Inverse Computational Spectral Geometry

Arianna Rampini Riccardo Marin Simone Melzi Luca Cosmo
Emanuele Rodolà Maks Ovsjanikov Michael Bronstein

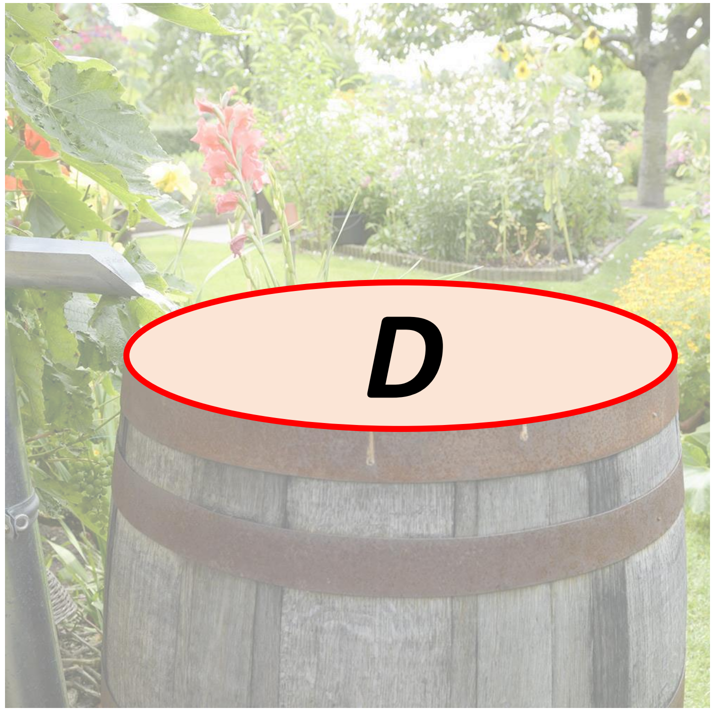
SAPIENZA
UNIVERSITÀ DI ROMA

GLADIA

Introduction

Wave equation

Wave equation



Wave equation

The wave equation for the height $f(x, y, t)$ of the water at point *(x,y)* after time *t*:

Wave equation

The wave equation for the height $f(x, y, t)$ of the water at point *(x,y)* after time *t*:

$$\Delta f = -\frac{1}{c^2} \frac{\partial^2 f}{\partial t^2}$$

speed of sound
in the fluid

First-order approximation of the motions under consideration.

Vibrating membrane equation

The wave equation for the normal motion $f(x, y, t)$ of a vibrating membrane («drum»):

$$\Delta f = -\frac{1}{c^2} \frac{\partial^2 f}{\partial t^2}$$

speed of sound
in the membrane

First-order approximation of sounds in a flat object.


SCI HD

Why the eigenvalue problem?

$$\Delta f = -\frac{1}{c^2}\frac{\partial^2 f}{\partial t^2}$$

To solve for *f*, we need only consider product functions:

$$f(x, y, t) = \phi(x, y)h(t)$$

spatial component

temporal component

Why the eigenvalue problem?

$$\Delta f = -\frac{\partial^2 f}{\partial t^2}$$

$$f(x,y,t) = \phi(x,y)h(t)$$

Laplacian eigenfunction

oscillating functions with frequency λ

$$\Delta \phi h = -\frac{\partial^2 \phi h}{\partial t^2}$$

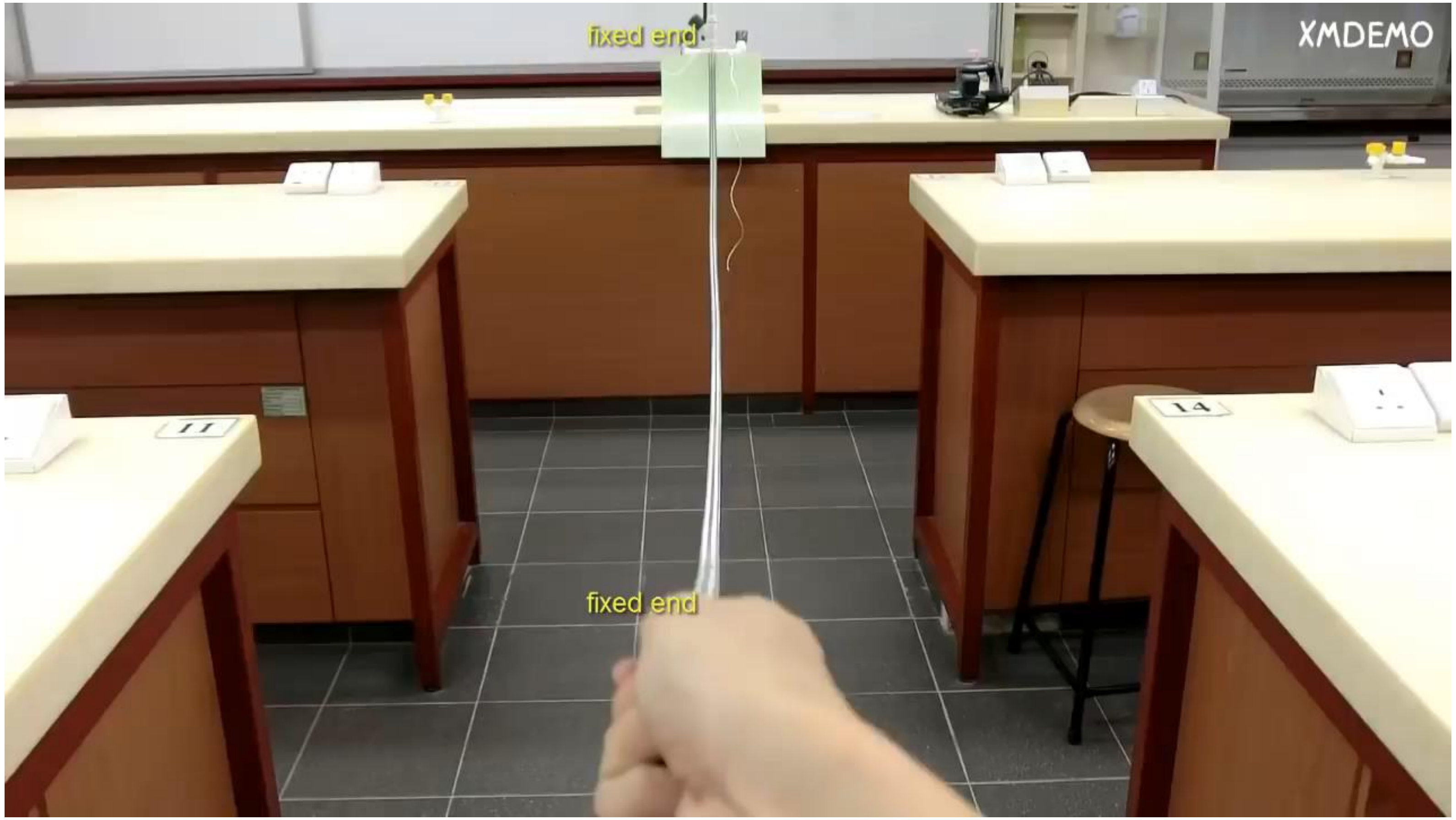
$$\frac{\Delta \phi}{\phi} = -\frac{h''}{h}$$

$$\lambda = \lambda \rightarrow -\frac{h''}{h} = \lambda \rightarrow h(t) = e^{it\sqrt{\lambda}}$$

$$\Delta \phi = \lambda \phi$$

Stationary waves

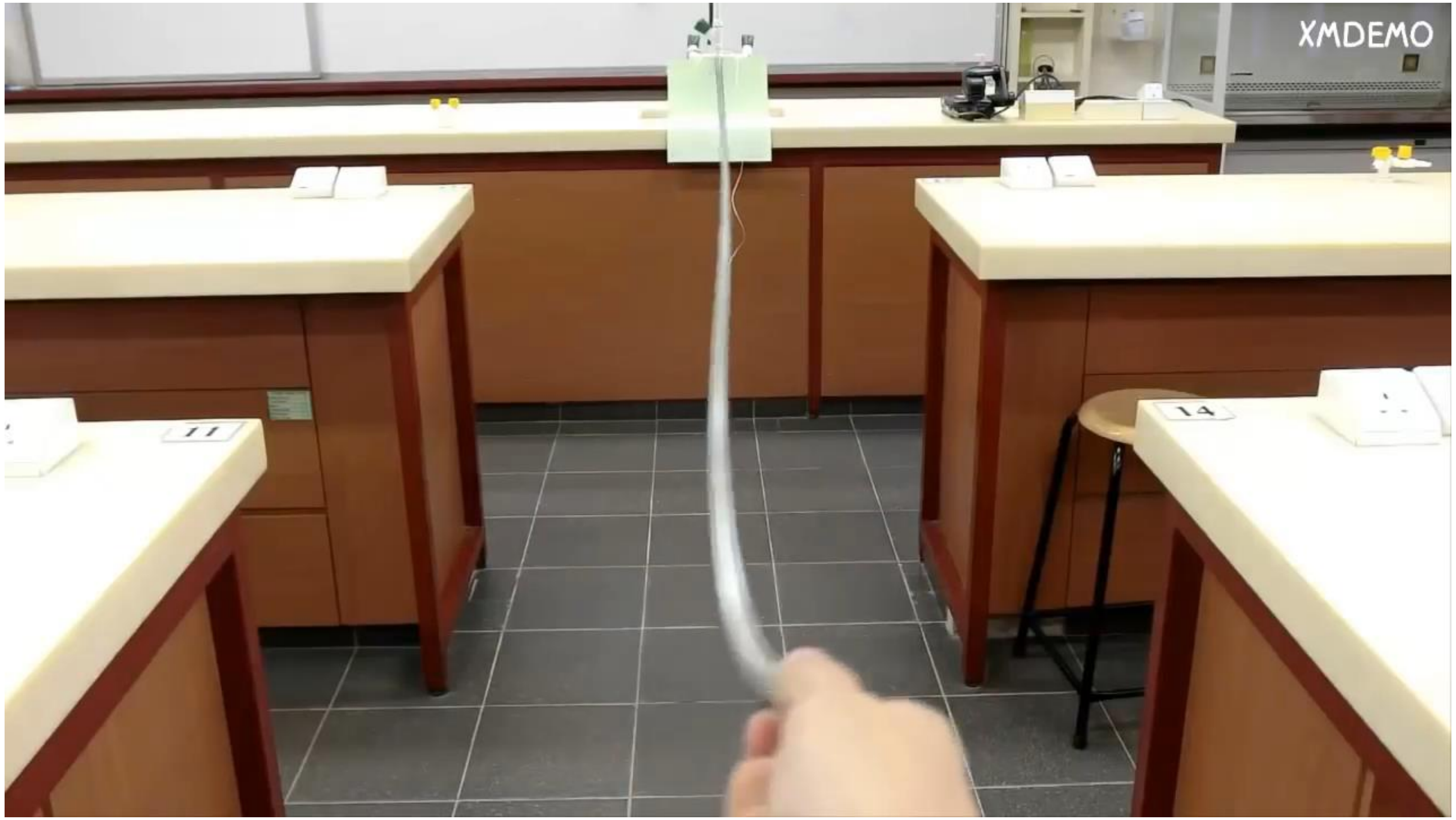
Physically, the product motions $f(x, y, t) = \phi(x, y)h(t)$ are **stationary**.



Video: Chua Kah Hean, 2016

Stationary waves

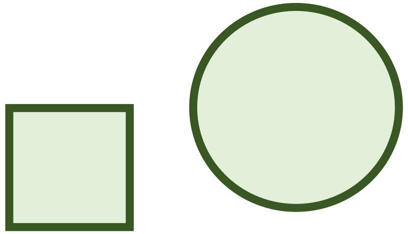
Physically, the product motions $f(x, y, t) = \phi(x, y)h(t)$ are **stationary**.



Video: Chua Kah Hean, 2016

Whispering galleries

Behavior is not always easy to grasp even on simple domains.



<u>Example:</u>

On the disk, there is high concentration along the boundary («whispering gallery effect»)

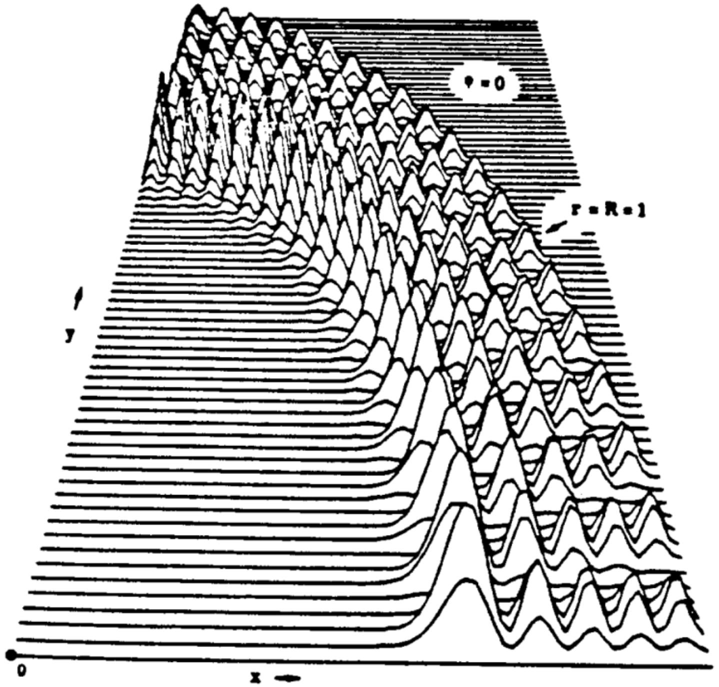


Figure: Sarnak, 1995

Voltone del Podestà, Bologna (Italy)

Computing eigenvalues

$$\Delta\phi_i = \lambda_i\phi_i$$

Very few examples where the spectrum can be determined explicitly.

«As a shocking example of our ignorance, we know nothing about regular hexagons, not even the first eigenvalue.»

[Marcel Berger, 2002]

Our drums

Direct and inverse problems

Given the (approximate) **shape** of a domain *D*,
what can I deduce about its **spectrum**?
(spectral geometry)

Given the (approximate) **spectrum** of a domain *D*,
what can I deduce about its **shape**?
(inverse spectral geometry)

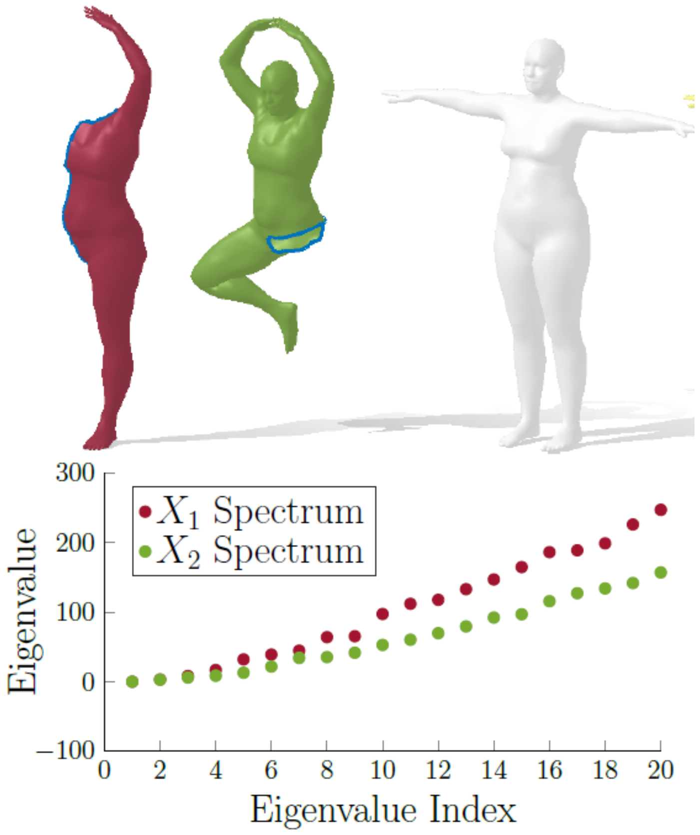
Direct problems

- Asymptotic expansion of the counting function:

$$N(\lambda) = \#\{\lambda_i \leq \lambda\}$$

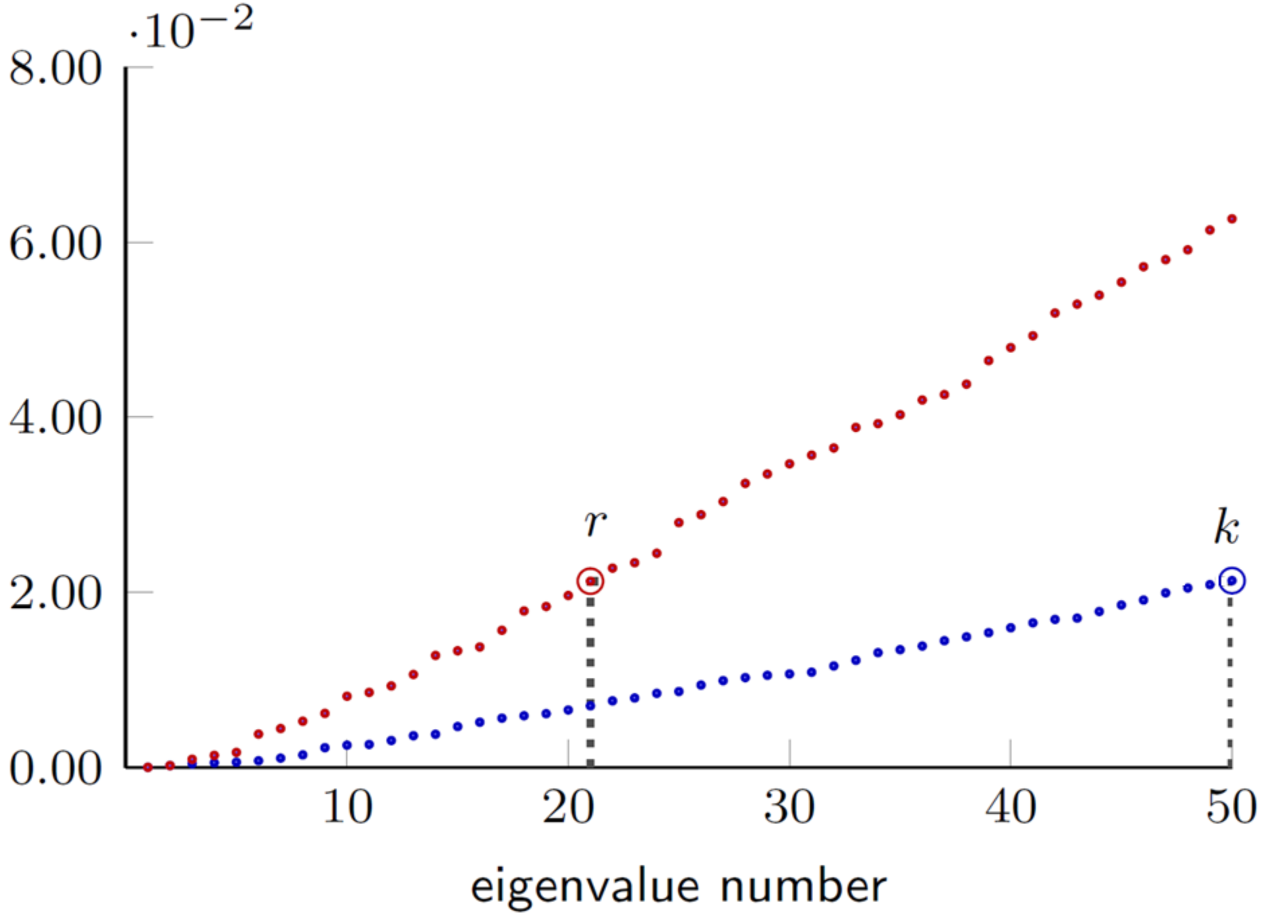
- Tight estimates of λ_1

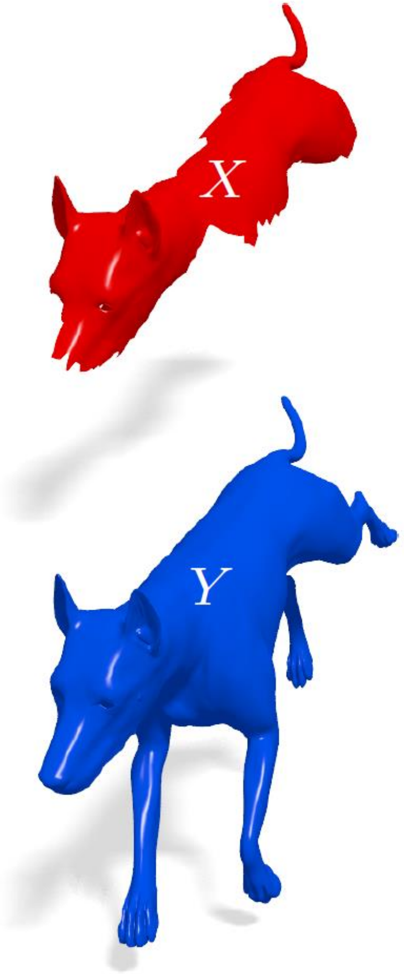
- Relation between eigenvalues of D and those of a sub-domain $P \subset D$



[Moschella et al 2021]

Inverse problems

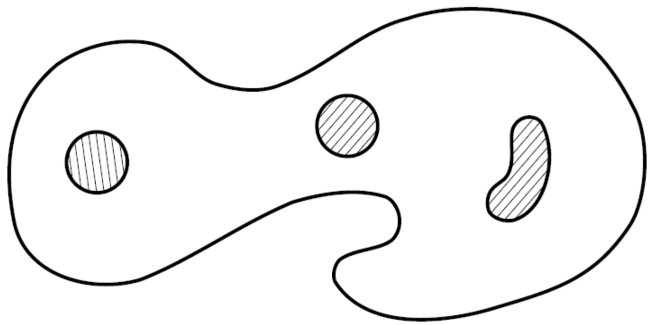




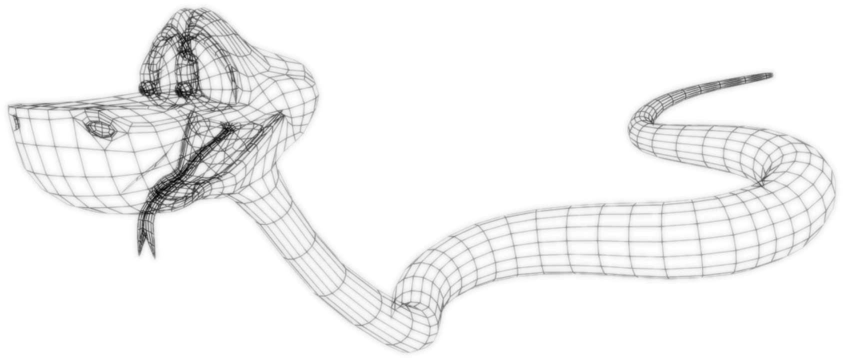
- Compute the area, perimeter, and number of holes in a shape from its eigenvalues.

Inverse problems

- Compute the area, perimeter, and number of holes in a shape from its eigenvalues.

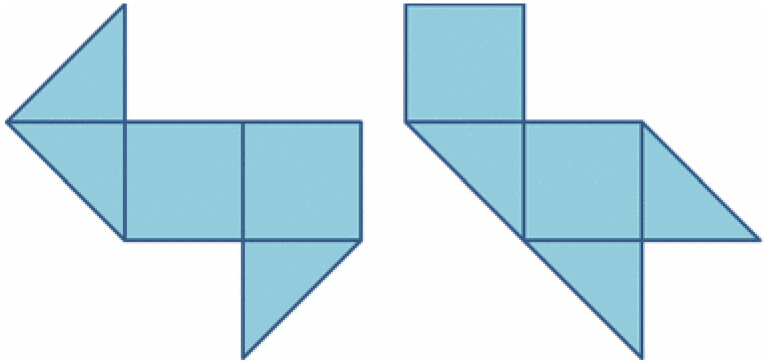


- Recover a 3D shape from its eigenvalues ~~and eigenfunctions.~~



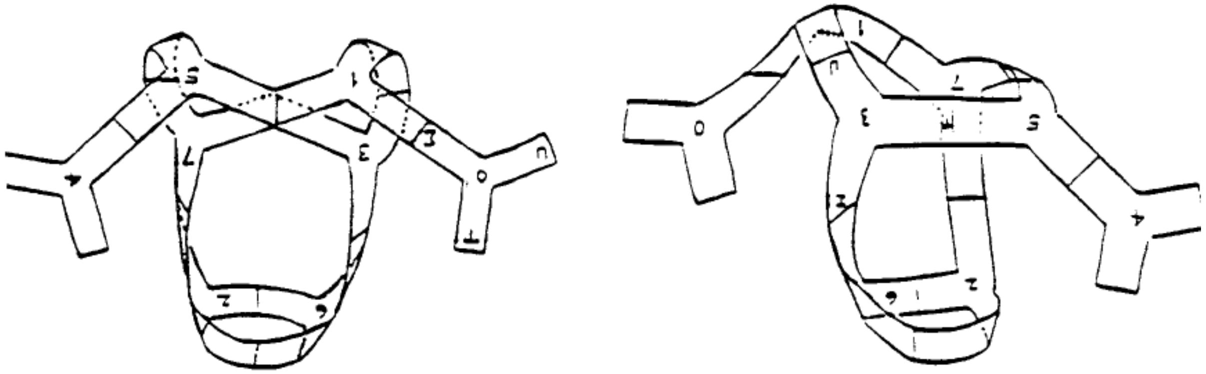
Isospectral domains

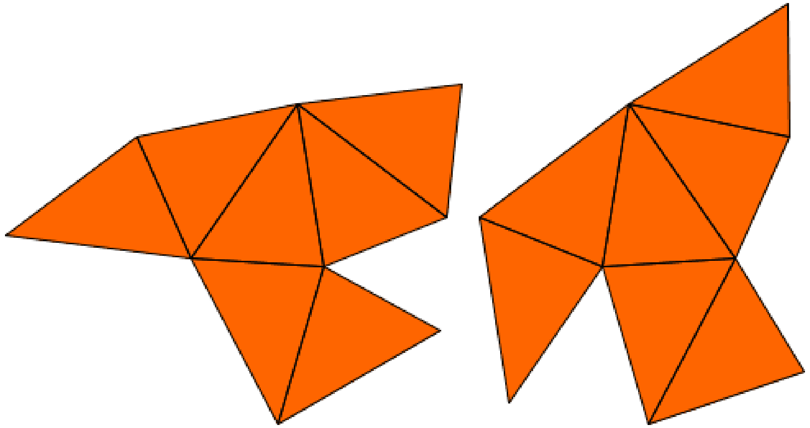
Are eigenvalues enough?



- Conjecture: yes! [Gel'fand, 1962]
- Counterexample: no! [Milnor, 1964; Gordon et al, 1992]

Except for notable exceptions (disks, spheres), in general, shapes are not fully characterized by their spectrum.





Can it still be useful in practice?

Mathematically, the problem is beyond reach today.

Yet, in the Middle Ages, bell makers detected invisible cracks by tolling the bell.

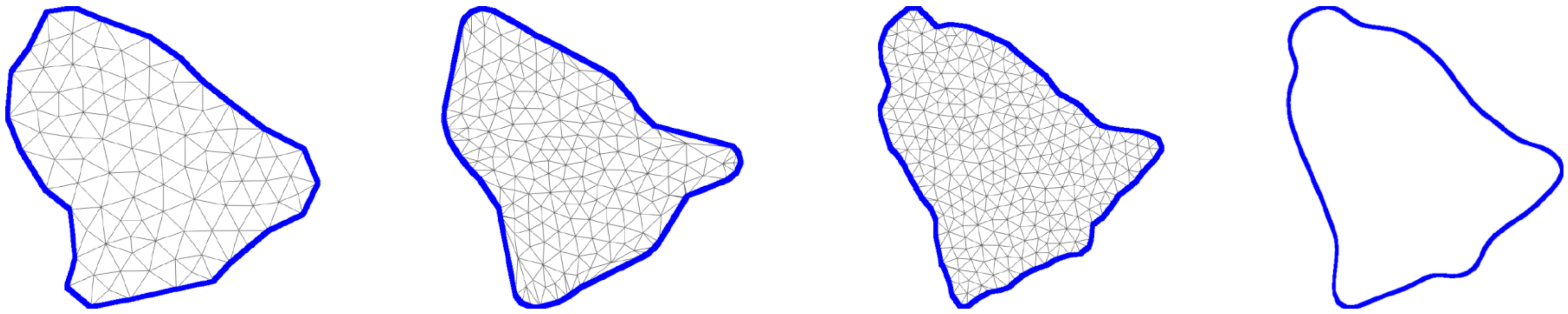
Antonio Delli Quadri, whose family is in the bell-making business since the 14th century

"This is a complex trade that involves precise understanding of mathematics, physics, **geometry** and music"

Can it still be useful in practice?

Mathematically, the problem is beyond reach today.

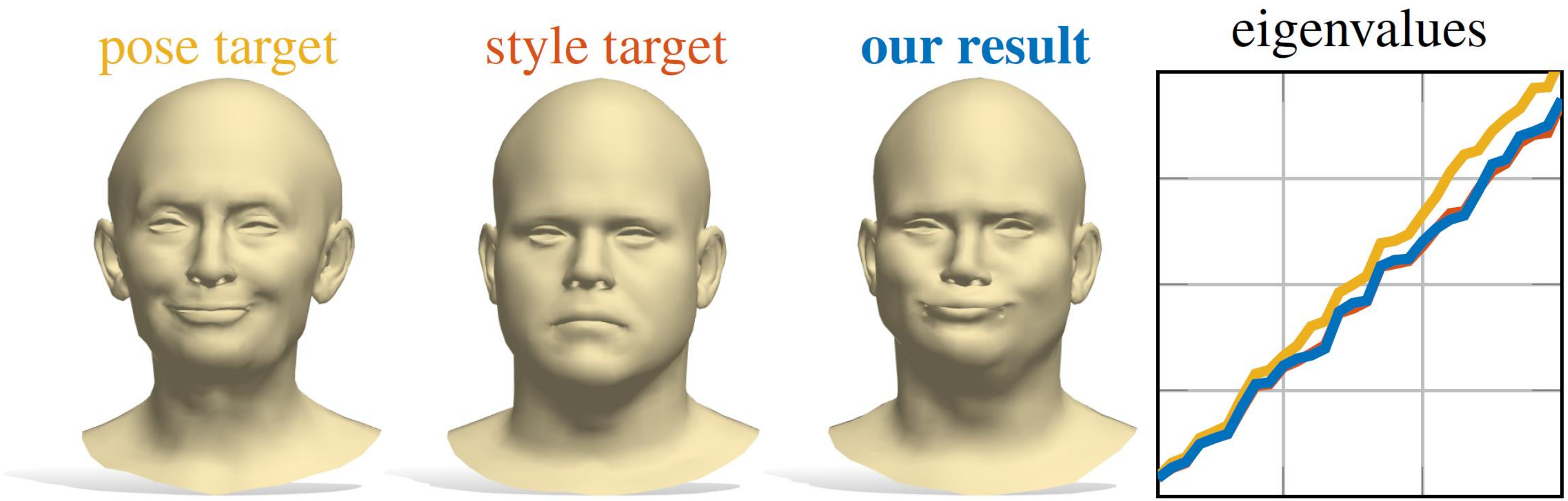
"This is a complex trade that involves precise understanding of mathematics, physics, **geometry** and music"



Can it still be useful in practice?

Mathematically, the problem is beyond reach today.

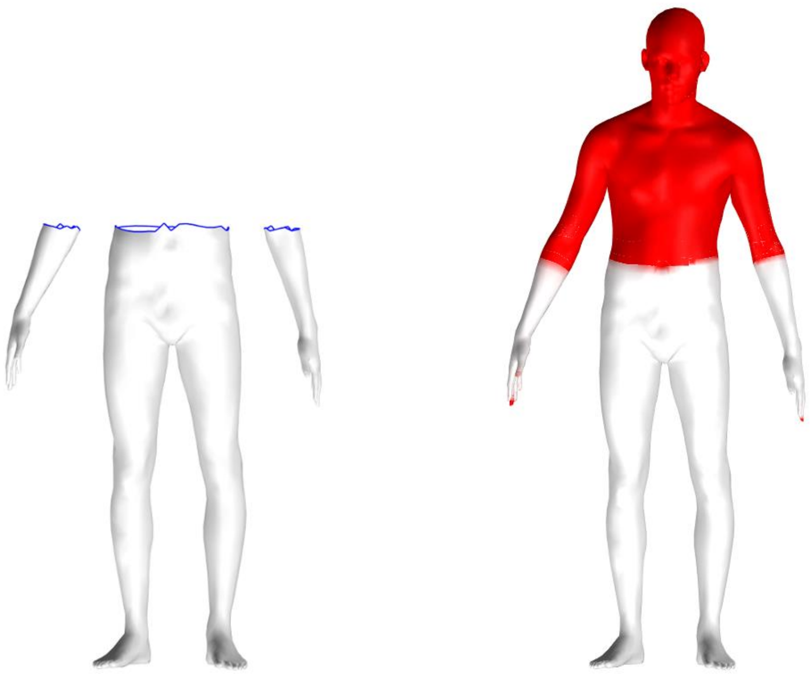
"This is a complex trade that involves precise understanding of mathematics, physics, **geometry** and music"

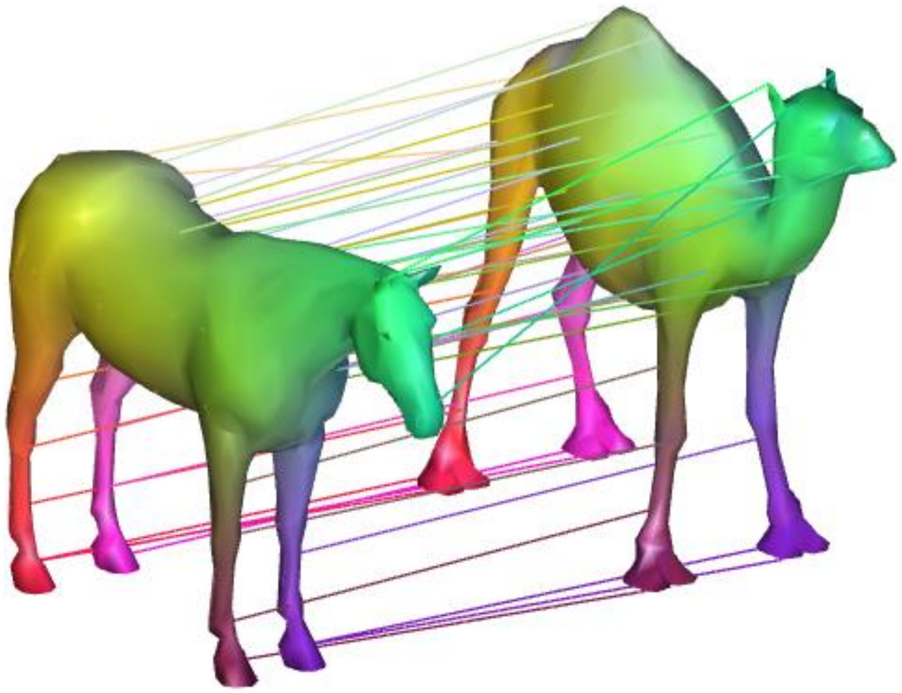


Can it still be useful in practice?

Mathematically, the problem is beyond reach today.

"This is a complex trade that involves precise understanding of mathematics, physics, **geometry** and music"





Inverse Computational Spectral Geometry

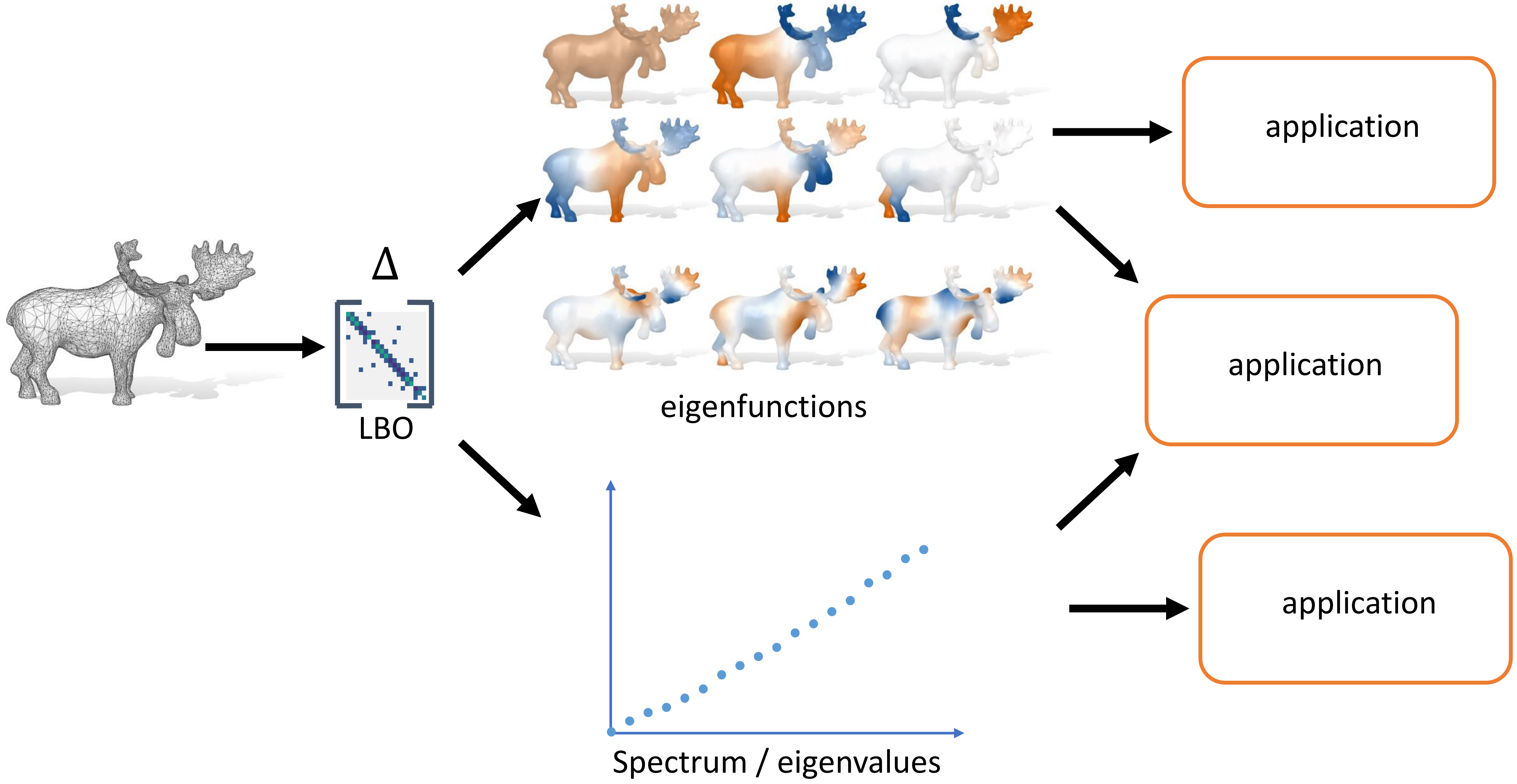
Arianna Rampini Riccardo Marin **Simone Melzi** Luca Cosmo
Emanuele Rodolà Maks Ovsjanikov Michael Bronstein

SAPIENZA UNIVERSITÀ DI ROMA

GLADIA

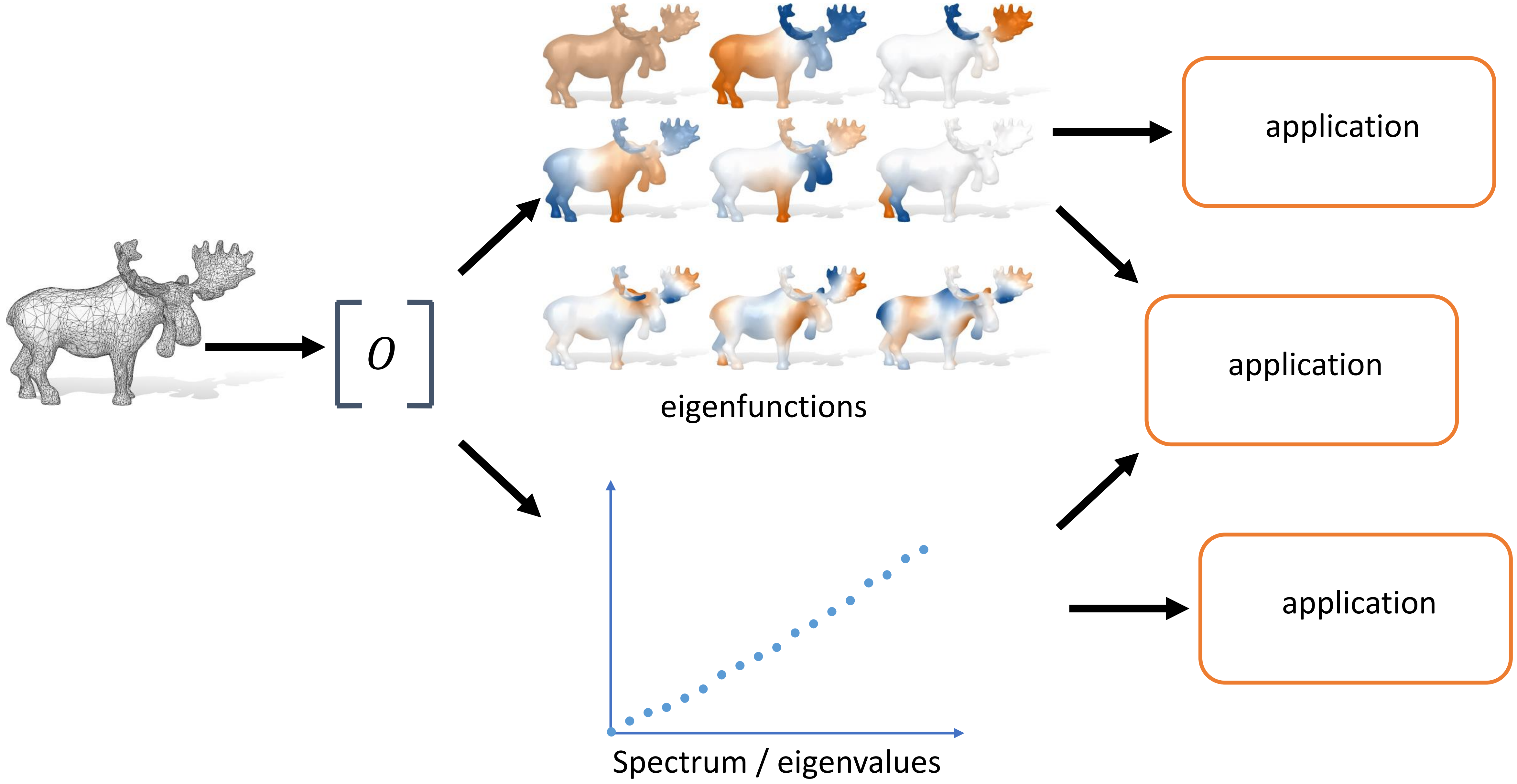
Applications of forward problem

Forward



Δ
LBO
eigenfunctions
Spectrum / eigenvalues
application
application
application

Forward

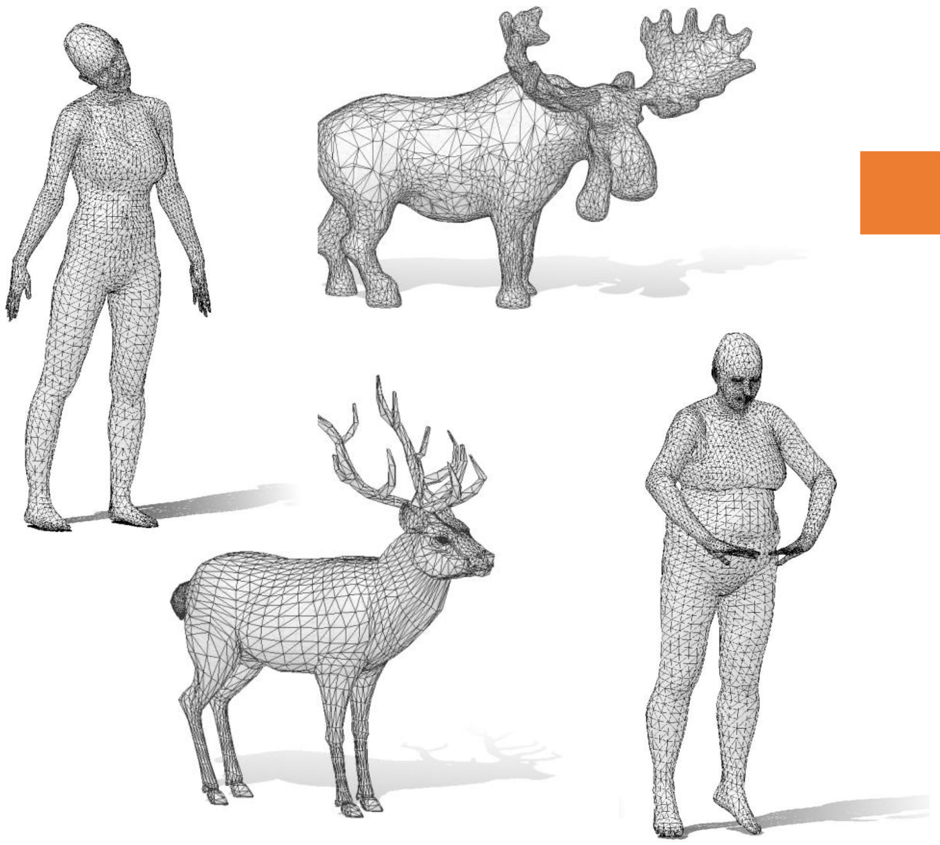


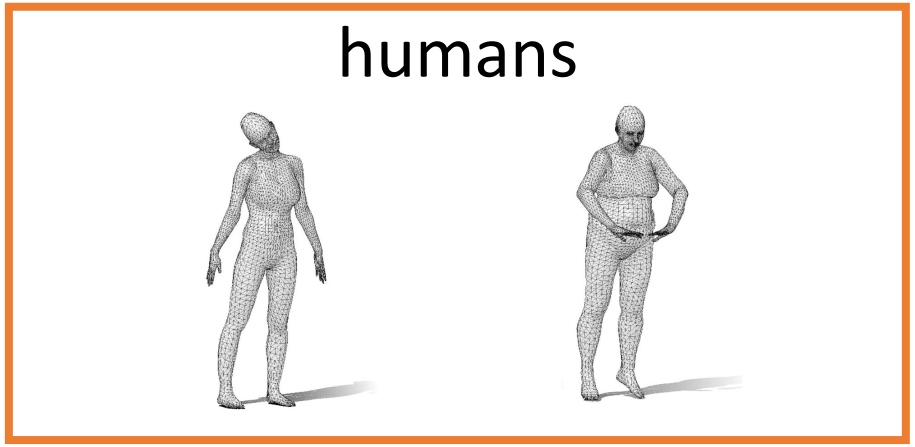
O
eigenfunctions
Spectrum / eigenvalues
application
application
application

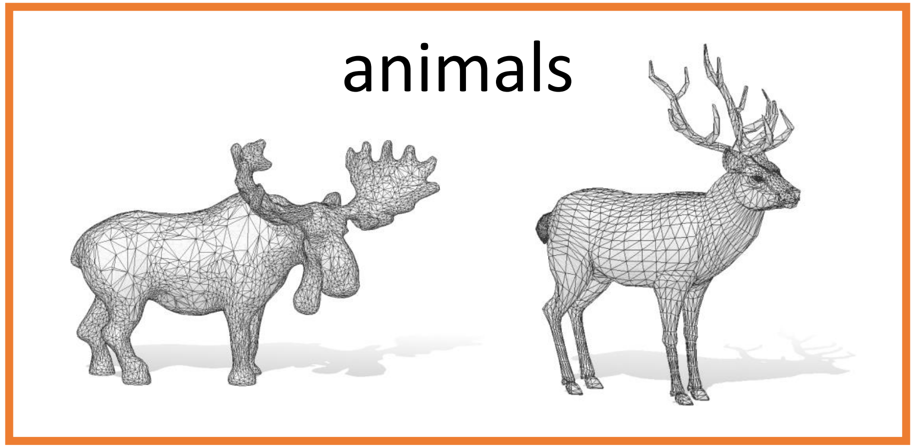
Shape Retrieval

Shape DNA

Given a collection of shapes → **Subdivide** them in the groups of **most similar**







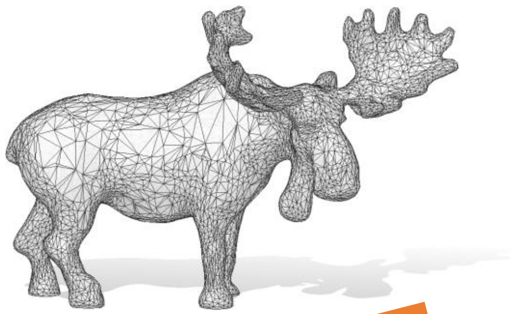
Reuter et al. «Laplace-Beltrami spectra as 'Shape-DNA' of surfaces and solids», 2005.

Shape DNA

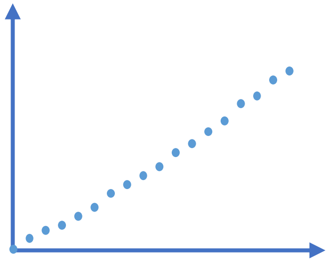
The set of eigenvalues:

$$\Lambda = [\lambda_1, \dots, \lambda_k] \in \mathbb{R}^k$$

A **global signature** of the shape

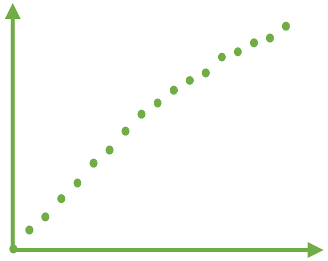


~~hard to compare~~



Spectrum

easy to compare (Euclidean)



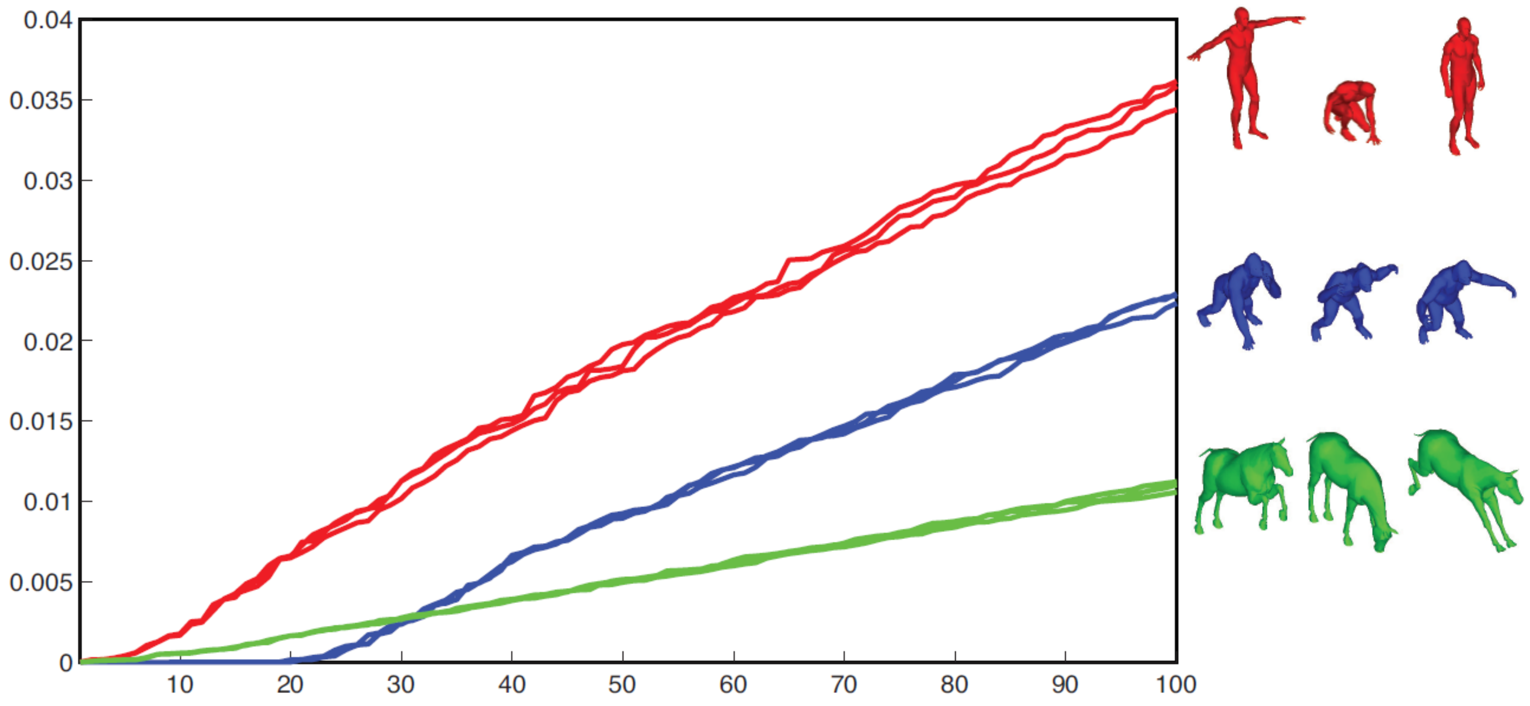
Spectrum

Reuter et al. «Laplace-Beltrami spectra as 'Shape-DNA' of surfaces and solids», 2005.

Shape DNA algorithm

For each shape in the collection:

1. Compute the LBO
2. Compute the set of the first k eigenvalues of the LBO
3. Compare the shapes by comparing the vectors of the eigenvalues



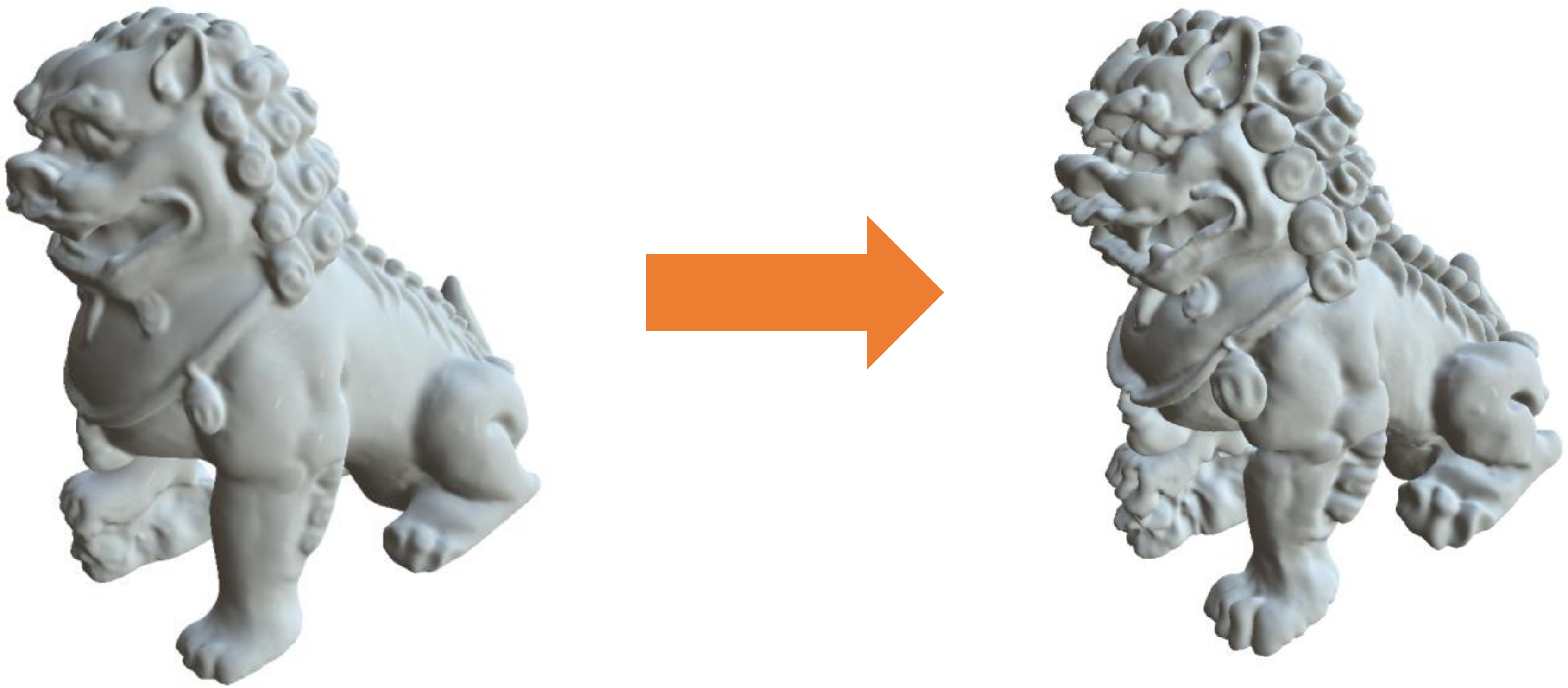
- normalizations
- choices of k

Geometry filtering

Frequency filtering

Given an input shape → **Modify its geometry**

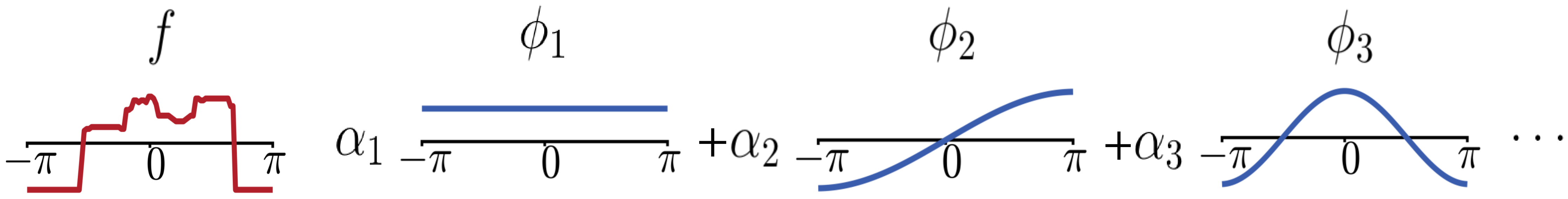
Avoiding the direct editing of the vertex positions



Vallet and Levy: «Spectral Geometry Processing with Manifold Harmonics», 2008.

Fourier

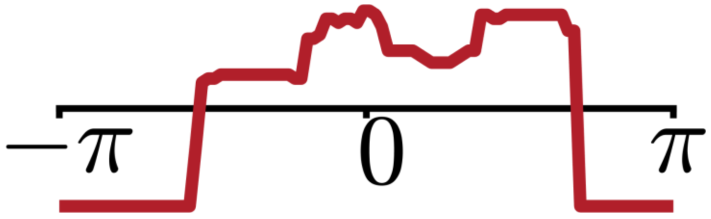
The Fourier basis functions = eigenfunctions of the Laplacian



Sorted w.r.t. the frequecncies = the square root of the Laplacian eigenvalues

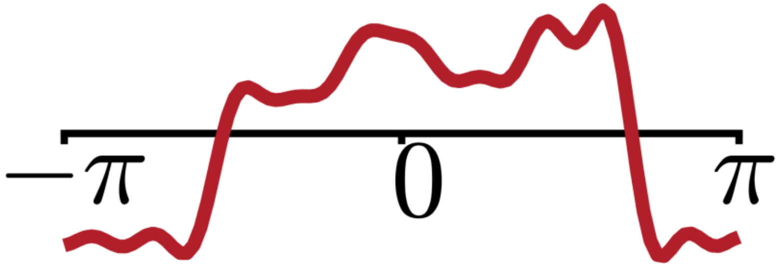
Fourier analysis and synthesis

Given a signal: f



The analysis: $$\alpha_l = \langle f, \phi_l \rangle = \int_{-\pi}^{\pi} f(x)\phi_l(x)dx$$

The synthesis: $$f = \sum_{l=1}^{n} \alpha_l \phi_l = \sum_{l=1}^{n} \langle f, \phi_l \rangle \phi_l \approx \sum_{l=1}^{k<n} \alpha_l \phi_l$$

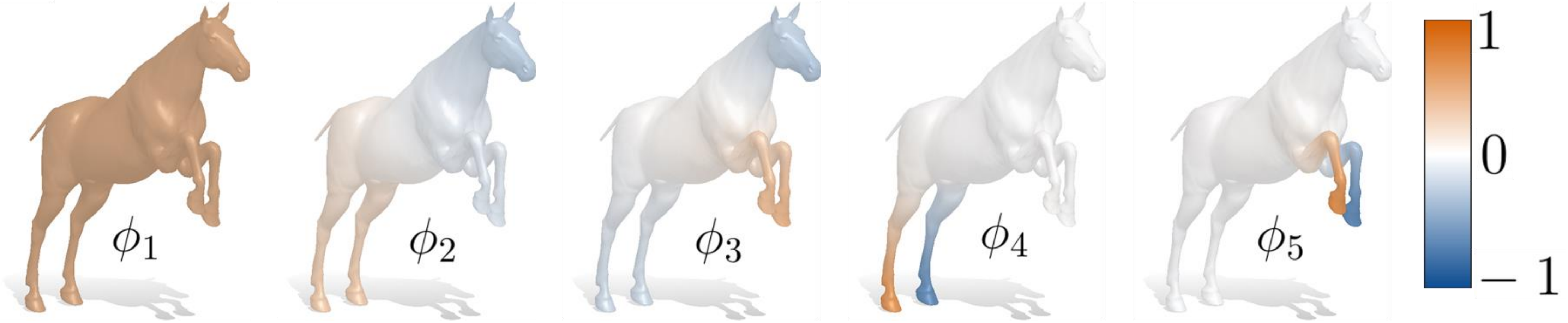


Fourier on surfaces

LBO eigenvectors ≈ Fourier basis for the functions on the mesh

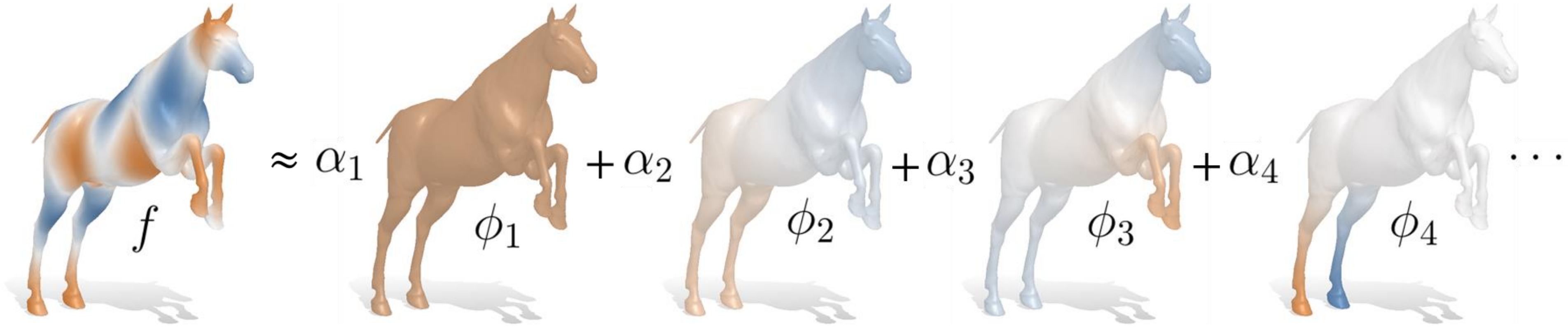
$$\mathbf{\Delta}_{\mathcal{M}}\phi_l = \lambda_l \phi_l \qquad \langle \phi_l, \phi_k \rangle_{\mathcal{M}} = \delta_l^k \qquad \lambda_l = \int_{\mathcal{M}} \|\nabla \phi_l\|^2 d\mu(x)$$

$\lambda_1 = 0$ $\quad \lambda_2 = 6.23$ $\quad \lambda_3 = 11.36$ $\quad \lambda_4 = 12.85$ $\quad \lambda_5 = 16.46$



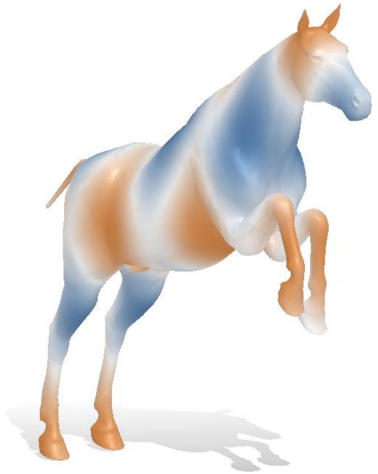
[Levy B., «Lapalce-Beltrami eigenfunction towards an algorithm that understands geometry», 2006.]

Fourier representation



Fourier operations on surfaces

Given a signal: f



The analysis: $\alpha_l = \langle f, \phi_l \rangle_{\mathcal{M}} = \int_{\mathcal{M}} f(x)\phi_l(x)d\mu(x)$

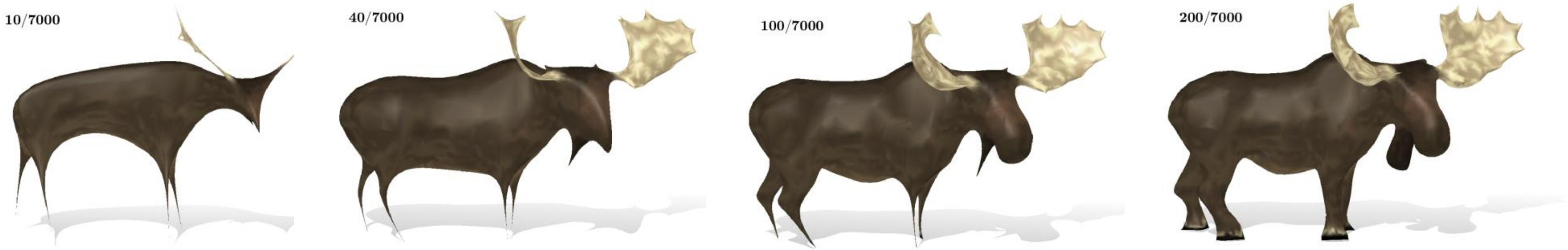
The synthesis: $f = \sum_{l=1}^{n} \alpha_l \phi_l = \sum_{l=1}^{n} \langle f, \phi_l \rangle_{\mathcal{M}} \phi_l \approx \sum_{l=1}^{k<n} \alpha_l \phi_l$

Coordinates approximation

the 3 coordinates X, Y and Z as functions
we reconstruct the geometry exploiting Fourier:

$$\tilde{X} = \sum_{i=1}^{k} \alpha_i \varphi_i \text{, where } \alpha_i = \langle \varphi_i, X \rangle_{\mathcal{X}} = {\varphi_i}^T \Omega_{\mathcal{X}} X = {\varphi_i}^{\dagger} X$$

The same for Y and Z and then plot $\tilde{X}, \tilde{Y}, \tilde{Z}$

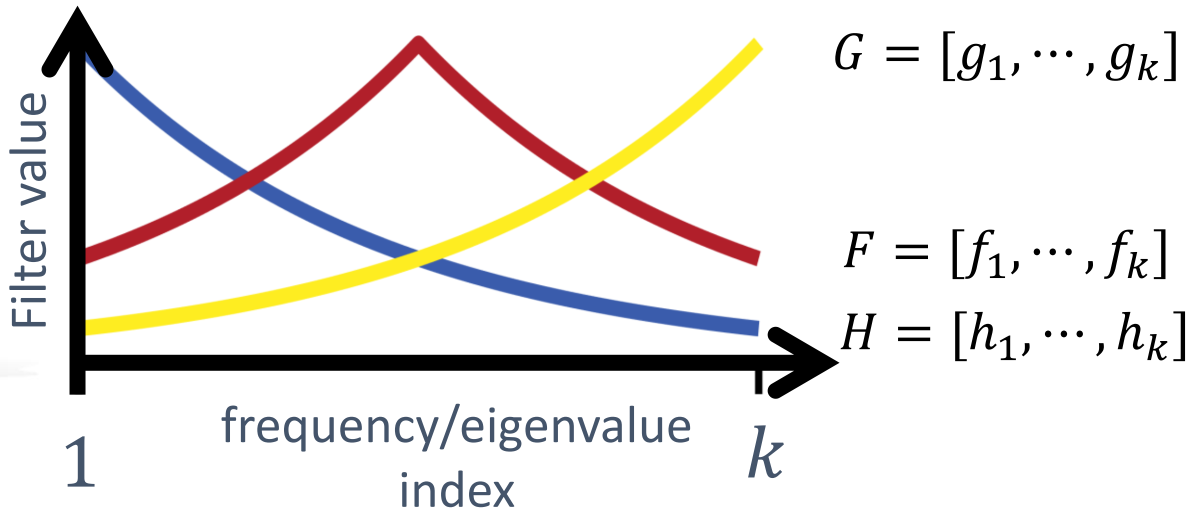


$k = 10$ $k = 40$ $k = 100$ $k = 200$

Slide credits to M. Ovsjanikov

Frequency filtering

$A = [\alpha_1, \cdots, \alpha_k]$

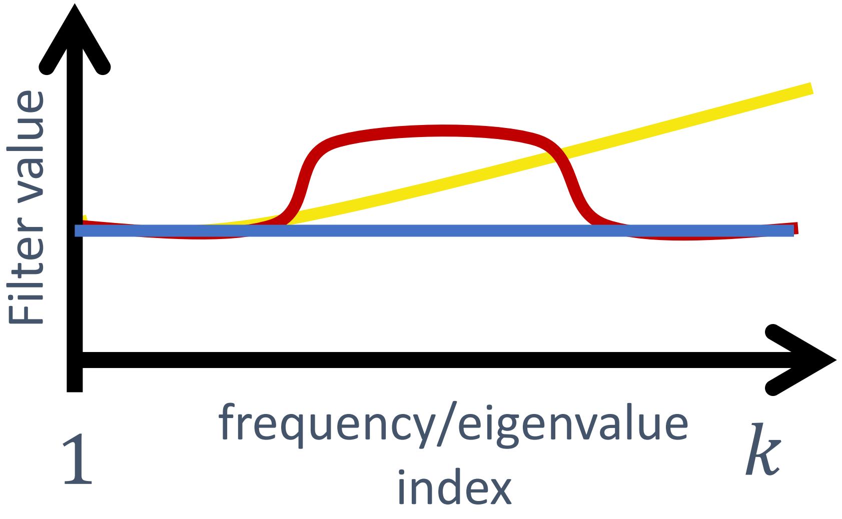


$A_H = [h_1 \cdot \alpha_1, \cdots, h_k \cdot \alpha_k]$

$A_F = [f_1 \cdot \alpha_1, \cdots, f_k \cdot \alpha_k]$

$A_G = [g_1 \cdot \alpha_1, \cdots, g_k \cdot \alpha_k]$

Geometry filtering

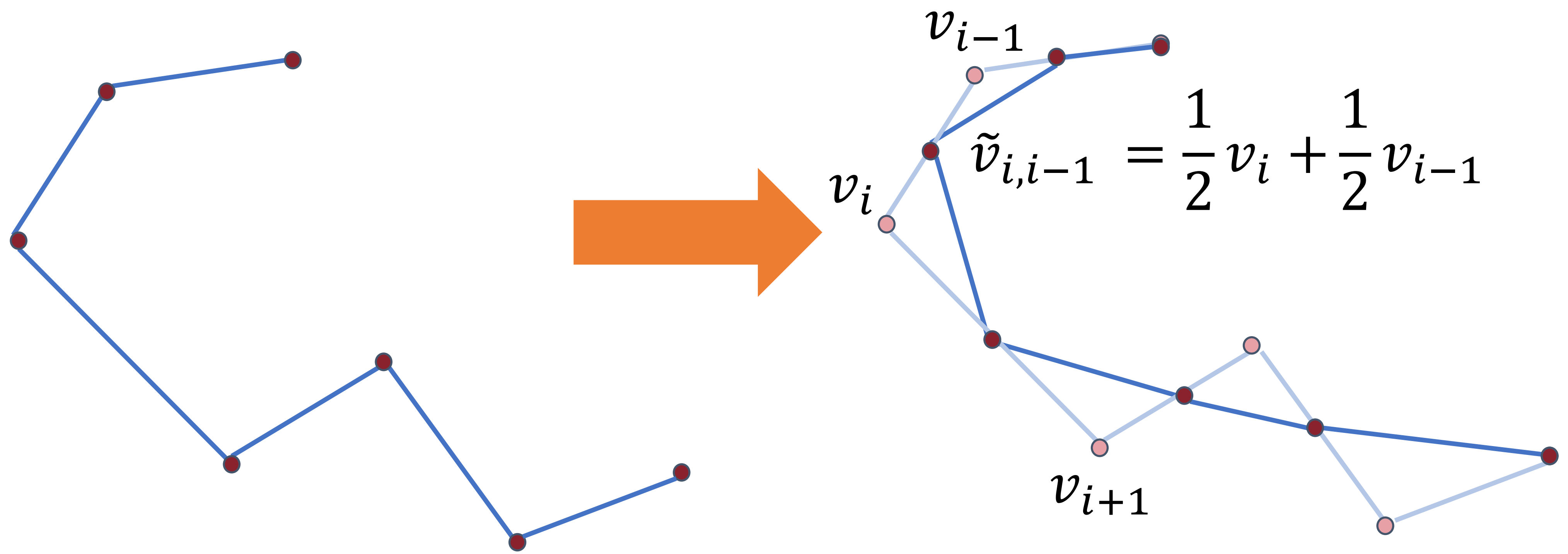


[Vallet and Levy: «Spectral Geometry Processing with Manifold Harmonics», 2008.](#)

Laplacian Smoothing

Smoothing

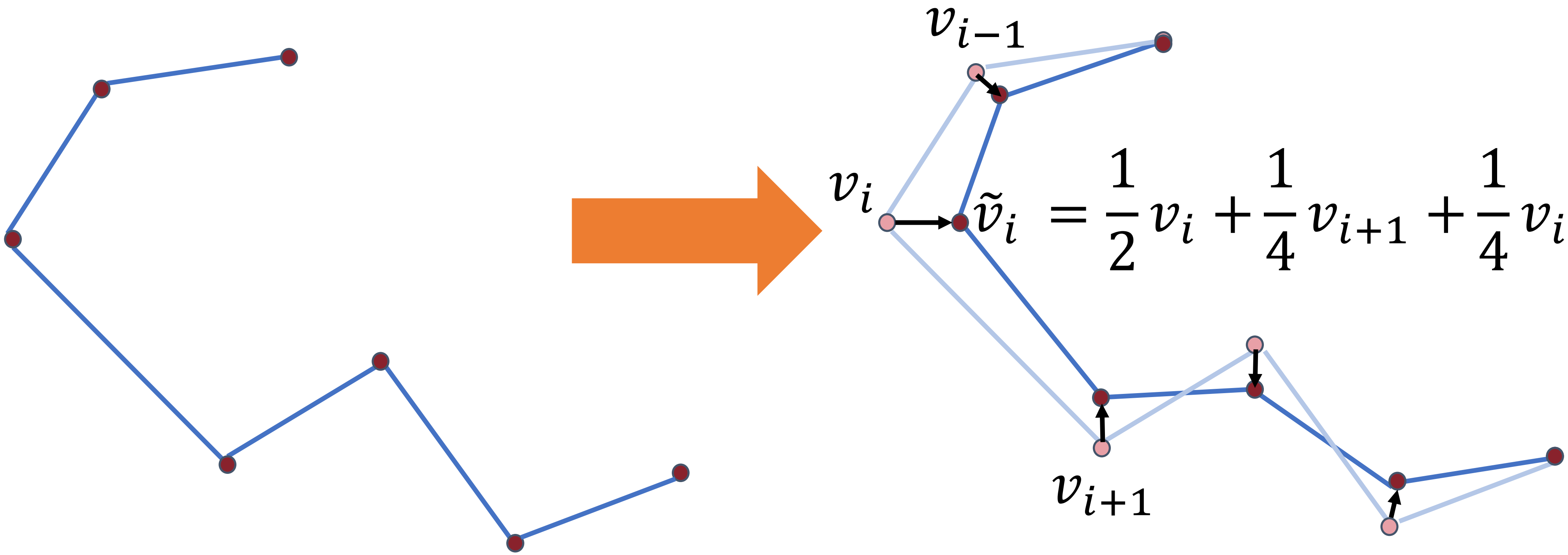
Given a discrete manifold (a 1D curve, or a surface) → Find a **smoother version** that approximate it



$$\tilde{v}_{i,i-1} = \frac{1}{2} v_i + \frac{1}{2} v_{i-1}$$

[Levy and Zhang: «Spectral Mesh processing», 2009.]

Laplacian smoothing



$$\tilde{v}_i = \frac{1}{2} v_i + \frac{1}{4} v_{i+1} + \frac{1}{4} v_{i-1}$$

Smoothing operator:

$$S = \begin{pmatrix} 1/2 & 1/4 \cdots 0 \cdots 0 & \cdots & 0 & 1/4 \\ \vdots & \ddots & & \vdots & \\ 1/4 & 0 \cdots 0 \cdots 0 & & \cdots 1/4 & 1/2 \end{pmatrix}$$

Levy and Zhang: «Spectral Mesh processing», 2009.

Laplacian?

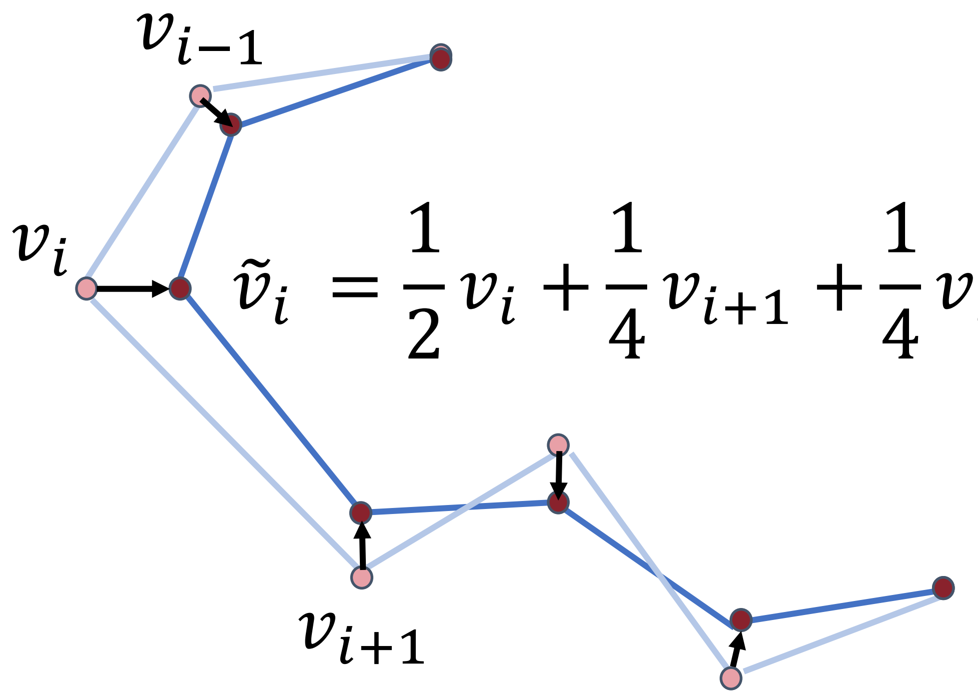
Graph Laplacian:

$$\Delta = \begin{pmatrix} 2 & -1 \cdots 0 \cdots 0 & \cdots\ 0 & -1 \\ \vdots & \ddots & \vdots & \\ 1 & 0 \cdots 0 \cdots 0 & \cdots -1 & 2 \end{pmatrix}$$

$$S = Id - 0.5(\alpha)\Delta \qquad for \quad \alpha = \frac{1}{2}$$

Smoothing operator:

$$S = \begin{pmatrix} 1/2 & 1/4 \cdots 0 \cdots 0 & \cdots & 0 & 1/4 \\ \vdots & \ddots & & \vdots & \\ 1/4 & 0 \cdots 0 \cdots 0 & \cdots & 1/4 & 1/2 \end{pmatrix}$$



$$\tilde{v}_i = \frac{1}{2} v_i + \frac{1}{4} v_{i+1} + \frac{1}{4} v_{i-1}$$

Levy and Zhang: «Spectral Mesh processing», 2009.

Laplacian smoothing and meshes

$$\tilde{v}_i = w_{ii}v_i + \sum_{j=1, j\neq i}^{n} w_{ij}v_j \; s.t. \sum_{j=1}^{n} w_{ij}v_j = 1$$

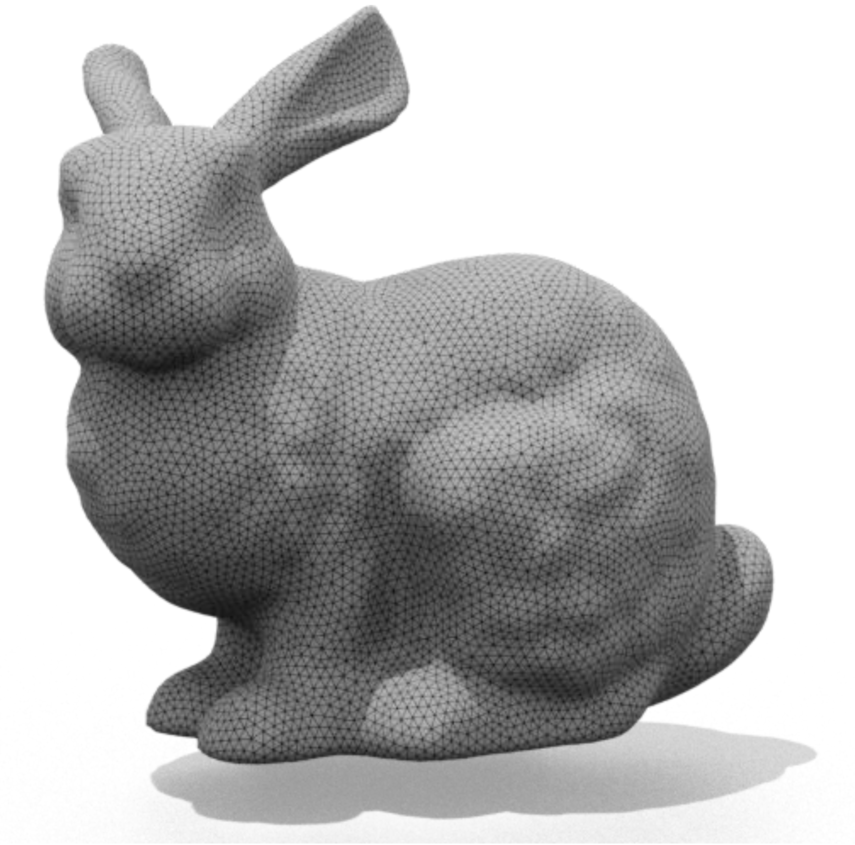
$$w_{ij} \neq 0 \Leftrightarrow e_{ij} \in E$$

Obtain a smoothing operator from Δ

Smoothing operator:

$$S = Id - 0.5(\alpha)\Delta$$

We need to set α



Levy and Zhang: «Spectral Mesh processing», 2009.

An example

$V_0 = V = [X, Y, Z] \in \mathbb{R}^{n\times 3}$ = the 3D coordinates
the LBO $\Delta \in \mathbb{R}^{n\times n}$

Compute iteratively $\forall t$:

$$V_t = V_{t-1} - LV_{t-1}$$

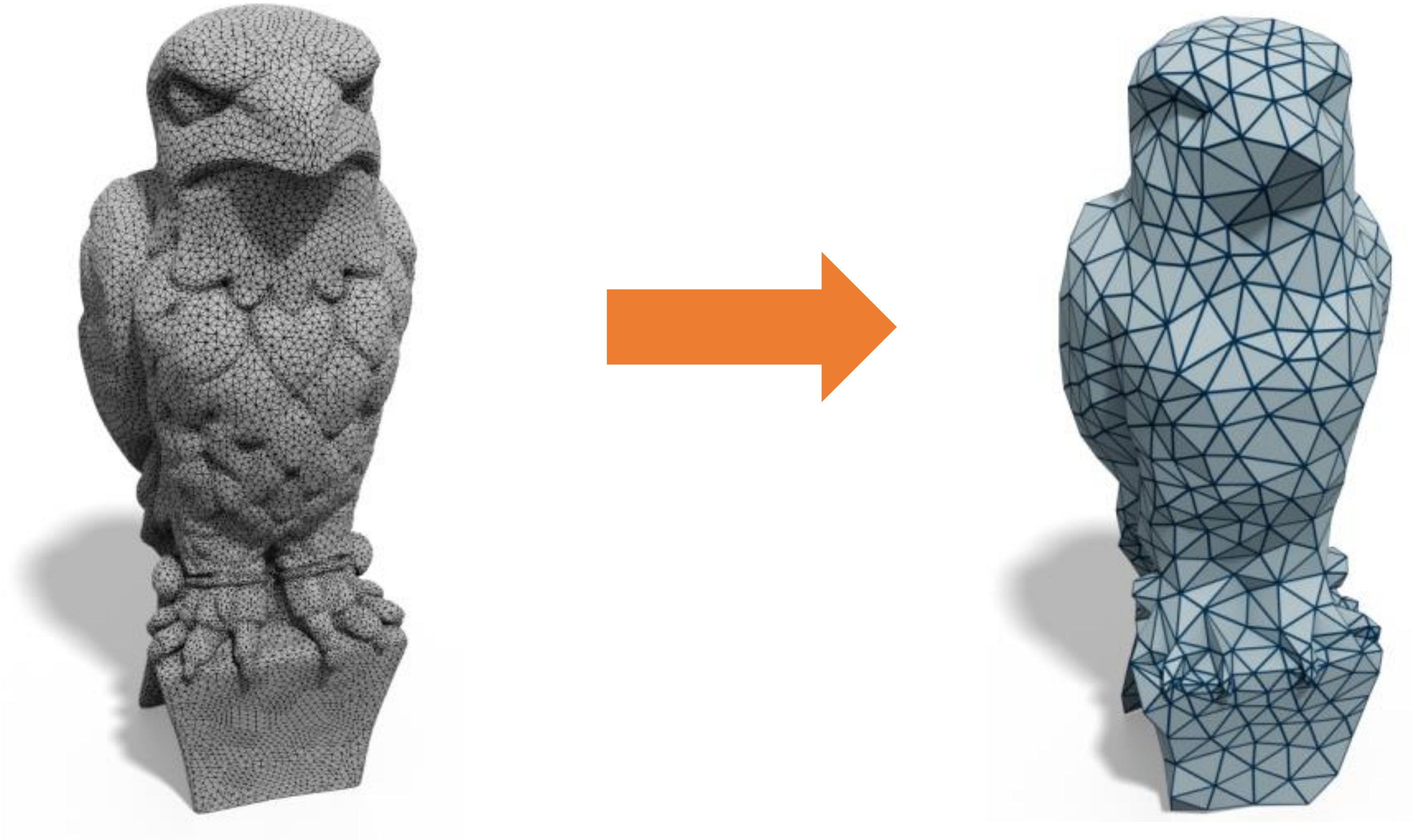
for $L = \alpha\, \Delta = diag(\Delta\,)^{-1}\, \Delta$

source   5 iterations   20 iterations

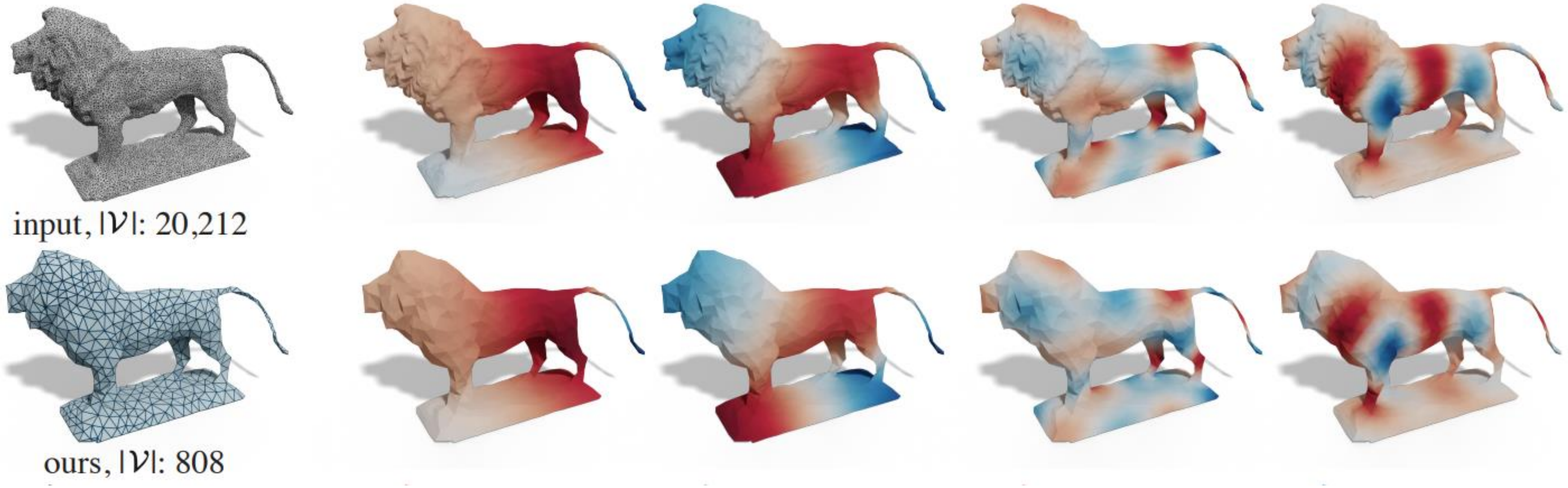
Mesh simplification

Mesh simplification

Given a discrete manifold → **Reduce** the **vertices** used to resprent it **preserving** its **geometry**



Spectral mesh simplification



The edges (and vertices) to remove are selected w.r.t. a spectral energy

The simplification process should preserve the LBO and its eigedecomposition

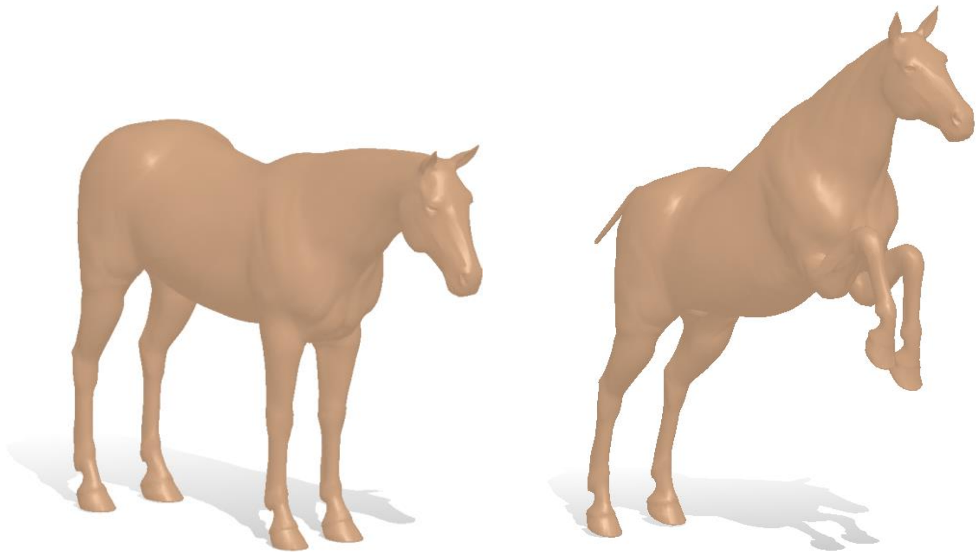
Slide credits to M. Ovsjanikov

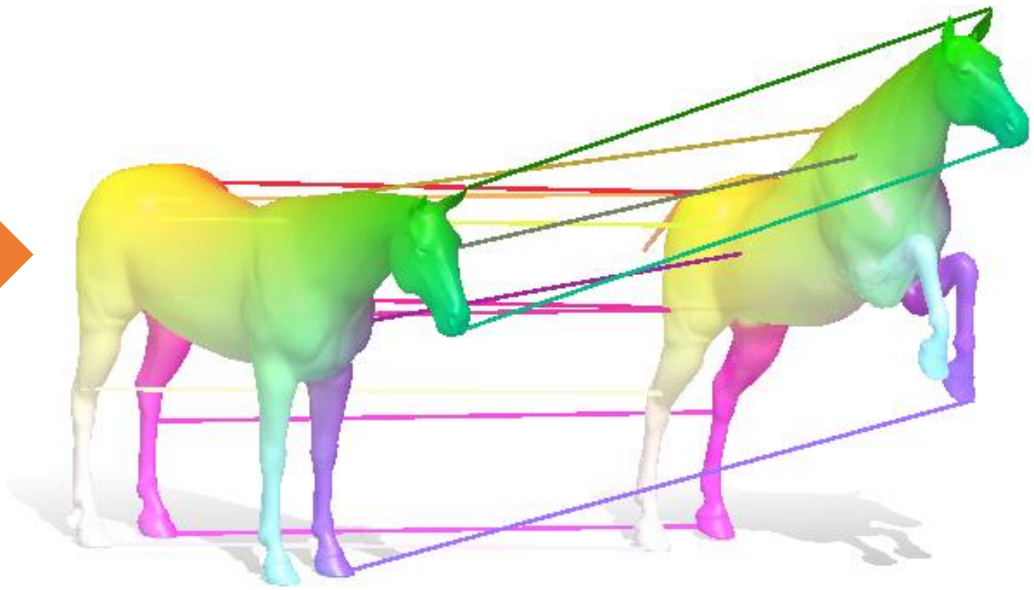
Lescoat et al.: «Spectral Mesh simplification», 2020.

Non Rigid Matching
Pointwise descriptors

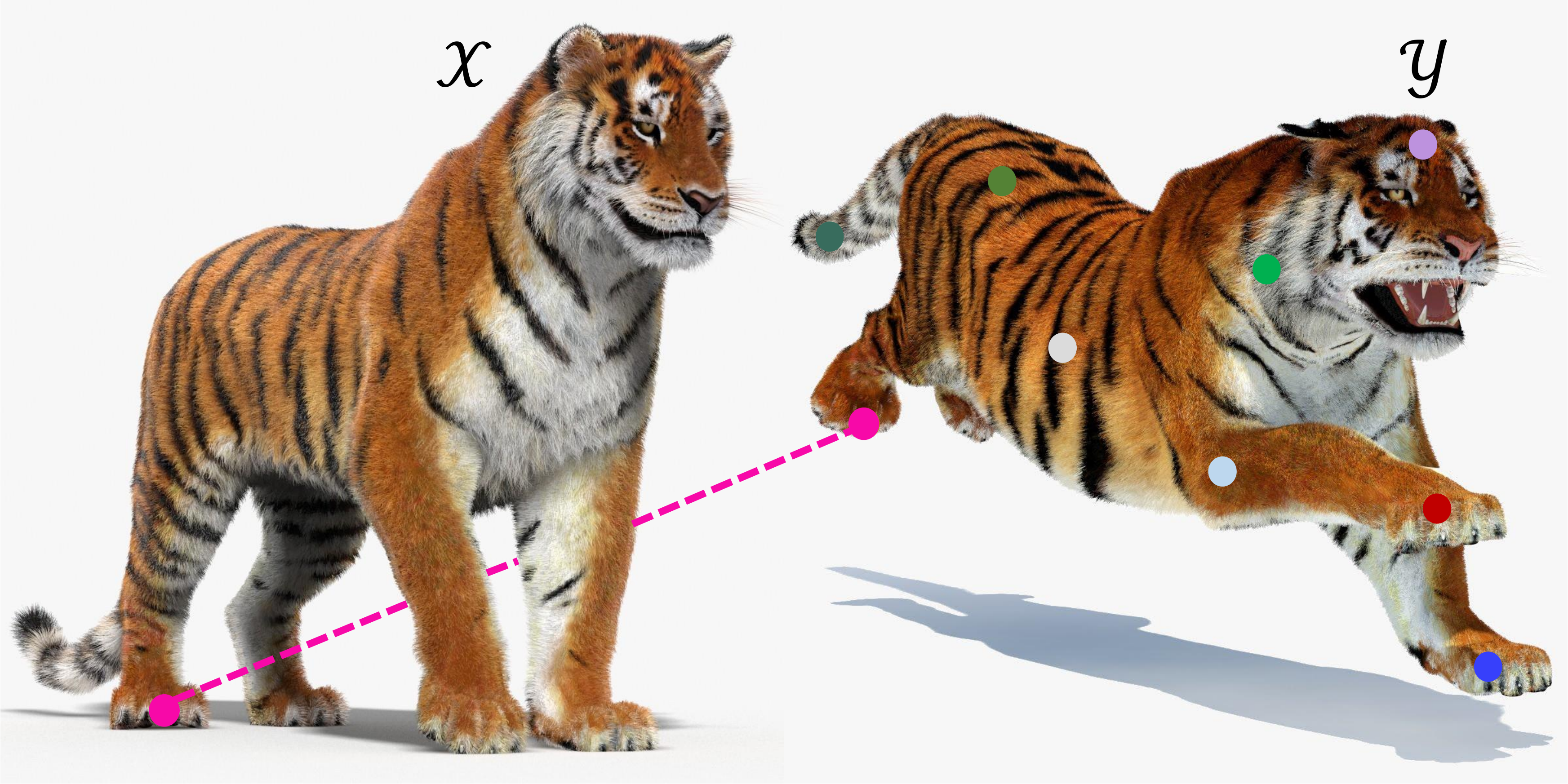
Non-rigid correspondence

Given a non-rigid deformation between 2 shapes → Find a **point-to-point** correspondence between the 2 shapes

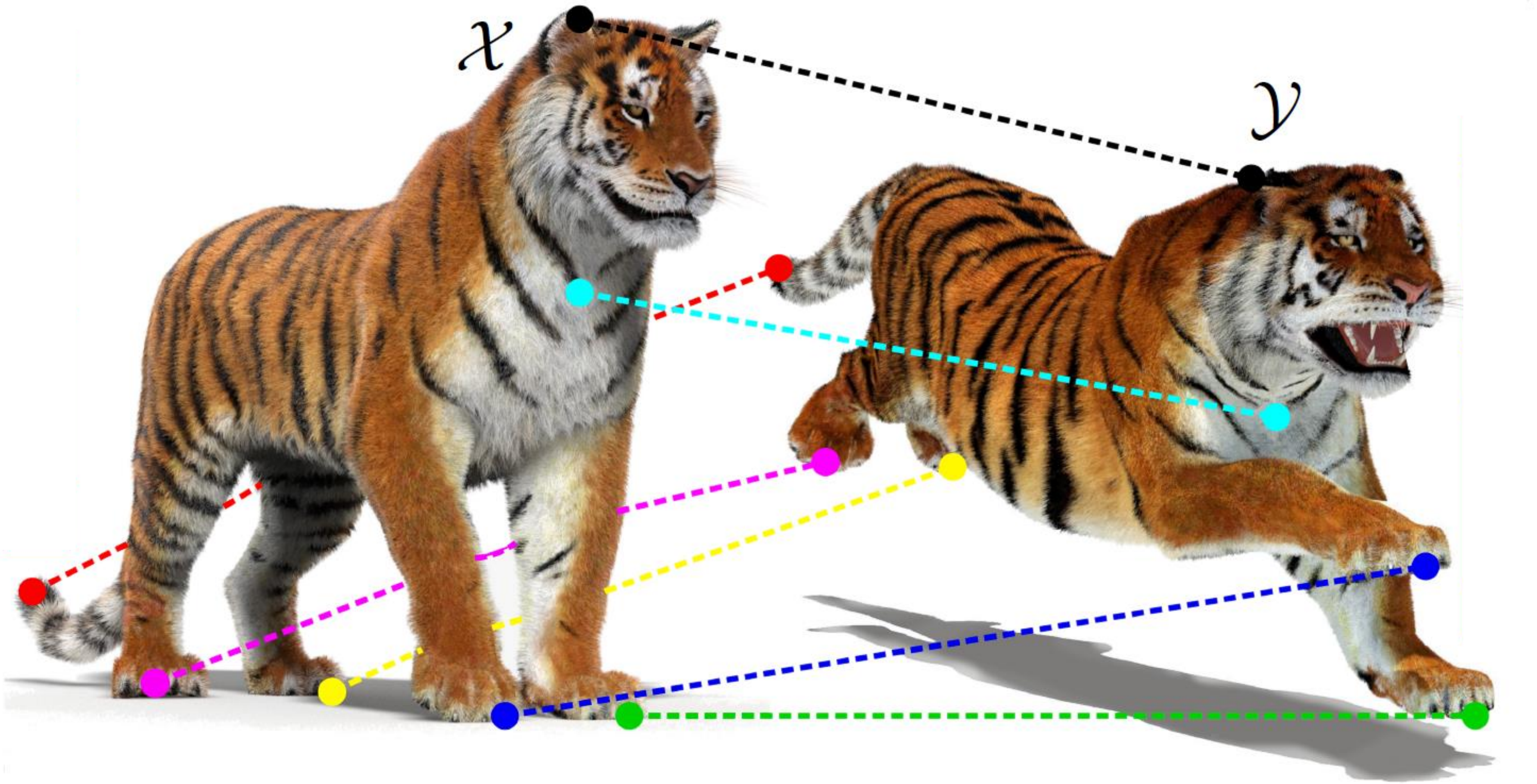




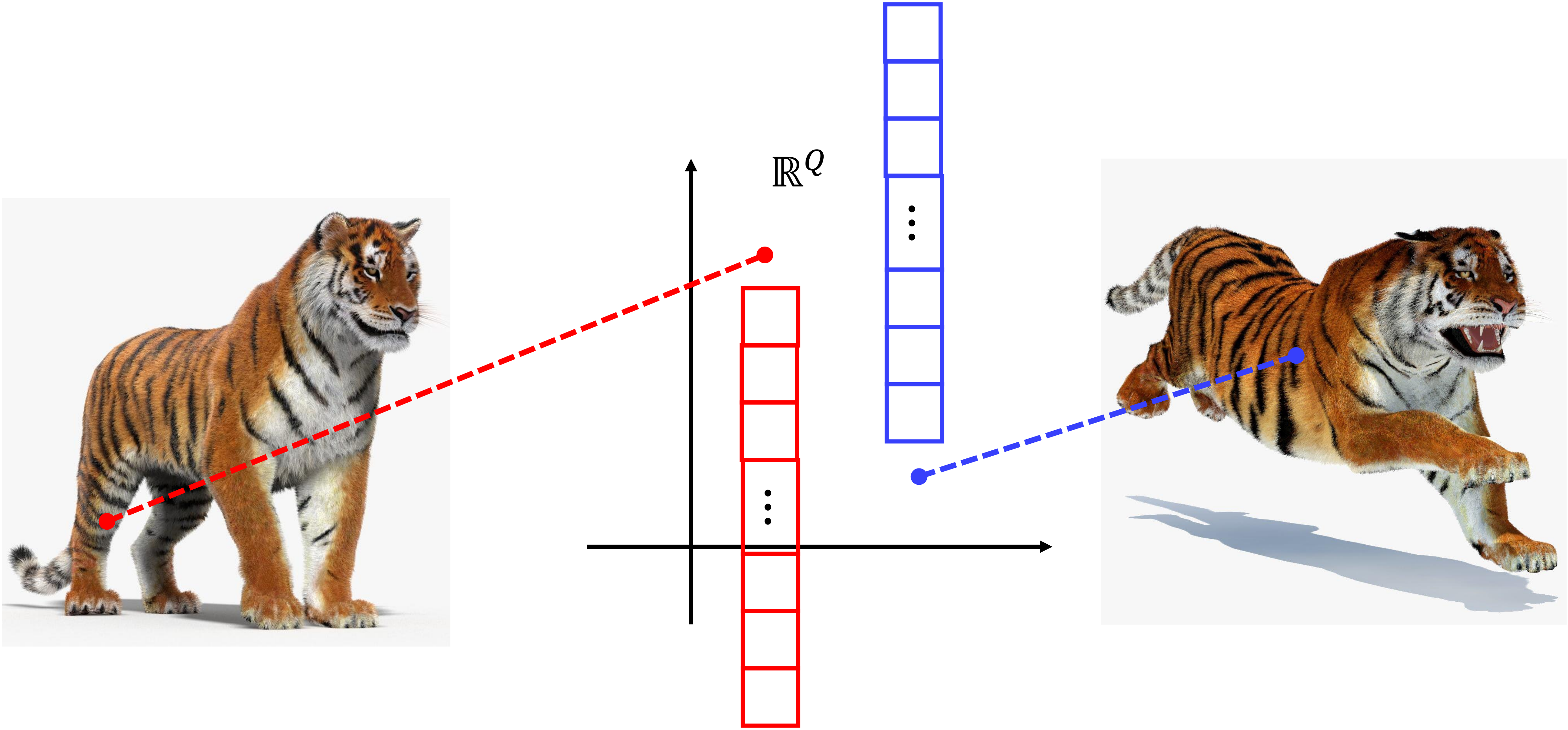
Non-rigid matching



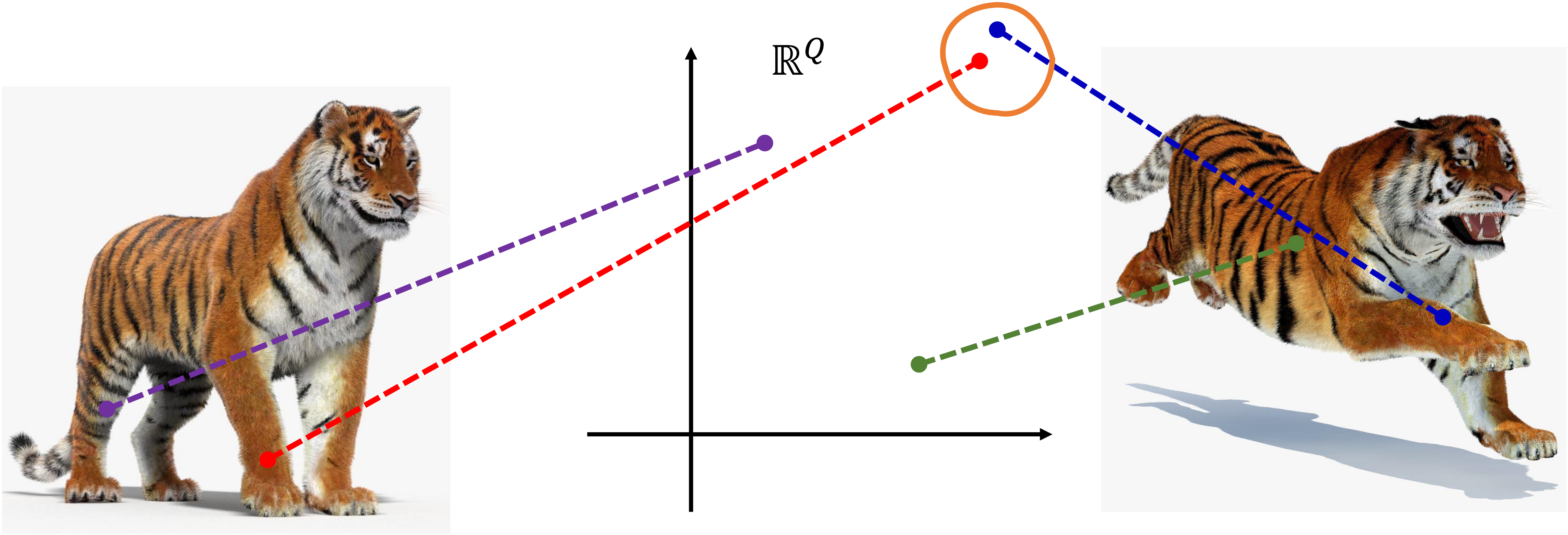
Non-rigid matching



Pointwise descriptor

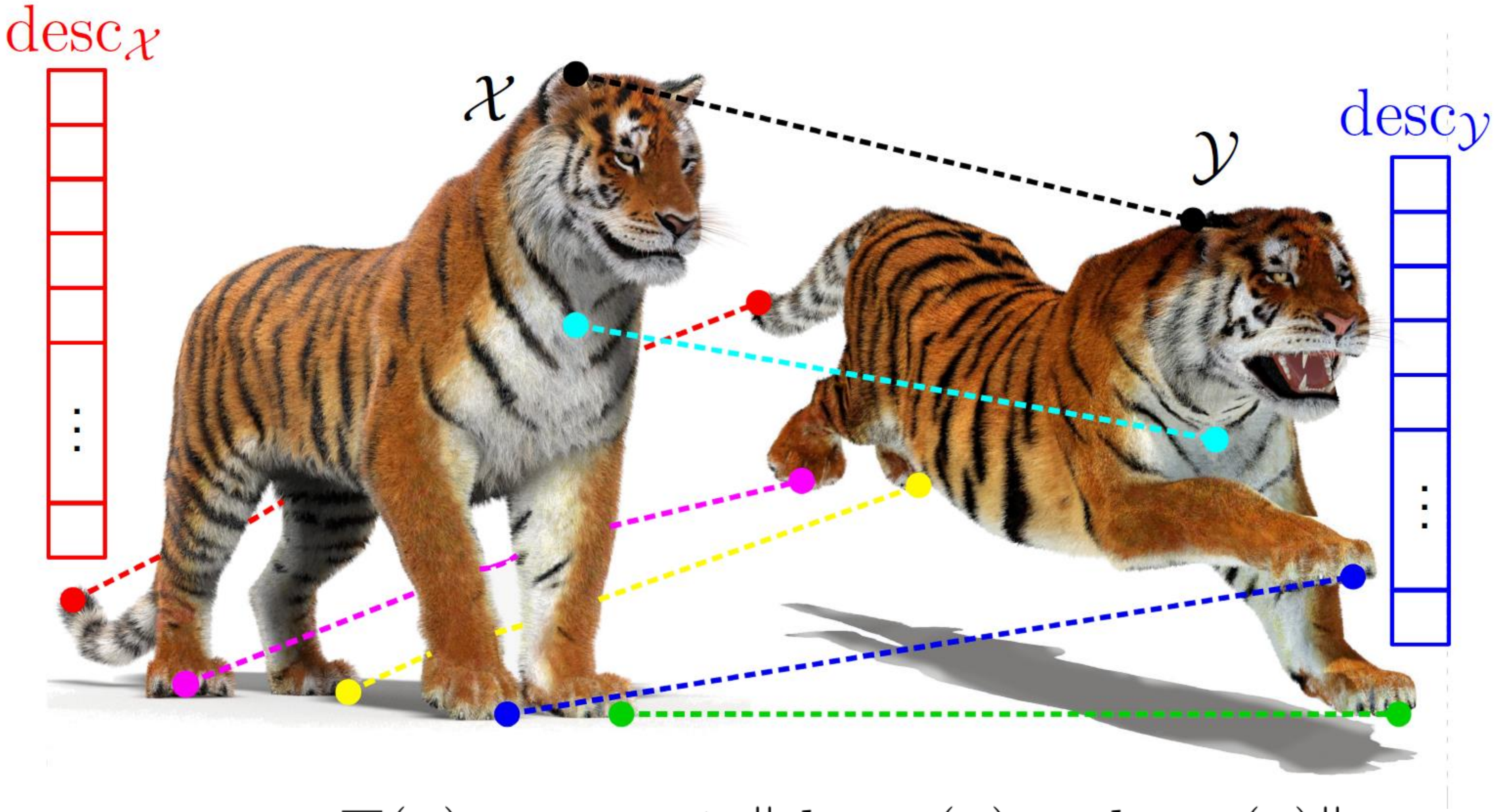


Pointwise descriptor



How can we find the most **similar** point?

Pointwise descriptor



$$y = \Pi(x) = \underset{y \in \mathcal{Y}}{argmin} \| desc_{\mathcal{X}}(x) - desc_{\mathcal{Y}}(y) \|$$

Heat diffusion

From physics the heat diffusion is governed by the

heat equation:

$$\Delta_{\mathcal{X}} u(x,t) = -\frac{\partial u(x,t)}{\partial t}$$

The LBO

=

derivatives in space

derivative in time

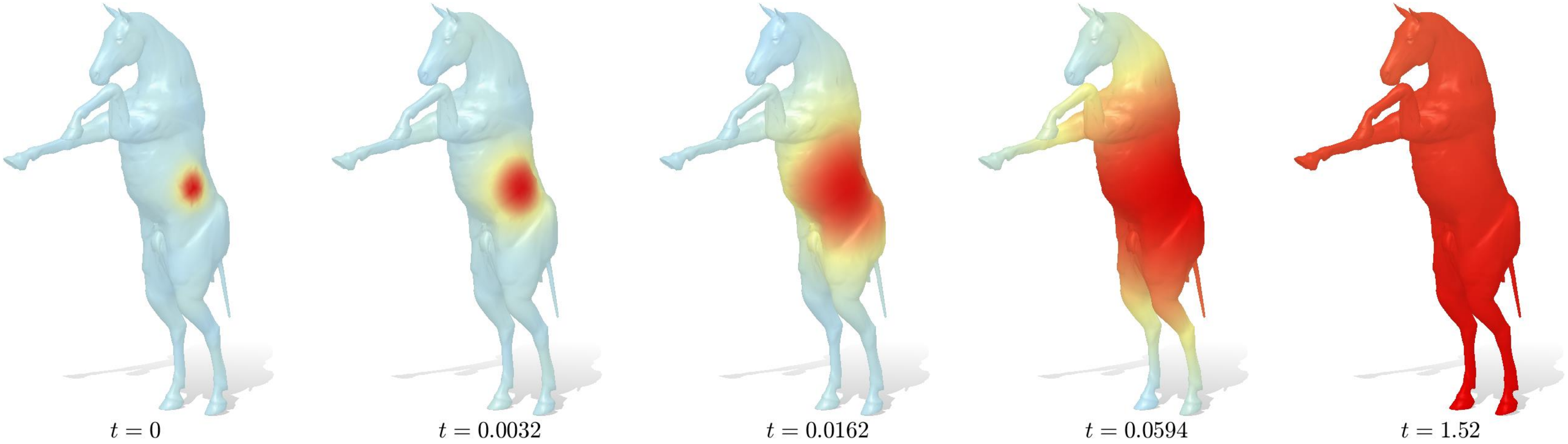
$u(x,t)$ solution of the heat equation is a function of $x \in \mathcal{X}$ and time $t \in \mathbb{R}$

which satisfies the **heat equation** for a given initial condition $u_0(x) = u(x,0)$.

Heat kernel

For a delta heat source δ_x in the point $x \in \mathcal{X}$, the **heat kernel** $h_t(x, y)$ measures how much heat passes from x to y in a time interval t

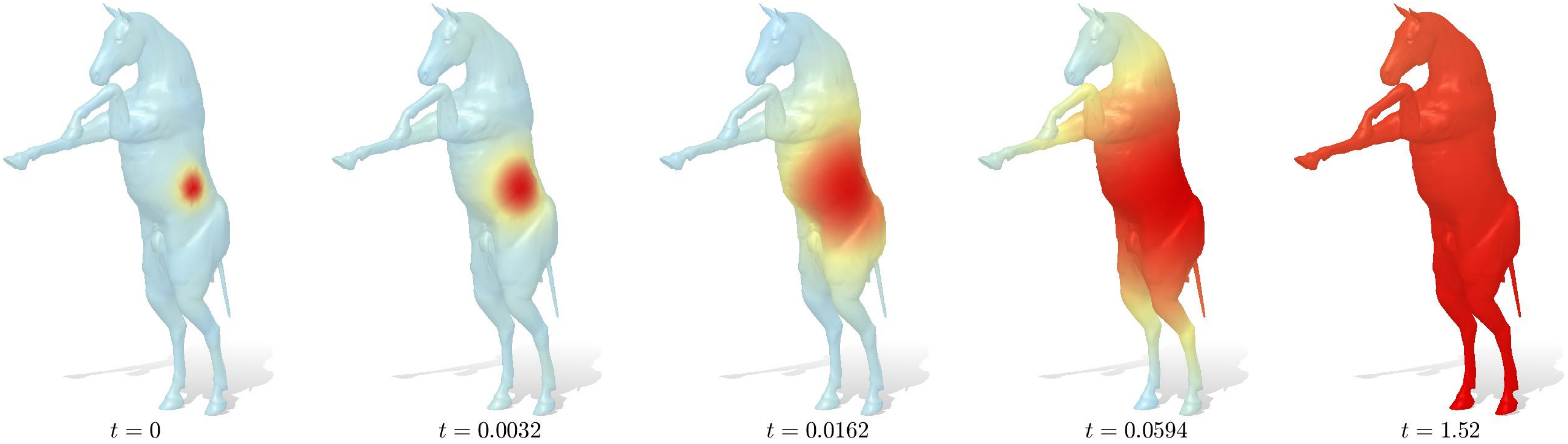
the heat kernel $$h_t(x,y) = \sum_{l=1}^{+\infty} e^{-t\lambda_l} \phi_l(x)\phi_l(y)$$



Heat kernel

For a delta heat source δ_x in the point $x \in \mathcal{X}$, the **heat kernel** $h_t(x, y)$ measures how much heat passes from x to y in a time interval t

the heat kernel $h_t(x, y) = \sum_{l=1}^{k} e^{-t\lambda_l} \phi_l(x)\phi_l(y)$



HKS: Heat kernel signature

For an initial delta distribution of heat $\delta_x, x \in \mathcal{X}$

For any $t \in \mathbb{R}^+$ in the set of time scales $\{t_1, t_2, \dots, t_Q\}$

$$h_t(x,x) = \sum_{l=1}^{k} e^{-t\lambda_l} \phi_l(x)\phi_l(x)$$

$h_t(x,x)$ is the amount of heat remaining at x after time t

$$\mathbf{HKS}(x) = \left[\, h_{t_1}(x,x) \; \vdots \; h_{t_2}(x,x) \; \vdots \; \cdots \qquad \cdots \; \vdots \; h_{t_Q}(x,x) \, \right]$$

$\mathbf{HKS}(x)$ is the heat kernel signature at the point $x \in \mathcal{X}$

Sun et al., «A Concise and Provably Informative Multi-scale Signature Based on Heat Diffusion», 2009.

The wave equation (Schrodinger)

Heat equation: $$\Delta_{\chi} u(x,t) = -\frac{\partial u(x,t)}{\partial t}$$

Wave equation: $$i\Delta_{\chi} u(x,t) = \frac{\partial u(x,t)}{\partial t}$$

presence of the *i*

missing a minus

It governs the temporal evolution of a quantum particle

It encodes oscillation rather than dissipation as done by the heat equation

[The wave kernel signature: A quantum mechanical approach to shape analysis, *Aubry et al.*, 2011.]

WKS: Wave kernel signature

For an initial quantum particles probability distribution over $\mathcal{X}$ depending on the the energy $E \in \mathbb{R}$ in the set of energy scales $\{E_1, E_2, \ldots, E_Q\}$

$$k_E(x,x) = \sum_{l=1}^{k} e^{-\frac{(\log(E)-\log(\lambda_l))^2}{2\sigma^2}} \phi_l(x)\phi_l(x)$$

$k_E(x,x)$ is the average probability over the time to find a particle in x given the initial energy E.

$$\mathbf{WKS}(x) = \left[k_{E_1}(x,x) \;\vdots\; k_{E_2}(x,x) \;\vdots\; \cdots \quad\quad \cdots \;\vdots\; k_{E_Q}(x,x) \right]$$

$\mathbf{WKS}(x)$ is the wave kernel signature at the point $x \in \mathcal{X}$

The wave kernel signature: A quantum mechanical approach to shape analysis, *Aubry et al.*, 2011.

Spectral descriptors

the spectral descriptors **HKS** and **WKS** share a common structure

$$desc_q(x) = \sum_{l=1}^{k} g_{t_q}(\lambda_l)\,\phi_l(x)\phi_l(x), \forall q \in 1, \dots, Q$$

A set of filters on the frequencies

=

functions of the eigenvalues

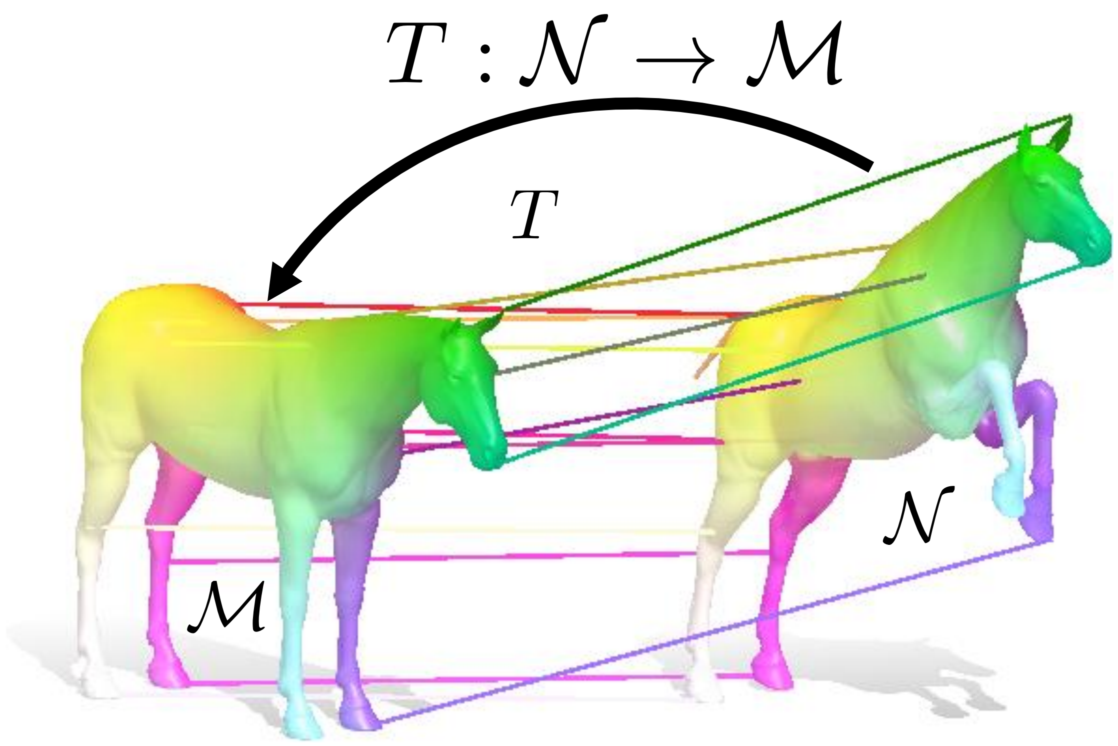
The square of each dimension of the spectral embedding

We can learn the filters as functions o the eigenvalues to obtain better descriptors!

"Learning spectral descriptors for deformable shape correspondence", *Litman et al.*, 2014.

Non Rigid Matching
Functional Maps

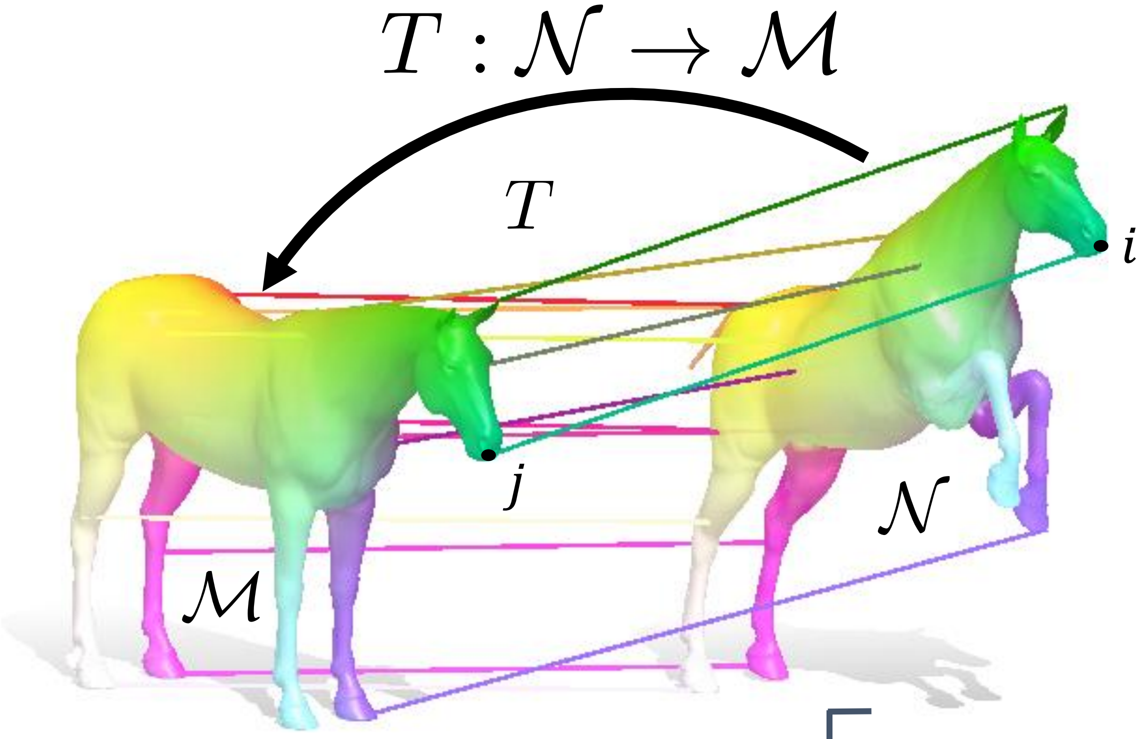
Functional maps



T is a point-to-point map

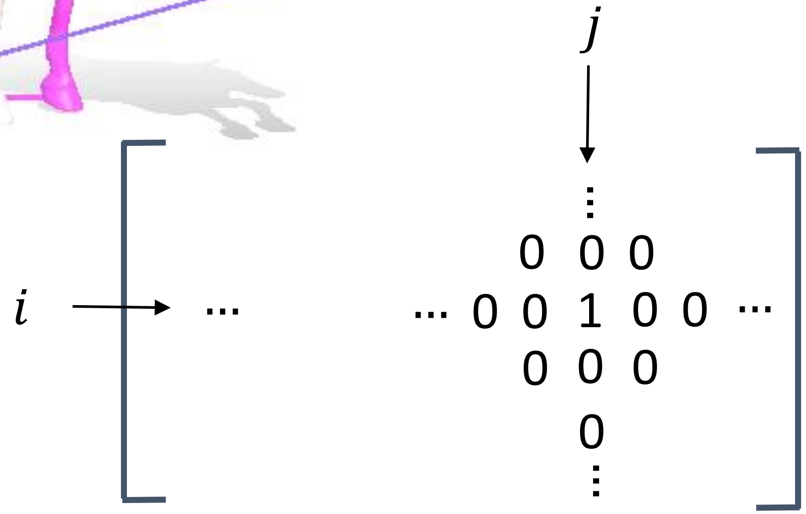
Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Functional maps



We write T as a binary matrix

$$\Pi_{\mathcal{N}\mathcal{M}}(i, j) = 1 \Longleftrightarrow T(i) = j$$

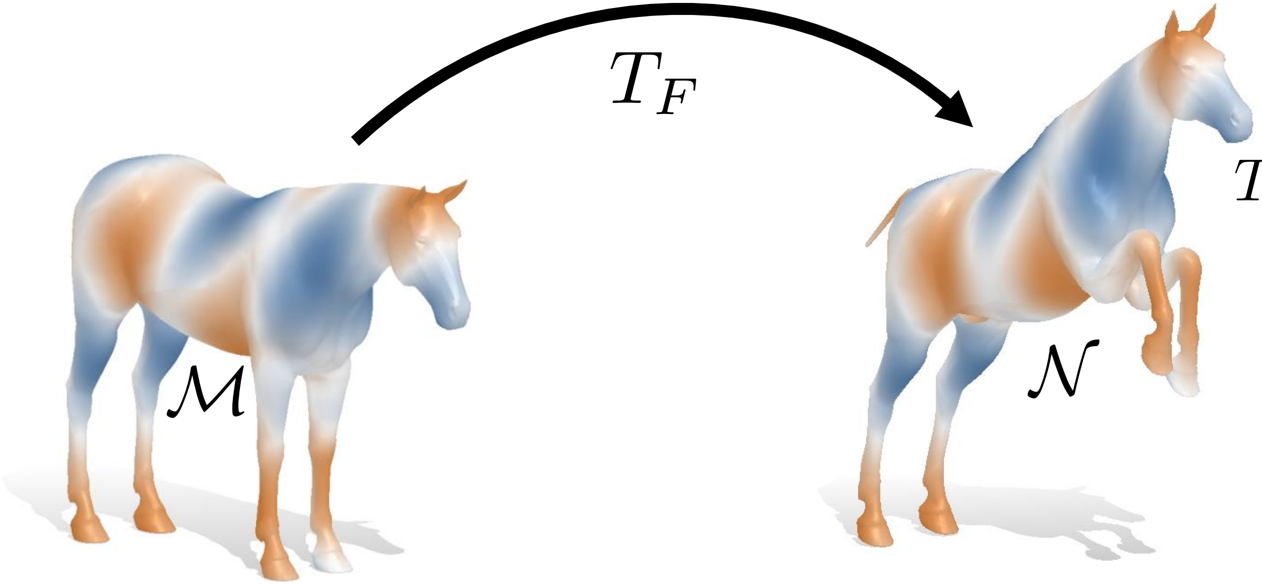


[*Ovsjanikov at al.*, "Functional maps: a flexible representation of maps between shapes", 2012]

Functional maps

$$T_F : F(\mathcal{M}) \longrightarrow F(\mathcal{N})$$

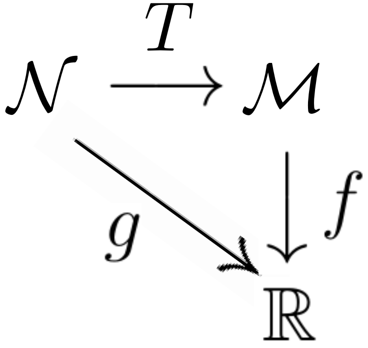
$f : \mathcal{M} \to \mathbb{R}$



$T_F(f) = g : \mathcal{N} \to \mathbb{R}$

The transfer is defined as:

$$g(y) = f(T(y)) \quad \text{or} \quad g = \Pi_{\mathcal{N}\mathcal{M}} f$$



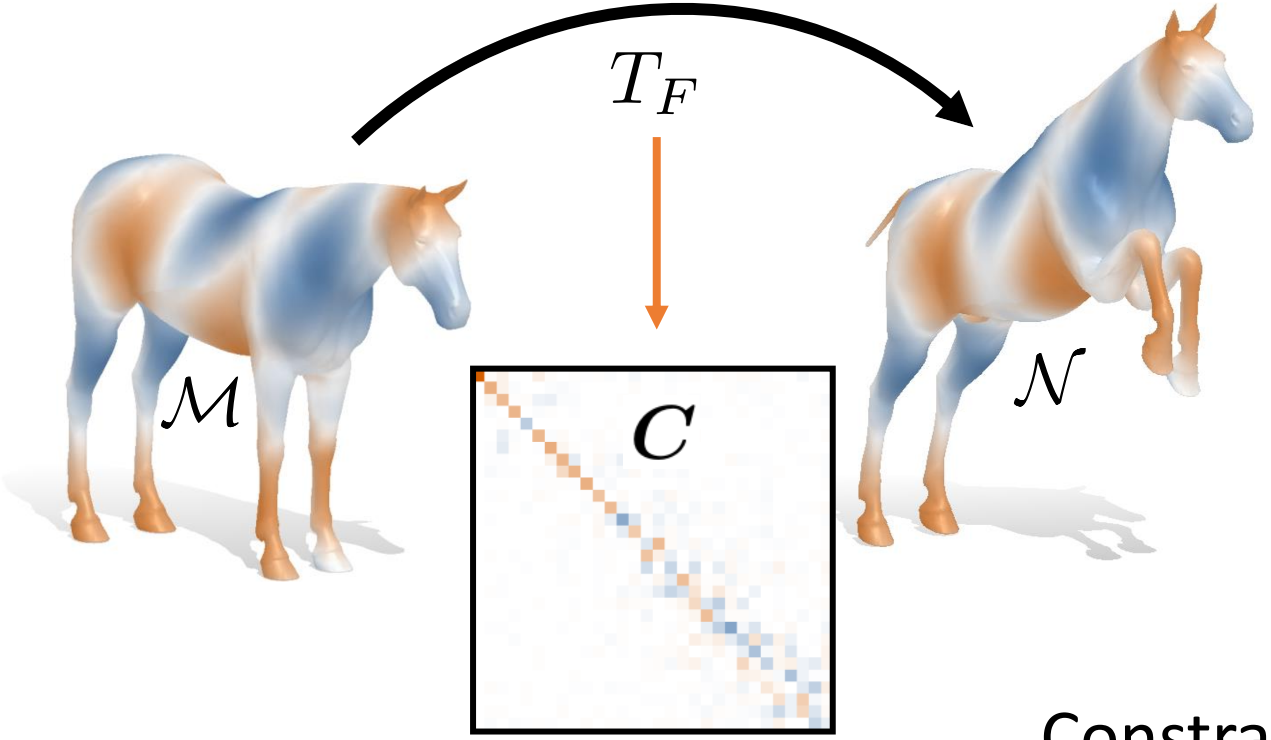
Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Functional maps

$$T_F : F(\mathcal{M}) \longrightarrow F(\mathcal{N})$$

$f : \mathcal{M} \to \mathbb{R}$

$\boldsymbol{a}$



$T_F(f) = g : \mathcal{N} \to \mathbb{R}$

$g = \Pi_{\mathcal{N}\mathcal{M}} f$

$\boldsymbol{b}$

Explicit formula:

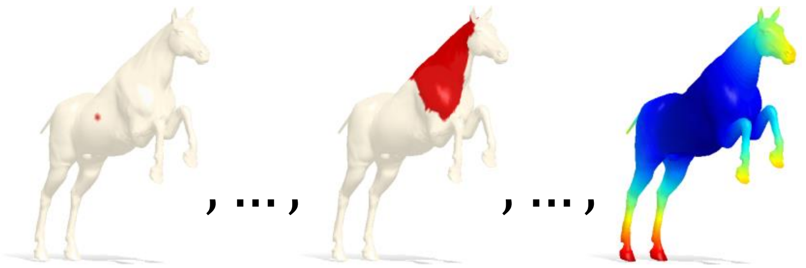
$$\boldsymbol{C} = \boldsymbol{\Phi}_{\mathcal{N}}^{\dagger} \boldsymbol{\Pi}_{\mathcal{N}\mathcal{M}} \boldsymbol{\Phi}_{\mathcal{M}}$$

Constraint:

$$\boldsymbol{b} = \boldsymbol{C}\boldsymbol{a}$$

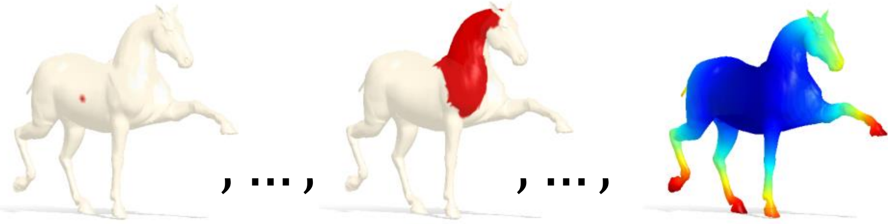
Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Functional maps optimization

$F =$ []

f_1 (delta) f_l (region) f_q (descriptor)

$\hat{F} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_l, \ldots, \boldsymbol{\alpha}_q]$

$G =$ []

g_1 (delta) g_l (region) g_q (descriptor)

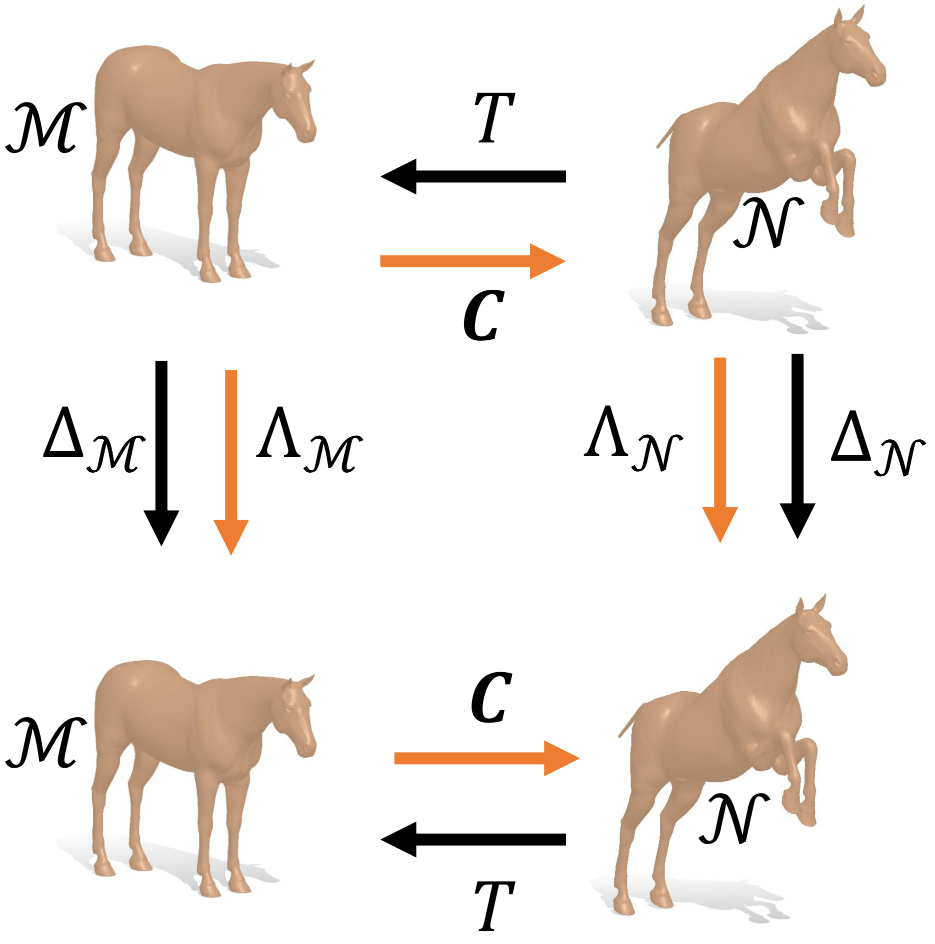
$\hat{G} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_l, \ldots, \boldsymbol{\beta}_q]$

The matrix ***C*** should align the coefficients of all the given **probe functions**

$$\boldsymbol{C} = \underset{\boldsymbol{C} \in \mathbb{R}^{h \times k}}{\operatorname{argmax}} \left\| \boldsymbol{C}\hat{F} - \hat{G} \right\|_2 + \mathcal{R}(\boldsymbol{C})$$

Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Functional maps regularization



The map T is an isometry $\Leftrightarrow$ it commutes with the LBOs, that is: $\Delta_{\mathcal{M}} T = T \Delta_{\mathcal{N}}$

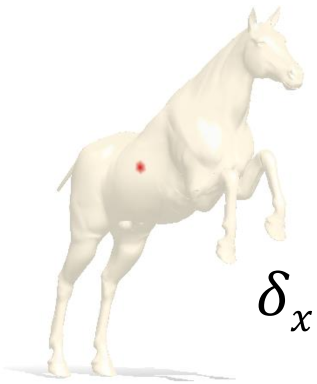
$$\Delta_{\mathcal{M}} T = T \Delta_{\mathcal{N}} \Leftrightarrow \boldsymbol{C}\Lambda_{\mathcal{M}} = \Lambda_{\mathcal{N}} \boldsymbol{C}$$

$$\boldsymbol{C} = \underset{\boldsymbol{C} \in \mathbb{R}^{h \times k}}{\operatorname{argmax}} \left\| \boldsymbol{C}\hat{F} - \hat{G} \right\|_2 + \mathcal{R}(\boldsymbol{C})$$

$$\mathcal{R}(\boldsymbol{C}) = \| \boldsymbol{C}\Lambda_{\mathcal{M}} - \Lambda_{\mathcal{N}} \boldsymbol{C} \|_2$$

Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Matching from Functional maps

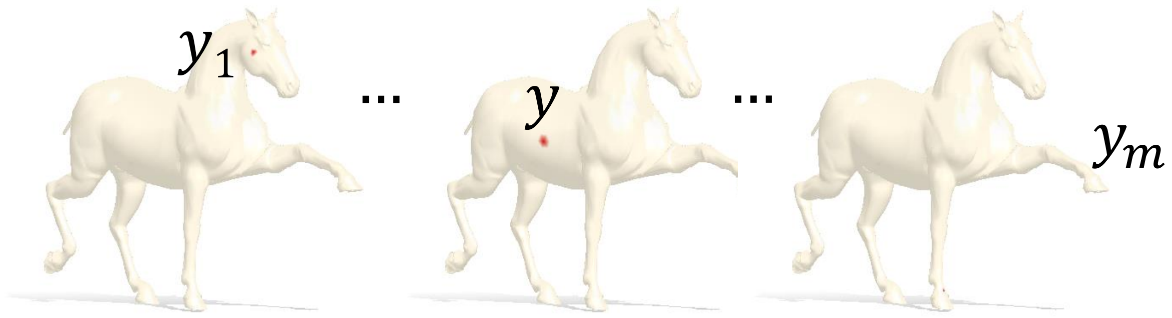


$\forall x \in \mathcal{M}$

$$\delta_x(p) = \begin{cases} 1 \; if \; x = p \\ 0 \; if \; x \neq p \end{cases}$$

$\delta_x \in F(\mathcal{M}, \mathbb{R})$

$y_1, \ldots, y, \ldots, y_m \in \mathcal{N}$



$\delta_{y_1}, \ldots, \qquad \delta_y, \ldots, \qquad \delta_{y_m} \in F(\mathcal{N}, \mathbb{R})$

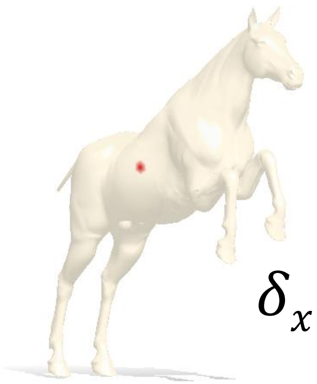
$\boldsymbol{C}$ maps $\boldsymbol{\alpha}_x$ in the space of coefficients of B

$\boldsymbol{\alpha}_x \longrightarrow \mathrm{B} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}, \ldots, \boldsymbol{\beta}_m]$

$$y = \underset{y \in \mathcal{N}}{\operatorname{argmin}}(dist_{\mathbb{R}^h}(\boldsymbol{C}\boldsymbol{\alpha}_x - \boldsymbol{\beta}_y))$$

Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Matching from Functional maps

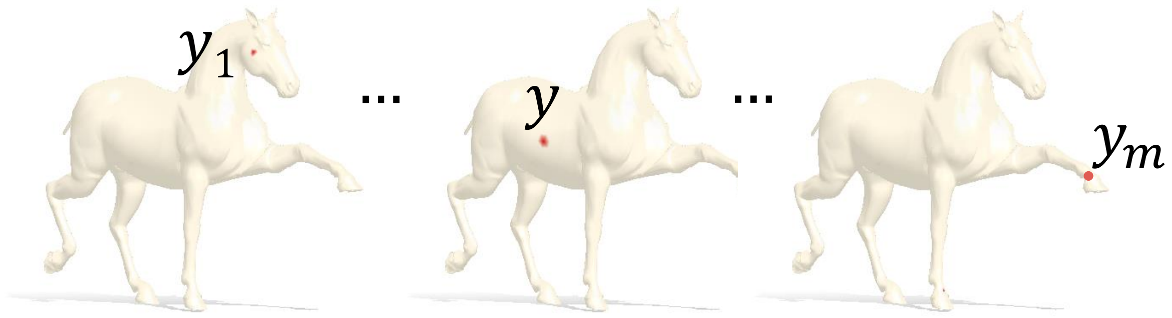


$\forall x \in \mathcal{M}$

$$\delta_x(p) = \begin{cases} 1 \; if \; x = p \\ 0 \; if \; x \neq p \end{cases}$$

$\delta_x \in F(\mathcal{M}, \mathbb{R})$

$y_1, \dots, y, \dots, y_m \in \mathcal{N}$



$\delta_{y_1}, \dots, \quad \delta_y, \dots, \quad \delta_{y_m} \in F(\mathcal{N}, \mathbb{R})$

$\boldsymbol{C}$ maps $\boldsymbol{\alpha}_x$ in the space of coefficients of B

$\boldsymbol{\alpha}_x \longrightarrow$ B $= [\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}, \dots, \boldsymbol{\beta}_m]$

$$y = \underset{y \in \mathcal{N}}{\operatorname{argmin}}(dist_{\mathbb{R}^h}(\boldsymbol{C}\Phi_{\mathcal{M}}(x) - \Phi_{\mathcal{N}}(y)));$$

$$\Phi_{\mathcal{M}}(x) = [\phi_1(x), \dots, \phi_k(x)]$$
$$\Phi_{\mathcal{N}}(y) = [\psi_1(y), \dots, \psi_h(y)]$$

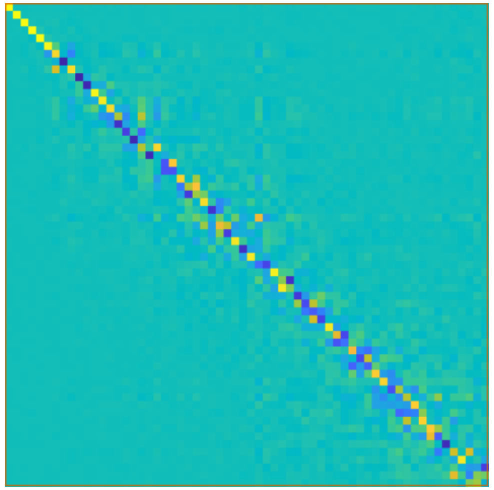
Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Spectral ICP refinement

$$y = T(x) = \underset{y \in \mathcal{N}}{\operatorname{argmin}} \left(dist_{\mathbb{R}^h}\left(\boldsymbol{C}\Phi_{\mathcal{M}}(x) - \Phi_{\mathcal{N}}(y)\right)\right), \forall x \in \mathcal{M}$$

Optimize for $\boldsymbol{C}$ as a rotation in the spectral domain as the best rotation to align $\Phi_{\mathcal{M}}$ and $\Phi_{\mathcal{N}}$:

$$\min_{\substack{C,T \\ C^T C = I}} \|\Phi_{\mathcal{N}}\boldsymbol{C} - \Phi_{\mathcal{M}}(T,:)\|_F^2$$

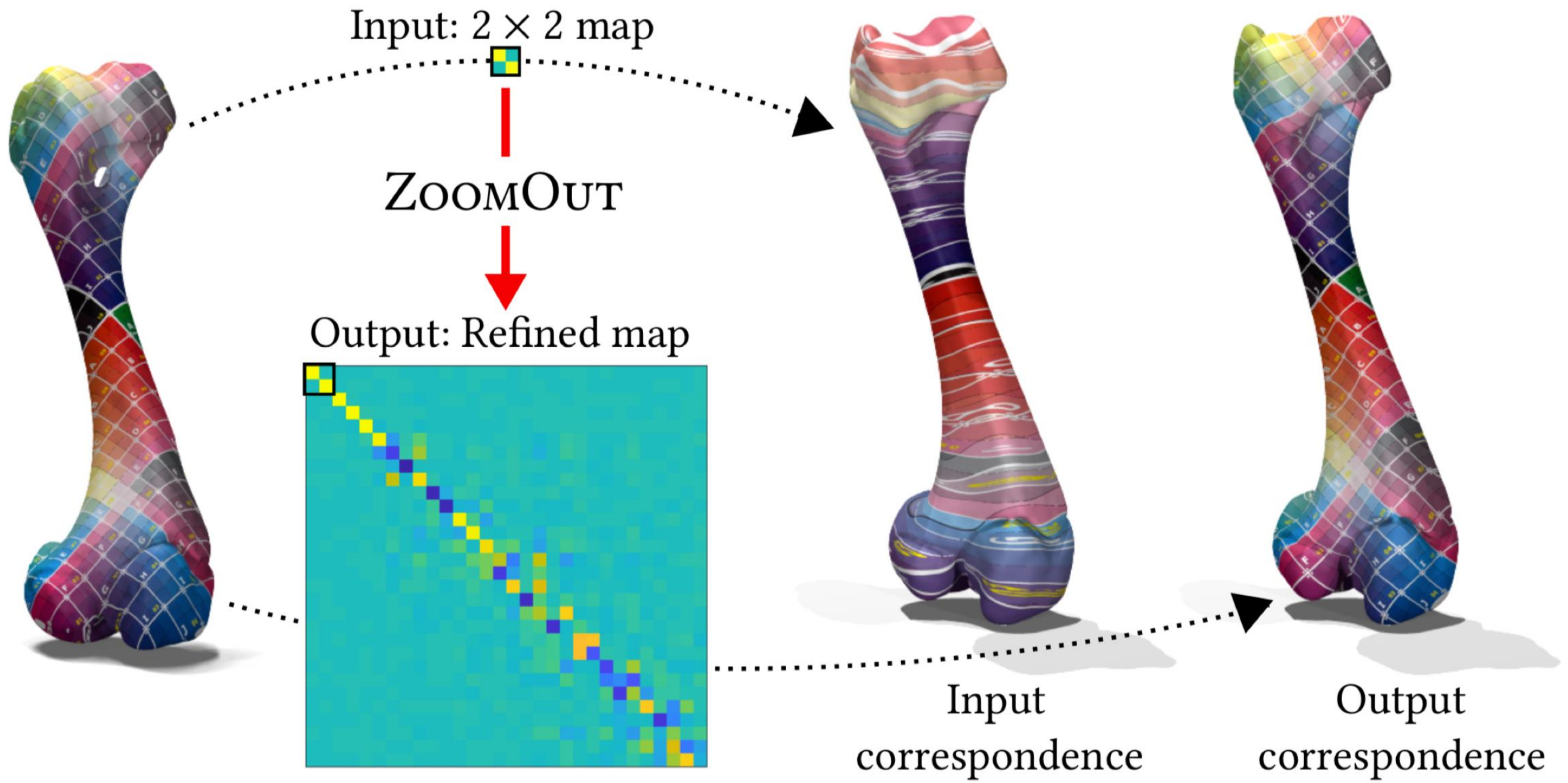


Iteratively solve for $\boldsymbol{C}$ and T ($\boldsymbol{C}$ with fixed size = k)

Ovsjanikov at al., "Functional maps: a flexible representation of maps between shapes", 2012

Spectral Upsampling

Dataset provided by the Natural History Museum in Paris



[ZoomOut: Spectral Upsampling for Efficient Shape Correspondence, *Melzi* et al., 2019]

ZoomOut idea

- Progressively registering the eigenfunctions
- Exploiting the connection between functional and point-to-point map

ZoomOut

- 5 lines of code
- Similar complexity to ICP

```
function [C,P]=ZoomOut(M,N,C,k_final)

for k=size(C,1):k_final-1
  x = knnsearch(N.Phi(:,1:k)*C',M.Phi(:,1:k));
  P = sparse(1:M.n,x,1,M.n,N.n);
  C = M.Phi(:,1:k+1)'*M.A*P*N.Phi(:,1:k+1);
end
```

ZoomOut: Spectral Upsampling for Efficient Shape Correspondence, *Melzi* et al., 2019

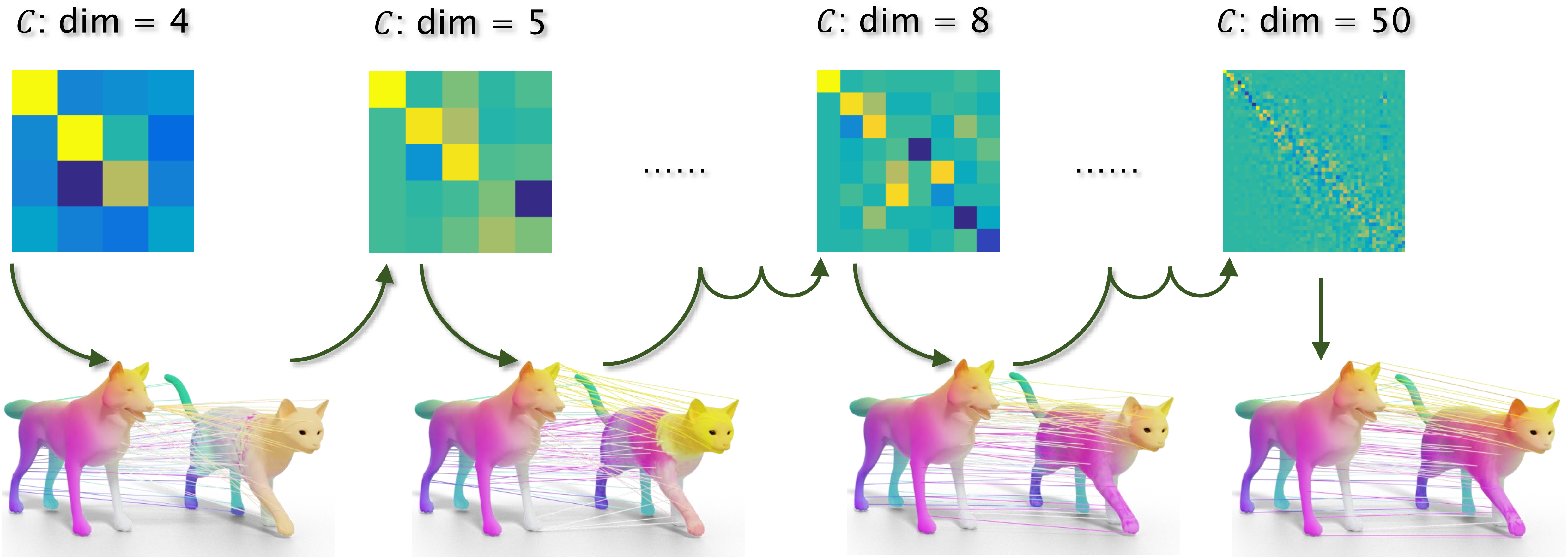
ZoomOut Algorithm

1. Input: an initial map Π and an integer k
2. Solve $C^k = \underset{C}{\operatorname{argmin}} \left\| \Phi_1^k C - \Phi_2^k(\Pi, :) \right\|_F^2$
3. Update $\Pi = \underset{\Pi}{\operatorname{argmin}} \left\| \Phi_1^k C^k - \Phi_2^k(\Pi, :) \right\|_F^2$
4. Update $k = k + 1$
5. Return to step 2.

$\boldsymbol{\Phi_i^k}$ are the first $\boldsymbol{k}$ eigenfunctions of $\boldsymbol{S_i}$

ZoomOut: Spectral Upsampling for Efficient Shape Correspondence, *Melzi* et al., 2019

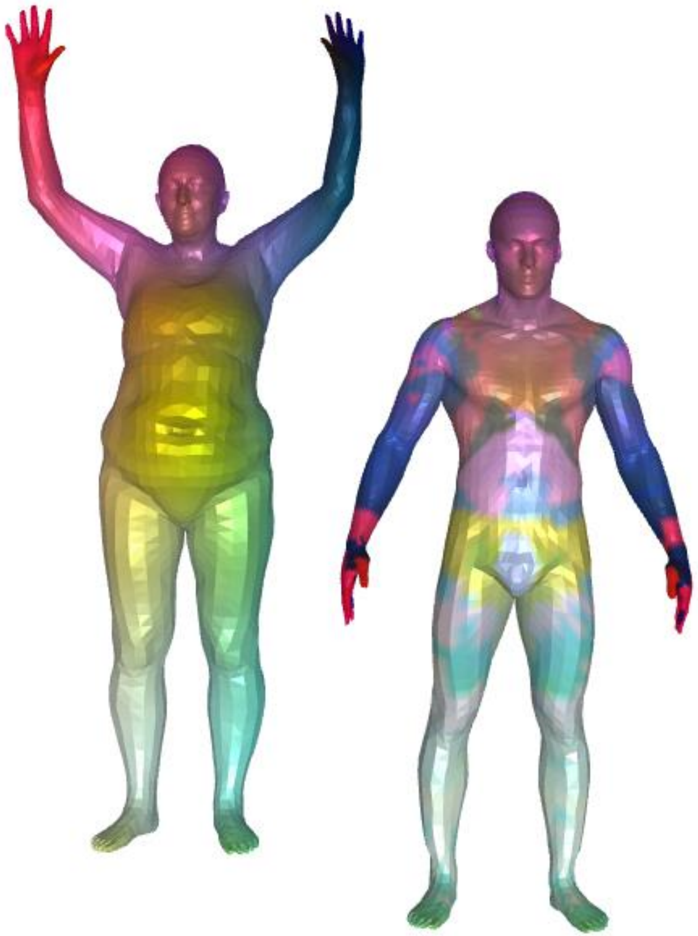
ZoomOut Visualization



[ZoomOut: Spectral Upsampling for Efficient Shape Correspondence, *Melzi* et al., 2019]

Live Demo

- Simple demo on non-rigid matching



https://github.com/riccardomarin/EG22_Tutorial_Spectral_Geometry

Inverse Computational Spectral Geometry

Arianna Rampini Riccardo Marin Simone Melzi Luca Cosmo
Emanuele Rodolà Maks Ovsjanikov Michael Bronstein

The inverse problem in applications

SAPIENZA
UNIVERSITÀ DI ROMA

GLADIA

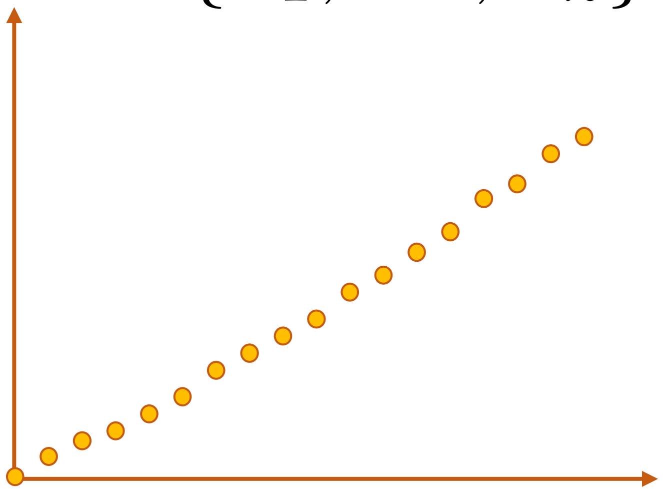
Outline

- The inverse eigenvalue problem
- Methods: optimization
- Methods: data-driven approaches
- Applications
- Demo

The inverse eigenvalue problem

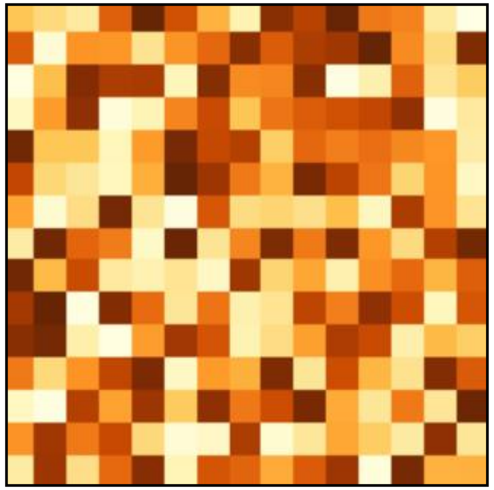
Reconstruction of a matrix from prescribed spectral data:

$$\lambda = \{\lambda_1, \ldots, \lambda_n\}$$



$\longrightarrow$

$$A$$

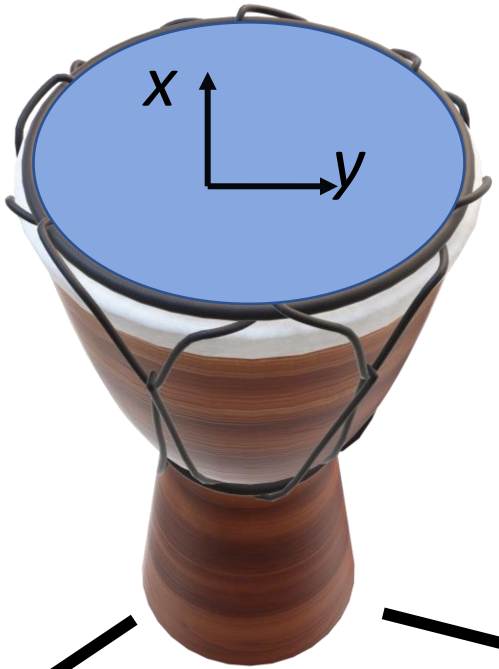


$$\sigma(A) = \lambda$$

Chu M.T. *Inverse eigenvalue problems*. SIAM review. 1998.

«Can one hear the shape of the drum?»

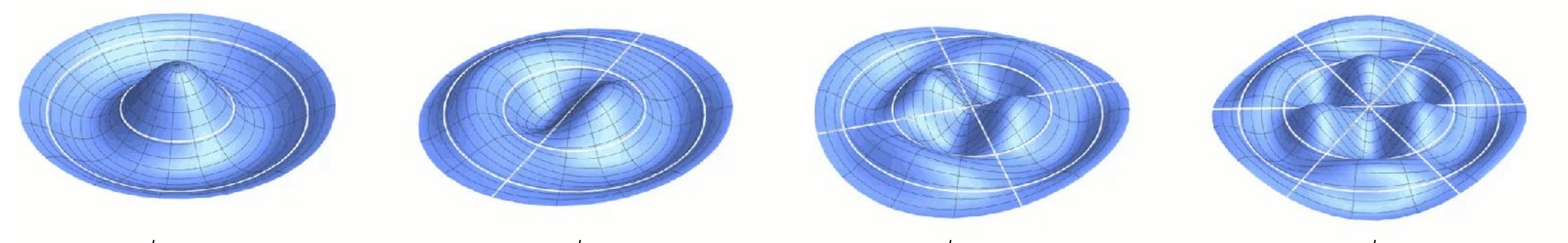
«Can one hear the shape of the drum?»



Wave equation

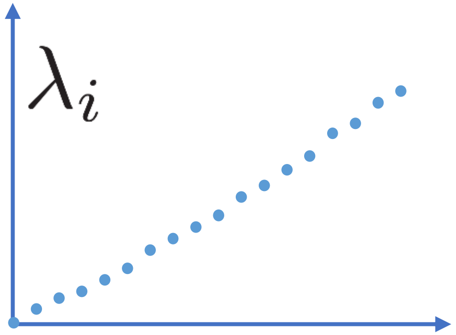
$$\Delta u(x, y; t) = -\frac{\partial^2 u(x,y;t)}{\partial t^2}$$

Eigenfunctions



ϕ_1 ϕ_2 ϕ_3 ϕ_4

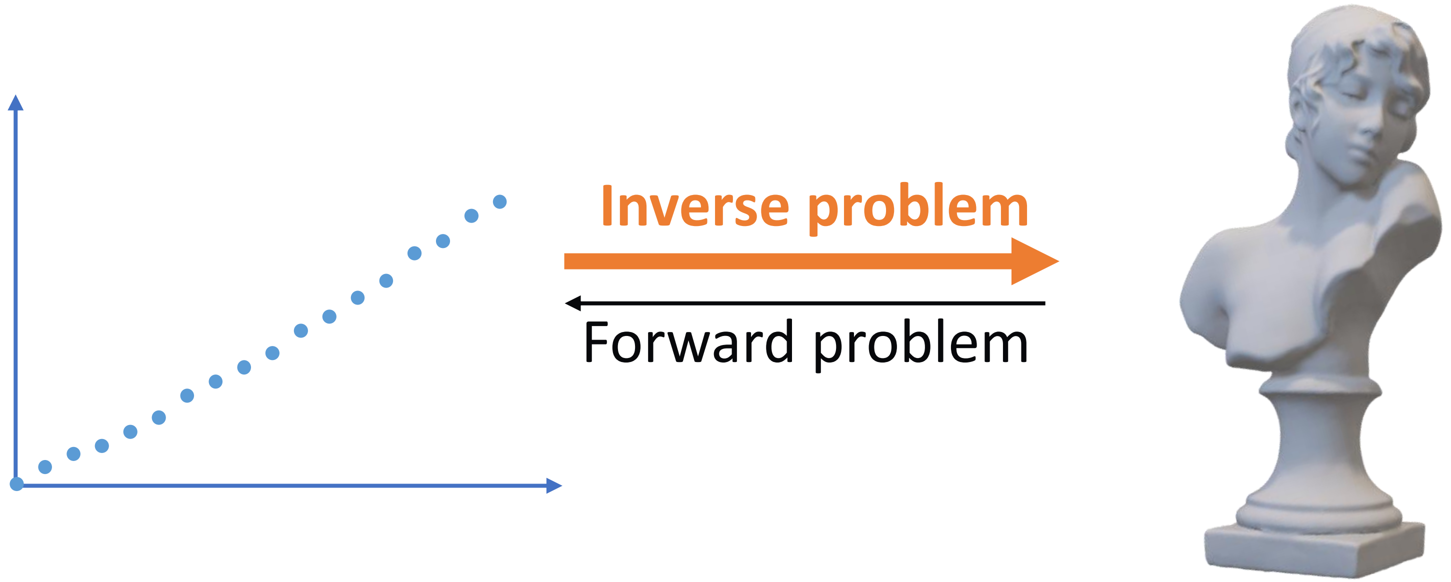
Spectrum / eigenvalues



Our drum

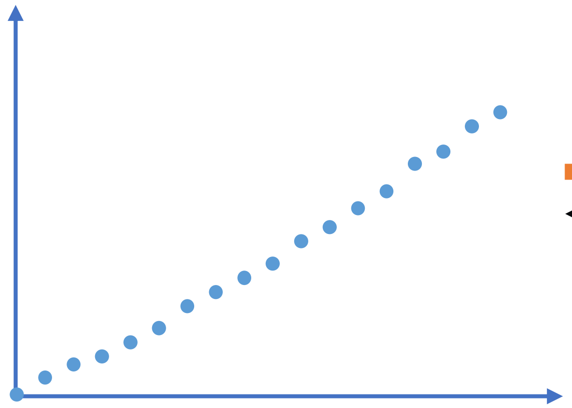
Solvability

Can we recover the shape from the eigenvalues?



Computability

...how?



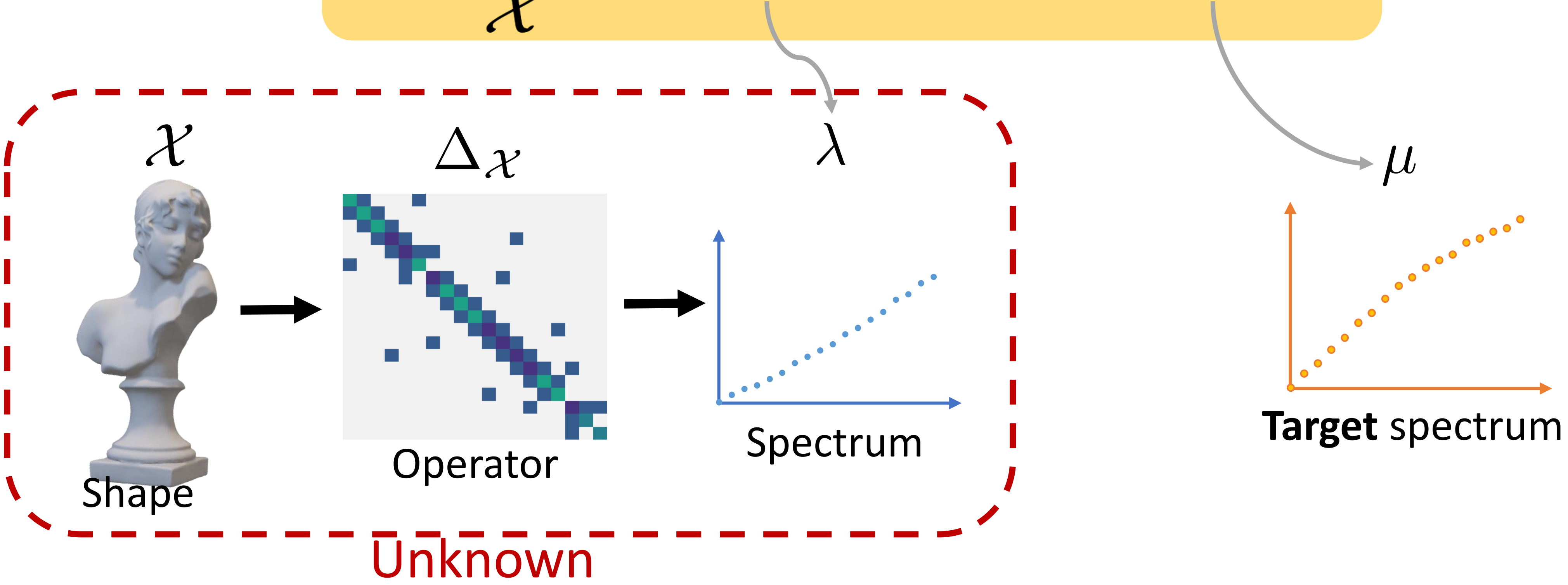
Inverse problem

Forward problem

Isospectralization

Optimization directly on the 3D coordinates:

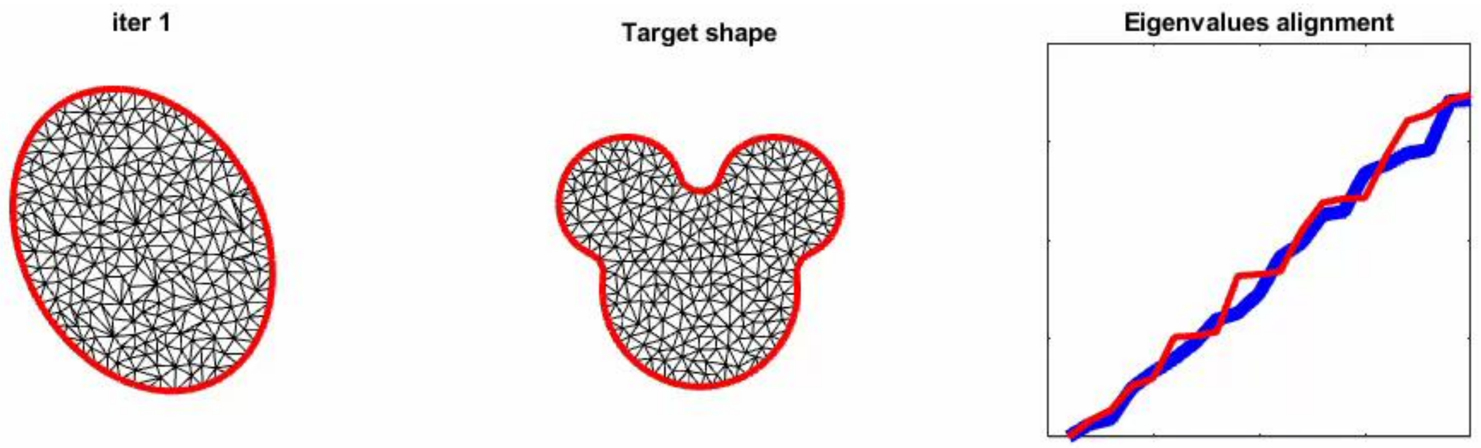
$$\arg\min_{\mathcal{X}} \|\boldsymbol{\lambda}(\boldsymbol{\Delta}_{\mathcal{X}}) - \boldsymbol{\mu}\|_{\omega}$$



Cosmo et al., *Isospectralization, or how to hear shape, style and correspondence* (CVPR 2019)

Mickey from spectrum

$$\underset{\mathcal{X}}{\arg\min} \left\| \boldsymbol{\lambda}\left(\boldsymbol{\Delta}_{\mathcal{X}}\right) - \boldsymbol{\mu} \right\|_{\omega}$$



3D: several local minima

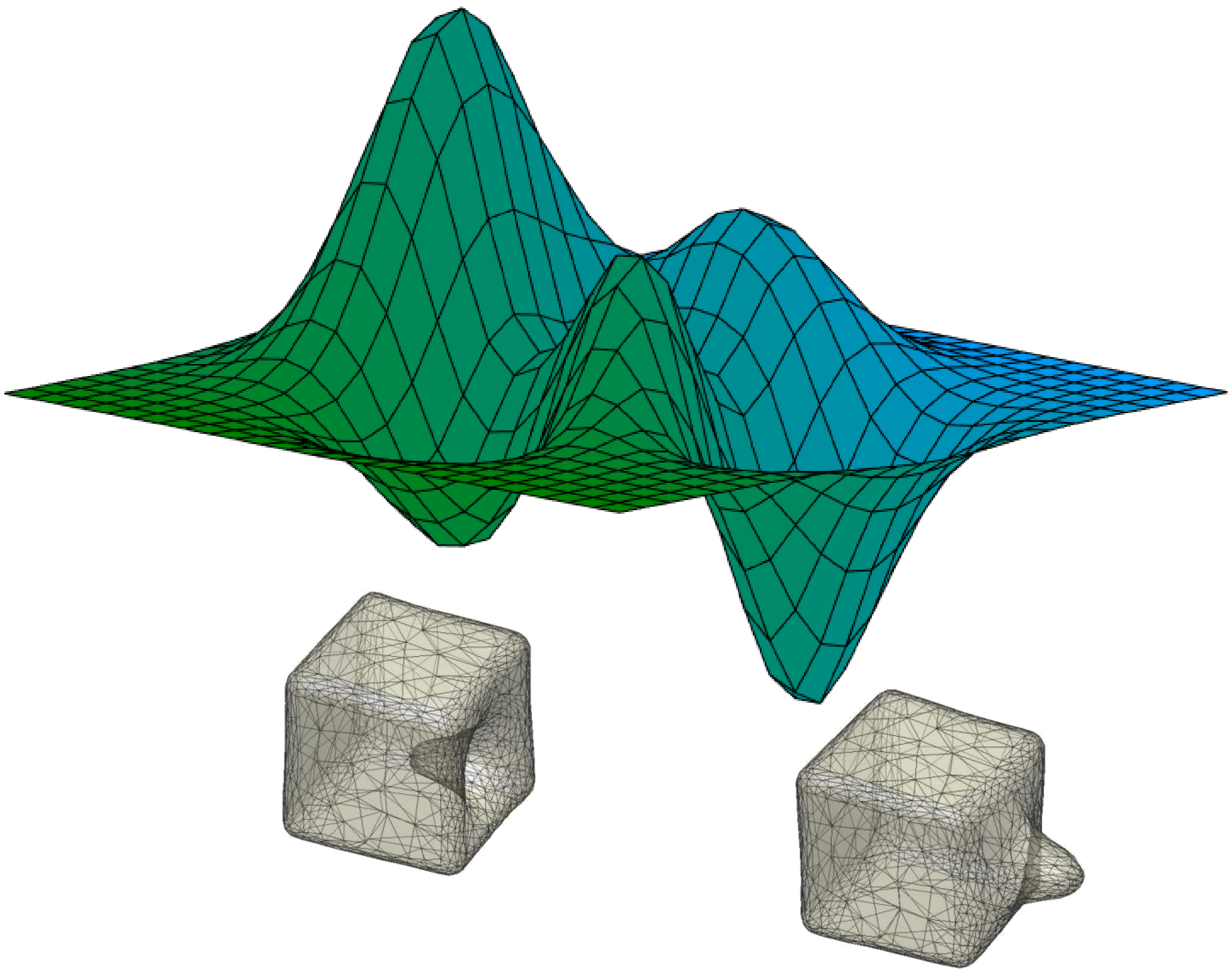
symmetries and **isometries**

↓

existence of several

local minima in the

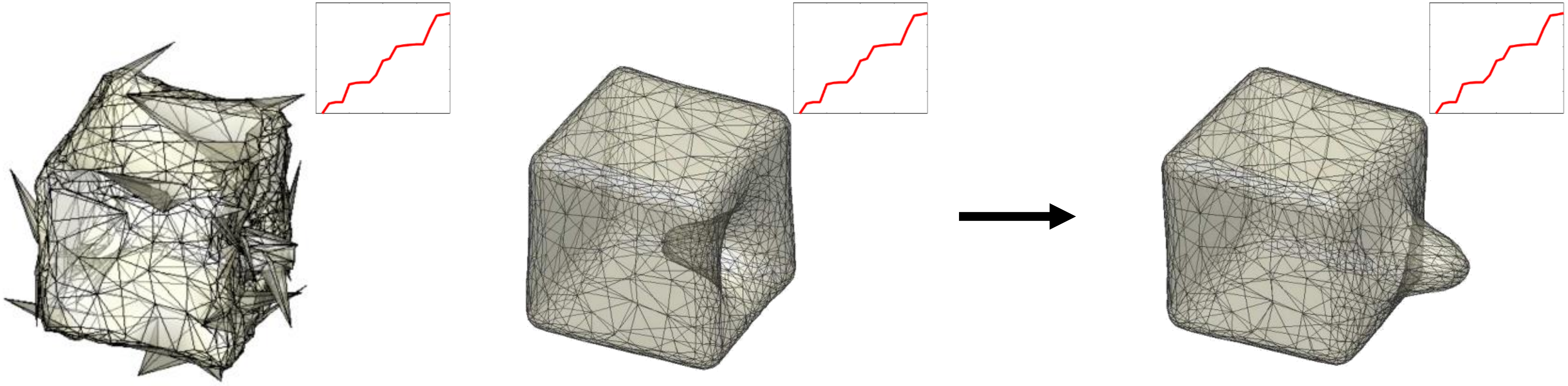
isospectralization problem



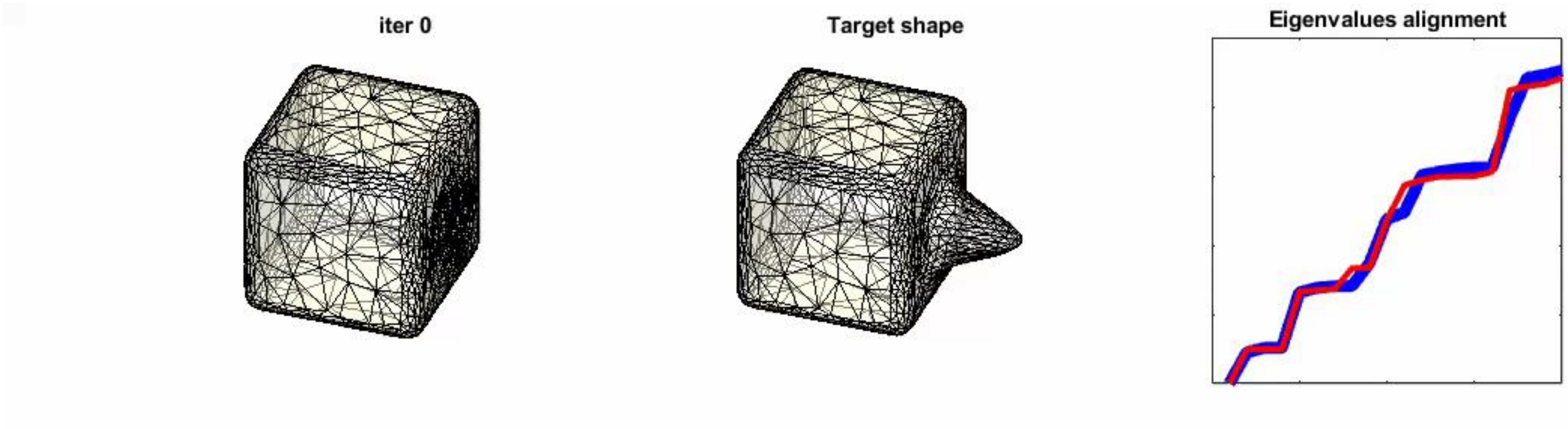
Regularizers

$$\min_{\mathbf{X}\in\mathbb{R}^{n\times d}} \|\boldsymbol{\lambda}(\boldsymbol{\Delta}(\mathbf{X})) - \boldsymbol{\mu}\|_\omega + \rho_X(\mathbf{X})$$

To promote smoothness and maximize volume:



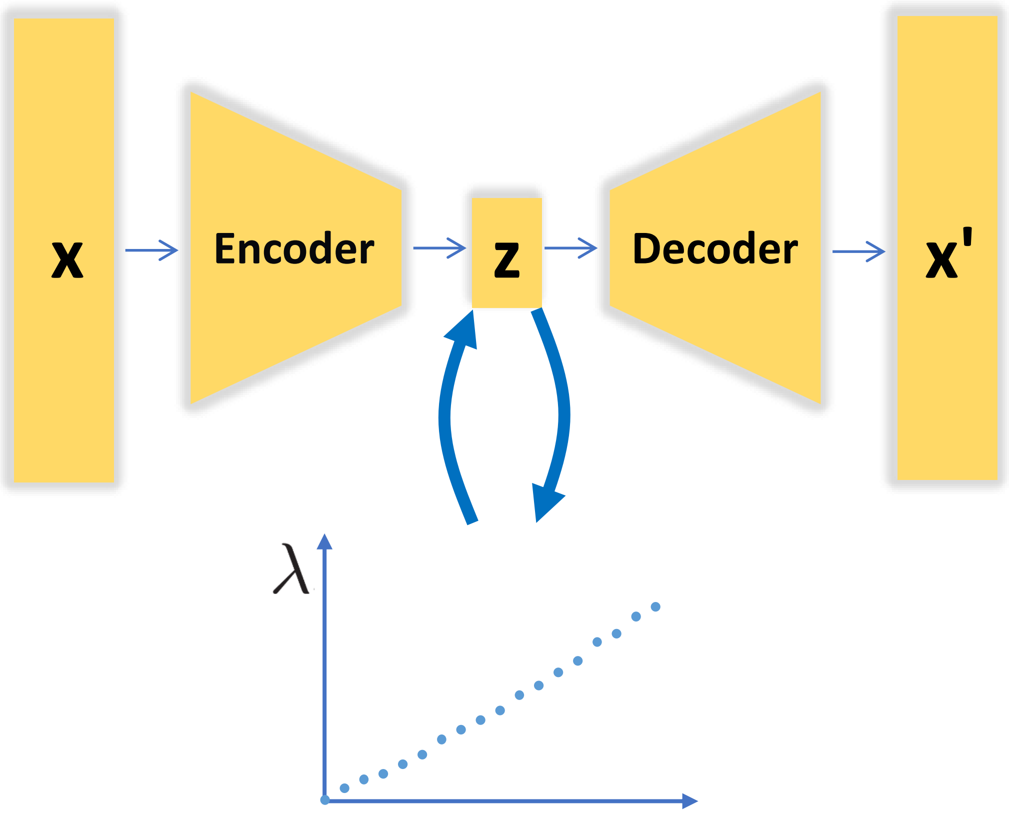
Isospectralization on surfaces



Cosmo et al., *Isospectralization, or how to hear shape, style and correspondence* (CVPR 2019)

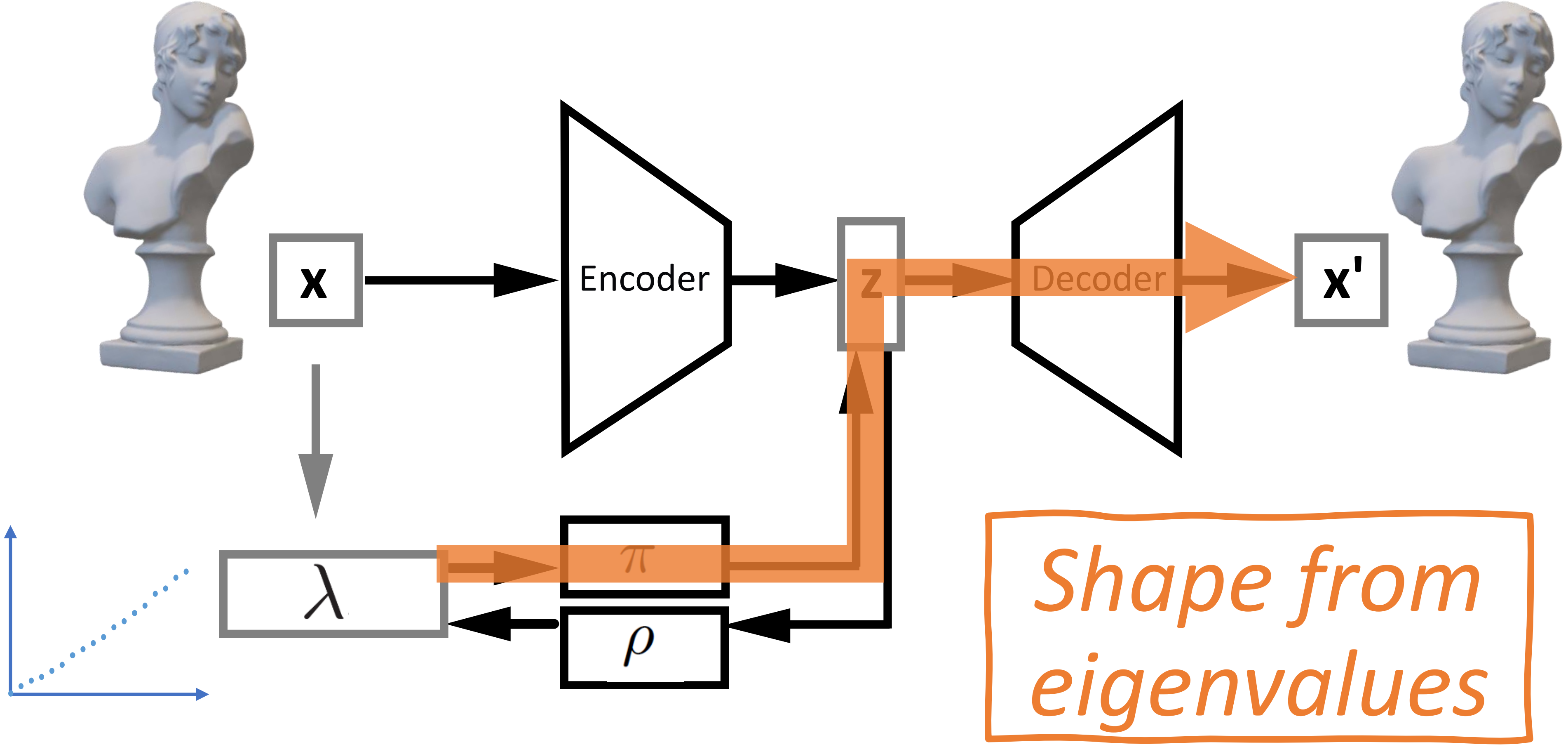
Learning to hear shapes

We can use a **cycle-consistent module** to map latent vectors to spectra:



Marin et al., *Instant recovery of shape from spectrum via latent space connections* (3DV 2020)

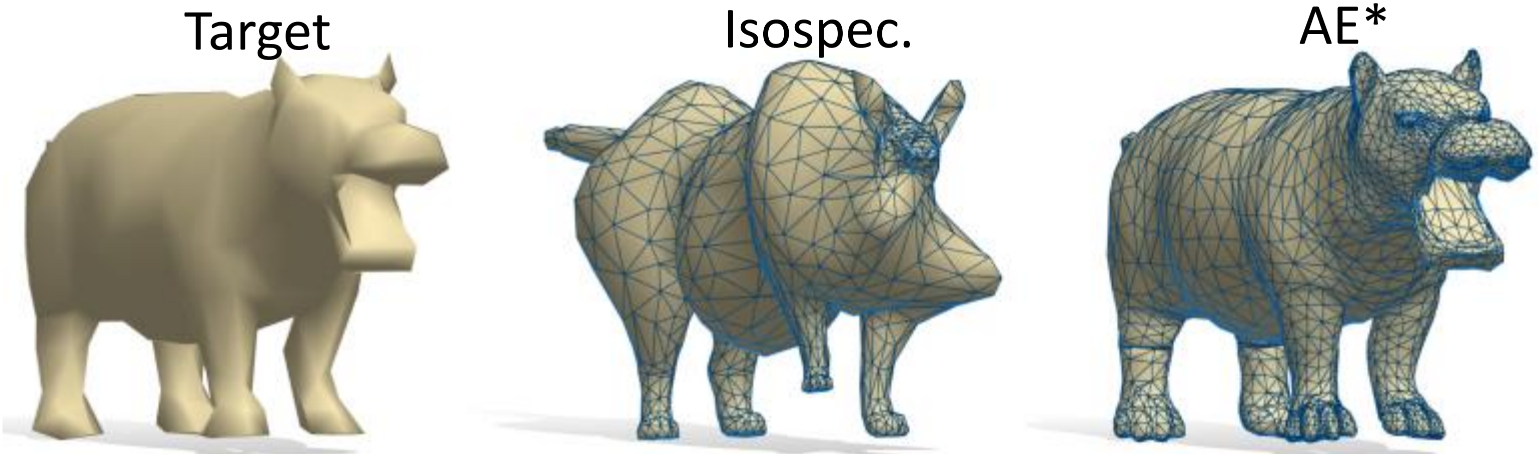
Learning to hear shapes



Marin et al., *Instant recovery of shape from spectrum via latent space connections* (3DV 2020)

Pros of data-driven approach

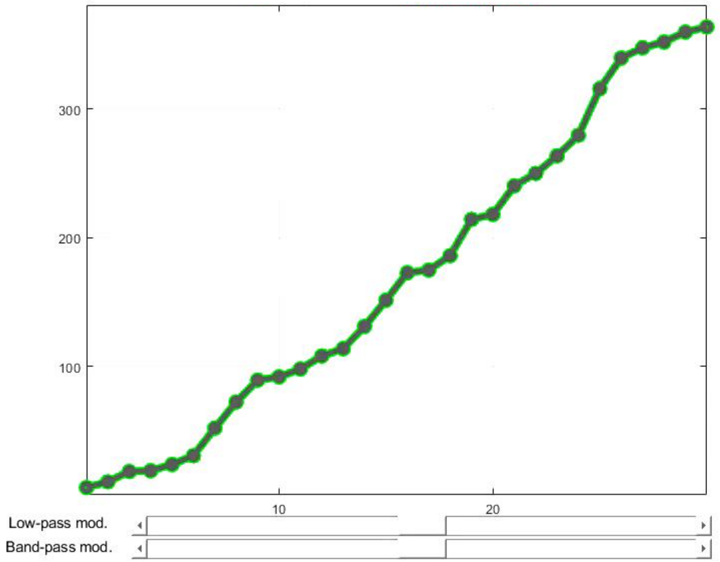
- Fast: *instant* recovery
- Accuracy
- No dependence from initialization
- Larger meshes and point clouds
- No need of regularizers



* Marin et al., Instant recovery of shape from spectrum via latent space connections (3DV 2020)

Pros of data-driven approach

Input: **spectrum**

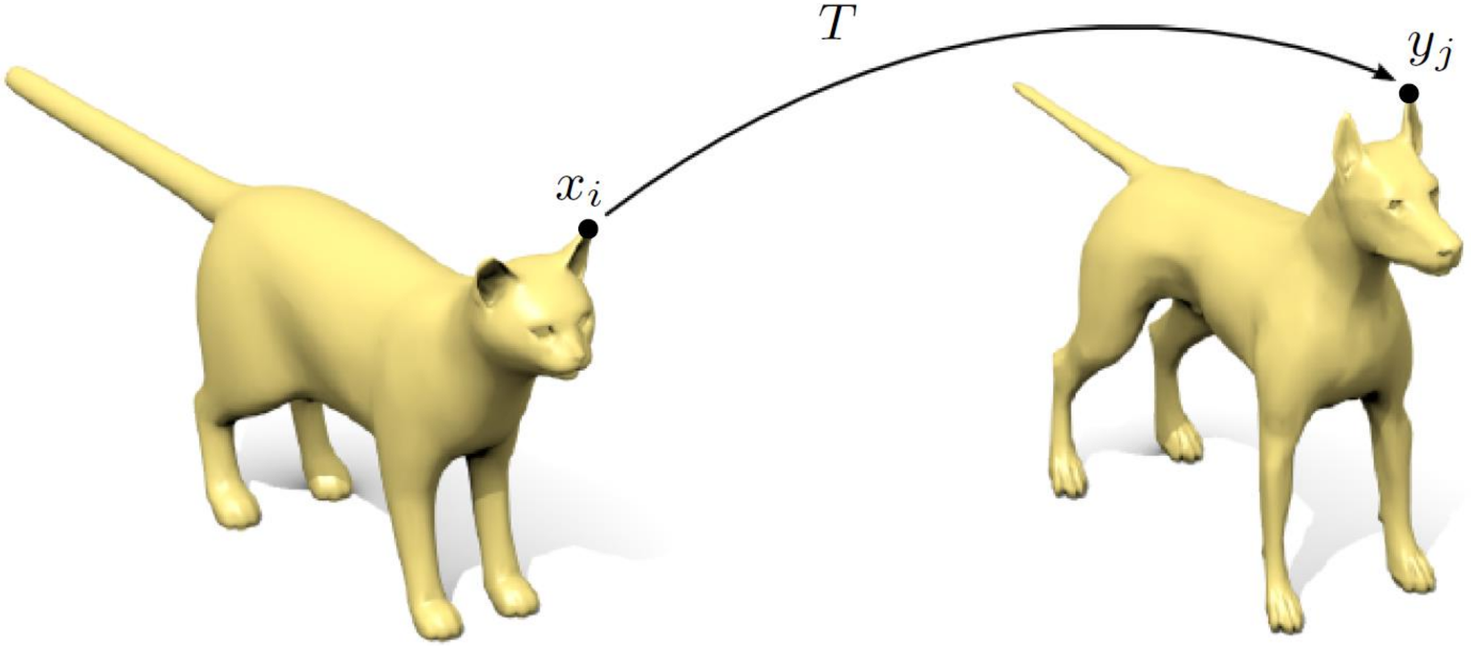


Output: **shape**

Marin et al., *Instant recovery of shape from spectrum via latent space connections* (3DV 2020)

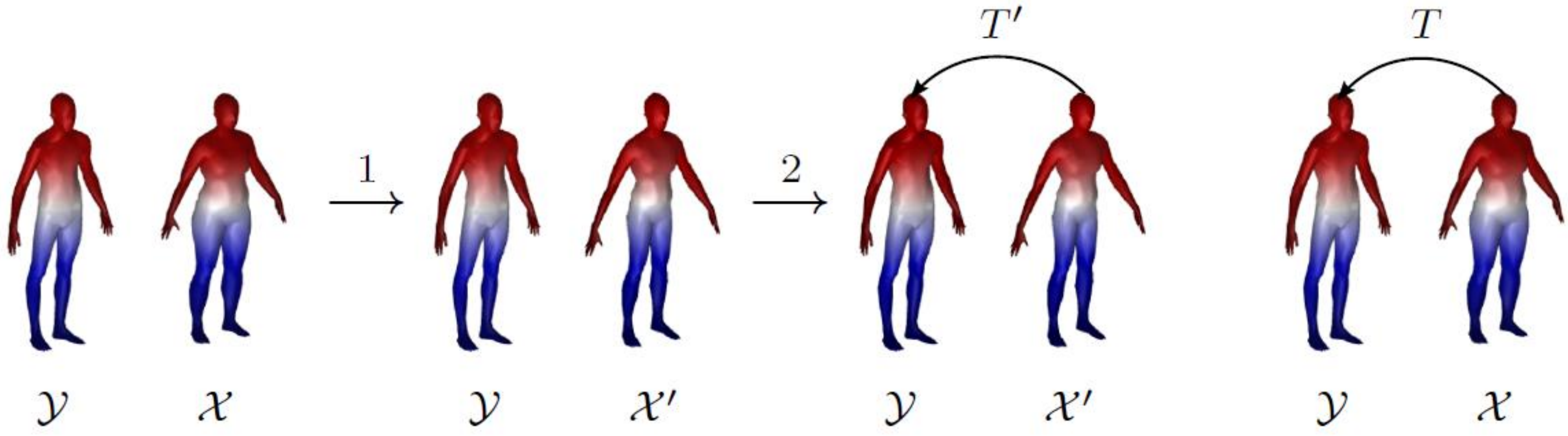
Non-isometric shape matching

Goal



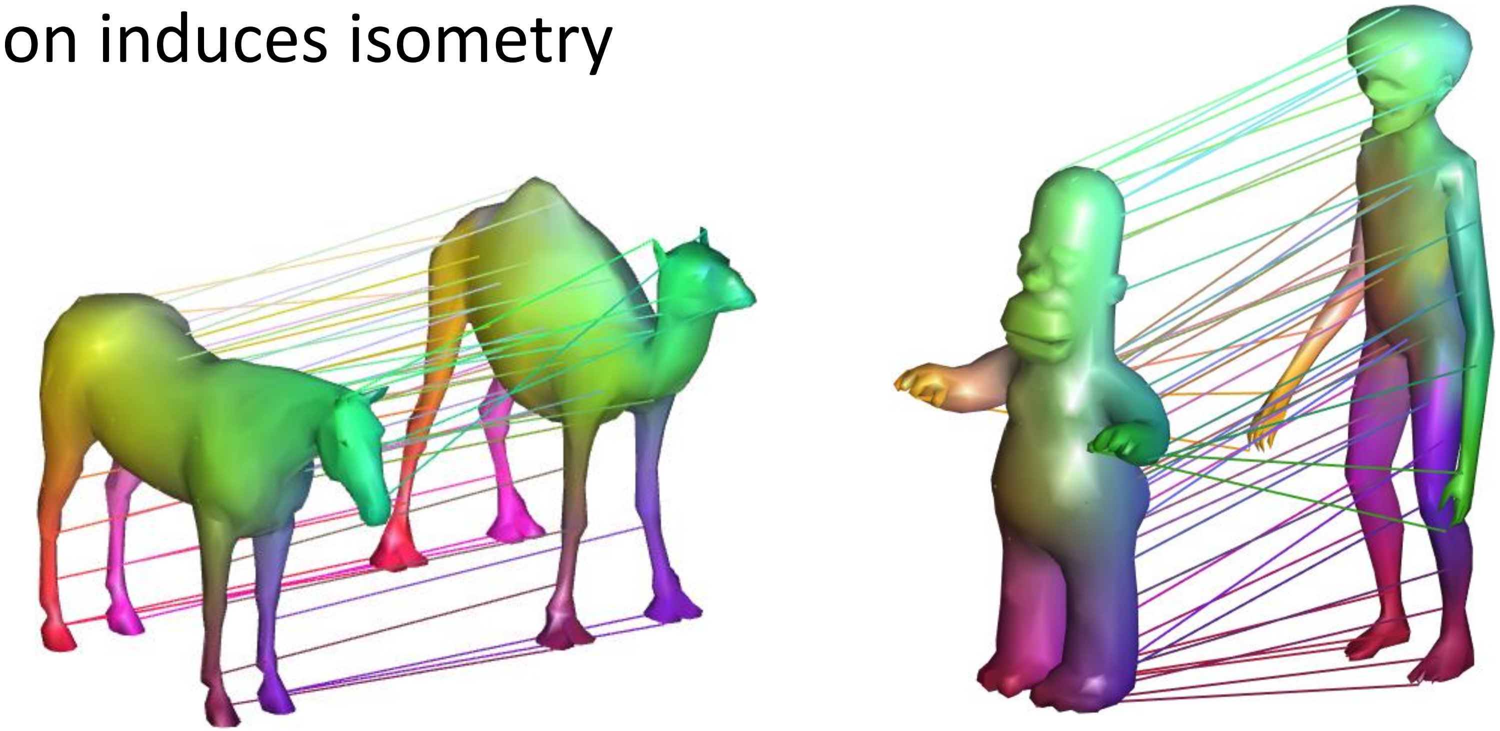
Isospectralization

- Preprocessing step in Functional Map based matching algorithms
- Isospectralization induces isometry



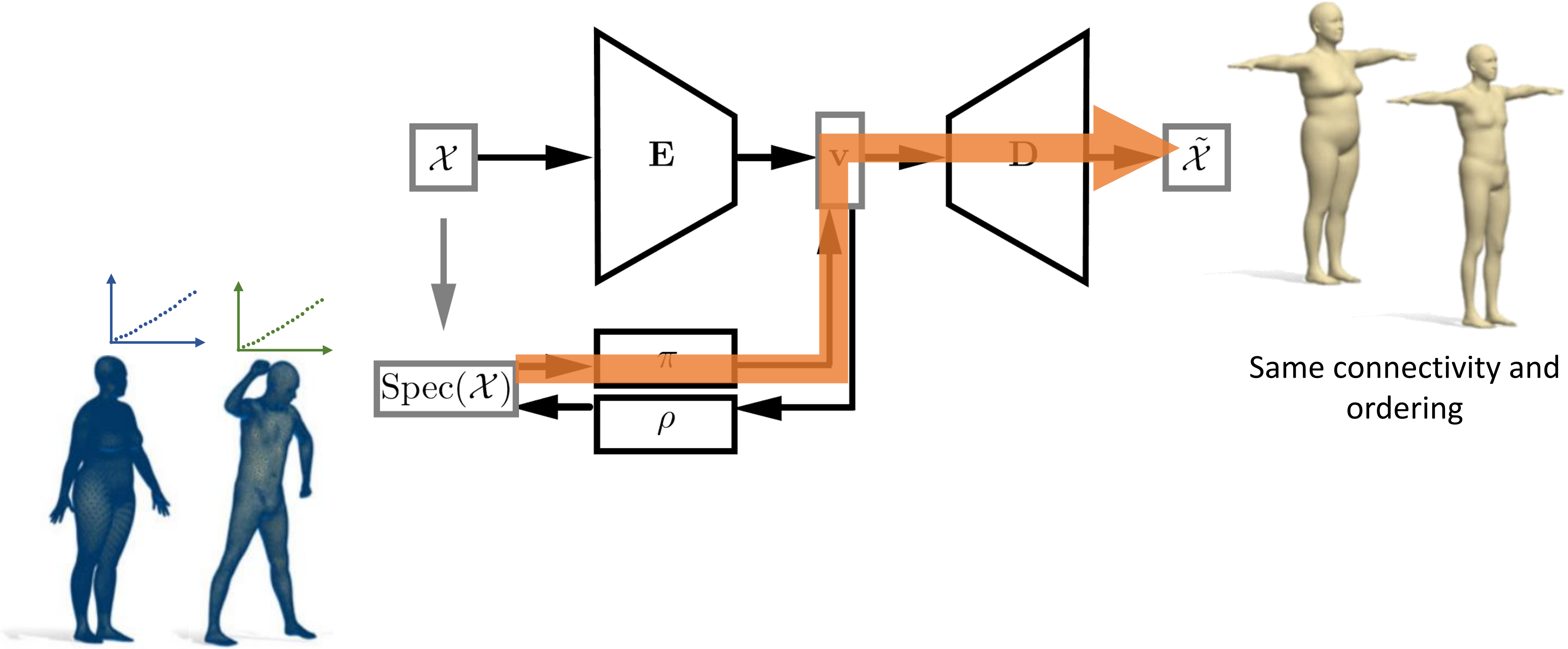
Isospectralization

- Preprocessing step in Functional Map based matching algorithms
- Isospectralization induces isometry

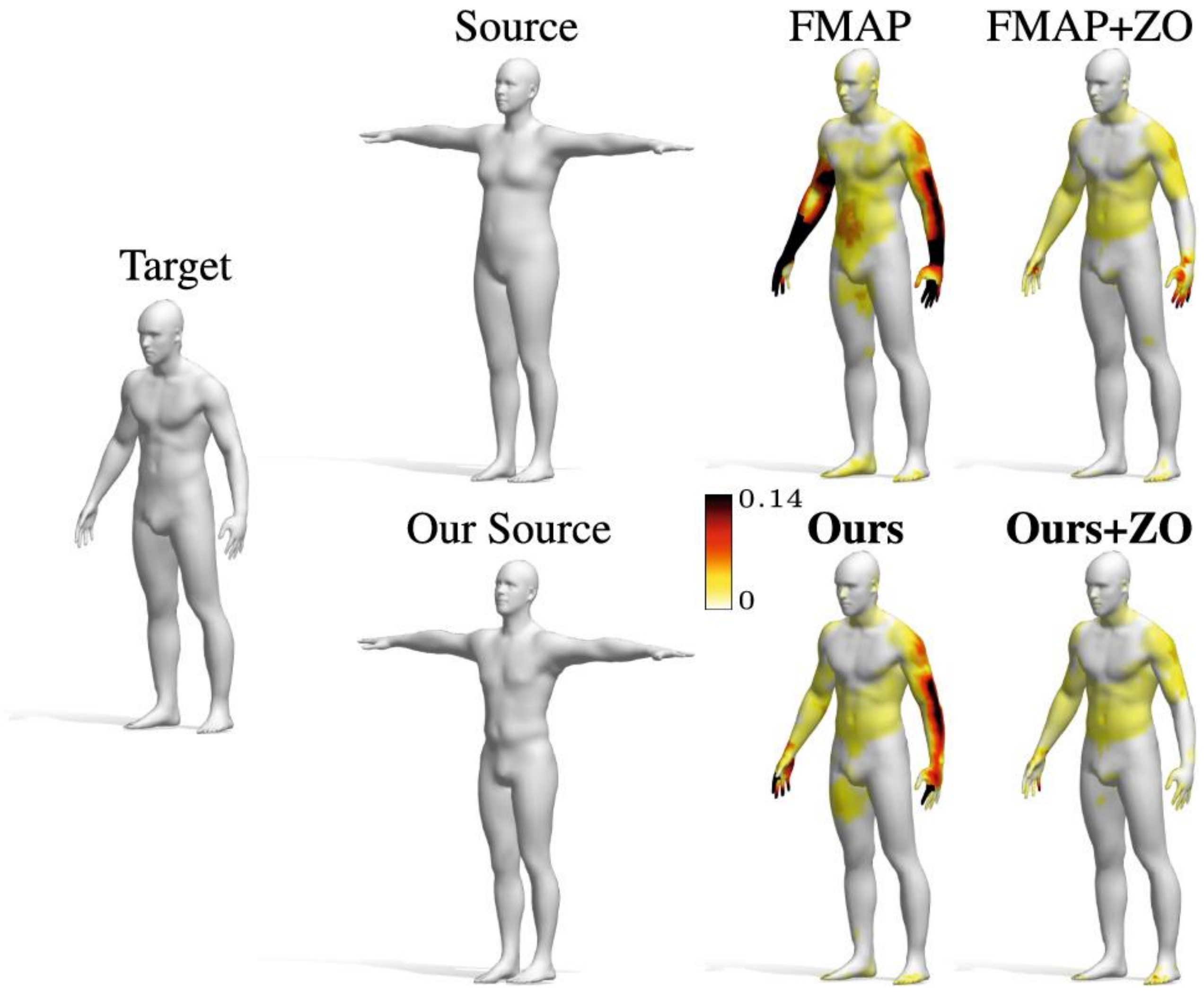


After isospectralization

Data-driven approach

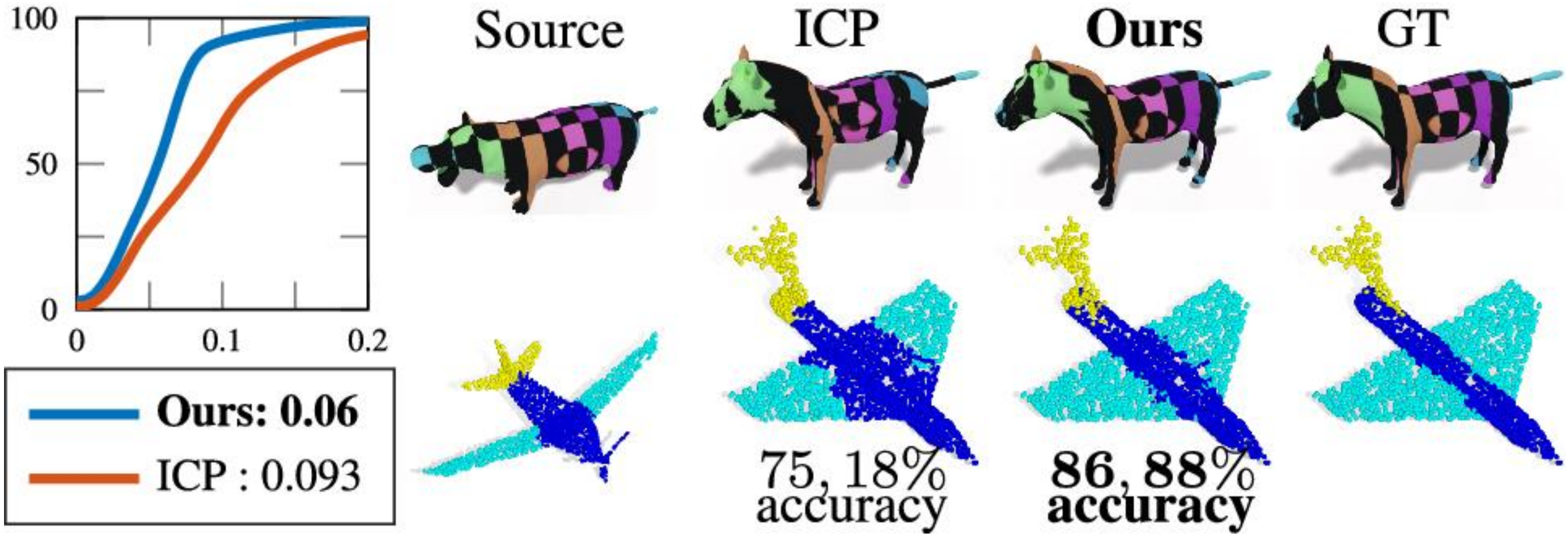


Results



Marin et al., *Spectral Shape Recovery and Analysis Via Data-driven Connections* (IJCV 2021)

Results: segmentation and texture transfer

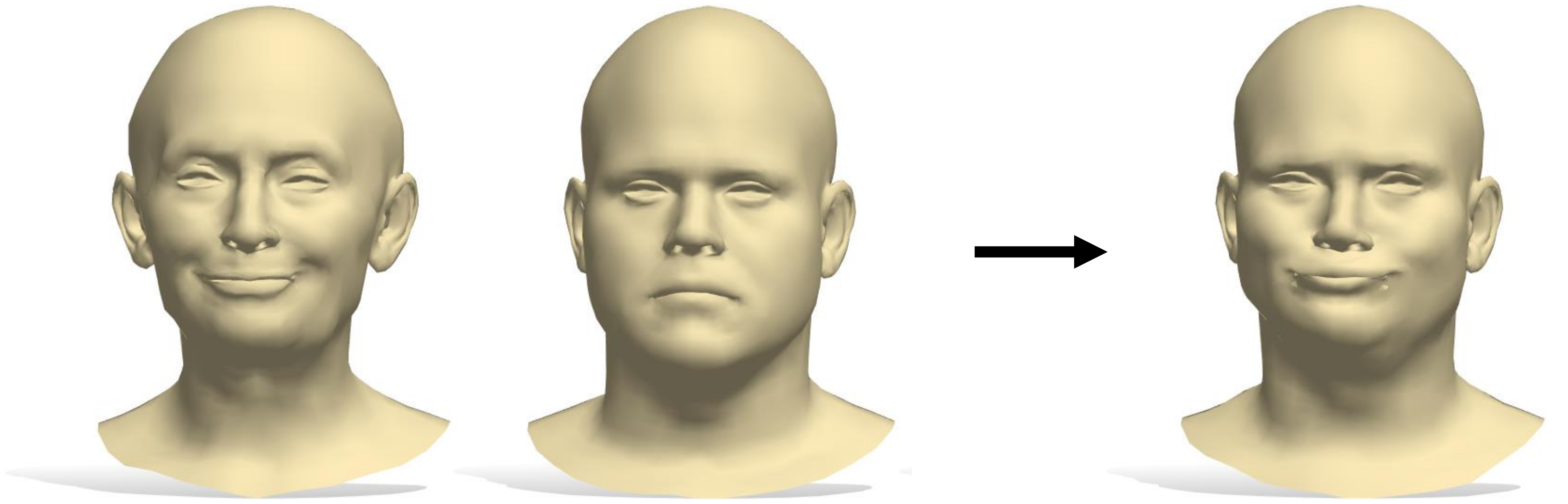


Marin et al., *Spectral Shape Recovery and Analysis Via Data-driven Connections* (IJCV 2021)

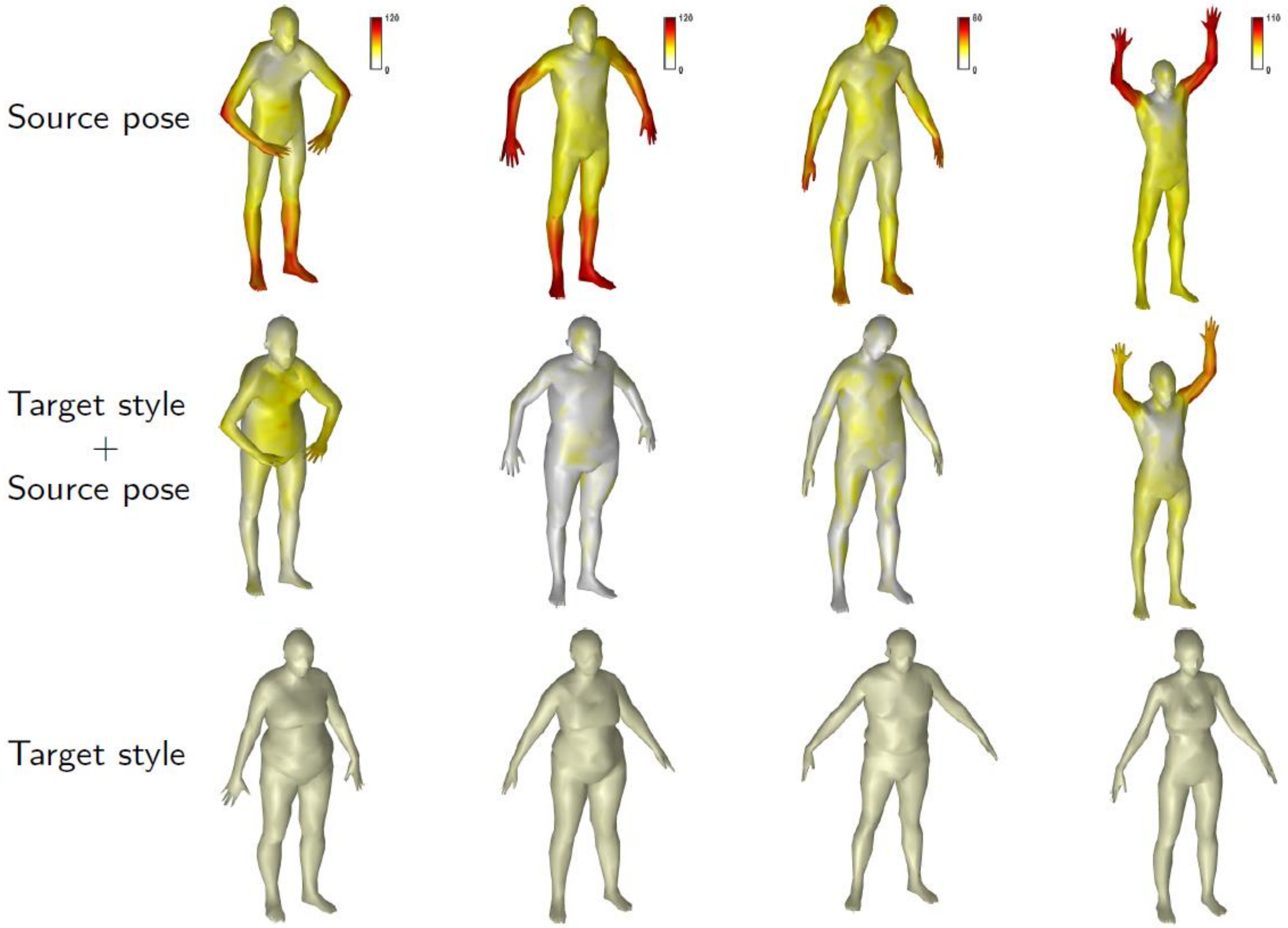
Style transfer

Goal

Input: **pose and style donors** Output: **new shape**



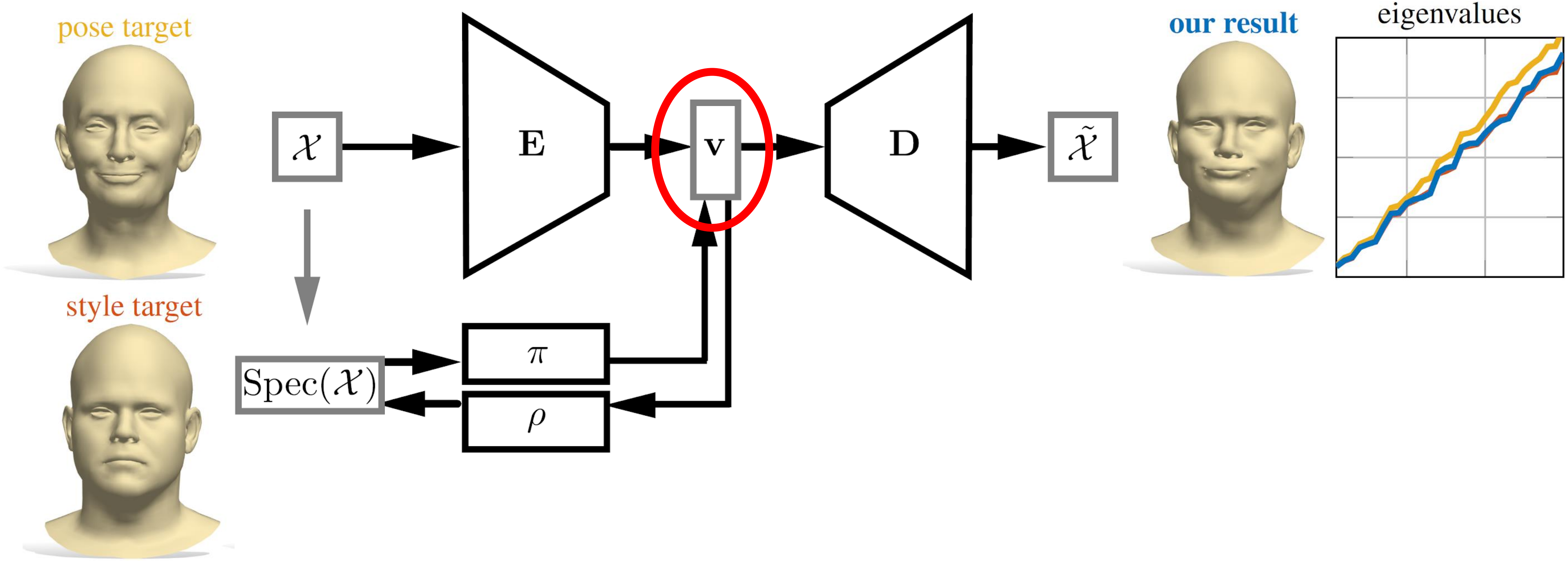
Isospectralization



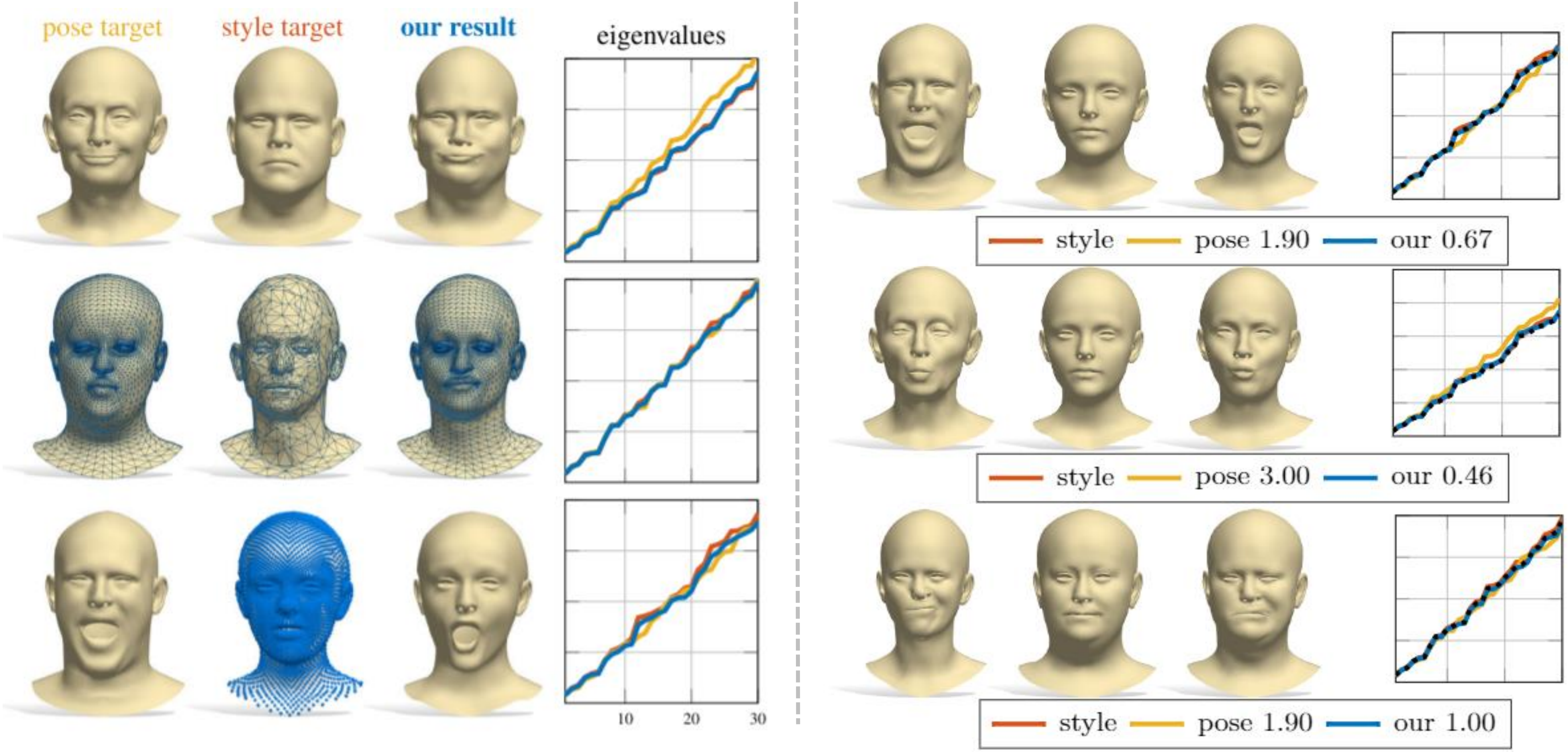
Cosmo et al., *Isospectralization, or how to hear shape, style and correspondence* (CVPR 2019)

Data-driven approach

$$\min_{\mathbf{v}} \|\mathrm{Spec}(\mathcal{X}_{\mathrm{style}}) - \rho(\mathbf{v})\|_2^2 + w\|\mathbf{v} - E(\mathcal{X}_{\mathrm{pose}})\|_2^2$$



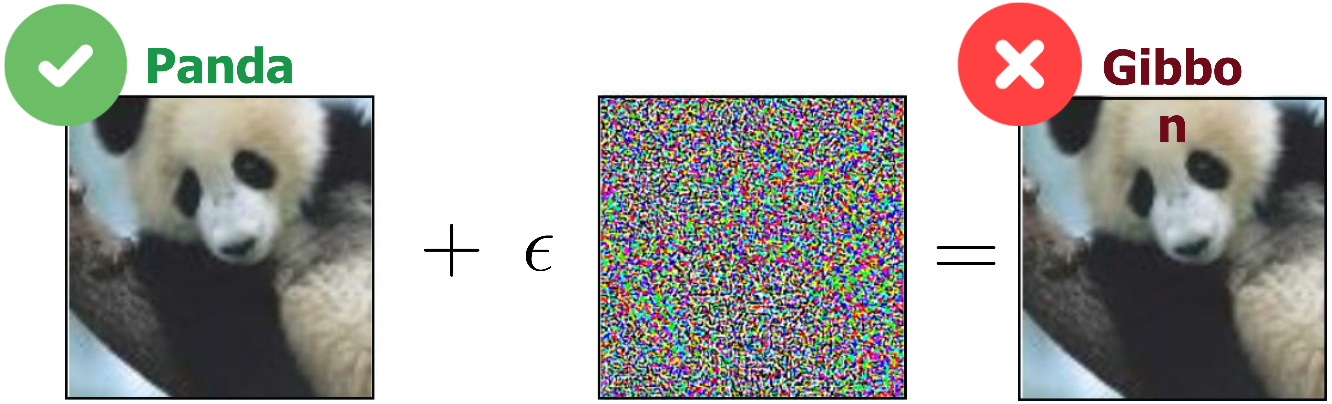
Results



Marin et al., *Instant recovery of shape from spectrum via latent space connections* (3DV 2020)

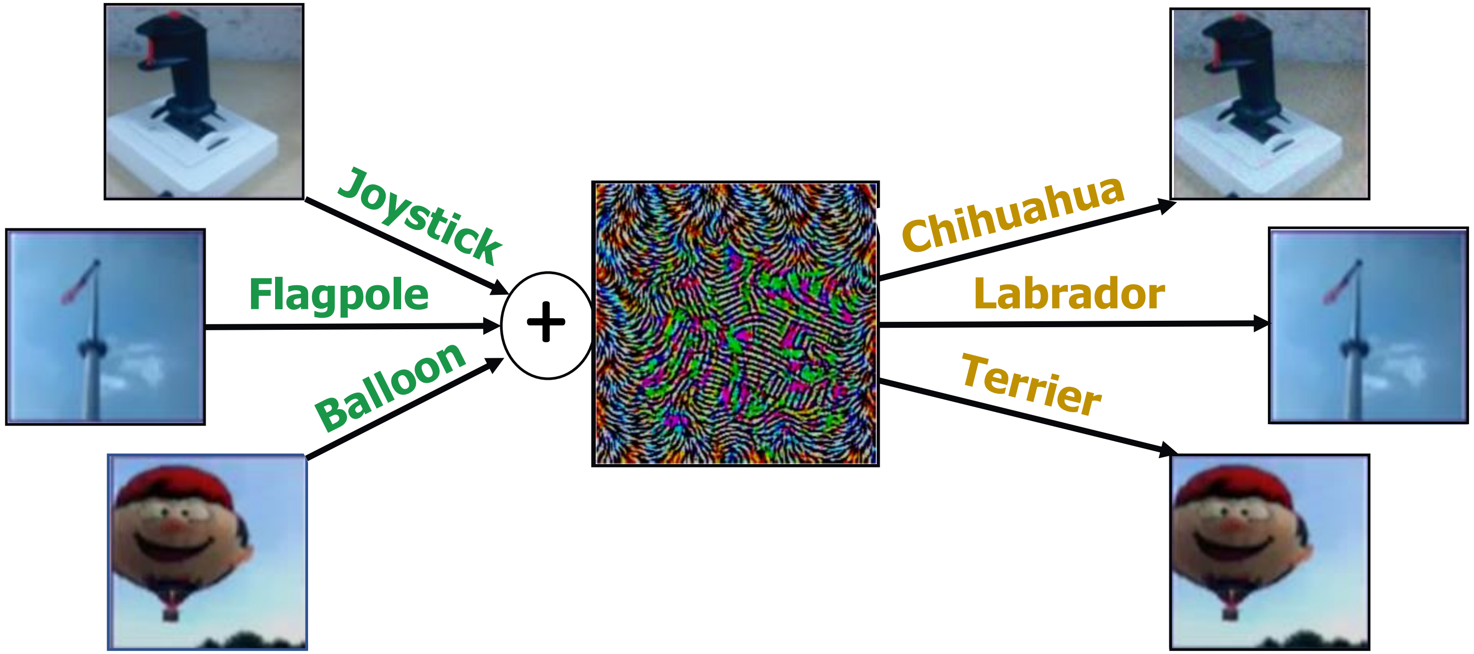
Adversarial attacks

Adversarial attacks



Goodfellow et al., *Explaining and harnessing adversarial examples*, ICLR (2014).

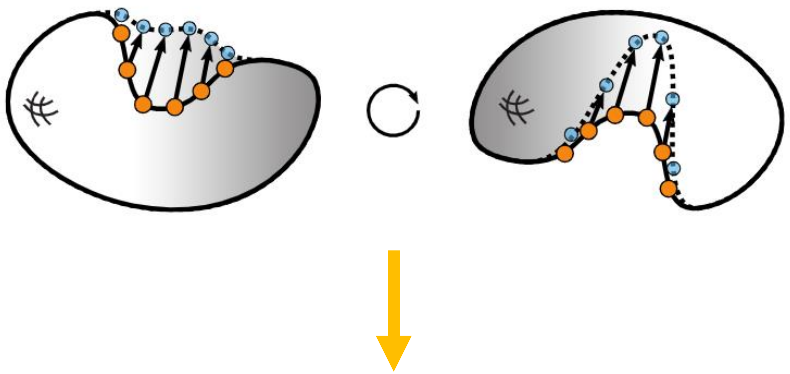
Universal adversarial attacks



Moosavi-Dezfooli et al. “Universal adversarial perturbations”. CVPR (2017)

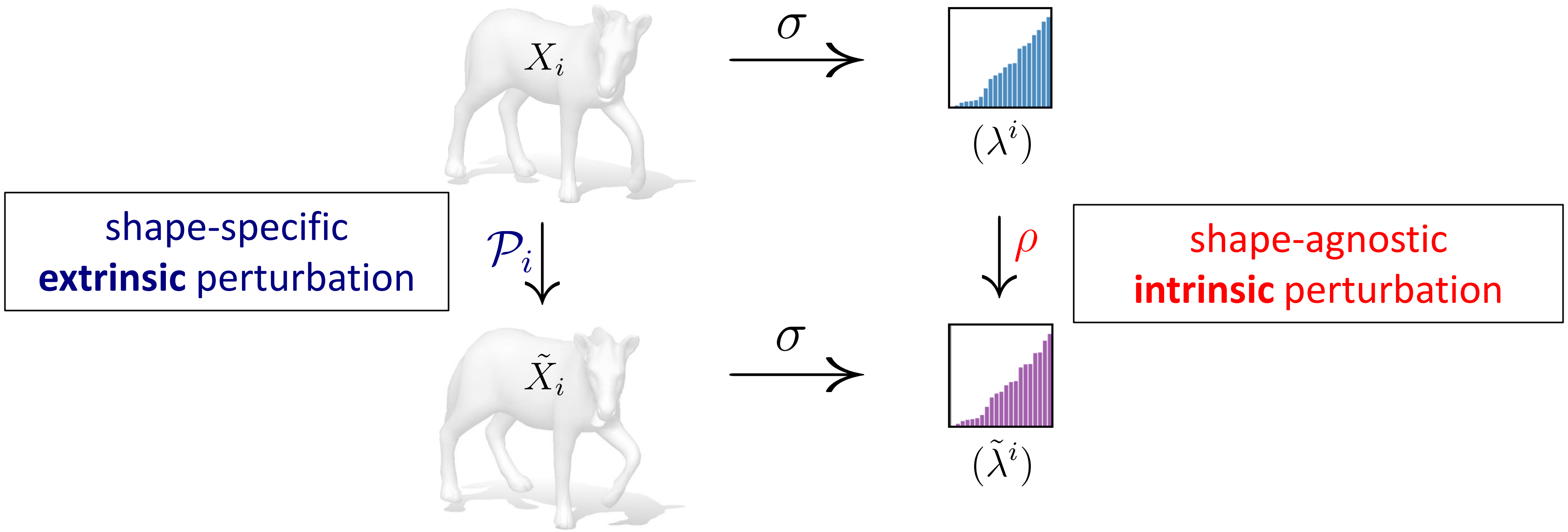
Universal attacks for deformable shapes

An extrinsic perturbation needs correspondence and can not be *deformation-invariant*.

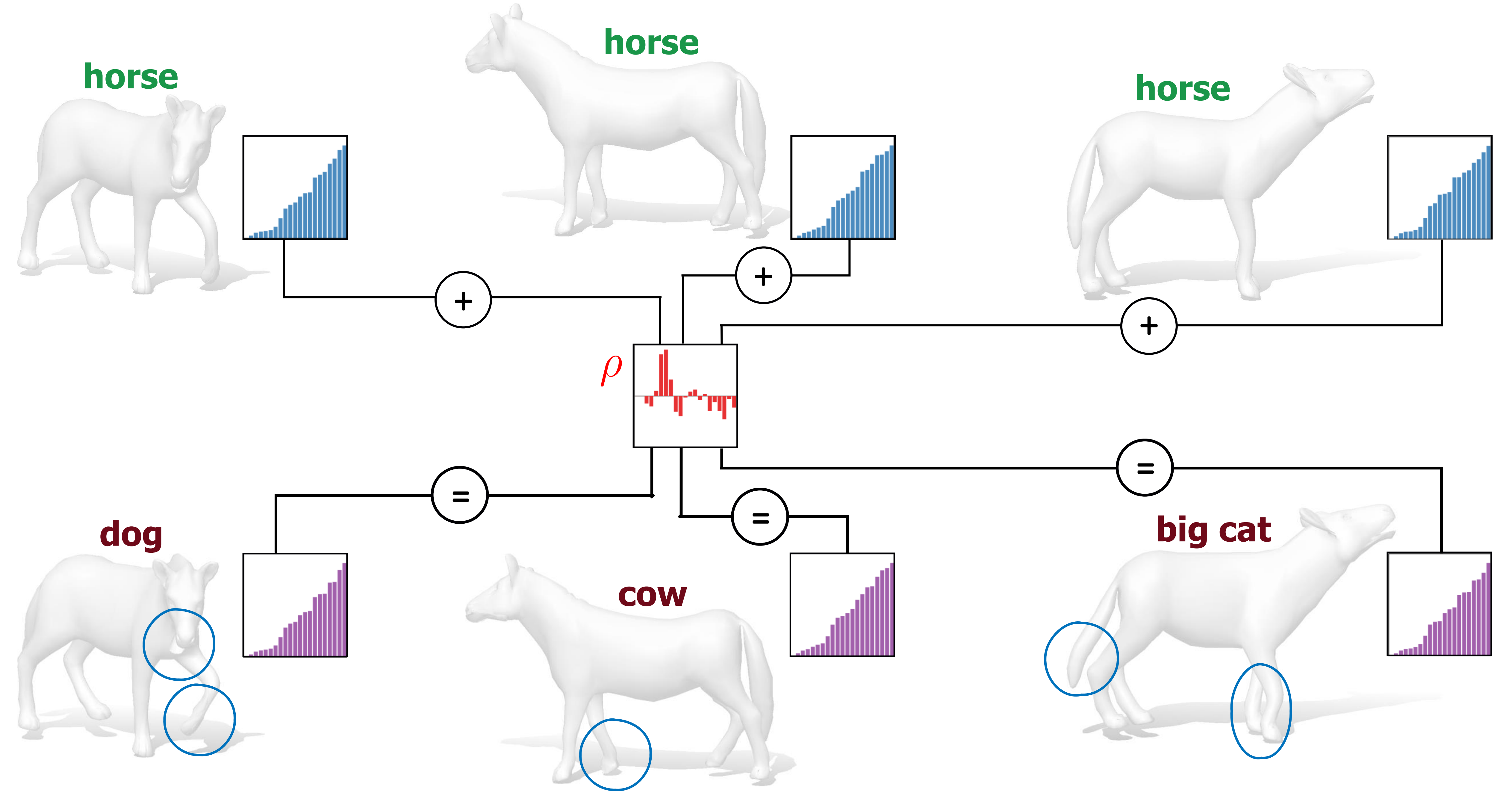


spectral domain

Perturbation in the spectral domain

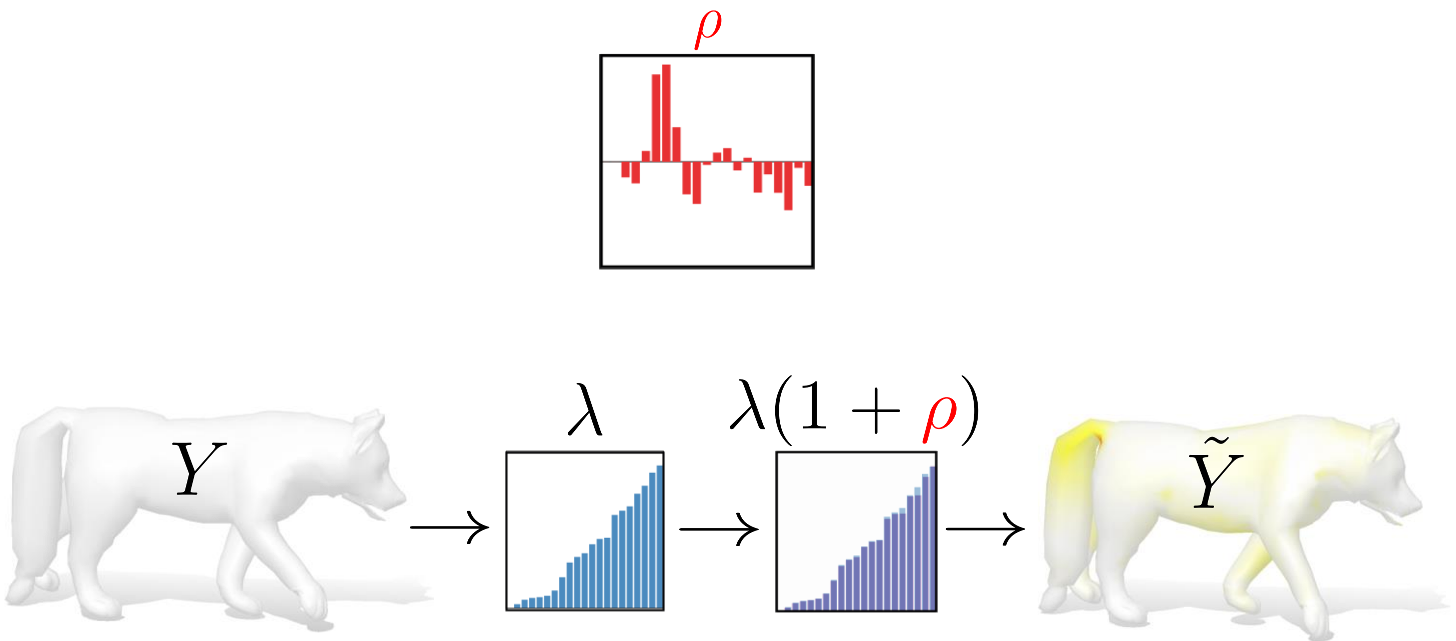


Rampini et al. "Universal Spectral Adversarial Attacks for Deformable Shapes" (CVPR 2021)



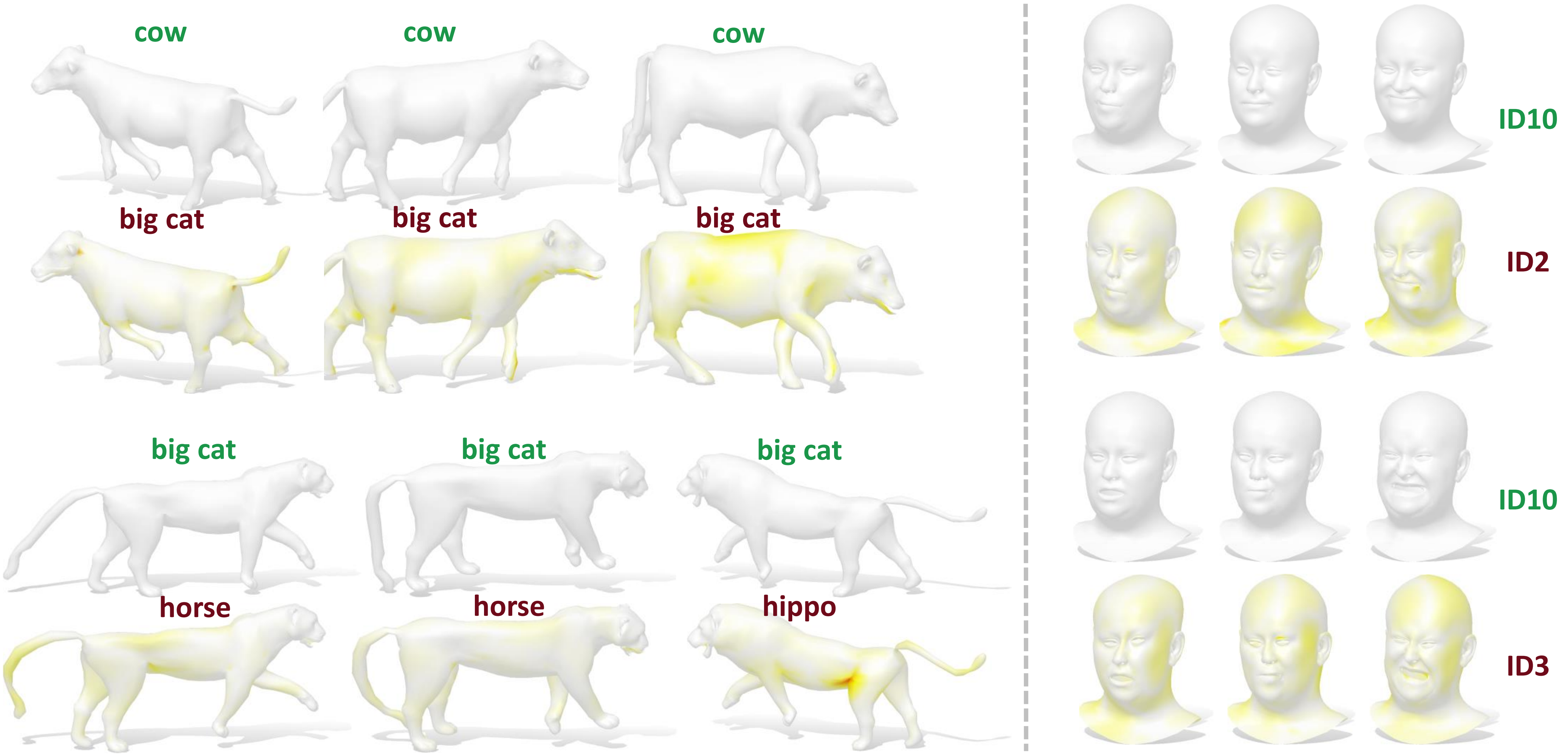
Rampini et al., *Universal Spectral Adversarial Attacks for Deformable Shapes* (CVPR 2021)

Generalization to unseen shapes



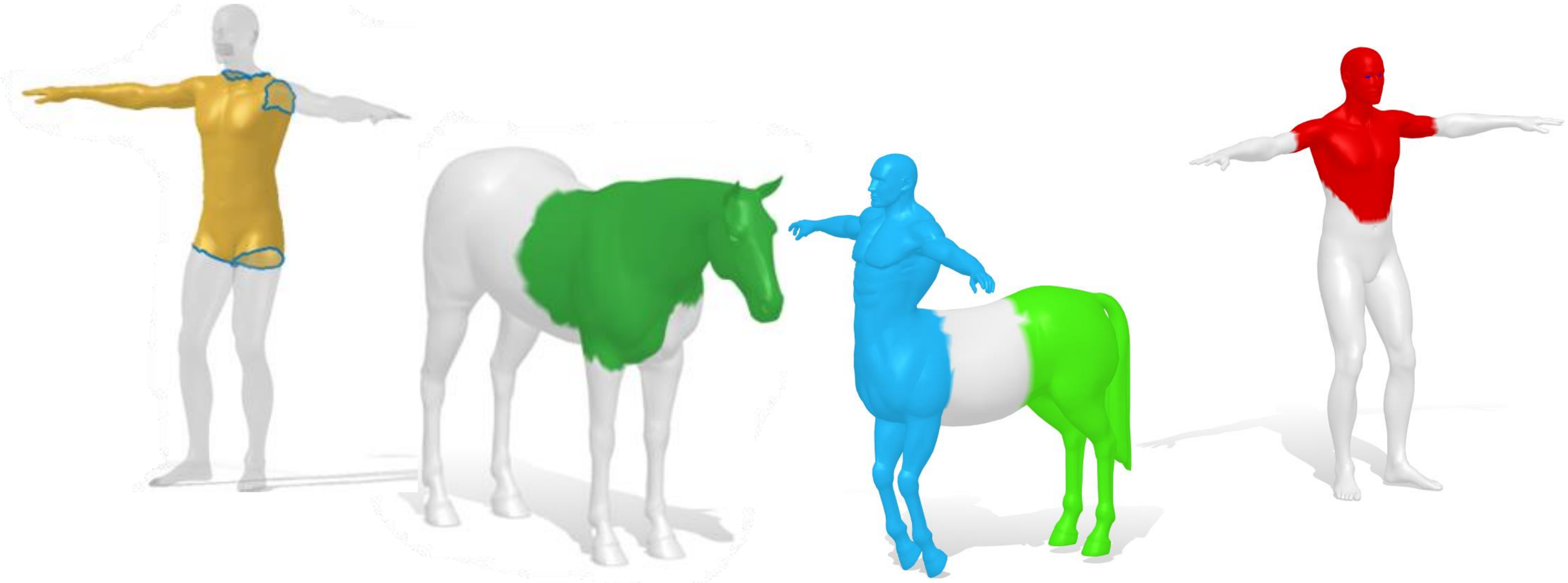
Rampini et al., *Universal Spectral Adversarial Attacks for Deformable Shapes* (CVPR 2021)

Examples

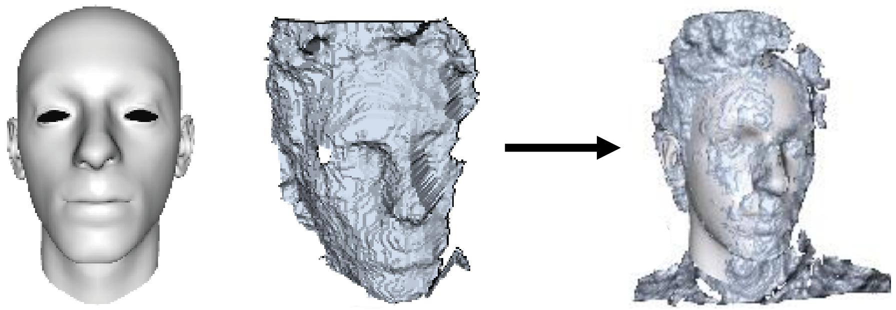

cow
cow
cow
big cat
big cat
big cat
big cat
big cat
big cat
horse
horse
hippo
ID10
ID2
ID10
ID3

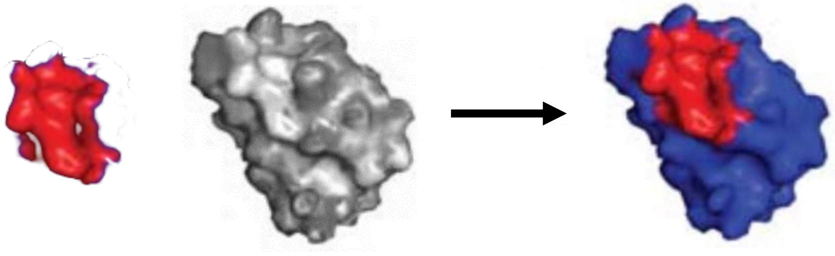
Partial shape localization

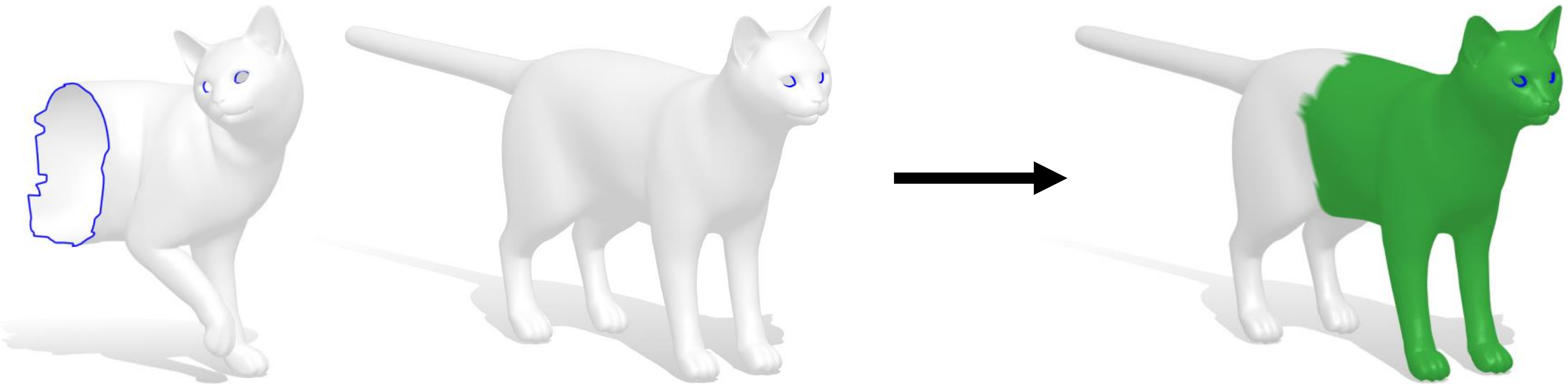
Subregion of a given shape



Motivation

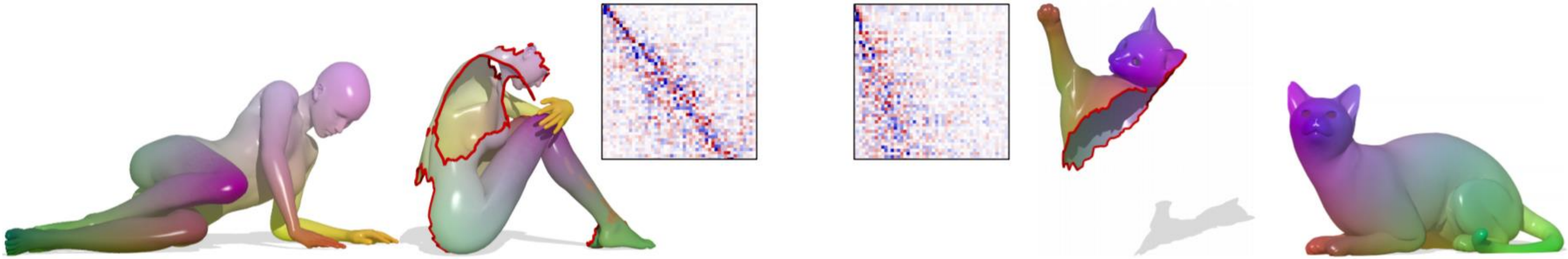

Template
Scan
Registration


Protein A
Protein B
Interface / Non-interface


Partial shape
Full shape
Mask

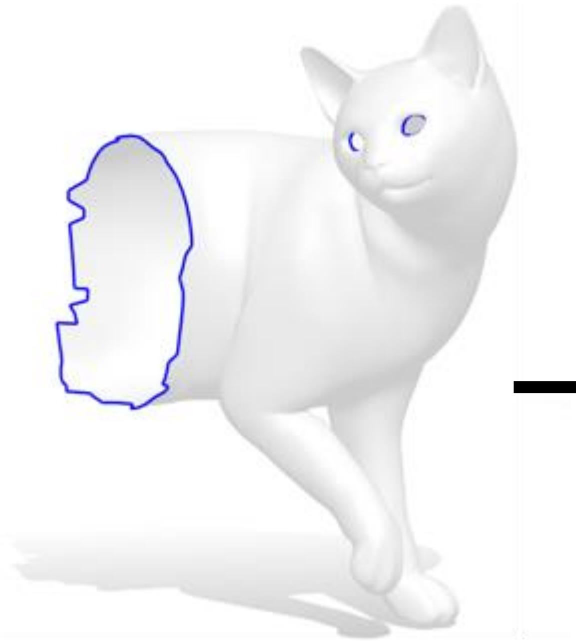
Remark

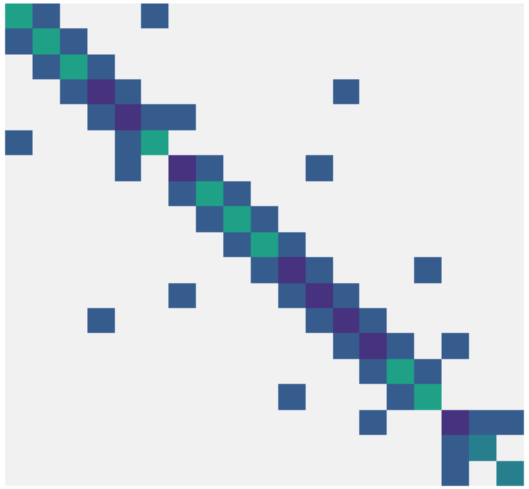
- Spectral quantities can be used to analyze partialities of 3D objects



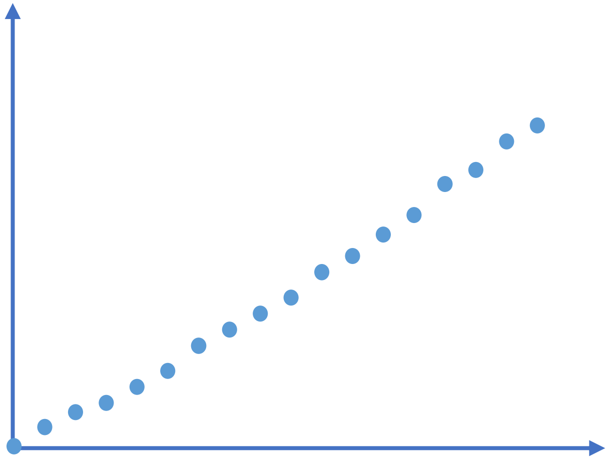
Rodolà et al., *Partial functional correspondence* (2017)

Which operator?



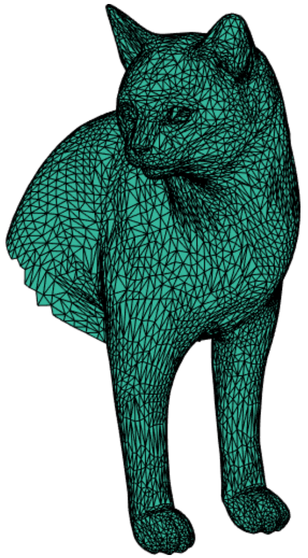


Operator



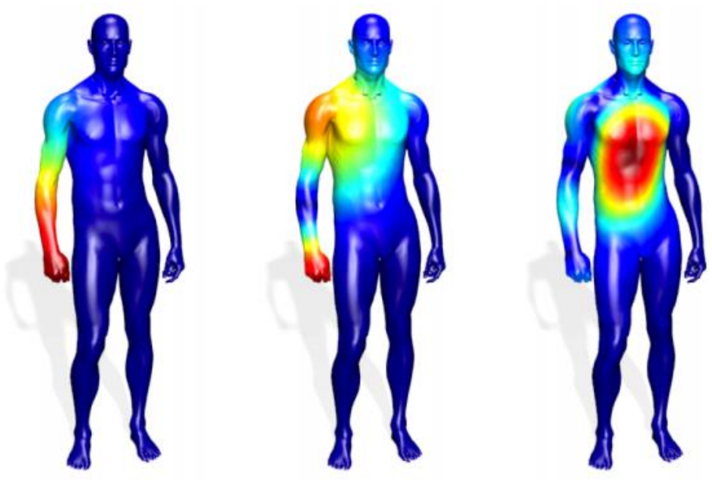
Spectrum

Which operator?



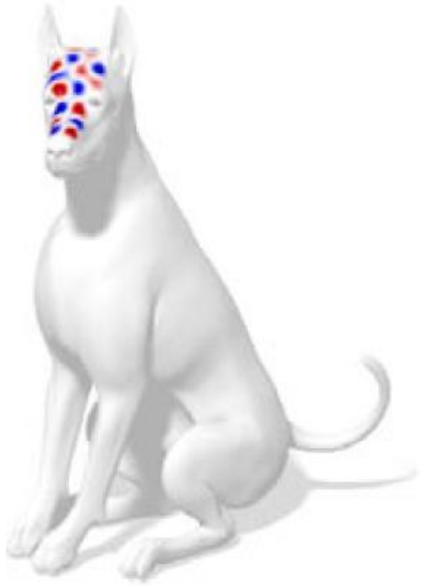
Laplacian of the patch

"Computing Discrete Minimal Surfaces and Their Conjugates", U. Pinkall et al. 1993.



Hamiltonian

"Hamiltonian operator for spectral shape analysis", Y. Choukroun et al. 2018.



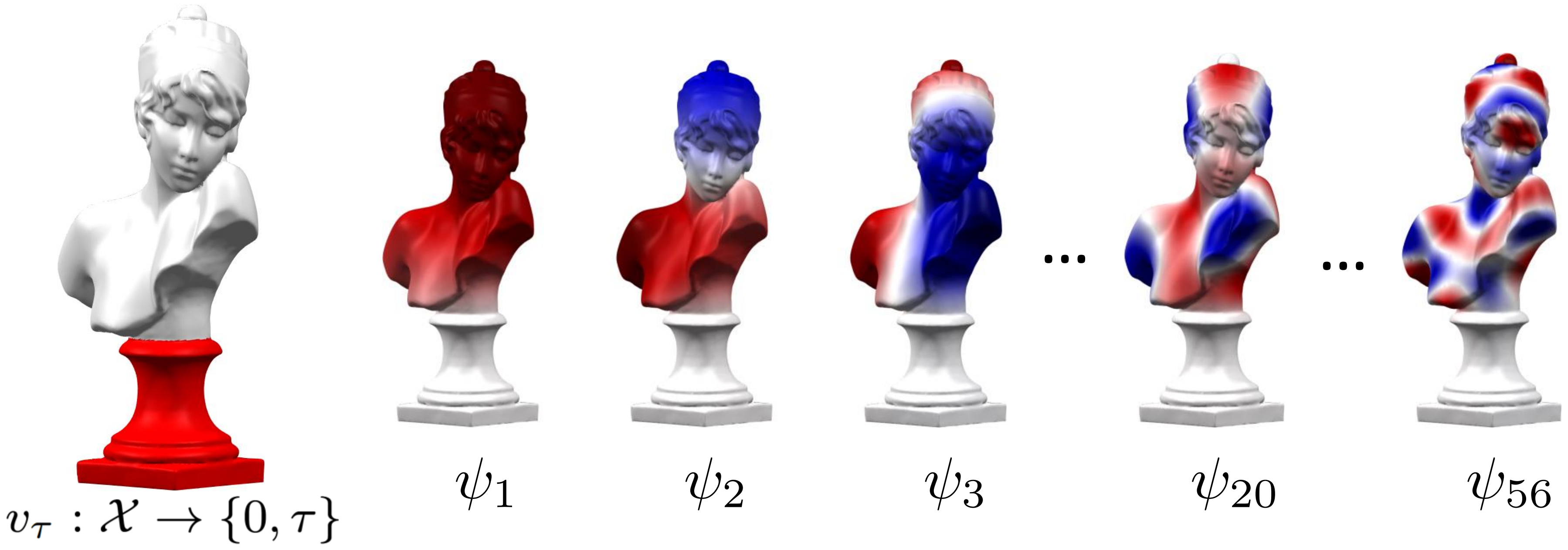
LMH

"Localized Manifold Harmonics for Spectral Shape Analysis", S. Melzi et al. 2018.

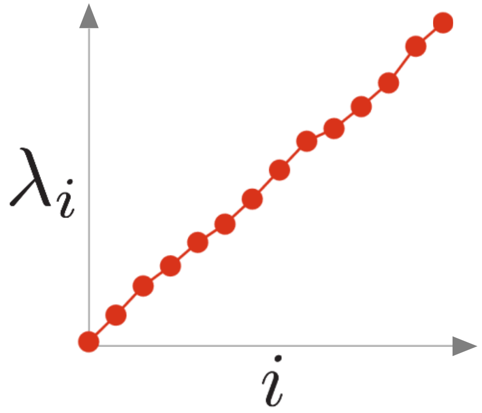
The Hamiltonian operator

$$H\psi_i(x) = \lambda_i \psi_i(x)$$

Step potential

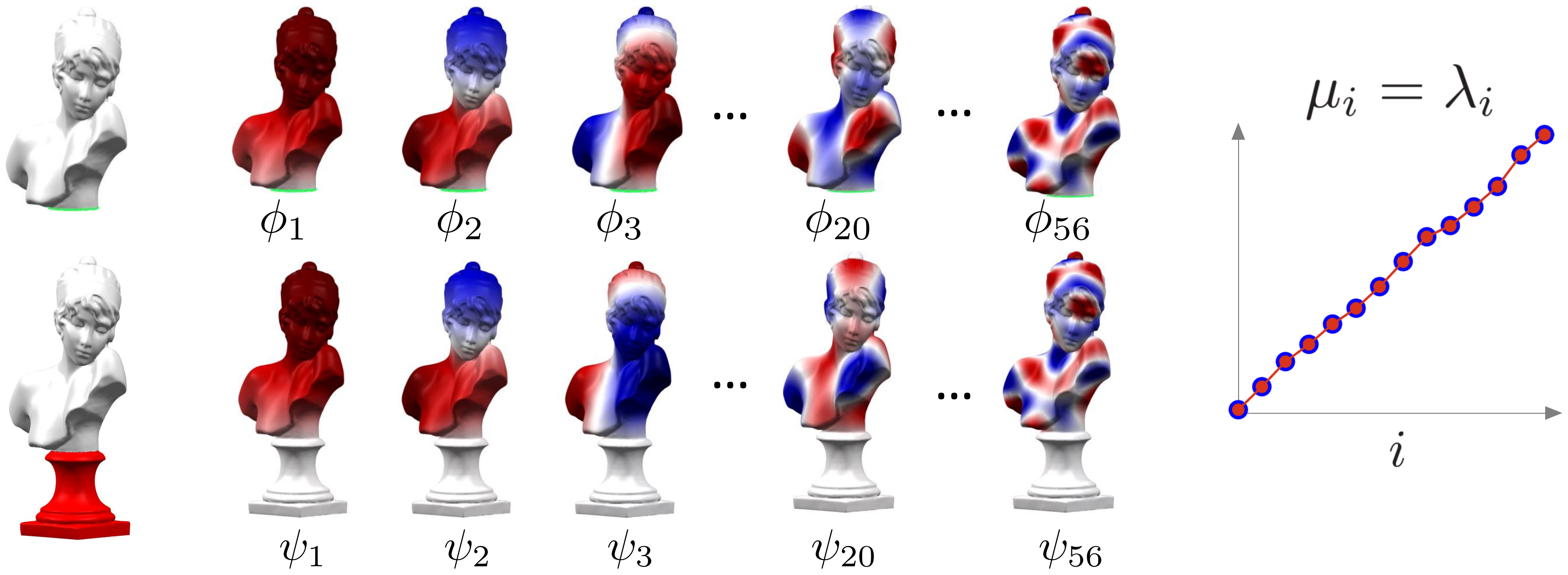


Hamiltonian spectrum



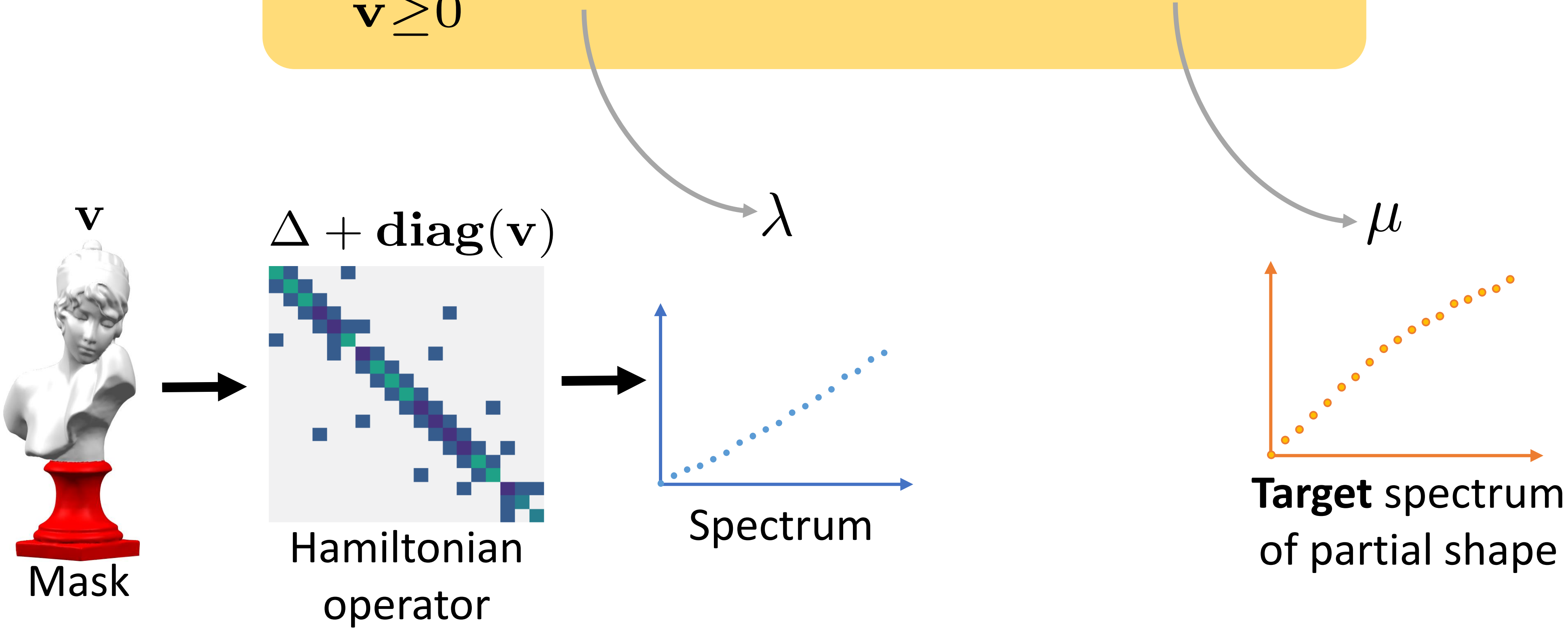
The Hamiltonian operator

Theorem: There exists a step potential for which the Hamiltonian on the full shape and the LBO on the partial shape share the same spectrum:



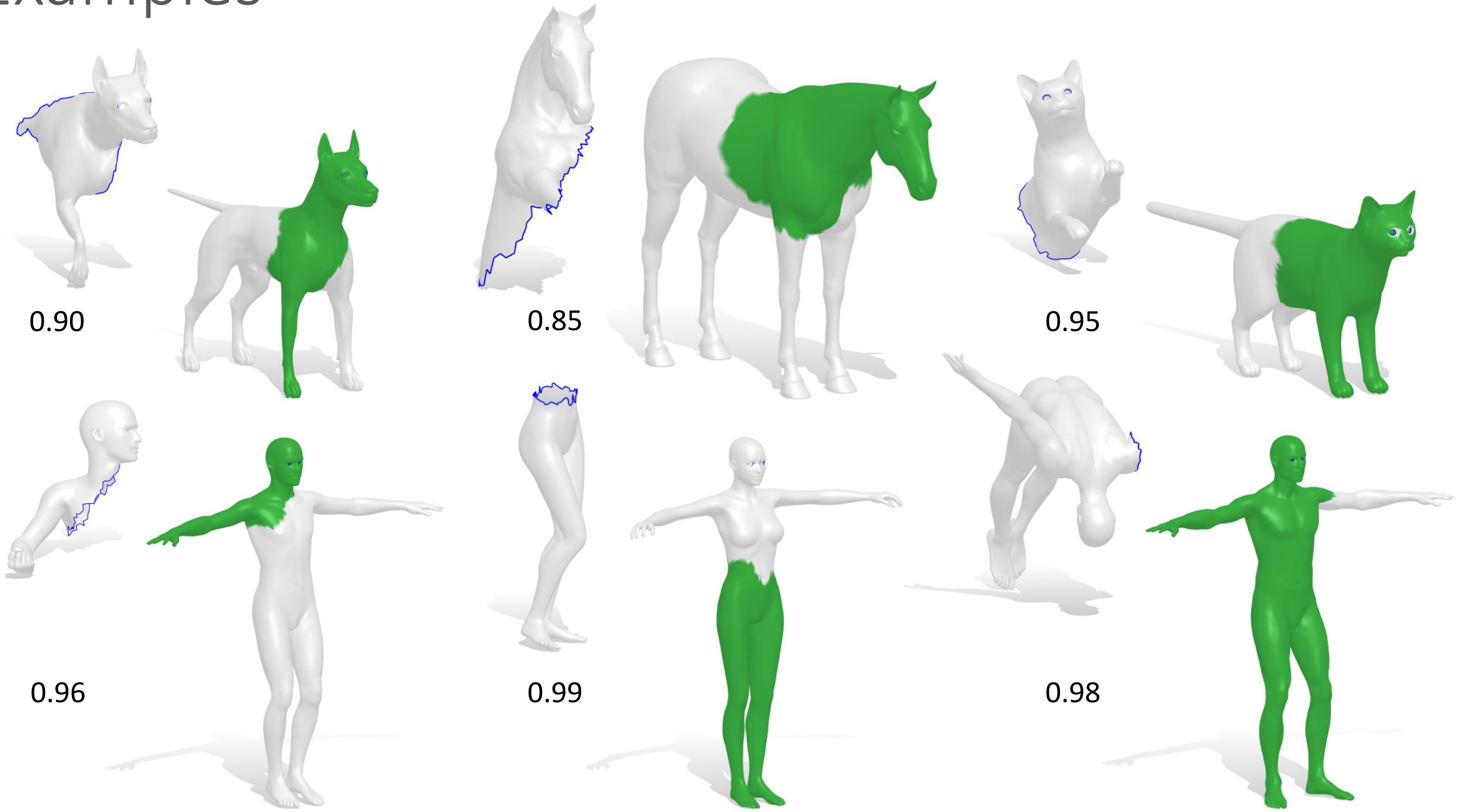
Optimization problem

$$\min_{\mathbf{v}\geq 0} \ \|\lambda(\boldsymbol{\Delta}+\mathbf{diag}(\mathbf{v}))-\mu\|_w^2$$



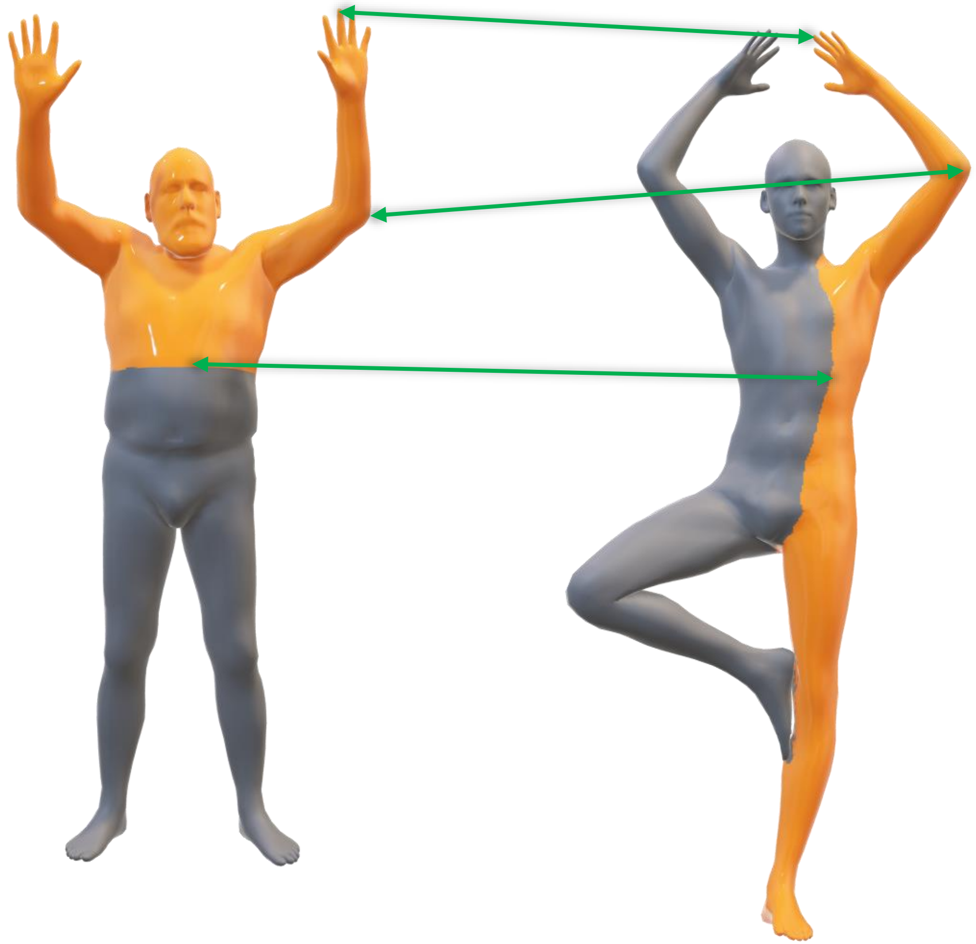
Rampini et al., *Correspondence-Free Region Localization for Partial Shape Similarity via Hamiltonian Spectrum Alignment* (3DV 2019)

Examples

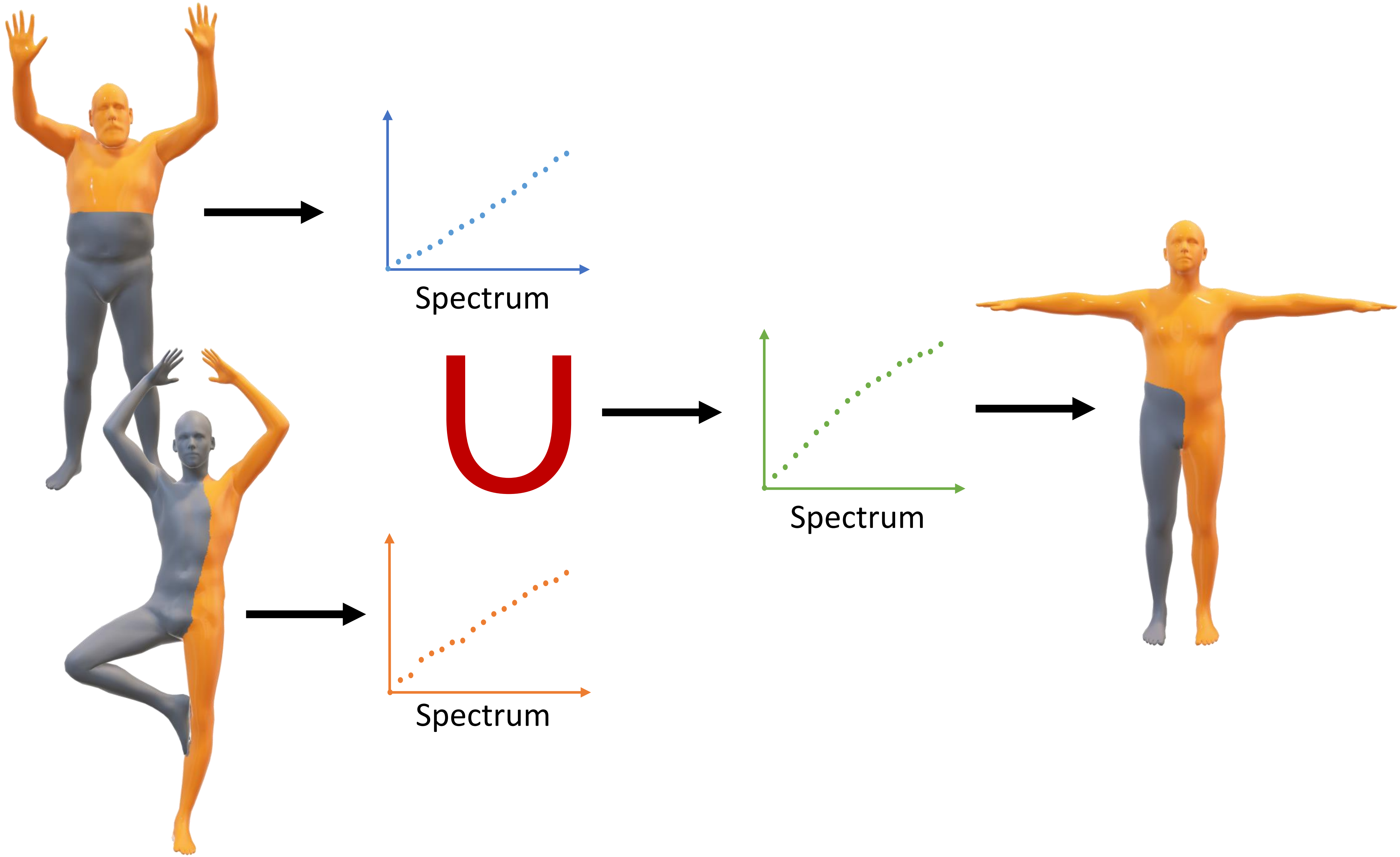


Set operations

Typical pipeline



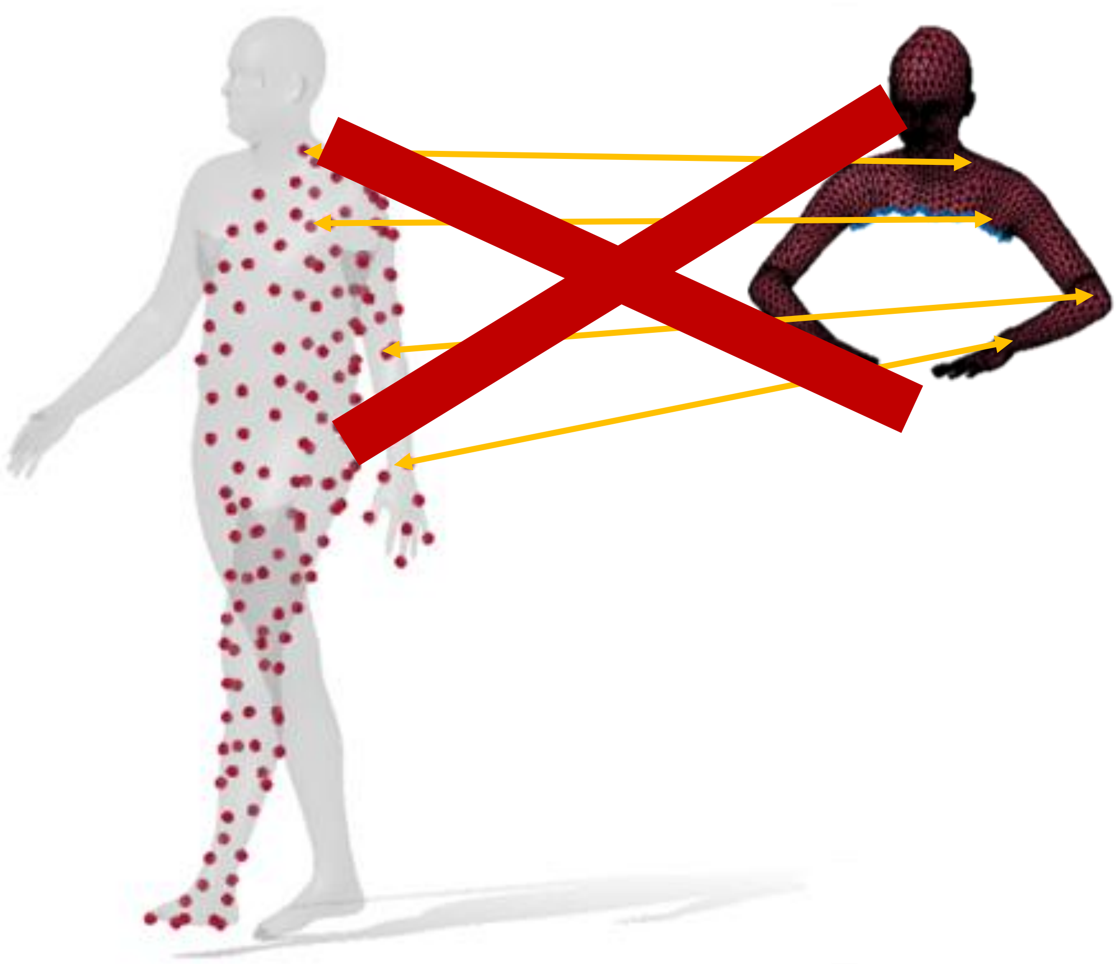
1. Find partial correspondence
2. Extract non-rigid transformation
3. Merge partial views into a consistent discretization



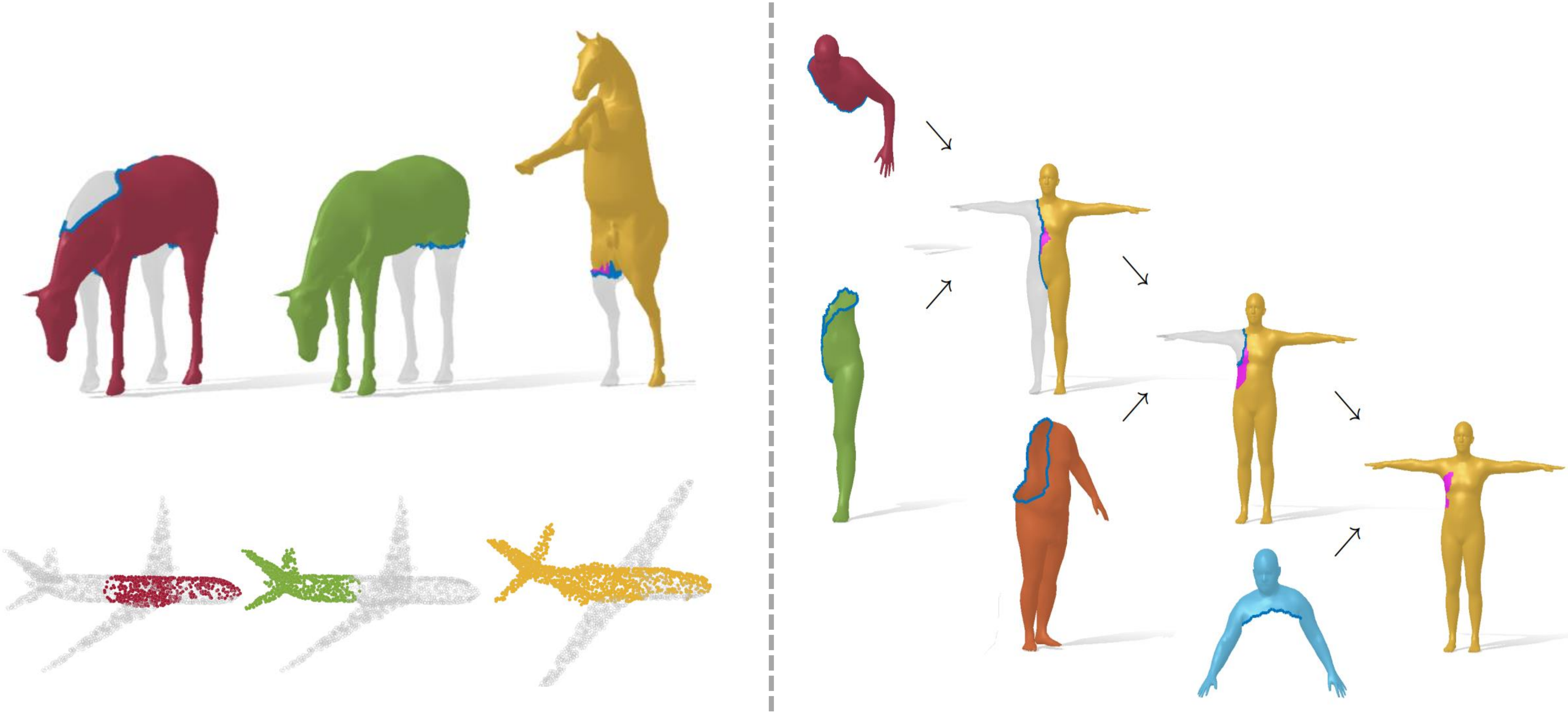
Moschella et al., *Spectral Unions of Partial Deformable 3D Shapes* (EUROGRAPHICS 2022)

The spectrum is the right tool

- Invariant to isometries
- Invariant to different representations
- Does not require a correspondence

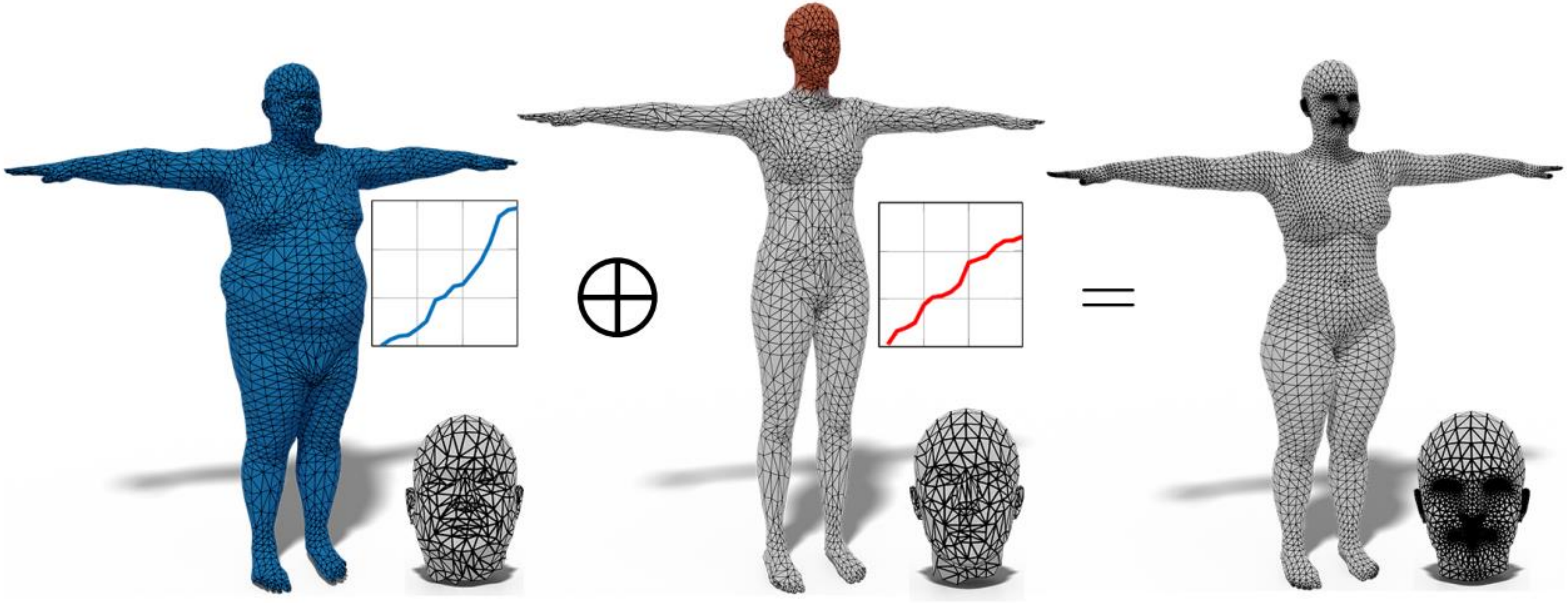


Results



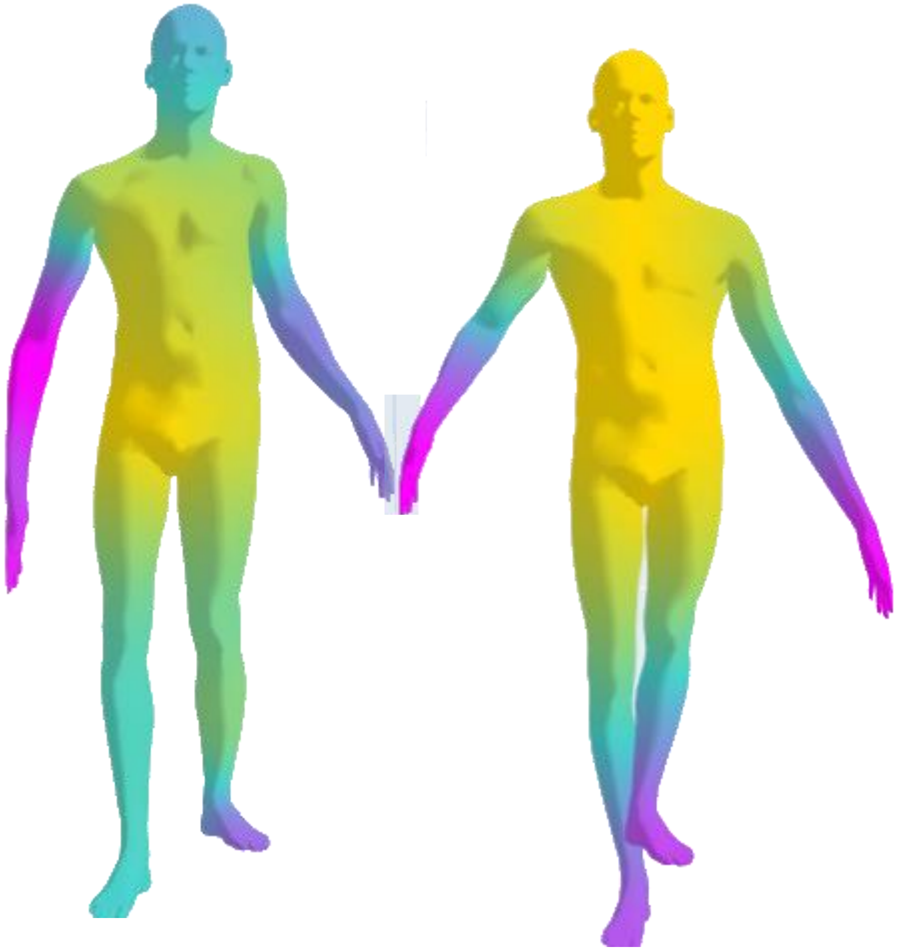
Moschella et al., *Spectral Unions of Partial Deformable 3D Shapes* (EUROGRAPHICS 2022)

Shape generation

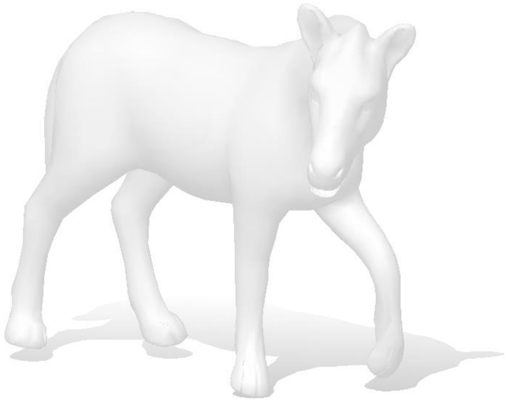


Pegoraro et al., *Learning to generate shape from global-local spectra* (2021)

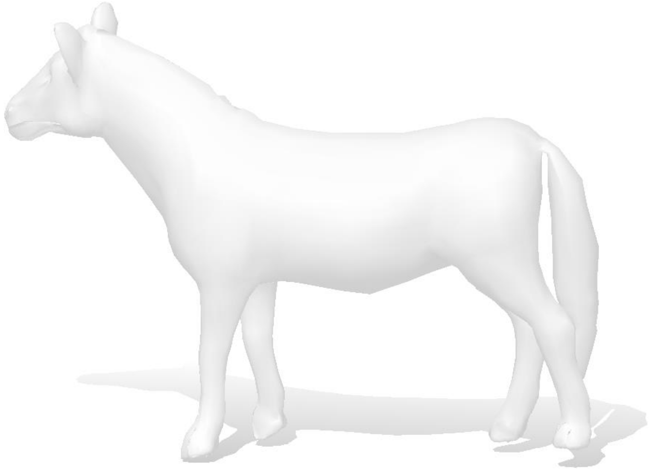
Hearing shapes with PyTorch



https://github.com/riccardomarin/EG22_Tutorial_Spectral_Geometry



Thank you!



Special thanks to S. Melzi, E. Postolache and L. Moschella for some of these slides