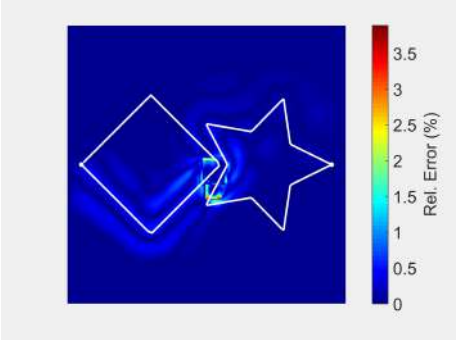
(d) $10x$ zoom on Fig IV-D.3.4(c)

Figure IV-D.3.4: ANSYS FEA simulation results for the same test case presented in Fig. IV-D.3.3

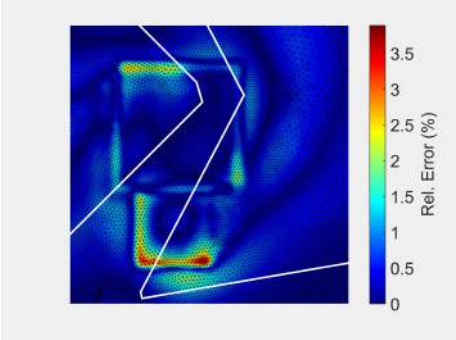
The software ANSYS is used to perform the FEA simulations. ANSYS element SHELL131 is employed. Element thickness and material properties are set as per Table IV-D.3.1. The elements are configured to have a constant temperature along the thickness and to evaluate convection at the sheet surface, so as per Eq. (IV-D.3.1). To represent the area heated by the laser beams at every time step, ANSYS surface loads (Heat Fluxes) are applied on the FEA elements that lie inside the heated zone.

Fig. IV-D.3.4 plots the FEA results for the study case presented in Table IV-D.3.2. The mesh of the domain is generated so that it is more dense in the neighborhoods of the laser trajectories. Fig. IV-D.3.4(a) shows the initial triangular mesh computed using the FEA software which is then refined several times (Fig. IV-D.3.4(b)) to improve the numerical accuracy of the solution. After five re-meshing iterations, the final mesh contains 126k triangles and 63k nodes. Fig. IV-D.3.4(c) plots the FEA temperature at the moment the two laser beams get closer to each other ($t = 0.094\,s$). The temperature peak is in the middle of the two trajectories (Fig. IV-D.3.4(d)), due to both paths not intersecting each other.

Fig. IV-D.3.5 plots the relative error distribution of our analytic solution (Fig. IV-D.3.3(d)) with respect to FEA (Fig. IV-D.3.4(c)). The maximum relative error is 3.9%, located around the laser spots (Fig. IV-D.3.5(b)). The small square shape of the error in Fig. IV-D.3.5(b) is due to



(a) Relative error. Analytic solution w.r.t. FEA (max. relative error: 3.9%).



(b) $10x$ zoom on Fig IV-D.3.5(a)

Figure IV-D.3.5: Appraisal of our analytic solution (Fig IV-D.3.3(d)) vs the FEA solution (Fig IV-D.3.4(c))

the squared laser model presented in Eq. (IV-D.3.3).

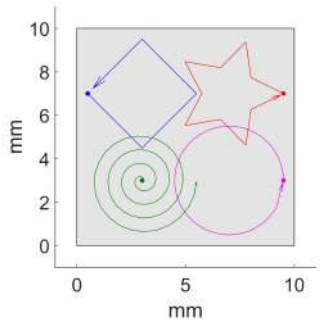
IV-D.3.4.2 Multiple Laser Beams

The algorithm introduced in this manuscript allows multiple laser beams heating the surface at the same time. This section presents additional experiments with more than just two laser beams. Fig. IV-D.3.6 presents a study case with 4 simultaneous laser beams drawing different shapes on the sheet: a square, a star, a spiral and a circle trajectory (Fig. IV-D.3.6(a)). All the laser beams have the same parameters (power and radius) and start at the same time (Fig. IV-D.3.6(b)). However, they do not finish at the same time (Fig. IV-D.3.6(c)).

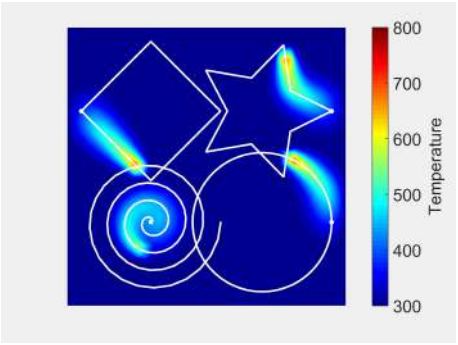
As discussed in Section IV-D.3.3.3, our algorithm allows to define different independent time frames for each laser beam by turning on/off specific laser beams. Such approach even allows to turn off all the laser beams and continue the simulation. Fig. IV-D.3.6(d) illustrates this approach by continuing the simulation after all the laser beams have finished their trajectories, where only thermal conduction and thermal convection are taken into account.

In order to visually compare our algorithm with other simulation approaches in the literature, the study case presented in [160] is replicated in this manuscript (Fig. IV-D.3.7). In this study case, 7 simultaneous laser are distributed uniformly in the y-axis. Each laser beam follows a straight line trajectory and together they draw an arc in the heating front (Figs. IV-D.3.7(a) and IV-D.3.7(b)). Our algorithm is able to produce similar results to [160] despite the simplification of the analytic model (Figs. IV-D.3.7(c) and IV-D.3.7(d)).

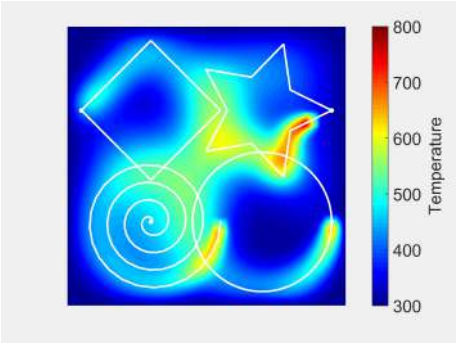
Fig. IV-D.3.8 plots the temperature distribution along the arc for different number of laser beams. As the number of laser beams increase (and the arc length remains the same), the oscillations of the temperature are mitigated and the temperature increases. Such a result is consistent with the analysis presented in [160]. As discussed in Section IV-D.3.3.3, our algorithm allows to compute easily the temperature along the arc without even requiring to calculate the temperature on the rest of the sheet, which improves the computation time for this particular case.



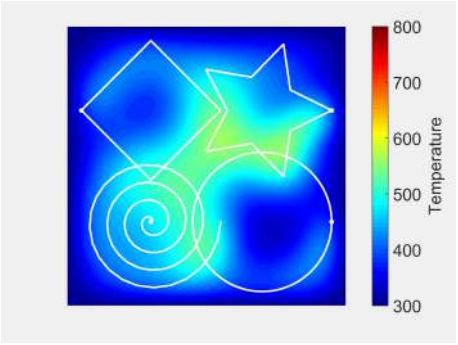
(a) Planned laser trajectories: Square, Star, Spiral, Circle



(b) $t = 0.03\,s$. All trajectories start at $t = 0\,s$.

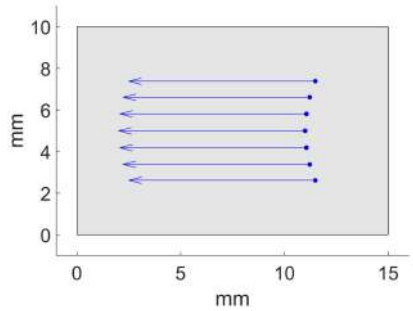


(c) $t = 0.158\,s$. Square and Circle trajectories finished.

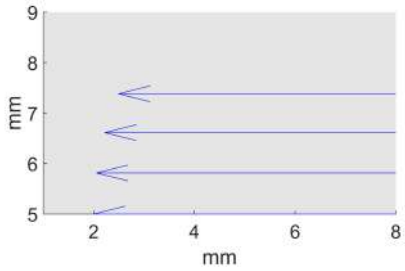


(d) $t = 0.2\,s$. All trajectories finished. The sheet cools down.

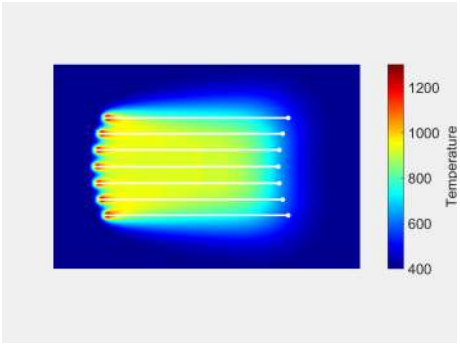
Figure IV-D.3.6: Laser trajectories and temperature history for time and space overlapping trajectories



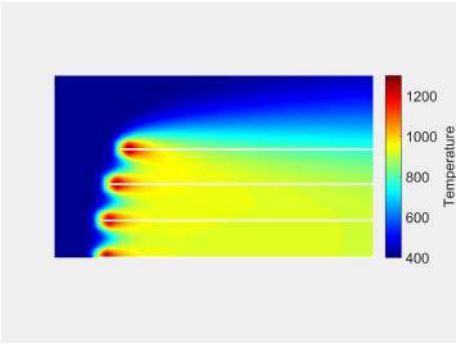
(a) Complete sheet. Simultaneous linear trajectories based on [160].



(b) Laser trajectories (zoom near laser spots).



(c) Temperature map (full sheet).



(d) Temperature map (zoom).

Figure IV-D.3.7: Laser trajectories and temperature results for the simulation case presented in [160]. Similar simulation parameters have been used to replicate the experiment.

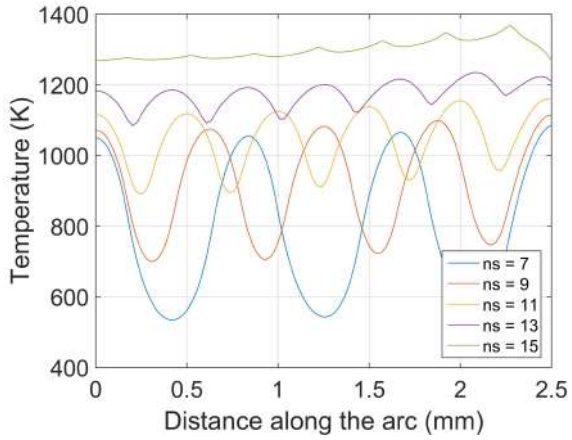


Figure IV-D.3.8: Temperature along the arc for the simulation presented in Fig. IV-D.3.7. Our results are similar to those in [160]. *ns* represents the number of lasers.

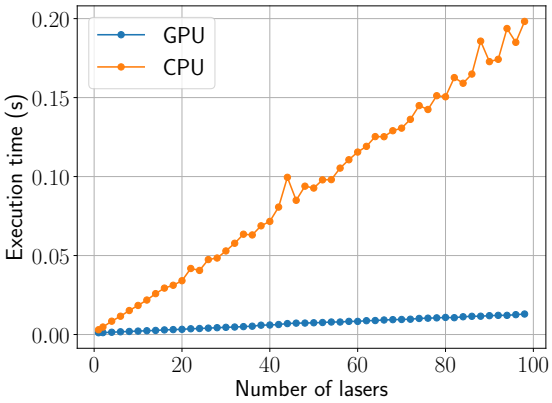


Figure IV-D.3.9: Comparison of CPU and GPU execution times (s) for the computation of 512×512 Fourier coefficients as the number of lasers increase

IV-D.3.4.3 Performance Assessment

This section evaluates the performance of our algorithm under hardware-accelerated (GPU) and non-accelerated (CPU) platforms. The presented algorithm has been implemented using the OpenCL framework to create an optimized solution that targets both CPU and GPU. On systems where only a CPU is available, our implementation makes use of multi-core parallelization and vectorization to speed-up computation. On systems where a GPU is available, the memory hierarchy can be explicitly controlled using the OpenCL API. The workload is divided into small groups, in order to exploit reuse of computed data using local memory. In this manner, a high speed-up is achieved due to both efficient use of memory and massive parallelization.

The performance results have been measured with the following test platform: A desktop PC using Windows 10 OS with an Intel i5-6500 (CPU), 8GB RAM and NVIDIA GTX 960 graphics card. Our algorithm is able to simulate any number of laser beams. Fig. IV-D.3.9 plots the execution times for the computation of the Fourier coefficients as a function of the number of laser beams. The figure compares the execution times of the CPU against the GPU to compute 512×512 Fourier coefficients. The computational cost increases with a large slope in the CPU approach while being nearly constant in the GPU approach. The more laser beams are added, the more it benefits from GPU parallelization. In addition, the computation of the Fourier coefficients in the GPU is preferable. In this way, there is no need to transfer the coefficients back and forth from host-to-device on each iteration since they always stay in GPU memory. For this analysis, a single time step has been considered instead of the whole simulation.

Fig. IV-D.3.10 shows the computation times of the temperature retrieval in a mesh grid as a function of the grid size. While the curves for Fourier resolutions (number of coeffs) below 512×512 display a computational cost relatively low ($< 0.5\,s$) for any spatial resolution, Fourier resolutions above that point (1024×1024) become expensive ($> 0.5\,s$) for our simulation purposes. Therefore, on our test platform, we have observed that a good balance between accuracy and computation time can be achieved by setting a resolution of 512×512 for both the frequency (Fourier) and spatial (grid) domains.

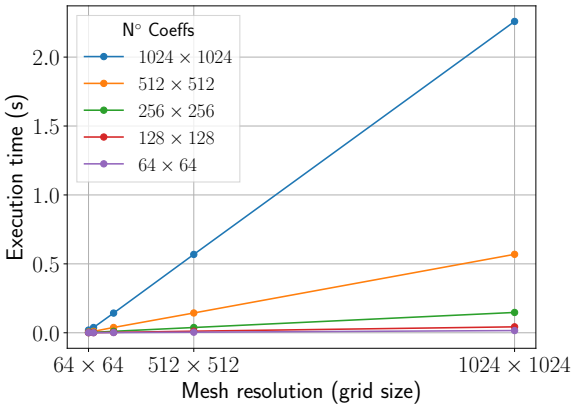


Figure IV-D.3.10: Execution times (s) for the computation of the temperature as the resolution (mesh size) increases. Results for different number of Fourier coefficients are presented.

Additionally, since the computation of the temperature is not compulsory at each iteration of the algorithm (as discussed in Section IV-D.3.3.3), the simulation may ignore irrelevant time steps (defined by the user). Moreover, the independence of the algorithm accuracy from the spatial discretization allows the user to specify specific domains (e.g. a sheet section, a curve or a single point) without requiring the whole sheet temperature. These aspects improve the computation efficiency of the simulation for specific requirements of the user.

IV-D.3.4.4 Integration within an Interactive Laser Cutting Simulation Environment

Nowadays, multi-laser machines with multi-trajectory capabilities do not represent a significant share of the market. The current state-of-the-art laser cutting machines can be divided in three main groups: (1) multi-laser machines with independent and disjoint working areas for each laser head, (2) multi-laser machines with parallel heads working simultaneously and (3) multi-laser machines with fully individual and autonomous laser heads (whose trajectories may intersect or get close enough). The first scenario can be reduced to a collection of mono-laser machining scenarios, since the individual working areas for each laser head avoid interference with the neighbouring laser heads and their heat effects. Therefore, the approach introduced in [162] is still valid to simulate the temperature for each laser head of these multi-laser machines.

In the second type of multi-laser machines, all laser heads receive the same trajectories and machining commands, but their separation or offset can be setup and reconfigured during the machining. Additionally, each laser head can be switched off and on individually. The offset between the heads does not guarantee disjoint working zones. Therefore, the simulation approach presented in this work is used to address the potential interference between the multiple heat sources.

The last type of multi-laser machines use mirrors and lenses to move the spot of the available laser heads. Each laser spot can be commanded independently of the others, even allowing two or

more spots to converge at the same physical point on the sheet metal. The methodology presented in this work is used to simulated such scenarios.

In the context of the second scenario, this section presents an interactive simulator of a laser cutting machine with three laser heads. All laser heads are arranged side by side in the bridge of the machine (X axis of the machine). Each individual laser head can be switched on and off individually and their relative positions among them (Y axis of the machine) can be changed by means of specific machining instructions.

The virtual simulator itself uses a contour-based representation to model the geometry of the processed sheet [138, 147], presenting a virtual 3D interactive scenario with the multi-laser CNC machine that receives the machining instructions. In this virtual scene, the moving and cutting instructions move the corresponding axis of the machines (bridge, laser heads offset along the bridge and laser head height over the sheet metal). The heat sources are then calculated and sent to the heat simulation procedure, updating the temperature of the sheet, which is rendered as a texture over the virtual sheet metal.

Since all the laser heads receive the same machining instructions (although their position along the bridge differ), all movements start and end at the same time. As discussed in Section IV-D.3.3.3, the laser beam trajectories must share the same time discretization, which in this case, is guaranteed by design. If a unrestricted multi-laser machine is used with independent laser heads, i.e, receiving different machining instructions, a global time clock must be used in order to trigger the update of the temperature computation with the correct positions of the moving laser heads.

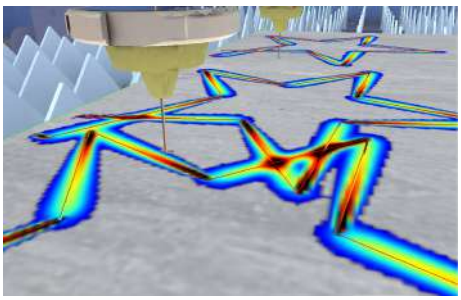
The introduction of the multi-laser machines in the industrial world has an impact from the design point of view. Even with just one laser, the designer of the NC (Numerical Control) programs must take into account the expected produced heat and how it spreads over the sheet metal in order to optimize the nesting of the produced parts. With a multi-laser machine, this procedure is even more critical, as the resulting heat of the multiple sources can accumulate in some areas. A NC designer is expected to use the presented virtual simulator to visualize how the cutting process produces the parts and to analyze the computed temperature through the sheet. If any situation becomes problematic, the designer would make changes to the NC program in order to address the situation.

The presented multi-laser simulator runs at interactive rates. Therefore, the designer can improve the optimization workflow, as many simulations can be run in a short period of time. In contrast, high quality simulations with FEA software, although being highly precise, are computationally expensive. Thus, the number of test configurations that the designer can test during the design phase is limited. Nevertheless, at the end of the optimization phase, the interactive heat simulations can be complemented with high quality FEA simulations.

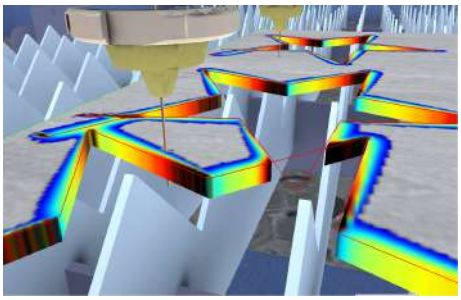
From the industrial point of view, this improved design workflow, assisted with the interactive multi-laser simulation, (1) provides better NC programs in terms of the quality of the produced parts, (2) produces economical benefits due to shorter machining times or less wasted resources, and (3) improves the safety of the operators in the factory floor, reducing unnecessary risks due to unexpected behaviour of the NC programs.

Figure IV-D.3.11(a) shows a machine with three laser heads working simultaneously, i.e, they receive the same machining instructions with fixed spatial offsets). Each laser head machines a star figure. The stars machined by the first and second laser heads overlap, while the third laser head produce an isolated star figure. Figure IV-D.3.11(b) shows a closer view of the cutting area while IV-D.3.11(c) shows the same viewpoint but the cut stars removed from the visualization. The simulation results show that intersecting trajectories present temperature peaks (black zones)

(a) Virtual multi-laser machine



(b) Temperature shown on the sheet surface



(c) Geometric cut of the sheet

Figure IV-D.3.11: Multi-laser machining in the interactive simulator. A star shape is machined by the three laser heads.

where the trajectories overlap (see Fig. IV-D.3.11(b)).

IV-D.3.5 Conclusions

This manuscript presents a novel methodology for the simulation of heat transfer on rectangular sheet metal under multi-laser beam heating. The algorithm is based on an analytic solution to the heat transfer equation which considers some simplifications (2D domain, constant material parameters) in favor of simulation speed. Such simplifications allow the algorithm to solve the transient temperature map on large space / time domains and complex laser trajectories. Furthermore, the algorithm allows many simultaneous laser beams with independent parameters (laser power, speed and spot radius) and time frames (i.e., each laser beam can be turned on / off during the simulation). The numerical accuracy of the algorithm is independent from the space / time discretization, which helps to improve its numerical efficiency. Our simulation tests show that the algorithm is able to render correctly the temperature maps for several laser beams with different trajectories using mesh grids. A numerical comparison with FEA shows that our algorithm solution deviates from the FEA one a maximum of 3.9%, and only around the laser spot. An assessment of the algorithm performance shows that in an implementation using GPUs, the number of laser beams barely affects the simulation time. Finally, the algorithm is implemented into an interactive laser cutting simulation environment for the assessment in real time of laser cutting processes in real

manufacturing scenarios.

The presented algorithm simplifies by design the mathematical model of the problem in favor of interactive simulations. As future work we work on: (1) to simulate the non-linear behaviour of the material properties which arises due to the high temperature gradients, (2) to simulate physically the laser ablation and material removal in sheet cutting processes, (3) to couple the model with an analytic stress model in order to evaluate the potential structural damage due to thermal stress.

IV-D.4

Fast Simulation of Laser Heating Processes on Thin Metal Plates with FFT using CPU/GPU Hardware

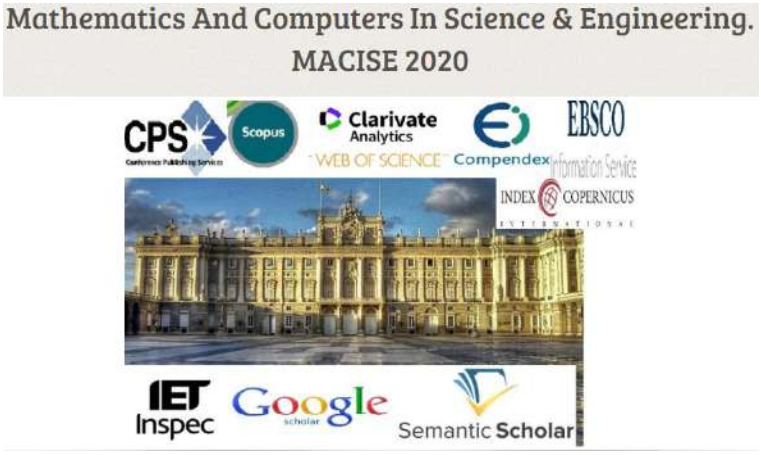
Daniel Mejia-Parra [1,2], Ander Arbelaiz [2], Oscar Ruiz-Salguero [1], Juan Lalinde-Pulido [3], Aitor Moreno [2] and Jorge Posada [2]

[1] Laboratory of CAD CAM CAE, Universidad EAFIT, Colombia

[2] Vicomtech Foundation, Basque Research and Technology Alliance (BRTA)

[3] High Performance Computing Facility APOLO, Universidad EAFIT

Context



International Conference Manuscript

Daniel Mejia-Parra, Ander Arbelaiz, Oscar Ruiz-Salguero, Aitor Moreno and Jorge Posada (**2020**). DST and FFT - based algorithms for laser heating simulation on thin metal plates in GPU. In

Mathematics And Computers in Science & Engineering (MACISE 2020). January 18-20, Madrid, Spain. ADDITIONAL INDEXING PENDING.

applied sciences

an Open Access Journal by MDPI

International Journal Extended Manuscript

Daniel Mejia-Parra, Ander Arbelaiz, Oscar Ruiz-Salguero, Juan Lalinde-Pulido, Aitor Moreno and Jorge Posada. Fast simulation of laser heating processes on thin metal plates with FFT using CPU/GPU hardware. *Applied Sciences*, **10**, 9, pp. 3281:1-3281:25. DOI: 10.3390/app10093281.

Indexing: ISI (Q2), SCOPUS (Q1), Publindex (A1)

Abstract

In flexible manufacturing systems, fast feedback from simulation solutions is required for effective tool path planning and parameter optimization. In the particular sub-domain of laser heating/cutting of thin rectangular plates, current state-of-the-art methods include frequency-domain (spectral) analytic solutions that greatly reduce the required computational time in comparison to industry standard finite element based approaches. However, these spectral solutions have not been presented previously in terms of Fourier methods and FFT implementations. This manuscript presents four different schemes that translate the problem of laser heating of rectangular plates into equivalent FFT problems. The presented schemes reduce the computational time complexity of the problem from $\mathcal{O}(M^2N^2)$ to $\mathcal{O}(MN\log(MN))$ (with $M \times N$ being the discretization size of the plate). The test results show that the implemented FFT schemes outperform previous approaches both in CPU and GPU hardware, resulting in 100× faster runs. Future work addresses thermal/stress analysis, non-rectangular geometries and non-linear interactions (such as material melting/ablation).

Keywords: Spectral Method; Fast Fourier Transform; Laser Heating; GPU; Rectangular Metal Plate; Industry 4.0.

Glossary

PDE	Partial Differential Equation

DST	Discrete Sine Transform
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
$a, b, \Delta z$	Width, height and thickness of the thin plate (m^3).
T_f	Total simulation time (s).
$\vec{x}, t$	Spatial $\vec{x} = (x, y) \in [0, a] \times [0, b]$ and temporal $0 \leq t \leq T_f$ coordinates.
$u = u(\vec{x}, t)$	Temperature field $u : [0, a] \times [0, b] \times [0, T_f] \to \mathbb{R}$ on the metal plate (K).
ρ	Plate density (kg/m^3).
c_p	Plate specific heat ($J/(kg\,K)$).
κ	Plate thermal conductivity ($W/(m\,K)$).
R	Plate reflectivity ($0 \leq R < 1$).
$q = q(u)$	Temperature-dependent heat convection field $q : \mathbb{R} \to \mathbb{R}$ (W/m^2).
h	Natural convection coefficient at the plate surface ($W/(m^2\,K)$)
u_∞	Ambient temperature (K).
$\vec{x}_0 = \vec{x}_0(t)$	Laser spot location at a given time $\vec{x}_0(t) = (x_0(t), y_0(t))$.
$f = f(\vec{x}, t)$	Power Density Field $f : [0, a] \times [0, b] \times [0, T_f] \to \mathbb{R}$ for the laser beam (W/m^2).
P	Laser power (W).
r	Laser spot radius (m).
$M \times N$	2D plate discretization size ($M, N \in \mathbb{N}$).
$\theta_{mn}(t)$	m^{th}, n^{th} Fourier coefficient ($m, n = 0, 1, \dots$) for the temperature solution u at time t.
α_m, β_n	Coefficients $\alpha_m = (m+1)\pi/a$ and $\beta_n = (n+1)\pi/b$ for the Fourier basis in the X- and Y- axis, respectively.
γ_m, δ_n	$\gamma_m = m\pi/M$ and $\delta_n = n\pi/N$ are the discrete equivalent of α_m ($m = 0, 1, \dots, M-1$) and β_n ($n = 0, 1, \dots, N-1$), respectively.
ω_{mn}	m^{th}, n^{th} eigenvalue of the heat (Laplace) operator defined on the rectangular plate.
$\vec{C}_i(t)$	Piecewise linear discretization of the laser trajectory $\vec{x}_0(t)$.

IV-D.4.1 Introduction

Based on virtual modelling and simulation of physical phenomena, Industry 4.0 solutions aim to integrate interactive virtual worlds to their equivalent physical part (e.g. using digital twins). These solutions enable the development of decision making tools that can be of great use in the optimization of manufacturing processes.

In this context, engineering solutions use extensively Finite Element Analysis (FEA) for simulation of such physical phenomena (e.g. acoustics, heat transfer, structural analysis, fluid flow, etc). However, FEA approaches require a great amount of computation resources. In contrast, spectral analysis and spectral methods are competitive alternatives to numerical simulations. These methods provide frequency-domain solutions (infinite sum of trigonometric functions) to the Partial Difference Equations (PDEs) that model such physical phenomena.

In the particular sub-domain of laser heating/cutting simulation, frequency-based algorithms have been developed for heat transfer analysis on rectangular plates. These algorithms are faster than traditional numerical methods (such as Finite Element Methods) at the cost of some model simplifications. In addition, these methods provide some advantages over FEA, such as allowing to zoom into asynchronous time intervals without computing or storing the complete history of the

solution.

This property makes frequency-based algorithms more adequate for decision making tools that require rapid response times, allowing to be more flexible towards changes in the heating/cutting manufacturing process. Fast simulation of laser heating/cutting problems is very important for different engineering problems, such as: tool path planning, laser parameter optimization, waste and resources optimization, etc. Moreover, interactive simulation and visualization of laser machining processes contributes to many different challenges and opportunities currently present in the Industrie 4.0 framework (already identified in Ref. [55]).

The aforementioned methods for laser heating/cutting simulation allow simulation of complex laser trajectories on rectangular plates, including parametric trajectories and the introduction of multiple laser beams simultaneously. However, there are no Fast Fourier Transform (FFT)-based solutions to the laser heating/cutting problem in the current state of the art.

The FFT is a widely used algorithm not only in the context of PDEs simulation, but also in other areas such as signal analysis and image processing. Thus, its development has been refined and studied extensively in the literature. Several FFT algorithms exist in the literature that further optimize the computation in function of the input signal properties (e.g. symmetry, real/imaginary, size, etc.). In general, the FFT is a key algorithm that retrieves the original spatial-based solution by performing a factorization of the Discrete Fourier Transform (DFT) and avoids redundant computations, reducing the computational complexity of the original DFT problem.

This article presents four different schemes that cast the laser heating/cutting problem into DST (Discrete Sine Transform) and DFT (Discrete Fourier Transform) ones. Such casting enables the use of FFT libraries to implement these schemes. The test results show a significant improvement of the computational time, both in CPU and GPU over existing methods, due to the computational complexity reduction.

This manuscript is an extension of the work presented in Ref [164], in which only two schemes were briefly introduced for the FFT computation of the laser heating problem. The current research discusses in more detail each of the four schemes, including mathematical and algorithmic descriptions but also the intuition behind the schemes followed by illustrations. Furthermore, a different simulation case is designed and tested. Finally, the presented schemes are in the process of being applied in an Industry 4.0 application prototype. The ongoing prototype implements an interactive virtual model of a laser heating/cutting machine using geometry operations and physical simulation.

The remainder of this manuscript is organized as follows: Sect. IV-D.4.2 discusses the relevant literature. Sect. IV-D.4.3 presents the proposed FFT schemes. Sect. IV-D.4.4 discusses the test results. Finally, Sect. IV-D.4.5 presents the conclusions and discusses what remains for future work.

IV-D.4.2 Literature review

IV-D.4.2.1 Laser Heating/Cutting Simulation

Finite Element Analysis (FEA) is one of the most used methods for thermodynamic simulation of laser heating/cutting of metal plates. Using non-linear FEA, Ref. [165] simulates triangular cuts for residual stress analysis. Similarly, Refs. [166, 167] perform the same non-linear FEA analysis for rectangular cuts, and Ref. [168] studies circular cuts using the same approach. In order to account for laser ablation (material melting and evaporation), different methods such as the

enthalpy method [165–168], element birth and death [113], volume fractions [140], and temperature thresholds [114, 116, 158] have been presented.

Other numerical methods include Finite Differences [117,119,120], Boundary Elements [121,122] and Finite Volumes [123, 124]. However, numerical methods are computationally expensive in general, limiting their application to small plate geometries and simple laser trajectories, requiring full time history simulations.

Analytic methods provide significantly faster computations at the cost of some model simplifications. Ref. [129] presents a uni-dimensional analytic model for laser drilling processes when the laser beam is static. Ref [128] presents a solution for a moving laser on an infinite 2D plate. Ref. [132] presents a frequency-based solution for rectangular plates when the moving laser follows a straight path. Similarly, Refs. [54, 169] present a frequency-based solution for arbitrary laser trajectories. Finally, Ref. [170] extends the previous frequency-based solutions to multiple laser beams simultaneously heating the plate surface.

IV-D.4.2.2 FFT-based Laser Heating Simulation

FFT-based methods are relevant in the solution of physical problems by solving the inherent PDE in the frequency domain. As a consequence, these methods have been successfully implemented in the simulation of different physics phenomena. For example, in the context of heat transfer analysis, Ref. [171] presents an FFT-based method for the solution of the thermoelastic equation on infinite domains, while Ref. [172] applies the FFT to the solution of a heat transfer problem that arises in treatments of tissue with cancer. In structural analysis, Refs. [173–176] develop FFT-based methods for the solution of different elasticity and plasticity problems, and Ref. [177] presents a FFT-based solver for fluid mechanics. Other applications of the FFT include electromagnetism [178], 1D signal processing [179], and 2D image processing [180].

As discussed previously, Refs. [54, 132, 169, 170] solve the problem of laser heating simulation in the frequency domain. However, none of these references are able to cast their problems into the FFT domain.

IV-D.4.2.3 Conclusions of the Literature Review

Current analytic methods for simulation of the laser heating/cutting problem already provide fast solutions to the problem in the frequency domain. However, such methods perform brute-force evaluation of the Fourier transforms, whose computation complexity for a 2D plate is $\mathcal{O}(M^2N^2)$. As a consequence, these applications quickly become computationally expensive as more resolution of the plate is required.

To overcome this problem, this manuscript presents four different schemes that cast the existing brute-force solutions into equivalent DST and DFT problems. Mathematical proof for the validity of each scheme is presented and algorithms that make use of FFT libraries are introduced, reducing the computational complexity of the problem from $\mathcal{O}(M^2N^2)$ (squared) to $\mathcal{O}(MN\log(MN))$ (logarithmic). These algorithms are implemented both in CPU and GPU architectures. Numerical validation against the brute-force approach results in a measured absolute error that is below $10^{-10}K$ along the 2D plate. The results show significant computation improvements to such brute-force simulations (i.e. Refs. [54, 132, 169, 170]), reducing the measured computation times from $1s$ to $0.01s$ ($100\times$ faster) for a 1024×1024 rectangular plate, and enabling simulations for larger plate discretization sizes (up to 4096×4096).

This manuscript extends the work presented in Ref. [164]. In this previous work, two of the four presented schemes are briefly introduced. The research presented in this paper presents two new FFT schemes, and provides further details of the four schemes (with added illustrations) for better understanding of the algorithms. Furthermore, new simulations have been executed and an application case of the algorithms being implemented into an interactive simulator is presented.

IV-D.4.3 Methodology

IV-D.4.3.1 Heat Transfer Equation for Laser Heating on Thin Plates

The temperature $u(x, y, t)$ on a 2D rectangular plate for a continuous laser beam source satisfies the following partial differential equation with initial and boundary conditions:

$$\begin{aligned} \frac{f-q}{\Delta z} &= \rho c_p \frac{\partial u}{\partial t} - \nabla \cdot (\kappa \nabla u) \\ q(x,y,t) &= h \cdot (u(x,y,t) - u_\infty) \\ u(x,y,0) &= u_\infty \\ u(0,y,t) &= u(a,y,t) = u(x,0,t) = u(x,b,t) = u_\infty \end{aligned} \tag{IV-D.4.1}$$

where $a \times b \times \Delta z$ are the plate dimensions, ρ is the plate density, c_p is the specific heat and κ is the thermal conductivity. $q = q(x, y, t)$ is the heat loss due to convection at the plate surface, h is the convection coefficient and u_∞ is the ambient temperature. Finally, the heat source $f = f(x, y, t)$ is defined as a square-shape moving laser beam:

$$f(x,y,t) = \begin{cases} \frac{P(1-R)}{\pi r^2}, & \|\vec{x} - \vec{x}_0(t)\|_\infty < \frac{r\sqrt{\pi}}{2} \\ 0, & \text{otherwise} \end{cases} \tag{IV-D.4.2}$$

where R is the plate reflectivity, P is the laser power, r is the laser radius and $\vec{x}_0(t) = [x_0(t), y_0(t)]$ is the location of the laser spot at time t. $\vec{x}_0$ is the parametric curve that defines the laser trajectory, discretized as a sequence of piecewise linear trajectories as described in Refs. [54,169]. The function f describes the laser power on the plate according to the distance (infinity norm) $\|\vec{x} - \vec{x}_0(t)\|_\infty = \max(x - x_0(t), y - y_0(t))$ of each plate point $\vec{x} = [x, y]$ to the laser spot $\vec{x}_0(t) = [x_0(t), y_0(t)]$. Fig. IV-D.4.1 presents an scheme of the laser heating problem on thin metal plates.

IV-D.4.3.2 Analytic Solution

According to Refs. [54,169], the solution to Eq. (IV-D.4.1) can be expressed as Fourier series:

$$u(x,y,t) = u_\infty + \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \theta_{mn}(t) \sin(\alpha_m x) \sin(\beta_n y) \tag{IV-D.4.3}$$

with $\alpha_m = (m+1)\pi/a$ and $\beta_n = (n+1)\pi/b$. Each Fourier coefficient $\theta_{mn}(t)$ is defined as:

$$\theta_{mn}(t) = \frac{4}{ab\rho c_p \Delta z} \int_0^t \int_0^b \int_0^a f(x,y,\tau) \sin(\alpha_m x) \sin(\beta_n y) e^{-\omega_{mn}(t-\tau)} dx dy d\tau \tag{IV-D.4.4}$$

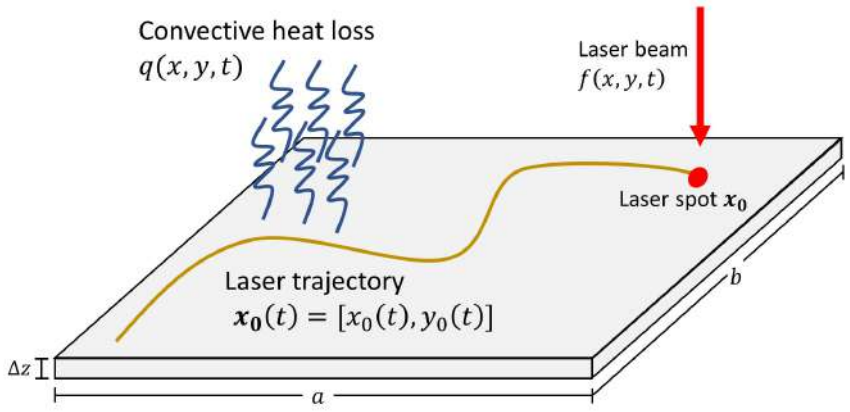


Figure IV-D.4.1: Scheme for the laser heating problem on thin metal plates.

with Laplace eigenvalues ω_{mn}:

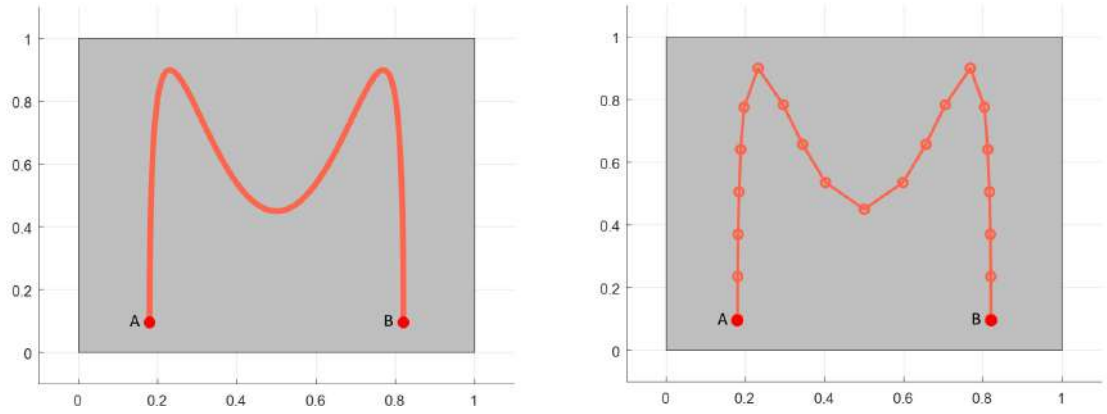
$$\omega_{mn} = \frac{\kappa}{\rho c_p}(\alpha_m^2 + \beta_n^2) + \frac{h}{\rho c_p \Delta z} \tag{IV-D.4.5}$$

Let $\vec{C}_1(t), \vec{C}_2(t), \ldots$ be a sequence of piecewise linear sub-trajectories that discretize the complete laser trajectory (see Fig. IV-D.4.2), i.e. $\vec{x}_0(t) \approx \vec{C}_1(t), \vec{C}_2(t), \ldots$. Each sub-trajectory $\vec{C}_i$ $(i > 0)$ is defined as a linear trajectory:

$$\vec{C}_i(t) = \vec{x}_0(t_i)\frac{t - t_{i-1}}{t_i - t_{i-1}} + \vec{x}_0(t_{i-1})\frac{t_i - t}{t_i - t_{i-1}}, \qquad t_{i-1} \leq t < t_i \tag{IV-D.4.6}$$

where the original laser trajectory $\vec{x}_0$ is sampled at $t = t_0, t_1, t_2, \ldots, T_f$.

The analytic solution for Eq. (IV-D.4.4) for the given piecewise linear discretization is presented in Refs. [54, 169].



(a) Continuous laser trajectory

(b) (Coarse) Piecewise linear discretization of the trajectory

Figure IV-D.4.2: Continuous laser trajectory (from point A to B) and piecewise linear discretization of the trajectory on a rectangular plate

IV-D.4.3.3 Discrete Fourier Transform (DFT) and Fast Fourier Transform (FFT)

The Discrete Fourier Transform (DFT) [181] allows to write any sequence of M real numbers as a finite sum of sine and cosine functions, i.e. a Fourier series. The (1D) DFT of the sequence of real values $G = \{g_0, g_1, \ldots, g_{M-1}\} \subset \mathbb{R}$ is defined as:

$$g_k = \sum_{m=0}^{M-1} \phi_m e^{-\frac{i2\pi}{M}km} = \sum_{m=0}^{M-1} \phi_m \left[\cos\frac{2\pi km}{M} - i\sin\frac{2\pi km}{M}\right] \tag{IV-D.4.7}$$

where $\phi_m \in \mathbb{C}$ is the m^{th} Fourier coefficient and $i = \sqrt{-1}$ is the imaginary unit. The computational complexity for direct evaluation of Eq. (IV-D.4.7) is $\mathcal{O}(M^2)$, in which each g_k requires M evaluations (one for each Fourier term ϕ_m).

The Fast Fourier Transform (FFT) [179] is an algorithm that performs a factorization of the DFT, reordering the Fourier terms and grouping them (into pairs) in order to avoid redundant computations between different g_k terms. Such a grouping is possible due to symmetries of the sine and cosine functions, and the resulting evaluation is performed in binary-tree recursive form [179]. As a consequence, the FFT algorithm reduces the computational complexity of the problem to $\mathcal{O}(M \log M)$[179].

The above DFT and FFT complexity orders are true for 1D arrays. Therefore, for a 2D discrete plate of size $M \times N$, the computational complexities become $\mathcal{O}(M^2 \times N^2)$ and $\mathcal{O}(MN \log(MN))$ for the DFT and the FFT, respectively.

The remainder of this section describes how to cast Eq. (IV-D.4.3) as a DFT problem and therefore, solve it using any FFT algorithm. Such casting effectively improves the computational complexity of the problem with respect to the current state of the art [54, 132, 169, 170].

IV-D.4.3.4 Scheme 1 - Discrete Sine Transform (DST)

The Discrete Sine Transform (DST) [182] is a particular case of the DFT transform in which only the sine terms of the Fourier series are considered. The (1D) DFT of the sequence $G = \{g_0, g_1, \ldots, g_{M-1}\} \subset \mathbb{R}$ is defined as:

$$g_k = \sum_{m=0}^{M-1} \phi_k \sin \frac{(m+1)(k+1)\pi}{M+1} \tag{IV-D.4.8}$$

Intuitively, this is the easiest of the schemes for casting the problem as Eq. (IV-D.4.3) only considers the sine terms of a Fourier series. The algorithm of such casting is discussed below. The reader may refer to Appendix V-.0.B.1 for the mathematical proof of the scheme.

IV-D.4.3.4.1 Algorithm

Algorithm 1 presents the method used to retrieve the temperature at any given time t with the DST method (see Eq. (8)). Line 2 applies the 2D DST of any FFT library, while Line 3 applies the initial and boundary conditions presented in Eq. (IV-D.4.1) to the computed solution. The complexity of the presented algorithm is $\mathcal{O}(MN \log(MN))$.

Algorithm 1: Retrieve temperature using a 2D DST

Require: $\Theta \in \mathbb{R}^{(M-2)\times(N-2)}, \quad u_\infty \in \mathbb{R}$

Ensure: $U \in \mathbb{R}^{M\times N}$

1: $U \leftarrow \text{zeros}(M, N)$

2: $U[1 : M-1, 1 : N-1] \leftarrow \text{dst2d}(\Theta)$

3: $U \leftarrow U + u_\infty$

4: **return** U

IV-D.4.3.5 Scheme 2 - FFT Padded with Zeros

In this scheme, the original list of Fourier coefficients is duplicated in size in each direction ($2M \times 2N$). The added coefficients are set to zero and the FFT algorithm is applied in each direction. The final temperature result is obtained from the imaginary (sine) components of the FFT result. The mathematical proof of the scheme is presented in Appendix V-.0.B.2.

IV-D.4.3.5.1 Algorithm

Algorithm 2 presents the method used to retrieve the temperature at any given time t using the zero padding method (see Eq. (10)). Line 1 initializes the extended matrix of Fourier coefficients with M, N trailing zeros (as per Fig. IV-D.4.3). Lines 4 and 8 compute the 1D FFT of the padded arrays for the y and x dimensions, respectively. Lines 5 and 9 extract the complex (imaginary component)

of the results. Finally, Line 12 removes the trailing zeros from the solution while Line 13 applies initial and boundary conditions. The complexity of the presented algorithm is $\mathcal{O}(MN\log(MN))$.

Algorithm 2: Retrieve temperature using a FFT with zero padding

Require: $\Theta \in \mathbb{R}^{(M-2)\times(N-2)}, \quad u_\infty \in \mathbb{R}$

Ensure: $U \in \mathbb{R}^{M\times N}$

1: $\Theta_{PADDED} \leftarrow \text{zeros}(2M, 2N)$

2: $\Theta_{PADDED}[1 : M-1, 1 : M-1] \leftarrow \Theta$

3: **for** $n = 1, n < N-1, n \leftarrow n+1$ **do**

4: $\quad arr \leftarrow \text{fft}(\Theta_{PADDED}[:, n])$

5: $\quad \Theta_{PADDED}[:, n] \leftarrow \text{imag}(arr)$

6: **end for**

7: **for** $m = 1, m < M-1, m \leftarrow m+1$ **do**

8: $\quad arr \leftarrow \text{fft}(\Theta_{PADDED}[m, :])$

9: $\quad \Theta_{PADDED}[m, :] \leftarrow \text{imag}(arr)$

10: **end for**

11: $U \leftarrow \text{zeros}(M, N)$

12: $U \leftarrow \Theta_{PADDED}[0 : M-1, 0 : N-1]$

13: $U \leftarrow U + u_\infty$

14: **return** U

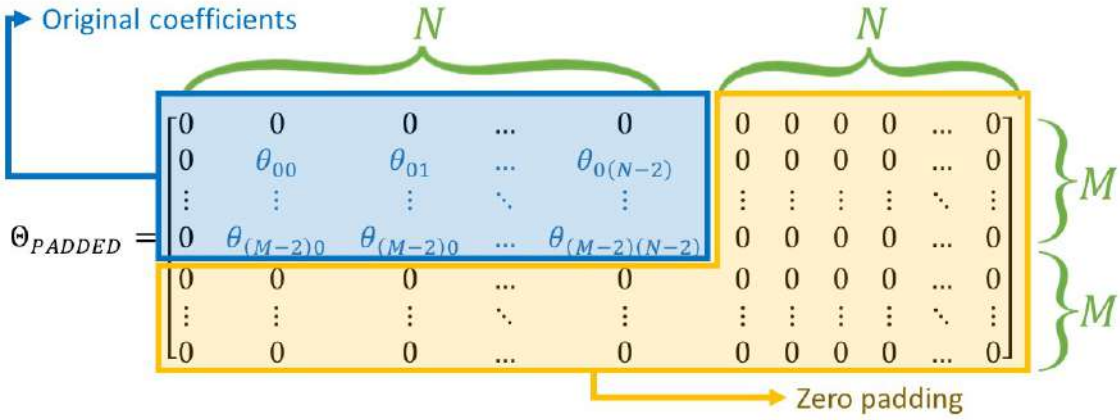


Figure IV-D.4.3: Matrix structure for the zero padding FFT. The blue block contains the original Fourier coefficients θ_{mn}. The remainder of the matrix is filled with zeros.

IV-D.4.3.6 Scheme 3 - Odd-Symmetry 1D FFT

In this scheme, the original list of Fourier coefficients is also duplicated in size in each direction ($2M \times 2N$). The idea is to take advantage of the odd symmetry of the sine function at $k\pi$ (with $k \in \mathbb{N}$, see Fig. IV-D.4.4). Therefore, the added coefficients are set by mirroring the original M or N coefficients (multiplied by -1) in each direction (rows and columns). The final temperature is obtained from the imaginary (sine) components of the 1D FFT result in each direction. The mathematical proof of this scheme is presented in Appendix V-.0.B.3.

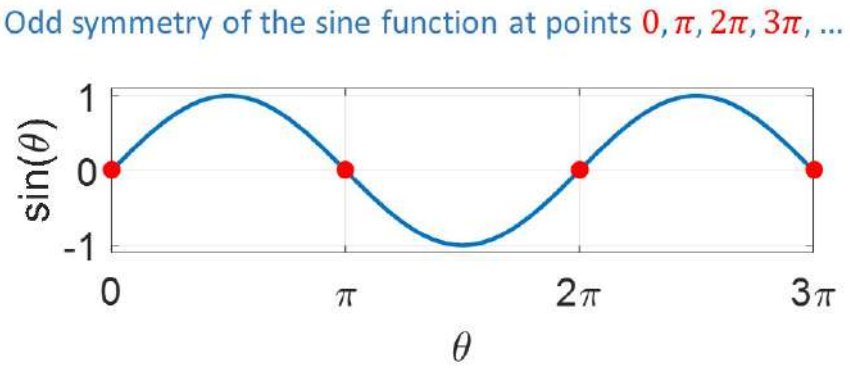


Figure IV-D.4.4: Odd symmetry of the sine function at $k\pi$ ($k = 0, 1, 2, \dots$)

IV-D.4.3.6.1 Algorithm

Algorithm 3 presents the method used to retrieve the temperature of Eq. (16) at any given time t using two nested 1D FFTs. Line 1 initializes the extended matrix of Fourier coefficients with M, N trailing zeros. Lines 3-5 and Lines 6-8 add the reversed sequences of Fourier coefficients (with

negative sign) in each dimension, respectively (see Fig. IV-D.4.5). Lines 10 and 14 compute the 1D FFT of the padded arrays for the y and x dimensions, respectively. Lines 11 and 15 extract the complex (imaginary component) of the results. Finally, Line 12 removes the mirrored part from solution while Line 13 applies initial and boundary conditions. The complexity of the presented algorithm is $\mathcal{O}(MN\log(MN))$.

Algorithm 3: Retrieve temperature using 1D FFTs by applying odd symmetry to the original coefficients

Require: $\Theta \in \mathbb{R}^{(M-2)\times(N-2)}, \quad u_\infty \in \mathbb{R}$

Ensure: $U \in \mathbb{R}^{M\times N}$

1: $\Theta_{ODD\_SYM} \leftarrow \text{zeros}(2M, 2N)$

2: $\Theta_{ODD\_SYM}[1 : M-1, 1 : M-1] \leftarrow \Theta$

3: **for** $m = M+1, m < 2M-1, m \leftarrow m+1$ **do**

4: $\quad \Theta_{ODD\_SYM}[m, :] \leftarrow -\Theta_{ODD\_SYM}[2M-m-1]$

5: **end for**

6: **for** $n = N+1, n < 2N-1, n \leftarrow n+1$ **do**

7: $\quad \Theta_{ODD\_SYM}[n, :] \leftarrow -\Theta_{ODD\_SYM}[2N-n-1]$

8: **end for**

9: **for** $n = 1, n < N-1, n \leftarrow n+1$ **do**

10: $\quad arr \leftarrow \text{fft}(\Theta_{ODD\_SYM}[:, n])$

11: $\quad \Theta_{ODD\_SYM}[:, n] \leftarrow \text{imag}(arr)$

12: **end for**

13: **for** $m = 1, m < M-1, m \leftarrow m+1$ **do**

14: $\quad arr \leftarrow \text{fft}(\Theta_{ODD\_SYM}[m, :])$

15: $\quad \Theta_{ODD\_SYM}[m, :] \leftarrow \text{imag}(arr)$

16: **end for**

17: $U \leftarrow \text{zeros}(M, N)$

18: $U \leftarrow \Theta_{ODD\_SYM}[0 : M-1, 0 : N-1]$

19: $U \leftarrow U + u_\infty$

20: **return** U

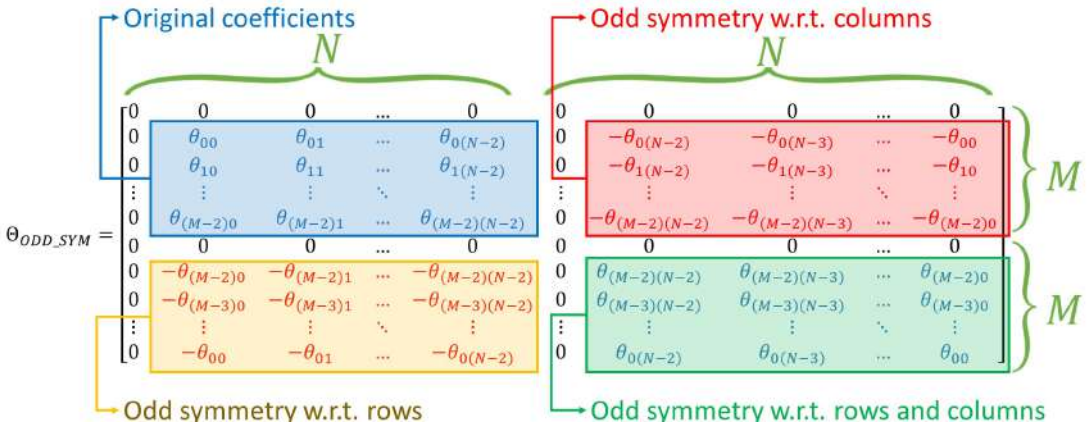


Figure IV-D.4.5: Matrix structure for the odd symmetric FFT (1D and 2D). The blue block contains the original coefficients and the remaining blocks contain their odd symmetric counterpart. All blocks are separated by rows and columns of zeros.

IV-D.4.3.7 Scheme 4 - Odd-Symmetry 2D FFT

Finally, in this scheme the original list of coefficients is duplicated and mirrored in each direction exactly as in Sect. IV-D.4.3.6. However, this scheme also takes advantage of the even symmetry of the cosine function at $2k\pi$ ($k \in \mathbb{N}$, see Fig. IV-D.4.6). Similar to the 1D odd symmetric approach, the duplicated coefficients are mirrored in each direction (rows and columns), and multiplied by -1. The final temperature is retrieved from the the real component of the 2D FFT, which considers the sine components and the cosine components (that become 0 due to the cosine symmetry). The mathematical proof of the scheme is presented in Appendix V-.0.B.4.

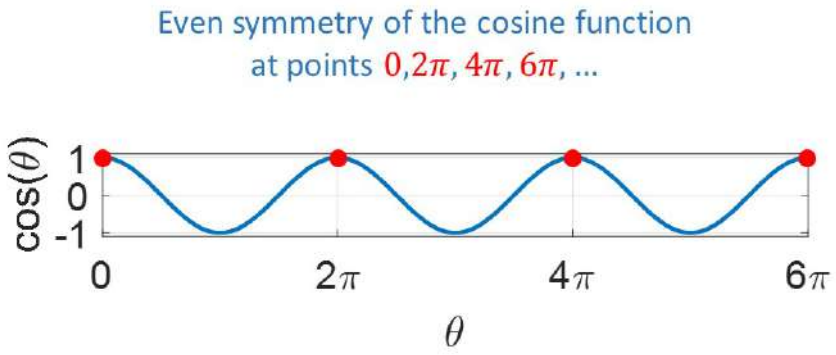


Figure IV-D.4.6: Even symmetry of the cosine function at $2k\pi$ ($k = 0, 1, 2, \dots$)

IV-D.4.3.7.1 Algorithm

Algorithm 4 presents the method used to retrieve the temperature of Eq. (23) at any given time t using a 2D FFT. Line 1 initializes the extended matrix of Fourier coefficients with M, N trailing

zeros. Similar to the 1D odd symmetric method, Lines 3-8 add the reversed sequences of Fourier coefficients (with negative sign) in each direction (see Fig. IV-D.4.5). Line 9 computes the 2D FFT of the extended Fourier matrix. Line 11 extracts the real part of the FFT solution and removes the mirrored part. Finally, Line 12 applies the initial and boundary conditions. The complexity of the presented algorithm is $\mathcal{O}(MN\log(MN))$

Algorithm 4: Retrieve temperature using 2D FFTs by applying odd sine symmetry and even cosine symmetry to the original coefficients

Require: $\Theta \in \mathbb{R}^{(M-2)\times(N-2)}, \quad u_\infty \in \mathbb{R}$

Ensure: $U \in \mathbb{R}^{M\times N}$

1: $\Theta_{ODD\_SYM} \leftarrow \text{zeros}(2M, 2N)$

2: $\Theta_{ODD\_SYM}[1 : M-1, 1 : M-1] \leftarrow \Theta$

3: **for** $m = M+1, m < 2M-1, m \leftarrow m+1$ **do**

4: $\quad \Theta_{ODD\_SYM}[m, :] \leftarrow -\Theta_{ODD\_SYM}[2M-m-1]$

5: **end for**

6: **for** $n = N+1, n < 2N-1, n \leftarrow n+1$ **do**

7: $\quad \Theta_{ODD\_SYM}[n, :] \leftarrow -\Theta_{ODD\_SYM}[2N-n-1]$

8: **end for**

9: $Mat \leftarrow \text{fft2d}(\Theta_{ODD\_SYM})$

10: $U \leftarrow \text{zeros}(M, N)$

11: $U \leftarrow \text{real}(Mat[0 : M-1, 0 : N-1])$

12: $U \leftarrow U + u_\infty$

13: **return** U

IV-D.4.4 Results

This section presents the results of the implementation of the presented DST and FFT schemes using different state-of-the-art FFT libraries for the solution of the laser heating problem on thin metal plates. All the simulations are executed with the parameters presented in Table IV-D.4.2 and the laser trajectory presented in Fig. IV-D.4.2(a). Sect. IV-D.4.4.1 presents the numerical validation of the presented schemes with respect to the brute-force algorithms [54, 132, 169, 170]. Finally, Sect. IV-D.4.4.2 discusses the computational performance of the implemented schemes using available FFT libraries.

Table IV-D.4.2: Parameters for the physical simulation

Parameter	Description	Value	Units
a	Plate width	0.01	m
b	Plate height	0.01	m
Δz	Plate thickness	0.001	m
ρ	Plate density	8030	kg/m^3
c_p	Specific heat	574	$J/(kg\,K)$
κ	Thermal conductivity	20	$W/(m\,K)$
R	Plate reflectivity	0	1
h	Convection coefficient	20	$W/(m^2\,K)$
u_∞	Ambient temperature	300	K
P	Laser power	500	W
r	Laser spot radius	0.0003	m

IV-D.4.4.1 Numerical Validation

Sect. IV-D.4.3 validates the mathematical correctness of the presented schemes. However, a numerical validation is presented in this section with numerical and graphical results for a $0.01 \times 0.01 \times 0.001$ rectangular plate. Laser and material parameters are presented in Table IV-D.4.2 while the laser trajectory used for the tests is the same presented in Fig. IV-D.4.2(a). As a groundtruth, we choose the method presented in Refs. [54, 169, 170]. This method already solves the problem presented in Eq. (IV-D.4.1) using a brute-force approach, which requires $\mathcal{O}(M^2N^2)$ operations (as already discussed in Sect. IV-D.4.3.3. Fig. IV-D.4.7 plots the temperature distribution results obtained with this brute-force method.

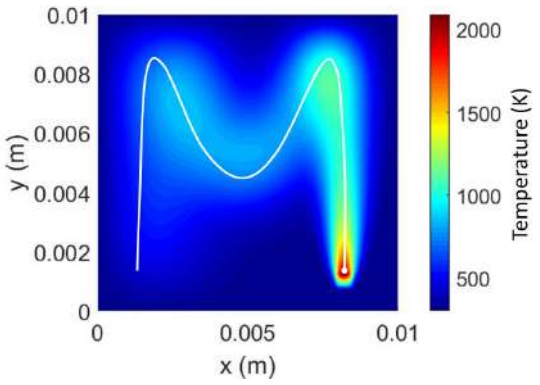
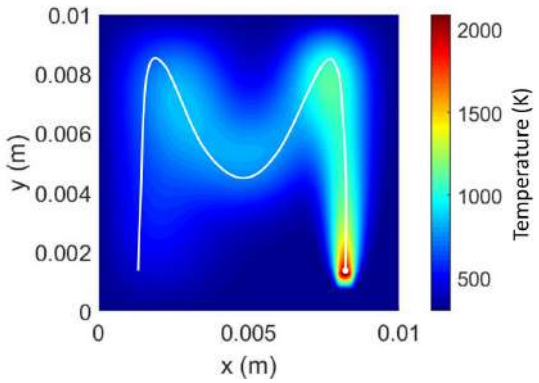


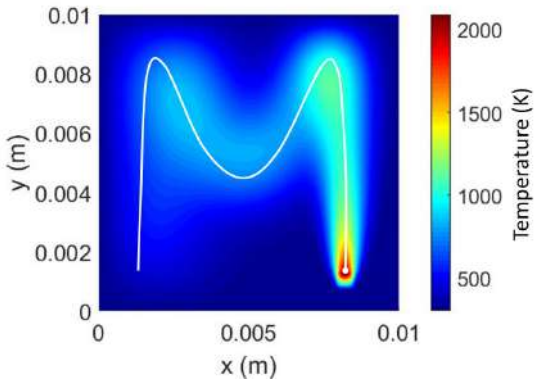
Figure IV-D.4.7: Temperature solution for the laser trajectory presented in Fig. IV-D.4.2(a) obtained by the brute-force method [169]. No FFT or DST is used.

Fig. IV-D.4.8(a) plots the temperature distribution at the end of the laser trajectory, computed with the DST algorithm for a 1024×1024 plate discretization. Fig. IV-D.4.8(b) plots the same result computed with the zero padding FFT algorithm. The absolute error for the DST and the zero paddding FFT result (w.r.t. the brute-force approach) is presented in Figs. IV-D.4.8(c) and

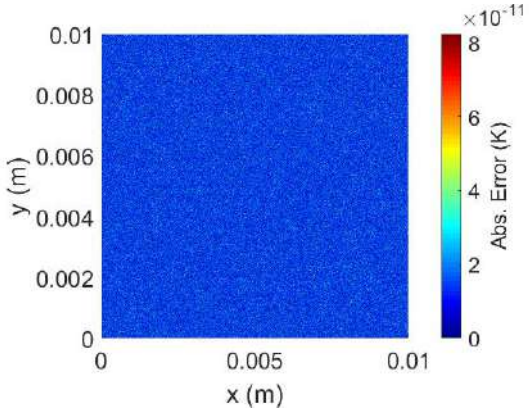
IV-D.4.8(d), respectively. The measured absolute error is below $10^{-10}(K)$ in both cases. It is worth pointing out that this error is evenly distributed through the 2D plate, which means that such error is not sensitive to the laser path or any other geometric features (such as the domain boundaries).



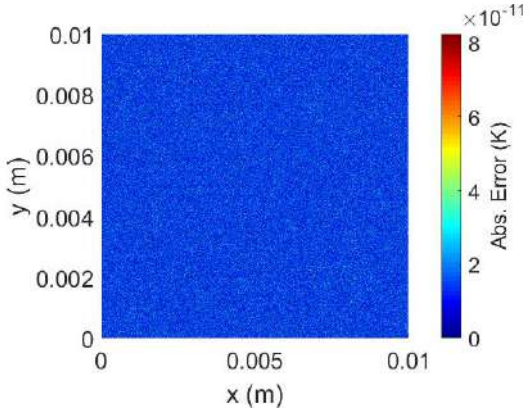
(a) DST temperature distribution



(b) Zero padding temperature distribution



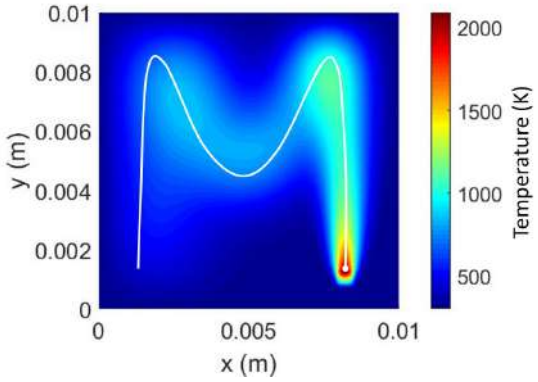
(c) DST error distribution



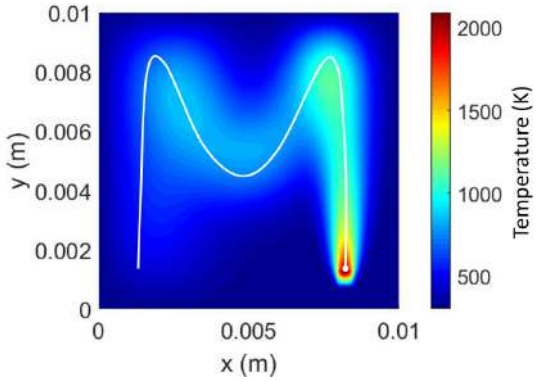
(d) Zero padding error distribution

Figure IV-D.4.8: Temperature and absolute error distributions (w.r.t. the brute-force approach [169]) on the thin plates for the DST and the zero padding FFT simulations

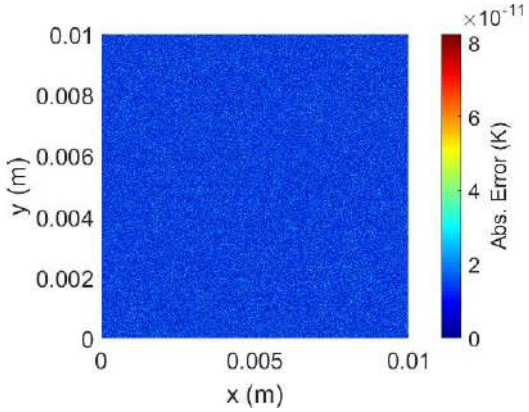
Similarly, Figs. IV-D.4.9(a) and IV-D.4.9(b) plot the temperature distributions at the end of the laser trajectory for the 1D symmetric FFT and the 2D symmetric FFT algorithms, respectively. Figs. IV-D.4.9(c) and IV-D.4.9(d) plot the absolute error for the 1D symmetric and 2D symmetric FFTs, respectively. Again, the error is below 10^{-10}, evenly distributed through the 2D plate.



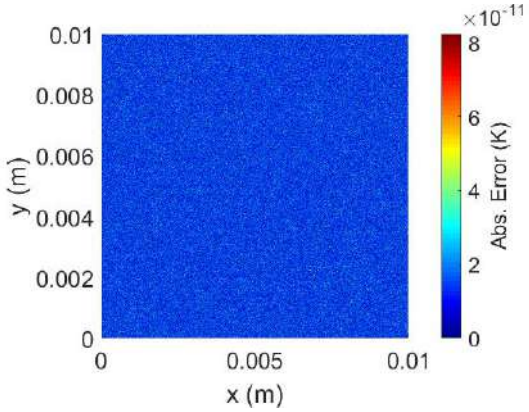
(a) 1D odd symmetry temperature distribution



(b) 2D odd symmetry temperature distribution



(c) 1D odd symmetry error distribution



(d) 2D odd symmetry error distribution

Figure IV-D.4.9: Temperature and absolute error distributions (w.r.t. the brute-force approach [169]) on the thin plates for the odd symmetry FFT approaches (1D and 2D)

IV-D.4.4.2 Computational performance

This section evaluates the performance of the proposed methods under CPU and GPU hardware architectures by making use of highly optimized FFT libraries. The Python programming language includes in its scientific package ecosystem high level wrappers to C/C++ libraries. For this reason, Python has been selected for the rapid prototyping of the proposed schemes in this work.

The FFT algorithm is used in a wide range of performance demanding applications. Therefore, the optimization degree of its implementation is highly relevant. On the one hand, to target the CPU, the FTTPACK, MKL and FFTW libraries have been selected. On the other hand, to target the GPU, the cuFFT library from the NVIDIA CUDA Toolkit has been used. All these libraries make use of multi-core parallelization, vectorization instructions, efficient memory usage, and apply specific FFT algorithms to exploit the underlying hardware to the highest degree. It is worth noticing that the FFTPACK library is the only one (between the aforementioned ones) that

provides an implementation of the DST.

Table IV-D.4.3 summarizes the selected libraries along the Python wrapper packages and the targeted hardware device during the performance tests.

Table IV-D.4.3: Selected libraries and corresponding Python packages

Library	**Python package**	**Hardware**
FFTPACK	scipy.fft	CPU
MKL	numpy.fft	CPU
FFTW	pyfftw	CPU
cuFFT	pyCUDA, scikit-cuda	GPU

Two test platforms have been used for the performance measurements: *i)* a desktop PC using Windows 10 with an Intel Core i5-6500 (CPU), 16 GB RAM and NVIDIA GeForce GTX 960 (GPU) and *ii)* a desktop PC using Manjaro (GNU/Linux) with an Intel Core i7-4700K (CPU), 16GB RAM and NVIDIA GeForce RTX 2060 (GPU). To measure the execution times of each proposed method, each test has been computed 5 times and the minimum time has been registered.

This section is divided into four subsections. Sect. IV-D.4.4.2.1 presents the computation times using the CPU, while Sect. IV-D.4.4.2.2 presents the computational times using GPU hardware. Then, Sect. IV-D.4.4.2.3 compares the performance difference between both devices. Finally, Sect. IV-D.4.4.2.4 presents the achieved speed-up against the state of the art brute-force solution [54, 132, 169, 170].

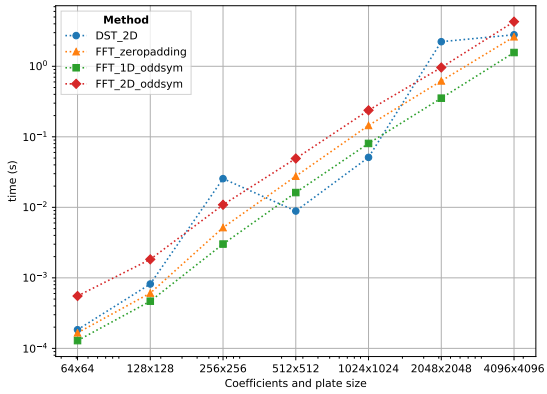
IV-D.4.4.2.1 CPU performance measurements

Fig. IV-D.4.10 shows the computation time of the proposed schemes using the FFTPACK, MKL and FFTW libraries, respectively. These are all implemented to be executed in general CPU hardware.

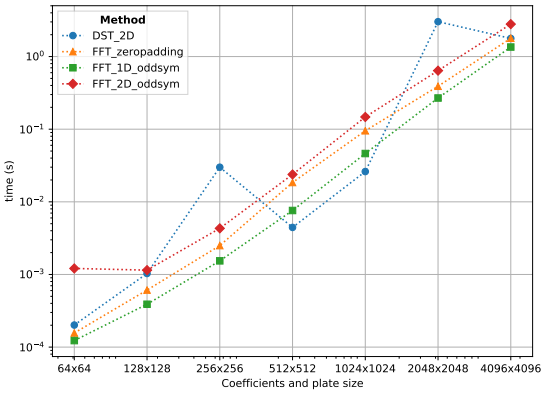
Fig. IV-D.4.10(a) and IV-D.4.10(b) show all the proposed schemes implemented with the FFTPACK library. The FFTPACK is the only library (between the used ones in this manuscripts) that has an implementation of the DST algorithm. This DST implementation is efficient for plate discretization sizes of 512×512 and 1024×1024. However, its performance is not as consistent as the FFT based methods. Overall, the performance of the FFT-based methods with different input size are more stable, being the 1D odd symmetric FFT scheme the best approach using the FFTPACK library.

Fig. IV-D.4.10(c) and IV-D.4.10(d) show the execution times of the temperature evaluation making use of the MKL library. In this case, from the FFT-based methods, both the 1D and 2D odd symmetric schemes are the most efficient.

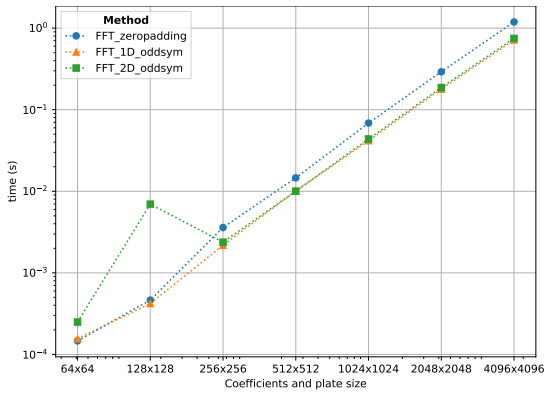
Fig. IV-D.4.10(e) and IV-D.4.10(f) show the computation times using the FFTW library. Although, quite close to the results obtained with the MKL library, the FFTW results are the best when using the CPU device. In this case, also both the 1D and 2D odd symmetric schemes are the most efficient.



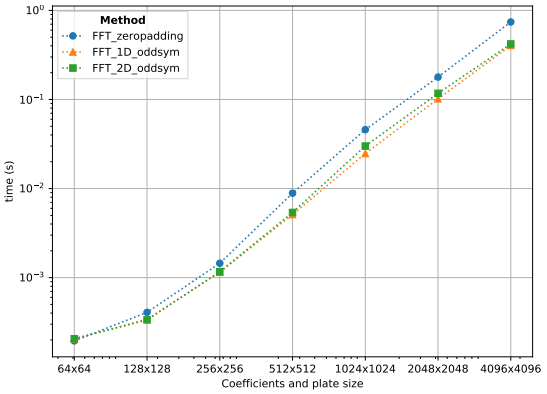
(a) FFTPACK with Intel i5-6500



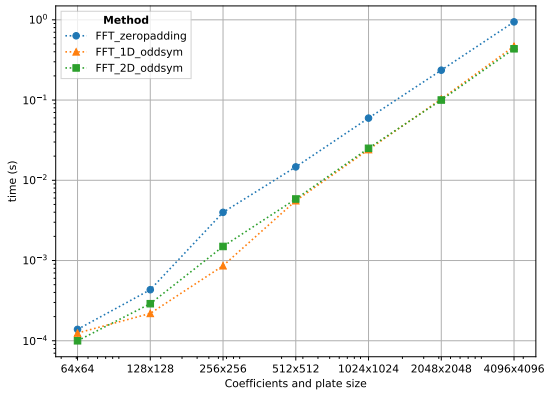
(b) FFTPACK with Intel i7-4700K



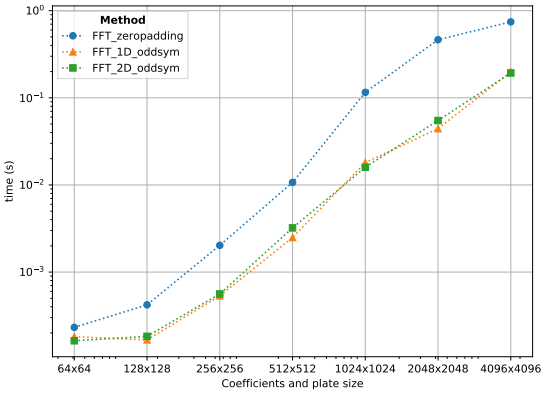
(c) MKL with Intel i5-6500



(d) MKL with Intel i7-4700K



(e) FFTW with Intel i5-6500



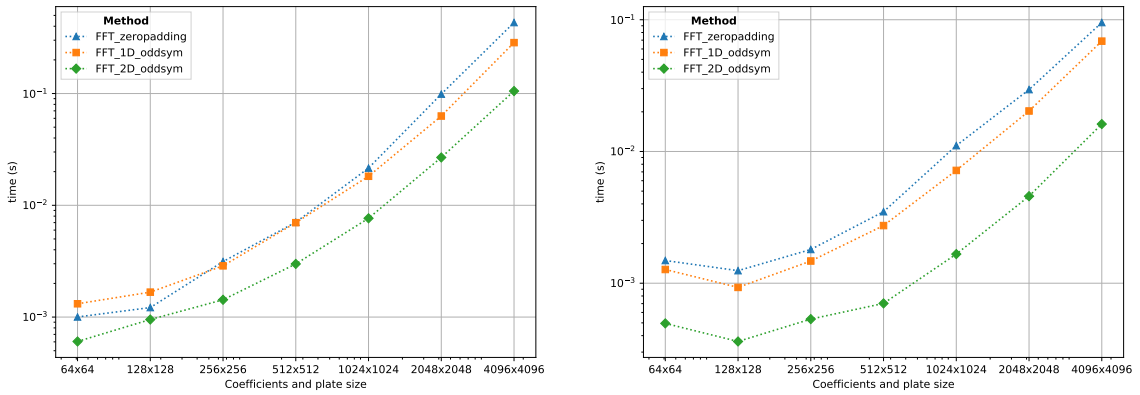
(f) FFTW with Intel i7-4700K

Figure IV-D.4.10: CPU computation times using the FFTPACK, MKL and FFTW libraries for the proposed schemes using different plate sizes. The odd symmetric schemes (1D and 2D) present the best performance overall.

The optimization degree achieved for the FFT algorithms with the MKL and FFTW libraries is higher. These libraries make better use of the underlying hardware, obtaining faster results than the FFTPACK library for the FFT-based methods. Results obtained with the FFTW library are slightly better (faster) than the MKL ones. However, this can be due to the usage of wrappers, as the pyfftw (FFTW) wrapper offers more control over the implementation. Nonetheless, the obtained results greatly surpass the state of art, both FFTW and MKL have shown execution times under $1s$ for plate sizes up to 4096×4096.

IV-D.4.4.2.2 GPU performance measurements

Fig. IV-D.4.11 shows the computation time for the three proposed FFT schemes using different GPU hardware: *i)* GeForce GTX 960 and *ii)* GeForce RTX 2060. The implementation is based in the cuFFT (CUDA toolkit) library and makes use of the PyCUDA and scikit-cuda python packages.



(a) NVIDIA GeForce GTX960

(b) NVIDIA GeForce RTX2060

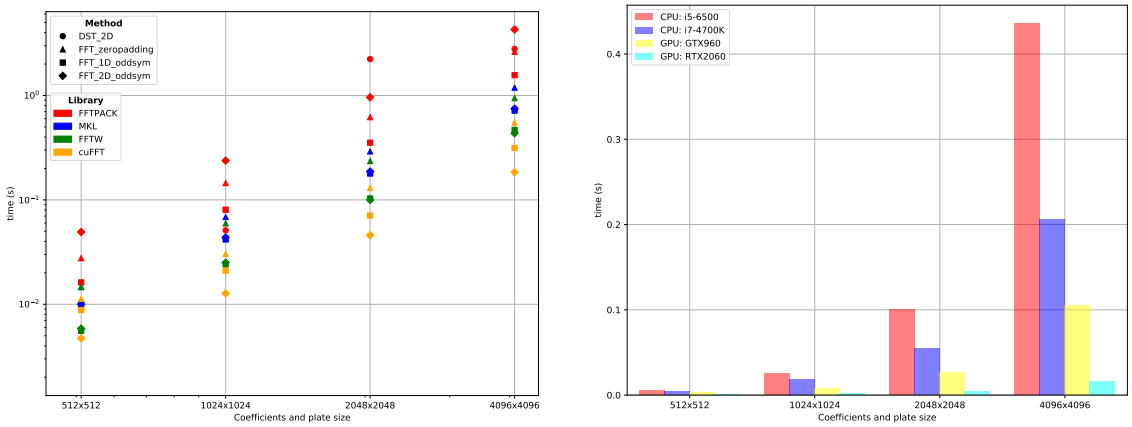
Figure IV-D.4.11: GPU computation times for the FFT-based methods using the cuFFT library with different plate sizes. The 2D odd symmetric scheme outperforms the remainder FFT-based ones.

While the zero padding and the 1D odd symmetric implementations produce similar results (in terms of computation time), the 2D odd symmetric scheme is by far the most performant. As the Fourier coefficients can be computed in the GPU before performing the temperature computation, the input for the FFT is already in GPU memory. It is worth to point out that the transfer of these coefficients from host memory (CPU) to device memory (GPU) is not measured.

IV-D.4.4.2.3 Comparison of CPU and GPU performance

Fig. IV-D.4.12(a) shows an overview of the computation times for the proposed DST (FFTPACK only) and FFT (FFTPACK, MKL, FFTW and cuFFT) methods. The FFTPACK (red) is the slowest and cuFFT (yellow) is the fastest. Execution times for both the MKL (blue) and FFTW (green) libraries are similar, obtaining slightly faster results with FFTW. Overall, the GPU hardware acceleration (with cuFFT) provides a considerable speed-up, making it a good alternative to consider for simulations on plates with large discretization sizes.

Fig. IV-D.4.12(b) compares the execution times of the two test platforms considering both CPU and GPU devices for the most performant FFT method: the 2D odd symmetric algorithm. This comparison shows that the GPU hardware effectively accelerates the computation time, between the fastest CPU (i7-4700K) and the slowest GPU (GTX 960), obtaining up to a $2\times$ speed-up for plate sizes larger than 1024×1024. The performance difference increases as the input plate increases in size. Using more recent GPU hardware (RTX 2060) results show a bigger difference in the achievable compute time speed-up.



(a) Computing time for each scheme, grouped by library as per Table IV-D.4.3

(b) Computing time for the 2D odd symmetric FFT scheme

Figure IV-D.4.12: CPU and GPU computation time comparison using different plate resolutions

IV-D.4.4.2.4 Comparison against state of the art

Fig. IV-D.4.13 compares the proposed FFT method with the state of the art (SoA) GPU brute-force solution [170]. The presented FFT method is much faster for plate sizes larger than 128×128, showing a big difference in computing times with a plate of size 1024×1024, where the FFT approach obtains a $124\times$ speed-up ($2.255s$ against $0.018s$). Fig. IV-D.4.13 demonstrates the potential of the presented FFT method to perform the temperature evaluation for high resolution plate sizes (1024×1024 and beyond). Furthermore, the current brute-force solution [170] has a limit size of 1024×1024 due to GPU shared memory usage, while the proposed FFT approach can compute the temperature for plates of sizes up to 4096×4096 under the same GPU hardware, without resorting to out-of-core GPU memory management. For small plate sizes (smaller than 128×128), the brute-force approach is faster due to the FFT method requiring extra processing of input coefficients and dispatching of kernels (scheduling time), adding a small computation overhead.

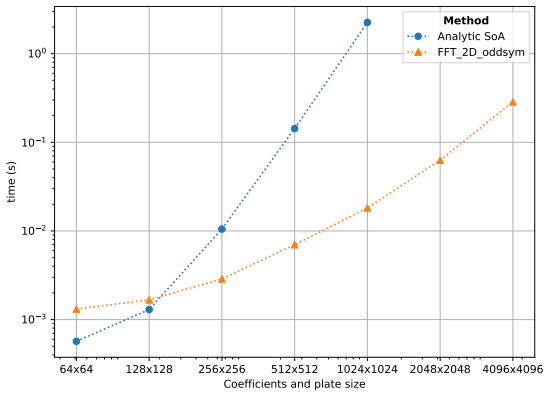
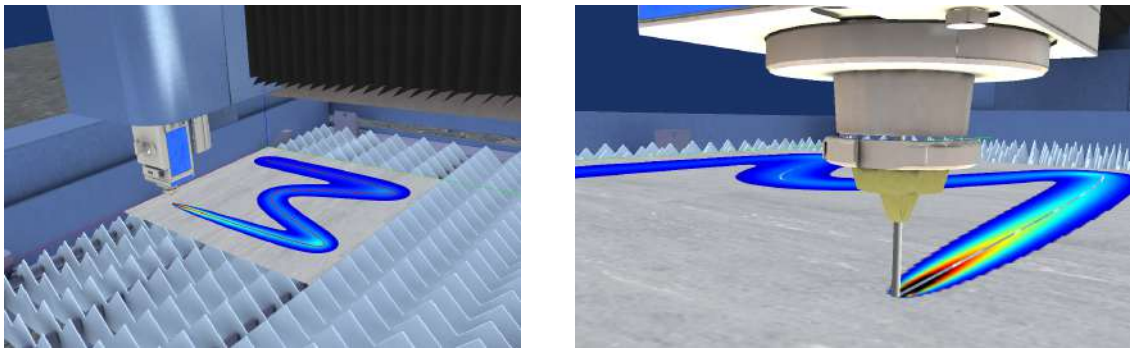


Figure IV-D.4.13: Appraisal of the computation times using an NVIDIA GeForce GTX960 (GPU) for the presented 2D odd symmetric FFT method vs the brute-force method presented in [170]

IV-D.4.4.3 Interactive Simulator Prototype

This section presents the integration of the presented FFT-based schemes into a 3D interactive simulator for CNC (Computer Numeric Control) laser machining. The prototype integrates a physical module for the temperature computation and a geometry module that computes the plate cutting through time (see Ref. [54]). The physical module implements the GPU-based FFT algorithms presented in this manuscript for the temperature computation at interactive rates while the geometry module performs boolean operations as discussed in Refs. [138, 183].

The current prototype provides interactive simulation of the laser heating/cutting process, visualized as a continuous animation. Interactively, the user can inspect the plate and its temperature at any specific timestep. Furthermore, the fast computation speed enables the possibility to run different simulations with different parameters in an interactive manner. Fig. IV-D.4.14 shows the virtual simulator for the test case discussed in this manuscript.



(a) M-trajectory interactive simulation

(b) Close-up view of the laser head and laser spot

Figure IV-D.4.14: Interactive laser heating/cutting simulator. A virtual CNC machine follows the laser trajectory defined by the program and the physical module computes the temperature using the FFT.

The development of interactive virtual worlds connected with physical objects (e.g. digital twins), has become a key technology for fast assessment of manufacturing processes [55]. In this context, an interactive CNC machine (as the integrated prototype) provides several tools to the engineer for the design of efficient CNC programs (plate, laser parameters and trajectory), reducing the requirement of real-world tests and consequently, reducing costs in terms of energy consumption, material waste, machining times, etc.

IV-D.4.5 Conclusions and Future Work

This manuscript presents four different schemes for the solution of the laser heating problem on thin metal plates using the DST and the FFT. The presented methods reduce the computational complexity of the problem from $\mathcal{O}(M^2N^2)$ to $\mathcal{O}(MN\log(MN))$ (with $M \times N$ being the discretization size of the metal plate).

These schemes are implemented in both CPU and GPU architectures using available optimized FFT libraries. Mathematical and numerical proofs of the correctness of the schemes are presented and the numerical error is measured below $10^{-10}K$ (and independent of the laser trajectory).

The performance evaluation shows that the minimum achievable computation time varies in function of the used library, specially for big input sizes. Furthermore, the obtained results improve the state of the art [170] in both CPU and GPU platforms for all the proposed schemes. Specifically, using GPU hardware, the computation times for the temperature evaluation are reduced from 1s to 0.01s (100× faster), measured in an NVIDIA GeForce GTX 960 (GPU).

Authorcontributions: D.M.-P., A.A. and A.M. conceived, designed and implemented the algorithms and performed the simulations. O.R.-S. and J.P. supervised the Computational Geometry, Heat Transfer and Spectral Methods research. J.L.-P. supervised the Parallel Computing, Data Structures and High Performance Programming aspects of this research. All the authors contributed to the writing of this manuscript.

Funding: This research received no external funding.

Conflictsofinterest: The authors declare no conflict of interest.

IV-E

In-line Dimensional Assessment of Forged Parts

IV-E.1

Perfect Spatial Hashing for Point-cloud-to-mesh Registration

Daniel Mejia-Parra[1,2], Juan Lalinde-Pulido[3], Jairo R. Sánchez[2], Oscar Ruiz-Salguero[1] and Jorge Posada[2]

[1] Laboratory of CAD CAM CAE, Universidad EAFIT, Colombia

[2] Vicomtech, España

[3] High Performance Computing Facility APOLO, Universidad EAFIT, Colombia

Citation

Daniel Mejia-Parra, Juan Lalinde-Pulido, Jairo R. Sánchez, Oscar Ruiz-Salguero and Jorge Posada (**2019**). Perfect Spatial Hashing for point-cloud-to-mesh registration. In *Spanish Computer Graphics Conference (CEIG 2019)*. June 26-28, Donostia - San Sebastián, Spain. ISBN: 978-3-03868-093-2. DOI: 10.2312/ceig.20191202.

Indexing: EUROGRAPHICS Digital Library

Abstract

Point-cloud-to-mesh registration estimates a rigid transformation that minimizes the distance between a point sample of a surface and a reference mesh of such a surface, both lying in different coordinate systems. Point-cloud-to-mesh-registration is an ubiquitous problem in medical imaging, CAD CAM CAE, reverse engineering, virtual reality and many other disciplines. Common registration methods include Iterative Closest Point (ICP), RANdom SAmple Consensus (RANSAC) and Normal Distribution Transform (NDT). These methods require to repeatedly estimate the distance

between a point cloud and a mesh, which becomes computationally expensive as the point set sizes increase. To overcome this problem, this article presents the implementation of a Perfect Spatial Hashing for point-cloud-to-mesh registration. The complexity of the registration algorithm using Perfect Spatial Hashing is $\mathcal{O}(N_Y \times n)$ (N_Y: point cloud size, n: number of max. ICP iterations), compared to standard octrees and kd-trees (time complexity $\mathcal{O}(N_Y \log(N_T) \times n)$, N_T: reference mesh size). The cost of pre-processing is $\mathcal{O}(N_T + (N_H^3)^2)$ (N_H^3: Hash table size). The test results show convergence of the algorithm (error below 7e-05) for massive point clouds / reference meshes ($N_Y = 50k$ and $N_T = 28055k$, respectively). Future work includes GPU implementation of the algorithm for fast registration of massive point clouds.

CCS Concepts
Theory of computation → Convex optimization; Computational geometry
Computing methodologies → Mesh models; Point-based models
Applied computing → Computer-aided design

IV-E.1.1 Introduction

Point set registration is ubiquitous in Reverse Engineering, Medical Imaging, Visual (Dimensional) Inspection, Robotics, among other disciplines.

Consider two point set samples of an object, each one conducted in its own coordinate system. The points in one set do not exactly correspond to object locations sampled in the other set. Moreover, parts of the object visible in one coordinate system may be unaccessible for sample in the other coordinate system (e.g. two clipped depth scans of the same object). The point set registration problem consists of finding a rigid transformation that rotates and translates one point set onto the other, producing the best possible matching between the transformed and the static point sets.

Point set registration is strongly qualified by the underlying structure of the point sets. Registration of surface point samples is very different from registration of point samples obtained from the interior of the same object (such as the volumetric point sets obtained from Computed Tomography Scans) [184]. It is an important advantage the fact that a 2-manifold structure (i.e. non self-intersecting surface) might be recognized as underlying the point sets. The present publication refers to registration between a point set which is optically sampled on an object surface vs. a triangular mesh (i.e. a planar triangular graph) obtained from a CAD representation of the object. The problem of point-cloud-to-mesh registration is relevant in CAD CAM CAE applications where the CAD (or triangular mesh) model of the object to register is known a priori. These applications include (but are not limited to) Dimensional Inspection [169, 185] and Robotic Bin Picking [186].

Within point-cloud-to-mesh registration, the sub-problem of point-cloud-to-mesh distance is central and heavily contributes to the computing expenditure. For the later problem, existing literature relies on spatial partition structures (such as octrees or kd-trees), which produce logarithmic search times. Given the massive amount of points of the sets to be registered, it is of interest to find a more economic strategy. Therefore, this manuscript presents the implementation of a point-cloud-to-mesh registration algorithm based on a Spatial Hashing data structure. This Spatial Hashing structure provides constant time access ($\mathcal{O}(1)$) to the list of close triangles to a given point $\mathbf{p}$. Consequently, the point-cloud-to-mesh registration based on Perfect Spatial Hashing is significantly faster than its hierarchical-based counterparts for massive point sets.

In this manuscript, Section 2 presents a literature review of relevant approaches. Section 3 conveys the methodology applied. Section 4 discusses the results obtained with several data sets. Section 5 concludes the manuscript and mentions possible related future enhancements to the present approach.

IV-E.1.2 Literature Review

The problem of point cloud registration has gotten a lot of research interest due to its relevancy in many engineering areas. Refs. [187, 188] present a survey on point cloud registration algorithms. The Iterative Closest Point (ICP) algorithm is one of the most widely used method for mesh registration in such literature. The algorithm consists of computing the closest points (correspondences) between the point cloud to register and the reference mesh. Such a procedure is performed iteratively until a convergence criteria is met [189]. The ICP extends the quaternion method [190] for correspondent point-to-point registration.

To avoid local minima, the ICP requires the point-cloud-to-register and the reference mesh to be locally close enough. User-assisted alignment of correspondences is used to compute a pre-registration of the point cloud, which is finally registered by the ICP [185]. Other ICP variations include feature-based mesh registration, in which some key points are automatically matched between the point-cloud-to-register and the reference mesh [187]. These feature-based registration methods rely on spherical harmonics [191] or surface signatures [192].

The main problem with ICP registration is the computation of correspondences (set of closest points from the reference mesh to the point cloud to register). The most naive approach is the exhaustive search, which is quadratic in time complexity $\mathcal{O}(N_Y \times N_T)$ (N_Y is the point-cloud-to-register size and N_T is the number of triangles in the reference mesh). Thus, spatial partitions of the domain are usually used to reduce the computational cost of the registration. Approaches to such spatial partitions include kd-trees [193], heuristic search [194], R-trees [195] and octrees [196], whose search complexity becomes $\mathcal{O}(N_Y \log(N_T))$. Refs. [197, 198] use 1-D hash tables to index octree entries, reducing the octree search to $\mathcal{O}(N_Y \log(\log(N_T)))$. Ref. [199] computes a regular grid that encloses the reference mesh, reducing the registration search complexity to linear $\mathcal{O}(N_Y)$. However, this last approach demands excessive storage resources as the full rectangular grid needs to be stored.

Other algorithms for cloud-to-mesh registration have been presented in the literature. RANdom SAmple Consensus (RANSAC) is a registration algorithm which takes many different sets of samples from the point cloud to register, and then fits a different model to each of these sets. The algorithm returns the best fitted model according to the optimization criteria [200]. The Normal Distribution Transform (NDT) algorithm computes a 3D grid enclosing the point cloud to register and the reference mesh, which are used to compute a spatial probability distribution function. The registration of the obtained probability functions is performed using the Hessian matrix method [201]. RANSAC and NDT methods have shown to perform faster than standard ICP methods. However, their result is non-deterministic and highly sensitive to algorithm parameters. A full review on mesh registration algorithms is presented in [187, 202].

IV-E.1.2.1 Conclusions of the Literature Review

Current mesh registration algorithms rely on spatial partitions of the 3D domain to search the cloud-to-mesh closest points. Most of these algorithms are linear-logarithmic. Table IV-E.1.1 summarizes

the mesh registration algorithms presented in the literature with their respective time complexity.

Table IV-E.1.1: Summary of closest point search algorithms in the literature. N_Y is the point-cloud-to-register size and N_T is the reference mesh size.

Reference	Computational Complexity
K-d tree search [193]	$\mathcal{O}(N_Y \log(N_T))$
Heuristic search [194]	$\mathcal{O}(N_Y \log(N_T))$
R-tree search [195]	$\mathcal{O}(N_Y \log(N_T))$
Octree search [196]	$\mathcal{O}(N_Y \log(N_T))$
Hash-Octree search [197, 198]	$\mathcal{O}(N_Y \log(\log(N_T)))$
Cubic grid search [199]	$\mathcal{O}(N_Y)$
Perfect Spatial Hash (this manuscript)	$\mathcal{O}(N_Y)$

To overcome these problems, this manuscript presents the integration and implementation of a Perfect Spatial Hashing [203] data structure into the ICP registration process. Given a point to be registered, the Perfect Spatial Hashing defines a hash function which returns the closest point from the reference mesh in constant time. As a consequence, the complexity of our registration algorithm is $\mathcal{O}(N_Y \times n)$, improving previous spatial partition approaches. In contrast to the discretization presented in [199], the Spatial Hash partition reduces significantly the storage requirements of the data structure, as the Hash table is optimized to reach the smallest size possible, at the cost of some pre-processing time.

IV-E.1.3 Methodology

Given a point cloud to register $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{N_Y}\}$ and a reference triangle mesh $\mathcal{M} = (\mathcal{T}, \mathbf{P})$ ($\mathcal{T} = \{t_1, t_2, \ldots, t_{N_T}\}$, $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{N_P}\}$), the mesh registration problem consists of finding a rigid transformation (rotation $\mathbf{R} \in SO(3)$ and translation $\mathbf{p}_0 \in \mathbb{R}^3$) that minimizes the distance between the point cloud $\mathbf{Y}$ and the reference mesh $\mathcal{M}$:

$$\min_{\mathbf{R},\mathbf{p}_0} \sum_{i=1}^{N_Y} d(\mathbf{R}\mathbf{y}_i + \mathbf{p}_0, \mathcal{M})^2 \tag{IV-E.1.1}$$

where $d(\mathbf{y}_i^*, \mathcal{M})$ is shortest distance between the registered point $\mathbf{y}_i^*$ and the mesh $\mathcal{M}$. The registered point cloud is the set of points $\mathbf{Y}^* = \{\mathbf{y}_1^*, \mathbf{y}_2^*, \ldots, \mathbf{y}_{N_Y}^*\}$ such that $\mathbf{y}_i^* = \mathbf{R}\mathbf{y}_i + \mathbf{p}_0$.

The following sections describe the Iterative Closest Point (ICP) algorithm [189] that solves the above minimization problem and the integration of Perfect Spatial Hashing [203] in the registration process.

IV-E.1.3.1 Mesh Registration of Correspondences

Let $\mathbf{x}_i \in \mathcal{M}$ be the closest point to the registered point $\mathbf{y}_i^*$ (see Fig. IV-E.1.1). The set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_Y}\}$ is a resample of $\mathcal{M}$, known as the set of correspondences of $\mathbf{Y}$. As a consequence, Eq. IV-E.1.1 becomes:

$$\min_{\mathbf{R},\mathbf{p}_0} \sum_{i=1}^{N_Y} \|\mathbf{R}\mathbf{y}_i + \mathbf{p}_0 - \mathbf{x}_i\|^2 \tag{IV-E.1.2}$$

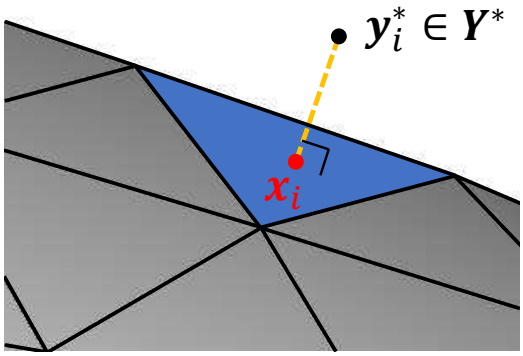


Figure IV-E.1.1: Registered point $\mathbf{y}_i^*$ and its correspondent (closest) point $\mathbf{x}_i \in \mathcal{M}$. $\mathbf{x}_i$ does not belong to the original discretization of $\mathcal{M}$.

It is worth noting that $\mathbf{X}$ and $\mathbf{Y}$ share the same number of points ($|\mathbf{X}| = |\mathbf{Y}| = N_Y$) and the set $\mathbf{X}$ does not contain the same points as the initial discretization of $\mathcal{M}$ (i.e. $\mathbf{X} \neq \mathbf{P}$). In addition, since the solution $\mathbf{Y}^*$ is an unknown of the problem, the set $\mathbf{X}$ is not known a priori. However, an estimation of $\mathbf{X}$ can be computed using the initial point set $\mathbf{Y}$. Such an estimation is discussed later in this section.

The minimization problem presented in Eq. IV-E.1.2 becomes the following maximization problem [189]:

$$\max_{\mathbf{R}} \sum_{i=1}^{N_Y} (\mathbf{y}_i - \mu_{\mathbf{y}})^T \mathbf{R} (\mathbf{x}_i - \mu_{\mathbf{x}}) \tag{IV-E.1.3}$$

and the optimal solution to $\mathbf{p}_0$ becomes:

$$\mathbf{p}_0 = \mu_{\mathbf{x}} - \mathbf{R}\mu_{\mathbf{y}} \tag{IV-E.1.4}$$

where $\mu_{\mathbf{x}} \in \mathbb{R}^3$ and $\mu_{\mathbf{y}} \in \mathbb{R}^3$ are the centroids of the point clouds $\mathbf{X}$ and $\mathbf{Y}$, respectively. Let $\mathbf{S}$ be the 3×3 cross-covariance matrix between $\mathbf{X}$ and $\mathbf{Y}$, defined as follows:

$$\mathbf{S} = \sum_{i=1}^{N_Y} (\mathbf{x}_i - \mu_{\mathbf{x}})(\mathbf{y}_i - \mu_{\mathbf{y}})^T \tag{IV-E.1.5}$$

The rotation $\mathbf{R}$ can be expressed as a unit quaternion $\dot{\mathbf{q}} \in \mathbb{R}^4$, $\|\dot{\mathbf{q}}\| = 1$. Using quaternion algebra [190], Eq. IV-E.1.3 becomes:

$$\max_{\|\dot{\mathbf{q}}\|=1} \sum_{i=1}^{N_Y} \dot{\mathbf{q}}^T \mathbf{Q}_i \dot{\mathbf{q}} = \max_{\|\dot{\mathbf{q}}\|=1} \dot{\mathbf{q}}^T \mathbf{Q} \dot{\mathbf{q}} \tag{IV-E.1.6}$$

where $\dot{\mathbf{q}} \in \mathbb{R}^4$ is the unit quaternion ($\|\dot{\mathbf{q}}\| = 1$) representation of $\mathbf{R}$ and $\mathbf{Q}_i$ is the 4×4 symmetric matrix associated to the cross-covariance $(\mathbf{x}_i - \mu_{\mathbf{x}})(\mathbf{y}_i - \mu_{\mathbf{y}})^T$. The matrix $\mathbf{Q}$ ($\mathbf{Q} = \sum_i \mathbf{Q}_i$) is defined in terms of the cross-covariance matrix $\mathbf{S}$ as follows [190]:

$$\mathbf{Q} = \begin{bmatrix} S_{00}+S_{11}+S_{22} & S_{12}-S_{21} & S_{20}-S_{02} & S_{01}-S_{10} \\ S_{12}-S_{21} & S_{00}-S_{11}-S_{22} & S_{01}+S_{10} & S_{02}+S_{20} \\ S_{20}-S_{02} & S_{01}+S_{10} & S_{11}-S_{22}-S_{00} & S_{12}+S_{21} \\ S_{01}-S_{10} & S_{02}+S_{20} & S_{12}+S_{21} & S_{22}-S_{00}-S_{11} \end{bmatrix} \tag{IV-E.1.7}$$

Finally, Eq. IV-E.1.6 has the form of a Rayleigh quotient, thus becoming an eigenvector problem. The optimal rotation $\dot{\mathbf{q}}$ that registers the set of correspondences $\mathbf{X}, \mathbf{Y}$ is the eigenvector of the matrix $\mathbf{Q}$, corresponding to its largest eigenvalue.

IV-E.1.3.2 Iterative Closest Point

As previously discussed, the set of correspondences $\mathbf{X}$ is not known a priori since the solution $\mathbf{y}_i^* = \mathbf{R}\mathbf{y}_i + \mathbf{p}_0$ is not known. The ICP algorithm [189] proposes to estimate a sequence of correspondences $\mathbf{X}^{(k)}$ based on a previous known point cloud $\mathbf{Y}^{(k-1)}$. The correspondent point $\mathbf{x}_i^{(k)} \in \mathcal{M}$ is the closest point in $\mathcal{M}$ to the point $\mathbf{y}_i^{(k-1)}$:

$$\mathbf{x}_i^{(k)} = \arg\min_{\mathbf{x}\in\mathcal{M}} \|\mathbf{x} - \mathbf{y}_i^{(k-1)}\| \tag{IV-E.1.8}$$

In Eq. IV-E.1.8 it is reasonable to assume that $\|\mathbf{x}_i^{(k)} - \mathbf{y}_i^{(k-1)}\| < \Delta$, with $\Delta > 0$ being a distance threshold. This assumption means that the point cloud $\mathbf{Y}^{(k-1)}$ is locally close enough to the reference mesh $\mathcal{M}$ (i.e., $d(\mathbf{y}_i^{(k-1)}, \mathcal{M}) < \Delta$). Any point $\mathbf{y}_i^{(k)}$ not satisfying such assumption is discarded from $\mathbf{Y}^{(k-1)}$. Such an assumption is made in order to: (1) avoid falling in local minima and, (2) filter outliers from $\mathbf{Y}^{(k-1)}$ [189]. Other methods already presented in the literature can be used as a pre-processing to guarantee that most of the points in $\mathbf{Y}$ satisfy the previous assumption before our algorithm starts [185].

With such a set of correspondences, it is possible to solve the optimization problem presented in Eq. IV-E.1.2, which becomes:

$$\min_{\mathbf{R}^{(k)}, \mathbf{p}_0^{(k)}} \sum_{i=1}^{N_Y} \|\mathbf{R}^{(k)}\mathbf{y}_i^{(k-1)} + \mathbf{p}_0^{(k)} - \mathbf{x}_i^{(k)}\|^2 \tag{IV-E.1.9}$$

where $\mathbf{R}^{(k)} \in SO(3)$, $\mathbf{p}_0^{(k)} \in \mathbb{R}^3$ originate the rigid transformation at the current iteration k. Finally, the point cloud $\mathbf{Y}^{(k)}$ is updated by using the obtained transformation:

$$\mathbf{y}_i^{(k)} = \mathbf{R}^{(k)}\mathbf{y}_i^{(k-1)} + \mathbf{p}_0^{(k)} \tag{IV-E.1.10}$$

The sequences $\mathbf{Y}^{(k)}$, $\mathbf{R}^{(k)}$ and $\mathbf{p}_0^{(k)}$ have been proved to converge to the optimal solution $\mathbf{Y}^*$, $\mathbf{R}$ and $\mathbf{p}_0$, respectively [189]:

$$\begin{aligned}
\mathbf{y}_i^* &= \lim_{n\to\infty} \mathbf{y}_i^{(n)} \\
\mathbf{R} &= \lim_{n\to\infty} \prod_{i=0}^{n} \mathbf{R}^{(k)} \\
\mathbf{p}_0 &= \lim_{n\to\infty} \left[\sum_{k_1=0}^{n-1} \left(\prod_{k_2=k_1+1}^{n} \mathbf{R}^{(k_2)} \right) \mathbf{p}_0^{(k_1)} \right] + \mathbf{p}_0^{(n)}
\end{aligned} \tag{IV-E.1.11}$$

The ICP works iterating over $k = 1, 2, \ldots, n$ for the previous sequences, until either one of the following criteria is satisfied:

1. Max. number of iterations n reached.

2. Approximation error below a given threshold ($\sum_i \frac{\|\mathbf{y}_i^{(k)} - \mathbf{y}_i^{(k-1)}\|^2}{N_Y} < \epsilon$)

The algorithm is initialized from the original point cloud $\mathbf{Y}^{(0)} = \mathbf{Y}$, and the identity transformation $\mathbf{R}^{(0)} = \mathbf{I}_{3\times3}$, $\mathbf{p}_0^{(0)} = \mathbf{0}_{3\times1}$. Fig. IV-E.1.2 summarizes the mesh registration algorithm. The most expensive procedure in the ICP algorithm is the computation of the cloud-to-mesh distance (steps 4 and 5), which computed by an exhaustive search drives the complexity of the registration to $\mathcal{O}(N_Y \times N_T \times n)$, with N_Y being the point cloud size, N_T being the number of triangles in the mesh $\mathcal{M}$ and n being the maximum number of ICP iterations. It is common in the literature to use hierarchical partition structures (such as kd-trees and octrees) which improve such a search to $\mathcal{O}(N_Y \log(N_T) \times n)$. Our registration algorithm implements instead a Perfect Spatial Hashing strategy (step 1), whose search complexity is constant ($\mathcal{O}(1)$) [203]. As a consequence, the overall time complexity of our mesh registration algorithm becomes $\mathcal{O}(N_Y \times n)$. The following sections discuss the construction of the Spatial Perfect Hash and the distance computation.

Figure IV-E.1.2: Scheme of the Iterative Closest Point mesh registration algorithm. Our registration uses Perfect Spatial Hashing to compute the cloud-to-mesh distances.

IV-E.1.3.3 Perfect Spatial Hash

Given a triangular mesh $\mathcal{M} \subset \mathbb{R}^3$, consider $\mathcal{V} \subset \mathcal{P}(\mathbb{R}^3)$ ($\mathcal{P}(\cdot)$ is the power set) as a rectangular prism, oriented along the coordinate axes, which contains $\mathcal{M}$ and is the union of small (disjoint) cubic cells (voxels v_{ijk}) of side length Δ (Fig. IV-E.1.3):

$$\mathcal{V} = \{v_{ijk} | i \in [0, N_V) \wedge j = [0, N_V) \wedge k \in [0, N_V)\} \tag{IV-E.1.12}$$

where each voxel v_{ijk} is also oriented along the coordinate axes, and the interiors of two different voxels never intersect.

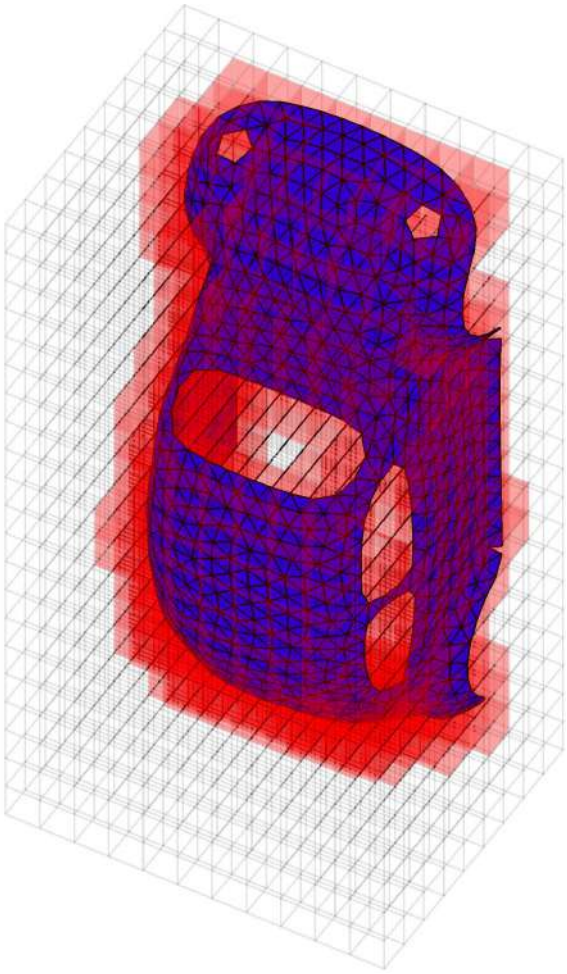


Figure IV-E.1.3: Full min-max voxel set $\mathcal{V}$ (gray). Non triangle-empty voxel set $\mathcal{V}_M$ (red). Triangle mesh $\mathcal{M}$ (blue). $|\mathcal{V}_M| << |\mathcal{V}|$.

The size of the previous spatial partition is $|\mathcal{V}| = N_V^3$, with $i < N_V$, $j < N_V$ and $k < N_V$ being the 3D indices of each voxel. Define $D(v_{ijk})$ as the triangles of $\mathcal{M}$ that intersect v_{ijk} (Fig. IV-E.1.4), i.e.:

$$D(v_{ijk}) = \{t \in \mathcal{T} | t \cap v_{ijk} \neq \emptyset\} \tag{IV-E.1.13}$$

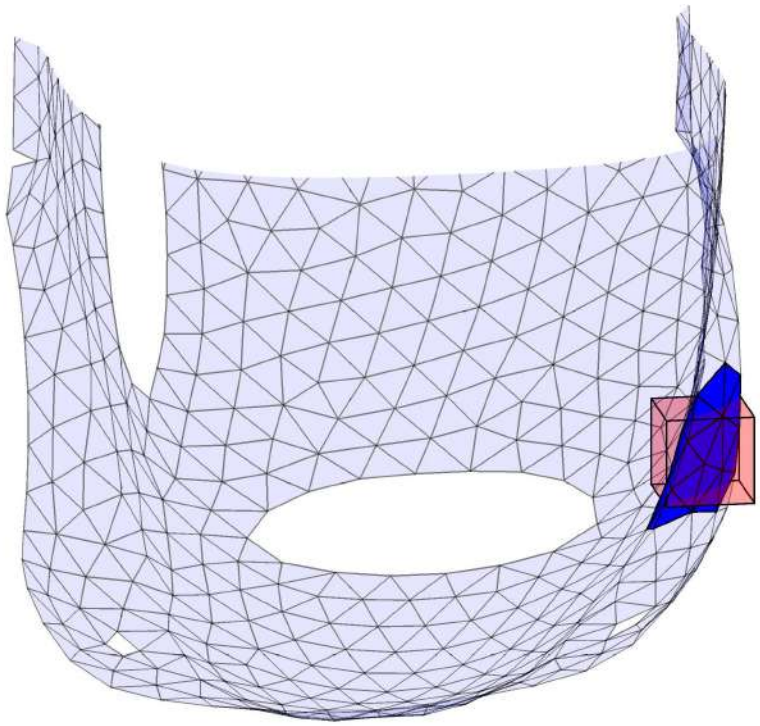


Figure IV-E.1.4: Set of triangles $D(v_{ijk})$ (dark blue) that intersect the voxel v_{ijk} (red)

Finally, the set $\mathcal{V}_M \subset \mathcal{V}$ is the set of voxels $v_{ijk} \in \mathcal{V}$ that intersect at least one triangle of $\mathcal{M}$, i.e. $\mathcal{V}_M = \{v_{ijk} \in \mathcal{V} | D(v_{ijk}) \neq \emptyset\}$. It is worth noting that the set size $|\mathcal{V}_M|$ is much smaller than the full grid size $|\mathcal{V}|$ (Fig. IV-E.1.3).

A Perfect Spatial Hash table $\mathbf{H} : \mathbb{N}^3 \rightarrow \mathcal{P}(\mathcal{T})$, is a 3D table with indices h_i, h_j, h_k. Each entry $\mathbf{H}[h_i, h_j, h_k]$ contains the set of triangles associated to the voxel $\mathbf{h}^{-1}(h_i, h_j, h_k)$, i.e.:

$$\mathbf{H}[\mathbf{h}(v_{ijk})] = \mathbf{H}[h_i, h_j, h_k] = D(v_{ijk}) \tag{IV-E.1.14}$$

where $\mathbf{h} : \mathcal{V}_M \rightarrow \mathbb{N}^3$ is a function which takes a voxel v_{ijk} and returns its respective position indices h_i, h_j, h_k in the Hash table $\mathbf{H}$. $\mathbf{h}$ is known as the hash function of $\mathbf{H}$. The Perfect Spatial Hash is denoted as $(\mathbf{H}, \mathbf{h})$.

The objective of the Perfect Spatial Hash is to produce a table $\mathbf{H}$ which stores the information $\mathcal{V}_M$, and its respective hash function $\mathbf{h}$. A trivial hash function would be the identity function $\mathbf{h}(v_{ijk}) = [i, j, k]$ (implicitly used by [199]). However, such a function implies storing the full rectangular prism $\mathcal{V}$ in the table $\mathbf{H}$ ($|\mathbf{H}| = |\mathcal{V}| >> |\mathcal{V}_M|$), and the content of most of the table cells would be empty (most cells of $\mathcal{V}$ are empty, Fig. IV-E.1.3). Instead, the Perfect Spatial Hash [203] aims to produce the smallest table $\mathbf{H}$ possible able to store the set $\mathcal{V}_M$, such that $|\mathcal{V}_M| \leq |\mathbf{H}| << |\mathcal{V}|$ (ideally, $|\mathbf{H}| = |\mathcal{V}_M|$).

The Perfect Spatial Hashing $(\mathbf{H}, \mathbf{h})$ satisfies by definition the following conditions:

1. The function $\mathbf{h}$ is bijective. As a consequence, there are no collisions in the table $\mathbf{H}$ (i.e. different voxels in $\mathcal{V}_M$ never point to the same cell of $\mathbf{H}$).

2. The size of $\mathbf{H}$ is greater or equal than the size of $\mathcal{V}_M$ ($|\mathbf{H}| > |\mathcal{V}_M|$).

In addition, $(\mathbf{H}, \mathbf{h})$ should satisfy (by construction) the following conditions:

1. The size of $\mathbf{H}$ is smaller than the size of $\mathcal{V}$ ($|\mathbf{H}| < N_V^3$).

2. Evaluation of the hash function $\mathbf{h}$ should be $\mathcal{O}(1)$.

The first step to build the Spatial Hash $(\mathbf{H}, \mathbf{h})$ is to compute the table size $|\mathbf{H}| = N_H^3$, as the smallest table size able to store the set $\mathcal{V}_M$:

$$N_H = \arg\min_{N_H \in \mathbb{N}} |\mathcal{V}_M| \leq N_H^3 \tag{IV-E.1.15}$$

The hash function $\mathbf{h}$ is then defined as a sum of an auxiliar function $\mathbf{f}$ and a displacement Φ [203]:

$$\mathbf{h}(v_{ijk}) = \mathbf{f}(v_{ijk}) + \Phi[g(v_{ijk})] \tag{IV-E.1.16}$$

The auxiliar function $\mathbf{f} : \mathcal{V}_M \to \mathbb{N}^3$ is defined as:

$$\mathbf{f}(v_{ijk}) = [f_i, f_j, f_k] = [i, j, k] \mod N_H \tag{IV-E.1.17}$$

By taking the modulo of each of the voxel indices, the values of the function $\mathbf{f}$ are guaranteed to never exceed the size of the Hash table $\mathbf{H}$ (i.e. $f_i < N_H$, $f_j < N_H$ and $f_k < N_H$). The function $\mathbf{f}$ is not bijective as $N_H \leq N_V$. As a consequence, an auxiliar 3D table Φ is computed as follows:

Let $\Phi \circ \mathbf{g} : V_M \to \mathbb{N}^3$ be an (auxiliar) 3D table of size $N_\Phi^3, N_\Phi \neq N_H$, and its corresponding auxiliar function $\mathbf{g} : \mathcal{V}_M \to \mathbb{N}^3$. The objective of the table $(\Phi, \mathbf{g})$ is to provide a translation term $\Phi[\mathbf{g}(v_{ijk})] = [\phi_i, \phi_j, \phi_k]$ such that $\mathbf{f}(v_{ijk}) + \Phi[\mathbf{g}(v_{ijk})]$ is bijective, guaranteeing that there are no collisions in $\mathbf{H}$.

Similar to the auxiliar function $\mathbf{f}$, the function $\mathbf{g}$ is defined as:

$$\mathbf{g}(v_{ijk}) = [g_i, g_j, g_k] = [i, j, k] \mod N_\Phi \tag{IV-E.1.18}$$

where $g_i < N_\Phi$, $g_j < N_\Phi$ and $g_k < N_\Phi$ indicate the position of the voxel v_{ijk} in the auxiliar table Φ, i.e. $[\phi_i, \phi_j, \phi_k] = \Phi[g_i, g_j, g_k]$. It is worth noting that, by construction, $\mathbf{f} \neq \mathbf{g}$ (since $N_\Phi \neq N_H$).

Fig. IV-E.1.5 illustrates the aforementioned translation Φ. In the example, the non-empty voxels v_{11} and v_{33} map to the same $\mathbf{f}$ value. However, the same voxels map to a different $\mathbf{g}$ value. The Φ table stores the respective translations $\phi_{11} = [0, 0]$ and $\phi_{33} = [1, 1]$. The Perfect Hash Table presents no collisions as the hash function is bijective ($\mathbf{h}_{11} = [1, 1]$, $\mathbf{h}_{33} = [0, 0]$).

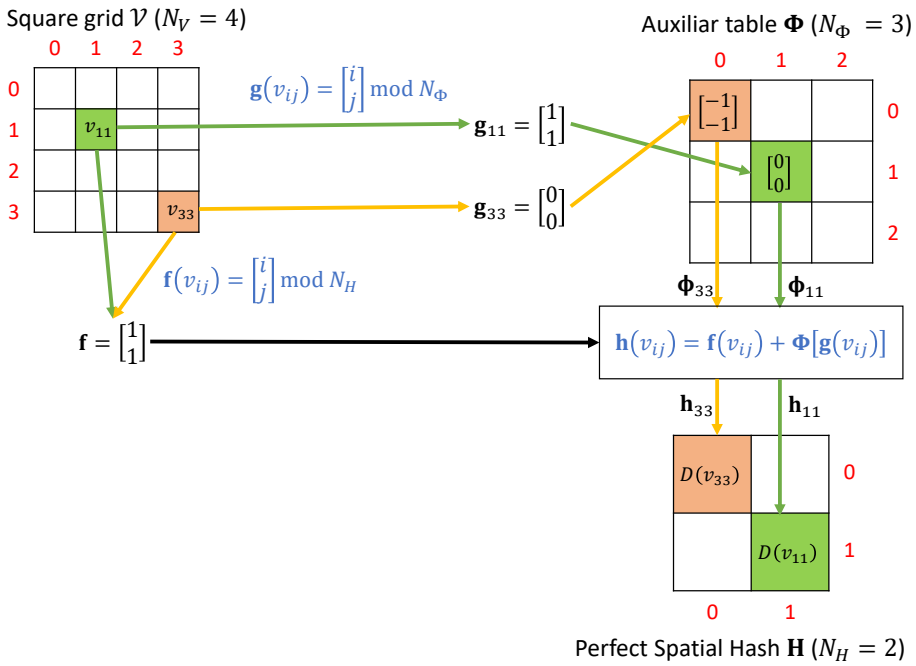


Figure IV-E.1.5: Perfect Spatial Hash 2D example. The auxiliar function $\mathbf{f}$ is not bijective, but the Hash function $\mathbf{h}$ is.

The table Φ and its size N_Φ is computed using an heuristic approach as described in Ref. [203], as follows:

1. Locate all collisions in $\mathbf{f}$.
2. Initialize the size of Φ as $N_\Phi \leftarrow \text{ceil}(\sqrt[3]{|\mathcal{V}_M|/6})$.
3. Initialize Φ as an empty N_Φ^3 3D table.
4. Locate all free indices of $\mathbf{f}$ (i.e. $\mathbf{f}(v_{ijk})$ is undefined).
5. For each collision $\mathbf{f}(v_{ijk})$, set $\Phi[\mathbf{g}(v_{ijk})]$ as $\mathbf{c} - \mathbf{f}(v_{ijk})$, where $\mathbf{c} = [c_i, c_j, c_k] \in \mathbb{N}^3$ is a free index in $\mathbf{f}$.
6. If there are no collisions in $\mathbf{f} + \Phi$, return Φ.
7. Otherwise, increase N_Φ and go to step 3.

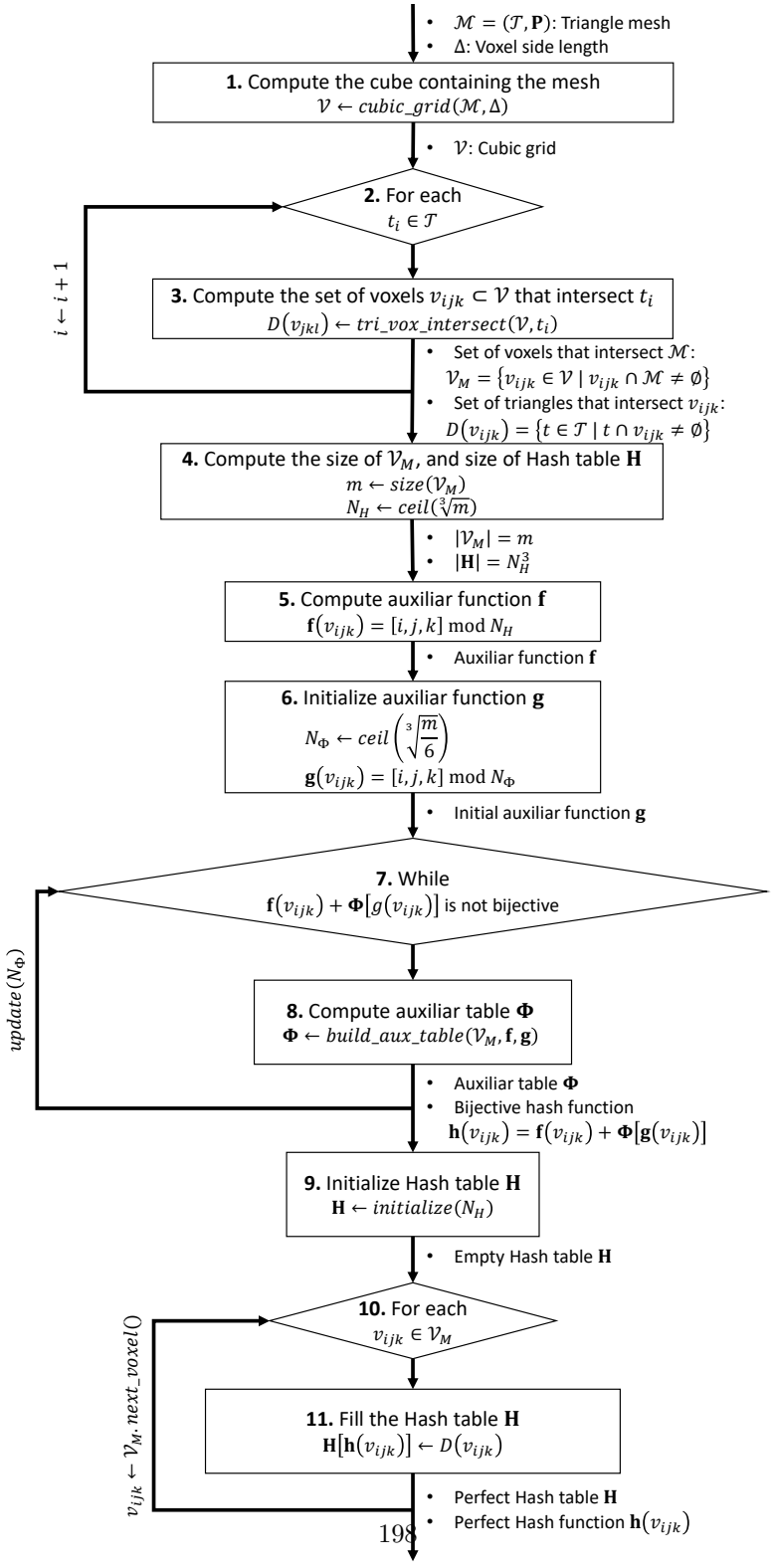


Figure IV-E.1.6: Algorithm scheme for the construction of the Perfect Spatial Hash $(\mathbf{H}, \mathbf{h})$

In the previous heuristic, it is worth noting that there is no theoretical guarantee that the computed Perfect Spatial Hash ($\mathbf{H}$ and Φ) is smaller than the full grid $\mathcal{V}$. In fact, it is possible that $|\mathbf{H}| + |\Phi|$ is larger than $|\mathcal{V}|$. However, our experiments and the experiments presented in [203] have shown that the Perfect Spatial Hash is always smaller than the full grid discretization (i.e., $|\mathbf{H}| + |\Phi| < |\mathcal{V}|$).

After Φ, $\mathbf{h}$ and N_H have been computed, the table $\mathbf{H}$ is filled with the elements of the set $\mathcal{V}_M$. At this point, the function $\mathbf{h}$ is guaranteed to be bijective and as a consequence, $\mathbf{H}$ presents no collisions. Fig. IV-E.1.6 summarizes the algorithm to compute the Perfect Spatial Hashing. Note that if the reference mesh $\mathcal{M}$ slightly changes (due to a small rigid transformation or shape deformation), the Perfect Spatial Hash table changes dramatically, requiring to rebuild it from scratch. However, since our registration algorithm assumes that $\mathcal{M}$ does not change at any time, the aforementioned problem is out of the scope of this research.

For the computation of the set of voxels that intersect the triangulation (i.e. $\mathcal{V}_M$), our algorithm visits each triangle of the mesh as illustrated in steps 2-3 of Fig. IV-E.1.6. The triangle-voxel intersection for each $t_i \in \mathcal{T}$ is implemented as follows: (1) all the voxels that intersect the bounding box of t_i are identified and then, (2) all the voxels inside the bounding box, which also intersect the plane defined by t_i are kept, discarding the non-intersecting ones.

From the algorithm presented in Fig. IV-E.1.6, steps 2-3 are $\mathcal{O}(N_T)$, steps 7-8 are $\mathcal{O}((N_H^3)^2)$ and steps 10-11 are $\mathcal{O}(N_H^3)$. Therefore, the computational cost for the Perfect Spatial Hash construction is $\mathcal{O}(N_T + (N_H^3)^2)$. Such a cost becomes reasonable for large point cloud and reference mesh sizes as this pre-processing is performed only once. In addition, the storage complexity of the Perfect Spatial Hash is $\mathcal{O}(N_H^3 + N_\Phi^3)$, which is considerably less expensive than storing the full grid $\mathcal{O}(N_V^3)$ (such as in Ref. [199]).

IV-E.1.3.4 Point-to-mesh Distance Computation

Given a point $\mathbf{y}_i \in \mathbf{Y}$, it is necessary to locate its closest point $\mathbf{x}_i \in \mathcal{M}$ (as per Eq. IV-E.1.8, Fig. IV-E.1.1). This problem is equivalent to find the closest triangle $t \in \mathcal{T}$ to $\mathbf{y}_i$, and then find the closest point $\mathbf{x}_i \in t$ to $\mathbf{y}_i$, as described below.

Given any triangle $t \in \mathcal{T}$, the distance from a point $\mathbf{y}_i \in \mathbf{Y}$ to t is defined as follows:

$$\begin{aligned} d(\mathbf{y}_i, t) = & \min_{\alpha,\beta \in \mathbb{R}} \|\alpha\mathbf{q}_0 + \beta\mathbf{q}_1 + (1-\alpha-\beta)\mathbf{q}_2 - \mathbf{y}_i\| \\ & s.t. \\ & \quad \alpha + \beta \leq 1 \\ & \quad \alpha, \beta \geq 0 \end{aligned} \tag{IV-E.1.19}$$

where $\mathbf{q}_0$, $\mathbf{q}_1$ and $\mathbf{q}_2$ are the vertices of the triangle t, and α, β, $(1-\alpha-\beta)$ are their corresponding barycentric coordinates, respectively. Therefore, the closest point $\mathbf{q}^* \in t$ to $\mathbf{y}_i$ is defined as the point $\mathbf{q}^* = \alpha\mathbf{q}_0 + \beta\mathbf{q}_1 + (1-\alpha-\beta)\mathbf{q}_2$ that minimizes Eq. (IV-E.1.19). The closest point $\mathbf{x}_i \in \mathcal{M}$ to $\mathbf{y}_i$ is defined as:

$$\mathbf{x}_i = \arg\min_{\mathbf{q}^* \in \mathcal{M}} \|\mathbf{y}_i - \mathbf{q}^*\| \tag{IV-E.1.20}$$

A naive evaluation of Eq. IV-E.1.20 requires searching the closest triangle t through the full mesh $\mathcal{M}$. However, the Perfect Spatial Hash $\mathbf{H}$ reduces such an evaluation by only requiring to evaluate triangles that are already close to $\mathbf{y}_i$. Let $v_{jkl} \in \mathcal{V}$ be the voxel that contains the point

$\mathbf{y}_i$. The Hash cell $\mathbf{H}[\mathbf{h}(v_{jkl})]$ stores the set of triangles $D(v_{jkl})$ that intersect v_{jkl} (as illustrated in Fig. IV-E.1.4).

Let $B_{jkl} \subset \mathcal{V}$ be the set of adjacent voxels to v_{jkl} (v_{jkl} included). The set of closest triangles to $\mathbf{y}_i$ can be extracted from the intersection between B_{jkl} and $\mathcal{M}$, i.e. the set $\mathbf{H}[\mathbf{h}(B_{jkl})]$ (see Fig. IV-E.1.7). Therefore, Eq (IV-E.1.20) is equivalent to:

$$\mathbf{x}_i = \arg \min_{\mathbf{q}^* \in \mathbf{H}[\mathbf{h}(B_{jkl})]} \|\mathbf{y}_i - \mathbf{q}^*\| \tag{IV-E.1.21}$$

where clearly $|\mathbf{H}[\mathbf{h}(B_{jkl})]| << \mathcal{T}$. Since each voxel side size is Δ, the set B_{jkl} is guaranteed to contain a triangle whose distance to $\mathbf{y}_i$ is less than Δ (if such triangle exists in $\mathcal{M}$). It is worth noting that if such triangle does not exist, then $d(y_i, \mathcal{M}) > \Delta$, and the registration algorithm treats $\mathbf{y}_i$ as an outlier (as discussed at the beginning of Sect. IV-E.1.3.2) [189].

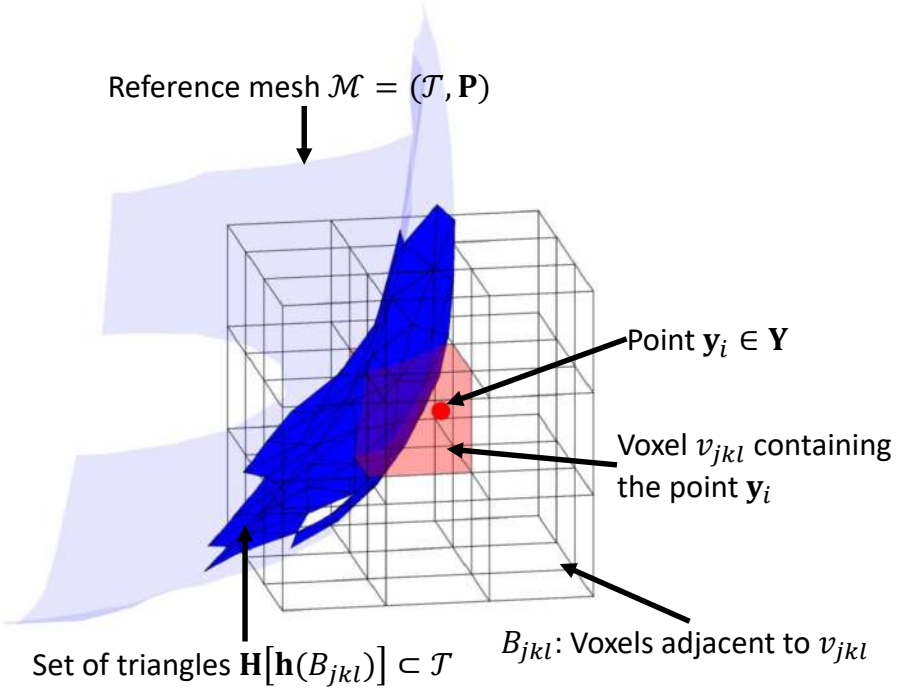


Figure IV-E.1.7: The closest point of $\mathcal{M}$ to $\mathbf{y}_i$ is in the set B_{jkl} ($|B_{jkl}| << |\mathcal{T}|$). B_{jkl} is the set of triangles that intersect v_{jkl} and all its adjacent voxels.

The algorithm for computing the closest point $\mathbf{x}_i$ is summarized as follows:

1. Compute the voxel v_{jkl} that contains the point to register $\mathbf{y}_i$ (i.e., $\mathbf{y}_i \in v_{jkl}$).
2. Compute the set of voxels B_{jkl}, adjacent to v_{jkl} (as illustrated in Fig. IV-E.1.7).
3. Compute the Hash indices $\mathbf{h}(B_{jkl})$ as per Eq. (IV-E.1.16).
4. Extract from the Spatial Hash, the triangles $\mathbf{H}[\mathbf{h}(B_{jkl})]$ closest to $\mathbf{y}_i$ (Fig. IV-E.1.7).

5. Compute the closest triangle $t \in \mathbf{H}[\mathbf{h}(B_{jkl})]$ as per Eq. IV-E.1.19.

6. Compute $\mathbf{x}_i$ as per Eq. (IV-E.1.21).

Since the evaluation of $\mathbf{h}$ in Eq. (IV-E.1.16) and the access to the table $\mathbf{H}$ is $\mathcal{O}(1)$, the computational cost of the above algorithm is $\mathcal{O}(1)$.

IV-E.1.4 Results

Four different models have been used to test our registration algorithm: Gargoyle, Dragon, Buddha and Lucy [204]. The point-cloud-to-register is extracted from the original model by computing a uniform re-sample of each model surface. Figs. IV-E.1.8(a), IV-E.1.8(c), IV-E.1.8(e) and IV-E.1.8(g) plot the unregistered point-clouds of each model, respectively. As mentioned in Sect. IV-E.1.3.2, the point-cloud-to-register should be close enough to the reference mesh to avoid falling into a local minima solution [189]. Figs. IV-E.1.8(b), IV-E.1.8(d), IV-E.1.8(f) and IV-E.1.8(h) plot the result of our registration process for each model, respectively. The registration algorithm minimizes the point-cloud-to-mesh distance as per Eq. (IV-E.1.1).

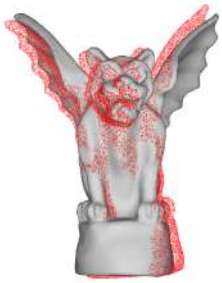
Table IV-E.1.2 shows Spatial Hashing and ICP convergence results of our registration algorithm. The 4 point-clouds-to-register are of size $N_Y = 50k$, while the size of the reference meshes (N_T) is 20k, 871.4k, 1631.6k and 28055.7k for the Gargoyle, Dragon, Buddha and Lucy, respectively. The smallest Spatial Hash constructed is for the Gargoyle dataset, consisting of a $N_H^3 = 512$ Hash table and a $N_\Phi^3 = 1.3\text{k}^3$ auxiliar table, and the largest Spatial Hash is constructed for the Lucy ($N_H^3 = 5.8\text{k}^3$ Hash table and $N_\Phi^3 = 2.2\text{k}^3$ auxiliar table). The convergence error is measured as the difference between the last iteration and the previous iteration $\frac{\sum_i \|y_i^{(n)} - y_i^{(n-1)}\|^2}{N_Y}$, as discussed in Sect. IV-E.1.3.2. All the 4 test cases converge at 34, 19, 30 and 53 ICP iterations (n), respectively, with an error below 7e-05.

Table IV-E.1.2: Perfect Spatial Hashing and ICP convergence results for the 4 datasets presented in Fig. IV-E.1.8

Dataset	N_Y	N_T	N_H^3	N_Φ^3	n	$\frac{\sum_i \|\mathbf{y}_i^{(n)} - \mathbf{y}_i^{(n-1)}\|^2}{N_Y}$
Gargoyle	50k	20k	512	1.3k	34	6.20e-05
Dragon	50k	871.4k	2.1k	4.9k	19	5.87e-05
Buddha	50k	1631.6k	3.4k	1.3k	30	6.06e-05
Lucy	50k	28055.7k	5.8k	2.2k	53	5.97e-05

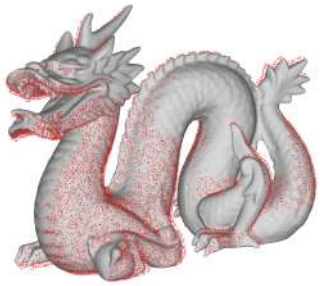
Table IV-E.1.3: Buddha dataset. Execution times for the construction of the Perfect Spatial Hashing $(\mathbf{H}, \mathbf{h})$ and the registration of a $N_Y = 50$k point cloud for different voxel resolutions N_V. Note that the performance for the registration significantly improves as N_V increases.

N_V^3	N_H^3	N_Φ^3	n	Time to build $(\mathbf{H}, \mathbf{h})$ (min)	Registration time (min)	Total time (min)
4096	343	216	31	0.0317	232.6	232.7
32.8k	1728	2197	30	0.0334	41.077	41.111
262.1k	8000	4913	32	0.0407	9.6378	9.6785
2097.2k	32.8k	19.7k	40	0.0587	2.6352	2.6939
16777.2k	140.6k	91.1k	0	0.2210	NA	NA



(a) Gargoyle mesh and unregistered point cloud

(b) Gargoyle point cloud registration



(c) Dragon mesh and unregistered point cloud

(d) Dragon point cloud registration

(e) Buddha mesh and unregistered point cloud

(f) Buddha point cloud registration

(g) Lucy mesh and unregistered point cloud

(h) Lucy point cloud registration

Figure IV-E.1.8: Point-cloud-to-mesh registration of 4 different models: Gargoyle, Dragon, Buddha and Lucy [204]. The registration algorithm minimizes the cloud-to-mesh distance.

Table IV-E.1.3 presents the execution times for the registration of a $N_V^3 = 50$k point cloud to the Buddha mesh (Figs. IV-E.1.8(e), IV-E.1.8(f)). For a prism of size $N_V^3 = 4096$, the construction of the Hash table requires 0.032 minutes, while the ICP registration takes about 232.6 minutes to perform 31 iterations and converge to the solution. Increasing the prism resolution to $N_V^3 = 32.8k$,

the construction of the Hash table requires 0.033 minutes while the registration takes 41.1 minutes to perform 30 iterations. In the case of a prism of size $N_V^3 = 2097.2k$, the construction of the Hash table and the mesh registration times are 0.06 and 2.6352 minutes, respectively, which is $15\times$ faster than the $N_V^3 = 32.8k$ and $86\times$ faster than the $N_V^3 = 4096$ test cases. Finally, the high resolution of the last test case ($N_V = 16777.2k$) implies that the voxel size Δ is significantly smaller than the average distance between the point cloud $\mathbf{Y}$ and the reference mesh $\mathcal{M}$, resulting in the registration algorithm exiting at 0 iterations without converging.

IV-E.1.5 Conclusions

This manuscript presents the implementation of a Perfect Spatial Hash Hashing for point-cloud-to-mesh registration. The registration algorithm uses the Perfect Spatial Hashing data structure to aid the computation of point-to-mesh distance of the Iterative Closest Point (ICP) algorithm. Compared to standard spatial partition techniques (such as octrees and kd-trees), our algorithm reduces the closest-point-search complexity from logarithmic ($\mathcal{O}(\log(N_T))$, N_T: reference mesh size) to constant $\mathcal{O}(1)$ complexity. As a consequence, the cost of the mesh registration algorithm becomes $\mathcal{O}(N_Y \times n)$ (N_Y: point-cloud-to-register size, n: number of max. ICP iterations). The cost of pre-processing (pre-computation of the Perfect Spatial Hashing) is $\mathcal{O}(N_T + (N_H^3)^2)$ (N_H^3: Hash table size). Our algorithm is able to register a point cloud of size $N_Y = 50k$ against a mesh of size $N_T = 28055.7k$, converging with an error below 7e-05. We also show that the mesh registration algorithm improves significantly in performance as the Spatial Hashing resolution increases. However, if the voxel size Δ becomes too small (smaller than the average distance between the point cloud and the reference mesh), the registration algorithm fails.

IV-E.1.5.1 Future Implementation on GPU

The main shortcoming of our point-cloud-to-mesh registration algorithm lies in the construction of the Perfect Spatial Hashing computational cost, as the worst case scenario complexity is squared in the size of the Hash table ($\mathcal{O}((N_H^3)^2)$, see Sect. IV-E.1.3.3). To mitigate this problem, we intend to implement Perfect Spatial Hash mesh registration in a Graphic Processing Unit (GPU) parallelization architecture. By taking advantage of Graphics Processing Units (GPUs), the Hash structure can be computed in a more efficient way, reducing the pre-processing time [203]. In addition, the independence in the computation of the closest point (Eq. (IV-E.1.21)) between any two different points $\mathbf{y}_i, \mathbf{y}_j \in \mathbf{Y}$ permits an implementation following a highly parallelizable approach, resulting in fast registration of considerably larger point clouds.

Glossary

ICP:	Iterative Closest Point.
$\mathcal{M}$:	Triangular mesh $\mathcal{M} = (\mathcal{T}, \mathbf{P})$ of a 2-manifold embedded in $\mathbb{R}^3$, defined by the triangle set $\mathcal{T} = \{t_1, t_2, \cdots, t_{N_T}\}$ and the point set $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{N_P}\}$. $\mathcal{M}$ is the reference mesh for registration.
$\mathbf{Y}$:	Point cloud to register $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{N_Y}\}$. $\mathbf{Y}$ is a noisy sample of $\mathcal{M}$, conducted in an unknown coordinate system.

$\mathbf{R}, \mathbf{p}_0$:	Rigid transformation $\mathbf{R} \in SO(3)$ (Special Orthogonal Group), $\mathbf{p}_0 \in \mathbb{R}^3$, that matches the coordinate system of $\mathbf{Y}$ to the coordinate system of $\mathcal{M}$.
$\mathbf{Y}^*$:	Rigidly transformed point cloud $\mathbf{Y}^* = \{\mathbf{y}_1^*, \mathbf{y}_2^*, \cdots, \mathbf{y}_{N_Y}^*\}$, such that $\mathbf{y}_i^* = \mathbf{R}\mathbf{y}_i + \mathbf{p}_0$.
$\mathbf{X}$:	Point cloud $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_{N_Y}\}$ sampled from $\mathcal{M}$, such that $\mathbf{x}_i$ is the closest point in $\mathcal{M}$ to $\mathbf{y}_i$ ($\lvert\mathbf{X}\rvert = \lvert\mathbf{Y}\rvert$). $\mathbf{X}$ is the *set of correspondences* of $\mathbf{Y}$.
$\mu_\mathbf{x}$, $\mu_\mathbf{y}$:	Centroids $\mu_\mathbf{x}, \mu_\mathbf{y} \in \mathbb{R}^3$ of the point sets $\mathbf{X}$ and $\mathbf{Y}$, respectively.
$\mathbf{S}$:	3×3 matrix of cross-covariances between $\mathbf{X}$ and $\mathbf{Y}$.
$\dot{\mathbf{q}}$:	Unit quaternion $\dot{\mathbf{q}} \in \mathbb{R}^4$ ($\lVert\dot{\mathbf{q}}\rVert = 1$), equivalent to the rotation matrix $\mathbf{R}$.
$\mathbf{Y}^{(k)}, \mathbf{X}^{(k)}$:	Values for the points sets $\mathbf{Y}$, $\mathbf{X}$ at the current ICP iteration k.
$\mathbf{R}^{(k)}, \mathbf{p}_0^{(k)}$:	Values for the rigid transformation $\mathbf{R}$, $\mathbf{p}_0$ at the current ICP iteration k.
n:	Maximal Number of iterations $n > 0$ allowed by the ICP algorithm.
Δ:	Distance below which a point $\mathbf{y}_i \in \mathbf{Y}$ is not considered an outlier w.r.t. mesh $\mathcal{M}$ (i.e. $d(\mathbf{y}_i, \mathcal{M}) < \Delta$).
$\mathcal{P}(A)$:	Power set of A, defined as all the subsets of A. $\mathcal{P}(A) = \{a \lvert a \subset A\}$.
v_{ijk}:	A cubic cell $(i, j, k) \in \mathbb{N}^3$, of side length Δ, oriented along the coordinate axes.
$\mathcal{V}$:	Rectangular prism $\mathcal{V} \subset \mathcal{P}(\mathbb{R}^3)$ oriented along the coordinate axes, defined as a set of disjoint voxels v_{ijk} that build the bounding box of $\mathcal{M}$. $\lvert\mathcal{V}\rvert = N_V^3$.
$D(v_{ijk})$:	Set of triangles in $\mathcal{M}$ that intersect voxel $v_{ijk} \in \mathcal{V}$. $D : \mathcal{V} \to \mathcal{P}(\mathcal{T})$.
$\mathcal{V}_M$:	Set of voxels $v_{ijk} \in \mathcal{V}$ that intersect at least one triangle of $\mathcal{M}$ (i.e. $D(v_{ijk}) \neq \emptyset$).
$\mathbf{H}$:	Perfect Spatial Hash table $\mathbf{H} : \mathbb{N}^3 \to \mathcal{P}(\mathcal{T})$. $\mathbf{H}$ is a 3D table where each entry $\mathbf{H}[h_i, h_j, h_k]$ stores a subset of triangles $D(v_{ijk})$. $\lvert\mathbf{H}\rvert = N_H^3$.
$\mathbf{h}$:	(Bijective) Hash function $\mathbf{h} : \mathcal{V}_M \to \mathbb{N}^3$ of $\mathbf{H}$. $\mathbf{h}$ takes a voxel $v_{ijk} \in \mathcal{V}_M$ and returns the respective indices h_i, h_j, h_k in $\mathbf{H}$, such that $H[h_i, h_j, h_k] = D(v_{ijk})$.
$\mathbf{f}, \mathbf{g}$:	Auxiliar functions $\mathbf{f}, \mathbf{g} : \mathcal{V}_M \to \mathbb{N}^3$ used by the function $\mathbf{h}$ to compute a bijective mapping.
Φ:	Auxiliar 3D table table $\Phi : \mathbb{N}^3 \to \mathbb{N}^3$ used by the function $\mathbf{h}$ to compute a bijective mapping. $\lvert\Phi\rvert = N_\Phi^3$.
$\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2$:	Vertices of triangle $t_j \in \mathcal{T}$ with $\mathbf{q}_i \in \mathbf{P}$.
α, β:	Barycentric coordinates on a triangle $t \in \mathcal{T}$ with $\alpha, \beta \geq 0$, and $\alpha + \beta \leq 1$.
B_{jkl}:	Set $B_{jkl} \subset \mathcal{V}$ of all adjacent voxels to v_{jkl} (including v_{jkl}).

IV-E.2

In-Line Dimensional Inspection of Warm-Die Forged Revolution Workpieces Using 3D Mesh Reconstruction

Daniel Mejia-Parra [1,2], Jairo R. Sánchez [2,*], Oscar Ruiz-Salguero [1], Marcos Alonso [3], Alberto Izaguirre [4], Erik Gil [5], Jorge Palomar [5] and Jorge Posada[2]

[1] Laboratory of CAD CAM CAE, Universidad EAFIT, Cra 49 no 7-sur-50, 050022 Medellín, Colombia; dmejiap@eafit.edu.co (D.M.-P.); oruiz@eafit.edu.co (O.R.-S.)
[2] Vicomtech, Paseo Mikeletegi 57, Parque Científico y Tecnológico de Gipuzkoa, 20009 Donostia/San Sebastián, Spain; jposada@vicomtech.org
[3] Computational Intelligence Group, CCIA Department, UPV/EHU, Paseo Manuel Lardizabal 1, 20018 Donostia/San Sebastián, Spain; malonso117@ikasle.ehu.es
[4] CIS & Electronics Department, University of Mondragon, Loramendi Kalea 5, 20500 Mondragon, Spain; aizagirre@mondragon.edu
[5] GKN Driveline Legazpi S.A., Calle Urola 10, 20230 Legazpi, Spain; Erik.Gil@gkndriveline.com (E.G.); Jorge.Palomar@gkndriveline.com (J.P.)

applied sciences

IMPACT FACTOR 2.217

an Open Access Journal by MDPI

Citation

Daniel Mejia-Parra, Jairo R. Sánchez, Oscar Ruiz-Salguero, Marcos Alonso, Alberto Izaguirre, Erik Gil, Jorge Palomar and Jorge Posada. In-line dimensional inspection of warm-die forged revolution workpieces Using 3D Mesh Reconstruction. *Applied Sciences*, **2019**, 9, pp. 1069:1-1069:21. DOI: 10.3390/app9061069.

Indexing: ISI (Q2), SCOPUS (Q1), Publindex (A1)

Abstract

Industrial dimensional assessment presents instances in which early control is exerted among "warm" (approx. 600 °C) pieces. Early control saves resources, as defective processes are timely stopped and corrected. Existing literature is devoid of dimensional assessment on warm workpieces. In response to this absence, this manuscript presents the implementation and results of an optical system which performs in-line dimensional inspection of revolution warm workpieces singled out from the (forming) process. Our system can automatically measure, in less than 60 s, the circular runout of warm revolution workpieces. Such a delay would be 20 times longer if cool-downs were required. Off-line comparison of the runout of T-temperature workpieces (27 ° C $\leq$ T $\leq$ 560 °C) shows a maximum difference of 0.1 mm with respect to standard CMM (Coordinate Measurement Machine) runout of cold workpieces (27 °C), for workpieces as long as 160 mm. Such a difference is acceptable for the forging process in which the system is deployed. The test results show no correlation between the temperature and the runout of the workpiece at such level of uncertainty. A prior-to-operation Analysis of Variance (ANOVA) test validates the repeatability and reproducibility (R&R) of our measurement system. In-line assessment of warm workpieces fills a gap in manufacturing processes where early detection of dimensional misfits compensates for the precision loss of the vision system. The integrated in-line system reduces the number of defective workpieces by 95%.

Keywords: in-line dimensional inspection; warm forming; 3D mesh reconstruction; optical system; revolution workpiece

IV-E.2.1 Introduction

In the context of warm forming of motorcar parts, current production lines of stub axles process around 1200 pieces per hour. The tools used to form these parts are constantly subjected to high structural and thermal stresses [205–207], requiring continuous monitoring and dimensional assessment of the produced parts for process and quality control.

In the case of forged revolution workpieces, the produced parts are not final product, requiring subsequent machining operations. The assessment of the punch orientation with respect to the forming matrix orientation in the forging process is crucial since a severe misalignment between the punch press and the forming matrix axes disables the posterior machining process, resulting in a scrapped part. The circular runout [208] of the forged revolution workpiece indicates the deviation between the punch orientation and the forming matrix axis.

Standard tools for dimensional assessment of these workpieces rely on contact between the probe and the measured workpiece. Such is the case of Coordinate Measurement Machines (CMMs), which provide highly accurate measurements [209]. However, dimensional assessment with standard CMMs (and contact methods in general) is not convenient due to (1) the high temperatures directly affect (or even damage) the probe and, (2) long measurement times for the cooled-down workpieces. Consequently, a delay of nearly 20 min between the production of a single part and its dimensional assessment (including its cooling down, transportation to the metrology office and measuring times) arises. Such time delay translates into an uncertainty in the quality control process of approximately 400 potential defective workpieces (worst case scenario) for each measurement.

This manuscript presents an optical (i.e., contact-avoiding) system for in-line dimensional assessment of warm forming of revolution workpieces. Our system can continuously measure the

circular runout of the parts at around 600 °C in less than 60 s per part. Results from experiments conducted in this manuscript show no temperature vs. runout correlation for the system uncertainty level (0.1 mm). The system is integrated and deployed in a warm-die forge industry, in the supply chain of world-class auto makers. Such a system allows continuous monitoring for quality and process control of the production line, providing an early detection mechanism of manufacturing failures which reduces the number of potential defective parts from 400 to 20 (95%) between consecutive measurements. This dimensional assessment for warm workpieces fills a gap in warm-die manufacturing processes in which the advantage of early detection of process bias compensates for the disadvantage of precision loss regarding higher-precision mechanisms (such as contact-based CMM).

The deployed system improves on the classic approaches for dimensional assessment in warm-die industry. Using technologies from Visual Computing and Industry 4.0 [55], the system allows in-line visual assessment of warm workpieces, either by metrologists, engineers, or operators. Furthermore, the increased cadence of the measurements (from 1 measurement every 20 min to 1 measurement per minute) improves the efficiency in product quality and process control, and leaves open future lines of dimensional assessment focused on data analytics.

Table IV-E.2.1: Comparison between standard contact-based Coordinate Measurement Machines (CMMs) vs optical scanners for dimensional assessment.

CMM	**Optical Scanner**
Highly accurate measurements [209].	Less-accurate measurements [209].
Data collection relying on probe vs. piece contact.	Contact probe vs. piece not required.
Technician assistance required for definition of piece feature coordinate systems [210].	Technician assistant required for point sample vs. B-Rep (i.e., CAD model) registration. [211, 212]
Time-consuming data acquisition protocol [210].	Real-time data acquisition and post-processing of the digitized mesh (triangulation, mesh registration, feature extraction) [213, 214].
Inherently sparse point samples, conducted according to discrete trajectories. Analytic form fitting needed as a consequence [210].	Dense point samples. Both mesh computation and analytic form fitting possible [211].
Competing equipment precision at the cost of off-line measurements [207].	Accurate measurement systems for in-line dimensional assessment [207].
Requires specific clamps for each reference model, introducing additional complexity in the management of measuring resources.	Allows the use of fixed universal setups for many different workpiece references.

The remainder of this manuscript is organized as follows: Section IV-E.2.2 reviews the relevant literature. Section IV-E.2.3 describes the developed system. Section IV-E.2.4 presents and discusses the results. Section IV-E.2.5 concludes the manuscript and introduces what remains for future work.

IV-E.2.2 Literature Review

In the automotive and aeronautic industry, dimensional inspection of manufactured parts requires high precision methods to assess the quality of the final product. Currently, CMMs are one of the

most common tools used to inspect forged workpieces due to their high precision [209]. However, CMMs are not suitable for in-line dimensional inspection of warm-die manufacturing parts due to: (1) their contact-based nature requires the workpieces to be in a cooled state to avoid damaging the measuring probe and, (2) taking measurements with the probe is highly time-consuming (even with cold workpieces). Visual computing provides contact-avoiding technologies and methodologies for Reverse Engineering and dimensional inspection which improve the productivity and efficiency of such CAD CAM CAE processes [55]. While sacrificing accuracy to some extent, optical scanners have become an alternative for dimensional inspection of different kinds of warm-die manufacturing processes [209, 215]. Table IV-E.2.1 presents a comparative of standard CMMs vs. optical scanner dimensional assessment.

IV-E.2.2.1 Off-Line Dimensional Inspection in Warm-Die Manufacturing

In warm-die manufacturing, constant monitoring of forming tools is crucial for the quality control of produced parts. Forming tools such as punches, are subjected to high structural and thermal stresses that limit their lifetime [205]. Refs. [216, 217] analyze the progressing wear of forging tools and forging defects by monitoring volume changes in the manufactured workpieces using 3D mesh reconstruction. In addition, the use of optical scanners allows the integration of numerical methods (such as Finite Element Analysis) in the dimensional inspection pipeline to quantify the thermal and structural damage of the forging tool [205, 207]. Other dimensional inspection methods in warm forming include computed tomography [215, 218], thermographic assessment [219], ultrasonic assessment [220], liquid penetrant testing [221], among others.

IV-E.2.2.2 In-Line Dimensional Inspection

All the previously presented methods only execute off-line measurements on cooled-down workpieces. Measurements directly performed on warm and hot workpieces have been rarely reported [205]. Their main shortcoming is that high temperatures affect the measurements of contact-based methods while the strong radiation affects the optical equipment, thus reducing the quality of the captured images [222]. The spectrum selective method presented in [223] filters specific wavelengths from the captured images to allow the reconstruction of the hot parts. Ref. [224] integrates a specific wavelength and power laser beam with surface fitting to measure the length and diameter of hot cylindrical workpieces. Refs. [225, 226] presents an in-line measurement system which uses two-dimensional laser range sensors (TLRS) coupled with servo motors for 3D reconstruction of hot cylindric workpieces. As an alternative to optical scanners, ref. [227] presents a vision system with two cameras that capture and process the hot workpiece without requiring any laser beams. These in-line approaches for hot dimensional assessment have been developed for workpieces with non-complex geometries (such as cylindrical workpieces).

It is worth mentioning in-line dimensional inspection approaches for cooled-down workpieces. Point cloud filtering [213] and accelerated mesh registration algorithms [214] have been developed to allow real-time inspection using 3D optical scanners and mesh reconstruction. Applications of these algorithms for in-line dimensional inspection of cooled-down workpieces include flatness inspection of rolled parts [228], inspection of large parts [229], and inspection of generic parts using geometric features [76, 211].

IV-E.2.2.3 Conclusions of the Literature Review

CMMs have been used in the warm forming industry due to their high measurement precision [209]. However, they are not suitable for in-line processes due to (1) the high temperatures of the workpiece affecting or even damaging the measuring probe and, (2) the long measurement times of the probe even for cooled-down workpieces. Other alternatives for dimensional inspection include optical scanner technologies, which do not rely on contact with the workpiece. However, the radiation due to the high temperatures can affect the data acquired by the scanners [222]. Current literature for in-line dimensional inspection of warm workpieces is very limited [205] and accounts only for very simple geometries (such as cylindrical workpieces) [223, 224, 226, 227].

Responding to the current state of the art, we present an optical system for in-line dimensional assessment of forged still-warm workpieces. Whereas previous cold-state methods would require approximately 20 min for the assessment of a single workpiece, our system spends less than 60 s per part. In-line assessment of these warm (approx. 600 °C) workpieces fills a gap in manufacturing processes in which early detection of an inherent planning, design, or manufacturing error is more important than the higher precision obtained with standard cold-state measurement methods. The system is implemented and deployed in a global automotive part maker plant, where the number of defective workpieces is reduced by a 95% with respect to previous dimensional assessment methods (i.e., cold-state CMM).

The implemented system executes 3D scanning, mesh registration and comparison (against a CAD database) of the geometry of a forged still-warm workpiece. The system is capable of in-line measurements of circular runout of revolution warm workpieces, singled out from the forming process. Contrast test against cold-state CMM measurements show that the warm-workpiece measurement is good enough for the manufacturing plant in which the system is deployed (error below 0.1 mm for parts as long as 160 mm). The temperature-vs.-runout analysis shows no correlation between these two variables at such level of uncertainty. A prior-to-operation ANOVA test with cold workpieces validates the repeatability and reproducibility (R&R) of our measurement system.

IV-E.2.3 Methodology

Given an input reference CAD model $\mathcal{C}$ and the triangular mesh $M = (X, T)$ of a scanned workpiece, the objective of the optical system is to compute key dimensional measurements on M with respect to $\mathcal{C}$. In the case of revolution workpieces, the circular runout dimension $\Delta\Phi$ measures how much a circular feature oscillates when the workpiece is rotated around the revolution (datum) axis $A = (\vec{v}, a_0)$ [210]. Such a dimension is crucial to assess the quality of the process and the produced parts in the production line.

The optical system for dimensional inspection has been designed as a process of two phases (Figure IV-E.2.1). In the first phase, the metrologist defines the parameters of the reference CAD model $\mathcal{C}$ required for the dimensional inspection of all workpieces of such reference. In the second phase, the system in-line and automatically estimates the revolution axis A and the circular runout $\Delta\Phi$ of each workpiece M. The operator in the production line is immediately provided with the results, with no intervention of the metrologist. The following sections describe the process in detail.

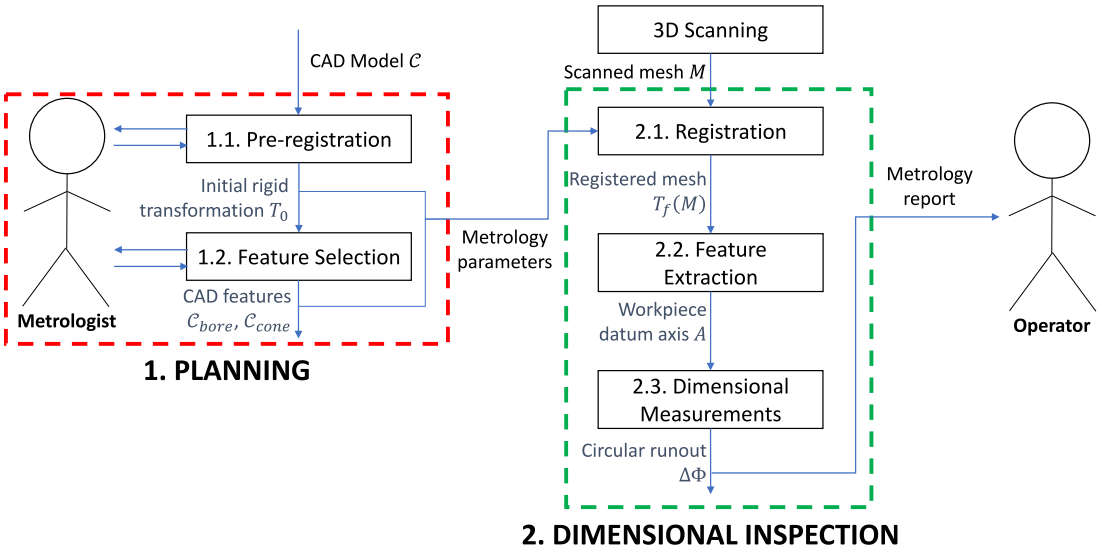


Figure IV-E.2.1: Workflow of the reverse engineering system for dimensional inspection. The system provides the dimensional inspection results to the operator directly in the production line.

IV-E.2.3.1 Planning for the Dimensional Inspection

The first phase of the optical system consists in the definition of the dimensional inspection parameters for a given reference CAD model $\mathcal{C}$. The metrologist defines the features of interest in $\mathcal{C}$, which are worth of early assessment in warm workpieces. This phase takes about 5 min, but is performed only once per CAD reference.

IV-E.2.3.1.1 Mesh Pre-Registration

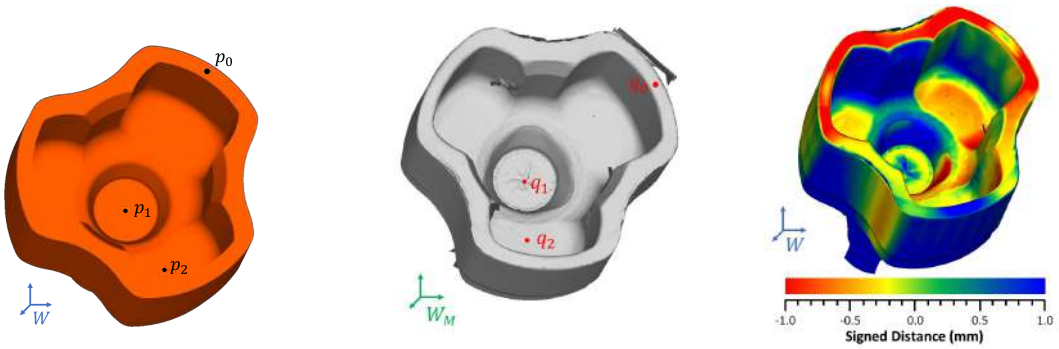
To compare the scanned mesh M with the reference CAD model $\mathcal{C}$, it is crucial that both of these surface representations share the same coordinate system $W = \{\vec{w_x}, \vec{w_y}, \vec{w_z}; \vec{p_w}\}$. If W and W_M are the coordinate systems of $\mathcal{C}$ and M, respectively, the objective is to compute a rigid transformation $T_0 \in SE(3)$ such that $T_0(W_M) \approx W$.

To compute T_0, the developed system uses an alignment-of-correspondences algorithm [230]. Let $\{p_0, p_1, p_2\} \subset \mathcal{C}$ and $\{q_0, q_1, q_2\} \subset M$ be three non-collinear points sampled from the reference CAD and the workpiece mesh, respectively. The alignment-of-correspondences algorithm computes the rigid transformation T_0 that minimizes the distance between the two sets of points:

$$T_0 = \arg\min \sum_{i=0}^{2} \|p_i - T_0(q_i)\| \qquad s.t. \quad T_0 \in SE(3) \tag{IV-E.2.1}$$

where $SE(3) = SO(3) \times \mathbb{R}^3$ is the special Euclidean group (group of all rigid transformations in $\mathbb{R}^3$).

In Equation (IV-E.2.1), p_i, q_i are corresponding points in the CAD and the mesh, respectively. These points are interactively selected by the metrologist as illustrated in Figure IV-E.2.2. This pre-registration is performed only once per CAD reference $\mathcal{C}$.



(a) (b) (c)

Figure IV-E.2.2: User-assisted alignment of correspondences [230]. The metrologist selects 3 corresponding points in both the CAD (orange) and a scanned mesh (gray). (**a**) CAD reference and its coordinate system W; (**b**) Scanned mesh and its coordinate system W_M; (**c**) Alignment of corresponding points [230].

IV-E.2.3.1.2 Feature Selection

As mentioned starting Section IV-E.2.3, the metrologist interactively selects the different CAD features (FACEs) from $\mathcal{C}$ associated with the workpiece revolution axis $A = (\vec{v}, a_0)$ (Figure IV-E.2.3a). In this case, the metrologist selects the CAD FACEs $\mathcal{C}_{bore}$ (blue) of the cylindric surface which dictate the rotation of the workpiece. The axis vector of $\mathcal{C}_{bore}$ defines the theoretical revolution axis vector $\vec{v}$ (green).

On the other hand, the metrologist must define the axis point a_0 as a reference point. In the context of stub axle forming, the point is computed as follows:

1. The metrologist selects the CAD FACEs $\mathcal{C}_{cone}$ (red) corresponding to a conical surface at the bottom of the punch zone of the workpiece (Figure IV-E.2.3a).

2. The metrologist defines the datum diameter $d > 0$. In this case, the metrologist defines d as the diameter of the supporting fixture for the machining of the workpiece (after it has been formed).

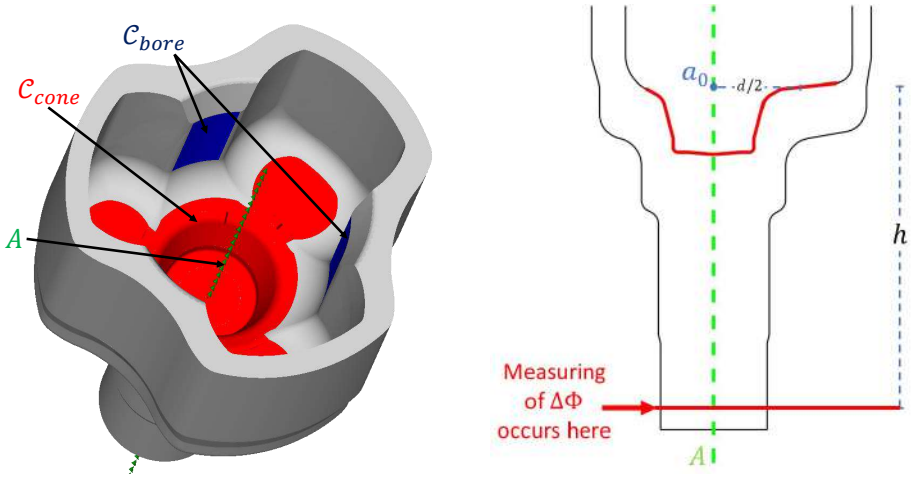
3. The point at the revolution axis A where the surface $\mathcal{C}_{cone}$ attains the diameter d is the theoretical axis point a_0. The diameter d is measured perpendicular to the axis A (see Figure IV-E.2.3b). The point a_0 is a reference point for machining operations (after the piece has been formed).

The features $\mathcal{C}_{bore}$ and $\mathcal{C}_{cone}$ (Figure IV-E.2.3a) have been chosen for the definition of the reference axis A due to two main reasons:

1. $\mathcal{C}_{bore}$ is the part of the punch that suffers less wearing since the direction of the compression load during the forging process is parallel to its axis. This fact makes this geometry more stable from a dimensional assessment perspective.

2. The surfaces $\mathcal{C}_{bore}$ and $\mathcal{C}_{cone}$ are the same surfaces used to hold the workpiece during the posterior machining process. In this way, the algorithm uses the same coordinate system that will be used in the next step of the manufacturing process. In addition, it can be said that any possible registration error induced by the tool wearing is not relevant given that the subsequent machining process will use the same defective geometries to establish its reference frame.

The runout height $h > 0$ is the distance from a_0 along the axis A (Figure IV-E.2.3b) where the circular runout is measured. This height h is manually defined by the metrologist.



(a) (b)

Figure IV-E.2.3: Cylindrical $\mathcal{C}_{bore}$ (blue) and conical $\mathcal{C}_{cone}$ (red) features on the CAD reference used to compute the revolution axis A (green). (**a**) CAD features $\mathcal{C}_{bore}$, $\mathcal{C}_{cone}$ and A; (**b**) Reference axis point a_0 defined where $\mathcal{C}_{cone}$ achieves an specific diameter d.

IV-E.2.3.2 In-Line Dimensional Inspection

After the planning has been carried for a given reference $\mathcal{C}$, the automatic inspection for every workpiece M related to that reference is automatically performed. The following sections detail the 3D scanning of the warm workpiece, the registration of the mesh regarding the reference CAD $\mathcal{C}$, the computation of the revolution axis A (datum) on M based on the CAD features, and finally the calculation of the circular runout $\Delta\Phi$ of the workpiece.

IV-E.2.3.2.1 3D Scanning System

Figure IV-E.2.4 presents the setup for the 3D scan of the warm workpiece.

Laser triangulation is used to reconstruct the surface. Two independent laser line projectors impact the workpiece inner (punch zone) and outer (forming matrix zone) sides, respectively. Since

the surface emits light in the red spectrum due to the high temperatures of the workpiece, the two lasers are chosen to work in the blue spectrum.

The first laser is placed above the workpiece, with an elevation near 45 degrees with respect to the plane that supports the workpiece. This laser allows to scan the inner (punch zone) surface of the axle. This laser is observed by 2 cameras since the geometry of such surface is self-occluding. The cameras must be as close to the workpiece as the heat emitted by it permits (approx. 500 mm), in accordance to the operating temperature prescribed for them.

The second laser is positioned underneath the plane supporting the workpiece (elevation near -45 degrees). The external surface (forming matrix zone) of the workpiece is scanned from below, through slots of the supporting table. The laser projection on such surface is captured by a single camera since no occlusions occur.

Since the laser projections do not lie on a plane parallel to their respective camera plane, Scheimpflug adapters [231] are incorporated in all the cameras to fix their plane of focus. In addition, each camera has an interference filter which allows it to only see light near the laser wavelength ($450\,\text{nm} \pm 25\,\text{nm}$, see Table IV-E.2.2).

A 3-grip system holds the workpiece from the shaft. The grip system is made of steel with ceramic coating to stand the high temperatures. The 3-grip system is mounted on a rotating disk such that the workpiece is rotated 360 degrees around its revolution axis during the scanning. The cameras capture a static image of the workpiece at each pulse of the encoder. Thus, the reconstruction is performed with 360 images per camera.

Table IV-E.2.2: Properties of the laser line projectors used to irradiate the warm workpiece surface.

Property	**Value**
Power	20 mW
Wavelength	450 nm

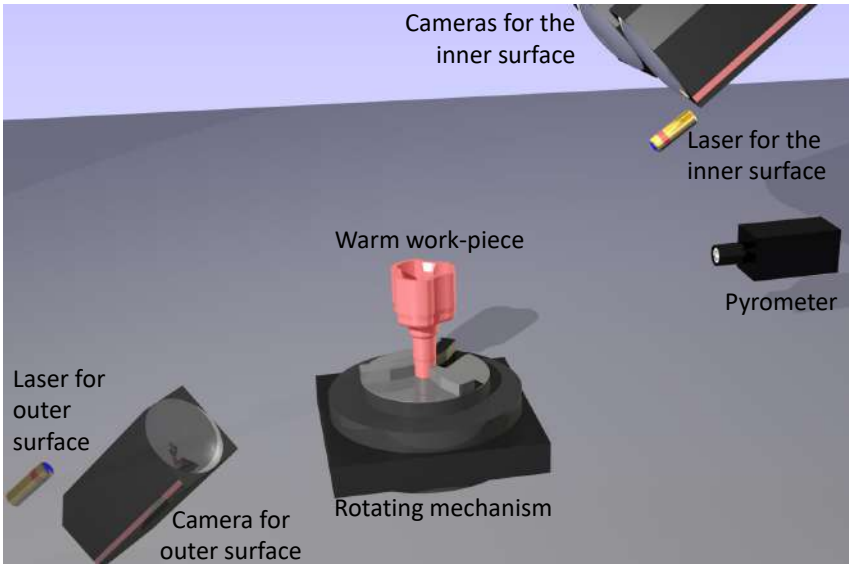


Figure IV-E.2.4: Setup of the 3D optical system designed to digitize the warm workpiece.

The 3D mesh reconstruction from the acquired images is executed with HALCON [232]. The acquisition and reconstruction of the warm workpiece takes about 5 s (average). Finally, a pyrometer is used to track the average temperature of the workpiece during the 3D reconstruction.

The full setup has been installed on a foam cushion layer to isolate the sensors form the vibrations induced by the forging presses.

IV-E.2.3.2.2 Device Calibration

The calibration of the scanner involves the characterization of the laser triangulators and their relative positions regarding the axis of the rotating disk. The scanner has a total of two triangulators, with one and two cameras respectively (see Figure IV-E.2.4). Assuming that the reference system of the machine is in the center of the rotating disk, the calibration involves the estimation of the three camera poses.

Both laser projectors have been mechanically positioned and aligned so that their intersection coincides with the axis of the rotating disk. During the construction of the scanner this alignment is verified with a gauge specifically designed for this task, and it does not need further adjustments. This alignment ensures that the points reconstructed by the cameras belong to the plane XZ of the reference system, which is defined by the rotation axis and the two laser lines.

The pose of the cameras is estimated using a calibration object with a hollow revolution geometry, as shown in Figure IV-E.2.5. Such a calibration object has been measured using a CMM by a metrology laboratory certified by an ENAC (National Accreditation Entity).

During the calibration process the object rotates around the axis of the turn table while the cameras observe the projections of the lasers on its surface. Each camera captures an image for each pulse of the encoder, which has been set to 360 pulses per complete revolution. From these images the intersection of the different segments (laser projections) are obtained.

Since the geometry of the calibration object is known, it is possible to establish 3D-2D point matches that relate points from a common reference frame with their corresponding observations in the camera images. In this way, it is possible to obtain the pose of the three cameras solving the homography matrices induced by the sets of correspondences [233].

Experimentally, it has been found that the residual error of the homography evaluated in the intersection points after the calibration is under 0.01 mm. Figure IV-E.2.6 shows some dimensions measured in the calibration object. As it can be deduced from the results, the uncertainty of the scanner is better (less deviation) in the central area of the scanning volume. This effect can be attributed to the fact that the images of the cameras have better focus quality in this area, even after adding the Scheimpflug adapters.

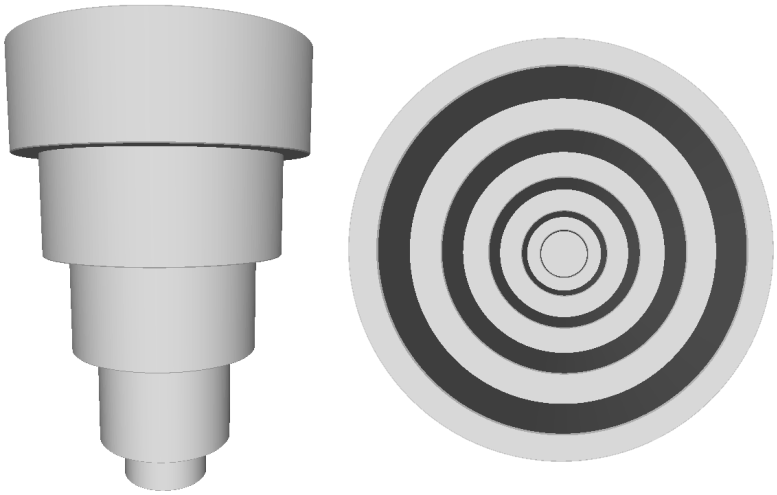


Figure IV-E.2.5: Front and top views of the calibration object geometry.

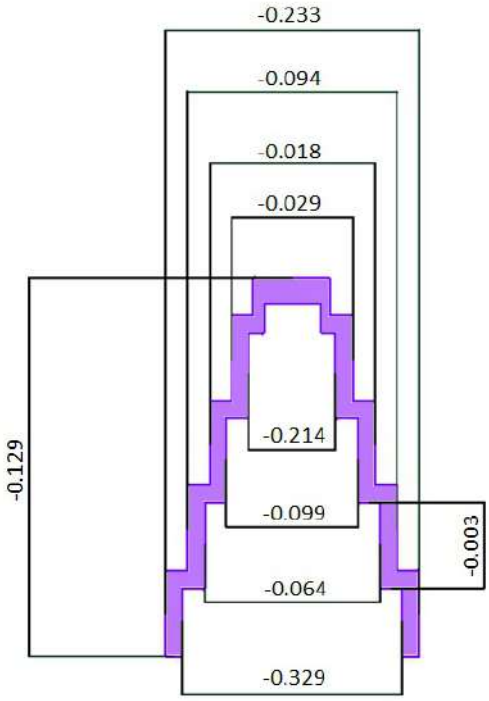


Figure IV-E.2.6: Deviations (mm) after calibration in a cross-section of the scanned pattern (purple), through the XZ plane.

IV-E.2.3.2.3 Mesh Registration

It is imperative for our system that the scanned triangular mesh M is represented in the same coordinate system W of the CAD reference $\mathcal{C}$. The mesh registration process computes the rigid transformation $T_f \in SE(3)$ that maps the mesh coordinate system W_M to the global coordinate system W i.e., $T_f(W_M) = W$. The rigid transformation T_f is computed by minimizing the distance between the transformed mesh and the reference CAD model:

$$\begin{aligned} T_f = \arg\min \sum_{x_i \in M} & d(T_f(x_i), \mathcal{C}) \\ s.t. \qquad & T_f \in SE(3) \end{aligned} \tag{IV-E.2.2}$$

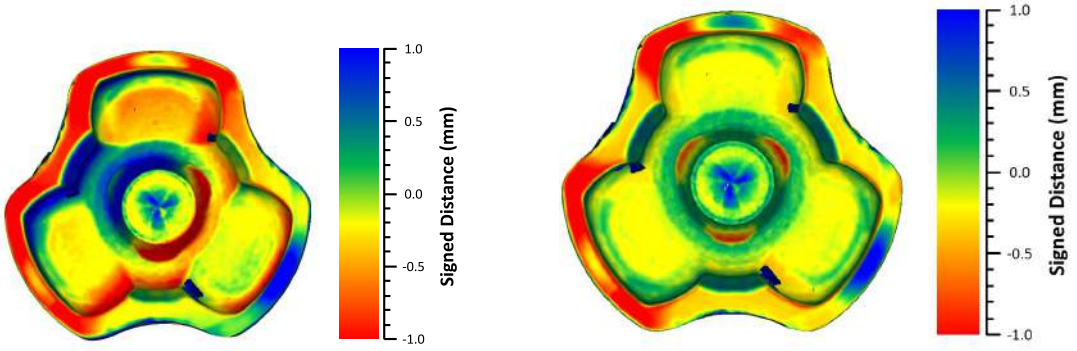
where $d(T_f(x_i), \mathcal{C})$ is the closest distance from the point $T_f(x_i)$ to the reference $\mathcal{C}$. To solve the minimization problem in Equation (IV-E.2.2), the Iterative Closest Point (ICP) algorithm is implemented [234]. The ICP algorithm, transforms the previously defined minimization problem into the following equivalent one:

$$\begin{aligned} T_{icp} = \arg\min \sum_{x_i \in M} & d((T_{icp} \circ T_0)(x_i), \mathcal{C}) \\ s.t. \qquad & T_{icp} \in SE(3) \end{aligned} \tag{IV-E.2.3}$$

where $T_f = T_{icp} \circ T_0$. To avoid local minima, the ICP algorithm requires an initial solution T_0 such that $T_0(W_0)$ is close to the optimal solution W. T_0 has been previously computed in the planning step (Equation (IV-E.2.1)).

In optical-based dimensional inspection, selection of reference geometries for mesh registration is crucial for adequate estimation of datums and measurements [211, 212]. Therefore, the registration of the scanned mesh M is performed using only the punch zone of the workpiece, which characterizes the revolution axis A. Figure IV-E.2.7 plots the results of the registration process. The colormap shows the distance from the scanned mesh M to the reference CAD $\mathcal{C}$, with green zones indicating closeness between the models ($d(x_i, \mathcal{C}) \approx 0$ mm), and red and blue zones indicating remoteness ($d(x_i, \mathcal{C}) > 0.5$ mm)

To save computational expenses, the distance $d(T_f(x_i), \mathcal{C})$ is computed by previously meshing the CAD reference. This is done because we have observed from our experiments that computing point-to-CAD distance is more time-consuming than point-to-mesh distance.



(a) (b)

Figure IV-E.2.7: Mesh registration results. The colormap shows the signed distance from the scanned mesh M to the CAD model $\mathcal{C}$. (**a**) Initial guess from np-align (Figure IV-E.2.2c); (**b**) Iterative Closest Point (ICP) mesh registration.

IV-E.2.3.2.4 Feature Extraction

After the mesh registration of M has been computed, the system proceeds to extract the mesh features required for the dimensional assessment. To compute the revolution axis, we need to first extract the sub-mesh $M_{bore} \subset M$ which corresponds to the cylindric surface that dictates the rotation of the workpiece. Such cylindric surface has been already identified by the metrologist in the CAD model during the planning phase (Figure IV-E.2.3a). M_{bore} is computed by extracting the mesh points close to the corresponding CAD feature $\mathcal{C}_{bore}$:

$$M_{bore} = \left\{ x \in M \quad | \quad d(x, \mathcal{C}_{bore}) < \varepsilon \quad \wedge \quad \cos^{-1}(\vec{n}(x) \cdot \vec{u}(x)) < \theta \right\} \tag{IV-E.2.4}$$

where $\varepsilon > 0$, $0° \leq \theta \leq 180°$ are a threshold distance (mm) and a threshold angle (degrees), respectively, $\vec{n}(x)$ is the vector normal to the surface at $x \in M$, and $\vec{u}(x)$ is a vector pointing to the theoretical revolution axis $A = (\vec{v}, c_0)$, defined as follows:

$$\vec{u}(x) = ((x - c_0) \cdot \vec{v})\vec{v} - (x - c_0) \tag{IV-E.2.5}$$

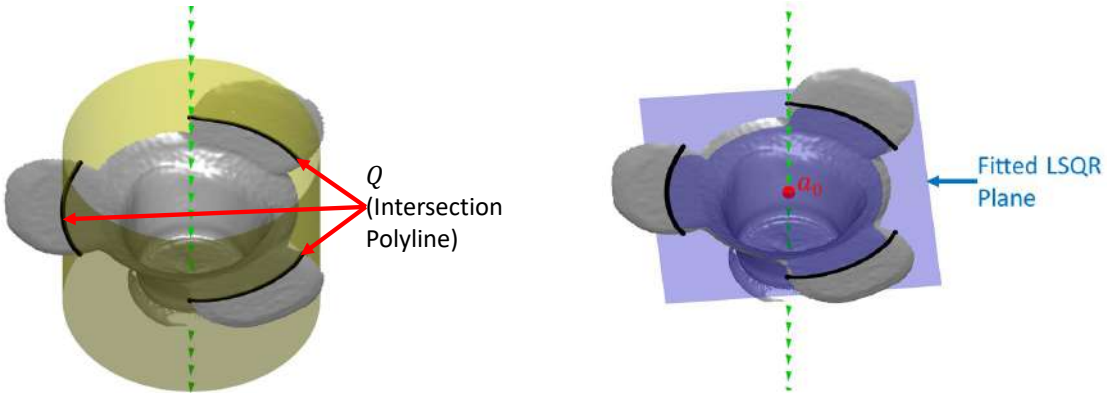
The term $\cos^{-1}(\vec{n}(x) \cdot \vec{u}(x))$ is introduced in Equation (IV-E.2.4) to filter mesh noise and improve the estimation of the revolution axis on the scanned mesh. From our experiments, we have found that the threshold values $\varepsilon = 0.5\,\text{mm}$ and $\theta = 10°$ produce good results, considering the thermal contraction of the workpiece, mesh noise, etc.

The vector $\vec{v}$ is computed by fitting a cylinder to the mesh M_{bore}. The RANSAC algorithm from the Point Cloud Library (PCL) [235] is used to perform the surface fitting.

An approach similar to the previous one is used to calculate the reference axis point a_0. The cone surface feature $M_{cone} \subset M$ is computed by extracting the mesh points close to the corresponding CAD feature $\mathcal{C}_{cone}$:

$$M_{cone} = \{ x \in M \quad | \quad d(x, \mathcal{C}_{cone}) < \varepsilon \} \tag{IV-E.2.6}$$

We have found in our experiments that fitting a cone surface to M_{cone} produces highly unstable results. Instead of fitting the analytical surface, the developed system computes a cylinder $S_{\vec{v},d/2}$ with axis vector $\vec{v}$ and cylinder radius $d/2$. This cylinder is then intersected with M_{cone}, which produces a polyline Q (Figure IV-E.2.8a). Finally, the intersection between $\vec{v}$ and the plane that contains the polyline Q (in a least-squares sense), is the point a_0 (Figure IV-E.2.8b).



(a) (b)

Figure IV-E.2.8: Computation of the reference point a_0 on $\mathcal{C}_{cone}$. (**a**) Mesh-cylinder intersection; (**b**) Plane-axis intersection.

IV-E.2.3.2.5 Dimensional Measurements

After the workpiece revolution axis A has been calculated, the circular runout can be computed on M. Given a circular feature $P \subset M$, perpendicular to the revolution axis A (i.e., $P \perp A$), the circular runout of P with respect to A measures how much the feature P oscillates when the workpiece is rotated around the axis A [210].

The following steps describe the system's approach to compute the circular runout of the workpiece:

1. Compute the plane T_h with normal vector $\vec{v}$ and pivot point $a_0 + h\vec{v}$ (the parameter h has been defined already by the metrologist in the planning step, Section IV-E.2.3.1.2). See Figure IV-E.2.9a,b.

2. Compute the circular feature P defined as:

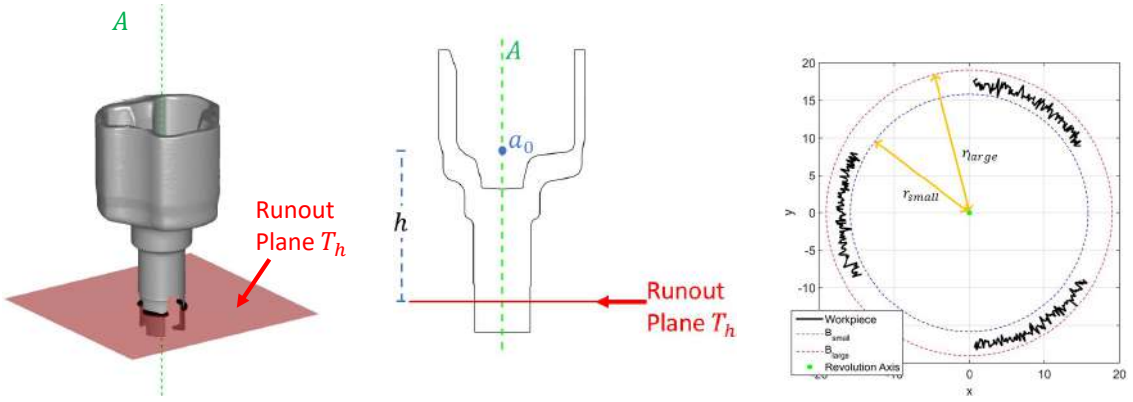
$$P = M \cap T_h \tag{IV-E.2.7}$$

3. Filter outliers by removing all points in P whose distance to the theoretical section is greater than a given threshold.

4. Compute the inscribed circle B_{small} and circumscribed circle B_{large} of P with center A and respective radii r_{small}, r_{large} (Figure IV-E.2.9c). $0 < r_{small} \leq r_{large}$.

5. Compute the circular runout $\Delta\Phi$ defined as [210]:

$$\Delta\Phi = r_{large} - r_{small} \tag{IV-E.2.8}$$



(a) (b) (c)

Figure IV-E.2.9: Calculation of the workpiece runout $\Delta\Phi$ regarding the revolution axis A. $\Delta\Phi = r_{large} - r_{small}$. (**a**) Runout plane T_h perpendicular to the revolution axis A; (**b**) The runout plane T_h is defined at a height h from the reference point a_0; (**c**) r_{small} and r_{large} radii computed at the plane T_h.

Before the computation of the circular runout, our system performs circle fitting on the feature P using RANSAC. Such a fitting improves the robustness of the runout estimation by filtering noise and outliers from the scanned mesh.

It is worth noting that the runout deviation includes the eccentricity of both axes and roundness deviations of the measured circle. This is an expected behavior following the standards for geometrical dimensioning and tolerancing [210].

IV-E.2.4 Results

Section IV-E.2.4.1 presents and discusses the application of the system to assess the runout of a scanned workpiece at different temperatures. Section IV-E.2.4.2 shows the results of prior-to-operation testing the system using an ANOVA R&R test on cold-state workpieces. Section IV-E.2.4.3 discusses the deployment of the developed system in the automotive manufacturing plant. Finally, Section IV-E.2.4.4 discusses the application of the system in the context of Visual Computing and Industry 4.0 technologies.

IV-E.2.4.1 Warm-Workpiece Measurements

In the manufacturing line, each workpiece leaves the forming press at nearly 800 °C. Each part is then left to be cooled naturally by air convection, which takes around 60 min. For this section, two different workpieces are measured continuously during the cool-down. The height of each workpiece

is 161.63 mm, and the runout height h (from the axis point a_0 to the runout plane—see Figure IV-E.2.9b) is 64 mm. The objective of this test is to evaluate the accuracy of the system (agreement with the CMM result). Figure IV-E.2.10 plots the runout measurements at different temperatures for the two different workpieces. The workpieces have been measured in the temperature range 27 °C $\leq$ T $\leq$ 560 °C. Each workpiece runout also has been measured with a CMM after cool-down (27 °C). The CMM value is used as ground-truth for assessment purposes. The CMM value obtained for the first workpiece is 0.66 mm. Figure IV-E.2.10a shows that our system measurements deviate in less than 0.1 mm from the CMM measurement. The CMM value obtained for the second workpiece is 0.8 mm. Similar to the first workpiece, our system measurements deviate in less than 0.1 mm from the CMM measurement (Figure IV-E.2.10b). It is worth noting that this deviation is dependent on the height h, increasing as longer workpieces are measured (and decreasing for shorter ones). Such a deviation (vs. height) is small enough for the dimensional assessment purposes of the forging process in which the system is deployed.

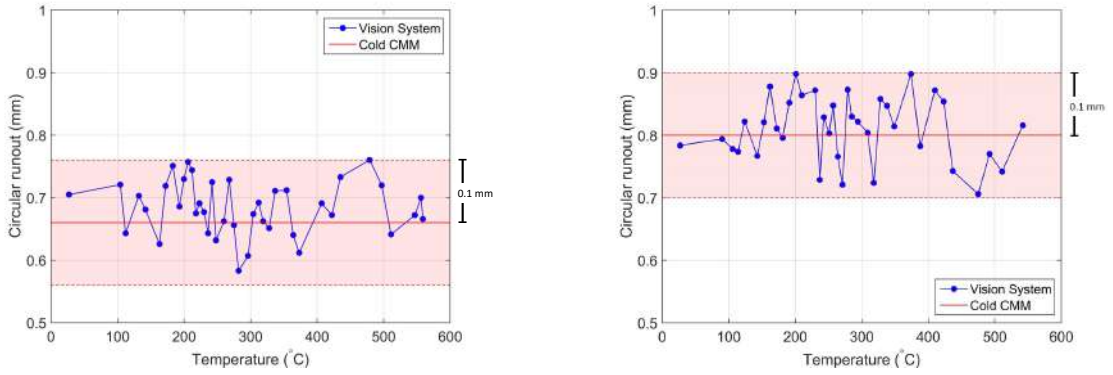


Figure IV-E.2.10: Measurements of 2 warm workpieces until cool-down (27 °C$\leq$ T $\leq$ 560 °C). Results of our measurement system (blue line) do not deviate more than 0.1 mm from Coordinate Measurement Machine (CMM) measurements (red line). (**a**) Workpiece 1. CMM runout = 0.66 mm; (**b**) Workpiece 2. CMM runout = 0.8 mm.

In Figure IV-E.2.10, there is no apparent correlation between the runout and the temperature of the workpiece at this scale of uncertainty (0.1 mm). Consequently, assessment of the workpiece runout can be performed in our system without the necessity of a correction due to thermal contraction. A more robust study on this matter for this measurement (and other dimensional measurements on the workpiece) is out of the scope of this manuscript, and it is left for future work.

IV-E.2.4.2 ANOVA Gauge Repeatability and Reproducibility (R&R) Test

To assess the robustness of the implemented system, a prior-to-operation ANOVA Gauge R&R test is executed. The ANOVA test is performed with cold-state workpieces, which is outside of normal

operations. The purpose of the test is to assess the precision (repeatability and reproducibility) of the system but not its accuracy (agreement with the real result). For the control testing, $a = 3$ different workpieces (Figure IV-E.2.11) are measured $m = 10$ times by $b = 3$ different operators, resulting in a sample of 90 measurements. Table IV-E.2.3 presents the measurement results of each experiment.

Figure IV-E.2.11: Cold workpieces used to run the ANOVA R&R test. The three workpieces share the same CAD reference.

Table IV-E.2.3: Runout results (mm) of our system for 3 different cold workpieces (Figure IV-E.2.11), measured 10 times by 3 different operators.

	Workpiece 1			**Workpiece 2**			**Workpiece 3**		
	Op 1	**Op. 2**	**Op. 3**	**Op. 1**	**Op. 2**	**Op. 3**	**Op. 1**	**Op. 2**	**Op. 3**
Msh 1	0.71	0.60	0.77	0.78	0.82	0.74	0.79	0.84	0.89
Msh 2	0.63	0.71	0.72	0.79	0.85	0.87	0.77	0.81	0.81
Msh 3	0.64	0.61	0.65	0.81	0.79	0.82	0.80	0.79	0.83
Msh 4	0.70	0.65	0.56	0.79	0.80	0.80	0.77	0.80	0.77
Msh 5	0.65	0.74	0.65	0.79	0.83	0.74	0.79	0.86	0.78
Msh 6	0.69	0.68	0.60	0.84	0.84	0.83	0.82	0.76	0.85
Msh 7	0.68	0.67	0.67	0.85	0.80	0.82	0.84	0.77	0.80
Msh 8	0.70	0.74	0.72	0.75	0.80	0.81	0.76	0.78	0.76
Msh 9	0.61	0.67	0.65	0.81	0.78	0.74	0.82	0.81	0.78
Msh 10	0.72	0.64	0.58	0.75	0.77	0.81	0.84	0.79	0.79
Mean	0.67	0.67	0.66	0.80	0.81	0.80	0.80	0.80	0.80
Std. Dev.	0.03	0.04	0.06	0.03	0.03	0.04	0.03	0.03	0.04
Max-Min	0.11	0.14	0.20	0.10	0.09	0.13	0.08	0.10	0.12

Table IV-E.2.4 shows the ANOVA results for the conducted experiments. The degrees of freedom

(DOG) for each sum of squares are defined as:

$$\begin{aligned} \mathrm{DOG}_{\mathrm{operator}} &= b - 1 = 2 \\ \mathrm{DOG}_{\mathrm{workpiece}} &= a - 1 = 2 \\ \mathrm{DOG}_{\mathrm{interaction}} &= (a-1) * (b-1) = 4 \\ \mathrm{DOG}_{\mathrm{vision\_system}} &= ab * (m-1) = 81 \end{aligned} \tag{IV-E.2.9}$$

The test shows that the variable Workpiece is statistically significant for the computed runout (p-value ¡ 0.05). This means that our system can difference each workpiece from the others due to the inherent manufacturing process bias. On the other hand, the operator and the interaction (operator/workpiece) are not statistically significant (p-value ¿ 0.05), which means that the operator is not a significant source of error for our measurement system (reproducibility).

Table IV-E.2.4: Analysis of Variance (ANOVA) table.

Source of Variability	**Degrees of Freedom**	**Sum of Squares**	**Mean Square**	F **Statistic**	p**-Value**
Operator	2	0.0010	0.0005	1.2282	0.3838
Workpiece	2	0.3575	0.1787	438.5964	0.0000
Interaction	4	0.0016	0.0004	0.2419	0.9137
Vision System	81	0.1364	0.0017		
Total	89	0.4966			

Table IV-E.2.5 presents the Gauge Repeatability & Reproducibility (GRR) [236] results for the executed experiments. The variations induced by the different operators account only for the 0.04% of the total variance of the data, which is an indicator of the reproducibility (less variation equals to more reproducibility) of the developed system. Similarly, the variation introduced by the vision system accounts for the 22.07% of the total variance of the data, which is the indicator of the repeatability (less variation equals to more repeatability) of the system. The manufacturing process accounts for the rest of the variance (77.89%). The Gauge R&R (repeatability + reproducibility) accounts for the 22.11% of the total variance. Such Gauge Repeatability and Reproducibility (GRR) variation is acceptable for the warm forge plant in which the system is deployed (GRR ¡ 25%), where the early detection of dimensional misfits in warm workpieces compensates for the precision loss of the measuring system.

Table IV-E.2.5: Gauge Repeatability and Reproducibility (GRR) table.

Source	**Variance**	**% of Total Variance**
Operators (Reproducibility)	0.0000	0.04%
Vision System (Repeatability)	0.0017	22.07%
Gauge R&R (GRR)	0.0017	22.11%
Interaction	0.0000	0.00%
Workpieces	0.0059	77.89%
Total	0.0076	100.00%

Figure IV-E.2.12 presents a boxplot of the data collected in Table IV-E.2.3. Each box represents the 10 measurements taken by an operator. The central marker shows the median of the samples, the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The whiskers are the maximum and minimum obtained runout values.

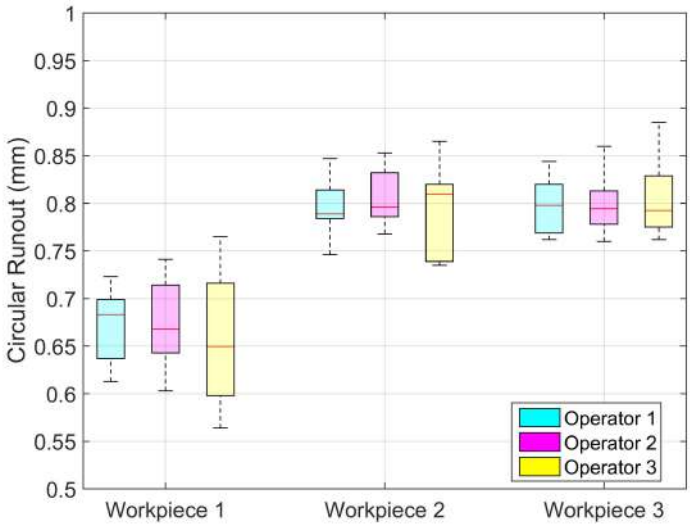


Figure IV-E.2.12: Boxplot of the runout measurements presented in Table IV-E.2.3.

As previously discussed, the manufacturing process induces most of the variance in the form of workpieces with three different runouts (0.67 mm, 0.8 mm and 0.8 mm, respectively). On the other hand, the variance induced by each operator and our measuring system is less significant (largest standard deviation—0.06 mm—in workpiece 1, operator 3).

IV-E.2.4.3 Deployment

The metrology system has been deployed in the shop floor of an automotive warm forge of motorcar stub axles, next to one of the press lines. The press line forges around 1200 workpieces per hour. As shown in Figure IV-E.2.13, in this initial setup an operator places manually the warm workpiece into the scanner directly from the press ramp. In the future this operation will be assumed by a robot arm to obtain higher measuring cadences.

To avoid damages to the optical equipment, all the devices are protected with a metal cover. The chassis has been mounted on a foam cushion layer to absorb the vibrations caused by the press.

The measurement software is connected to the Manufacturing Execution System (MES) of the factory. This integration allows automatic loading of the measurement program for the reference that is being manufactured, as well as other data, such as the manufacturing order, the operator name, etc. This data is stored in the report for each workpiece along with the calculated measurements for traceability purposes.

The implemented system automatically assesses the runout of the revolution-like workpieces in less than 60 s. Current measurements are performed with the warm workpiece at approximately

600 °C. The system works as an early detection mechanism for manufacturing and process failures, mainly due to deviations between the forging punch axis and the forming matrix axis. The previous dimensional assessment mechanism (cold-state CMM) used to take only one measurement for every 400 fabricated parts (i.e., every 20 min). On the contrary, our system performs a measurement every 20 fabricated parts (i.e., every minute). Therefore, the system improves the previous measurement method by reducing the number of defective workpieces by around an estimated 95%.

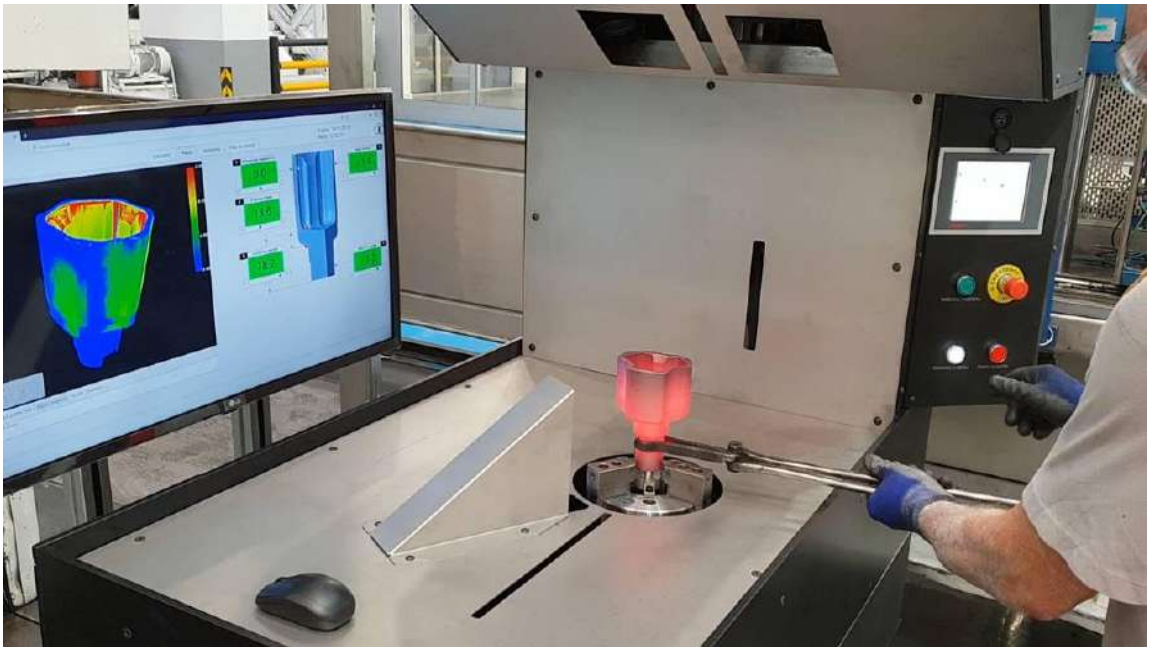


Figure IV-E.2.13: Deployment of the optical system into the production line of warm forming of motorcar stub axles. The temperature of the workpiece is approximately $600°C$.

IV-E.2.4.4 Industry 4.0 and Visual Computing

The use of optical systems and visual computing technologies has become an important factor for the improvement of recently developed and classic manufacturing processes [55]. In the context of Industry 4.0, the deployed system improves the classic approaches for dimensional assessment in the warm-die forge industry as follows:

1. Our system performs measurements directly on warm workpieces. Such approach changes the classic scheme for dimensional assessment, which demands workpieces in cold state, limiting in-line metrology application in the warm-die and hot-die forge industry.

2. Thanks to Visual Computing and Industry 4.0 technologies, the developed system can perform fast dimensional measurements on warm workpieces. Over standard cold-state measurement methods, our measurement system reduces the time required to process a warm part by a factor of 95% (from 20 min to 1 min per part).

3. As already mentioned, the Visual Computing technologies provide a framework that allows deployment of the measurement system directly in the manufacturing line. Consequently, the efficiency of the process and product control highly increases as measurements and lines of action can be performed in-line.

4. The deployed system results are shown in a display using a web report tool with visual feedback about the dimensional quality of the measured workpiece. Thanks to the use of such web technologies [76], the report becomes available in real time to any computer of the factory and any member of the manufacturing plant, including operators, metrologists, and engineers.

5. The visual feedback provided by the visual computing techniques allows easier understanding and more intuitive dimensional assessment of scanned workpieces [211], in contrast to standard CMM numerical data.

6. The automation of the process, together with the high cadence of data acquisition and the aid of web reporting tools, enable a global perspective of the manufacturing process in the context of data analytics. However, such approach is out of scope of the current manuscript, and it is left for future work.

IV-E.2.5 Conclusions

This manuscript presents the implementation and deployment of an optical system for automatic in-line dimensional inspection of revolution warm workpieces. The circular runout of warm-forged revolution workpieces is critical as a severe misalignment between the punch press and the forming matrix axes disables the posterior machining, resulting in a scrapped part. The system splits the inspection in two steps: (1) the dimensional assessment planning, performed only once by the metrologist, off-line the production, and (2) the in-line automated dimensional inspection. The developed system automatically assesses, in less than 60 s, the circular runout of the workpiece, whose temperature nears 600 °C. Our prior-to-operation test results show that the measurements of the developed system for warm workpieces (27 °C $\leq$ T $\leq$ 560 °C) deviate less than 0.1 with respect to the standard CMM measurements of the cooled-down workpieces, for workpieces as long as 160 mm. In addition, the temperature-vs.-runout analysis shows no correlation between these two variables at such level of uncertainty. The measuring system repeatability and reproducibility (R&R) has been validated with an ANOVA test. This assessment of dimensions in warm workpieces fills a gap in processes in which the advantage of early detection of an inherent planning, design, or manufacturing error compensates for the disadvantage of precision loss due to the cooling of the workpiece.

Our system has been deployed by an automotive part manufacturer in a warm forming production line of stub axles, working as early detection of dimensional misfits. This early detection reduces the time needed to detect a defective part from 20 min to 1 min. Since the forge is a highly repetitive manufacturing process, when a defective part is found, all the pieces between the last correct part and the defective one are systematically scrapped. Thus, considering the production cadence, the number of parts that are scrapped each time a defective part is detected has been reduced from 400 to 20 (95%).

Future work concerns: (1) A warm-workpiece assessment method for data which are highly sensitive to cooling effects, which accounts for the heat loss effects on the workpiece geometry. (2) The

integration of a robot arm for automatic placement of forged workpieces, to increase the measurement efficiency. (3) Metrological certification of the equipment by an ENAC accredited laboratory.

Author Contributions: D.M.-P. and J.R.S. conceptualized and implemented the Computational Geometry and Dimensional Inspection methodology. M.A. and A.I. designed, implemented and executed the system calibration and the Computer Vision system for data acquisition. E.G. and J.P. designed the methodology and experiments, and supervised the the Dimensional Inspection of warm-forge parts. O.R.-S. and J.P. supervised the Stochastic Computational Geometry aspects and applications of this research. All the authors contributed in the writing of this manuscript.
Funding: This research was partially supported by the Basque Government under the grant Basque Industry 4.0.
Acknowledgments: We want to thank our colleagues from Sariki Metrología S.A. whose knowledge of dimensional inspection has been very valuable during this research.
Conflicts of Interest: The authors declare no conflict of interest.

IV-E.2.6 Abbreviations

The following abbreviations are used in this manuscript:

CMM	Coordinate Measurement Machine.
ANOVA	ANalysis Of VAriance.
R&R	Repeatability and Reproducibility.
FACE	A connected region on a parametric surface in $\mathbb{R}^3$.
$\mathcal{C}$	Boundary representation (CAD model) of the reference geometry. $\mathcal{C} \subset \mathbb{R}^3$ is a 2-manifold, represented as set of BODY, LUMPSs, FACEs, LOOPs, EDGEs, and VERTICES.
M	Triangular mesh $M = (X, T)$ of the scanned workpiece. $X = \{x_0, x_1, \ldots, n\}$ and $T = \{t_0, t_1, \ldots, n\}$ are the points (geometry) and triangles (topology) of the mesh, respectively. $M \subset \mathbb{R}^3$ is a 2-manifold.
A	Revolution (datum) axis $A = (\vec{v}, a_0)$ of the workpiece M. The vector $\vec{v} \in \mathbb{R}^3$ defines the direction of the axis and $a_0 \in \mathbb{R}^3$ is a point lying on the axis.
$\Delta\Phi$	Circular runout (mm) of the scanned workpiece M. $\Delta\Phi \geq 0$ measures how much a cylindrical feature oscillates when rotated around the revolution axis A.
h	Height $h > 0$ (mm) where the circular runout $\Delta\Phi$ is measured in the workpiece. This height is measured from a_0, in the direction of $\vec{v}$.
W	Reference coordinate system $W = \{w_x, w_y, w_z; p_w\}$. W is the coordinate system of $\mathcal{C}$ and the coordinate system of M after mesh registration.
W_M	Coordinate system of M before mesh registration.
$SE(3)$	Special Euclidean group. Group of all rigid transformations in $\mathbb{R}^3$. $SE(3)$ is composed by all the possible rotation matrices and all possible translations in $\mathbb{R}^3$, i.e., $SE(3) = SO(3) \times \mathbb{R}^3$.
T_0	Initial rigid transformation $T_0 \in SE(3)$ that approximately maps W_M to W.
T_{icp}	Rigid transformation $T_{icp} \in SE(3)$ that maps $T_0(W_M)$ to W. T_{icp} is the result of registering the mesh M to the reference $\mathcal{C}$.
T_f	Rigid transformation $T_f \in SE(3)$ that maps W_M to W. $T_f = T_{icp} \circ T_0$.
M_{bore}	Cylindrical surface $M_{cylinder} \subset M$ whose axis vector is the revolution axis vector of the workpiece $\vec{v}$.

M_{cone}	Conical surface $M_{cone} \subset M$ used to compute the axis reference point a_0.
$\mathcal{C}_{bore}$	Subset of FACES $\mathcal{C}_{bore} \subset \mathcal{C}$ which define a cylindrical surface in the CAD reference. These set of faces are used to extract M_{bore} from M.
$\mathcal{C}_{cone}$	Subset of FACES $\mathcal{C}_{cone} \subset \mathcal{C}$ which define a conical surface in the CAD reference. These set of faces are used to extract M_{cone} from M.
ε	Distance threshold (mm) used to extract the mesh features M_{bore}, M_{cone}.
d	Datum diameter $d > 0$ (mm). The point a_0 is located on the plane where the conical surface M_{cone} attains the diameter d.
P	Circular feature $P \subset M$ where the circular runout $\Delta\Phi$ is measured with respect to A. P defines a polyline perpendicular to the axis A ($P \perp A$).
T_h	Plane $T_h \subset \mathbb{R}^3$ used to extract P from M. The plane T_h has normal $\vec{v}$ and pivot point $a_0 + h\vec{v}$.
T	Temperature of the workpiece ($^\circ C$).
a	Number of workpieces for the ANOVA test.
b	Number of operators for the ANOVA test.
m	Number of measurements for the ANOVA test.

IV-E.3

Mesh Segmentation and Texture Mapping for Dimensional Inspection in Web3D

Daniel Mejia[1,2], Jairo R. Sánchez[2], Álvaro Segura[2], Oscar Ruiz-Salguero[1], Jorge Posada[2], Carlos Cadavid[1].

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, Carrera 49 N°7 Sur-50, Medellín, Colombia.

[2] Vicomtech-IK4, Paseo Mikeletegi 57, Parque Científico y Tecnológico de Guipuzcoa, Donostia/San Sebastián, Spain.

Citation

Daniel Mejia, Jairo R. Sánchez, Álvaro Segura, Oscar Ruiz-Salguero, Jorge Posada and Carlos Cadavid (**2017**). Mesh segmentation and texture mapping for dimensional inspection in Web3D. In *Proceedings of the 22nd International Conference on 3D Web Technology (Web3D ´17)*. June 05-07, Brisbane, Australia. ISBN: 978-1-4503-4955-0. DOI: 10.1145/3055624.3075954.

Indexing: SCOPUS, dblp, Semantic Scholar

Abstract

Traditionally, the data generated by industrial metrology software is stored as static reports that metrology experts produce for engineering and production departments. Nevertheless, industry demands new approaches that provide ubiquitous and real time access to overall geometry, manufacturing and other data. Web3D technologies can help to improve the traditional metrology

methods and offer new ways to convey this information in web-based continuous friendly manner. However, enriched point clouds may be massive, thus presenting transmission and display limitations. To partially overcome these limitations, this article presents an algorithm that computes efficient metrology textures, which are then transferred and displayed through Web3D standards. Texture coordinates are computed only once for the reference CAD mesh on the server using in-house thermal-based segmentation and Hessian-based parameterization algorithms. The metrology data is then encoded in a texture file, which becomes available instantly for interactive visual inspection through the Web3D platform.

Keywords: Web3D, Texture Mapping, Mesh Segmentation, Dimensional Inspection, Metrology.

IV-E.3.1 Introduction

Dimensional inspection is usually considered as the last step in a manufacturing process, to verify the deviations between the ideal model (coming from CAD systems) and the actual piece. In many cases, this happens in specialized metrology laboratories (in-house or external) depending from the quality control department. However, advanced production metholodologies such as lean manufacturing, are demanding real time control of dimensional deviations in order to eliminate manufacturing defects.

In contrast to traditional metrology approaches, these production environments need new tools that can be deployed directly on the manufacturing line instead of a laboratory. In this context, the authors have already presented a non-destructive dimensional inspection solutions based on three key tools specifically targeted to in-process metrology processes [237]: (*i*) a tool that defines the measurement process according to a base (CAD) model, (*ii*) a tool that aids the measuring of the manufactured part in the production line, and (*iii*) a tool that provides a complete web metrology report according to such measurements and the reference model. In the assembly line, these three tools target different roles: (*i*) the metrologist, (*ii*) the machine operator, and (*iii*) the production manager, respectively.

Visual computing technologies provide a set of methodologies that improve the productivity and efficiency of CAD and Manufacturing processes [55]. In such context, tools (*i*) to (*iii*) should allow an interactive data workflow with small time overheads. Current limitations in visual metrology are mostly centered in shortcomings and interaction among (*i*) analytics, (*ii*) visualization [238], and (*iii*) transfer of massive measurement and CAD data [239].

Web3D technology enables easy deployment of tools for visual metrology software and data. This paper presents a WebGL-based application that presents metrology results as a color-mapped model of the manufactured part. That is, a 3D representation of the part in which the color at each surface point represents a deviation from its theoretical shape. Our metrology software computes deviations at each vertex of the mesh obtained from scanning the real part. There are different possible approaches for this visualization: to present a mesh with per-vertex colors or to present a mesh with a texture map that contains the colors. The mesh itself can be the one obtained generated by the scanner, which represents the real part, or a mesh obtained from the reference CAD model, which represents the perfect theoretical shape. Displaying the reference shape with a texture map has clear advantages: when inspecting results of different parts corresponding to the same reference all share the same mesh but differ in their applied texture. The mesh data is transmitted only once for the given design. For each tested piece only the metrology texture

image is transmitted, instead of the full per-vertex deviation color code. In addition, image files containing the color maps are easily compressed and use much less bandwidth than per-vertex color information.

Applying a texture map to a triangular mesh requires the mesh to have a valid parameterization. That is, each vertex needs a pair of (u, v) coordinates that map this vertex to a corresponding pixel in the texture image. In order to apply a meaningful color map, these texture coordinates must be unique (no two vertices can map to the same u, v point) and such parameterization should present low distortions (i.e. it should preserve to a large extent triangle areas and angles). Therefore, a segmentation of the mesh into developable components (i.e. submeshes that can be flattened with low distortion on the plane) must be achieved prior to computing such parameterization. A survey on several mesh parameterization/segmentation techiques is presented by [30].

This manuscript presents an approach for real time dimensional inspection on Web-based applications. Our method allows rendering of the metrology data on the surface for visual inspection. The color map of the manufactured part is transferred to the users through the WebGL interface as a texture file. The texture file is computed by segmenting the CAD mesh model with a heat-based segmentation algorithm and then parameterized with the Hessian-based parameterization algorithm. The texture map file poses the advantage of carrying the visual information of the metrology measures while being relatively small compared to colored meshes from point clouds of scanned pieces.

IV-E.3.2 Literature review

The use of color maps to represent metrology data on a 3D surface is a well known technique for metrology analysis. [240] use VTK libraries to plot metrology data on the surface. [241] plot surface deviations between fitted NURBs surfaces and the CAD reference model using end-user commercial reverse engineering tools. At the same time, current quality standards require that these metrology applications preserve high precisions resulting in large volume data [242], expensive to handle, render and transfer through Web-based applications. [238] render the metrology data directly on the web server and transfers screenshots of the rendering. However, the metrology data is lost during the transfer.

Other metrology applications only collect the key information from the measured object. [215] estimate deviations of the scanned primitive shape parameters (such as planes and spheres) from the reference CAD model while [243] use electromagnetic sensors that measure the conductance of carbon-fiber polymer composites at different frequencies in order to find manufacturing defects. These closed reports can be computed more efficiently. However, they do not allow visual interaction with the metrology data nor the use of such data in advanced analytics.

The aforementioned metrology methods produce metrology files which can not be processed in web-based visual applications due to (i) large file sizes in case of visual reports, or (ii) the unavailability of visual data in text-based reports. This manuscript presents a methodology for visual dimensional inspection of manufactured parts in a Web3D context. A texture map of the metrology data is computed with mesh segmentation/parameterization algorithms. Such texture map is transferred and rendered on the reference CAD mesh of the workpiece using WebGL. The texture maps present the advantage of carrying visual information while being relatively small in size, allowing real time visualization of the data. Other geometry formats such as the *BinaryGeometry* container [244], provide more efficient data structures including incremental geometry update. However, they

are not suitable for this application due to the topologic incompatibility between different scans.

IV-E.3.3 Methodology and Results

A typical dimensional inspection pipeline based on 3D optical systems has the following steps:

1. Piece geometry acquisition: It can be done using several techniques such as structured light, laser scanners, etc. The output data is a triangle mesh in the reference frame of the scanner.
2. Registration: to align the data in (1) with its CAD representation so that they share the same reference system. There are several methods to get this alignment, being ICP fitting the most common [245].
3. Comparison: the distance of each point of the 3D scan to the CAD reference model is computed. One common method for representing visually the distance is converting it to a color using a transfer function.

In our previous work [237] the result from (3) is sent to a server and displayed using a Web3D application. However, the size of the geometry makes it prohibitive to store the full 3D data for each measurement. In order to overcome this problem, this manuscript presents an approach to generate a texture image from (3). The texture is mapped to the CAD model so that it is necessary to send the geometry only once for each reference that is going to be processed with the system.

The mapping from the 3D digitization to the texture map is done as follows:

1. For each vertex of the scanned mesh, the nearest point of the CAD model is obtained as the barycentric coordinates of the vertex projection in the nearest triangle.
2. The corresponding texture coordinates are obtained interpolating the texture coordinates of the intersected triangle using the barycentric coefficients of the projected vertex.
3. The color of the projected vertex is assigned to its corresponding texel using the interpolated texture coordinates.

In order to map the CAD model, it is necessary to segment and parameterize its mesh representation. Following sections describe the implemented algorithms for achieving an optimal solution of (1) to (3) from a point of view of metrological data representation.

IV-E.3.3.1 Heat-based Mesh Segmentation

Mesh parameterization algorithms require the input mesh to be highly developable. That is, the mesh should be able to be flattened with low distortion onto the plane. As a precondition, a segmentation of the original mesh is required, which allows the parameterization.

We segment the mesh solving the following steady state, heat diffusion process on the surface:

$$\nabla T(x) = 0, \qquad \forall x \in \mathcal{M} \tag{IV-E.3.1}$$

where $T(x)$ is the temperature at a given point x on the mesh $\mathcal{M}$ and ∇ is the Laplace-Beltrami operator [30], which dictates how heat propagates by diffusion on the surface. To solve Eq. (IV-E.3.1),

we impose temperature constraints (addressed in future manuscripts) which define the segmentation submeshes. Fig. IV-E.3.1 plots the segmentation field for the CAD mesh. Temperature constraints have been selected manually (red dots).

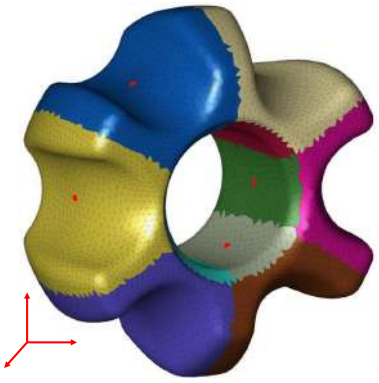


Figure IV-E.3.1: Heat-based mesh segmentation. The red dots show the location of temperature constraints (cluster centers).

IV-E.3.3.2 Hessian-based Texture Maps

After the mesh has been segmented into developable components, we process each submesh with our Hessian-based mesh parameterization algorithm [246]. Our parameterization algorithm adds HLLE dimensional reduction [69] with more robust local coordinates for degenerate cases. For computing texture coordinates, this parameterization algorithm estimates a Hessian functional $\mathcal{H}$ on each submesh:

$$\mathcal{H}f = \int_{\mathcal{M}_i} \|\mathbf{H}_x^{\tan} f\|_F^2 \mathrm{dA} \tag{IV-E.3.2}$$

where $\mathbf{H}_x^{\tan}$ is the tangent Hessian at a given point x on the submesh $\mathcal{M}_i$, $\|\cdot\|_F^2$ is the Frobenius (matrix) norm (for matrices) and dA is the area differential defined on the surface of $\mathcal{M}_i$. The texture coordinates $u(x), v(x)$ are extracted from the first two non-constant eigenvectors of the Hessian functional $\mathcal{H}$. Fig. IV-E.3.2 plots the texture coordinates for the segmented CAD mesh.

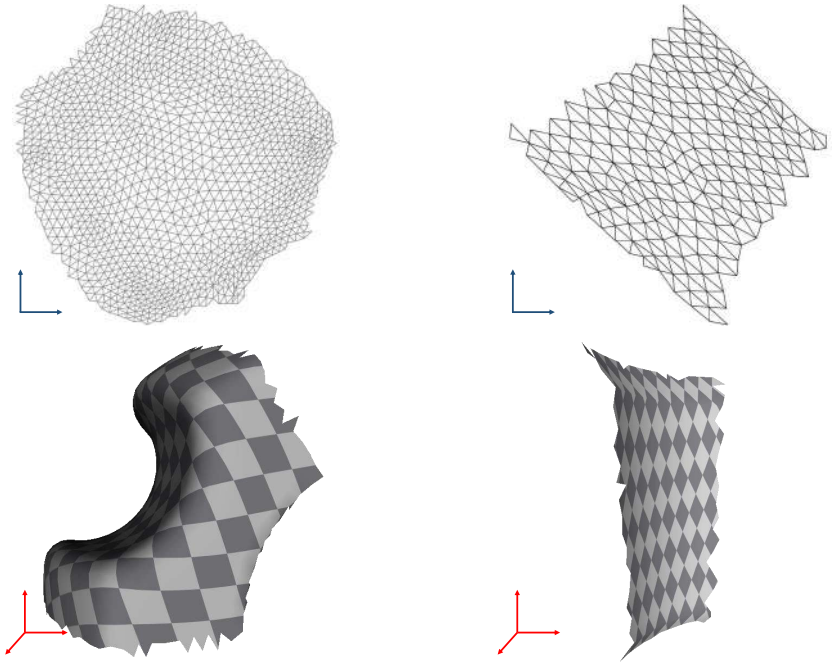


Figure IV-E.3.2: Texture coordinates for two submeshes of the segmented CAD mesh model using Hessian parameterization.

IV-E.3.3.3 3D Rendering with WebGL

Our metrology software processes scanned parts and uploads the resulting deviation data to a database server. Two kinds of files are uploaded: (a) for each reference CAD model, a mesh approximating it is stored in a JSON-based format that includes the generated texture coordinates, then for each measured part, (b) a texture image containing the measurements represented as colors is stored as a PNG image. The visualization application running in a web browser displays a list of currently measured parts and allows the user to select one. Upon selection, the application downloads the reference model mesh data only if it is different from the currently displayed part. Then, it downloads and applies the deviation color map image as a texture to the mesh. Fig. IV-E.3.3 presents the result of the metrology data rendered on the reference model using the computed texture map.

New measurement data are readily available to all users in the network as soon as it is computed. When an operator or engineer selects a different part for review corresponding to the same reference, only a texture image file needs to be downloaded. This takes a fraction of a second enabling a quick review of a number of results.

In addition to the mentioned color map image, a second image is stored which encodes the actual deviation values. This is used to retrieve the deviation at specific points clicked by the user on the part. Deviation distances are encoded into the RGB components of this second image in a way that can be reconstructed.

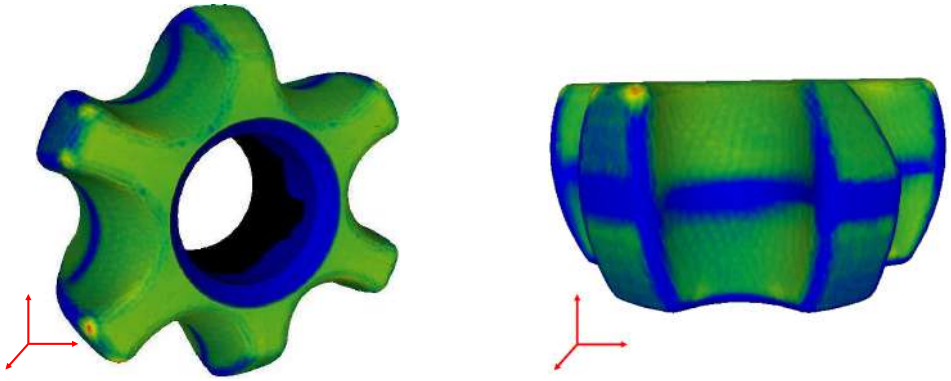


Figure IV-E.3.3: Rendering of the texture of the metrology data on the CAD mesh.

IV-E.3.3.3.1 X3D considerations

Measurement results can be represented in X3D using `IndexedFaceSet` or `IndexedTriangleSet`. In the case of per-vertex colored scanned meshes, the color filed is included with a `Color` node containing all `color` values, together with a `colorIndex` field. In the case of textured models, which is the solution addressed by this paper, meshes include the `texCoord` and `texCoordIndex` fields. The texture image is specified as a `TextureImage` node within an `Appearance` node. The part shown in the interactive viewer is embedded using an `Inline` node that includes the currently shown geometry. Normals can be omitted in order to reduce space and transfer times, letting the client compute them.

Experimentation has been done using X3DOM. In order to load the shape of a new CAD reference, the page script sets the `url` field of the `Inline` node. When a new scanned part measurement belonging to the same reference needs to be shown, the script simply sets the `url` field of the `TextureImage` node, and the color map quickly updates. In order to access the url field of this node which is inside an externally loaded `Inline` node, the system uses X3DOM's `nameSpaceName` attribute.

IV-E.3.4 Deployment

The metrology system comprising the three tools mentioned in section IV-E.3.1 has been implemented and deployed in a forging manufacturing plant. The third tool, the metrology reporting subsystem based on 3D Web technology is accessible throughout the plant, both in the shop floor HMIs and in the engineering and metrology labs. Fig. IV-E.3.4 shows the interactive web reporting tool on different devices. New measurements are made every few minutes and are instantly available to all interested staff members.

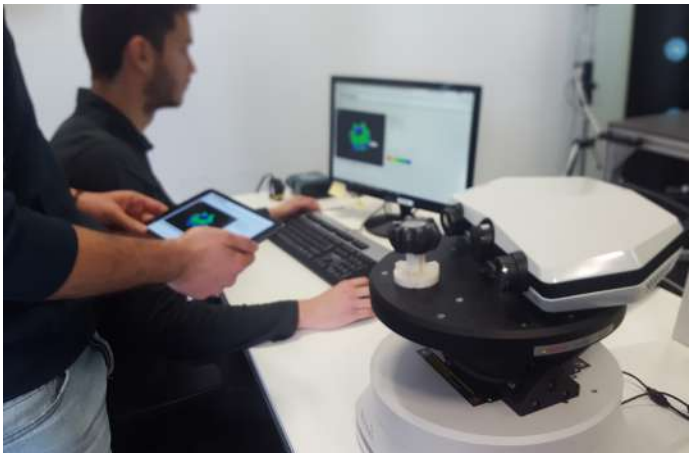


Figure IV-E.3.4: Interactive 3D web report of the metrology data deployed in a manufacturing plant.

IV-E.3.5 Conclusions and future work

This manuscript presents a methodology for Web3D real time rendering of metrology data for quality control of manufactured parts. The implemented method computes a texture map of the metrology data on the reference model by using a heat-based mesh segmentation algorithm and the Hessian-based mesh parameterization algorithm. Previous methods produce metrology files from scanned point cloud meshes which could not be processed in web-based applications in real time due to its large file sizes. In contrast, our approach produces small texture map files of the metrology data which are transferred and rendered on the CAD mesh reference model with WebGL. These metrology results are available for visualization and inspection on the server in real time.

Ongoing work addresses: (*i*) alternative mesh segmentation and parameterization algorithms for visual metrology, and (*ii*) automation of metrology web reporting.

Acknowledgements
This research was partially supported by the Basque Government under the grant Basque Industry 4.0, by GKN Legazpia, and by Sariki Metrología S.A. We thank our colleagues from GKN Legazpia who tested the algorithm in their manufacturing line and provided material for the experiments. We also thank our colleagues from Sariki Metrología S.A. whose knowledge on dimensional inspection has been very valuable during this research.

IV-F

Level Sets (Slicing) in Additive Manufacturing

IV-F.1

Level Sets of Weak-Morse Functions for Mesh Slicing in Additive Manufacturing

Daniel Mejia-Parra[1,2], Oscar Ruiz-Salguero[1], Carlos Cadavid[3], Aitor Moreno[2], Jorge Posada[2]

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, Medellín, Colombia.

[2] Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), Donostia / San Sebastián, Spain.

[3] Matemáticas y Aplicaciones, Departamento de Ciencias Matemáticas, Universidad EAFIT, Medellín, Colombia.

CONTEXT

Manuscript in progress.

Abstract

In the context of Additive Manufacturing, the problem of mesh slicing is relevant for the computation of level sets used to produce the additive infill patterns. The currently used methods make use of strong Morse functions for the generation of such level sets. However, in the presence of weak-Morse functions, these methods produce level sets with incomplete information that lead to wrong infill patterns and consequently, a 3D printed piece different from the original. To partially overcome these limitations, this article presents an application of Morse theory to improve the fidelity of the computed level sets. Our method develops the computation and characterization of Morse and non-Morse level sets on closed (without border) 2-manifold triangular meshes. The test runs show that the algorithm correctly characterizes the different types of level sets, used to generate the additive infill patterns. Ongoing work addresses the integration of the slicer into an Additive Manufacturing framework for industry applications.

Indexing: Additive Manufacturing, Morse Theory, Level Sets.

IV-F.1.1 Introduction

In the context of Additive Manufacturing, 3D printing of CAD models is mandatory for industry applications. These CAD workpiece are well-known for presenting weak-Morse behaviour. As a consequence, computer design of additive programs requires to bear in mind these weak-Morse behaviours and must correctly identify Morse and non-Morse cases during the slicing step.

This manuscript describes and classifies the different Morse and non-Morse cases for level sets of a height function on a triangular 2-manifold mesh without border. Such classification enables the computation of infill patterns using a line-polygon intersection algorithm. The resulting patterns can be used to design 3D printing programs for additive manufacturing of the original workpiece.

IV-F.1.2 Methodology

IV-F.1.2.1 Morse Function

Consider $\mathcal{M}$ as a closed (without boundary) 2-manifold embedded in $\mathbb{R}^3$. Then, for any given point $p \in \mathcal{M}$, there exists a plane $T_p\mathcal{M}$ that is tangent to $\mathcal{M}$ at the point p.

Let $\mathcal{M}$ be a closed (without boundary) 2-manifold embedded in $\mathbb{R}^3$. Then, consider a twice differentiable function $f : \mathcal{M} \to \mathbb{R}$. The function f, is a Morse function if all the critical points of f are non-degenerate, or equivalently:

$$\det(\mathbf{H}_{\mathbf{T_p}\mathcal{M}} f(p)) \neq 0 \tag{IV-F.1.1}$$

where $T_p\mathcal{M}$ is the plane that is tangent to $\mathcal{M}$ at the point $p \in \mathcal{M}$, and $\mathbf{H}_{\mathbf{T_p}\mathcal{M}}$ is the 2×2 Hessian matrix at the tangent plane $T_p\mathcal{M}$.

IV-F.1.2.2 Height Function

Consider $f : \mathcal{M} \to \mathbb{R}$ as the height function:

$$f(p) = p_z \tag{IV-F.1.2}$$

which maps each point $p \in \mathcal{M}$ to its corresponding elevation (z) value. The selection of such a function is not arbitrary as the level sets of f define the planar slices of $\mathcal{M}$, orthogonal to the $z-$axis.

IV-F.1.2.3 Level Sets of Morse Functions

A level set L_c of the function f is defined as:

$$L_c = \{p \in \mathcal{M} \mid f(p) = c\} \tag{IV-F.1.3}$$

which is the set of points in $\mathcal{M}$ that hold the value $c \in \mathbb{R}$ under the mapping f. Without loss of generality, assume L_c is connected.

In the case that f is a Morse function, one of the following three cases arises for the level set L_c:

1. If L_c contains no critical points, i.e.:

$$\nabla_{T_p\mathcal{M}} f(p) \neq 0, \forall p \in L_c \tag{IV-F.1.4}$$

then L_c defines a simple closed curve (1-manifold embedded in $\mathbb{R}^2$) without border (see Fig. IV-F.1.1(a)).

2. If L_c contains a single non-degenerate critical point, i.e.:

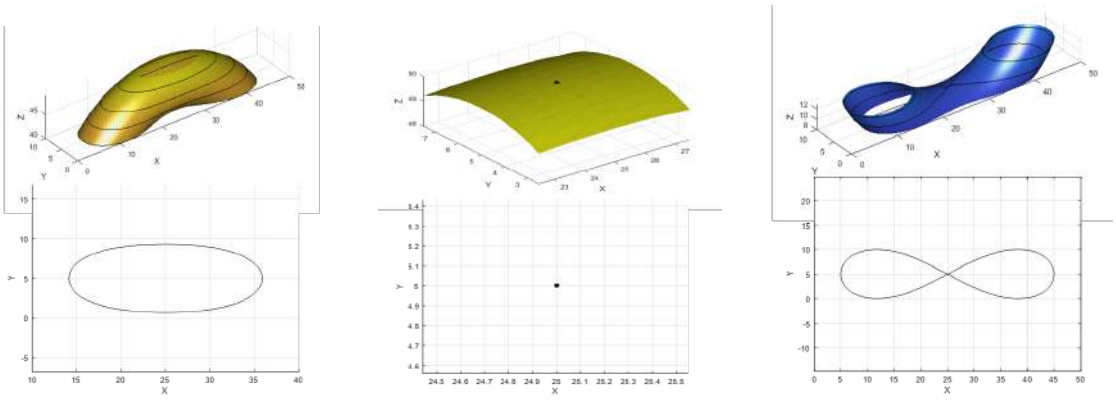
$$L_c = \{p\}, \nabla_{T_p\mathcal{M}} f(p) = 0 \tag{IV-F.1.5}$$

then $p \in \mathcal{M}$ is a local minimum (or local maximum), and L_c is composed of an isolated point (0-manifold, see Fig. IV-F.1.1(b)).

3. If L_c contains a non-isolated, non-degenerate critical point, i.e.:

$$\exists p, q \in L_c : \nabla_{T_p\mathcal{M}} f(p) = 0 \wedge \nabla_{T_q\mathcal{M}} f(q) \neq 0 \tag{IV-F.1.6}$$

then, p is a non-degenerate saddle point and L_c defines a non-manifold closed curve that self-intersects only once (see Fig. IV-F.1.1(c)).



(a) Non-intersecting L_c. No critical points.

(b) Isolated point at L_C. Local minimum/maximum.

(c) Self-intersecting level set. Non-degenerate saddle point.

Figure IV-F.1.1: Types of Morse level sets

It is important to note that $\nabla_{T_p\mathcal{M}}$ is the tangent gradient operator, defined only at the tangent plane $T_p\mathcal{M}$.

IV-F.1.2.4 Weak-Morse Function

We call f a weak-Morse function if the set of all degenerate critical values in f is finite, i.e. if the set:

$$D_f = \{f(p) \in \mathbb{R} \,|\, \det(\mathbf{H}_{\mathbf{T_p}\mathcal{M}} f(p)) = 0\} \tag{IV-F.1.7}$$

is finite.

IV-F.1.2.5 Level Sets at Non-Morse Critical Points

In the case that f is a weak-Morse function, the following additional cases are considered for the level set L_c:

1. If L_c contains a non-isolated, degenerate critical point p as per Eq. (IV-F.1.6), then p is a degenerate saddle point and L_c defines a closed (non-manifold) curve that self-intersects more than once. As an example, Fig. IV-F.1.2 presents a degenerate saddle point that self-intersects three times at p.

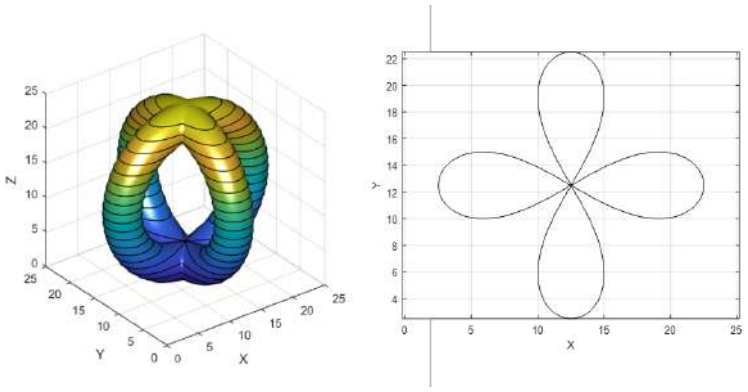


Figure IV-F.1.2: A **degenerate saddle point** occurs when the level set self-intersects more than once at the same critical point p

2. If L_c is a 1-manifold critical region, i.e.:

$$\forall p \in L_c, \nabla_{T_p\mathcal{M}} f(p) = 0 \wedge L_c \text{ is a 1-manifold} \tag{IV-F.1.8}$$

then, L_c is a local minimum (or maximum) along a 1-manifold path. Fig. IV-F.1.3 illustrates this case. It is worth noting that L_c can be an open path (i.e. with boundaries).

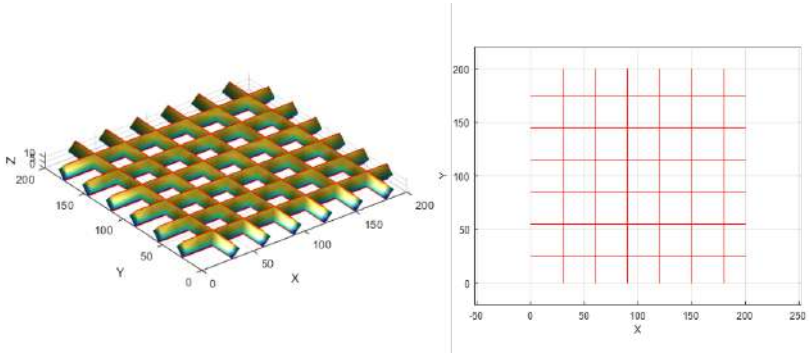


Figure IV-F.1.3: A **degenerate 1-manifold** occurs when the critical region is degenerate along a 1-manifold (possibly with boundary) path

3. If L_c is a 2-manifold critical region, i.e.:

$$\forall p \in L_c, \nabla_{T_p\mathcal{M}} f(p) = 0 \wedge L_c \text{ is an open 2-manifold} \tag{IV-F.1.9}$$

then, L_c is a critical 2-manifold region. Since L_c is embedded in a linear subspace (i.e. $\mathbb{R}^2$), then L_c is open (i.e. it has a boundary). Fig. IV-F.1.4 illustrates this case.

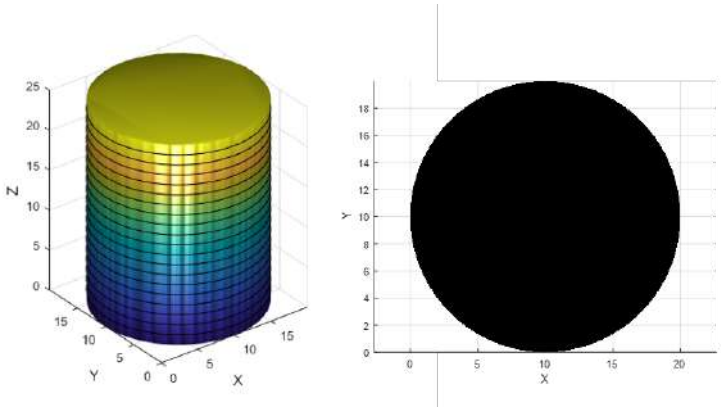


Figure IV-F.1.4: A **degenerate 2-manifold** occurs when the critical region is degenerate through a surface

IV-F.1.2.6 Summary of Level Set Cases

In summary, given a closed connected 2-manifold $\mathcal{M} \subset \mathbb{R}^3$ and a weak-Morse function $f : \mathcal{M} \to \mathbb{R}$, a level set L_c of f can be one of the following types:

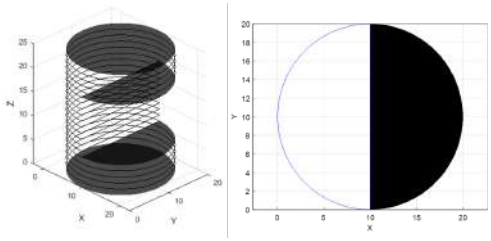
1. 1-manifold closed contour with no critical points (Morse set).
2. 0-manifold local minimum/maximum (Morse set).
3. Non-degenerate saddle point (Morse set).
4. Degenerate saddle point (non-Morse set).
5. Degenerate critical 1-manifold region (non-Morse set).
6. Degenerate critical 2-manifold region (non-Morse set).

IV-F.1.3 Results

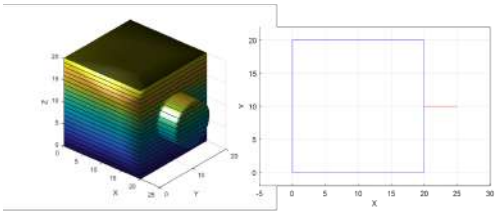
According to previously discussed classification of level sets, an algorithm for slicing triangular meshes is implemented. Given a height value c, the algorithm computes the level set L_c using the presented classification. The data structure that holds L_c stores the type of level set. Finally, since multiple cases can occur at once, the algorithm stores L_c as a union of subsets.

Fig. IV-F.1.5 presents the slicing results of different weak-Morse meshes. In Fig. IV-F.1.5(a), the level set L_c is computed as the union of a 1-manifold non-critical region and a 2-manifold critical

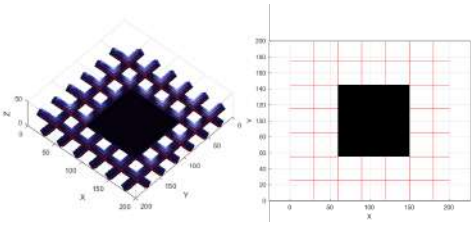
region. Similarly, Fig. IV-F.1.5(b) computes the union of 1-manifold non-critical region and a 1-manifold critical regions. Finally, Fig. IV-F.1.5(c) presents the union of degenerate 1-manifold and 2-manifold regions in L_c.



(a) Non-critical 1-manifold ∪ Critical 2-manifold



(b) Non-critical 1-manifold ∪ Critical 1-manifold



(c) Critical 1-manifold ∪ Critical 2-manifold

Figure IV-F.1.5: Computation of levels sets for several weak-Morse meshes

IV-F.1.3.1 Application in Additive Manufacturing

Our level set computation algorithms allows to back track easily the normals of $\mathcal{M}$ from each level set L_c. Using this information, and a line-polygon intersection algorithm (Clipper [247]), we are able to produce filling patterns for each L_c. Fig. IV-F.1.6 presents the computed fill patterns for a weak-Morse CAD workpiece. The resulting filling pattern can be used to produce 3D printing routines that accurately fabricate the original CAD model, even in the presence of non-Morse level sets.

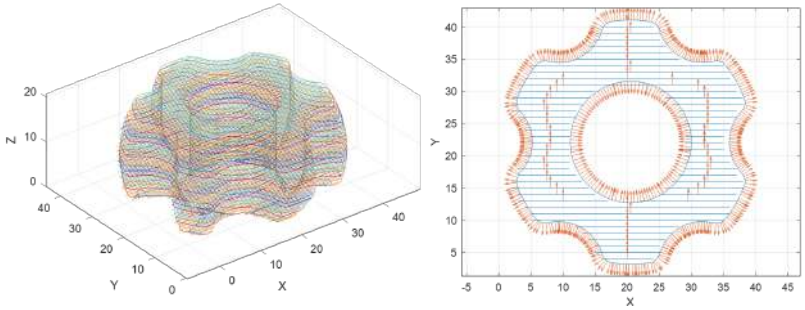


Figure IV-F.1.6: Pattern filling results on a weak-Morse CAD workpiece using level sets

IV-F.1.4 Conclusions and Future Work

This manuscripts develops a classification of closed 2-manifold level sets in terms of Morse and non-Morse critical points. The presented classification allows the identification of different types of critical points and the resulting topology of the level set. An implementation of such classification is presented for the computation of triangular mesh slices. Using existing line-polygon intersection algorithms, our method is used to produce fill patterns for the level sets of processed mesh. The implemented slicer algorithm allows backtracking of surface normals at each level set, enabling consistent line-polygon operations (specially in weak-Morse cases). The resulting filling patterns are relevant in the context of Additive Manufacturing, for the generation of 3D printing code.

Future work addresses: (a) the integration of these algorithms into an Additive Manufacturing framework, (b) analysis of optimal separation between level sets and (c) evaluation of different fill patterns in manufacture.

Part V
General Conclusions

This Thesis presents the following contributions in the topic of Differential Operators applied to different areas of CAD CAM CAE and Computer Graphics:

In the area of **Mesh Parameterization/Segmentation**, this Thesis implements different Differential Operators on Triangular Meshes such as: Laplace-Beltrami, Hessian, Heat, Curvature, and Angle/Area Distortion. These operators enable the computation of mesh fields that preserve geometric properties (area, angle, distance) as well as topologic properties (connectivity), from which mesh parameterizations and segmentations are retrieved.

The Reverse Engineering (RE) process is a user-intensive task, requiring extensive amounts of time from the engineer for the reconstruction and analysis of digitized CAD workpieces. The developed algorithms in this Thesis contribute in the automatization of segmentation/parameterization phases, reducing significantly the workload required from the user in such phases.

Furthermore, this Thesis contributes with the development of multiple Computational Geometry algorithms in Medicine (Dentistry) applications.

In the area of **Thermal Simulation of CNC Laser Machining**, this Thesis contributes with the development of an analytic method for the solution of the transient, non-homogeneous heat equation on 2-manifold rectangular plates under the effect of 1-manifold moving point sources. The developed solution enables the simulation of single-beam and multi-beam laser heating problems. In addition, this solution is implemented in GPU hardware following brute-force and Fast Fourier Transform (FFT) evaluation of the temperature for fast assessment of the simulation results.

The developed algorithms are integrated within an interactive simulator framework, which allows visualization of thermal and geometry changes in the plate during the laser heating/cutting process. Such an integration enables real-time visual interaction with the engineer for the development and optimization of CNC laser machining programs.

In the area of **Dimensional Inspection**, this Thesis contributes with:

(a) Implementation of spatial partition data structures (Perfect Spatial Hashing) for fast mesh registration of large point clouds.

(b) Implementation of mesh registration algorithms for In-line Dimensional Inspection of warm-die forged workpieces. Using Computer Vision techniques, fast reconstruction of manufactured parts is performed. Afterwards, spatial partition techniques and specific mesh sampling allow registration of the scanned workpiece with the theoretical model. Finally, assessment of dimensional compliance is performed for the registered model. The developed method enables fast direct inspection of warm-die forged workpieces in the manufacturing line, with measurement errors that satisfy industry thresholds.

(c) Implementation of mesh segmentation/parameterization algorithms for the application of dimensional deviation fields as texture maps. These texture maps enable the deployment of visual dimensional assessment reports under a Web3D framework, allowing all operators and engineers in the plant immediate access to such reports.

Finally, in the area of **Level Sets in Additive Manufacturing**, this Thesis contributes with the classification and implementation of Morse and non-Morse critical points for the computation of triangular mesh level sets. A line-polygon intersection algorithm is used to compute fill patterns for the computed level sets. The developed classification allows correct filling of non-Morse level sets. The computed fill patterns are relevant in the design of Additive Manufacturing programs.

Overall, contributions to CAD CAM CAE domains include Reverse Engineering, Medicine, CNC Laser Machining, Dimensional Inspection and Additive Manufacturing. Furthermore, Computer Graphics contributions in this Thesis include Mesh Segmentation/Parameterization, Texture Mapping, Interactive Visual Simulation and Virtual Environments, Computer Vision and Web3D.

The aforementioned contributions are the product of the joint collaboration of Universidad EAFIT and Vicomtech Research Center, Laboratorio de CAD CAM CAE (EAFIT), Department of Industry and Advanced Manufacturing (Vicomtech), and all the doctoral research team. Most of the aforementioned contributions have been screened and accepted by the international scientific community, achieving publication in indexed International Journals and Conferences. Non-published work . Published and non-published work has been approved, integrated and is being exploited commercially by different Manufacturing industries, including LANTEK Sheet Metal Solutions, GKN Driveline and BTI Biotechnology Institute.

Appendix

V-.0.A Closed Form for the Laser Heating Equation

The analytic solution to Eq. (IV-D.1.8) for a linear trajectory $\mathbf{x}_0(t) = \mathbf{v}t + \mathbf{p}$ ($\mathbf{v} = [v_x, v_y]$ and $\mathbf{p} = [p_x, p_y]$) is presented here for a point source (Dirac delta) laser beam and a square-shape laser beam.

V-.0.A.1 Dirac Delta Laser Coefficients

$$
\begin{aligned}
&\int_{t_0}^{t}\int_{0}^{b}\int_{0}^{a} f_d(\mathbf{x},\mathbf{x}_0)\mathbb{X}_i(x)\mathbb{Y}_j(y)e^{-\omega_{ij}(t-\tau)}dxdyd\tau \\
&\quad = P(1-R)\int_{t_0}^{t} e^{-\omega_{ij}(t-\tau)}\sin\frac{i\pi x_0(\tau)}{a}\sin\frac{j\pi y_0(\tau)}{b}d\tau \\
&\quad = P(1-R) \\
&\quad \Bigg[\sin\beta_x\sin\beta_y\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\cos\alpha_x\tau\cos\alpha_y\tau d\tau \\
&\quad + \sin\beta_x\cos\beta_y\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\cos\alpha_x\tau\sin\alpha_y\tau d\tau \\
&\quad + \cos\beta_x\sin\beta_y\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\sin\alpha_x\tau\cos\alpha_y\tau d\tau \\
&\quad + \cos\beta_x\cos\beta_y\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\sin\alpha_x\tau\sin\alpha_y\tau d\tau\Bigg]
\end{aligned}
\tag{1}
$$

V-.0.A.2 Square-Shape Laser Coefficients

$$
\begin{aligned}
&\int_{t_0}^{t}\int_{0}^{b}\int_{0}^{a} f_s(\mathbf{x},\mathbf{x}_0)\mathbb{X}_i(x)\mathbb{Y}_j(y)e^{-\omega_{ij}(t-\tau)}dxdyd\tau \\
&= \frac{P(1-R)}{\Delta x^2}\int_{t_0}^{t} e^{-\omega_{ij}((t-t_0)-\tau)} \\
&\left(\int_{x_0(\tau)-\frac{\Delta x}{2}}^{x_0(\tau)+\frac{\Delta x}{2}} \sin\frac{i\pi x}{a}dx\right)\left(\int_{y_0(\tau)-\frac{\Delta x}{2}}^{y_0(\tau)+\frac{\Delta x}{2}} \sin\frac{j\pi y}{b}dy\right)d\tau \\
&= \frac{abP(1-R)}{ij\pi^2\Delta x^2} \\
&\Big[c_1c_3\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\cos\alpha_x\tau\cos\alpha_y\tau d\tau \\
&- c_1c_4\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\cos\alpha_x\tau\sin\alpha_y\tau d\tau \\
&- c_2c_3\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\sin\alpha_x\tau\cos\alpha_y\tau d\tau \\
&+ c_2c_4\int_0^{t-t_0} e^{-\omega_{ij}((t-t_0)-\tau)}\sin\alpha_x\tau\sin\alpha_y\tau d\tau\Big]
\end{aligned}
\tag{2}
$$

Where:

$$
\begin{aligned}
c_1 &= \cos\beta_x - \cos\gamma_x, \quad c_2 = \sin\beta_x - \sin\gamma_x, \\
c_3 &= \cos\beta_y - \cos\gamma_y, \quad c_4 = \sin\beta_y - \sin\gamma_y, \\
\alpha_x &= \frac{i\pi v_x}{a}, \quad \alpha_y = \frac{j\pi v_y}{b}, \\
\beta_x &= \frac{i\pi(p_x+\Delta x/2)}{a}, \quad \beta_y = \frac{j\pi(p_y+\Delta x/2)}{b}, \\
\gamma_x &= \frac{i\pi(p_x-\Delta x/2)}{a}, \quad \gamma_y = \frac{j\pi(p_y-\Delta x/2)}{b}
\end{aligned}
\tag{3}
$$

$$
\begin{aligned}
&\int_0^t e^{-\omega(t-\tau)}\cos\alpha\tau\cos\beta\tau d\tau \\
&= C\left[\alpha^3\sin\alpha t\cos\beta t + \beta^3\cos\alpha t\sin\beta t + \omega^3\cos\alpha t\cos\beta t\right. \\
&- \alpha^2\beta\cos\alpha t\sin\beta t + \alpha^2\omega\cos\alpha t\cos\beta t - \alpha\beta^2\sin\alpha t\cos\beta t \\
&+ \beta^2\omega\cos\alpha t\cos\beta t + \alpha\omega^2\sin\alpha t\cos\beta t + \beta\omega^2\cos\alpha t\sin\beta t \\
&\left.+2\alpha\beta\omega\sin\alpha t\sin\beta t - \omega e^{-\omega t}(\alpha^2+\beta^2+\omega^2)\right]
\end{aligned}
\tag{4}
$$

$$
\begin{aligned}
&\int_0^t e^{-\omega(t-\tau)} \sin\alpha\tau \sin\beta\tau d\tau \\
&\quad = -C\left[\alpha^3 \cos\alpha t \sin\beta t + \beta^3 \sin\alpha t \cos\beta t - \omega^3 \sin\alpha t \sin\beta t\right. \\
&\quad - \alpha^2\beta \sin\alpha t \cos\beta t - \alpha^2\omega \sin\alpha t \sin\beta t - \alpha\beta^2 \cos\alpha t \sin\beta t \\
&\quad - \beta^2\omega \sin\alpha t \sin\beta t + \alpha\omega^2 \cos\alpha t \sin\beta t + \beta\omega^2 \sin\alpha t \cos\beta t \\
&\quad \left.-2\alpha\beta\omega(\cos\alpha t \cos\beta t - e^{-\omega t})\right]
\end{aligned} \tag{5}
$$

$$
\begin{aligned}
&\int_0^t e^{-\omega(t-\tau)} \cos\alpha\tau \sin\beta\tau d\tau \\
&\quad = C\left[\alpha^3 \sin\alpha t \sin\beta t - \beta^3 \cos\alpha t \cos\beta t + \omega^3 \cos\alpha t \sin\beta t\right. \\
&\quad + \alpha^2\beta \cos\alpha t \cos\beta t + \alpha^2\omega \cos\alpha t \sin\beta t - \alpha\beta^2 \sin\alpha t \sin\beta t \\
&\quad + \beta^2\omega \cos\alpha t \sin\beta t + \alpha\omega^2 \sin\alpha t \sin\beta t - \beta\omega^2 \cos\alpha t \cos\beta t \\
&\quad \left.-2\alpha\beta\omega \sin\alpha t \cos\beta t - \beta e^{-\omega t}(\alpha^2 - \beta^2 - \omega^2)\right]
\end{aligned} \tag{6}
$$

$$
C = \frac{1}{\alpha^4 + \beta^4 + \omega^4 - 2\alpha^2\beta^2 + 2\alpha^2\omega^2 + 2\beta^2\omega^2} \tag{7}
$$

V-.0.B Laser Heating Problem. Fast Fourier Transform Schemes.

V-.0.B.1 Scheme 1 - Discrete Sine Transform (DST)

Let $\{x_0, x_1, \ldots, x_M\}$ and $\{y_0, y_1, \ldots, y_N\}$ be uniform discretizations of the intervals $[0, a]$ and $[0, b]$, respectively. It is worth noting that for such a uniform sampling, the equalities $x_k/a = i/M$ and $y_l/b = l/N$ hold. Therefore, after truncating the number of Fourier coefficients to $(M-1)\times(N-1)$, Eq. (IV-D.4.3) is approximated as:

$$
\begin{gathered}
u_{kl}(t) = u_\infty + \sum_{m=0}^{M-2}\sum_{n=0}^{N-2} \theta_{mn} \sin(\gamma_{m+1}k) \sin(\delta_{n+1}l), \\
k = 0, 1, \ldots M-1, \qquad l = 0, 1, \ldots, N-1
\end{gathered} \tag{8}
$$

with $u_{kl}(t) = u(x_k, y_l, t)$ the temperature at the discrete points of the plate and $\gamma_m = m\pi/M$, $\delta_n = n\pi/N$ the discrete versions of α_m and β_n, respectively. This equation is equivalent to a 2D DST of the temperatures on the discrete plate (as per Eq. (IV-D.4.8)).

V-.0.B.2 Scheme 2 - FFT Padded with Zeros

Consider $\{x_0, x_1, \ldots, x_M\}$ be a uniform discretization of the interval $[0, a]$. For M Fourier coefficients the following equation holds:

$$
\begin{gathered}
\sum_{m=0}^{M-2} \theta_{mn} \sin(\gamma_{(m+1)}k) = -\mathcal{I}\left[-\sum_{m=0}^{M-2} \theta_{mn} i \sin\frac{2\gamma_{(m+1)}k}{2}\right] = -\mathcal{I}\left[\sum_{m=0}^{M-2} \theta_{mn} e^{-\frac{i2\pi}{2M}k(m+1)}\right] \\
n = 0, 1, \ldots N-1
\end{gathered} \tag{9}
$$

where $\mathcal{I}[\cdot]$ corresponds to the complex component of the series and $\gamma_m = m\pi/M$. This corresponds to a 1D DFT (Eq. (IV-D.4.7)) with M trailing zeros.

After applying the same procedure to the sequence $\{y_0, y_1, \ldots, y_N\}$, Eq. (IV-D.4.3) becomes:

$$\begin{aligned} u_{kl}(t) = u_\infty + \mathcal{I}\left[\sum_{n=0}^{N-2} \mathcal{I}\left[\sum_{m=0}^{M-2} \theta_{mn} e^{-\frac{i2\pi}{2M}k(m+1)}\right] e^{-\frac{i2\pi}{2N}l(n+1)}\right], \\ k = 0, 1, \ldots M-1, \qquad l = 0, 1, \ldots, N-1 \end{aligned} \tag{10}$$

The previous equation is equivalent to 2 nested 1D DFTs (Eq. (IV-D.4.7)) after adding 1 zero at the beginning of the Fourier sequence and M, N zeros (x and y components, respectively) at the end of the sequence.

V-.0.B.3 Scheme 3 - Odd-Symmetry 1D FFT

Consider $\{x_0, x_1, \ldots, x_M\}$ be a uniform discretization of the interval $[0, a]$. Since $\sin(x) = -\sin(-x)$ and $\sin(x) = \sin(x + 2k\pi)$ (with $k \in \mathbb{N}_+$) the following equation holds:

$$\begin{aligned} \sum_{m=0}^{M-2} \theta_{mn} \sin\frac{(m+1)k\pi}{M} &= -\sum_{m=0}^{M-2} \theta_{mn} \sin\frac{-(m+1)k\pi}{M} \\ &= -\sum_{m=0}^{M-2} \theta_{mn} \sin\left(\frac{-(m+1)k\pi}{M} + 2k\pi\right) \\ &= -\sum_{m=0}^{M-2} \theta_{mn} \sin\left(\frac{(2M-m-1)k\pi}{M}\right), \\ n =& 0, 1, \ldots N \end{aligned} \tag{11}$$

The previous series can be expressed in reverse form by setting $m \leftarrow M - m - 2$:

$$\sum_{m=0}^{M-2} \theta_{mn} \sin\frac{(m+1)k\pi}{M} = -\sum_{m=0}^{M-2} \theta_{(M-m-2)n} \sin\left(\frac{(M+m+1)k\pi}{M}\right) \tag{12}$$

Afterwards, consider the sequence shift $m = M+1, M+2, \ldots, 2M-1$. Eq. (12) becomes:

$$\sum_{m=0}^{M-2} \theta_{mn} \sin\frac{(m+1)k\pi}{M} = -\sum_{m=M+1}^{2M-1} \theta_{(2M-m-1)n} \sin\left(\frac{mk\pi}{M}\right) \tag{13}$$

which is the second half of a sine transform with negative coefficients in reverse order. Therefore, the series can be split in two as follows:

$$\sum_{m=0}^{M-2} \theta_{mn} \sin\frac{(m+1)k\pi}{M} = \frac{1}{2}\sum_{m=0}^{M-2} \theta_{mn} \sin\frac{(m+1)k\pi}{M} + \frac{1}{2}\sum_{m=M+1}^{2M-1} -\theta_{(2M-m-1)n} \sin\left(\frac{mk\pi}{M}\right) \tag{14}$$

On the other hand, from Eq. (IV-D.4.7):

$$\phi_m \sin\frac{mk\pi}{M} = -\mathcal{I}\left[-\phi_m i \sin\frac{2mk\pi}{2M}\right] = -\mathcal{I}\left[\phi_m e^{-\frac{i2\pi}{2M}km}\right] \tag{15}$$

where $\mathcal{I}[\cdot]$ corresponds to the complex component of the Fourier term.

Putting together Eqs. (14) and (15), Eq. (IV-D.4.3) becomes:

$$\begin{aligned} u_{kl}(t) = u_\infty &+ \frac{1}{4}\mathcal{I}\left[\sum_{n=0}^{N-2}\mathcal{I}\left[\sum_{m=0}^{M-2}\theta_{mn}e^{-\frac{i2\pi}{2M}k(m+1)}\right]e^{-\frac{i2\pi}{2N}l(n+1)}\right] \\ &- \frac{1}{4}\mathcal{I}\left[\sum_{n=0}^{N-2}\mathcal{I}\left[\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)n}e^{-\frac{i2\pi}{2M}km}\right]e^{-\frac{i2\pi}{2N}l(n+1)}\right] \\ &- \frac{1}{4}\mathcal{I}\left[\sum_{n=N+1}^{2N-1}\mathcal{I}\left[\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}e^{-\frac{i2\pi}{2M}k(m+1)}\right]e^{-\frac{i2\pi}{2N}ln}\right] \\ &+ \frac{1}{4}\mathcal{I}\left[\sum_{n=N+1}^{2N-1}\mathcal{I}\left[\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)(2N-n-1)}e^{-\frac{i2\pi}{2M}km}\right]e^{-\frac{i2\pi}{2N}ln}\right] \\ k = 0, 1, \ldots M-1, &\qquad l = 0, 1, \ldots, N-1 \end{aligned} \tag{16}$$

The previous equation is equivalent to 2 nested 1D DFTs (Eq. (IV-D.4.7)) after padding the $M-2, N-2$ coefficients in reverse order (and negative) at the end of the original Fourier coefficients in each direction (x and y), respectively. The final result is retrieved by taking the complex part (i.e. the sine component) of each 1D DFT.

V-.0.B.4 Scheme 4 - Odd-Symmetry 2D FFT

In this scheme, consider the real part D_{kl} (instead of the complex one) of Eq. (16) as follows:

$$\begin{aligned} D_{kl} = \frac{1}{4}&\mathcal{R}e\left[\sum_{n=0}^{N-2}\mathcal{R}e\left[\sum_{m=0}^{M-2}\theta_{mn}e^{-\frac{i2\pi}{2M}k(m+1)}\right]e^{-\frac{i2\pi}{2N}l(n+1)}\right] \\ &- \frac{1}{4}\mathcal{R}e\left[\sum_{n=0}^{N-2}\mathcal{R}e\left[\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)n}e^{-\frac{i2\pi}{2M}km}\right]e^{-\frac{i2\pi}{2N}l(n+1)}\right] \\ &- \frac{1}{4}\mathcal{R}e\left[\sum_{n=N+1}^{2N-1}\mathcal{R}e\left[\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}e^{-\frac{i2\pi}{2M}k(m+1)}\right]e^{-\frac{i2\pi}{2N}ln}\right] \\ &+ \frac{1}{4}\mathcal{R}e\left[\sum_{n=N+1}^{2N-1}\mathcal{R}e\left[\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)(2N-n-1)}e^{-\frac{i2\pi}{2M}km}\right]e^{-\frac{i2\pi}{2N}ln}\right] \\ k = 0, 1, \ldots M-1, &\qquad l = 0, 1, \ldots, N-1 \end{aligned} \tag{17}$$

which in fact consists of the cosine parts of the Fourier series:

$$\begin{aligned} D_{kl} = & \frac{1}{4}\sum_{n=0}^{N-2}\sum_{m=0}^{M-2}\theta_{mn}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi l(n+1)}{2N}\right) \\ & -\frac{1}{4}\sum_{n=0}^{N-2}\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)n}\cos\left(\frac{2\pi km}{2M}\right)\cos\left(\frac{2\pi l(n+1)}{2N}\right) \\ & -\frac{1}{4}\sum_{n=N+1}^{2N-1}\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi ln}{2N}\right) \\ & +\frac{1}{4}\sum_{n=N+1}^{2N-1}\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)(2N-n-1)}\cos\left(\frac{2\pi km}{2M}\right)\cos\left(\frac{2\pi ln}{2N}\right) \\ & k=0,1,\dots M-1, \qquad l=0,1,\dots,N-1 \end{aligned} \tag{18}$$

Consider the second term of the previous expansion, and the change of the series variable $m \leftarrow 2M-m-1$. Since $\cos(x)=\cos(-x)$ and $\cos(x)=\cos(x+2\pi k)$ (with $k\in\mathbb{N}_+$), then the following equation holds:

$$\begin{aligned} & \frac{1}{4}\sum_{n=0}^{N-2}\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)n}\cos\left(\frac{2\pi km}{2M}\right)\cos\left(\frac{2\pi l(n+1)}{2N}\right) \\ & =\frac{1}{4}\sum_{n=0}^{N-2}\sum_{m=0}^{M-2}\theta_{mn}\cos\left(\frac{2\pi k(2M-1-m)}{2M}\right)\cos\left(\frac{2\pi l(n+1)}{2N}\right) \\ & =\frac{1}{4}\sum_{n=0}^{N-2}\sum_{m=0}^{M-2}\theta_{mn}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi l(n+1)}{2N}\right) \end{aligned} \tag{19}$$

Applying the same procedure to the fourth term in Eq. (18), we obtain:

$$\begin{aligned} & \frac{1}{4}\sum_{n=N+1}^{2N-1}\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)(2N-n-1)}\cos\left(\frac{2\pi km}{2M}\right)\cos\left(\frac{2\pi ln}{2N}\right) \\ & =\frac{1}{4}\sum_{n=N+1}^{2N-1}\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}\cos\left(\frac{2\pi k(2M-1-m)}{2M}\right)\cos\left(\frac{2\pi ln}{2N}\right) \\ & =\frac{1}{4}\sum_{n=N+1}^{2N-1}\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi ln}{2N}\right) \end{aligned} \tag{20}$$

Substituting Eqs. (19) and (20) into Eq. (18):

$$\begin{aligned}
D_{kl} &= \frac{1}{4}\sum_{n=0}^{N-2}\sum_{m=0}^{M-2}\theta_{mn}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi l(n+1)}{2N}\right) \\
&- \frac{1}{4}\sum_{n=0}^{N-2}\sum_{m=0}^{M-2}\theta_{mn}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi l(n+1)}{2N}\right) \\
&- \frac{1}{4}\sum_{n=N+1}^{2N-1}\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi ln}{2N}\right) \\
&+ \frac{1}{4}\sum_{n=N+1}^{2N-1}\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}\cos\left(\frac{2\pi k(m+1)}{2M}\right)\cos\left(\frac{2\pi ln}{2N}\right) \\
k &= 0,1,\dots M-1, \qquad l = 0,1,\dots,N-1
\end{aligned} \tag{21}$$

in which the first and second terms cancel out, as well as terms three and four, respectively. Therefore:

$$\forall_{k,l} \qquad D_{kl} = 0 \tag{22}$$

Finally, we add D_{kl} to Eq. (16):

$$\begin{aligned}
u_{kl}(t) &= u_{kl}(t) + D_{kl} \\
&= u_\infty + \frac{1}{4}\mathcal{R}e\left[\sum_{n=0}^{N-2}\sum_{m=0}^{M-2}\theta_{mn}e^{-\frac{i2\pi}{2M}k(m+1)}e^{-\frac{i2\pi}{2N}l(n+1)}\right] \\
&- \frac{1}{4}\mathcal{R}e\left[\sum_{n=0}^{N-2}\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)n}e^{-\frac{i2\pi}{2M}km}e^{-\frac{i2\pi}{2N}l(n+1)}\right] \\
&- \frac{1}{4}\mathcal{R}e\left[\sum_{n=N+1}^{2N-1}\sum_{m=0}^{M-2}\theta_{m(2N-n-1)}e^{-\frac{i2\pi}{2M}k(m+1)}e^{-\frac{i2\pi}{2N}ln}\right] \\
&+ \frac{1}{4}\mathcal{R}e\left[\sum_{n=N+1}^{2N-1}\sum_{m=M+1}^{2M-1}\theta_{(2M-m-1)(2N-n-1)}e^{-\frac{i2\pi}{2M}km}e^{-\frac{i2\pi}{2N}ln}\right] \\
k &= 0,1,\dots M-1, \qquad l = 0,1,\dots,N-1
\end{aligned} \tag{23}$$

The above equation is true since $\mathcal{R}e(xy) = \mathcal{R}e(x)\mathcal{R}e(y) + \mathcal{I}(x)\mathcal{I}(y)$ (i.e. the real part of the product of two complex numbers is the sum of their real parts and their imaginary parts). Eq. (23) is equivalent to a 2D DFT (Eq. (IV-D.4.7)) after padding the $M-2, N-2$ coefficients in reverse order (and negative) at the end of the original Fourier coefficients in each direction (x and y), respectively. The final result is retrieved by taking the real part of the result.

With more modern GPU hardware (GeForce RTX2060) even faster results can be acquired which shows the potential of the proposed algorithms towards real-time laser heating/cutting simulations and flexible manufacturing scenarios that require fast tool-planning capability and laser parameter optimization to easily adapt to customer order changes.

Future work includes (1) the inclusion of thermal/stress models for structural analysis of the plate after the generated high temperature gradients, (2) analysis of non-rectangular plate ge-

ometries, and (3) consideration of non-linear interactions such as temperature-dependent thermal properties and phase changes.

Bibliography

[1] M. Desbrun, M. Meyer, and P. Alliez. Intrinsic parameterizations of surface meshes. In *Eurographics 2002 Conference Proceedings*, 2002.

[2] P. V. Sander, S. J. Gortler, J. Snyder, and H. Hoppe. Signal-specialized parametrization. In *Proceedings of the 13th Eurographics Workshop on Rendering*, EGRW '02, pages 87–98, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association.

[3] S. Yoshizawa, A. Belyaev, and H.-P. Seidel. A fast and simple stretch-minimizing mesh parameterization. In *Shape Modeling Applications, 2004. Proceedings*, pages 200–208, June 2004.

[4] K. Hormann and G. Greiner. MIPS: An efficient global parametrization method. In P.-J. Laurent, P. Sablonnière, and L. L. Schumaker, editors, *Curve and Surface Design: Saint-Malo 1999*, Innovations in Applied Mathematics, pages 153–162. Vanderbilt University Press, Nashville, TN, 2000.

[5] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371, July 2002.

[6] Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. A local/global approach to mesh parameterization. *Comput Graph Forum*, 27(5):1495–1504, 2008.

[7] R. Zayer, C. Rössl, and H.-P. Seidel. Setting the boundary free: A composite approach to surface parameterization. In *Proceedings of the Third Eurographics Symposium on Geometry Processing*, SGP '05, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[8] Theodoros Athanasiadis, Georgios Zioupos, and Ioannis Fudos. Efficient computation of constrained parameterizations on parallel platforms. *Comput Graph*, 37(6):596–607, 2013.

[9] L. Kharevych, B. Springborn, and P. Schröder. Discrete conformal mappings via circle patterns. *ACM Trans. Graph.*, 25(2):412–438, April 2006.

[10] M. Ben-Chen, C. Gotsman, and G. Bunin. Conformal flattening by curvature prescription and metric scaling. *Computer Graphics Forum*, 27(2):449–458, 2008.

[11] B. Springborn, P. Schröder, and U. Pinkall. Conformal equivalence of triangle meshes. *ACM Trans. Graph.*, 27(3):77:1–77:11, August 2008.

[12] A. Sheffer and E. de Sturler. Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers*, 17(3):326–337, 2001.

[13] Alla Sheffer, Bruno Lévy, Maxim Mogilnitsky, and Alexander Bogomyakov. Abf++: Fast and robust angle based flattening. *ACM Trans Graph*, 24(2):311–330, April 2005.

[14] Rhaleb Zayer, Bruno Lévy, and Hans-Peter Seidel. Linear angle based parameterization. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 135–141, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[15] Xianfang Sun and Edwin R. Hancock. Quasi-isometric parameterization for texture mapping. *Pattern Recognition*, 41(5):1732–1743, 2008.

[16] Oscar E. Ruiz, Daniel Mejia, and Carlos A. Cadavid. Triangular mesh parameterization with trimmed surfaces. *Int J Interact Des Manuf (IJIDeM)*, 9(4):303–316, Nov 2015.

[17] Jason Smith and Scott Schaefer. Bijective parameterization with free boundaries. *ACM Trans Graph*, 34(4):70:1–70:9, July 2015.

[18] A. Ravindran, K. M. Ragsdell, and G. V. Reklaitis. *Functions of Several Variables*, pages 78–148. John Wiley & Sons, Inc., 2007.

[19] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.

[20] T. F. Edgar and D. M. Himmelblau. *Optimization of chemical processes*. McGraw-Hill chemical engineering series. McGraw-Hill, 1988, New York, 1989.

[21] K. Ueda and N. Yamashita. Global complexity bound analysis of the levenberg–marquardt method for nonsmooth equations and its application to the nonlinear complementarity problem. *Journal of Optimization Theory and Applications*, 152(2):450–467, 2012.

[22] M. Botsch, D. Bommes, and L. Kobbelt. Efficient linear system solvers for mesh processing. In Ralph Martin, Helmut Bez, and Malcolm Sabin, editors, *Mathematics of Surfaces XI*, volume 3604 of *Lecture Notes in Computer Science*, pages 62–83. Springer Berlin Heidelberg, 2005.

[23] C. Cartis, N. I. M. Gould, and Ph. L. Toint. On the complexity of steepest descent, newton's and regularized newton's methods for nonconvex unconstrained optimization problems. *SIAM Journal on Optimization*, 20(6):2833–2852, 2010.

[24] ALICE. Unwrapped meshes. `http://alice.loria.fr/index.php/software/7-data/37-unwrapped-meshes.htm`, 2008. Accessed: 2015-07-20.

[25] Alla Sheffer and John C. Hart. Seamster: Inconspicuous low-distortion texture seam layout. In *Proceedings of the Conference on Visualization '02*, VIS '02, pages 291–298, Washington, DC, USA, 2002. IEEE Computer Society.

[26] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[27] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, June 2003.

[28] Michael S. Floater and Kai Hormann. Surface parameterization: a tutorial and survey. In Neil A. Dodgson, Michael S. Floater, and Malcolm A. Sabin, editors, *Advances in Multiresolution for Geometric Modelling*, pages 157–186, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[29] Alla Sheffer, Emil Praun, and Kenneth Rose. Mesh parameterization methods and their applications. *Found. Trends. Comput. Graph. Vis.*, 2(2):105–171, January 2006.

[30] Kai Hormann, Konrad Polthier, and Alia Sheffer. Mesh parameterization: Theory and practice. In *ACM SIGGRAPH ASIA 2008 Courses*, SIGGRAPH Asia '08, pages 12:1–12:87, New York, NY, USA, 2008. ACM.

[31] G. Zou, J. Hu, X. Gu, and J. Hua. Authalic parameterization of general surfaces using lie advection. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2005–2014, Dec 2011.

[32] Xin Zhao, Zhengyu Su, Xianfeng David Gu, Arie Kaufman, Jian Sun, Jie Gao, and Feng Luo. Area-preservation mapping using optimal mass transport. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2838–2847, December 2013.

[33] Kehua Su, Li Cui, Kun Qian, Na Lei, Junwei Zhang, Min Zhang, and Xianfeng David Gu. Area-preserving mesh parameterization for poly-annulus surfaces based on optimal mass transportation. *Computer Aided Geometric Design*, 46:76–91, 2016.

[34] Mei-Heng Yueh, Wen-Wei Lin, Chin-Tien Wu, and Shing-Tung Yau. An efficient energy minimization for conformal parameterizations. *Journal of Scientific Computing*, 73(1):203–227, Oct 2017.

[35] Zhao Wang, Zhongxuan Luo, Jielin Zhang, and Emil Saucan. A novel local/global approach to spherical parameterization. *Journal of Computational and Applied Mathematics*, 329:294–306, 2018. The International Conference on Information and Computational Science, 2–6 August 2016, Dalian, China.

[36] H. Yu, T. Lee, I. Yeh, X. Yang, W. Li, and J. J. Zhang. An rbf-based reparameterization method for constrained texture mapping. *IEEE Transactions on Visualization and Computer Graphics*, 18(7):1115–1124, July 2012.

[37] Chengming Liu, Zhongxuan Luo, Xiquan Shi, Fengshan Liu, and Xiaonan Luo. A fast mesh parameterization algorithm based on 4-point interpolatory subdivision. *Applied Mathematics and Computation*, 219(10):5339–5344, 2013.

[38] Rohan Sawhney and Keenan Crane. Boundary first flattening. *ACM Trans. Graph.*, 37(1):5:1–5:14, December 2017.

[39] Alon Bright, Edward Chien, and Ofir Weber. Harmonic global parametrization with rational holonomy. *ACM Transactions on Graphics*, 36(4):89:1–89:15, July 2017.

[40] Daniel Mejia, Oscar Ruiz-Salguero, and Carlos A. Cadavid. Hessian eigenfunctions for triangular mesh parameterization. In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 1: GRAPP, (VISIGRAPP 2016)*, pages 75–82. INSTICC, SciTePress, 2016.

[41] Daniel Mejia, Diego A. Acosta, and Oscar Ruiz-Salguero. Weighted area/angle distortion minimization for mesh parameterization. *Engineering Computations*, 34(6):1874–1895, 2017.

[42] Xiaokang Yu, Na Lei, Xiaopeng Zheng, and Xianfeng Gu. Surface parameterization based on polar factorization. *Journal of Computational and Applied Mathematics*, 329:24–36, 2018. The International Conference on Information and Computational Science, 2–6 August 2016, Dalian, China.

[43] Zhao Wang, Zhong-xuan Luo, Jie-lin Zhang, and Emil Saucan. Arap++: an extension of the local/global approach to mesh parameterization. *Front Inf Technol Electron Eng*, 17(6):501–515, Jun 2016.

[44] Michael Rabinovich, Roi Poranne, Daniele Panozzo, and Olga Sorkine-Hornung. Scalable locally injective mappings. *ACM Trans. Graph.*, 36(4), April 2017.

[45] Zhong Li, Yao Jin, Xiaogang Jin, and Lizhuang Ma. Approximate straightest path computation and its application in parameterization. *The Visual Computer*, 28(1):63–74, Jan 2012.

[46] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[47] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph.*, 32(5):152:1–152:11, October 2013.

[48] MATLAB®. *version 9.4.0.813654 (R2018a)*. MathWorks®, Natick, Massachusetts, USA, 2018.

[49] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE International Conference on Robotics and Automation*, pages 1–4, May 2011.

[50] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In Vittorio Scarano, Rosario De Chiara, and Ugo Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008.

[51] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Proceedings of the Symposium on Geometry Processing*, SGP '09, pages 1383–1392, Aire-la-Ville, Switzerland, Switzerland, 2009. Eurographics Association.

[52] Martin Reuter, Silvia Biasotti, Daniela Giorgi, Giuseppe Patanè, and Michela Spagnuolo. Discrete laplace–beltrami operators for shape analysis and segmentation. *Comput Graph*, 33(3):381 – 390, 2009. IEEE International Conference on Shape Modelling and Applications 2009.

[53] Michael A. A. Cox and Trevor F. Cox. *Multidimensional Scaling*, pages 315–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[54] Daniel Mejia, Oscar Ruiz-Salguero, Jairo R. Sánchez, Jorge Posada, Aitor Moreno, and Carlos A. Cadavid. Hybrid geometry / topology based mesh segmentation for reverse engineering. *Computers & Graphics*, 73:47–58, 2018.

[55] J. Posada, C. Toro, I. Barandiaran, D. Oyarzun, D. Stricker, R. de Amicis, E. B. Pinto, P. Eisert, J. Döllner, and I. Vallarino. Visual computing as a key enabling technology for industrie 4.0 and industrial internet. *IEEE Comput Graph Appl*, 35(2):26–40, Mar 2015.

[56] Rui S.V. Rodrigues, José F.M. Morgado, and Abel J.P. Gomes. A contour-based segmentation algorithm for triangle meshes in 3d space. *Comput Graph*, 49:24–35, 2015.

[57] Dong Xiao, Hongwei Lin, Chuhua Xian, and Shuming Gao. Cad mesh model segmentation by clustering. *Comput Graph*, 35(3):685–691, 2011.

[58] Bing Yi, Zhenyu Liu, Jianrong Tan, Fengbei Cheng, Guifang Duan, and Ligang Liu. Shape recognition of cad models via iterative slippage analysis. *Comput Aided Des*, 55:13–25, 2014.

[59] Jun Wang and Zeyun Yu. Surface feature based mesh segmentation. *Comput Graph*, 35(3):661–667, 2011.

[60] Daniel Mejia, Oscar Ruiz-Salguero, and Carlos A. Cadavid. Spectral-based mesh segmentation. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 11(3):503–514, Aug 2017.

[61] Juyong Zhang, Jianmin Zheng, Chunlin Wu, and Jianfei Cai. Variational mesh decomposition. *ACM Trans Graph*, 31(3):21:1–21:14, June 2012.

[62] Bruno Lévy and Hao (Richard) Zhang. Spectral mesh processing. In *ACM SIGGRAPH 2010 Courses*, SIGGRAPH '10, pages 8:1–8:312, New York, NY, USA, 2010. ACM.

[63] Rong Liu and Hao Zhang. Mesh segmentation via spectral embedding and contour analysis. *Computer Graphics Forum*, 26(3):385–394, 2007.

[64] Hao Wang, Tong Lu, Oscar Kin-Chung Au, and Chiew-Lan Tai. Spectral 3d mesh segmentation with a novel single segmentation field. *Graphical Models*, 76(5):440–456, 2014. Geometric Modeling and Processing 2014.

[65] Tao Liao, Xinge Li, Guoliang Xu, and Yongjie Jessica Zhang. Secondary laplace operator and generalized giaquinta–hildebrandt operator with applications on surface segmentation and smoothing. *Comput Aided Des*, 70:56–66, 2016.

[66] Colin Cartade, Christian Mercat, Rémy Malgouyres, and Chafik Samir. Mesh parameterization with generalized discrete conformal maps. *Journal of Mathematical Imaging and Vision*, 46(1):1–11, May 2013.

[67] Daniel Mejia, Diego A. Acosta, and Oscar Ruiz-Salguero. Weighted area/angle distortion minimization for mesh parameterization. *Engineering Computations*, 34(6):1874–1895, 2017.

[68] Zhonghua Xi, Yun hyeong Kim, Young J. Kim, and Jyh-Ming Lien. Learning to segment and unfold polyhedral mesh from failures. *Computers & Graphics*, 58(Supplement C):139–149, 2016. Shape Modeling International 2016.

[69] David L. Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *P Natl Acad Sci USA*, 100(10):5591–5596, 2003.

[70] Xiaobai Chen, Aleksey Golovinskiy, and Thomas Funkhouser. A benchmark for 3d mesh segmentation. *ACM Trans. Graph.*, 28(3):73:1–73:12, July 2009.

[71] Gopinath Chintala and Prasad Gudimetla. Optimum material evaluation for gas turbine blade using reverse engineering (re) and fea. *Procedia Eng*, 97:1332–1340, 2014.

[72] T. Waggoner. Exporting solid models into finite element analysis (fea) for reverse engineering of electrical components. In *Proceedings Electrical Insulation Conference and Electrical Manufacturing Expo, 2005.*, pages 231–235, Oct 2005.

[73] Manuel J. García, Pierre Boulanger, and Miguel Henao. Structural optimization of as-built parts using reverse engineering and evolution strategies. *Struct Multidiscip Optim*, 35(6):541–550, Jun 2008.

[74] Damir Vučina and Igor Pehnec. *Enhanced Reverse Engineering Using Genetic-Algorithms-Based Experimental Parallel Workflow for Optimum Design*, pages 172–183. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[75] Bahram Asiabanpour, Abel Ardis, and Andres Alvarez Andrade. A systematic use of reverse engineering in evaluating the overall accuracy of the fabricated parts. *Int J Rapid Manuf*, 4(2-4):165–178, 2014.

[76] Daniel Mejia, Jairo R. Sánchez, Álvaro Segura, Oscar Ruiz-Salguero, Jorge Posada, and Carlos Cadavid. Mesh segmentation and texture mapping for dimensional inspection in web3d. In *Proceedings of the 22Nd International Conference on 3D Web Technology*, Web3D '17, pages 3:1–3:4, New York, NY, USA, 2017. ACM.

[77] Stella Orozco, Arno Formella, Carlos A. Cadavid, Oscar Ruiz-Salguero, and Maria Osorno. Geometry and topology-based segmentation of 2-manifold triangular meshes in r3. *Br J Appl Sci Technol*, 21(1):1–14, 2017.

[78] Jun Wang, Dongxiao Gu, Zeyun Yu, Changbai Tan, and Laishui Zhou. A framework for 3d model reconstruction in reverse engineering. *Comput Ind Eng*, 63(4):1189–1200, 2012.

[79] Roseline Bénière, Gérard Subsol, Gilles Gesquière, Francois Le Breton, and William Puech. A comprehensive process of reverse engineering from 3d meshes to cad models. *Comput Aided Des*, 45(11):1382–1393, 2013.

[80] Daniel Mejia, Oscar Ruiz-Salguero, and Carlos A. Cadavid. Hessian eigenfunctions for triangular mesh parameterization. In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2016)*, pages 75–82, 2016.

[81] H. Zhang, O. Van Kaick, and R. Dyer. Spectral mesh processing. *Comput Graph Forum*, 29(6):1865–1894, 2010.

[82] Daniel Mejia, Oscar Ruiz-Salguero, and Carlos A. Cadavid. Spectral-based mesh segmentation. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 11(3):503–514, Aug 2017.

[83] Aleksey Golovinskiy and Thomas Funkhouser. Consistent segmentation of 3d models. *Comput Graph*, 33(3):262 – 269, 2009.

[84] Xiuping Liu, Jie Zhang, Risheng Liu, Bo Li, Jun Wang, and Junjie Cao. Low-rank 3d mesh segmentation and labeling with structure guiding. *Comput Graph*, 46:99–109, 2015.

[85] Truc Le, Giang Bui, and Ye Duan. A multi-view recurrent neural network for 3d mesh segmentation. *Comput Graph*, 66(Supplement C):103–112, 2017. Shape Modeling International 2017.

[86] Hao Wang, Tong Lu, Oscar Kin-Chung Au, and Chiew-Lan Tai. Spectral 3d mesh segmentation with a novel single segmentation field. *Graph Model*, 76(5):440–456, 2014.

[87] Youyi Zheng and Chiew-Lan Tai. Mesh decomposition with cross-boundary brushes. *Comput Graph Forum*, 29(2):527–535, 2010.

[88] K. Gĕbal, J. A. Bærentzen, H. Aanæs, and R. Larsen. Shape analysis using the auto diffusion function. *Comput Graph Forum*, 28(5):1405–1413, 2009.

[89] William Benjamin, Andrew Wood Polk, S.V.N. Vishwanathan, and Karthik Ramani. Heat walk: Robust salient segmentation of non-rigid shapes. *Comput Graph Forum*, 30(7):2097–2106, 2011.

[90] Y. Fang, M. Sun, M. Kim, and K. Ramani. Heat-mapping: A robust approach toward perceptually consistent mesh segmentation. In *CVPR 2011*, pages 2145–2152, June 2011.

[91] Min Meng, Lubin Fan, and Ligang Liu. A comparative evaluation of foreground/background sketch-based mesh segmentation algorithms. *Comput Graph*, 35(3):650–660, 2011.

[92] Marek Vanco and Guido Brunnett. Direct segmentation of algebraic models for reverse engineering. *Computing*, 72(1):207–220, Apr 2004.

[93] Kunal Soni, Daniel Chen, and Terence Lerch. Parameterization of prismatic shapes and reconstruction of free-form shapes in reverse engineering. *Int J Adv Manuf Technol*, 41(9):948, Apr 2009.

[94] Yi-Jun Yang, Wei Zeng, and Jian-Feng Chen. Equiareal parameterizations of nurbs surfaces. *Graphical Models*, 76(1):43–55, 2014.

[95] Yi-Jun Yang, Wei Zeng, and Xiang-Xu Meng. Conformal freeform surfaces. *Computer-Aided Design*, 81:48–60, 2016.

[96] Ulrich Pinkall and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15 – 36, 1993.

[97] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 81 – 90, New York, NY, USA, 2004. ACM.

[98] Phillip G. Schmitz and Lexing Ying. A fast direct solver for elliptic problems on general meshes in 2d. *Journal of Computational Physics*, 231(4):1314 – 1338, 2012.

[99] William C. Regli, Cheryl Foster, Erik Hayes, Cheuk Yiu Ip, David Mcwherter, Mitchell Peabody, Yuriy Shapirsteyn, and Vera Zaychik. National design repository project: A status report. In *International Joint Conferences on Artificial Intelligence (IJCAI) and AAAI/SIGMAN Workshop on AI in Manufacturing Systems*, 2001.

[100] Aleksey Golovinskiy and Thomas Funkhouser. Randomized cuts for 3d mesh analysis. *ACM Trans. Graph.*, 27(5):145:1–145:12, December 2008.

[101] Yu-Kun Lai, Shi-Min Hu, Ralph R. Martin, and Paul L. Rosin. Fast mesh segmentation using random walks. In *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*, SPM '08, pages 183–191, New York, NY, USA, 2008. ACM.

[102] Marco Attene, Bianca Falcidieno, and Michela Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer*, 22(3):181–193, Mar 2006.

[103] Y. Yan, L. Ji, Y. Bao, X. Chen, and Y. Jiang. CO2 laser high-speed crack-free cutting of thick-section alumina based on close-piercing lapping technique. *Int. J. Adv. Manuf. Tech.*, 64(9):1611 – 1624, 2012.

[104] A. Joshi, N. Kansara, S. Das, P. Kuppan, and K. Venkatesan. A study of temperature distribution for laser assisted machining of Ti-6Al-4 V alloy. *Procedia Engineering*, 97:1466 – 1473, 2014. 12th Global Congress on Manufacturing and Management GCMM - 2014.

[105] H. Hagenah, M. Geiger, and M. Merklein. Numerical simulation to improve robustness in sheet metal forming process chains. *AIP Conf. Proc.*, 1532(1):683 – 688, 2013.

[106] B.S. Yilbas and S.S. Akhtar. Laser bending of metal sheet and thermal stress analysis. *Opt. Laser Technol.*, 61:34 – 44, 2014.

[107] B.S. Yilbas, S.S. Akhtar, C. Karatas, H. Ali, K. Boran, M. Khaled, N. Al-Aqeeli, and A.B.J. Aleem. Laser treatment of aluminum composite and investigation of thermal stress field. *Int. J. Adv. Manuf. Tech.*, pages 1 – 15, 2016.

[108] M.B. Kadri, S. Nisar, S.Z. Khan, and W.A. Khan. Comparison of ANN and finite element model for the prediction of thermal stresses in diode laser cutting of float glass. *Optik - International Journal for Light and Electron Optics*, 126(19):1959 – 1964, 2015.

[109] S. Akhtar, O.O. Kardas, O. Keles, and B.S. Yilbas. Laser cutting of rectangular geometry into aluminum alloy: Effect of cut sizes on thermal stress field. *Opt. Laser Eng.*, 61:57 – 66, 2014.

[110] B.S. Yilbas, S.S. Akhtar, and C. Karatas. Laser cutting of rectangular geometry into alumina tiles. *Opt. Laser Eng.*, 55:35 – 43, 2014.

[111] S.S. Akhtar. Laser cutting of thick-section circular blanks: thermal stress prediction and microstructural analysis. *Int. J. Adv. Manuf. Tech.*, 71(5):1345 – 1358, 2014.

[112] B.S. Yilbas, S. Akhtar, and O. Keles. Laser cutting of triangular blanks from thick aluminum foam plate: Thermal stress analysis and morphology. *Appl. Therm. Eng.*, 62(1):28 – 36, 2014.

[113] I.A. Roberts, C.J. Wang, R. Esterlein, M. Stanford, and D.J. Mynors. A three-dimensional finite element analysis of the temperature field during laser melting of metal powders in additive layer manufacturing. *Int. J. Mach. Tool. Manu.*, 49(12-13):916 – 923, 2009.

[114] R. Akarapu, B.Q. Li, and A. Segall. A thermal stress and failure model for laser cutting and forming operations. *Journal of Failure Analysis and Prevention*, 4(5):51 – 62, 2004.

[115] K.Y. Nyon, C.Y. Nyeoh, M. Mokhtar, and R. Abdul-Rahman. Finite element analysis of laser inert gas cutting on inconel 718. *Int. J. Adv. Manuf. Tech.*, 60(9):995 – 1007, 2011.

[116] C.H. Fu, M.P. Sealy, Y.B. Guo, and X.T. Wei. Finite element simulation and experimental validation of pulsed laser cutting of nitinol. *Journal of Manufacturing Processes*, 19:81 – 86, 2015.

[117] M.F. Modest. Three-dimensional, transient model for laser machining of ablating/decomposing materials. *Int. J. Heat Mass Trans.*, 39(2):221 – 234, 1996.

[118] M.F. Modest. Laser through-cutting and drilling models for ablating/decomposing materials. *J. Laser Appl.*, 9(3):137 – 145, 1997.

[119] G. Han and S. Na. A study on torch path planning in laser cutting processes part 1: Calculation of heat flow in contour laser beam cutting. *J. Manuf. Syst.*, 18(2):54 – 61, 1999.

[120] W.J. Xu, J.C. Fang, X.Y. Wang, T. Wang, F. Liu, and Z.Y. Zhao. A numerical simulation of temperature field in plasma-arc forming of sheet metal. *J. Mater. Process. Tech.*, 164 - 165:1644 – 1649, 2005.

[121] M.J. Kim. Transient evaporative laser-cutting with boundary element method. *Appl. Math. Model.*, 25(1):25 – 39, 2000.

[122] M.J. Kim. Transient evaporative laser cutting with moving laser by boundary element method. *Appl. Math. Model.*, 28(10):891 – 910, 2004.

[123] K. Kheloufi, A.E. Hachemi, and A. Benzaoui. Numerical simulation of transient three-dimensional temperature and kerf formation in laser fusion cutting. *J. Heat Trans. - T. ASME*, 137(11), 2015.

[124] P. Yuan and D. Gu. Molten pool behaviour and its physical mechanism during selective laser melting of TiC/AlSi10Mg nanocomposites: simulation and experiments. *J. Phys. D. - Appl. Phys.*, 48(3):035303, 2015.

[125] Markus S Gross. On gas dynamic effects in the modelling of laser cutting processes. *Appl. Math. Model.*, 30(4):307 – 318, 2006.

[126] H. Boffy, M.C. Baietto, P. Sainsot, and A.A. Lubrecht. Detailed modelling of a moving heat source using multigrid methods. *Tribol. Int.*, 46(1):279 – 287, 2012.

[127] N. Gupta and N. Nataraj. A posteriori error estimates for an optimal control problem of laser surface hardening of steel. *Adv. Comput. Math.*, 39(1):69 – 99, 2013.

[128] M.F. Modest and H. Abakians. Evaporative cutting of a semi-infinite body with a moving CW laser. *J. Heat Trans. - T. ASME*, 108(3):602 – 607, 1986.

[129] K. Zimmer. Analytical solution of the laser-induced temperature distribution across internal material interfaces. *Int. J. Heat Mass Trans.*, 52(1-2):497 – 503, 2009.

[130] S. Mullick, Y.K. Madhukar, S. Roy, and A.K. Nath. An investigation of energy loss mechanisms in water-jet assisted underwater laser cutting process using an analytical model. *Int. J. Mach. Tool. Manu.*, 91:62 – 75, 2015.

[131] S. Mullick, Y.K. Madhukar, S. Roy, and A.K. Nath. Performance optimization of water-jet assisted underwater laser cutting of AISI 304 stainless steel sheet. *Opt. Laser Eng.*, 83:32 – 47, 2016.

[132] H.-J. Jiang and H.-L. Dai. Effect of laser processing on three dimensional thermodynamic analysis for HSLA rectangular steel plates. *Int. J. Heat Mass Trans.*, 82:98 – 108, 2015.

[133] J. Winczek. Analytical solution to transient temperature field in a half-infinite body caused by moving volumetric heat source. *Int. J. Heat Mass Trans.*, 53(25-26):5774 – 5781, 2010.

[134] Pedram Parandoush and Altab Hossain. A review of modeling and simulation of laser beam machining. *Int. J. Mach. Tool. Manu.*, 85:135 – 145, 2014.

[135] P. Di Pietro and Y.L. Yao. A numerical investigation into cutting front mobility in C02 laser cutting. *Int. J. Mach. Tool. Manu.*, 35(5):673 – 688, 1995.

[136] William M. Steen and Jyotirmoy Mazumder. *Laser Cutting, Drilling and Piercing*, pages 131 – 198. Springer London, London, 2010.

[137] A. D. Spence and Z. Li. Parallel processing for 2-1/2d machining simulation. In *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications*, SMA '01, pages 140–148, New York, NY, USA, 2001. ACM.

[138] Aitor Moreno, Álvaro Segura, Harbil Arregui, Jorge Posada, Álvaro Ruíz de Infante, and Natxo Canto. *Using 2D Contours to Model Metal Sheets in Industrial Machining Processes*, pages 135–149. Springer London, London, 2014.

[139] R. Dewil, P. Vansteenwegen, and D. Cattrysse. A review of cutting path algorithms for laser cutters. *The International Journal of Advanced Manufacturing Technology*, 87(5):1865 – 1884, 2016.

[140] B. Shi and H. Attia. Integrated process of laser-assisted machining and laser surface heat treatment. *J. Manuf. Sci. Eng.*, 135(6), 2013.

[141] B.S. Yilbas, S.S. Akhtar, and O. Keles. Laser cutting of aluminum foam: Experimental and model studies. *J. Manuf. Sci. Eng.*, 135(5), 2013.

[142] G. Han and S. Na. A study on torch path planning in laser cutting processes part 1: Calculation of heat flow in contour laser beam cutting. *J. Manuf. Syst.*, 18(2):54 – 61, 1999.

[143] N.S. Bailey, W. Tan, and Y.C. Shin. A parametric study on laser welding of magnesium alloy AZ31 by a fiber laser. *J. Manuf. Sci. Eng.*, 137(4), 2013.

[144] H.C. Kim, S.H. Lee, and D.Y. Yang. Toolpath planning algorithm for the ablation process using energy sources. *Computer-Aided Design*, 41(1):59 – 64, 2009.

[145] G.-C. Han and S.-J. Na. A study on torch path planning in laser cutting processes part 2: Cutting path optimization using simulated annealing. *Journal of Manufacturing Systems*, 18(2):62 – 70, 1999.

[146] Y. Kim, K. Gotoh, and M. Toyosada. Global cutting-path optimization considering the minimum heat effect with microgenetic algorithms. *Journal of Marine Science and Technology*, 9(2):70 – 79, 2004.

[147] Gorka Velez, Aitor Moreno, Álvaro Ruíz De Infante, and Raúl Chopitea. Real-time part detection in a virtually machined sheet metal defined as a set of disjoint regions. *International Journal of Computer Integrated Manufacturing*, 29(10):1089–1104, 2016.

[148] A. Wiesner and D. Schwarze. Multi-laser selective laser melting. In *8th International Conference on Photonic Technologies LANE 2014*, pages 1–3, 2014.

[149] S.Z. Shuja and B.S. Yilbas. Laser multi-beam heating of moving steel sheet: Thermal stress analysis. *Optics and Lasers in Engineering*, 51(4):446–452, 2013.

[150] F. Abe, K. Osakada, M. Shiomi, K. Uematsu, and M. Matsumoto. The manufacturing of hard tools from metallic powders by selective laser melting. *Journal of Materials Processing Technology*, 111(1):210–213, 2001. International symposium on advanced forming and die manufacturing technology.

[151] Jarno J. J. Kaakkunen, Petri Laakso, and Veli Kujanpää. Adaptive multibeam laser cutting of thin steel sheets with fiber laser using spatial light modulator. *Journal of Laser Applications*, 26(3):032008, 2014.

[152] T. Heeling, L. Zimmermann, and K. Wegener. Multi-beam strategies for the optimization of the selective laser melting process. In *Solid Freeform Fabrication 2016: proceedings of the 27th Annual International Solid Freeform Fabrication Symposium*, pages 1428–1438. The University of Texas at Austin, 2016.

[153] Flemming Ove Olsen, Klaus Schuett Hansen, and Jakob Skov Nielsen. Multibeam fiber laser cutting. *Journal of Laser Applications*, 21(3):133–138, 2009.

[154] S.S. Akhtar. Laser cutting of thick-section circular blanks: thermal stress prediction and microstructural analysis. *Int. J. Adv. Manuf. Tech.*, 71(5):1345–1358, 2014.

[155] S. Akhtar, O.O. Kardas, O. Keles, and B.S. Yilbas. Laser cutting of rectangular geometry into aluminum alloy: Effect of cut sizes on thermal stress field. *Opt. Laser Eng.*, 61:57–66, 2014.

[156] B.S. Yilbas, S.S. Akhtar, and C. Karatas. Laser cutting of rectangular geometry into alumina tiles. *Opt. Laser Eng.*, 55:35–43, 2014.

[157] B.S. Yilbas, S. Akhtar, and O. Keles. Laser cutting of triangular blanks from thick aluminum foam plate: Thermal stress analysis and morphology. *Appl. Therm. Eng.*, 62(1):28–36, 2014.

[158] K.Y. Nyon, C.Y. Nyeoh, M. Mokhtar, and R. Abdul-Rahman. Finite element analysis of laser inert gas cutting on inconel 718. *Int. J. Adv. Manuf. Tech.*, 60(9):995–1007, 2012.

[159] Thorsten Heeling and Konrad Wegener. Computational investigation of synchronized multi-beam strategies for the selective laser melting process. *Physics Procedia*, 83:899–908, 2016.

[160] Shahzada Zaman Shuja and Bekir Sami Yilbas. Multi-beam laser heating of steel: Temperature and thermal stress analysis. *Transactions of the Canadian Society for Mechanical Engineering*, 36(4):373–381, 2012.

[161] Verena Petzet, Christof Büskens, Hans Josef Pesch, Victor Karkhin, Maksym Makhutin, Andrey Prikhodovsky, and Vasily Ploshikhin. Optilas: Numerical optimization as a key tool for the improvement of advanced multi-beam laser welding techniques. In Arndt Bode and Franz Durst, editors, *High Performance Computing in Science and Engineering, Garching 2004*, pages 153–166, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[162] D. Mejia, A. Moreno, A. Arbelaiz, J. Posada, O. Ruiz-Salguero, and R. Chopitea. Accelerated thermal simulation for three-dimensional interactive optimization of computer numeric control sheet metal laser cutting. *J. Manuf. Sci. Eng.*, 140(3):031006–031006–9, 2017.

[163] B.S. Yilbas and S.S. Akhtar. Laser bending of metal sheet and thermal stress analysis. *Opt. Laser Technol.*, 61:34–44, 2014.

[164] Daniel Mejia-Parra, Ander Arbelaiz, Aitor Moreno, Jorge Posada, and Oscar Ruiz-Salguero. Fast spectral formulations of thin plate laser heating with gpu implementations. In *Proceedings of The 2nd International Conference on Mathematics and Computers in Science and Engineering (MACISE 2020)*, Madrid, Spain, TO BE PUBLISHED.

[165] Bekir Sami Yilbas, Sohail Akhtar, and Omer Keles. Laser cutting of triangular blanks from thick aluminum foam plate: Thermal stress analysis and morphology. *Applied Thermal Engineering*, 62(1):28 – 36, 2014.

[166] Sohail Akhtar, Omer Ozgur Kardas, Omer Keles, and Bekir Sami Yilbas. Laser cutting of rectangular geometry into aluminum alloy: Effect of cut sizes on thermal stress field. *Optics and Lasers in Engineering*, 61:57 – 66, 2014.

[167] B.S. Yilbas, S.S. Akhtar, and C. Karatas. Laser cutting of rectangular geometry into alumina tiles. *Optics and Lasers in Engineering*, 55:35 – 43, 2014.

[168] Syed Sohail Akhtar. Laser cutting of thick-section circular blanks: thermal stress prediction and microstructural analysis. *The International Journal of Advanced Manufacturing Technology*, 71(5):1345–1358, Mar 2014.

[169] Daniel Mejia-Parra, Jairo R. Sánchez, Oscar Ruiz-Salguero, Marcos Alonso, Alberto Izaguirre, Erik Gil, Jorge Palomar, and Jorge Posada. In-line dimensional inspection of warm-die forged revolution workpieces using 3d mesh reconstruction. *Applied Sciences*, 9(6), 2019.

[170] Daniel Mejia-Parra, Diego Montoya-Zapata, Ander Arbelaiz, Aitor Moreno, Jorge Posada, and Oscar Ruiz-Salguero. Fast analytic simulation for multi-laser heating of sheet metal in gpu. *Materials*, 11(11):2078, Oct 2018.

[171] Y. Ju and T. N. Farris. FFT Thermoelastic Solutions for Moving Heat Sources. *Journal of Tribology*, 119(1):156–162, 01 1997.

[172] Jean-Louis Dillenseger and Simon Esneault. Fast fft-based bioheat transfer equation computation. *Computers in Biology and Medicine*, 40(2):119 – 123, 2010.

[173] Stéphane Berbenni, Vincent Taupin, Komlan Sénam Djaka, and Claude Fressengeas. A numerical spectral approach for solving elasto-static field dislocation and g-disclination mechanics. *International Journal of Solids and Structures*, 51(23):4157 – 4175, 2014.

[174] Komlan Sénam Djaka, Aurélien Villani, Vincent Taupin, Laurent Capolungo, and Stéphane Berbenni. Field dislocation mechanics for heterogeneous elastic materials: A numerical spectral approach. *Computer Methods in Applied Mechanics and Engineering*, 315:921 – 942, 2017.

[175] Ran Ma and Timothy J. Truster. Fft-based homogenization of hypoelastic plasticity at finite strains. *Computer Methods in Applied Mechanics and Engineering*, 349:499 – 521, 2019.

[176] Chaitanya Paramatmuni and Anand K. Kanjarla. A crystal plasticity fft based study of deformation twinning, anisotropy and micromechanics in hcp materials: Application to az31 alloy. *International Journal of Plasticity*, 113:269 – 290, 2019.

[177] Jos Starn. A simple fluid solver based on the fft. *Journal of Graphics Tools*, 6(2):43–52, 2001.

[178] J. M. Taboada, L. Landesa, F. Obelleiro, J. L. Rodriguez, J. M. Bertolo, M. G. Araujo, J. C. Mouriño, and A. Gomez. High scalability fmm-fft electromagnetic solver for supercomputer systems. *IEEE Antennas and Propagation Magazine*, 51(6):20–28, Dec 2009.

[179] DOUGLAS F. ELLIOTT. Chapter 7 - fast fourier transforms. In Douglas F. Elliott, editor, *Handbook of Digital Signal Processing*, pages 527 – 631. Academic Press, San Diego, 1987.

[180] Ouarda Raaf and Abd El Hamid Adane. Pattern recognition filtering and bidimensional fft-based detection of storms in meteorological radar images. *Digital Signal Processing*, 22(5):734 – 743, 2012.

[181] DOUGLAS F. ELLIOTT. Chapter 1 - transforms and transform properties. In Douglas F. Elliott, editor, *Handbook of Digital Signal Processing*, pages 1 – 53. Academic Press, San Diego, 1987.

[182] Vladimir Britanak, Patrick C. Yip, and K.R. Rao. Chapter 1 - discrete cosine and sine transforms. In Vladimir Britanak, Patrick C. Yip, and K.R. Rao, editors, *Discrete Cosine and Sine Transforms*, pages 1 – 15. Academic Press, Oxford, 2007.

[183] Gorka Velez, Aitor Moreno, Álvaro Ruíz De Infante, and Raúl Chopitea. Real-time part detection in a virtually machined sheet metal defined as a set of disjoint regions. *International Journal of Computer Integrated Manufacturing*, 29(10):1089–1104, 2016.

[184] Yusuf Sahillioğlu and Ladislav Kavan. Skuller: A volumetric shape registration algorithm for modeling skull deformities. *Medical Image Analysis*, 23(1):15–27, 2015.

[185] Jairo R. Sánchez, Álvaro Segura, and Iñigo Barandiaran. Fast and accurate mesh registration applied to in-line dimensional inspection processes. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 12(3):877–887, Aug 2018.

[186] Kay Böhnke and Achim Gottscheber. Fast object registration and robotic bin picking. In Achim Gottscheber, David Obdržálek, and Colin Schmidt, editors, *Research and Education in Robotics - EUROBOT 2009*, pages 23–37, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[187] François Pomerleau, Francis Colas, and Roland Siegwart. A review of point cloud registration algorithms for mobile robotics. *Found. Trends Robot*, 4(1):1–104, May 2015.

[188] G. K. L. Tam, Z. Cheng, Y. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X. Sun, and P. L. Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, July 2013.

[189] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992.

[190] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4):629–642, Apr 1987.

[191] G. C. Sharp, S. W. Lee, and D. K. Wehe. Icp registration using invariant features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):90–102, Jan 2002.

[192] S. M. Yamany and A. A. Farag. Surface signatures: an orientation independent free-form surface representation scheme for the purpose of objects registration and matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1105–1120, Aug 2002.

[193] Huayi Wu, Xuefeng Guan, and Jianya Gong. Parastream: A parallel streaming delaunay triangulation algorithm for lidar points on multicore architectures. *Computers & Geosciences*, 37(9):1355–1363, 2011.

[194] Timothée Jost and Heinz Hügli. Fast icp algorithms for shape registration. In Luc Van Gool, editor, *Pattern Recognition*, pages 91–99, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[195] Jun Gong, Qing Zhu, Ruofei Zhong, Yeting Zhang, and Xiao Xie. An efficient point cloud management method based on a 3d r-tree. *Photogrammetric Engineering & Remote Sensing*, 78(4):373–381, 2012.

[196] Jan Elseberg, Dorit Borrmann, and Andreas Nüchter. One billion points in the cloud – an octree for efficient processing of 3d laser scans. *ISPRS Journal of Photogrammetry and Remote Sensing*, 76:76–88, 2013. Terrestrial 3D modelling.

[197] Bertram Drost and Slobodan Ilic. A hierarchical voxel hash for fast 3d nearest neighbor lookup. In Joachim Weickert, Matthias Hein, and Bernt Schiele, editors, *Pattern Recognition*, pages 302–312, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[198] Bertram H. Drost and Slobodan Ilic. Almost constant-time 3d nearest-neighbor lookup using implicit octrees. *Machine Vision and Applications*, 29(2):299–311, Feb 2018.

[199] Ping Yan and Kevin W. Bowyer. A fast algorithm for icp-based 3d shape biometrics. *Computer Vision and Image Understanding*, 107(3):195–202, 2007.

[200] D. Fontanelli, L. Ricciato, and S. Soatto. A fast ransac-based registration algorithm for accurate localization in unknown environments using lidar measurements. In *2007 IEEE International Conference on Automation Science and Engineering*, pages 597–602, Sep. 2007.

[201] Cihan Ulas and Hakan Temeltas. A 3d scan matching method based on multi-layered normal distribution transform. *IFAC Proceedings Volumes*, 44(1):11602–11607, 2011. 18th IFAC World Congress.

[202] Liang Cheng, Song Chen, Xiaogiang Liu, Hao Xu, Yang Wu, Manchun Li, and Yanming Chen. Registration of laser scanning point clouds: A review. *Sensors (Basel)*, 18(5):1641:1–1641:25, May 2018.

[203] Sylvain Lefebvre and Hugues Hoppe. Perfect spatial hashing. *ACM Trans. Graph.*, 25(3):579–588, July 2006.

[204] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM.

[205] Marek Hawryluk and Jacek Ziemba. Possibilities of application measurement techniques in hot die forging processes. *Measurement*, 110:284–295, 2017.

[206] Z. Gronostajski, M. Kaszuba, M. Hawryluk, and M. Zwierzchowski. A review of the degradation mechanisms of the hot forging tools. *Archives of Civil and Mechanical Engineering*, 14(4):528–539, 2014.

[207] Marek Hawryluk, Zbigniew Gronostajski, Marcin Kaszuba, Sławomir Polak, Paweł Widomski, Jacek Ziemba, and Jerzy Smolik. Application of selected surface engineering methods to improve the durability of tools used in precision forging. *The International Journal of Advanced Manufacturing Technology*, 93(5):2183–2200, Nov 2017.

[208] International Organization for Standardization. Iso 1101:2017 geometrical product specifications (gps) — geometrical tolerancing — tolerances of form, orientation, location and run-out. Standard, International Organization for Standardization, 2017.

[209] Marek Hawryluk, Jacek Ziemba, and Przemyslaw Sadowski. A review of current and new measurement techniques used in hot die forging processes. *Measurement and Control*, 50(3):74–86, 2017.

[210] Georg Henzold. 18 - inspection of geometrical deviations. In *Geometrical Dimensioning and Tolerancing for Design, Manufacturing and Inspection (Second Edition)*, pages 160–254. Butterworth-Heinemann, Oxford, second edition edition, 2006.

[211] Jairo R. Sánchez, Álvaro Segura, and Iñigo Barandiaran. Fast and accurate mesh registration applied to in-line dimensional inspection processes. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 12(3):877–887, Aug 2018.

[212] Paolo Minetola. The importance of a correct alignment in contactless inspection of additive manufactured parts. *International Journal of Precision Engineering and Manufacturing*, 13(2):211–218, Feb 2012.

[213] Q. Shi and N. Xi. Automated data processing for a rapid 3d surface inspection system. In *2008 IEEE International Conference on Robotics and Automation*, pages 3939–3944, May 2008.

[214] Liang Zhu, Jacob Barhak, Vijay Srivatsan, and Reuven Katz. Efficient registration for precision inspection of free-form surfaces. *The International Journal of Advanced Manufacturing Technology*, 32(5):505–515, Mar 2007.

[215] B. Gapinski, M. Wieczorowski, L. Marciniak-Podsadna, B. Dybala, and G. Ziolkowski. Comparison of different method of measurement geometry using cmm, optical scanner and computed tomography 3d. *Procedia Engineering*, 69:255 – 262, 2014.

[216] Zbigniew Gronostajski, Marek Hawryluk, Marcin Kaszuba, Paweł Widomski, and Jacek Ziemba. Application of the reverse 3d scanning method to evaluate the wear of forging tools divided on two selected areas. *International Journal of Automotive Technology*, 18(4):653–662, Aug 2017.

[217] Marek Hawryluk and Jacek Ziemba. Application of the 3d reverse scanning method in the analysis of tool wear and forging defects. *Measurement*, 128:204–213, 2018.

[218] K. H. Jung, S. Lee, Y. B. Kim, B. Ahn, E. Z. Kim, and G. A. Lee. Assessment of zk60a magnesium billets for forging depending on casting methods by upsetting and tomography. *Journal of Mechanical Science and Technology*, 27(10):3149–3153, Oct 2013.

[219] Antonello D'Annibale, Antoniomaria Di Ilio, Michele Trozzi, and Luigi Bonaventura. The use of infrared thermography for maintenance purposes in the production process of components for automotive alternators. *Procedia CIRP*, 38:143–146, 2015. Proceedings of the 4th International Conference on Through-life Engineering Services.

[220] K. T. Fendt, H. Mooshofer, S. J. Rupitsch, and H. Ermert. Ultrasonic defect characterization in heavy rotor forgings by means of the synthetic aperture focusing technique and optimization methods. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 63(6):874–885, June 2016.

[221] K. Ashok Reddy. Non-destructive testing, evaluation of stainless steel materials. *Materials Today: Proceedings*, 4(8):7302–7312, 2017. International Conference on Advancements in Aeromechanical Materials for Manufacturing (ICAAMM-2016): Organized by MLR Institute of Technology, Hyderabad, Telangana, India.

[222] S.B. Dworkin and T.J. Nye. Image processing for machine vision measurement of hot formed parts. *Journal of Materials Processing Technology*, 174(1):1–6, 2006.

[223] Zhenyuan Jia, Bangguo Wang, Wei Liu, and Yuwen Sun. An improved image acquiring method for machine vision measurement of hot formed parts. *Journal of Materials Processing Technology*, 210(2):267–271, 2010.

[224] Yu cun Zhang, Jun xia Han, Xian bin Fu, and Fu li Zhang. Measurement and control technology of the size for large hot forgings. *Measurement*, 49:52–59, 2014.

[225] Yueyang Du Zhengchun Du. Simple three-dimensional laser radar measuring method and model reconstruction for hot heavy forgings. *Optical Engineering*, 51:51–51–8, 2012.

[226] Zhengchun Du, Zhaoyong Wu, and Jianguo Yang. 3d measuring and segmentation method for hot heavy forging. *Measurement*, 85:43–53, 2016.

[227] Wei Liu, Zhenyuan Jia, Fuji Wang, Xin Ma, Wenqiang Wang, Xinghua Jia, and Di Song. An improved online dimensional measurement method of large hot cylindrical forging. *Measurement*, 45(8):2041–2051, 2012.

[228] Julio Molleda, Rubén Usamentiaga, Daniel F. García, and Francisco G. Bulnes. Real-time flatness inspection of rolled products based on optical laser triangulation and three-dimensional surface reconstruction. *Journal of Electronic Imaging*, 19:031206:1–031206:14, 2010.

[229] Manoj Babu, Pasquale Franciosa, and Darek Ceglarek. Adaptive measurement and modelling methodology for in-line 3d surface metrology scanners. *Procedia CIRP*, 60:26–31, 2017. Complex Systems Engineering and Development Proceedings of the 27th CIRP Design Conference Cranfield University, UK 10th – 12th May 2017.

[230] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4):629–642, Apr 1987.

[231] Cong Sun, Haibo Liu, Mengna Jia, and Shengyi Chen. Review of calibration methods for scheimpflug camera. *Journal of Sensors*, pages 3901431:1–3901431:15, 2018.

[232] Carsten Steger, Markus Ulrich, and Christian Wiedemann. *Machine Vision Algorithms and Applications*. Wiley-VCH, second completely revised and enlarged edition edition, 11 2017.

[233] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[234] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992.

[235] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE International Conference on Robotics and Automation*, pages 1–4, May 2011.

[236] Bill McNeese. Anova gage r&r - part 2. `https://www.spcforexcel.com/knowledge/measurement-systems-analysis/anova-gage-rr-part-2`, 2012. Accessed: 2019-02-04.

[237] J.R. Sánchez, A. Segura, I. Barandiaran, J. Muñoz, and J.A. Larrea. Dimensional inspection of manufactured parts with web-based 3d visualization. In *International Virtual Concept Workshop on INDUSTRIE 4.0*, Donostia-San Sebastián, Gipuzkoa, Spain, 2015.

[238] R. Richter and J. Dollner. Concepts and techniques for integration, analysis and visualization of massive 3d point clouds. *Computers, Environment and Urban Systems*, 45:114 – 124, 2014.

[239] I.-H. Song and S.-C. Chung. Data format and browser of lightweight cad files for dimensional verification over the internet. *Journal of Mechanical Science and Technology*, 23(5):1278, 2009.

[240] B. Muralikrishnan, J. Raja, and K.R. Subramanian. A 3d visualization tool for surface metrology. In *Proceedings of the Second National Conference in Precision Engineering*, Coimbatore, India, 2002.

[241] H. Zhang and L. Wang. Application and study of 3d alignment technology for the inspection of automobile punching parts. In *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*, volume 1, pages 264–267, 2011.

[242] R. Minguez, A. Arias, O. Etxaniz, E. Solaberrieta, and L. Barrenetxea. Framework for verification of positional tolerances with a 3d non-contact measurement method. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 10(2):85 – 93, 2016.

[243] B. Salski, W. Gwarek, and P. Korpas. Non-destructive testing of carbon-fiber-reinforced polymer composites with coupled spiral inductors. In *2014 IEEE MTT-S International Microwave Symposium (IMS2014)*, pages 1 – 4, June 2014.

[244] J. Behr, Y. Jung, T. Franke, and T. Sturm. Using images and explicit binary container for efficient and incremental delivery of declarative 3d scenes on the web. In *Proceedings of the 17th International Conference on 3D Web Technology*, Web3D '12, pages 17 – 25, New York, NY, USA, 2012. ACM.

[245] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *Proceedings Third International Conference on IEEE 3-D Digital Imaging and Modeling*, pages 145 – 152, 2001.

[246] D. Mejia, O. Ruiz-Salguero, and C.A. Cadavid. Hessian eigenfunctions for triangular mesh parameterization. In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 1: GRAPP,*, pages 75–82, 2016.

[247] Bala R. Vatti. A generic solution to polygon clipping. *Commun. ACM*, 35(7):56–63, July 1992.