Technische Universität München

BGCE Project: CAD – Integrated Topology Optimization

BGCE First Milestone Meeting

S. Joshi, J.C. Medina, F. Menhorn, S. Reiz, *B. Rüth*, *E. Wannerberg*, A. Yurova

August 6, 2015

Contents

1. Introduction
   - 1.1 Contents
   - 1.2 Motivation
   - 1.3 Workflow Overview

2. Schedule & Milestones

3. CAD to Optimized Surface
   - 3.1 CAD To STL
   - 3.2 STL To Voxels
   - 3.3 Topology Optimization
   - 3.4 Surface Extraction
   - 3.5 Short Summary

4. Optimized Surface to CAD
   - 4.1 B–Spline Fitting

5. Summary

6. Outlook

Contents

1. Introduction

1.1 Contents
1.2 Motivation
1.3 Workflow Overview

2. Schedule & Milestones

3. CAD to Optimized Surface

3.1 CAD To STL
3.2 STL To Voxels
3.3 Topology Optimization
3.4 Surface Extraction
3.5 Short Summary

4. Optimized Surface to CAD

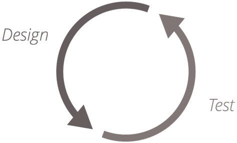
4.1 B–Spline Fitting

5. Summary

6. Outlook

Motivation

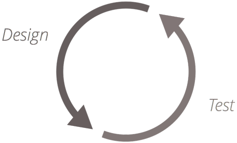
Current Design Process:



- Iterative and redundant
- Time consuming

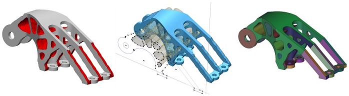
Motivation

Current Design Process:



- Iterative and redundant
- Time consuming

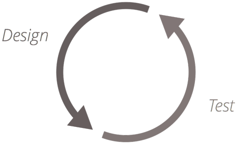
Topology optimization



- Promoted by additive manufacturing

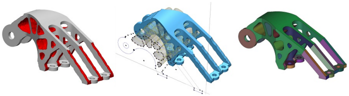
Motivation

Current Design Process:



- Iterative and redundant
- Time consuming

Topology optimization



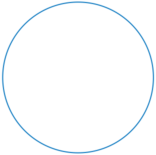
- Promoted by additive manufacturing

Focus:

Convert optimized geometry to **lightweight** and **scalable** CAD formats

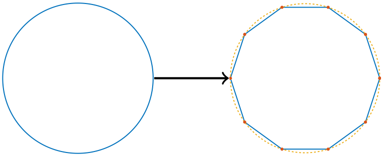
Workflow Overview

CAD design



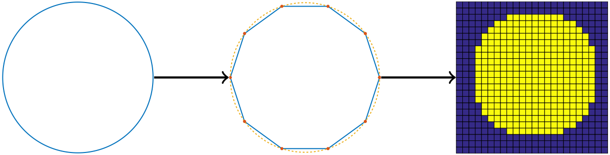
Workflow Overview

STL Interface



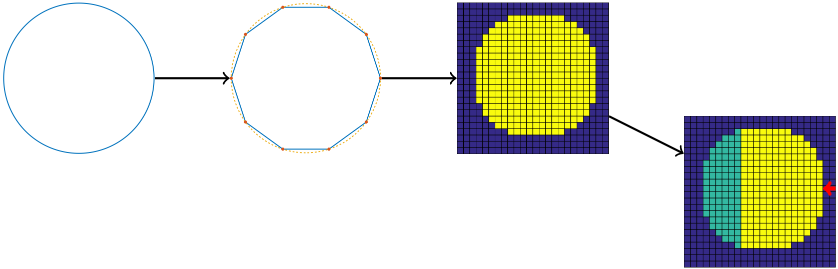
Workflow Overview

Voxelized topology



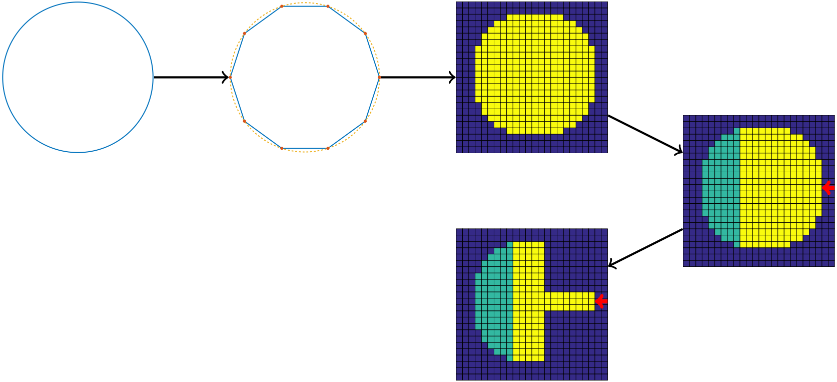
Workflow Overview

Specification of loads and fixtures



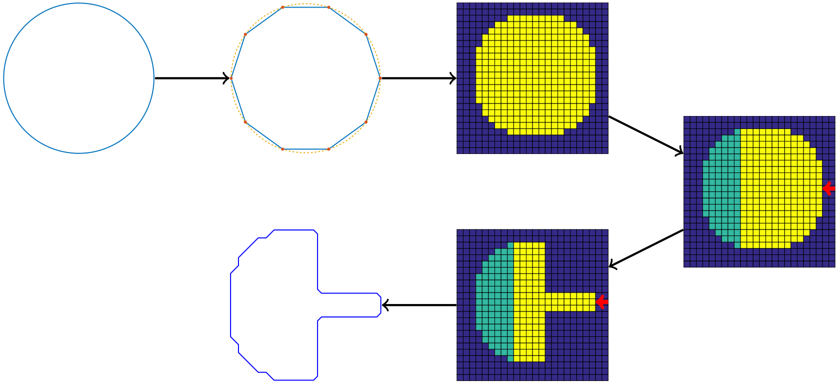
Workflow Overview

Optimized topology



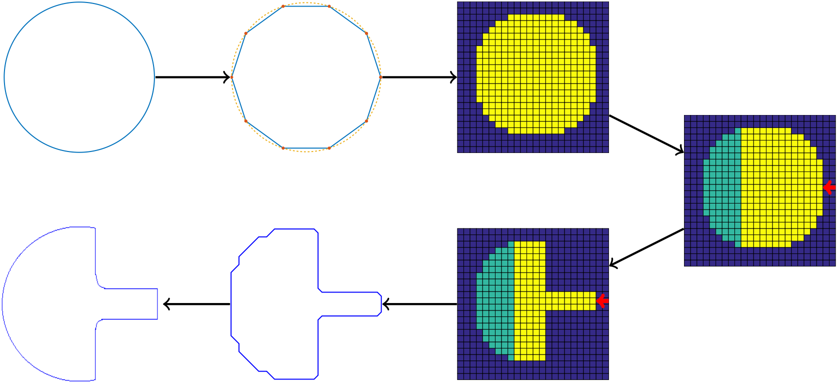
Workflow Overview

Surface extraction

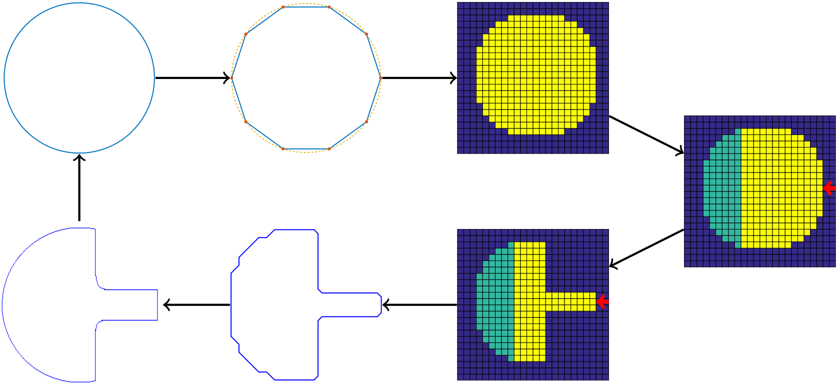


Workflow Overview

Parametrized CAD-geometries



Workflow Overview

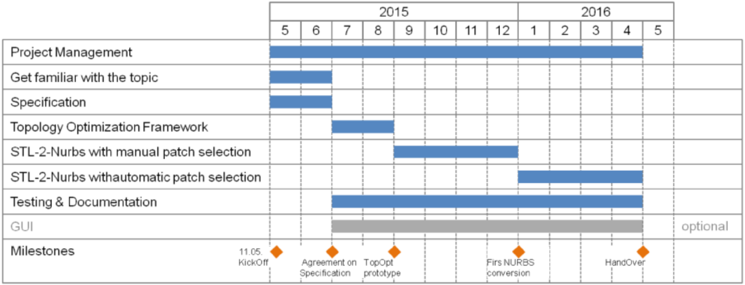


Contents

1. **Introduction**
   - 1.1 Contents
   - 1.2 Motivation
   - 1.3 Workflow Overview
2. **Schedule & Milestones**
3. **CAD to Optimized Surface**
   - 3.1 CAD To STL
   - 3.2 STL To Voxels
   - 3.3 Topology Optimization
   - 3.4 Surface Extraction
   - 3.5 Short Summary
4. **Optimized Surface to CAD**
   - 4.1 B–Spline Fitting
5. **Summary**
6. **Outlook**

Schedule & Milestones

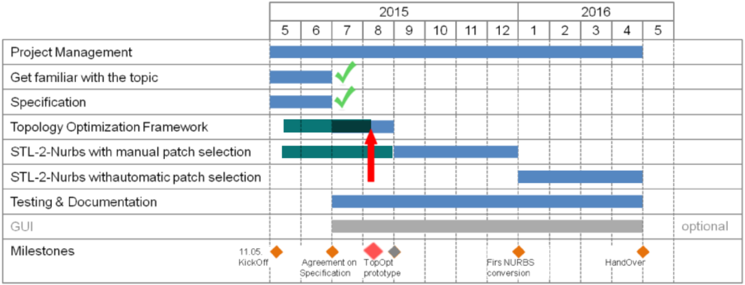
Schedule:



	2015								2016					
	5	6	7	8	9	10	11	12	1	2	3	4	5	
Project Management	■	■	■	■	■	■	■	■	■	■	■	■		
Get familiar with the topic	■	■												
Specification	■	■												
Topology Optimization Framework			■	■										
STL-2-Nurbs with manual patch selection					■	■	■	■						
STL-2-Nurbs withautomatic patch selection									■	■	■	■		
Testing & Documentation			■	■	■	■	■	■	■	■	■	■		
GUI			▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		optional
Milestones	11.05. KickOff ◆	◆ Agreement on Specification		◆ TopOpt prototype				◆ Firs NURBS conversion				◆ HandOver		

Schedule & Milestones

Schedule: (current)



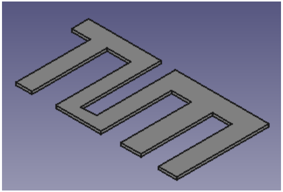
Contents

1. Introduction
   - 1.1 Contents
   - 1.2 Motivation
   - 1.3 Workflow Overview
2. Schedule & Milestones
3. CAD to Optimized Surface
   - 3.1 CAD To STL
   - 3.2 STL To Voxels
   - 3.3 Topology Optimization
   - 3.4 Surface Extraction
   - 3.5 Short Summary
4. Optimized Surface to CAD
   - 4.1 B–Spline Fitting
5. Summary
6. Outlook

CAD to STL

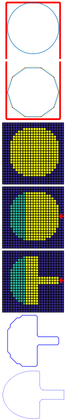
Tools:

- Create original CAD geometry in CAD program



Interface:

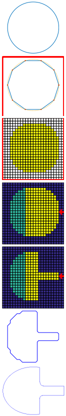
- Current approach: Export to STL directly.

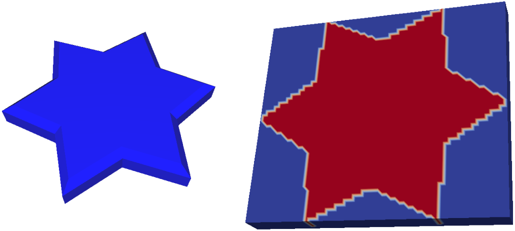


From STL To Voxels

Tools:

- Common Versatile Multi-purpose Library for C++ (CVMLCPP)
  - Converts STL format voxels of specified size (binary file)
- Custom script to read binary file and output it as ascii.vtk for visualisation

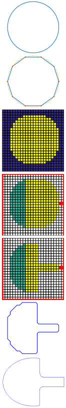


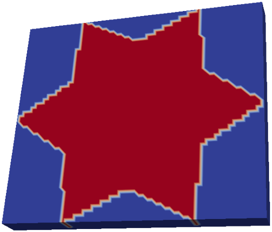


Topology Optimization

Tools:

- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
  - Reads voxelized data and generates ToPy input
  - Non-voxel cells set to passive elements
  - Boundary conditions added manually

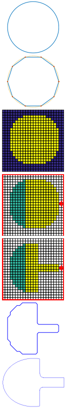


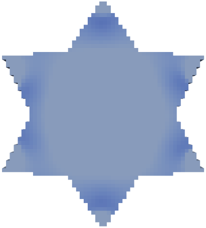


Topology Optimization

Tools:

- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
  - Reads voxelized data and generates ToPy input
  - Non-voxel cells set to passive elements
  - Boundary conditions added manually

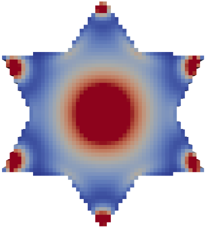


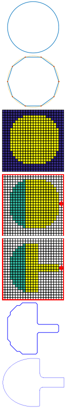


Topology Optimization

Tools:

- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
  - Reads voxelized data and generates ToPy input
  - Non-voxel cells set to passive elements
  - Boundary conditions added manually

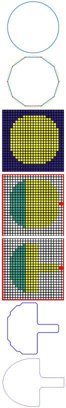


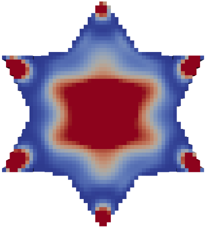


Topology Optimization

Tools:

- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
  - Reads voxelized data and generates ToPy input
  - Non-voxel cells set to passive elements
  - Boundary conditions added manually

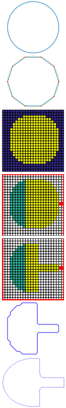


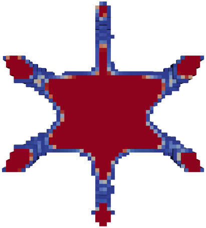


Topology Optimization

Tools:

- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
  - Reads voxelized data and generates ToPy input
  - Non-voxel cells set to passive elements
  - Boundary conditions added manually

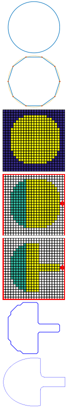


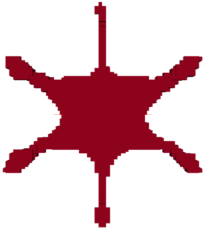


Topology Optimization

Tools:

- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
  - Reads voxelized data and generates ToPy input
  - Non-voxel cells set to passive elements
  - Boundary conditions added manually

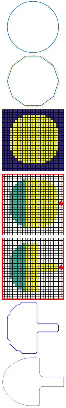


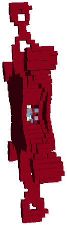


Topology Optimization

Tools:

- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
    - Reads voxelized data and generates ToPy input
    - Non-voxel cells set to passive elements
    - Boundary conditions added manually

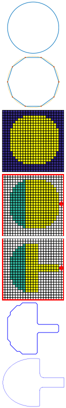


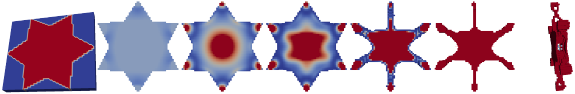


Topology Optimization

Tools:

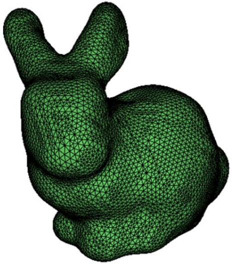
- ToPy: Open-Source SIMP/FEM topology optimizer in python
- Custom script for generating ToPy input file (.tpd)
  - Reads voxelized data and generates ToPy input
  - Non-voxel cells set to passive elements
  - Boundary conditions added manually

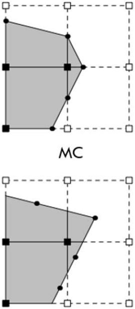




From Voxel to Mesh Geometry

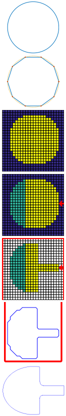
- Extract isosurface from voxel information
- Algorithms: Marching Cubes, Dual Contouring, Extended Models
- Implementations in VTK library





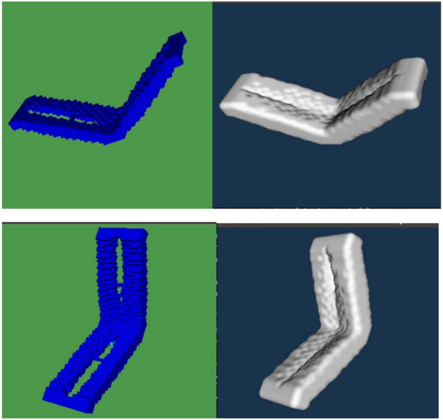
MC

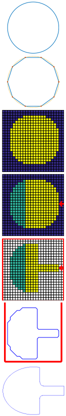
Dual method



Surface Extraction

Contour Filtering using Implicit Modelling

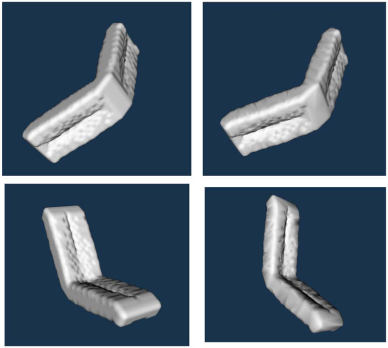


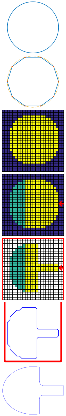


Problem: Holes are not taken into account

Decimation

- Fine mesh to a coarser mesh through Decimation ↔ Reduction of number of triangles. (Upper: 50% Lower: 90%)
- Smoothing step is needed in between





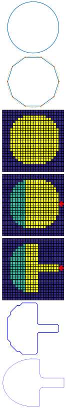
Short Summary

Direct interaction with CAD formats (STEP)

- Open-Source alternatives: OpenCascade...

Boundary conditions required - how to specify?

- Current state: Manual specification
- Extract metadata from CAD formats, extra voxelized files..



Contents

1. **Introduction**
   - 1.1 Contents
   - 1.2 Motivation
   - 1.3 Workflow Overview
2. **Schedule & Milestones**
3. **CAD to Optimized Surface**
   - 3.1 CAD To STL
   - 3.2 STL To Voxels
   - 3.3 Topology Optimization
   - 3.4 Surface Extraction
   - 3.5 Short Summary
4. **Optimized Surface to CAD**
   - 4.1 B–Spline Fitting
5. **Summary**
6. **Outlook**

Current Status

- What do we have so far?

Current Status

- What do we have so far?
- What if we try to pass it to an engineer?

How to make CAD understand our data?

B–Spline

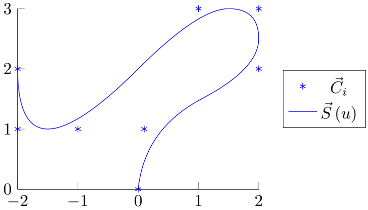
$$\vec{S}(u,v) = \sum_{i,j=1}^{n,m} \vec{C}_{i,j} N_i^p(u)\, N_j^p(v)\,,$$

where p – degree of the B–Spline surface and n, m – number of control points in each direction.

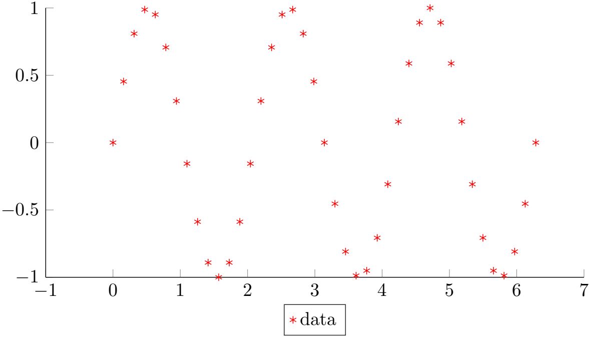
B–Splines

- offer great flexibility for handling arbitrary shapes
- are CAD–standard

Engineers are working with CAD



B–Spline Fitting

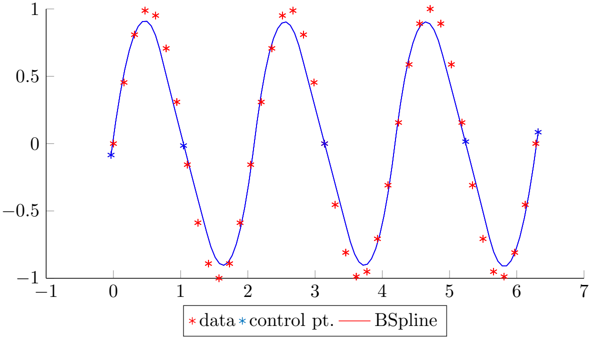


Goal:

Find B–Spline representation of our data!

$$\vec{S}\,(u_\alpha, v_\alpha) \approx \vec{P}_\alpha$$

B–Spline Fitting



Goal:

Find B–Spline representation of our data!

$$\vec{S}\left(u_\alpha, v_\alpha\right) \approx \vec{P}_\alpha$$

B–Spline Fitting: Least Squares

The task:

Find control points $C_{i,j}$, such that the B–Spline surface

$$\vec{S}\left(u,v\right)=\sum_{i,j=1}^{n,m}\vec{C}_{i,j}N_i^p\left(u\right)N_j^p\left(v\right)$$

approximates our dataset of points $\left\{\vec{P}_\alpha\right\}$.

This leads to *minimization problem*:

$$\vec{S}\left(u_\alpha,v_\alpha\right)\approx\vec{P}_\alpha\forall\alpha\leftrightarrow\min_{\vec{C}_{i,j}\in\mathbb{R}^3}\sum_\alpha\|\vec{P}_\alpha-\vec{S}\left(u_\alpha,v_\alpha\right)\|_2$$

B–Spline Fitting: Least Squares (cont.)

Resulting system looks like:

$$\sum_{i,j=1}^{n,m} \vec{C}_{i,j} N_i^p \left(u_\alpha\right) N_j^p \left(v_\alpha\right) \approx \vec{P}_\alpha \quad \forall \alpha$$

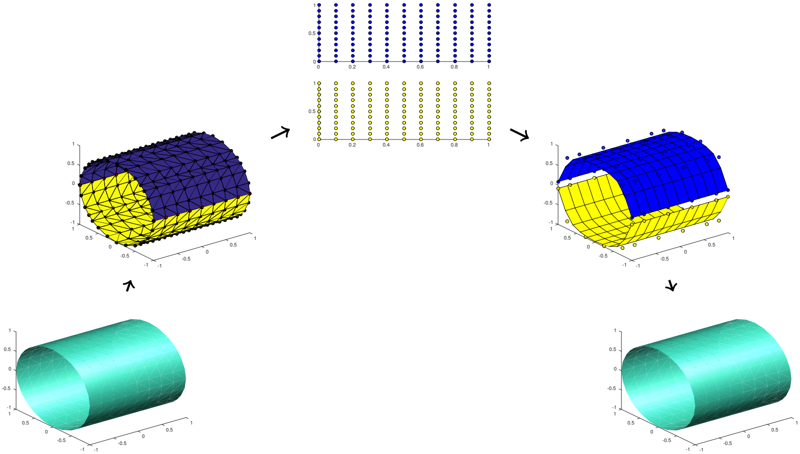
Or, in matrix–vector form:

$$AC \approx P$$

Our system matrix A **depends on** $\{u_\alpha, v_\alpha\}$

B–Spline Fitting Pipeline [Becker, Schäfer, Jameson]

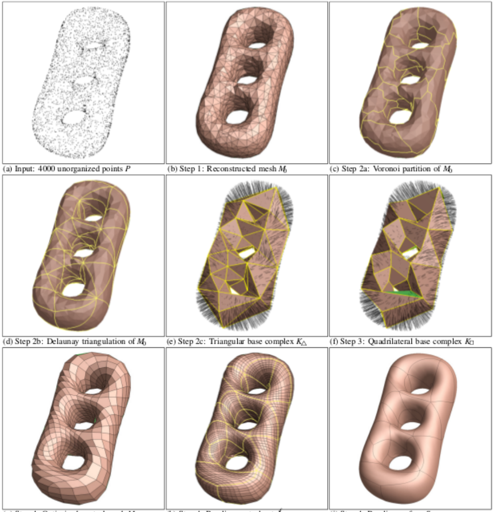
How to deal with a complex shape?



B–Spline Fitting: Open Questions

- How to distribute our data into patches?
- How to parameterize obtained patches?
- How to connect several patches after fitting?

B–Spline Fitting Pipeline [M. Eck & H. Hoppe]



(a) Input: 4000 unorganized points P

(b) Step 1: Reconstructed mesh M_0

(c) Step 2a: Voronoi partition of M_0

(d) Step 2b: Delaunay triangulation of M_0

(e) Step 2c: Triangular base complex $K_\triangle$

(f) Step 3: Quadrilateral base complex $K_\square$

(g) Step 4: Optimized control mesh M_X

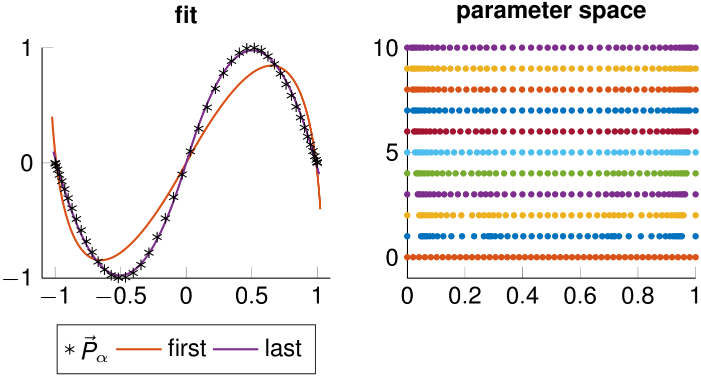
(h) Step 4: B-spline control net $\mathbf{d}_{j,s}$

(i) Step 4: B-spline surface S

B–Spline Fitting: Parameter Correction

The task:

For *fixed* control points $C_{i,j}$, find an optimal parametrization $\{u_\alpha, v_\alpha\}$.



Contents

1. Introduction
   1.1 Contents
   1.2 Motivation
   1.3 Workflow Overview
2. Schedule & Milestones
3. CAD to Optimized Surface
   3.1 CAD To STL
   3.2 STL To Voxels
   3.3 Topology Optimization
   3.4 Surface Extraction
   3.5 Short Summary
4. Optimized Surface to CAD
   4.1 B–Spline Fitting
5. **Summary**
6. Outlook

What is done?

- First part of the pipeline from CAD model to optimized voxel model:
  - ✓ CAD to STL with e.g. FreeCAD
  - ✓ STL to Voxels with CVMLCPP
  - ✓ Voxels to ToPy input with custom script
  - ✓ Topology optimized geometry with ToPy
  - 🕐 Surface reconstruction with VTKToolbox
- B–spline fitting
  - ✗ Automatic patch selection
  - ✗ Parametrization of obtained patches
  - ✓ B–spline fitting using least squares
  - 🕐 Smooth connection of patches
  - ✗ Conversion back to CAD

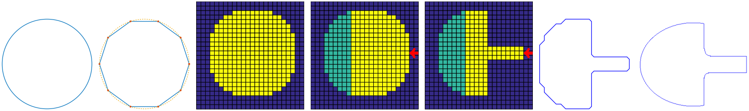
Contents

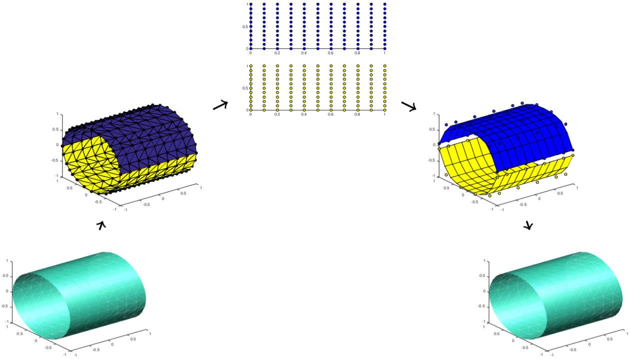
1. Introduction
    - 1.1 Contents
    - 1.2 Motivation
    - 1.3 Workflow Overview
2. Schedule & Milestones
3. CAD to Optimized Surface
    - 3.1 CAD To STL
    - 3.2 STL To Voxels
    - 3.3 Topology Optimization
    - 3.4 Surface Extraction
    - 3.5 Short Summary
4. Optimized Surface to CAD
    - 4.1 B–Spline Fitting
5. Summary
6. **Outlook**

What is next?

- Automation of the first part of the pipeline
- Integration of boundary conditions handling
- Implementation of remaining B–spline fitting steps (based on work of M.Eck & H.Hoppe)
- Further research on algorithms considering voxel geometry

Thank you for your attention!





Literature

- **William Hunter.** ”Predominantly solid-void three-dimensional topology optimisation using open source software”
- **Gerrit Becker, Michael Schäfer, Antony Jameson.** ”An advanced NURBS fitting procedure for post-processing of grid-based shape optimizations”
- **Matthias Eck, Hugues Hoppe.** ”Automatic Reconstruction of B-Spline Surfaces of Arbitrary Topological Type”