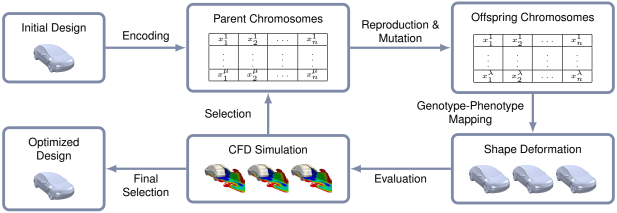
A Comprehensive Comparison of Shape Deformation Methods in Evolutionary Design Optimization



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

[1] Graphics & Geometry Group, Bielefeld University

[2] Honda Research Institute

Universität Bielefeld

Honda Research Institute EU

CITEC

Outline

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

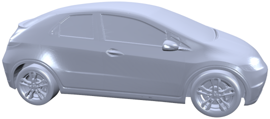
Shape Deformation Methods

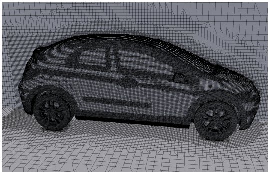
Shape Deformation Methods

- Fundamental requirements?

Shape Deformation Methods

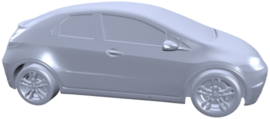
- Fundamental requirements?
- Different representations
  - Surface meshes
  - Volume meshes

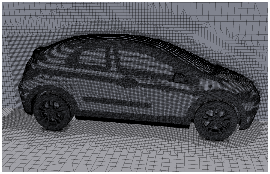


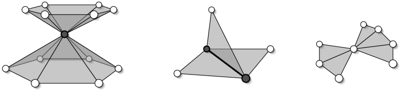


Shape Deformation Methods

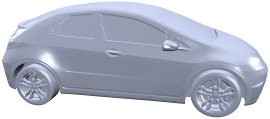
- Fundamental requirements?
- Different representations
  - Surface meshes
  - Volume meshes
- Defects
  - Badly shaped elements
  - Non-manifold meshes
  - Self-intersections

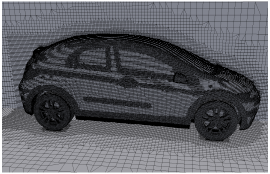




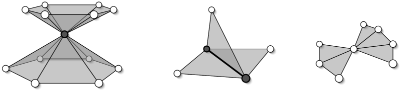


Shape Deformation Methods





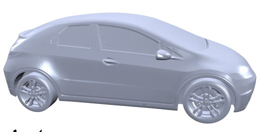
- Fundamental requirements?
- Different representations
  - Surface meshes
  - Volume meshes
- Defects
  - Badly shaped elements
  - Non-manifold meshes
  - Self-intersections



→ *Space deformations*

Space Deformation Methods

- Deformation function $\mathbf{d}\colon \mathbb{R}^3 \to \mathbb{R}^3$
- Warp embedding space around object $\mathcal{M}$



$\mathcal{M}$

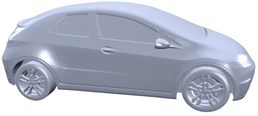
$$\mathbf{d}\colon \mathbb{R}^3 \to \mathbb{R}^3$$

$$\mathbf{p}' = \mathbf{p} + \mathbf{d}(\mathbf{p})$$

$\mathcal{M}'$

Space Deformation Methods

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



$\mathcal{M}$

$$\mathbf{d}\colon \mathbb{R}^3 \to \mathbb{R}^3$$

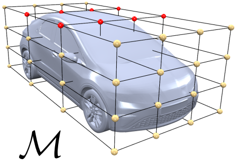
$$\mathbf{p}' = \mathbf{p} + \mathbf{d}(\mathbf{p})$$

$\mathcal{M}'$

Free-Form Deformation

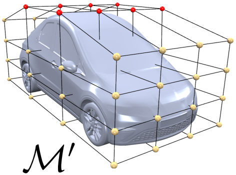
Free-Form Deformation (FFD)

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



$$\mathbf{d}_{\text{ffd}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

$$\mathbf{p}' = \mathbf{p} + \mathbf{d}_{\text{ffd}}(\mathbf{u})$$



Free-Form Deformation: Embedding

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

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

Free-Form Deformation: Embedding

Control points

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

Object point

Free-Form Deformation: Embedding

Control points

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

Object point

Basis functions

Free-Form Deformation: Embedding

Control points

Local coordinates

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

Object point

Basis functions

Free-Form Deformation Function

$$\mathbf{d}_{\text{ffd}}(\mathbf{u}) \; = \; \sum_p \delta \mathbf{c}_p N_p(\mathbf{u})$$

Free-Form Deformation Function

$$\mathbf{d}_{\text{ffd}}(\mathbf{u}) \;=\; \sum_{p} \delta\mathbf{c}_p N_p(\mathbf{u})$$

Local coordinates
$\mathbf{u} := (u_1, u_2, u_3)$

Free-Form Deformation Function

Control point displacements
$\delta \mathbf{c}_p := \delta \mathbf{c}_{ijk} = \mathbf{c}'_{ijk} - \mathbf{c}_{ijk}$

$$\mathbf{d}_{\text{ffd}}(\mathbf{u}) = \sum_p \delta \mathbf{c}_p N_p(\mathbf{u})$$

Local coordinates
$\mathbf{u} := (u_1, u_2, u_3)$

Free-Form Deformation Function

Control point displacements

$$\delta \mathbf{c}_p := \delta \mathbf{c}_{ijk} = \mathbf{c}'_{ijk} - \mathbf{c}_{ijk}$$

$$\mathbf{d}_{\text{ffd}}(\mathbf{u}) = \sum_p \delta \mathbf{c}_p N_p(\mathbf{u})$$

Local coordinates

$$\mathbf{u} := (u_1, u_2, u_3)$$

Basis functions

$$N_p(\mathbf{u}) := N_i(u_1) N_j(u_2) N_k(u_3)$$

Free-Form Deformation: Caveats

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

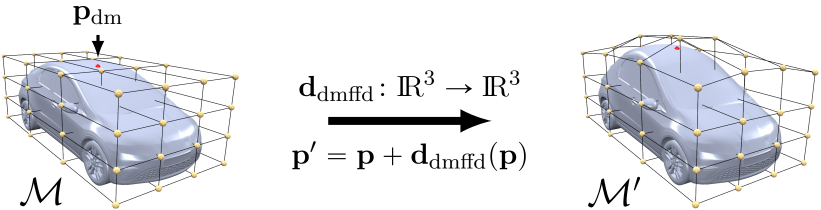
Free-Form Deformation: Caveats

- 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



Direct Manipulation FFD

- Solve linear system to compute control point displacements:

$$\begin{bmatrix} N_1(\mathbf{u}_1) & \dots & N_n(\mathbf{u}_1) \\ \vdots & \ddots & \vdots \\ N_1(\mathbf{u}_m) & \dots & N_n(\mathbf{u}_m) \end{bmatrix} \begin{pmatrix} \delta\mathbf{c}_1^T \\ \vdots \\ \delta\mathbf{c}_n^T \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{d}}_1^T \\ \vdots \\ \bar{\mathbf{d}}_m^T \end{pmatrix}.$$

- System is singular → solve using pseudo-inverse

Direct Manipulation FFD

- Solve linear system to compute control point displacements:

$$\begin{bmatrix} N_1(\mathbf{u}_1) & \dots & N_n(\mathbf{u}_1) \\ \vdots & \ddots & \vdots \\ N_1(\mathbf{u}_m) & \dots & N_n(\mathbf{u}_m) \end{bmatrix} \begin{pmatrix} \delta\mathbf{c}_1^T \\ \vdots \\ \delta\mathbf{c}_n^T \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{d}}_1^T \\ \vdots \\ \bar{\mathbf{d}}_m^T \end{pmatrix}.$$

- System is singular → 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:

$$\begin{bmatrix} N_1(\mathbf{u}_1) & \dots & N_n(\mathbf{u}_1) \\ \vdots & \ddots & \vdots \\ N_1(\mathbf{u}_m) & \dots & N_n(\mathbf{u}_m) \end{bmatrix} \begin{pmatrix} \delta\mathbf{c}_1^T \\ \vdots \\ \delta\mathbf{c}_n^T \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{d}}_1^T \\ \vdots \\ \bar{\mathbf{d}}_m^T \end{pmatrix}.$$

- 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
  - *Exactly* interpolate prescribed displacements
  - *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)

$$\mathbf{d}_{\mathrm{rbf}}(\mathbf{p}) \;=\; \sum_{j=1}^{m} \mathbf{w}_j \varphi_j(\mathbf{p}) + \boldsymbol{\pi}(\mathbf{p})$$

Radial Basis Function Deformation

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

→ Radial basis functions (RBFs)

Basis functions at centers $\mathbf{c}_j$

$$\mathbf{d}_{\mathrm{rbf}}(\mathbf{p}) = \sum_{j=1}^{m} \mathbf{w}_j \varphi_j(\mathbf{p}) + \boldsymbol{\pi}(\mathbf{p})$$

Radial Basis Function Deformation

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

→ Radial basis functions (RBFs)

Basis functions at centers $\mathbf{c}_j$

$$\mathbf{d}_{\mathrm{rbf}}(\mathbf{p}) = \sum_{j=1}^{m} \mathbf{w}_j \varphi_j(\mathbf{p}) + \boldsymbol{\pi}(\mathbf{p})$$

Weights

Radial Basis Function Deformation

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

→ Radial basis functions (RBFs)

Basis functions at centers $\mathbf{c}_j$

$$\mathbf{d}_{\mathrm{rbf}}(\mathbf{p}) = \sum_{j=1}^{m} \mathbf{w}_j \varphi_j(\mathbf{p}) + \boldsymbol{\pi}(\mathbf{p})$$

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)

Basis functions at centers $\mathbf{c}_j$

$$\mathbf{d}_{\mathrm{rbf}}(\mathbf{p}) = \sum_{j=1}^{m} \mathbf{w}_j \varphi_j(\mathbf{p}) + \boldsymbol{\pi}(\mathbf{p})$$

Weights

Polynomial term

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:

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

Radial Basis Functions

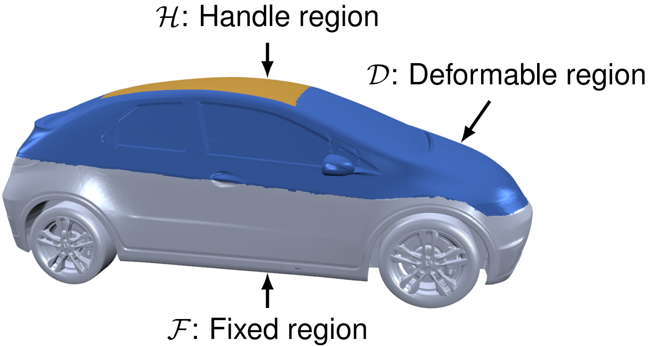
- Various choices: Gaussian, multiquadrics, thin plate spline...
- Choose $\varphi(r) = r^3$ so that $\mathbf{d}$ minimizes fairness energy:

$$\int_{\mathrm{IR}^3} \left\| \frac{\partial^3 \mathbf{d}}{\partial x^3} \right\|^2 + \left\| \frac{\partial^3 \mathbf{d}}{\partial x^2 \partial y} \right\|^2 + \ldots + \left\| \frac{\partial^3 \mathbf{d}}{\partial z^3} \right\|^2 \mathrm{dV}.$$

- Where to place kernels?

Radial Basis Function Deformation

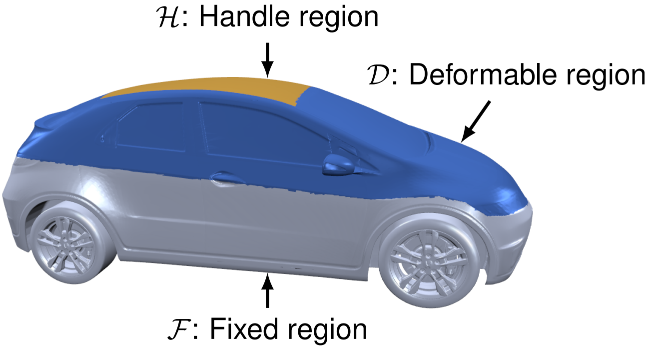
- Handle-based direct manipulation interface



Radial Basis Function Deformation

- Handle-based direct manipulation interface

→ Place kernels in handle and fixed regions



Radial Basis Function Deformation

- Determine weights and polynomial coefficients

Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

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

Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

Basis function weights

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

Polynomial coefficients

Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

$\Phi_{ij} = \varphi_j(\mathbf{p}_i)$

Basis function weights

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

Polynomial coefficients

Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

$\mathbf{\Phi}_{ij} = \varphi_j(\mathbf{p}_i)$

Basis function weights

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

$\mathbf{\Pi}_{ij} = \pi_j(\mathbf{p}_i)$

Polynomial coefficients

Radial Basis Function Deformation

- Determine weights and polynomial coefficients

→ Solve linear system

$\mathbf{\Phi}_{ij} = \varphi_j(\mathbf{p}_i)$

Basis function weights

Constraints

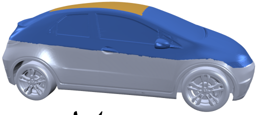
$$\begin{pmatrix} \mathbf{\Phi} & \mathbf{\Pi} \\ \mathbf{\Pi}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{W} \\ \mathbf{Q} \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{D}} \\ \mathbf{0} \end{pmatrix}$$

$\mathbf{\Pi}_{ij} = \pi_j(\mathbf{p}_i)$

Polynomial coefficients

RBF Deformation

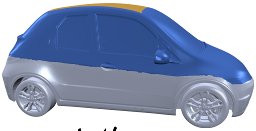
- + Smooth and physically plausible
- + Satisfies constraints exactly
- + No control lattice, flexible setup



$\mathcal{M}$

$$\mathbf{d}_{\mathrm{rbf}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

$$\mathbf{p}' = \mathbf{p} + \mathbf{d}_{\mathrm{rbf}}(\mathbf{p})$$



$\mathcal{M}'$

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

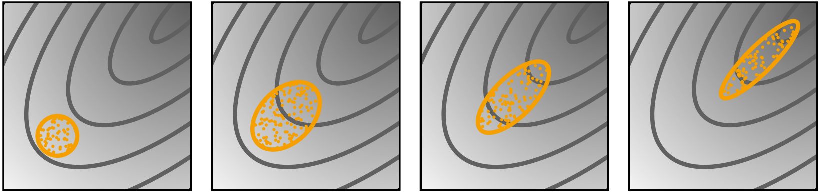
Evolution Strategies (ES)

- 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

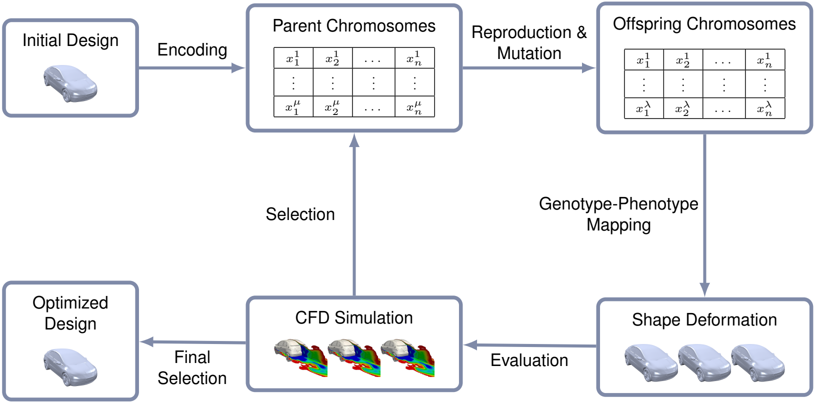
+ Fast convergence on small population sizes



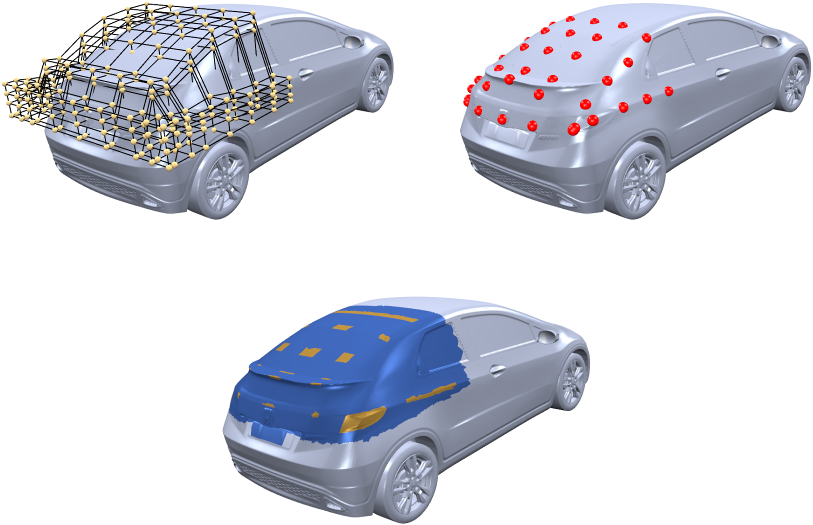
Passenger Car
Design Optimization

Passenger Car Design Optimization

Goal: Improve aerodynamic drag of a simplified Honda Civic



Deformation Setups



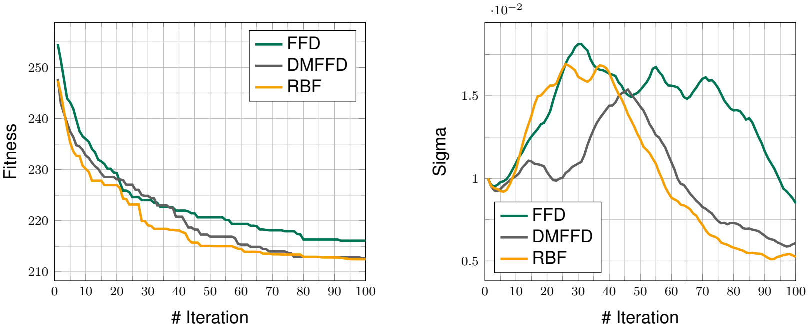
Fitness Function Evaluation

- Fitness function $f\colon \mathbb{R}^2 \to \mathbb{R}$:

$$f(\mathbf{x}) = w_1 v_1 + w_2 v_2.$$

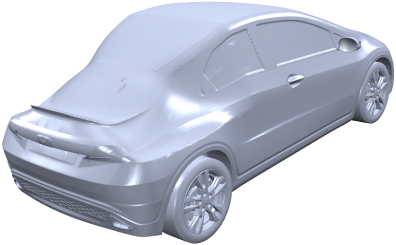
- v_1: aerodynamic drag computed by CFD simulation
- v_2: volume weight to penalize overly flat shapes

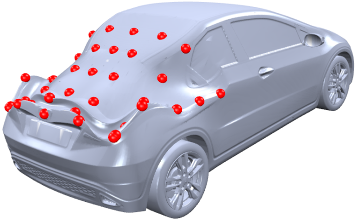
Results

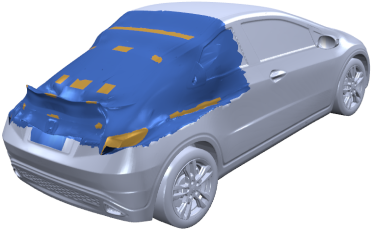


FFD
DMFFD
RBF
Fitness
250
240
230
220
210
0
10
20
30
40
50
60
70
80
90
100
Iteration
·10^−2
Sigma
1.5
1
0.5
FFD
DMFFD
RBF
0
10
20
30
40
50
60
70
80
90
100
Iteration

Results







Conclusions & Future Work

Conclusions & Future Work

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

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?