Constrained Space Deformation for Design Optimization

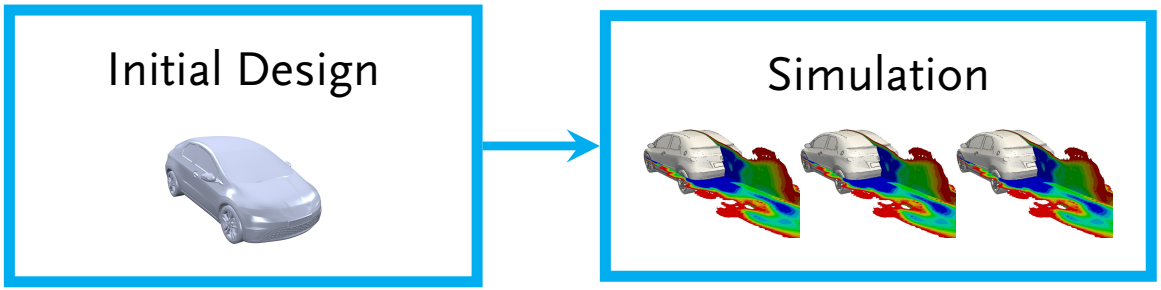
Daniel Sieger[1] Stefan Menzel[2] Mario Botsch[1]

[1]Graphics & Geometry Group, Bielefeld University [2]Honda Research Institute Europe

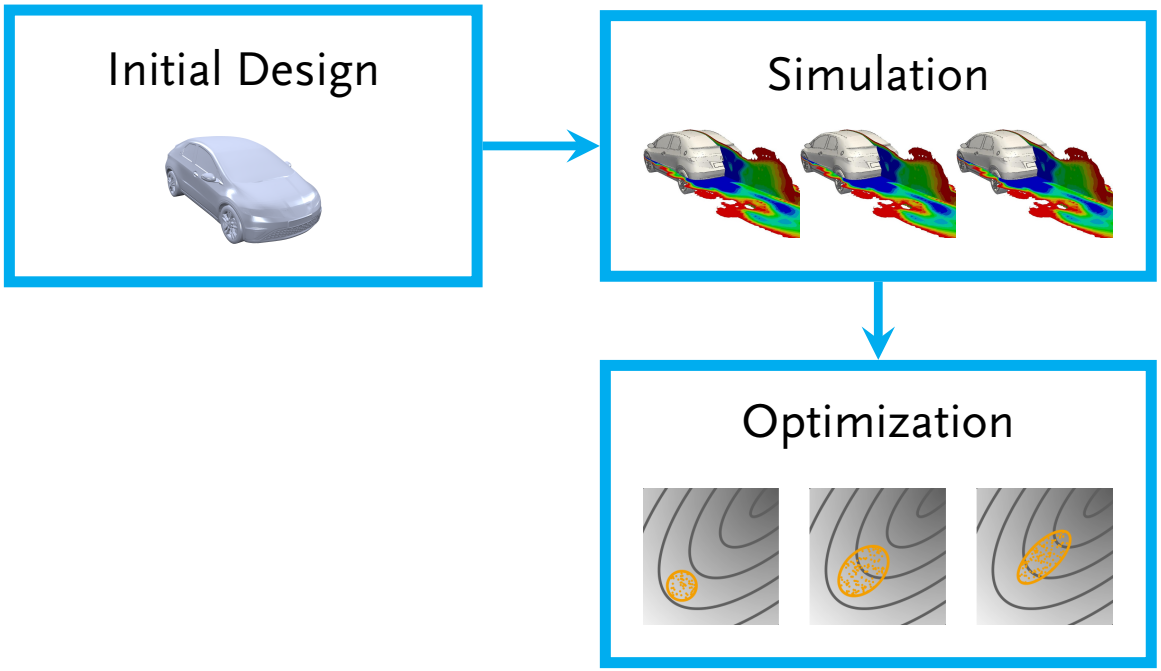
Shape Deformation & Design Optimization

Initial Design

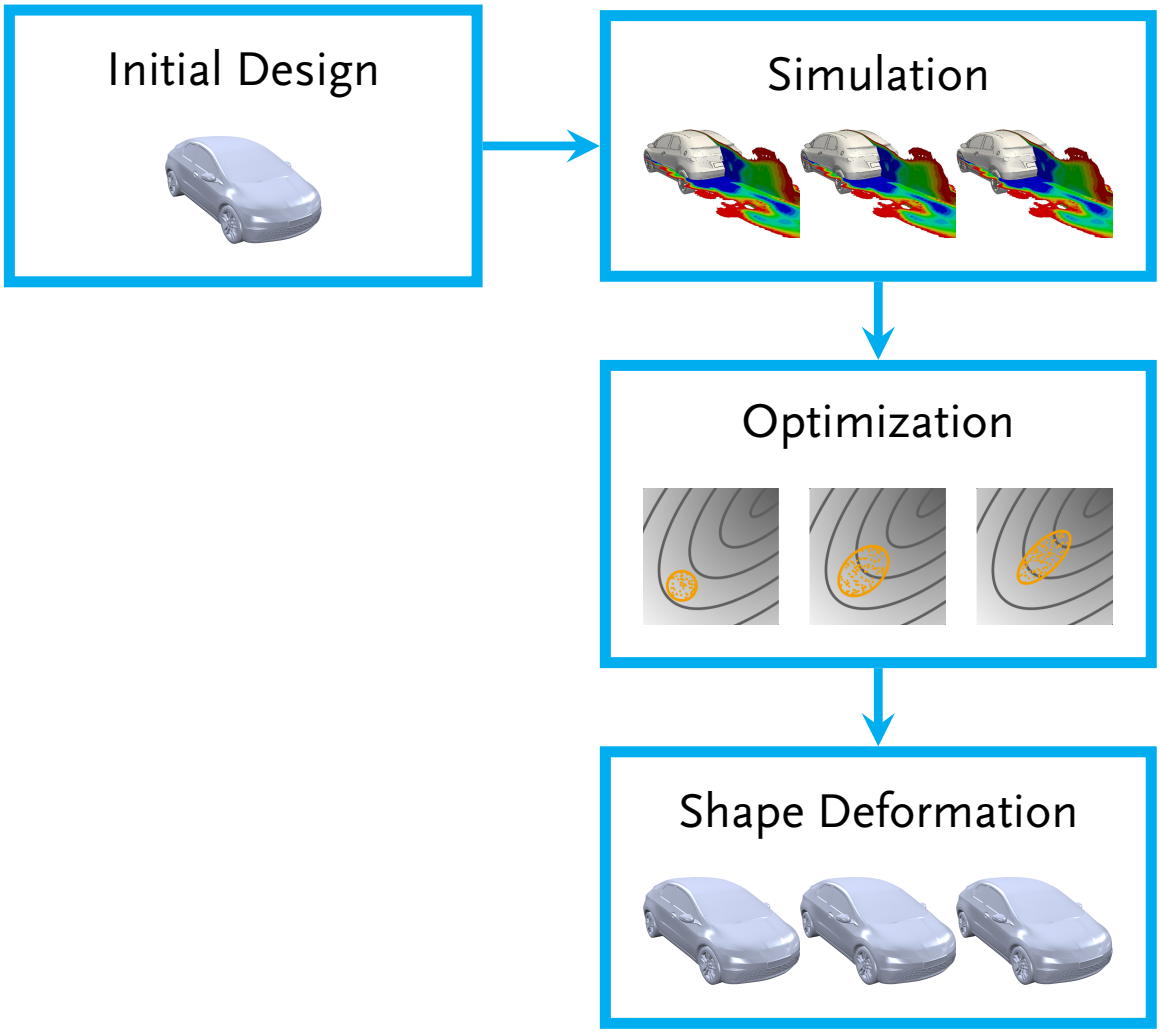
Shape Deformation & Design Optimization



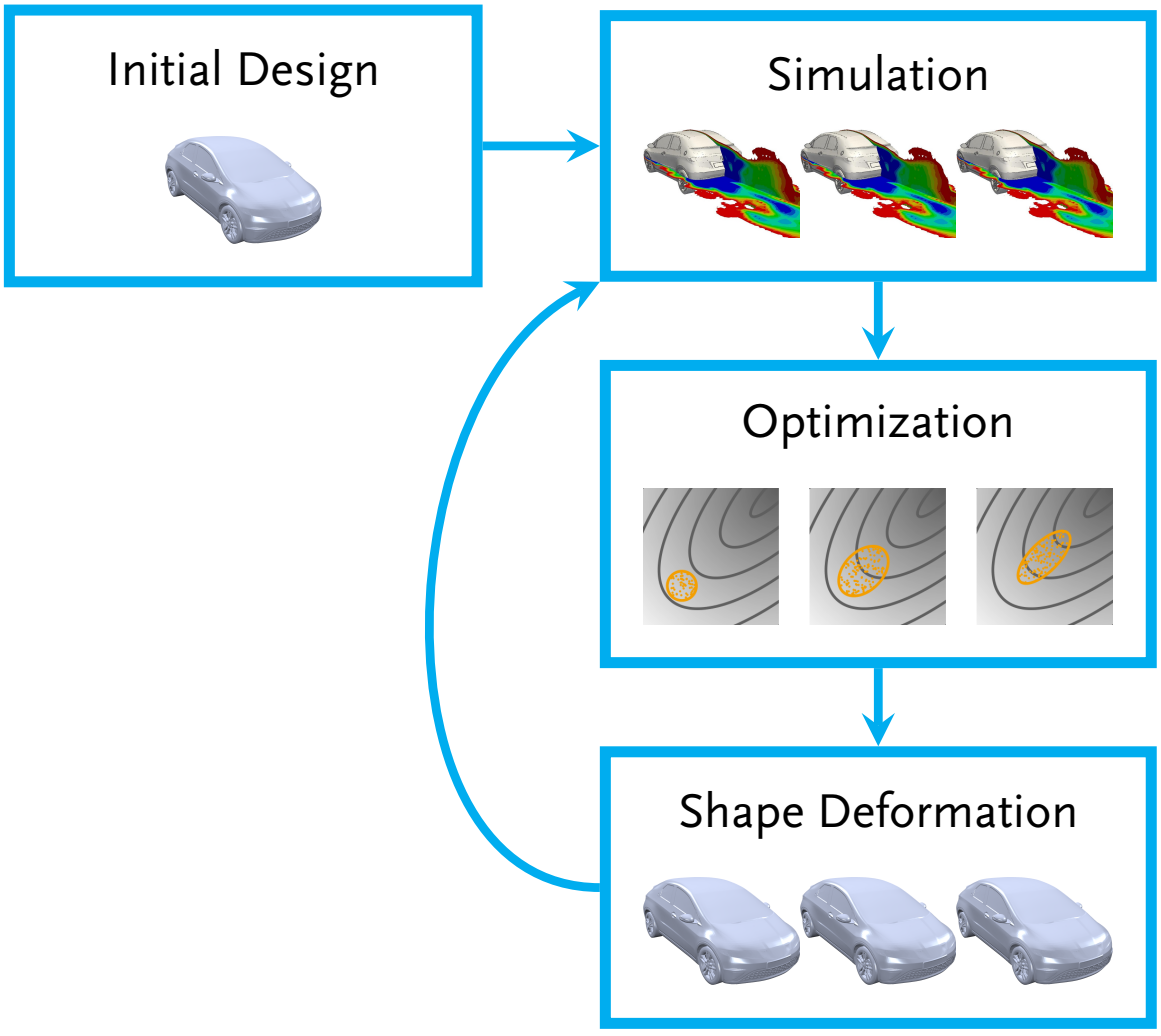
Shape Deformation & Design Optimization



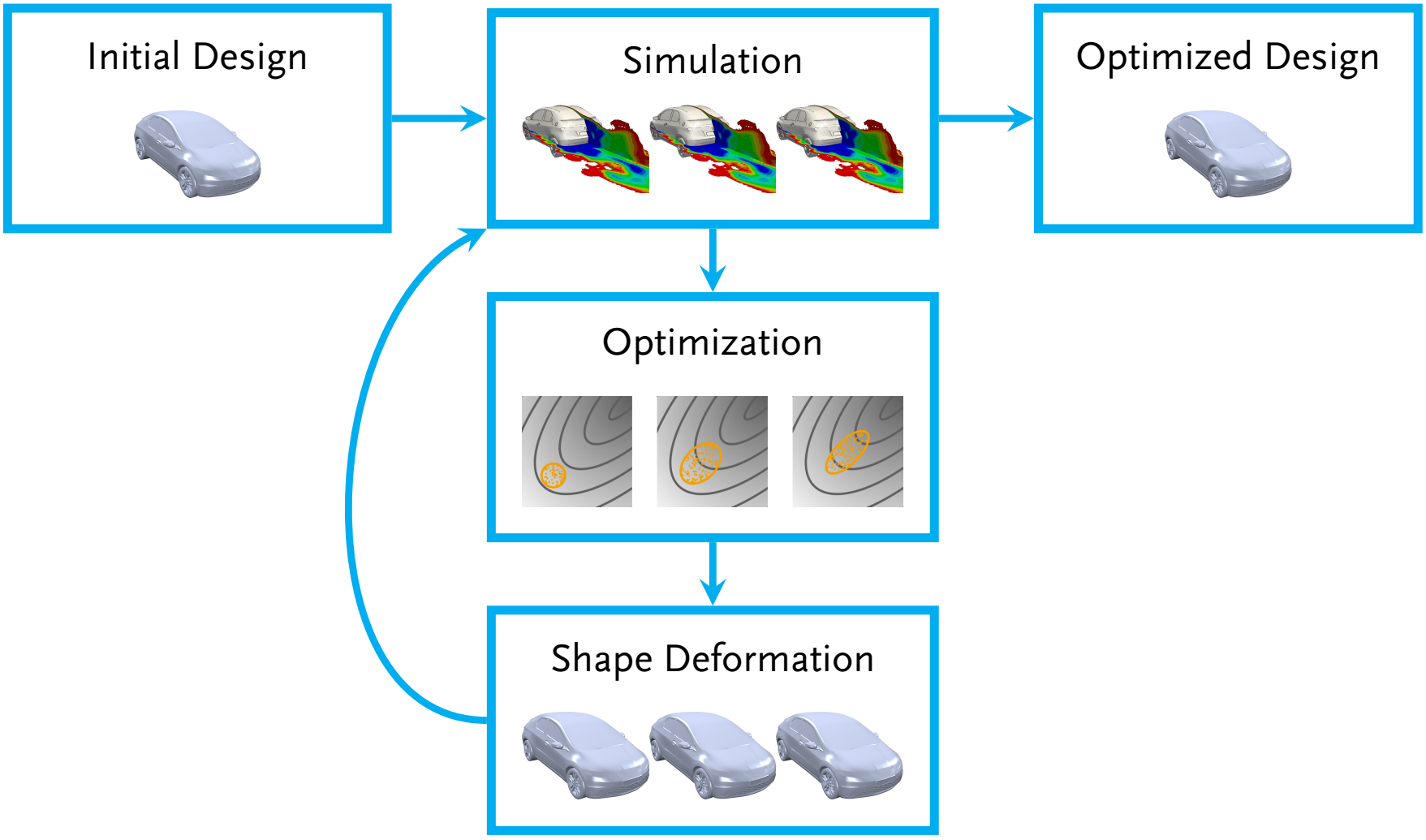
Shape Deformation & Design Optimization



Shape Deformation & Design Optimization

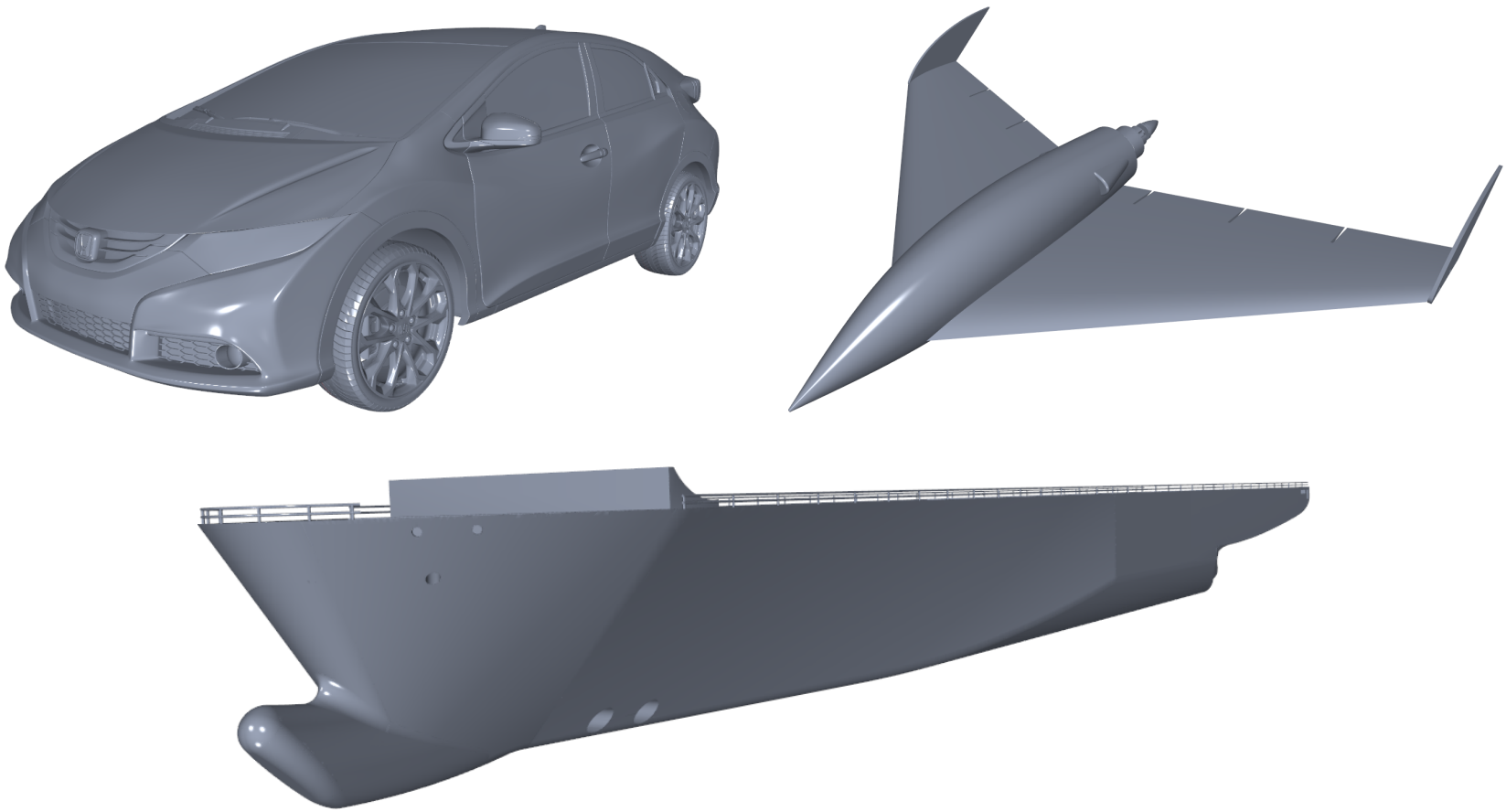


Shape Deformation & Design Optimization



Shape Deformation & Design Optimization

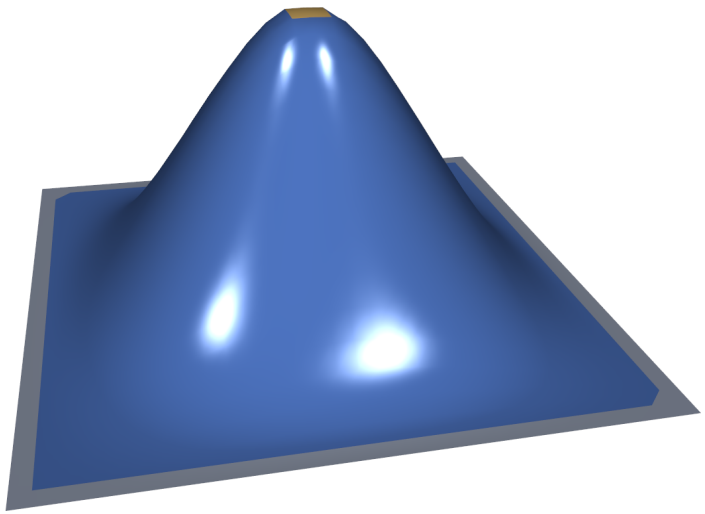
Common target designs are sheet metal *surfaces*

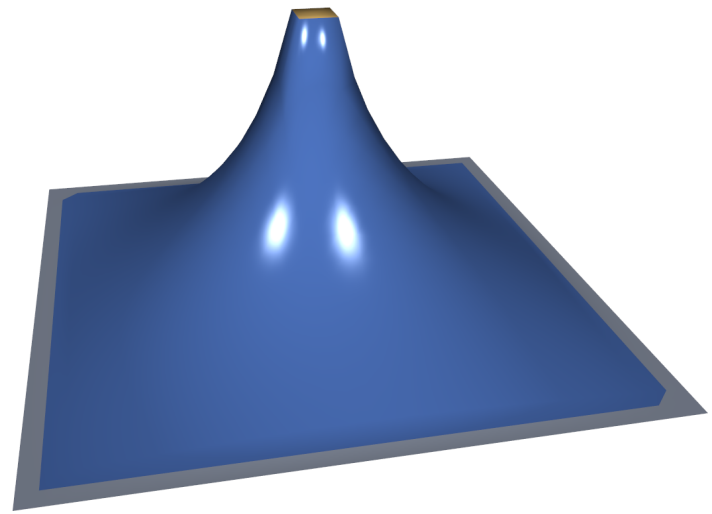


Mesh-Based Surface Deformation

Thin Shell Deformation [1]

- Physically-inspired technique suitable for sheet metal surfaces
- Flexible modeling of material behavior
- Based on the minimization of stretching and bending energies





1. Botsch et al., *On Linear Variational Surface Deformation Methods*, Trans. on Visualization and Computer Graphics, 2008

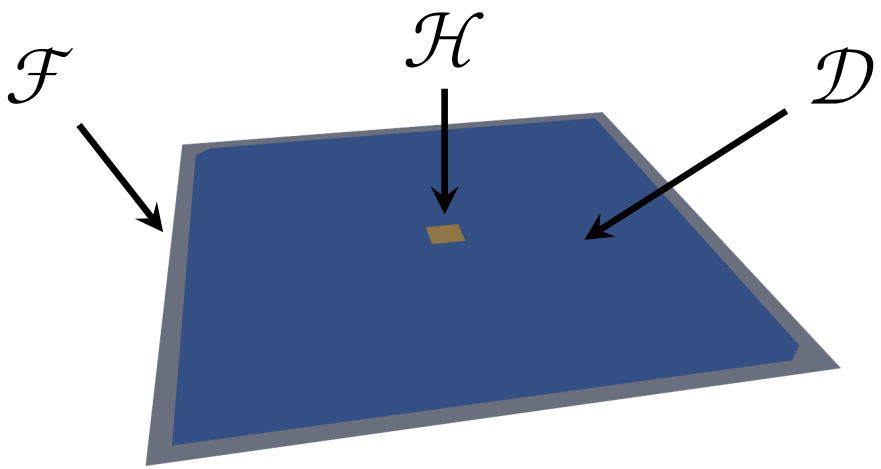
Thin Shell Deformation

Measure stretching and bending by 1st and 2nd order partial derivatives of the displacement function $\boldsymbol{d} : S \to \mathbb{R}^3$

$$E_{\text{stretch}}[\boldsymbol{d}] = \int_{\mathcal{D}} \|\nabla \boldsymbol{d}(\boldsymbol{x})\|^2 \, \mathrm{d}\boldsymbol{x}$$

$$E_{\text{bend}}[\boldsymbol{d}] = \int_{\mathcal{D}} \|\Delta \boldsymbol{d}(\boldsymbol{x})\|^2 \, \mathrm{d}\boldsymbol{x}$$

$$E_{\text{fix}}[\boldsymbol{d}] = \int_{\mathcal{H} \cup \mathcal{F}} \left\|\boldsymbol{d}(\boldsymbol{x}) - \bar{\boldsymbol{d}}(\boldsymbol{x})\right\|^2 \, \mathrm{d}\boldsymbol{x}$$



Surface-Based Discretization

- Discretize on the mesh using standard differential operators [2]

$$E_{\text{stretch}}[\boldsymbol{d}_h] = \sum_{\boldsymbol{x}_i \in \mathcal{D}} A_i \, \|\nabla \boldsymbol{d}_i\|^2$$

$$E_{\text{bend}}[\boldsymbol{d}_h] = \sum_{\boldsymbol{x}_i \in \mathcal{D}} A_i \, \|\Delta \boldsymbol{d}_i\|^2$$

$$E_{\text{fix}}[\boldsymbol{d}_h] = \sum_{\boldsymbol{x}_i \in \mathcal{H} \cup \mathcal{F}} A_i \, \left\|\boldsymbol{d}_i - \bar{\boldsymbol{d}}_i\right\|^2$$

➔ Solve a linear system:

$$\left(w_s \boldsymbol{G}^T \boldsymbol{G} + w_b \boldsymbol{L}^T \boldsymbol{L} + w_f \boldsymbol{F}^T \boldsymbol{F}\right) \boldsymbol{d} \;=\; w_f \boldsymbol{F}^T \boldsymbol{F} \bar{\boldsymbol{d}}$$

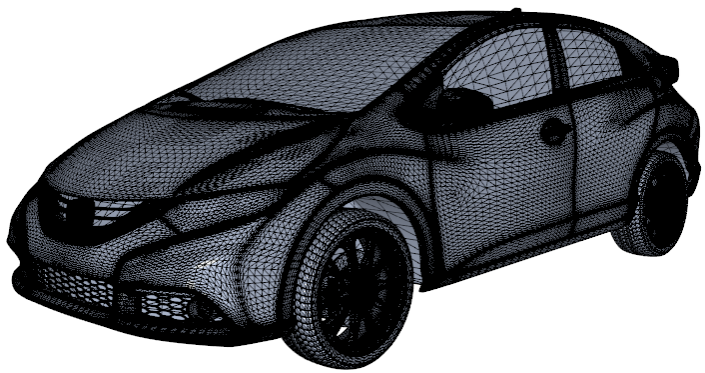
2. Meyer et al., *Discrete Differential-Geometry Operators for Triangulated 2-Manifolds*, Visualization and Mathematics, 2003

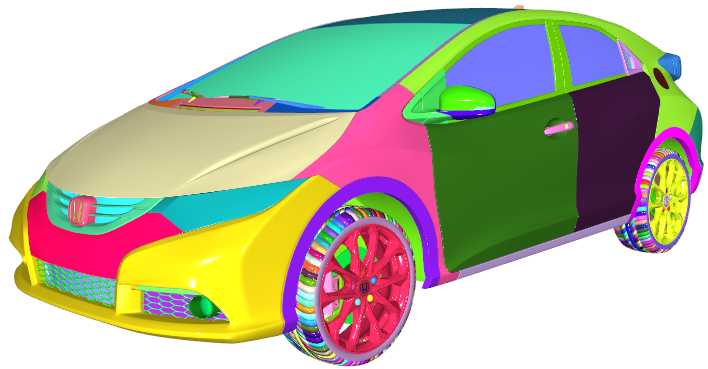
Limitations

- Assumption: Surface $\mathcal{S}$ is a proper triangle mesh

Limitations

- Assumption: Surface $\mathcal{S}$ is a proper triangle mesh





Space Deformation Methods

Space Deformation Methods

- Instead of deforming the surface $\mathcal{S}$, deform embedding space Ω
  - + Disconnected components
  - + Robustness against defects
  - + Representation independence

Space Deformation Methods

- Instead of deforming the surface $\mathcal{S}$, deform embedding space Ω
  - ✚ Disconnected components
  - ✚ Robustness against defects
  - ✚ Representation independence
- Construct a space deformation function $\boldsymbol{d}: \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}^3$

Space Deformation Methods

- Instead of deforming the surface $\mathcal{S}$, deform embedding space Ω
  - + Disconnected components
  - + Robustness against defects
  - + Representation independence
- Construct a space deformation function $\boldsymbol{d}: \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}^3$
  - – Use meshless approximation methods
  - – Represent $\boldsymbol{d}$ by basis functions φ_j located at centers $\boldsymbol{c}_j$:

$$\boldsymbol{d}(\boldsymbol{x}) = \sum_{j=1}^{k} \boldsymbol{w}_j \varphi_j(\boldsymbol{x})$$

Space Deformation Methods

- Instead of deforming the surface $\mathcal{S}$, deform embedding space Ω
  - + Disconnected components
  - + Robustness against defects
  - + Representation independence
- Construct a space deformation function $\boldsymbol{d}: \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}^3$
  - Use meshless approximation methods
  - Represent $\boldsymbol{d}$ by basis functions φ_j located at centers $\boldsymbol{c}_j$:

$$\boldsymbol{d}(\boldsymbol{x}) = \sum_{j=1}^{k} \boldsymbol{w}_j \varphi_j(\boldsymbol{x})$$

- Questions:
  - What basis functions to choose?

Space Deformation Methods

- Instead of deforming the surface $\mathcal{S}$, deform embedding space Ω
  - + Disconnected components
  - + Robustness against defects
  - + Representation independence
- Construct a space deformation function $\boldsymbol{d}: \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}^3$
  - Use meshless approximation methods
  - Represent $\boldsymbol{d}$ by basis functions φ_j located at centers $\boldsymbol{c}_j$:

$$\boldsymbol{d}(\boldsymbol{x}) = \sum_{j=1}^{k} \boldsymbol{w}_j \varphi_j(\boldsymbol{x})$$

- Questions:
  - What basis functions to choose?
  - Where to place basis functions?

Space Deformation Methods

- Instead of deforming the surface $\mathcal{S}$, deform embedding space Ω
  - + Disconnected components
  - + Robustness against defects
  - + Representation independence
- Construct a space deformation function $\boldsymbol{d}: \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}^3$
  - Use meshless approximation methods
  - Represent $\boldsymbol{d}$ by basis functions φ_j located at centers $\boldsymbol{c}_j$:

$$\boldsymbol{d}(\boldsymbol{x}) = \sum_{j=1}^{k} \boldsymbol{w}_j \varphi_j(\boldsymbol{x})$$

- Questions:
  - What basis functions to choose?
  - Where to place basis functions?

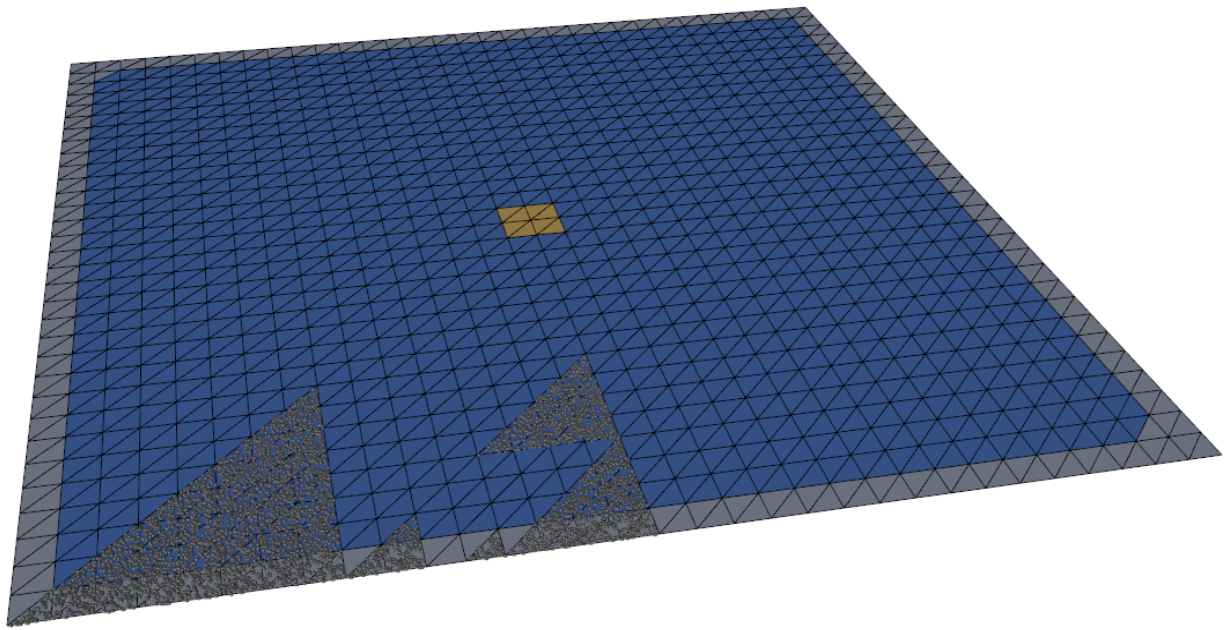
Surface Sampling

Goal: Generate uniformly distributed points $\boldsymbol{c}_j$ on the surface

Surface Sampling

Goal: Generate uniformly distributed points c_j on the surface

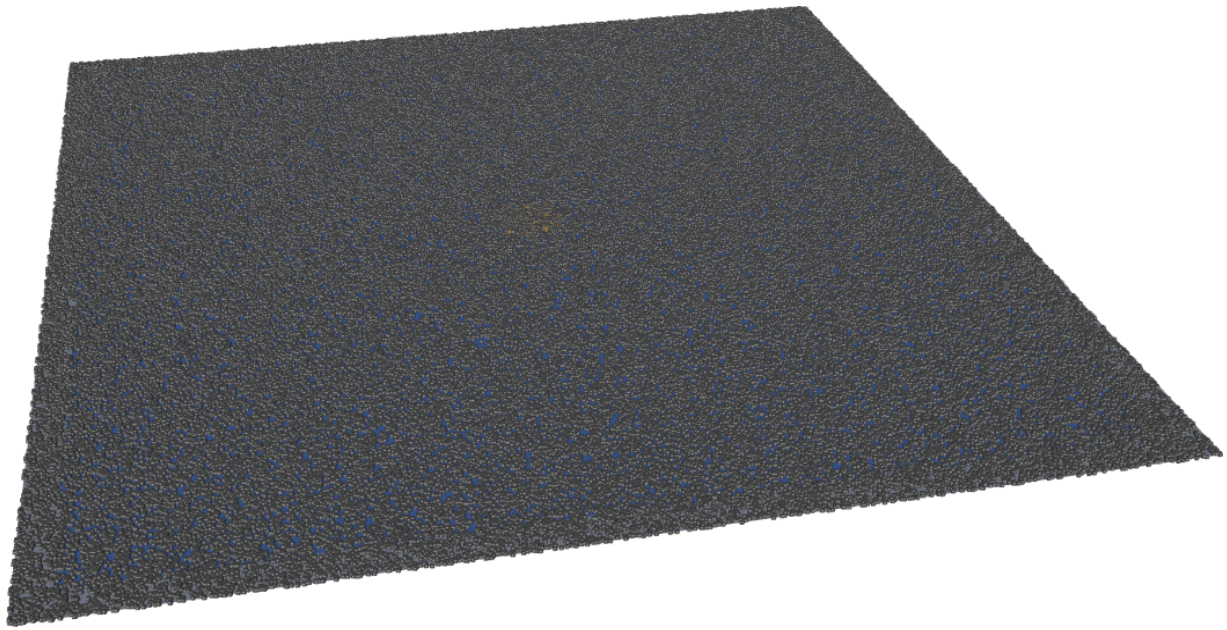
1. Dense random sampling of each mesh face



Surface Sampling

Goal: Generate uniformly distributed points $\boldsymbol{c}_j$ on the surface

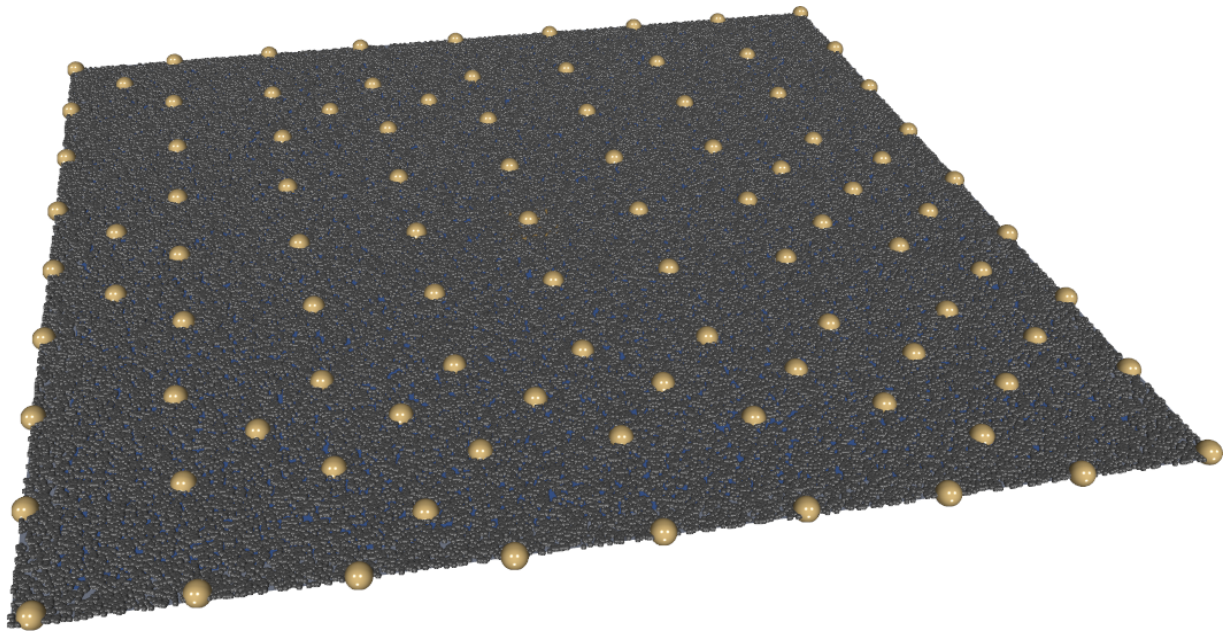
1. Dense random sampling of each mesh face



Surface Sampling

Goal: Generate uniformly distributed points c_j on the surface

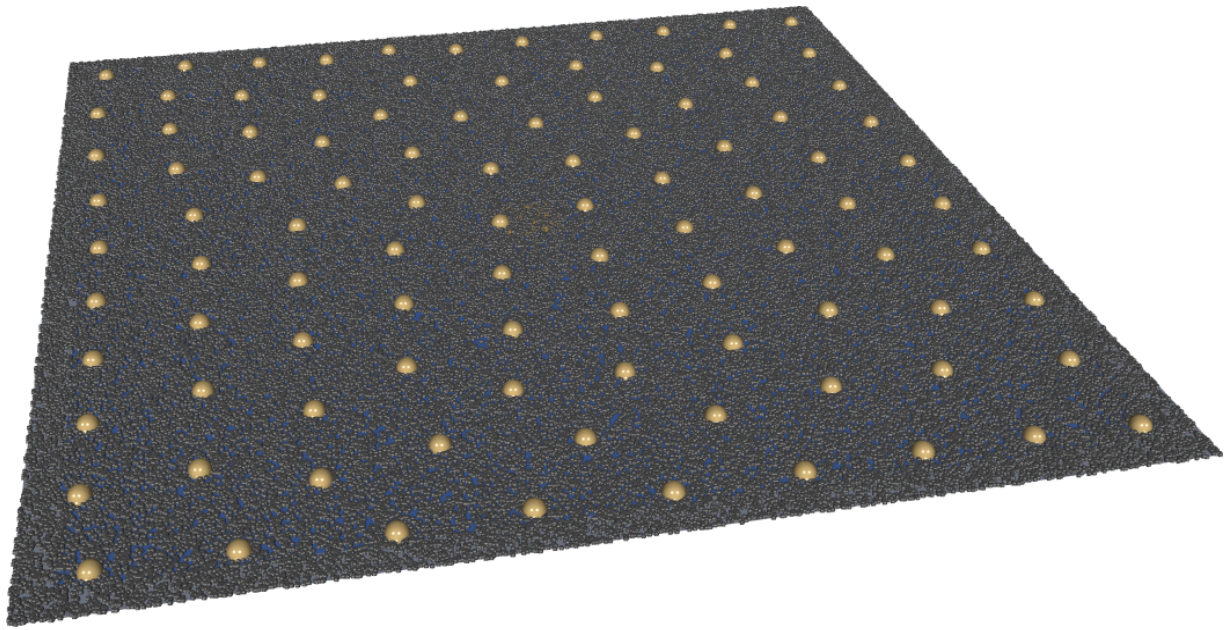
1. Dense random sampling of each mesh face
2. Farthest point selection



Surface Sampling

Goal: Generate uniformly distributed points c_j on the surface

1. Dense random sampling of each mesh face
2. Farthest point selection
3. Lloyd relaxation (k-means clustering)



Subspace Surface Deformation

- What basis functions φ_j to choose?

Subspace Surface Deformation

- What basis functions φ_j to choose?
- Goal: Achieve same modeling flexibility as surface deformation

Subspace Surface Deformation

- What basis functions φ_j to choose?
- Goal: Achieve same modeling flexibility as surface deformation
- Ground truth: Combine surface energy with space deformation

Subspace Surface Deformation

- What basis functions φ_j to choose?
- Goal: Achieve same modeling flexibility as surface deformation
- Ground truth: Combine surface energy with space deformation
- Express $\boldsymbol{d}$ through coefficients $\boldsymbol{w}$ and subspace matrix $\boldsymbol{\Phi}$

$$\boldsymbol{d} = \boldsymbol{\Phi}\boldsymbol{w} \quad \text{with} \quad \boldsymbol{\Phi}_{ij} = \varphi_j(\boldsymbol{x}_i)$$

Subspace Surface Deformation

- What basis functions φ_j to choose?
- Goal: Achieve same modeling flexibility as surface deformation
- Ground truth: Combine surface energy with space deformation
- Express $\boldsymbol{d}$ through coefficients $\boldsymbol{w}$ and subspace matrix $\boldsymbol{\Phi}$

$$\boldsymbol{d} = \boldsymbol{\Phi}\boldsymbol{w} \quad \text{with} \quad \boldsymbol{\Phi}_{ij} = \varphi_j(\boldsymbol{x}_i)$$

- Leads to a modified linear system:

$$\boldsymbol{\Phi}^T \left(w_s \boldsymbol{G}^T \boldsymbol{G} + w_b \boldsymbol{L}^T \boldsymbol{L} + w_f \boldsymbol{F}^T \boldsymbol{F} \right) \boldsymbol{\Phi}\, \boldsymbol{w} \;=\; \boldsymbol{\Phi}^T \left(w_f \boldsymbol{F}^T \boldsymbol{F} \bar{\boldsymbol{d}} \right)$$

Subspace Surface Deformation

- Global triharmonic radial basis functions (RBFs)

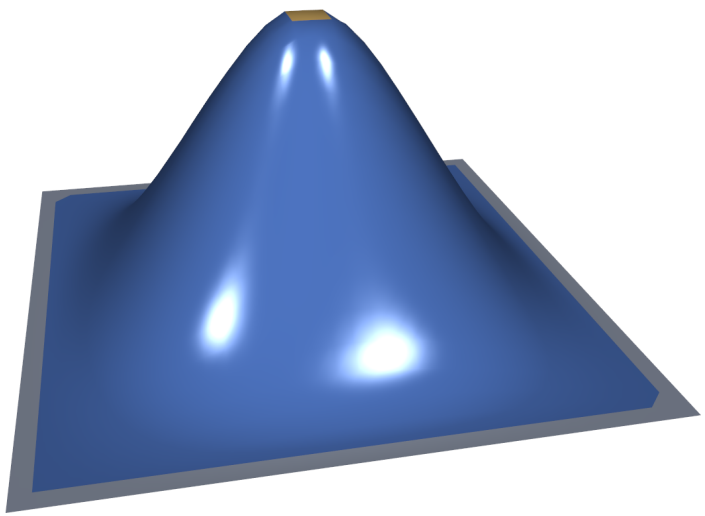
$$\varphi_j(\boldsymbol{x}) \;=\; \left\| \boldsymbol{x} - \boldsymbol{c}_j \right\|^3$$

Subspace Surface Deformation

- Global triharmonic radial basis functions (RBFs)

$$\varphi_j(\boldsymbol{x}) \;=\; \left\|\boldsymbol{x} - \boldsymbol{c}_j\right\|^3$$

- Good results for bending

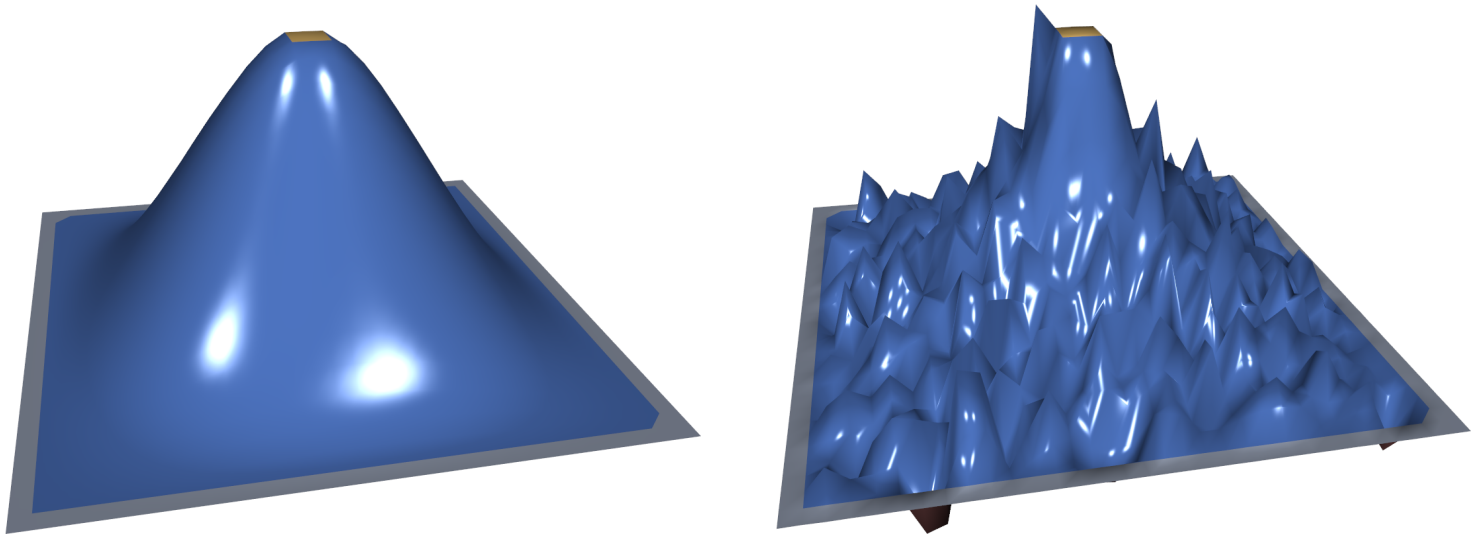


Subspace Surface Deformation

- Global triharmonic radial basis functions (RBFs)

$$\varphi_j(\boldsymbol{x}) \;=\; \left\|\boldsymbol{x} - \boldsymbol{c}_j\right\|^3$$

- Good results for bending, bad results for stretching

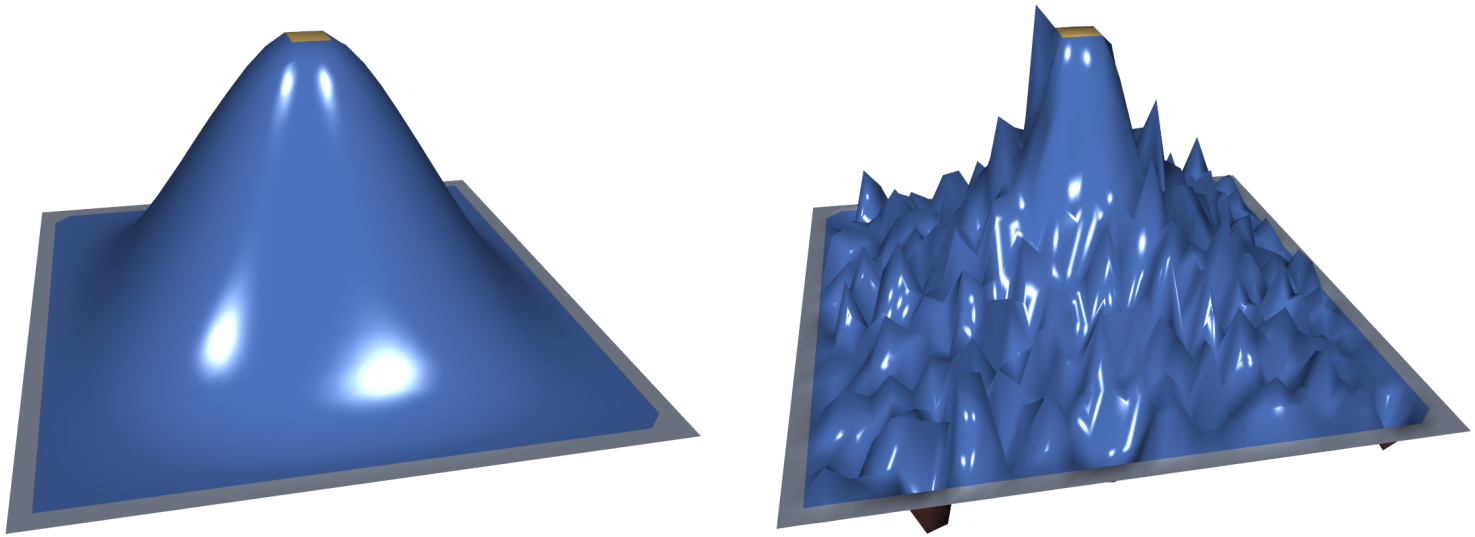


Subspace Surface Deformation

- Global triharmonic radial basis functions (RBFs)

$$\varphi_j(\boldsymbol{x}) \;=\; \left\|\boldsymbol{x} - \boldsymbol{c}_j\right\|^3$$

- Good results for bending, bad results for stretching

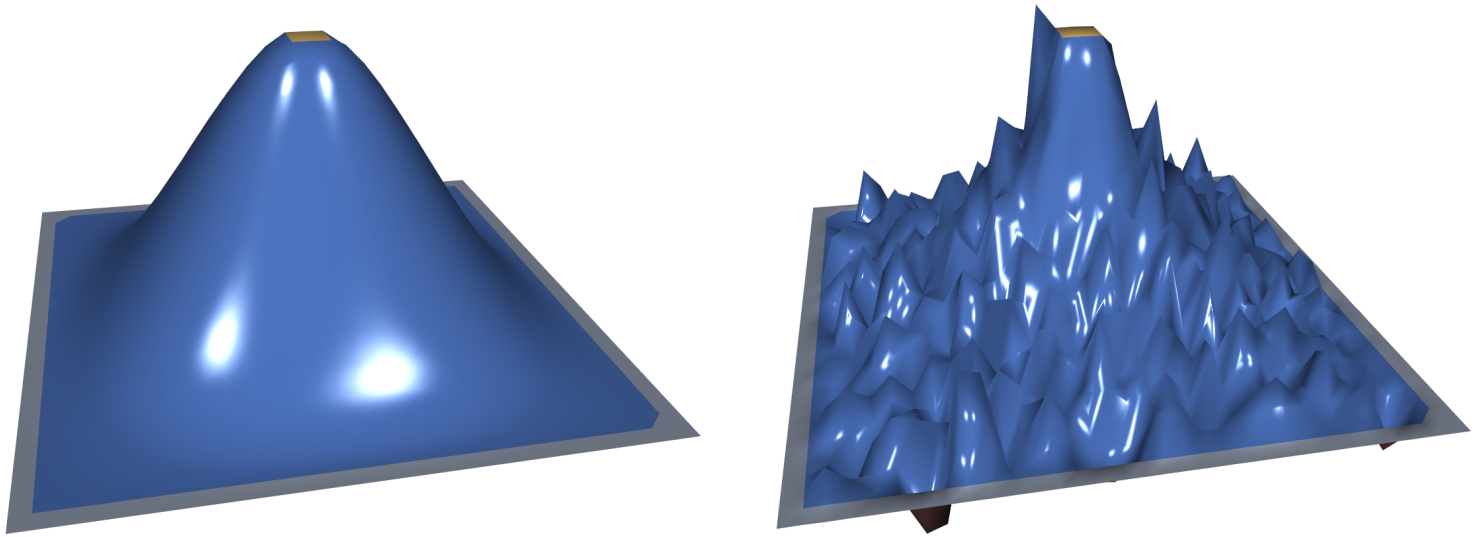


Subspace Surface Deformation

- Global triharmonic radial basis functions (RBFs)

$$\varphi_j(\boldsymbol{x}) \;=\; \left\|\boldsymbol{x} - \boldsymbol{c}_j\right\|^3$$

- Good results for bending, bad results for stretching
- Global support → dense linear systems



Subspace Surface Deformation

- Compactly supported RBFs

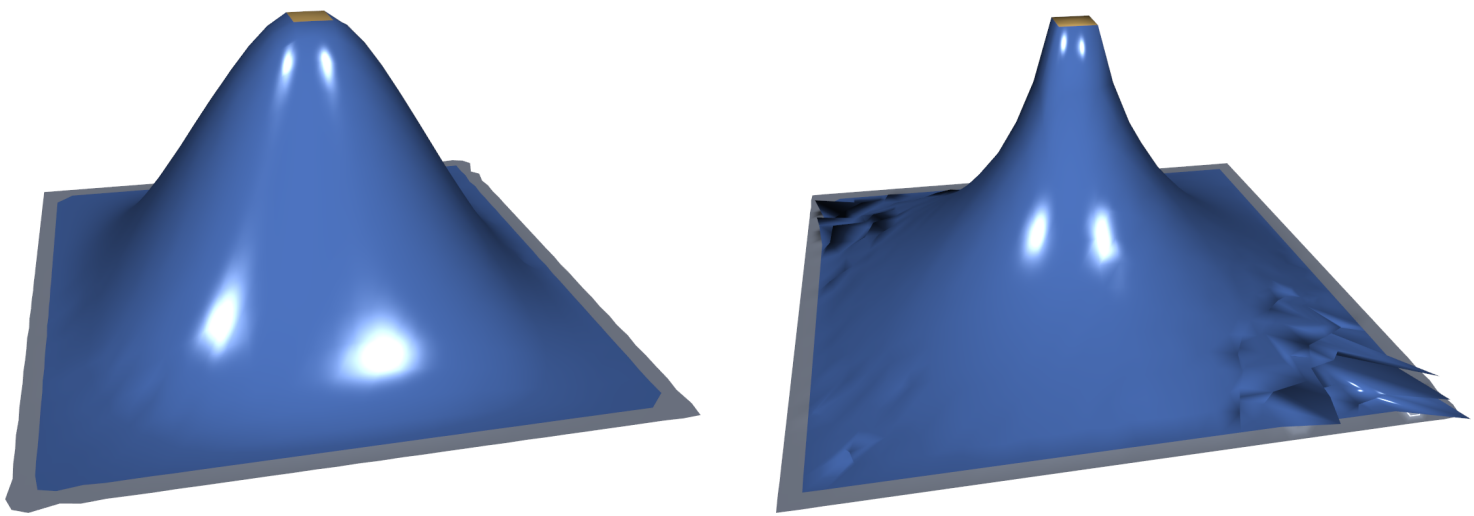
$$\varphi_j(\boldsymbol{x}) \;=\; \varphi\left(\left\|\boldsymbol{x}-\boldsymbol{c}_j\right\|\right) \;=\; \varphi(r) \;=\; \begin{cases} (1-r)^4(4r+1)\,, & r<\sigma\,, \\ 0\,, & \text{otherwise}\,. \end{cases}$$

Subspace Surface Deformation

- Compactly supported RBFs

$$\varphi_j(\boldsymbol{x}) \;=\; \varphi\left(\left\|\boldsymbol{x}-\boldsymbol{c}_j\right\|\right) \;=\; \varphi(r) \;=\; \begin{cases} (1-r)^4(4r+1)\,, & r<\sigma\,, \\ 0\,, & \text{otherwise}\,. \end{cases}$$

- *Small* support → *sparse* linear system, *bad* results

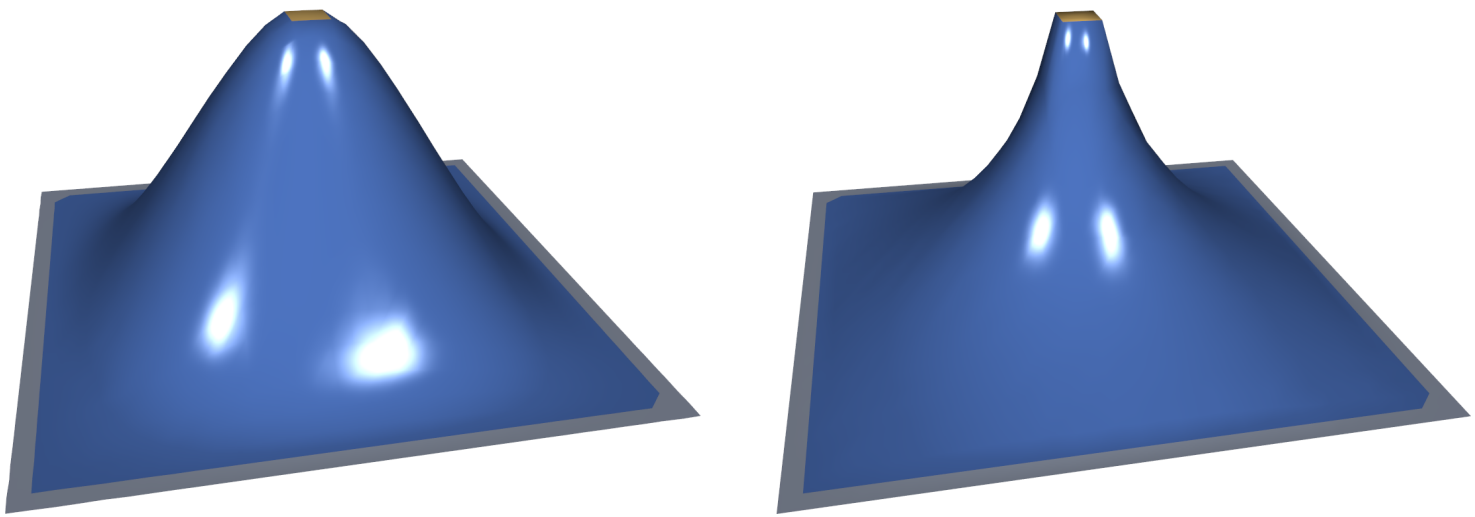


Subspace Surface Deformation

- Compactly supported RBFs

$$\varphi_j(\boldsymbol{x}) \;=\; \varphi\left(\left\|\boldsymbol{x}-\boldsymbol{c}_j\right\|\right) \;=\; \varphi(r) \;=\; \begin{cases} (1-r)^4(4r+1)\,, & r<\sigma\,, \\ 0\,, & \text{otherwise}\,. \end{cases}$$

- *Large* support → *dense* linear system, *good* results



Subspace Surface Deformation

- Moving Least Squares (MLS) basis functions[3]

$$\varphi_j(\boldsymbol{x}) \;=\; \boldsymbol{p}(\boldsymbol{x})^T \boldsymbol{M}^{-1}(\boldsymbol{x}) \boldsymbol{p}(\boldsymbol{c}_j) w(\boldsymbol{x} - \boldsymbol{c}_j)$$

3. Fries et al., *Classification and Overview of Meshfree Methods*, Technical Report, TU Braunschweig, 2003

Subspace Surface Deformation

- Moving Least Squares (MLS) basis functions[3]

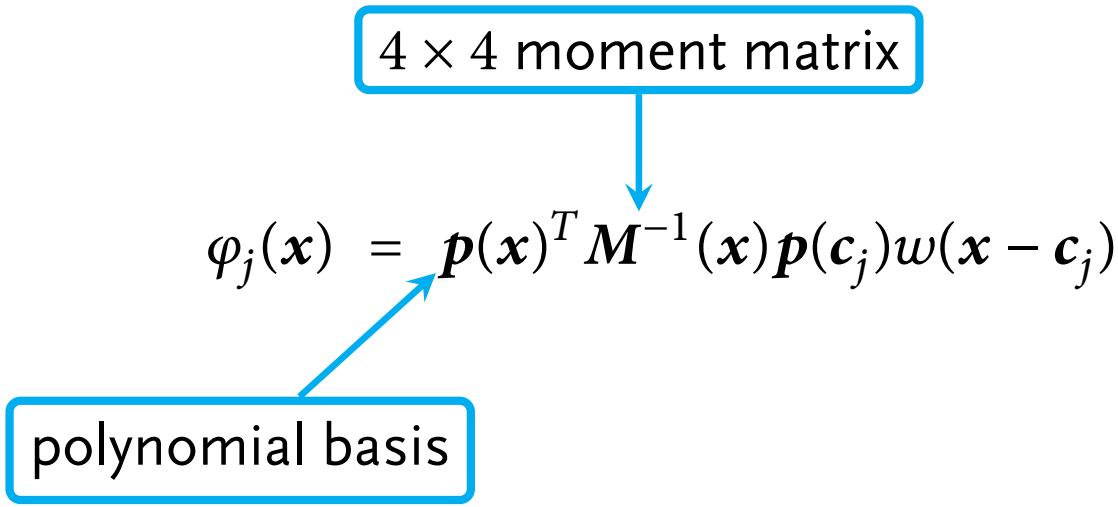
$$\varphi_j(\boldsymbol{x}) = \boldsymbol{p}(\boldsymbol{x})^T \boldsymbol{M}^{-1}(\boldsymbol{x}) \boldsymbol{p}(\boldsymbol{c}_j) w(\boldsymbol{x} - \boldsymbol{c}_j)$$

polynomial basis

3. Fries et al., *Classification and Overview of Meshfree Methods*, Technical Report, TU Braunschweig, 2003

Subspace Surface Deformation

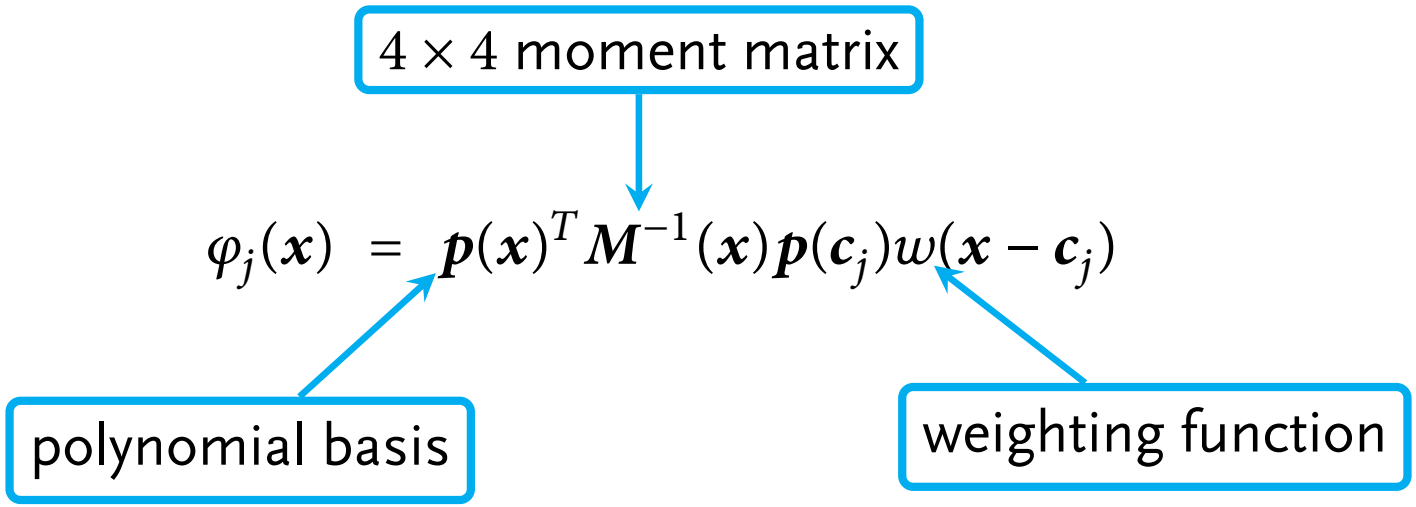
- Moving Least Squares (MLS) basis functions[3]



3. Fries et al., *Classification and Overview of Meshfree Methods*, Technical Report, TU Braunschweig, 2003

Subspace Surface Deformation

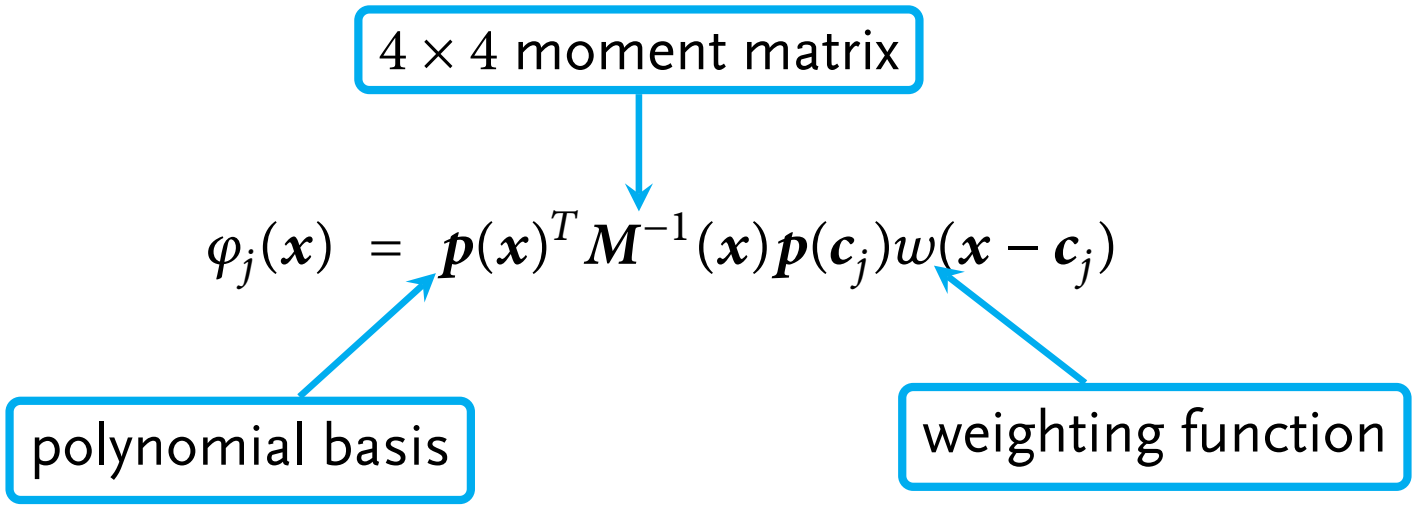
- Moving Least Squares (MLS) basis functions[3]



3. Fries et al., *Classification and Overview of Meshfree Methods*, Technical Report, TU Braunschweig, 2003

Subspace Surface Deformation

- Moving Least Squares (MLS) basis functions[3]

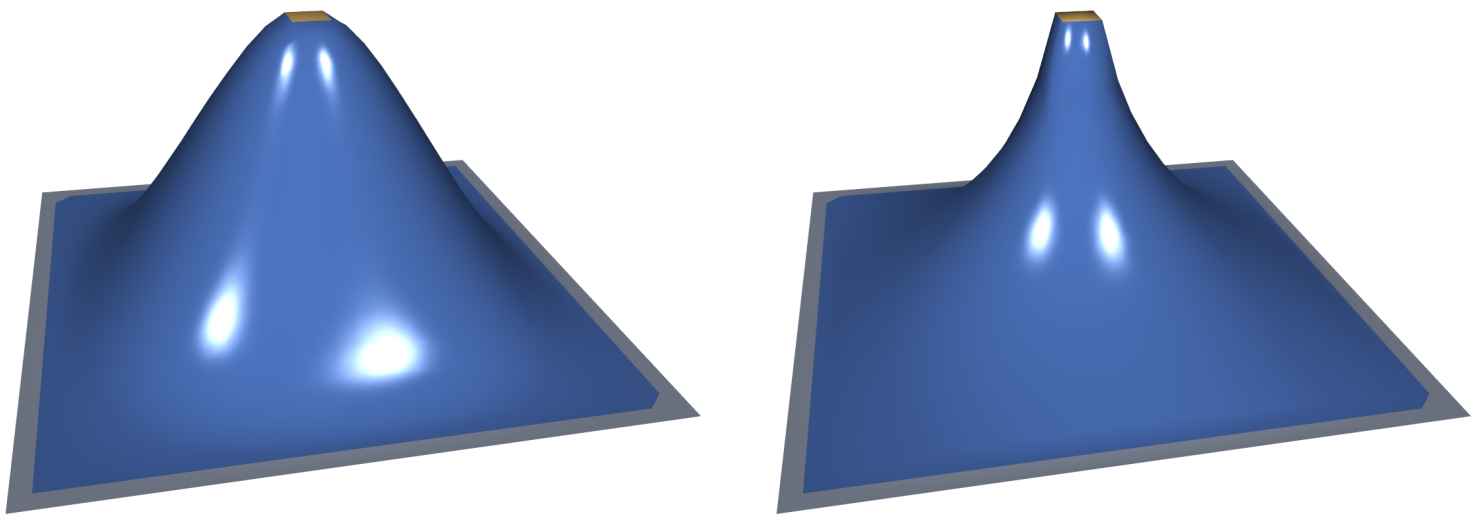


- More complex form, inversion of $\boldsymbol{M}$ required

3. Fries et al., *Classification and Overview of Meshfree Methods*, Technical Report, TU Braunschweig, 2003

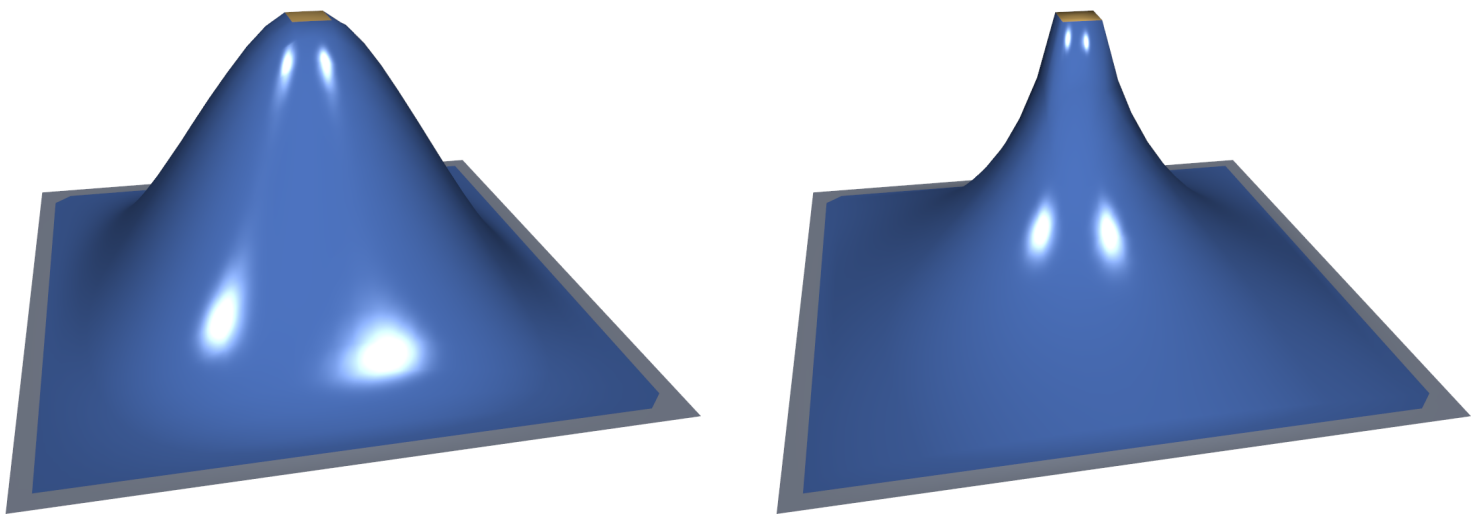
Subspace Surface Deformation

- Moving Least Squares (MLS) basis functions
- \+ *Small* support → *sparse* linear system, *good* results



Subspace Surface Deformation

- Moving Least Squares (MLS) basis functions
- \+ *Small* support → *sparse* linear system, *good* results



Volumetric Space Deformation

Volumetric Space Deformation

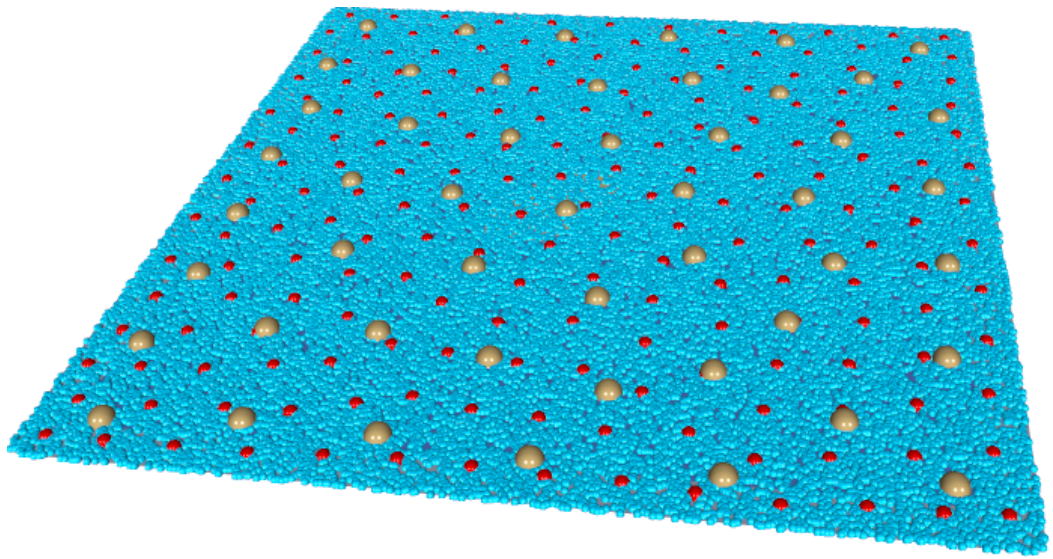
- Goal: Fully space-based discretization of stretching and bending energies using MLS approximation

Volumetric Space Deformation

- Goal: Fully space-based discretization of stretching and bending energies using MLS approximation
- Replace vertex-based integration over the surface $\mathcal{S}$ with purely space-based integration method

Volumetric Space Deformation

- Goal: Fully space-based discretization of stretching and bending energies using MLS approximation
- Replace vertex-based integration over the surface $\mathcal{S}$ with purely space-based integration method
- Use Lloyd-based sampling to determine integration points $\boldsymbol{q}_i$



Volumetric Space Deformation

- Evaluate gradients and Laplacians of φ_j at integration points $\boldsymbol{q}_i$:

$$\widetilde{E}_{\text{stretch}} = \sum_{i=1}^{N} V_i \left\| \nabla \boldsymbol{d}(\boldsymbol{q}_i) \right\|^2 = \sum_{i=1}^{N} V_i \left\| \sum_{j=1}^{k} \boldsymbol{w}_j \nabla \varphi_j(\boldsymbol{q}_i) \right\|^2$$

$$\widetilde{E}_{\text{bend}} = \sum_{i=1}^{N} V_i \left\| \Delta \boldsymbol{d}(\boldsymbol{q}_i) \right\|^2 = \sum_{i=1}^{N} V_i \left\| \sum_{j=1}^{k} \boldsymbol{w}_j \Delta \varphi_j(\boldsymbol{q}_i) \right\|^2$$

Volumetric Space Deformation

- Evaluate gradients and Laplacians of φ_j at integration points $\boldsymbol{q}_i$:

$$\widetilde{E}_{\text{stretch}} = \sum_{i=1}^{N} V_i \left\| \nabla \boldsymbol{d}(\boldsymbol{q}_i) \right\|^2 = \sum_{i=1}^{N} V_i \left\| \sum_{j=1}^{k} \boldsymbol{w}_j \nabla \varphi_j(\boldsymbol{q}_i) \right\|^2$$

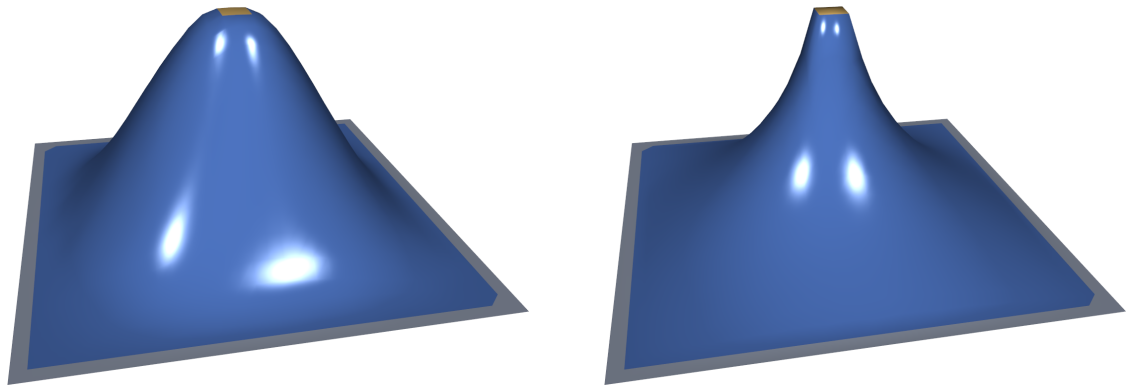
$$\widetilde{E}_{\text{bend}} = \sum_{i=1}^{N} V_i \left\| \Delta \boldsymbol{d}(\boldsymbol{q}_i) \right\|^2 = \sum_{i=1}^{N} V_i \left\| \sum_{j=1}^{k} \boldsymbol{w}_j \Delta \varphi_j(\boldsymbol{q}_i) \right\|^2$$

- Leads to a modified linear system

$$\left(w_s \widetilde{\boldsymbol{G}}^T \widetilde{\boldsymbol{G}} + w_b \widetilde{\boldsymbol{L}}^T \widetilde{\boldsymbol{L}} + w_f \boldsymbol{\Phi}^T \boldsymbol{F}^T \boldsymbol{F} \boldsymbol{\Phi} \right) \boldsymbol{w} = w_f \boldsymbol{\Phi}^T \boldsymbol{F}^T \boldsymbol{F} \bar{\boldsymbol{d}}$$

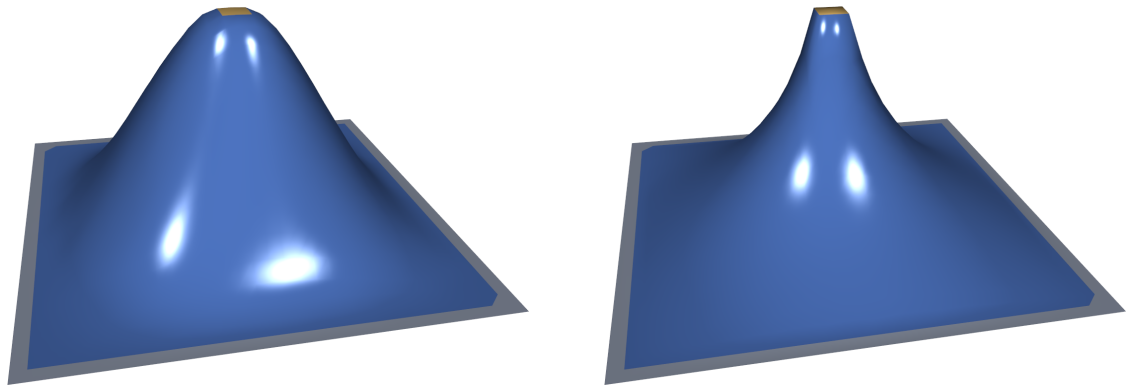
Volumetric Space Deformation

- Space-based discretization

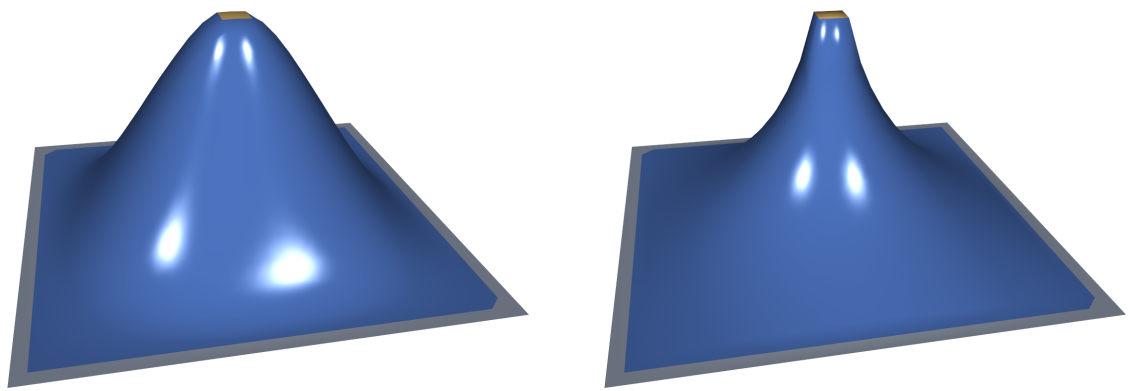


Volumetric Space Deformation

- Space-based discretization



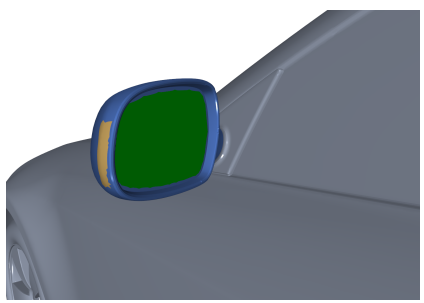
- Surface-based discretization

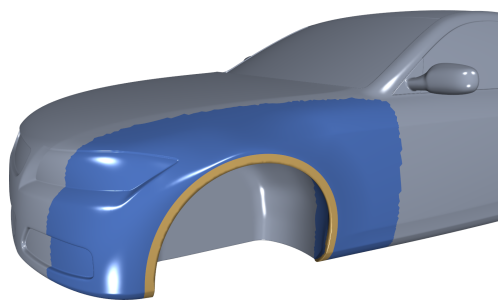


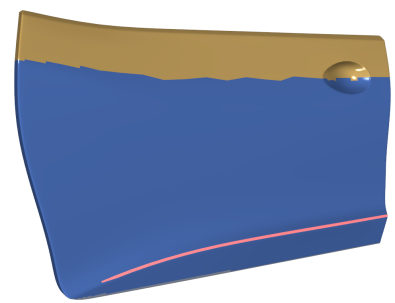
Constrained Space Deformation

Geometric Constraints

- Design prototypes contain important geometric features
  - Planar components
  - Circular couplings or wheelhouses
  - Characteristic feature lines

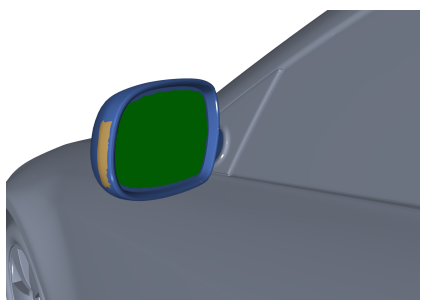


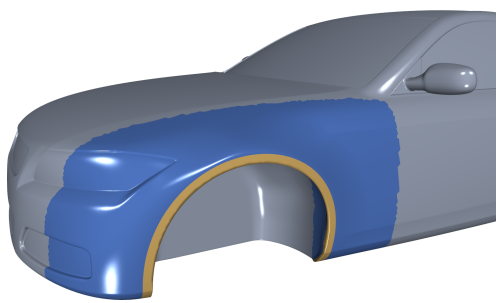


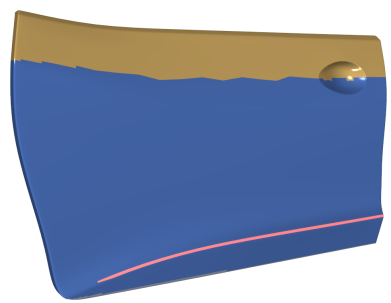


Geometric Constraints

- Design prototypes contain important geometric features
  - Planar components
  - Circular couplings or wheelhouses
  - Characteristic feature lines
- Deforming the design during optimization distorts features
  - Impaired functionality
  - Violated production limitations

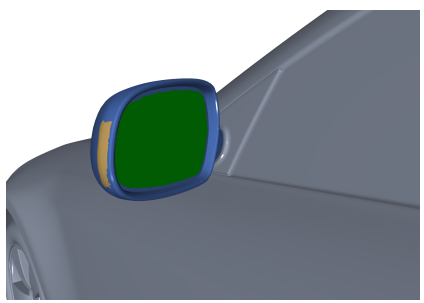
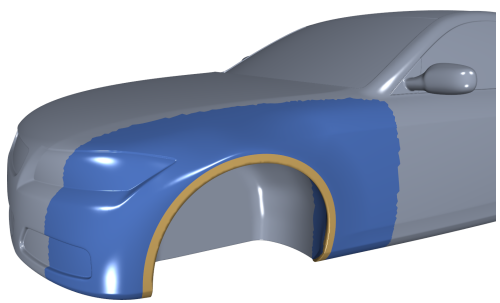
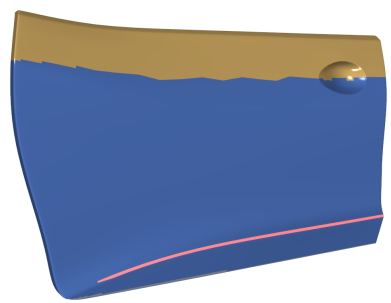






Geometric Constraints

- Design prototypes contain important geometric features
  - Planar components
  - Circular couplings or wheelhouses
  - Characteristic feature lines
- Deforming the design during optimization distorts features
  - Impaired functionality
  - Violated production limitations
- Classical solution: Add penalty terms to the cost function
  - *Creation* of infeasible designs
  - Costly *evaluation* (e.g., CFD)

Geometric Constraints

- Our approach: Prevent distortion of features by incorporating geometric constraints into the deformation
  - + Only create feasible designs
  - + Avoid unnecessary performance evaluations

Geometric Constraints

- Our approach: Prevent distortion of features by incorporating geometric constraints into the deformation
  - \+ Only create feasible designs
  - \+ Avoid unnecessary performance evaluations
- Constrained deformation techniques
  - – Most methods are surface-based

Geometric Constraints

- Our approach: Prevent distortion of features by incorporating geometric constraints into the deformation
  - \+ Only create feasible designs
  - \+ Avoid unnecessary performance evaluations
- Constrained deformation techniques
  - – Most methods are surface-based
  - \+ Projection-based constraints[4]

4. Bouaziz et al., *Shape-Up: Shaping Discrete Geometry with Projections*, Computer Graphics Forum, 2012

Projection-Based Constraints

- Define projection operators P_c: Plane, circle, ...
- Minimize deviation from prescribed constraints

$$E_{\text{constr}}(\boldsymbol{x}) = \sum_{c=1}^{s} \|\boldsymbol{x} - P_c(\boldsymbol{x})\|^2$$

Projection-Based Constraints

- Define projection operators P_c: Plane, circle, ...
- Minimize deviation from prescribed constraints

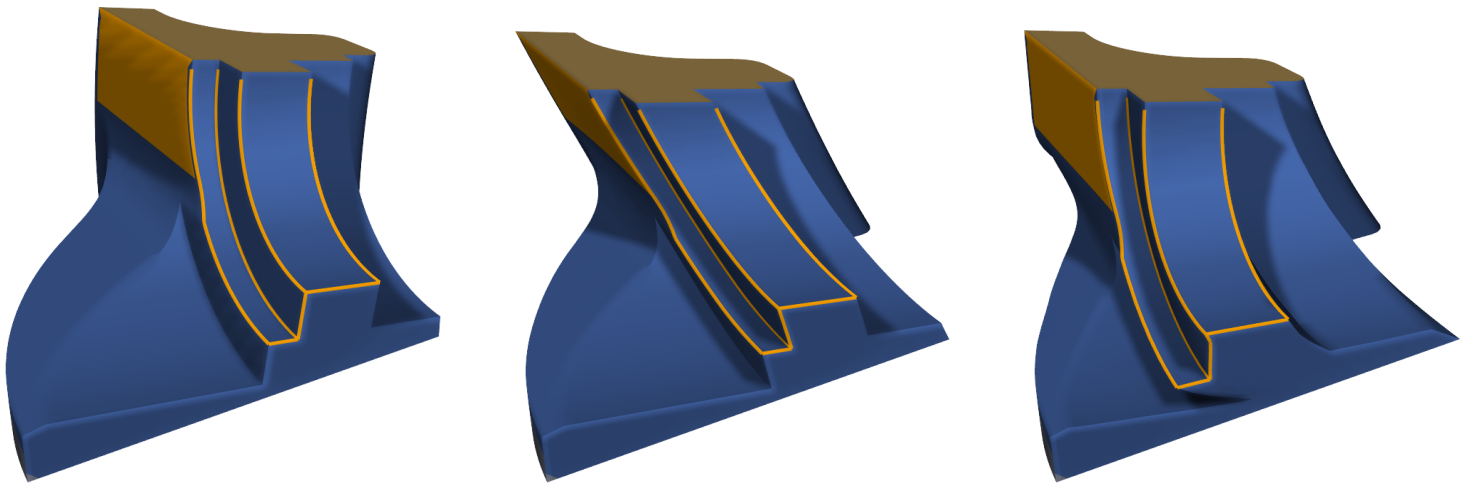
$$E_{\text{constr}}(\boldsymbol{x}) = \sum_{c=1}^{s} \|\boldsymbol{x} - P_c(\boldsymbol{x})\|^2$$

- Projections P_c typically are nonlinear functions of $\boldsymbol{x}$

➔ Minimize E_{constr} by iterative alternating optimization

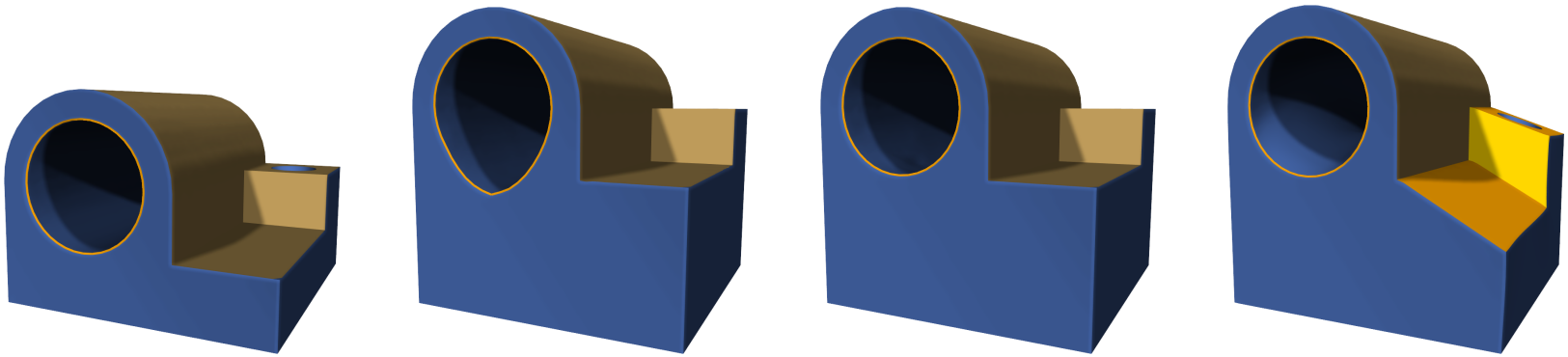
Geometric Constraint Examples

Fundamental geometric constraints: Planarity, circularity, feature lines



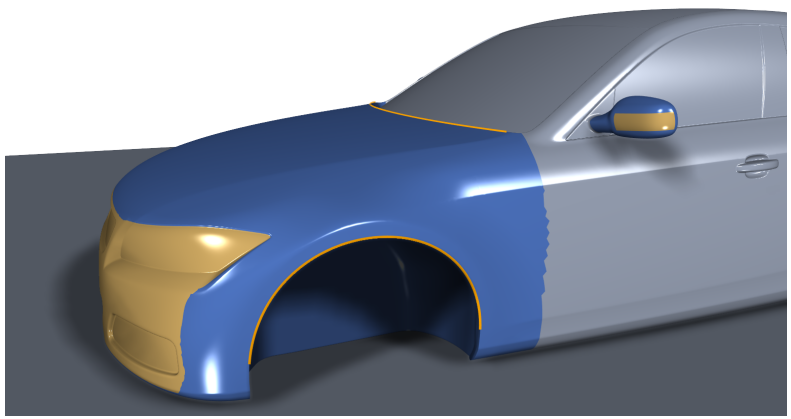
Geometric Constraint Examples

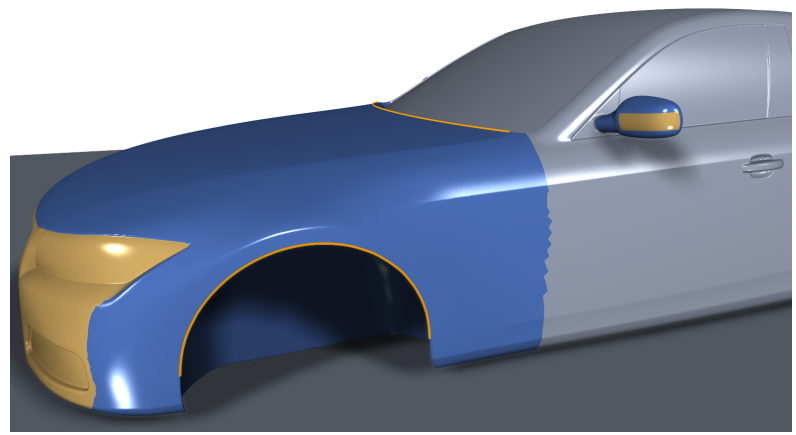
Fundamental geometric constraints: Planarity, circularity, feature lines



Geometric Constraint Examples

Fundamental geometric constraints: Planarity, circularity, feature lines

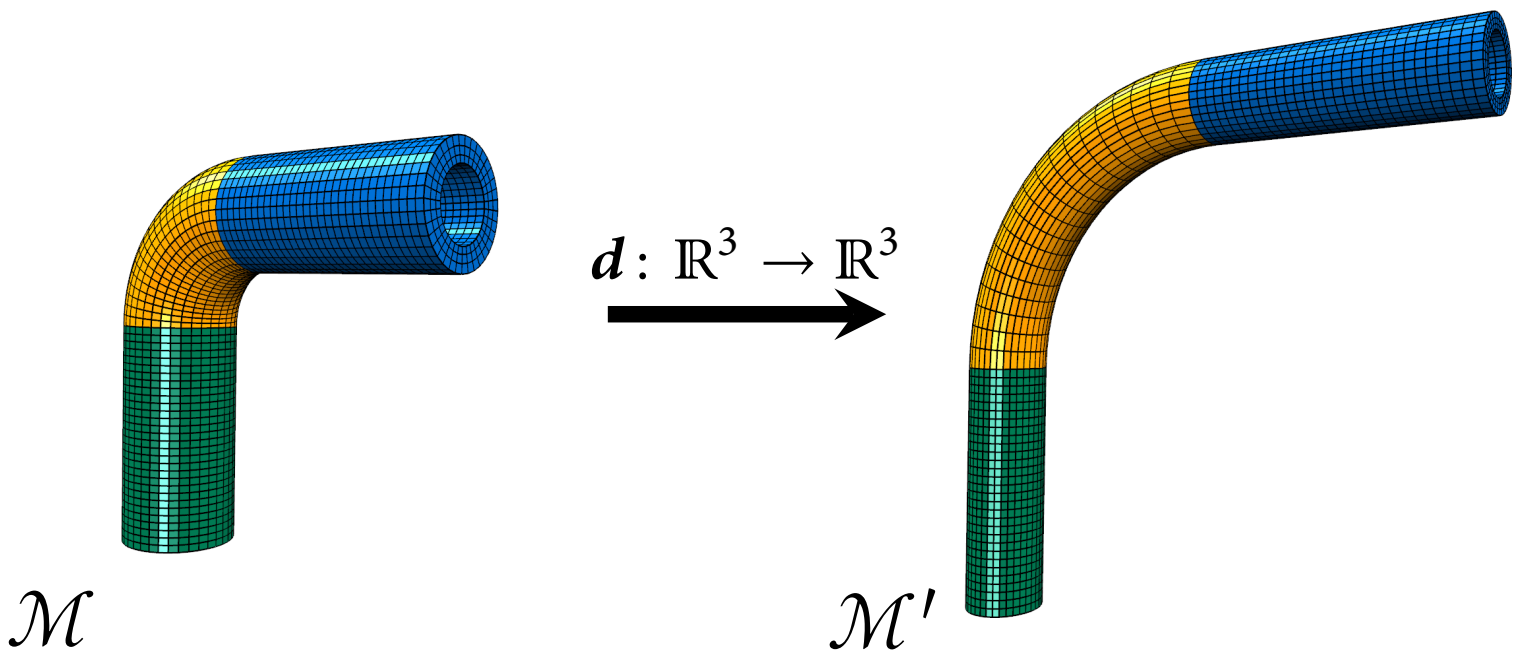




Volume Deformation Examples

Comparison to previous results[5]

- Original mesh quality: 0.98
- After RBF deformation: 0.951
- Using our new method: 0.954



5. Sieger et al., *RBF Morphing Techniques for Simulation-based Design Optimization*, Engineering with Computers, 2014

Summary & Outlook

Summary

- A deformation technique for design optimization
  - + Modeling flexibility like surface-based methods
  - + Representation-independence of space deformations
  - + High quality comparable to RBFs
  - + Improved scalability through sparse linear systems
  - + Geometric constraints

Summary

- A deformation technique for design optimization
  - \+ Modeling flexibility like surface-based methods
  - \+ Representation-independence of space deformations
  - \+ High quality comparable to RBFs
  - \+ Improved scalability through sparse linear systems
  - \+ Geometric constraints
  - – Precomputation time of MLS basis functions
  - – Slow convergence for complex constraints

Summary

- A deformation technique for design optimization
  - \+ Modeling flexibility like surface-based methods
  - \+ Representation-independence of space deformations
  - \+ High quality comparable to RBFs
  - \+ Improved scalability through sparse linear systems
  - \+ Geometric constraints
  - – Precomputation time of MLS basis functions
  - – Slow convergence for complex constraints
- Central idea: Improve the design optimization process by integrating constraints *directly* into the deformation

Summary

- A deformation technique for design optimization
  - \+ Modeling flexibility like surface-based methods
  - \+ Representation-independence of space deformations
  - \+ High quality comparable to RBFs
  - \+ Improved scalability through sparse linear systems
  - \+ Geometric constraints
  - – Precomputation time of MLS basis functions
  - – Slow convergence for complex constraints
- Central idea: Improve the design optimization process by integrating constraints *directly* into the deformation
- Observation: Significant differences in the modeling flexibility and quality of RBFs and MLS

Future Work

- Additional constraint types:
  - Rigid components
  - Width, height, distances
  - Symmetry relations
  - Angle relations
- Automatic constraint detection
- Performance improvements

Future Work?

- Additional constraint types:
  - Rigid components
  - Width, height, distances
  - Symmetry relations
  - Angle relations
- Automatic constraint detection
- Performance improvements

Thanks for your attention!