A Comprehensive Comparison of Shape Deformation Methods in Evolutionary Design Optimization

Daniel Sieger \(^1\), Stefan Menzel \(^2\), and Mario Botsch \(^1\)

\(^1\)Graphics & Geometry Group, Bielefeld University

\(^2\)Honda Research Institute
Outline

1. Shape deformation methods
2. Evolutionary design optimization
3. Application: Passenger car design optimization
Shape Deformation Methods
Shape Deformation Methods

• Fundamental requirements?
Shape Deformation Methods

- Fundamental requirements?
- Different representations
  - Surface meshes
  - Volume meshes
Shape Deformation Methods

• Fundamental requirements?

• Different representations
  - Surface meshes
  - Volume meshes

• Defects
  - Badly shaped elements
  - Non-manifold meshes
  - Self-intersections
Shape Deformation Methods

- Fundamental requirements?
- Different representations
  - Surface meshes
  - Volume meshes
- Defects
  - Badly shaped elements
  - Non-manifold meshes
  - Self-intersections

→ Space deformations
Space Deformation Methods

- Deformation function $d : \mathbb{R}^3 \rightarrow \mathbb{R}^3$
- Warp embedding space around object $\mathcal{M}$
Space Deformation Methods

- Deformation function \( d : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \)
- Warp embedding space around object \( M \)
- Methods:
  - Free-form deformation (FFD)
  - Direct manipulation FFD
  - Radial basis functions

\( M \) \hspace{2cm} \( d: \mathbb{R}^3 \rightarrow \mathbb{R}^3 \) \hspace{2cm} \( M' \)

\( p' = p + d(p) \)
Free-Form Deformation
Free-Form Deformation (FFD)

- Embed object in control lattice
- Compute local coordinates
- Move control points
- Deform object according to updated control points

$\mathcal{M}$

$\mathcal{M}'$

\[
d_{\text{ffd}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3
\]

$p' = p + d_{\text{ffd}}(u)$
Free-Form Deformation: Embedding

\[ p = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} c_{ijk} N_i(u_1) N_j(u_2) N_k(u_3) \]
Free-Form Deformation: Embedding

\[ p = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} c_{ijk} N_i(u_1) N_j(u_2) N_k(u_3) \]
Free-Form Deformation: Embedding

\[
p = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} c_{ijk} N_i(u_1) N_j(u_2) N_k(u_3)
\]
Free-Form Deformation: Embedding

\[ p = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} c_{ijk} N_i(u_1) N_j(u_2) N_k(u_3) \]
Free-Form Deformation: Embedding

\[ \mathbf{p} = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} \mathbf{c}_{ijk} N_i(u_1) N_j(u_2) N_k(u_3) \]

Object point

Control points

Basis functions

Local coordinates
Free-Form Deformation Function

\[ d_{	ext{ffd}}(u) = \sum_{p} \delta c_p N_p(u) \]
Free-Form Deformation Function

\[ d_{ffd}(\mathbf{u}) = \sum_p \delta \mathbf{c}_p N_p(\mathbf{u}) \]

Local coordinates
\[ \mathbf{u} := (u_1, u_2, u_3) \]
Free-Form Deformation Function

Control point displacements

\[ \delta c_p := \delta c_{ijk} = c'_{ijk} - c_{ijk} \]

Local coordinates

\[ u := (u_1, u_2, u_3) \]
Free-Form Deformation Function

Control point displacements

\[ \delta c_p := \delta c_{ijk} = c'_{ijk} - c_{ijk} \]

Local coordinates

\[ u := (u_1, u_2, u_3) \]

Basis functions

\[ N_p(u) := N_i(u_1)N_j(u_2)N_k(u_3) \]

Deformation function

\[ d_{ffd}(u) = \sum_p \delta c_p N_p(u) \]
Free-Form Deformation: Caveats

- Difficult control grid generation
- Numerical coordinate computation
- Tedious control point manipulation
- Only indirect influence on the shape
Free-Form Deformation: Caveats

- Difficult control grid generation
- Numerical coordinate computation
- Tedious control point manipulation
- Only indirect influence on the shape
Direct Manipulation FFD
Direct Manipulation FFD

- Move object points directly
- Automatically compute control point displacements satisfying new object point locations

\[ \mathcal{M} \]

\[ \mathcal{M}' \]

\[ p_{dm} \]

\[ \mathbf{p}' = \mathbf{p} + d_{dmffd}(\mathbf{p}) \]

\[ d_{dmffd} : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \]
Direct Manipulation FFD

- Solve linear system to compute control point displacements:

\[
\begin{bmatrix}
N_1(u_1) & \ldots & N_n(u_1) \\
\vdots & \ddots & \vdots \\
N_1(u_m) & \ldots & N_n(u_m)
\end{bmatrix}
\begin{bmatrix}
\delta c_1^T \\
\vdots \\
\delta c_n^T
\end{bmatrix} =
\begin{bmatrix}
\bar{d}_1^T \\
\vdots \\
\bar{d}_m^T
\end{bmatrix}.
\]

- System is singular → solve using pseudo-inverse
Direct Manipulation FFD

• Solve linear system to compute control point displacements:

\[
\begin{bmatrix}
N_1(u_1) & \ldots & N_n(u_1) \\
\vdots & \ddots & \vdots \\
N_1(u_m) & \ldots & N_n(u_m)
\end{bmatrix}
\begin{bmatrix}
\delta c_1^T \\
\vdots \\
\delta c_n^T
\end{bmatrix}
= 
\begin{bmatrix}
\bar{d}_1^T \\
\vdots \\
\bar{d}_m^T
\end{bmatrix}.
\]

• System is singular \(\rightarrow\) solve using pseudo-inverse

– Minimizes constraint error and control point movement

– Not necessarily physically plausible

– Still requires control lattice
Direct Manipulation FFD

• Solve linear system to compute control point displacements:

\[
\begin{bmatrix}
N_1(u_1) & \ldots & N_n(u_1) \\
\vdots & \ddots & \vdots \\
N_1(u_m) & \ldots & N_n(u_m)
\end{bmatrix}
\begin{bmatrix}
\delta c_1^T \\
\vdots \\
\delta c_n^T
\end{bmatrix}
= 
\begin{bmatrix}
\bar{d}_1^T \\
\vdots \\
\bar{d}_m^T
\end{bmatrix}.
\]

• System is singular → solve using pseudo-inverse

– Minimizes constraint error and control point movement
– Not necessarily physically plausible
– Still requires control lattice
Radial Basis Function Deformation
Radial Basis Function Deformation

• Space deformation as interpolation problem
  - *Exactly* interpolate prescribed displacements
  - *Smoothly* interpolate displacements through space
Radial Basis Function Deformation

- Space deformation as interpolation problem
  - \textit{Exactly} interpolate prescribed displacements
  - \textit{Smoothly} interpolate displacements through space

→ Radial basis functions (RBFs)
Radial Basis Function Deformation

- Space deformation as interpolation problem
  - *Exactly* interpolate prescribed displacements
  - *Smoothly* interpolate displacements through space

→ Radial basis functions (RBFs)

\[
d_{\text{rbf}}(p) = \sum_{j=1}^{m} w_j \varphi_j(p) + \pi(p)
\]
Radial Basis Function Deformation

- Space deformation as interpolation problem
  - *Exactly* interpolate prescribed displacements
  - *Smoothly* interpolate displacements through space

→ Radial basis functions (RBFs)

\[
d_{\text{rbf}}(p) = \sum_{j=1}^{m} w_j \phi_j(p) + \pi(p)
\]
Radial Basis Function Deformation

- Space deformation as interpolation problem
  - *Exactly* interpolate prescribed displacements
  - *Smoothly* interpolate displacements through space

→ Radial basis functions (RBFs)

\[
\begin{align*}
\mathbf{d}_{\text{rbf}}(\mathbf{p}) &= \sum_{j=1}^{m} \mathbf{w}_j \varphi_j(\mathbf{p}) + \pi(\mathbf{p}) \\
\text{Basis functions at centers } \mathbf{c}_j \\
\text{Weights}
\end{align*}
\]
Radial Basis Function Deformation

- Space deformation as interpolation problem
  - *Exactly* interpolate prescribed displacements
  - *Smoothly* interpolate displacements through space

→ Radial basis functions (RBFs)

\[
d_{\text{rbf}}(p) = \sum_{j=1}^{m} w_j \varphi_j(p) + \pi(p)
\]

Basis functions at centers \( c_j \)

Weights

Polynomial term
Radial Basis Function Deformation

• Space deformation as interpolation problem
  - *Exactly* interpolate prescribed displacements
  - *Smoothly* interpolate displacements through space

→ Radial basis functions (RBFs)

\[ d_{\text{rbf}}(p) = \sum_{j=1}^{m} w_j \varphi_j(p) + \pi(p) \]
Radial Basis Functions

- Various choices: Gaussian, multiquadrics, thin plate spline...
Radial Basis Functions

- Various choices: Gaussian, multiquadrics, thin plate spline...
- Choose $\varphi(r) = r^3$ so that $\mathbf{d}$ minimizes fairness energy:

$$\int_{\mathbb{R}^3} \left\| \frac{\partial^3 \mathbf{d}}{\partial x^3} \right\|^2 + \left\| \frac{\partial^3 \mathbf{d}}{\partial x^2 \partial y} \right\|^2 + \ldots + \left\| \frac{\partial^3 \mathbf{d}}{\partial z^3} \right\|^2 \, dV.$$
Radial Basis Functions

- Various choices: Gaussian, multiquadrics, thin plate spline...
- Choose $\varphi(r) = r^3$ so that $d$ minimizes fairness energy:
\[
\int_{\mathbb{R}^3} \left( \left\| \frac{\partial^3 d}{\partial x^3} \right\|^2 + \left\| \frac{\partial^3 d}{\partial x^2 \partial y} \right\|^2 + \ldots + \left\| \frac{\partial^3 d}{\partial z^3} \right\|^2 \right) dV.
\]
- Where to place kernels?
Radial Basis Function Deformation

- Handle-based direct manipulation interface

\( \mathcal{H} \): Handle region

\( \mathcal{D} \): Deformable region

\( \mathcal{F} \): Fixed region
Radial Basis Function Deformation

- Handle-based direct manipulation interface

→ Place kernels in handle and fixed regions

$\mathcal{H}$: Handle region

$D$: Deformable region

$\mathcal{F}$: Fixed region
Radial Basis Function Deformation

- Determine weights and polynomial coefficients
Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

\[
\begin{pmatrix}
\Phi & \Pi \\
\Pi^T & 0
\end{pmatrix}
\begin{pmatrix}
W \\
Q
\end{pmatrix}
= 
\begin{pmatrix}
\bar{D} \\
0
\end{pmatrix}
\]
Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

\[
\begin{pmatrix}
\Phi & \Pi \\
\Pi^T & 0 \\
\end{pmatrix}
\begin{pmatrix}
W \\
Q \\
\end{pmatrix}
= 
\begin{pmatrix}
\bar{D} \\
0 \\
\end{pmatrix}
\]
Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

\[ \Phi_{ij} = \varphi_j(p_i) \]

Basis function weights

\[
\begin{pmatrix}
\Phi & \Pi \\
\Pi^T & 0
\end{pmatrix}
\begin{pmatrix}
W \\
Q
\end{pmatrix}
= 
\begin{pmatrix}
\bar{D} \\
0
\end{pmatrix}
\]

Polynomial coefficients
Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

\[
\begin{bmatrix}
\Phi & \Pi \\
\Pi^T & 0
\end{bmatrix}
\begin{bmatrix}
W \\
Q
\end{bmatrix}
= 
\begin{bmatrix}
\bar{D} \\
0
\end{bmatrix}
\]
Radial Basis Function Deformation

- Determine weights and polynomial coefficients

\[ \Phi \quad \Pi \]

\[ \Pi^T \quad 0 \]

\[ \begin{pmatrix} \Phi & \Pi \\ \Pi^T & 0 \end{pmatrix} \begin{pmatrix} W \\ Q \end{pmatrix} = \begin{pmatrix} \tilde{D} \\ 0 \end{pmatrix} \]

\[ \Phi_{ij} = \varphi_j(p_i) \]

\[ \Pi_{ij} = \pi_j(p_i) \]

Basis function weights

Polynomial coefficients

Constraints
RBF Deformation

- Smooth and physically plausible
- Satisfies constraints exactly
- No control lattice, flexible setup

\[ d_{rbf} : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \]

\[ p' = p + d_{rbf}(p) \]
Evolutionary Design Optimization
Evolutionary Algorithms

+ Global optimization
+ Generate novel designs
+ Robustness to noise
+ Non-smooth, multi-objective target functions
Evolutionary Algorithms

+ Global optimization
+ Generate novel designs
+ Robustness to noise
+ Non-smooth, multi-objective target functions
  - Computationally expensive
Evolution Strategies (ES)

- Represent solutions as vectors of real numbers
- Create offspring by adding zero mean random vector
- Advantages:
  - Self-adaptation of strategy parameters during optimization
  - Simple incorporation of constraints
Covariance Matrix Adaptation ES

- Adapt covariance matrix to previously successful solutions
  + Fast convergence on small population sizes
Passenger Car Design Optimization
Passenger Car Design Optimization

Goal: Improve aerodynamic drag of a simplified Honda Civic

- Initial Design
- Parent Chromosomes
- Encoding
- Reproduction & Mutation
- Offspring Chromosomes
- Selection
- CFD Simulation
- Genotype-Phenotype Mapping
- Evaluation
- Final Selection
- Optimized Design

Diagram sequence:
1. Initial Design → Encoding → Parent Chromosomes
2. Reproduction & Mutation → Offspring Chromosomes
3. Selection
4. CFD Simulation
5. Genotype-Phenotype Mapping
6. Evaluation
7. Final Selection → Optimized Design
Deformation Setups
Fitness Function Evaluation

• Fitness function $f : \mathbb{R}^2 \rightarrow \mathbb{R}$:
  \[ f(x) = w_1 v_1 + w_2 v_2. \]

• $v_1$: aerodynamic drag computed by CFD simulation

• $v_2$: volume weight to penalize overly flat shapes
Results
Conclusions & Future Work
Conclusions & Future Work

• Conclusions:
  - RBFs: Flexible setup with equivalent or better results
  - Strong coupling is important

• Future work:
  - Additional methods
  - Unified interface
  - Synthetic benchmarks
Conclusions & Future Work

• Conclusions:
  - RBFs: Flexible setup with equivalent or better results
  - Strong coupling is important

• Future work:
  - Additional methods
  - Unified interface
  - Synthetic benchmarks
Thanks

... for your attention.

Questions?