Mathématiques - Informatique

ALICE

Geometry and Light

Bruno Lévy
ALICE Géométrie & Lumière
CENTRE INRIA Nancy Grand-Est

OVERVIEW

Part. 1. Research axes, Evolutions, Applications

Part. 2. From Graphics to Fabrication

Part. 3. From Geometry Processing to Applied Math.

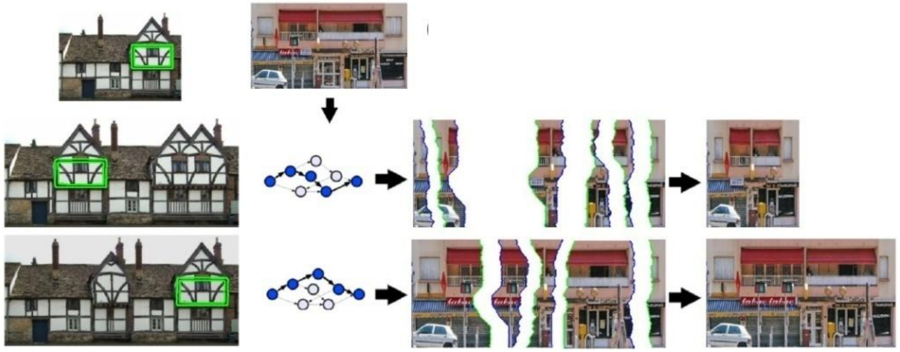
Part. 4. Future Works

1

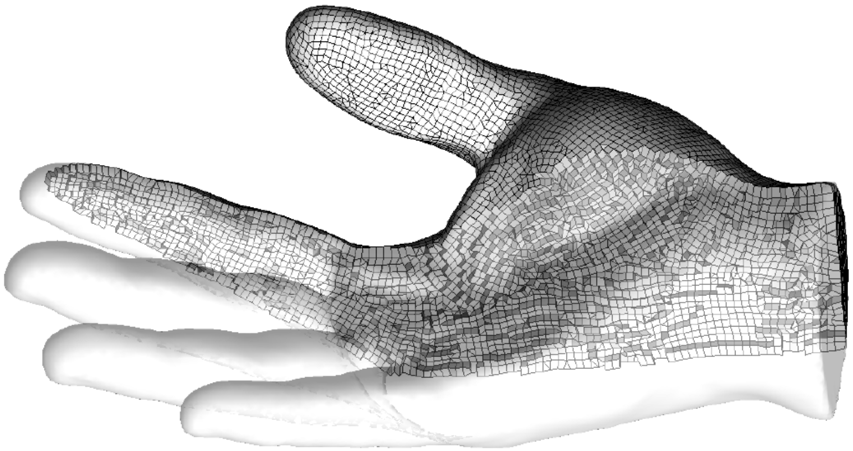
Research Axes
Evolution
Application Domains

Part. 1. Research Axes 2006 - 2012

Computer Graphics / Automatic Content Creation



Geometry Processing



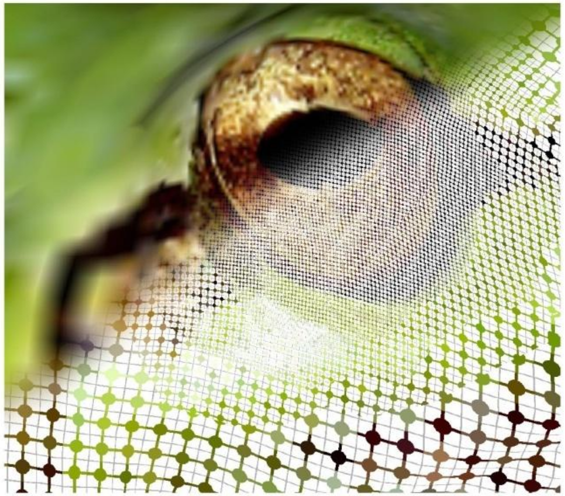
$$
\begin{aligned}
F_{L_p}^{T} &= \int_{T} \|\mathbf{M}_T(\mathbf{y}-\mathbf{x}_0)\|_p^p \, d\mathbf{y} \\
&= \frac{|T|}{\binom{n+p}{n}} \sum_{\alpha+\beta+\gamma=p} \overline{\mathbf{U}_1^{\alpha} * \mathbf{U}_2^{\beta} * \mathbf{U}_3^{\gamma}}
\end{aligned}
$$

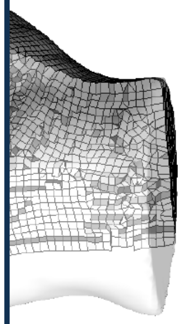
$$
\text{where:} \quad
\begin{cases}
\mathbf{U_i} = \mathbf{M}_T(\mathbf{C}_i - \mathbf{x}_0) \\
\mathbf{V}_1 * \mathbf{V}_2 = [x_1 x_2, y_1 y_2, z_1 z_2]^t \\
\mathbf{V}^{\alpha} = \mathbf{V} * \mathbf{V} * \ldots * \mathbf{V} (\alpha \text{ times}) \\
\overline{\mathbf{V}} = x + y + z
\end{cases}
$$

Part. 1. Research Axes 2006 - 2012

Computer Graphics / Automatic Content Creation

Geometry Processing





Texture Mapping

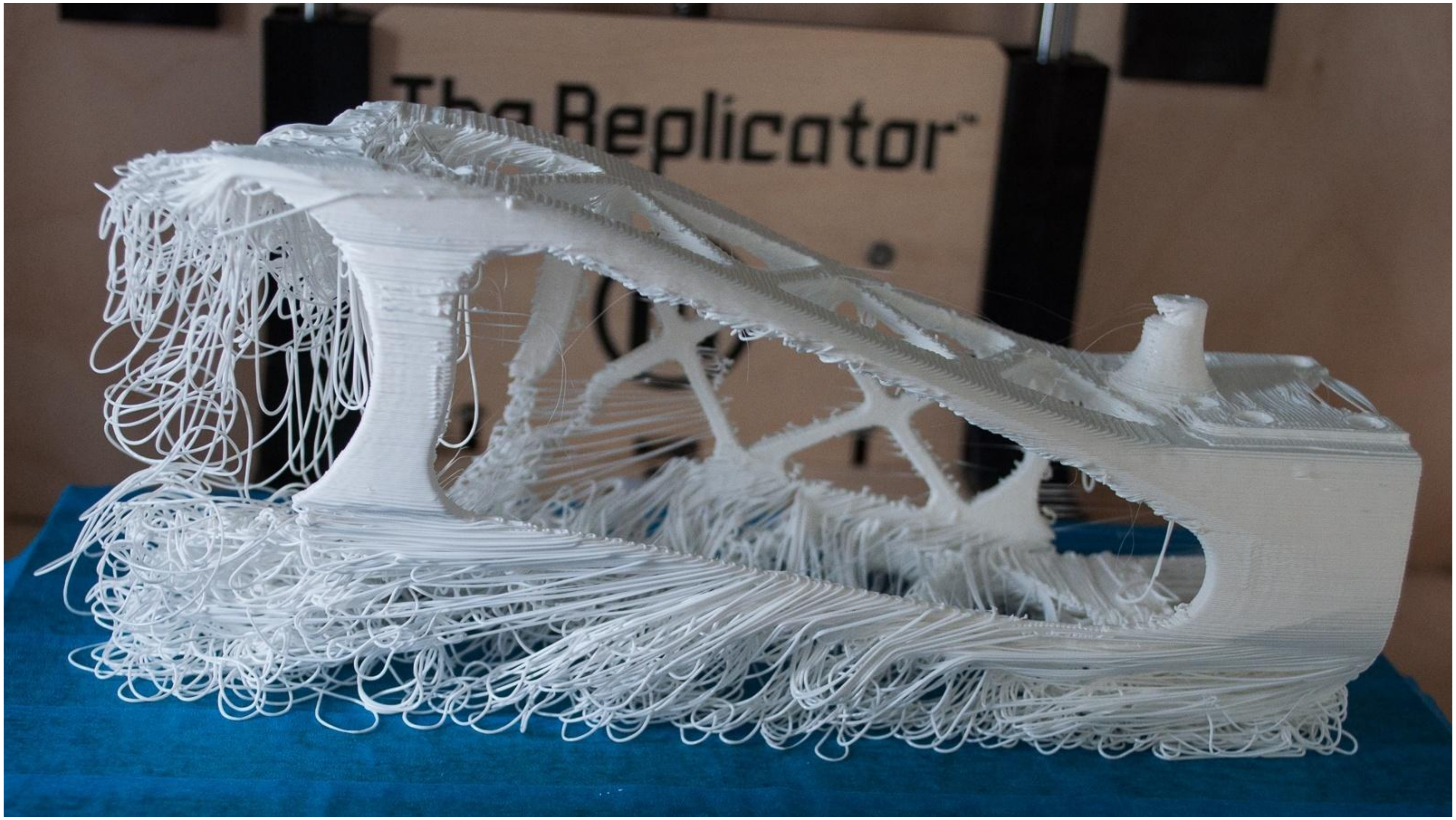
$$\text{where: } \begin{cases} \mathbf{U_i} & = \mathbf{M}_T(\mathbf{C}_i - \mathbf{x}_0) \\ \mathbf{V}_1 * \mathbf{V}_2 & = [x_1x_2, y_1y_2, z_1z_2]^t \\ \mathbf{V}^\alpha & = \mathbf{V} * \mathbf{V} * \ldots * \mathbf{V}(\alpha \text{ times}) \\ \overline{\mathbf{V}} & = x + y + z \end{cases}$$

Part. 1. Research Axes 2013 - 2016

Into reality

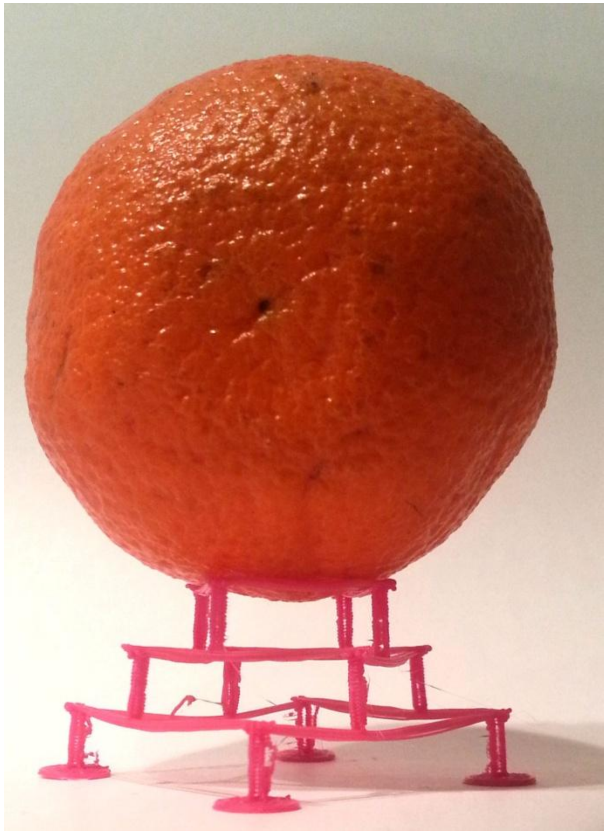
Poppy – Inria project Flowers

Part. 1. Research Axes 2013 - 2016



Print your own “Poppy Robot” at home … not that easy !!!

Part. 1. Research Axes 2013 - 2016



Print a “scaffold” with the object

Part. 1. Research Axes 2013 - 2016

Into reality

Poppy – Inria project Flowers

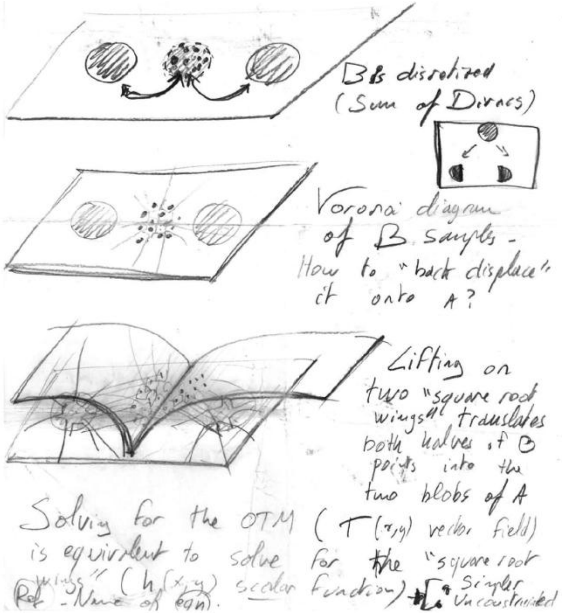
Make it easy for everybody (“it” = object modeling, 3d printing …)

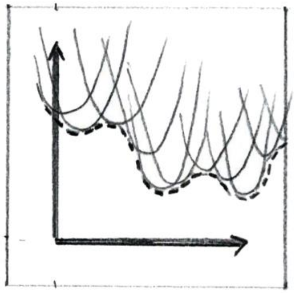
Part. 1. Research Axes 2013 - 2016

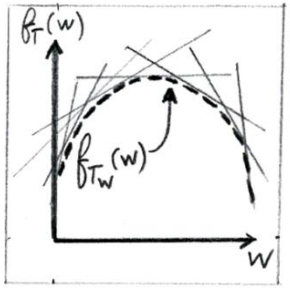
Into reality

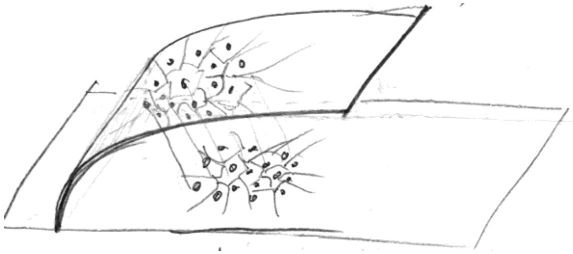
Poppy – Inria project Flowers

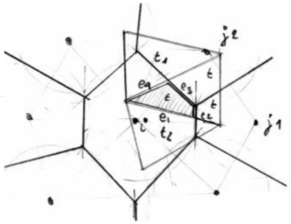
Into abstraction









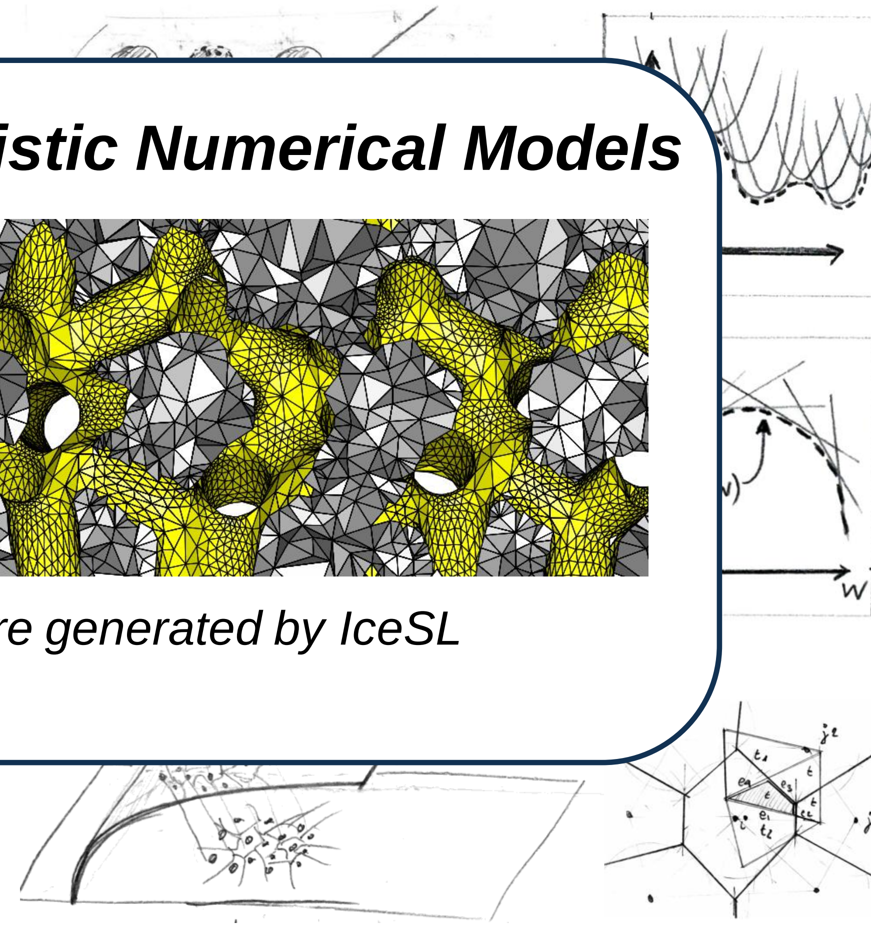


Part. 1. Research Axes 2013 - 2016

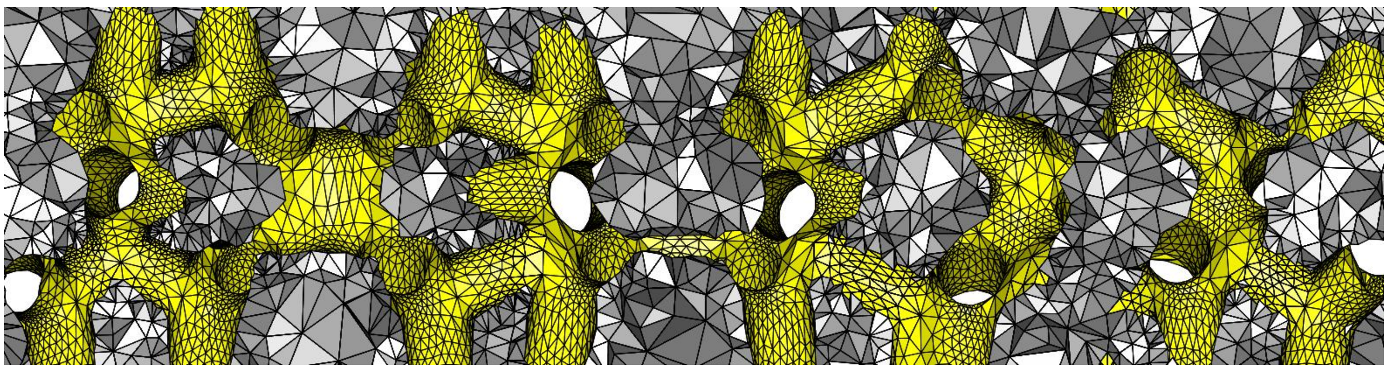
Into reality

Poppy – Inria project Flowers

Into abstraction



Simulating Reality – Realistic Numerical Models



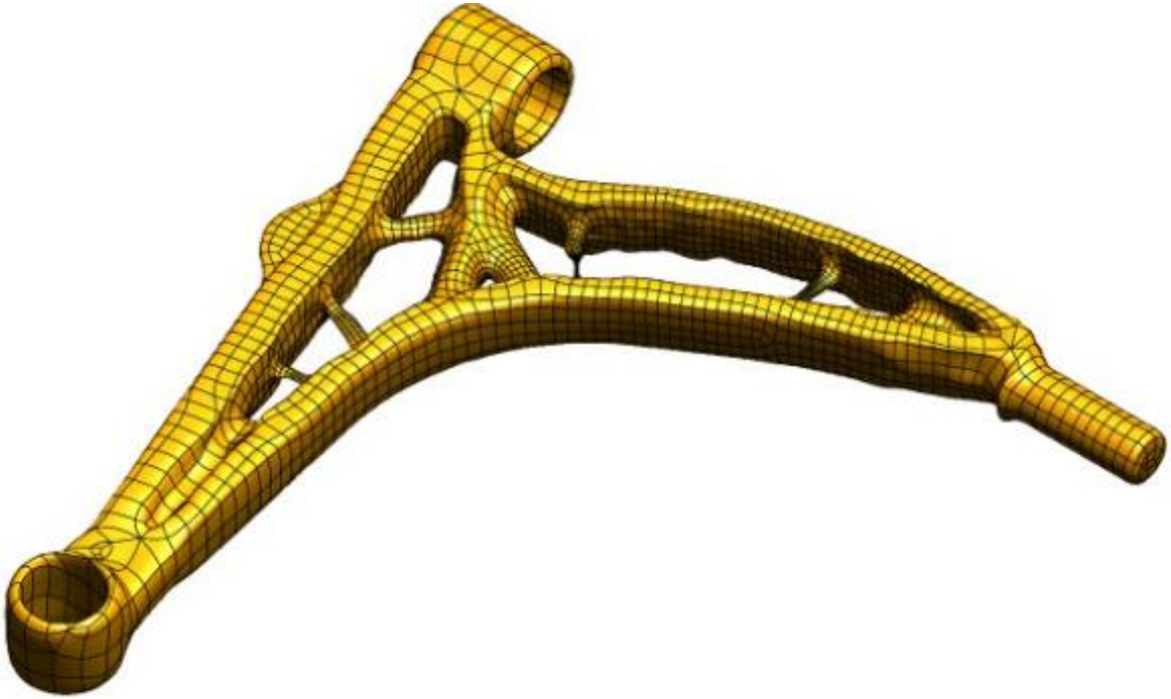
3D mesh of a microstructure generated by IceSL

Part. 1. Application Domains

Inserting Shape Optimization into the loop

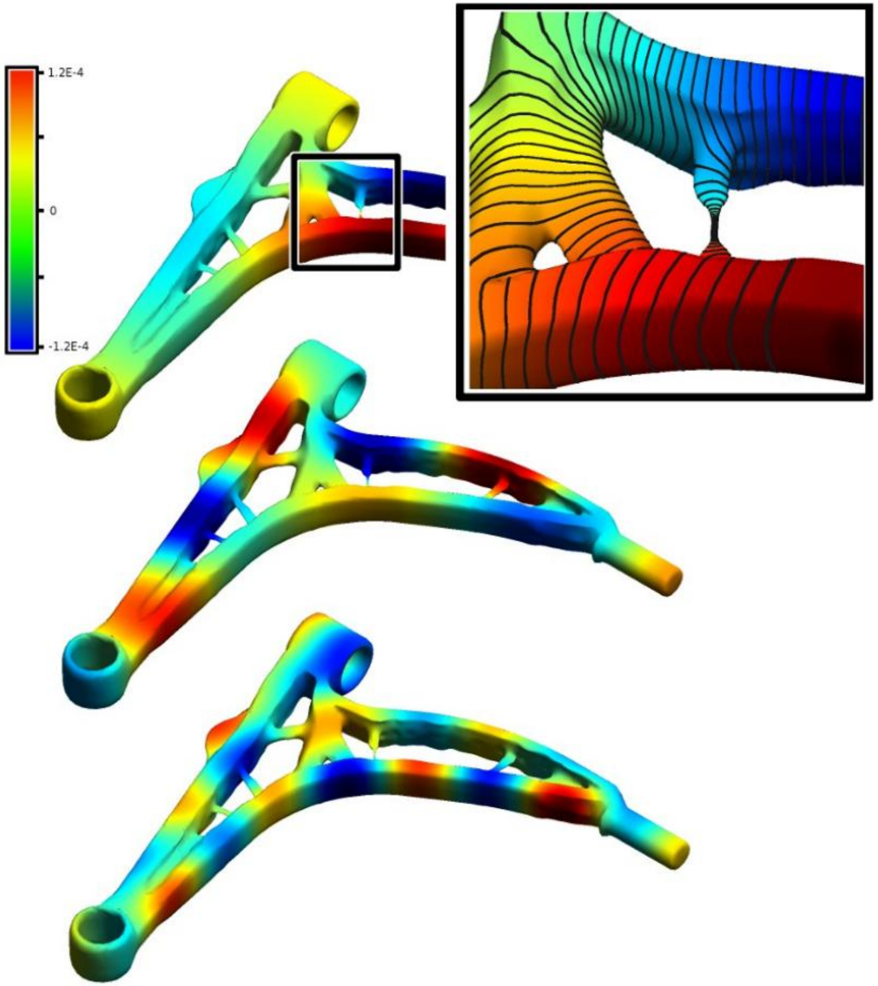
Part. 1. Application Domains

Inserting Shape Optimization into the loop



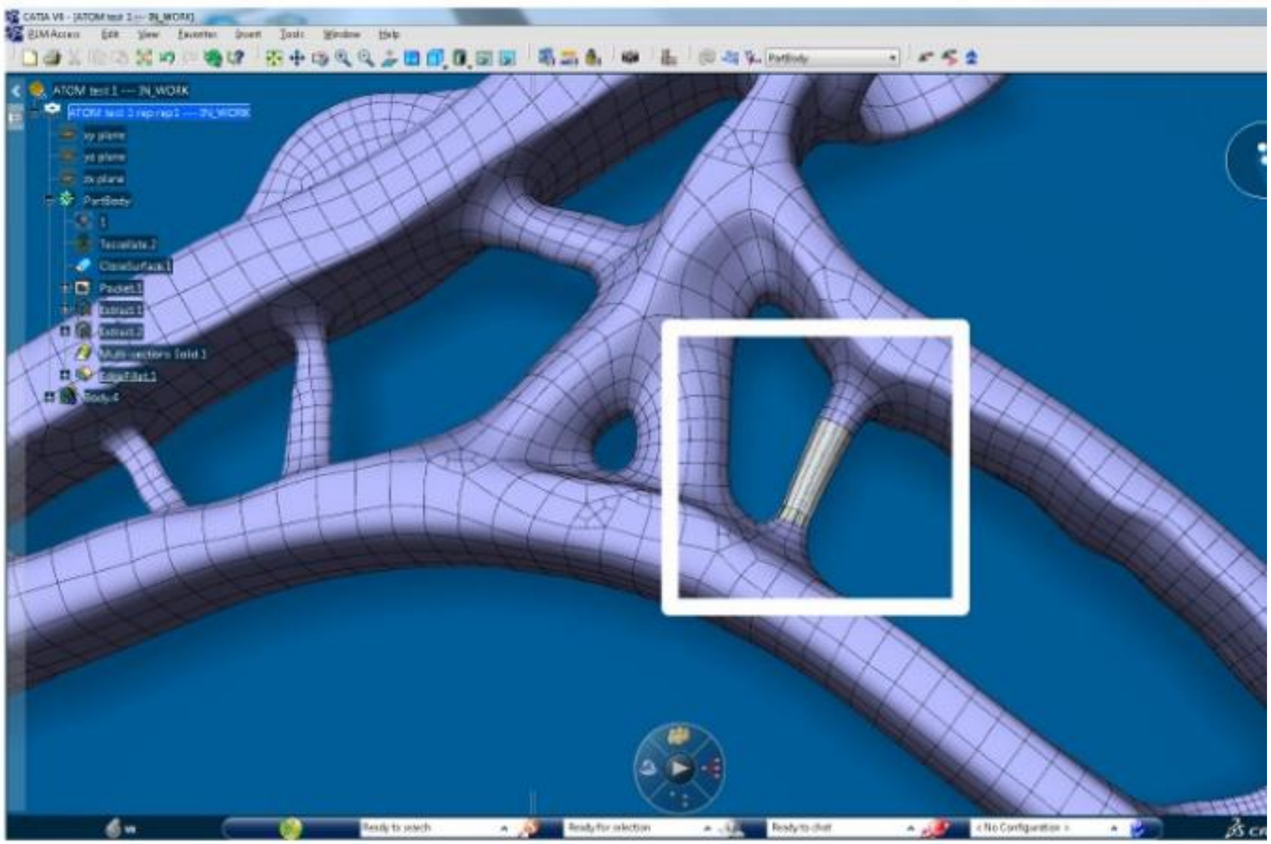
Part. 1. Application Domains

Inserting Shape Optimization into the loop



Part. 1. Application Domains

Inserting Shape Optimization into the loop

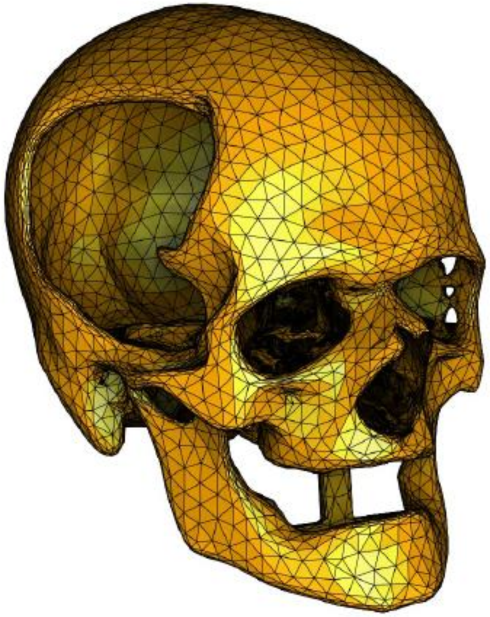


Part. 1. Application Domains

Inserting Shape Optimization into the loop

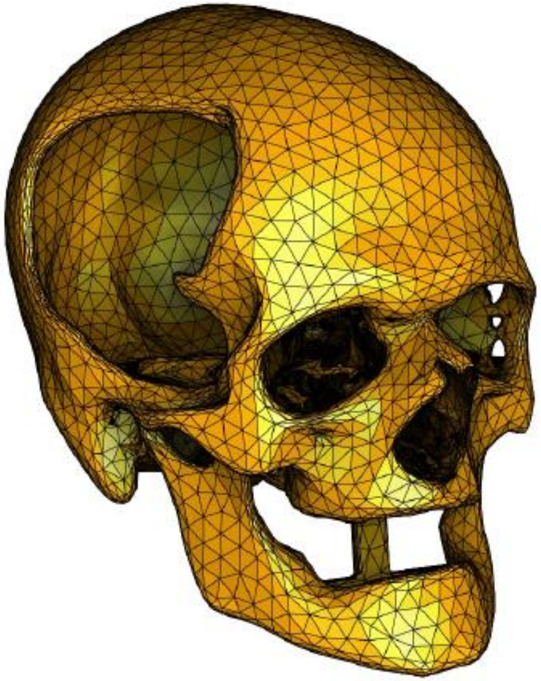
Part. 1. Application Domains

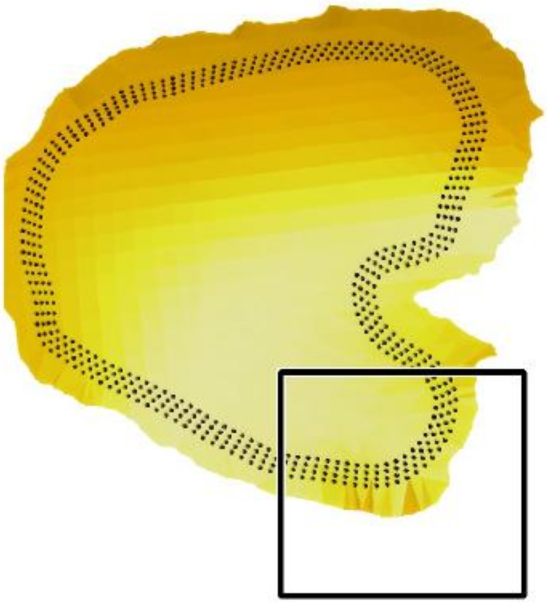
Reparative Surgery



Part. 1. Application Domains

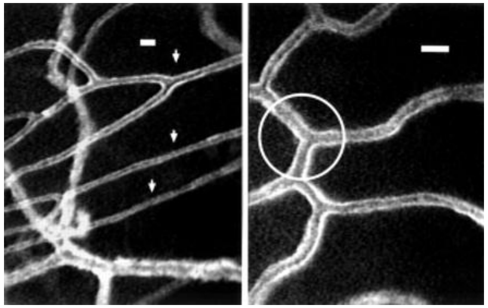
Reparative Surgery

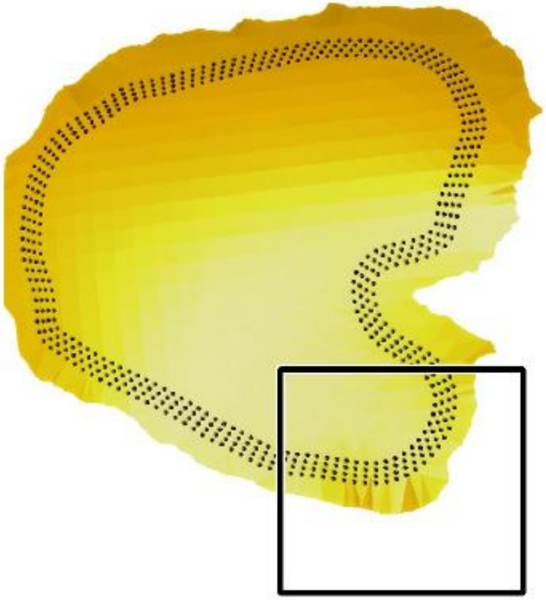


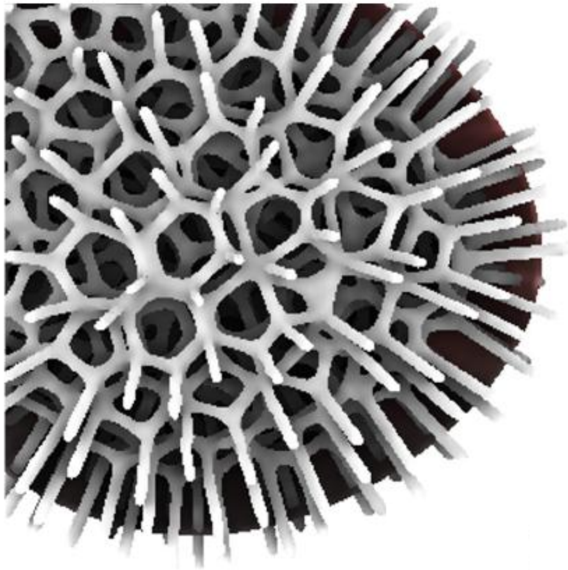


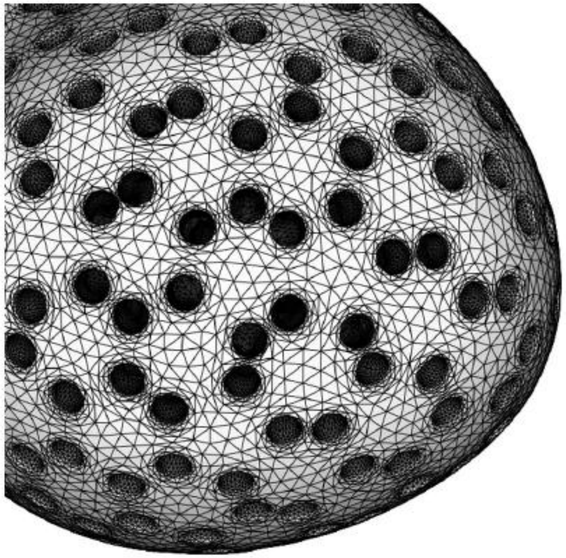
Part. 1. Application Domains

Reparative Surgery



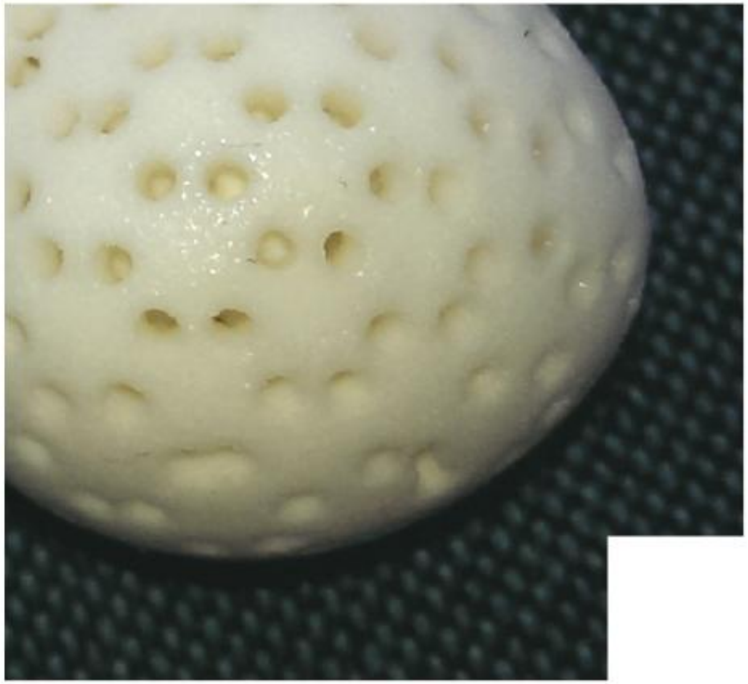






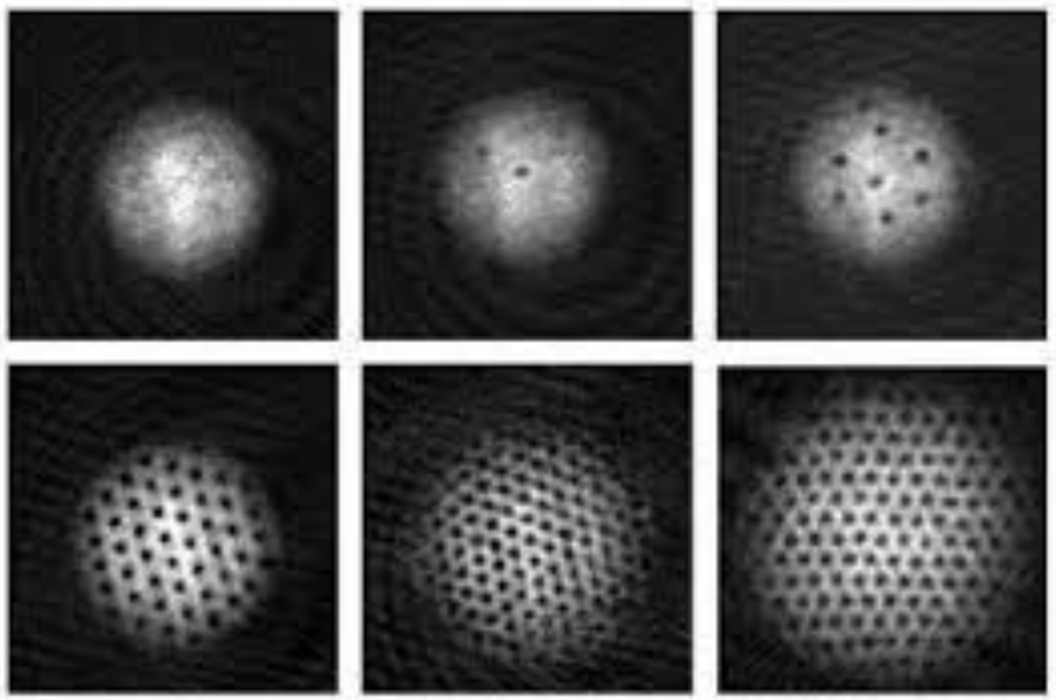
Part. 1. Application Domains

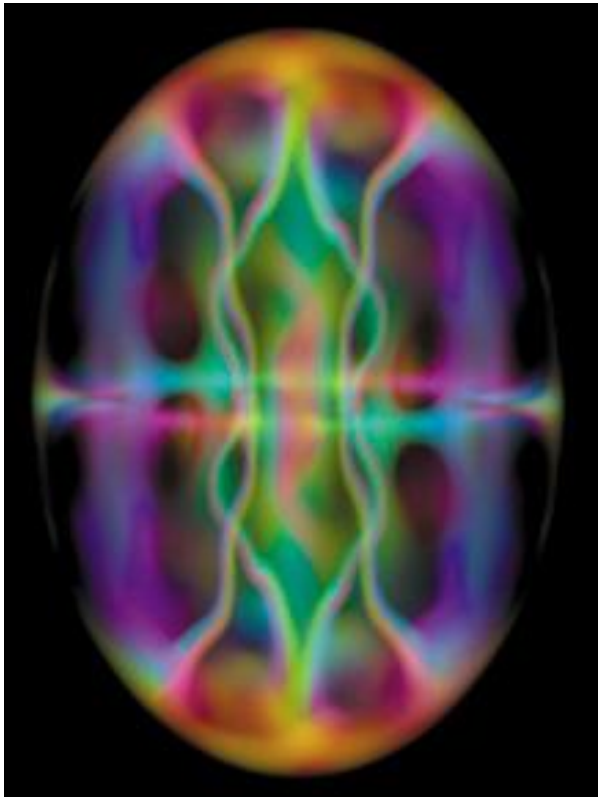
Reparative Surgery



Part. 1. Application Domains

Computational Physics





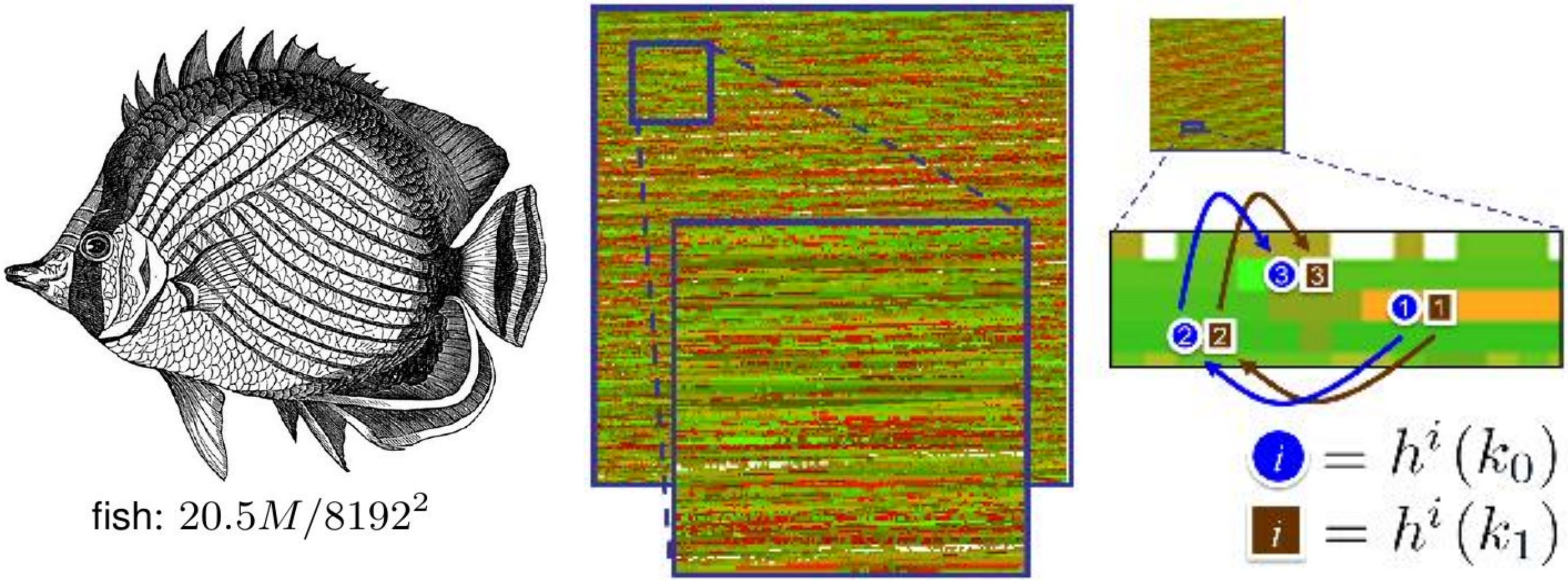
Bose-Einstein Condensate

ANR BECASIM – cooperation with physicists and mathematicians

2

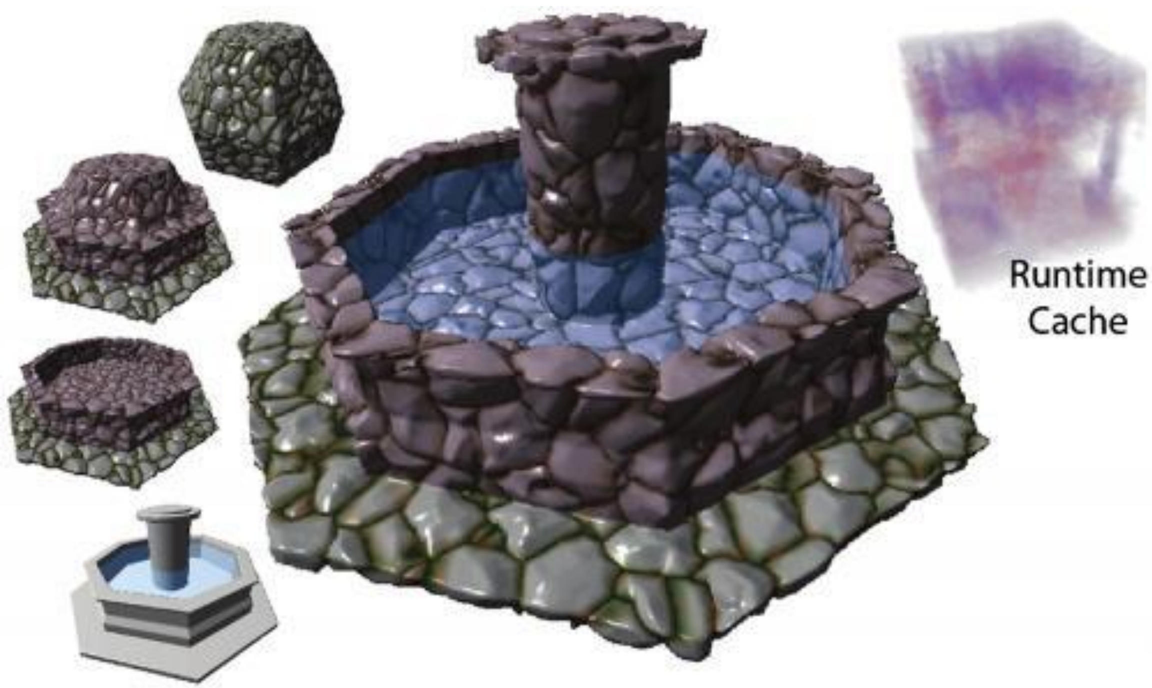
From Graphics to Fabrication

Part. 2 From Graphics to Fabrication



Coherent Parallel Hashing
Garcia, Lefebvre, Hornus, Lasram
SIGGRAPH Asia 2011

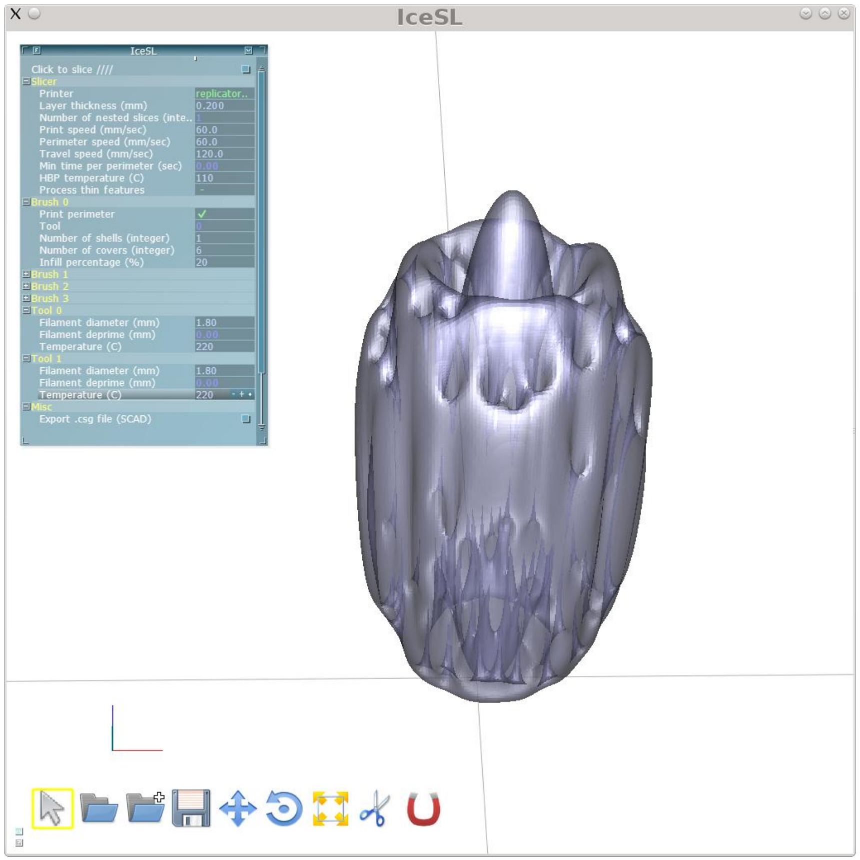
Part. 2 From Graphics to Fabrication

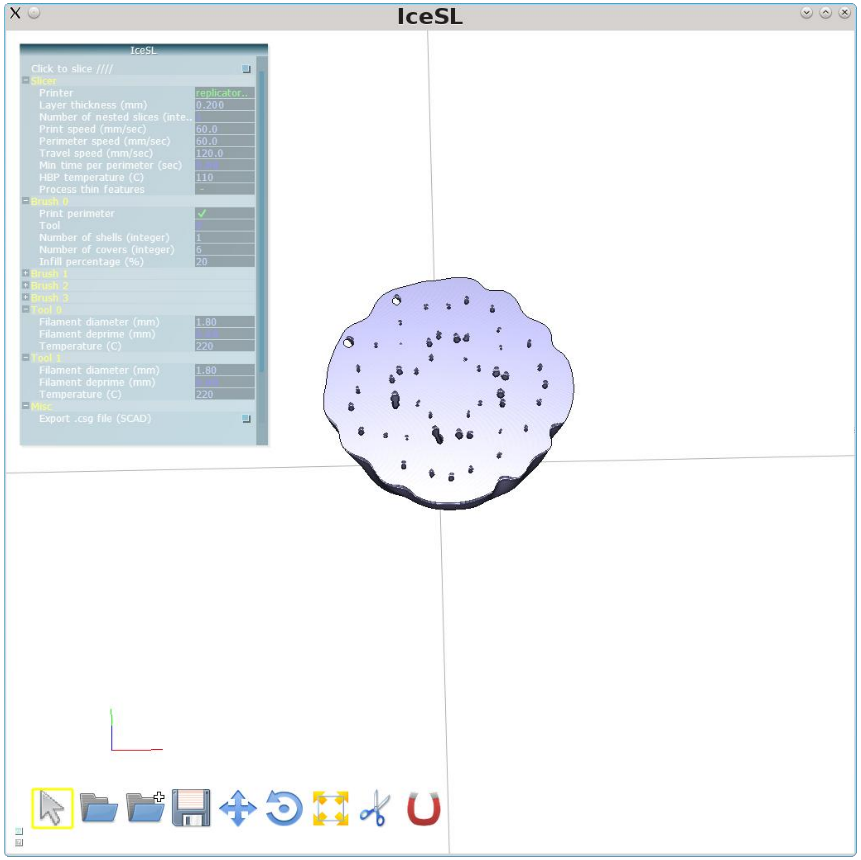


A runtime cache for interactive procedural modeling
Reiner, Lefebvre, Diener, Garcia, Jobard, Dachsbacher
SMI 2012

Part. 2 From Graphics to Fabrication

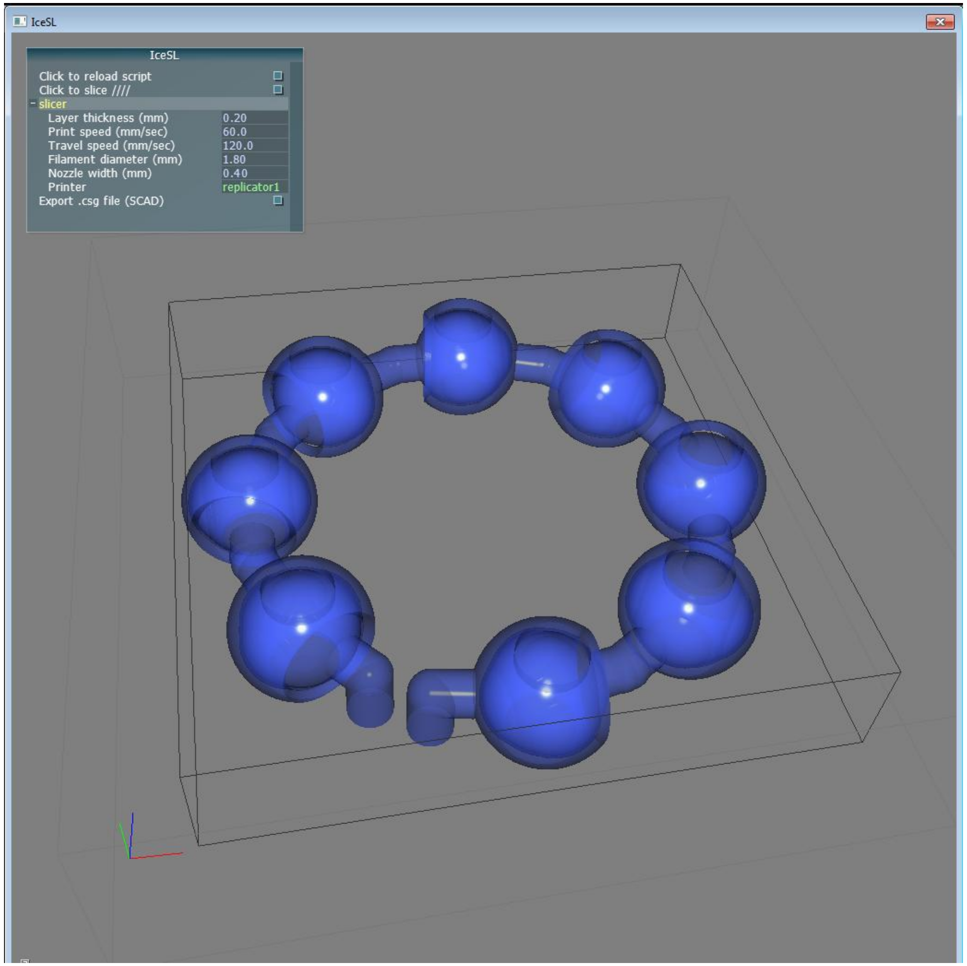
Visualization of Bose-Einstein condensates with IceSL



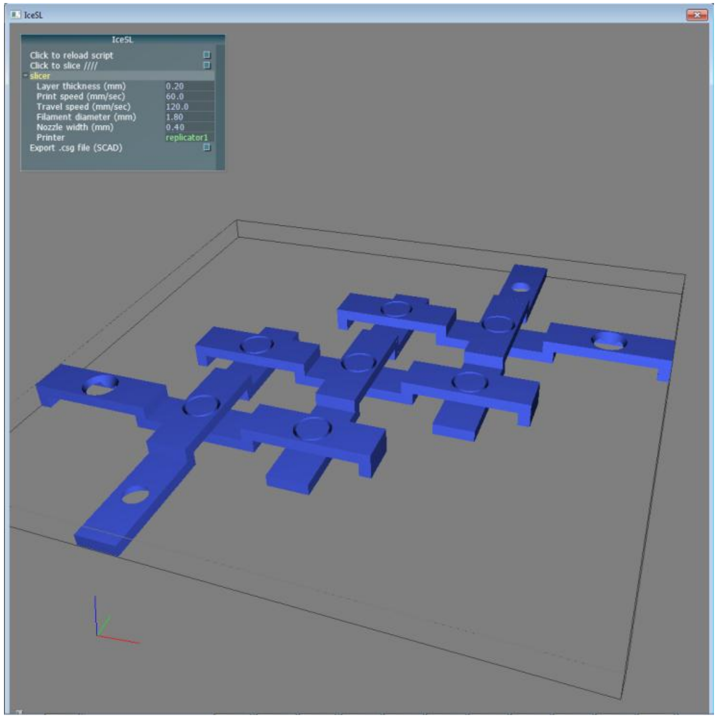


ANR BECASIM – cooperation with physicists and mathematicians

Part. 2 From Graphics to Fabrication



Part. 2 From Graphics to Fabrication

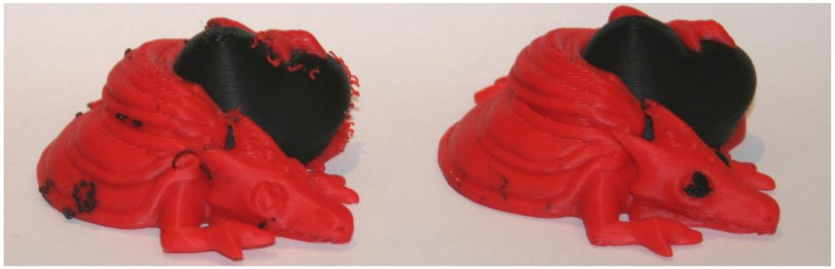


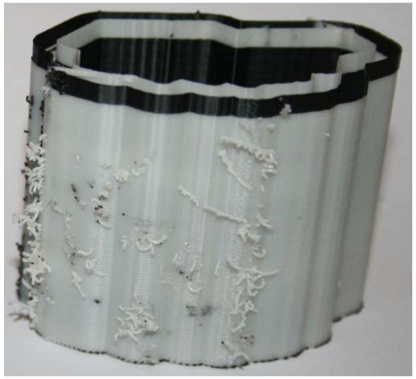
Part. 2 From Graphics to Fabrication

Make it stand, Prevost, Whiting, Lefebvre, Sorkine, SIGGRAPH 2012

Part. 2 From Graphics to Fabrication

Make it stand, Prevost, Whiting, Lefebvre, Sorkine, SIGGRAPH 2012

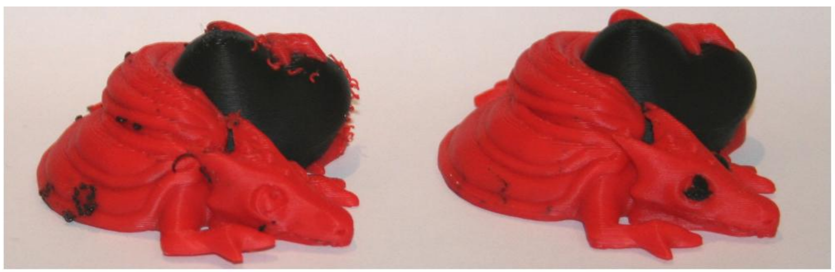


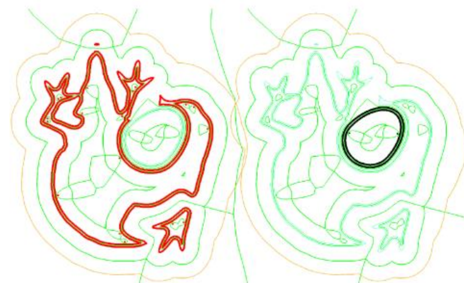


Clean Color, Hergel, Lefebvre, Eurographics 2014

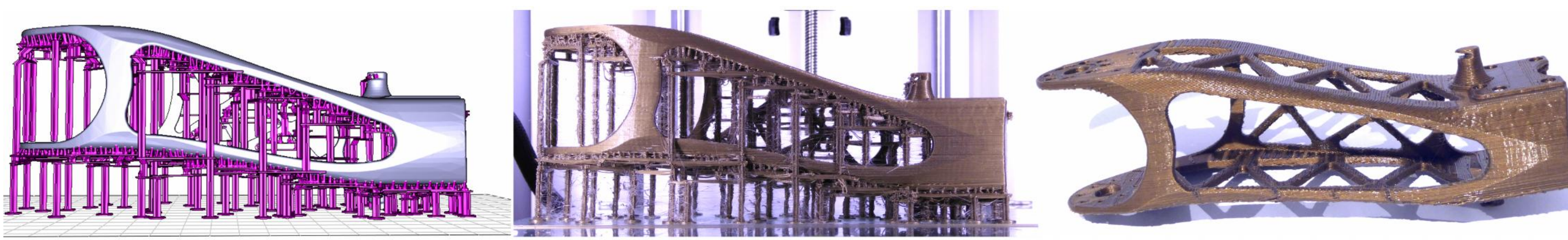
Part. 2 From Graphics to Fabrication

Make it stand, Prevost, Whiting, Lefebvre, Sorkine, SIGGRAPH 2012





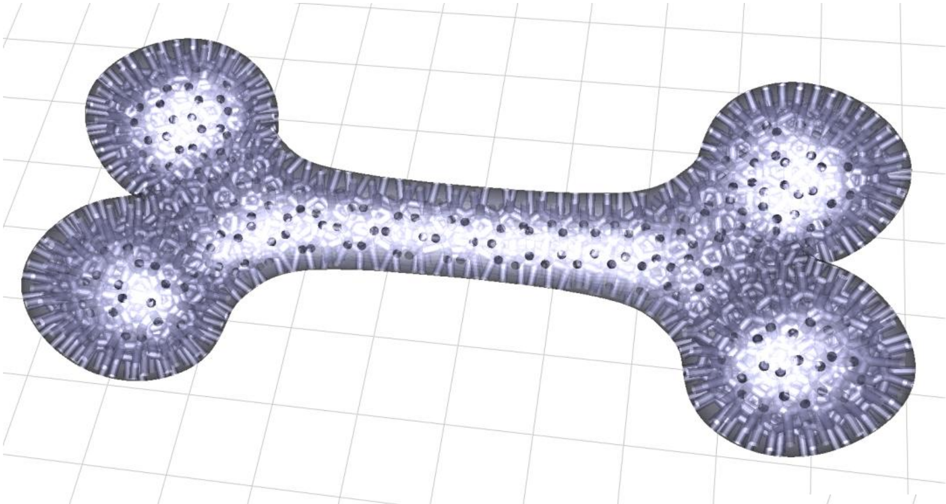
Clean Color, Hergel, Lefebvre, Eurographics 2014

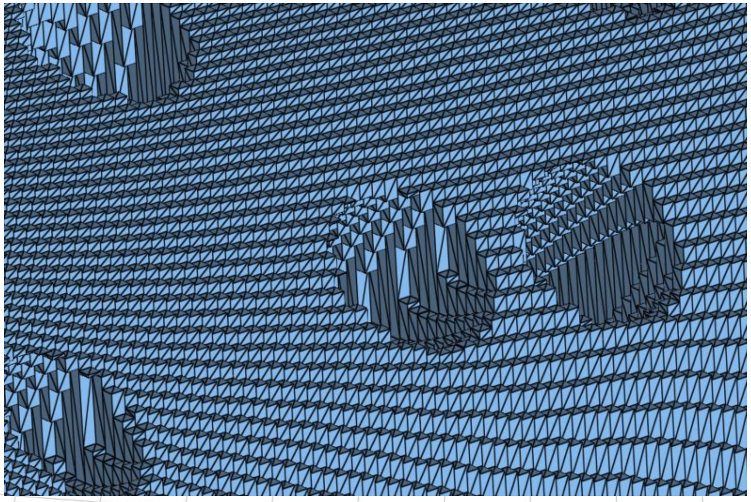


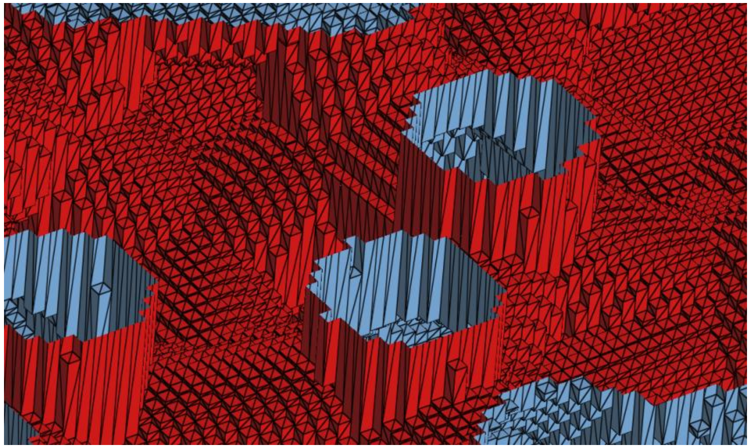
Bridge the gap, Dumas, Hergel, Lefebvre, SIGGRAPH 2014

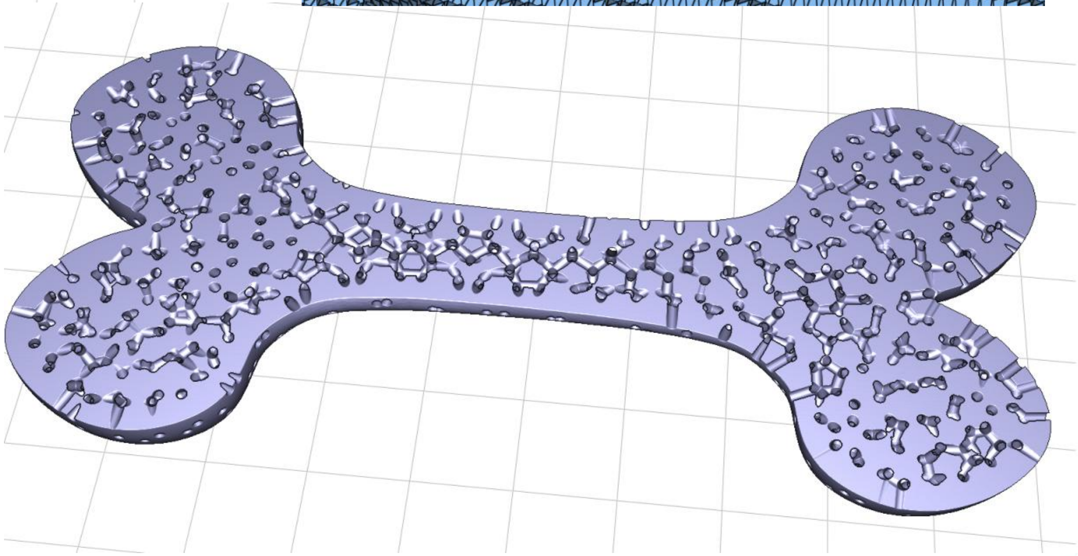
Part. 2 From Graphics to Fabrication

Reparative Surgery – toy example





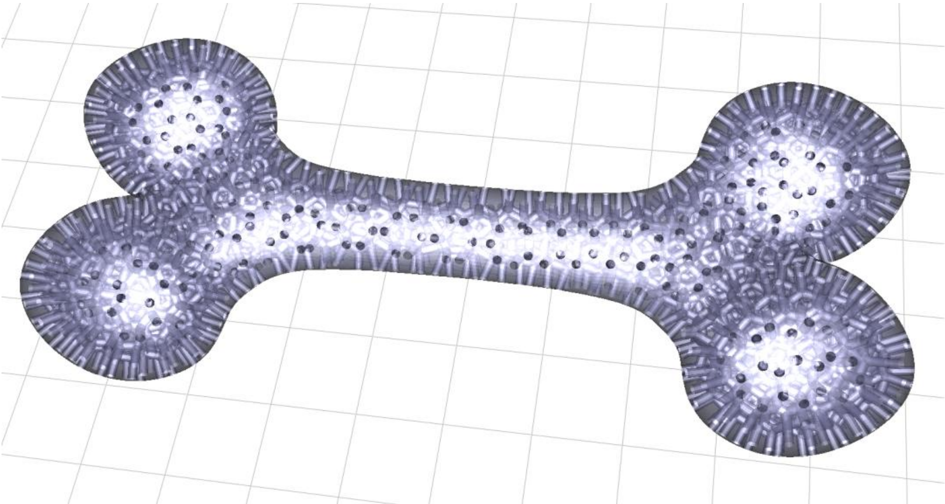




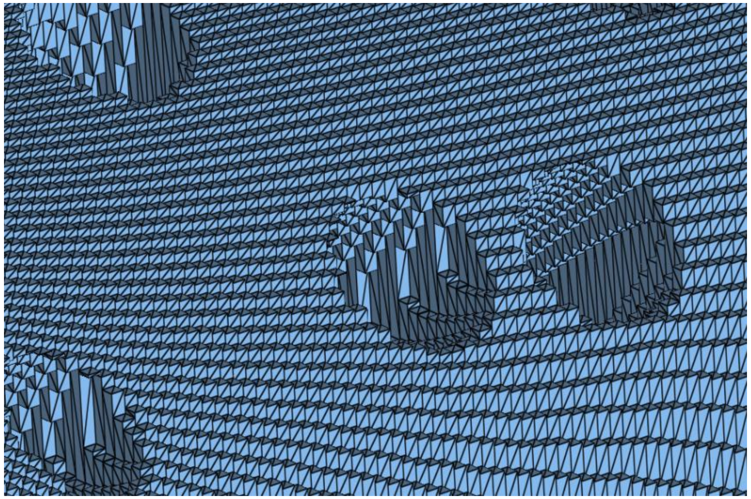
3

From Geometry Processing to Applied Mathematics

Part. 3 From Geometry Processing to Applied Math.



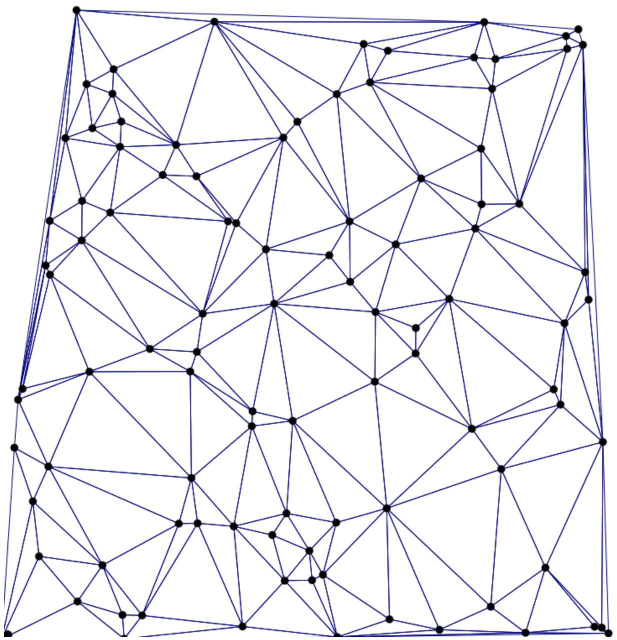
Exotic representation (Dexels)



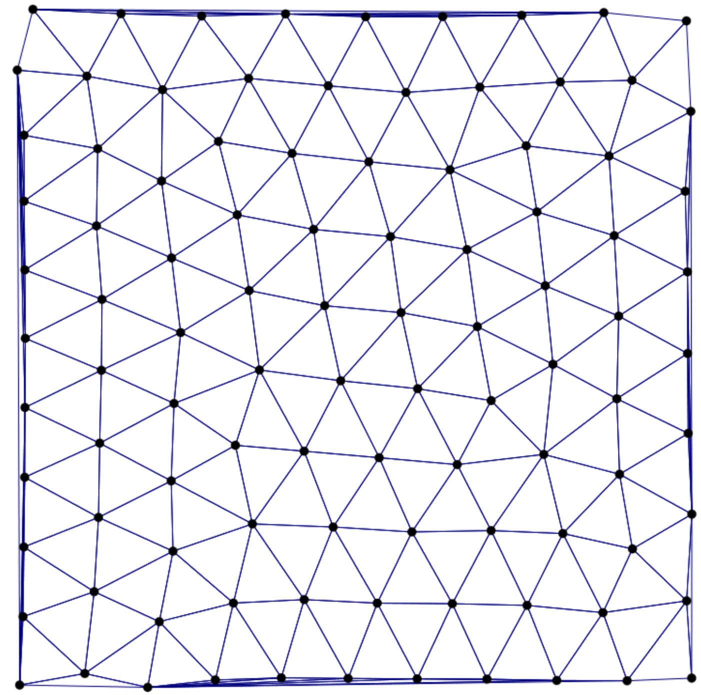
Back to the standard modeling pipeline…
Finite Element Modeling ?
How ?

Part. 3 From Geometry Processing to Applied Math.

Optimize a Voronoi diagram from the point of view of sampling regularity
(quantization noise power)



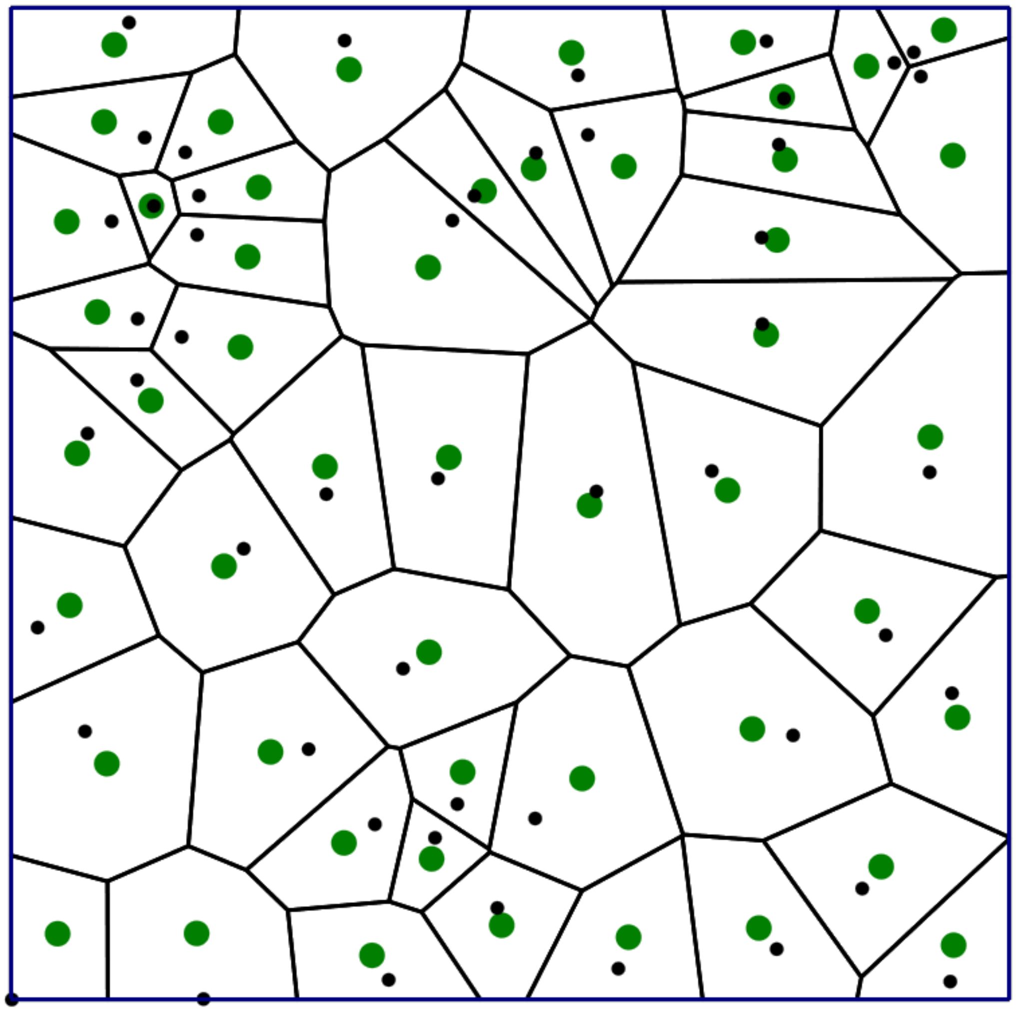
Minimize



$$F = \sum_i \int_{Vor(i)} \left\| \mathbf{x}_i - \mathbf{x} \right\|^2 d\mathbf{x}$$

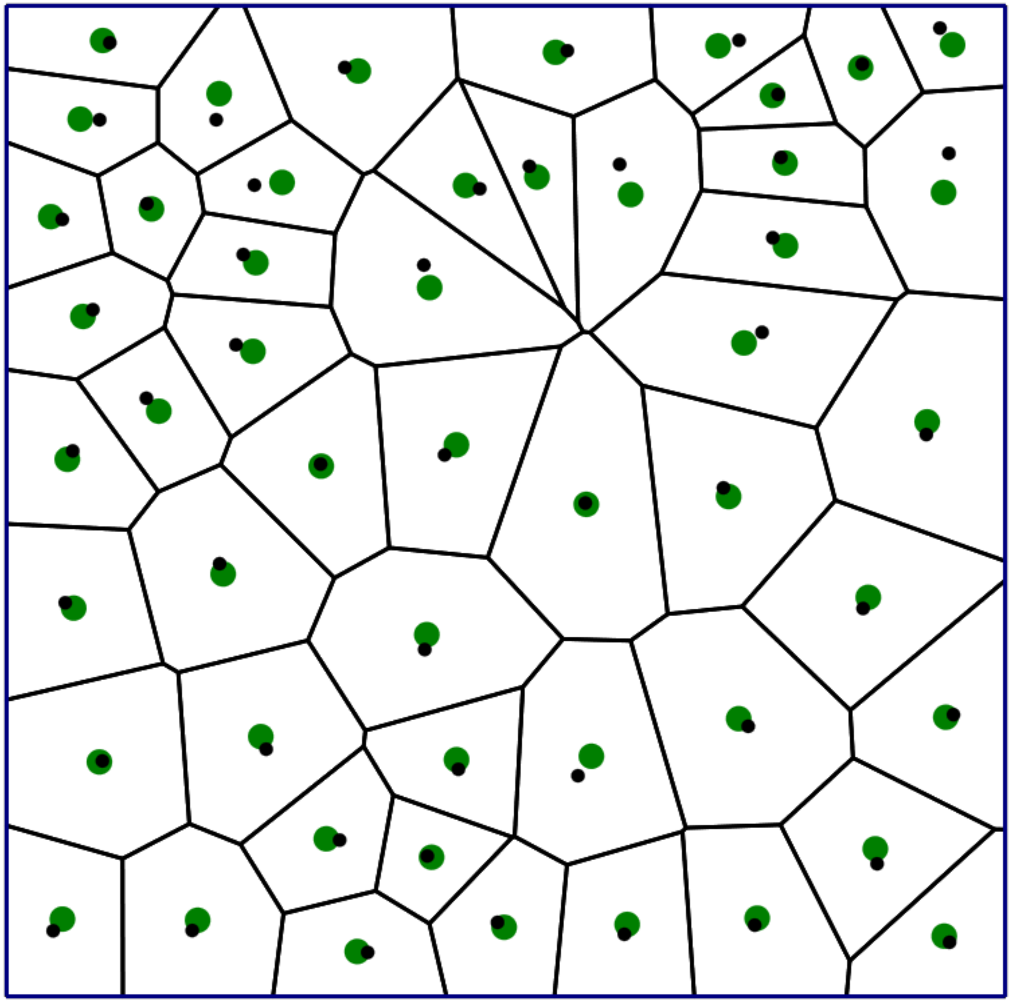
Theorem: F is of class C^2 [Liu, Wang, **L**, Yan, Lu, ACM TOG 2008]

Part. 3 From Geometry Processing to Applied Math.



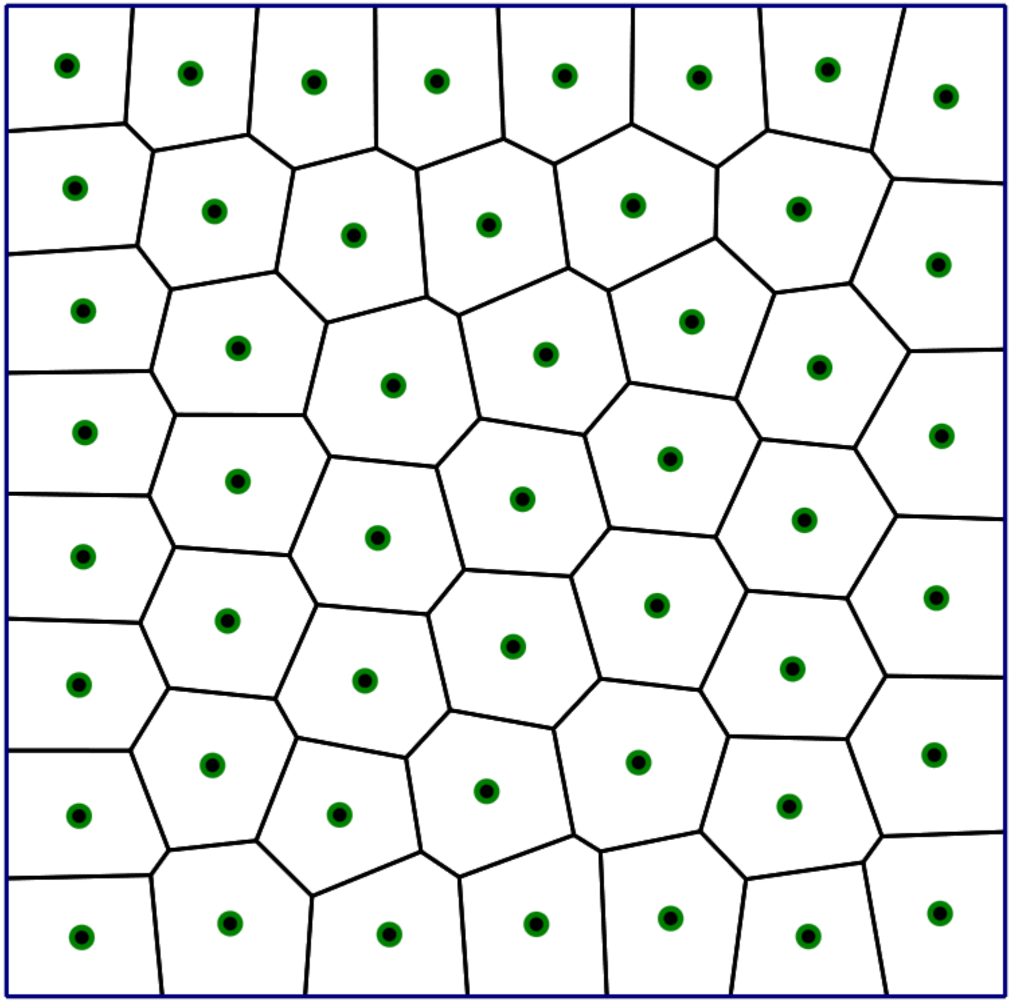
Theorem: F is of class C^2 [Liu, Wang, **L**, Yan, Lu, ACM TOG 2008]

Part. 3 From Geometry Processing to Applied Math.



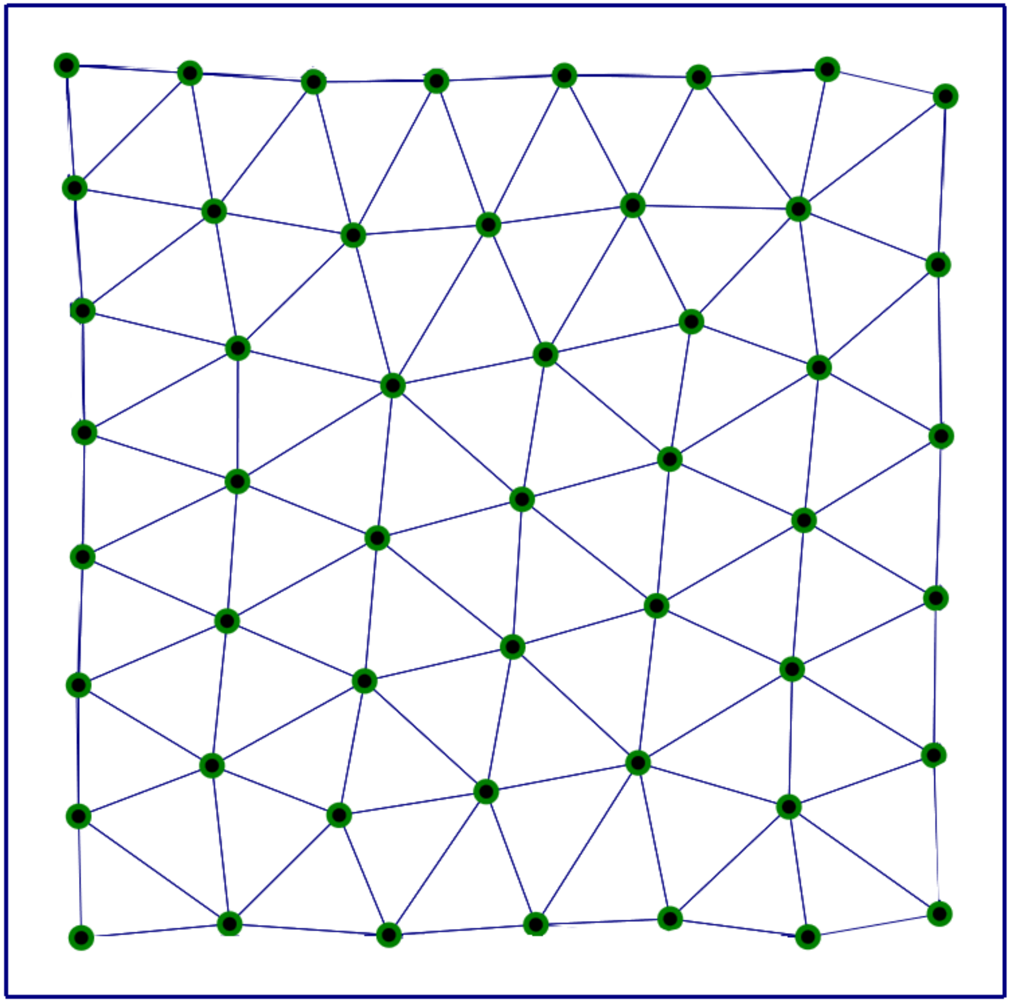
Theorem: F is of class C^2 [Liu, Wang, **L**, Yan, Lu, ACM TOG 2008]

Part. 3 From Geometry Processing to Applied Math.



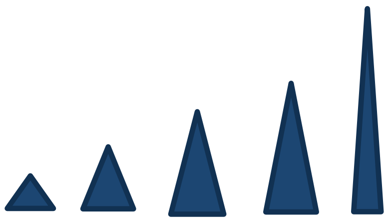
Theorem: F is of class C^2 [Liu, Wang, **L**, Yan, Lu, ACM TOG 2008]

Part. 3 From Geometry Processing to Applied Math.

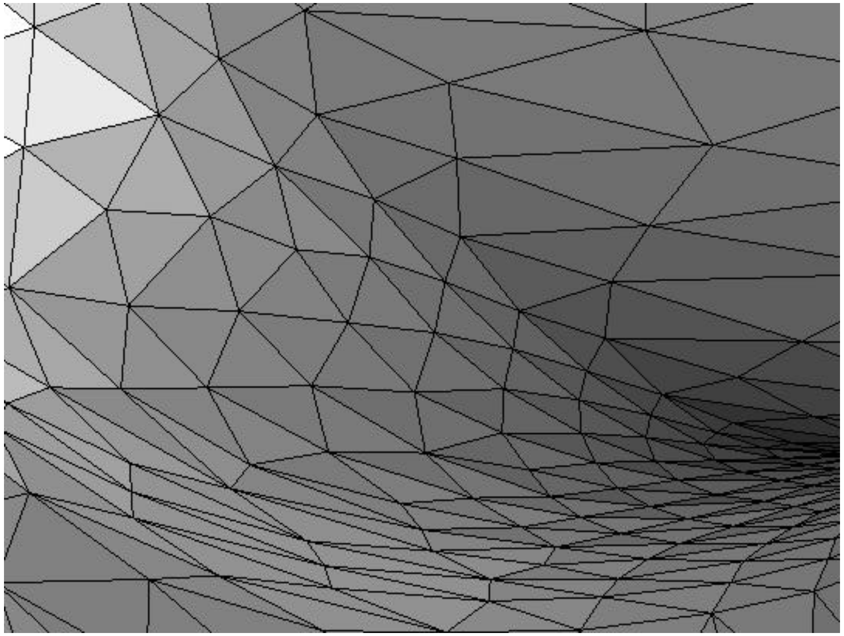


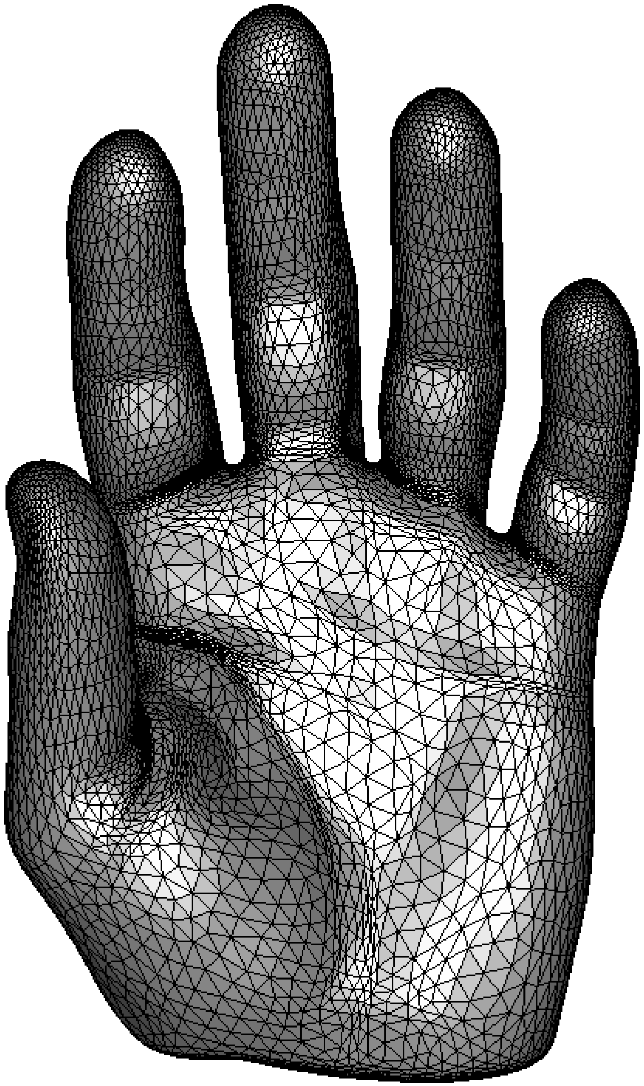
Theorem: F is of class C^2 [Liu, Wang, **L**, Yan, Lu, ACM TOG 2008]

Part. 3 From Geometry Processing to Applied Math.

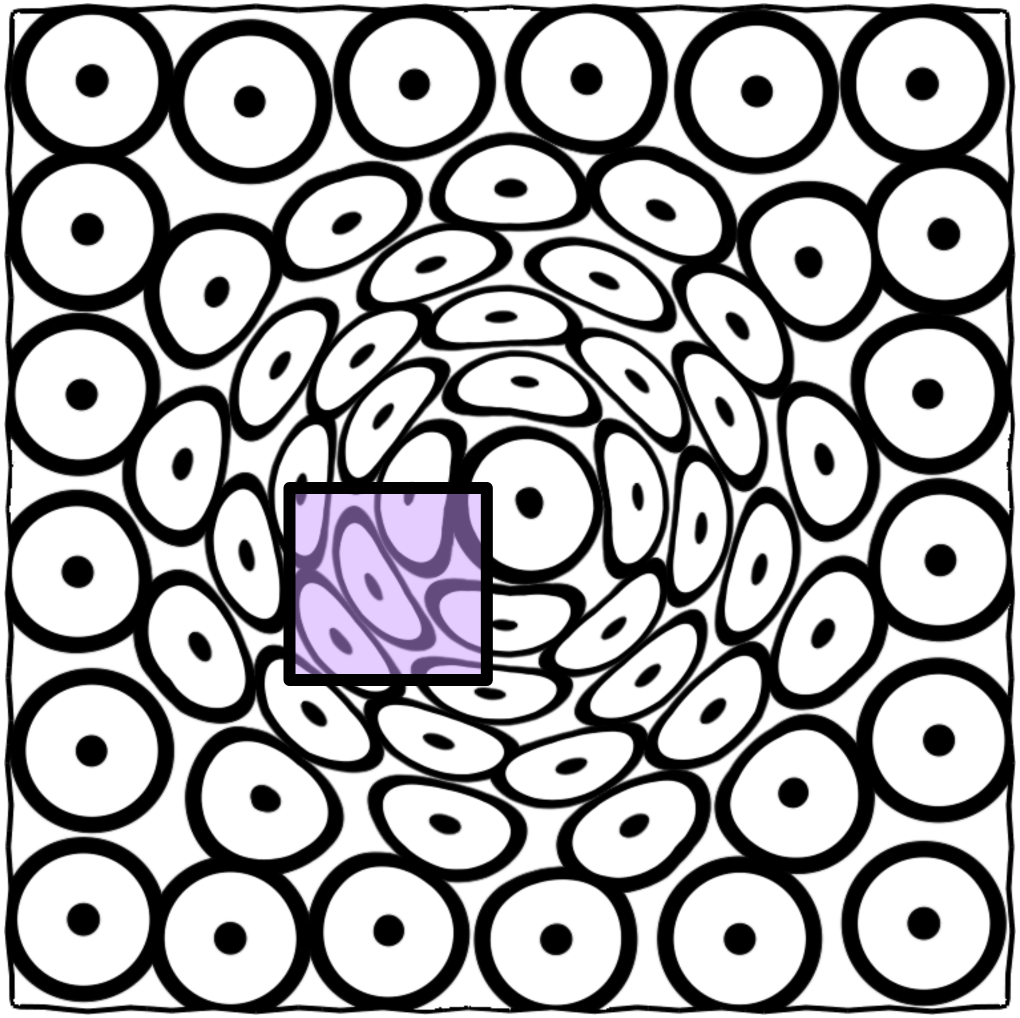


Anisotropic mesh:
* shape can vary
* size can vary



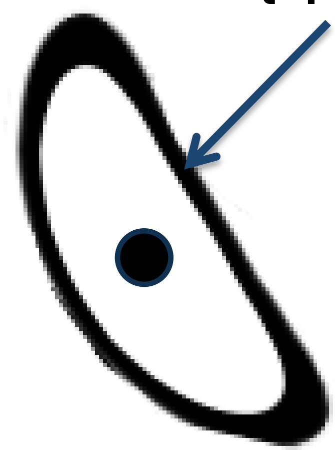


Part. 3. Anisotropy



The input: anisotropy field
Specifies shape and orientation

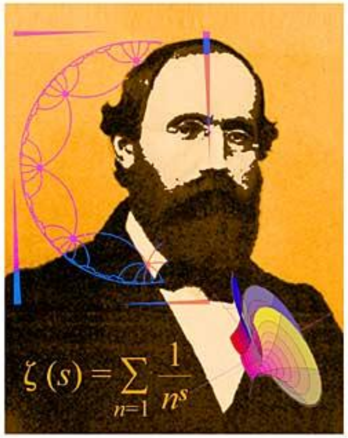
Anisotropy: An "alteration" of of distances and angles.



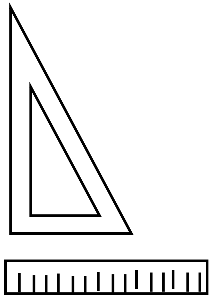
Part. 3. Anisotropy

The dot product: a geometric tool

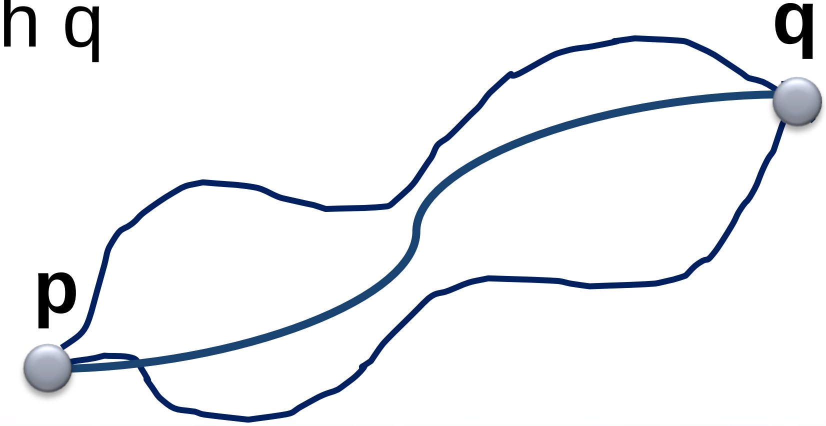
Anisotropic distance
between **p** *and* **q** *w.r.t.* G



Georg Friedrich Bernhard Riemann 1826 - 1866

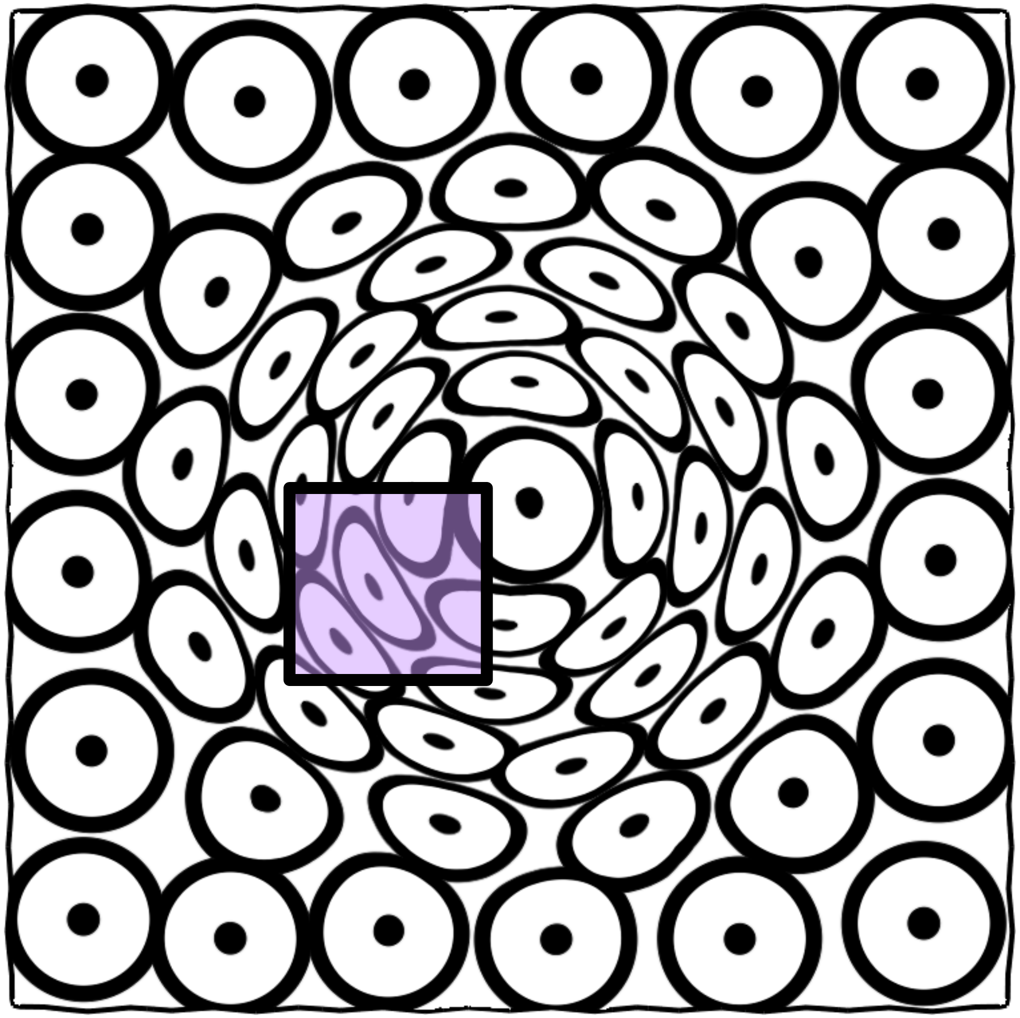


$d_G(\mathbf{p},\mathbf{q})$ = (anisotropic) length of
shortest curve
that connects p with q



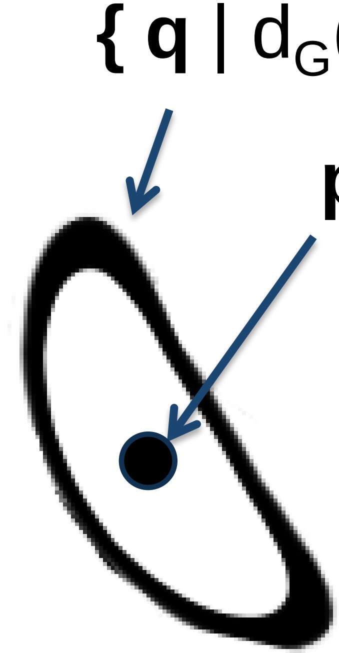
$$l_G(C) = \int_{t=0}^{1} \sqrt{v(t)^t \, G(t) \, v(t) \, dt}$$

Part. 3. Anisotropy

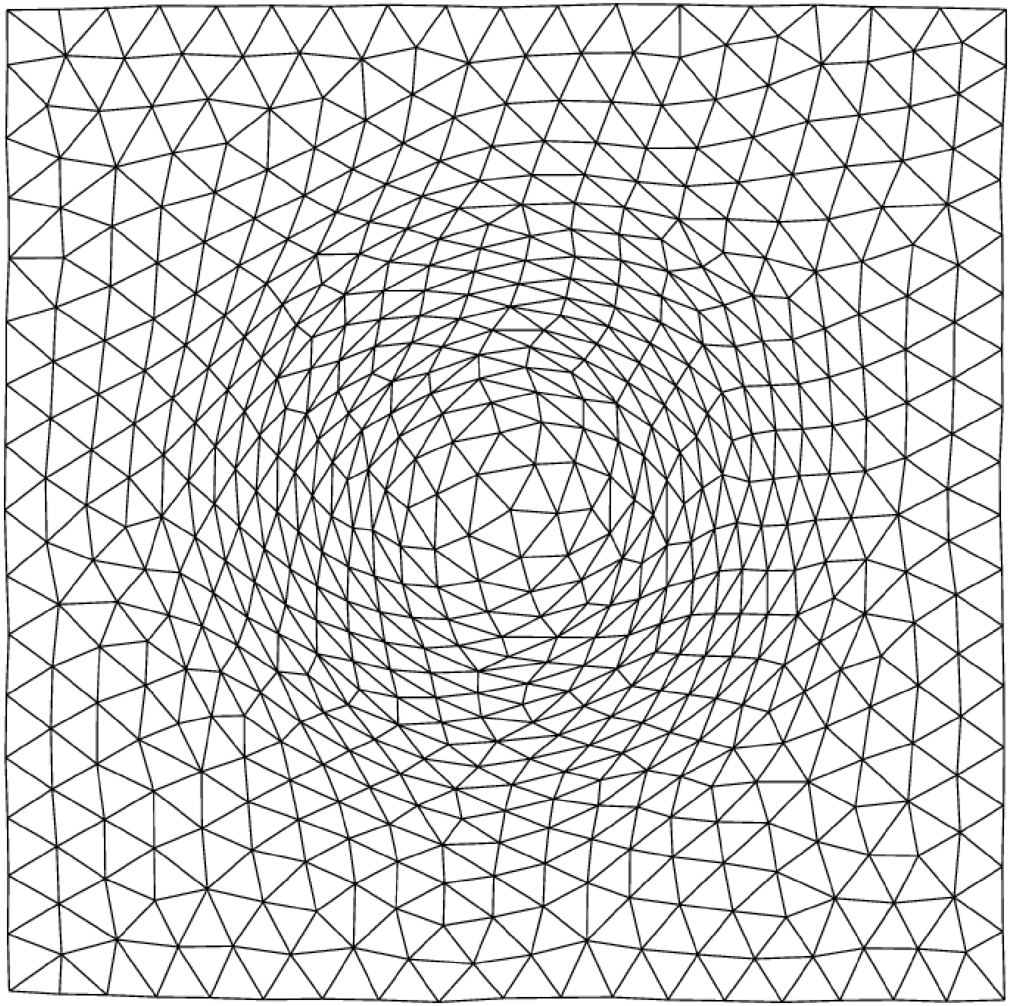


The input: anisotropy field

$$G(x,y) = \begin{bmatrix} a(x,y) & b(x,y) \\ b(x,y) & c(x,y) \end{bmatrix}$$

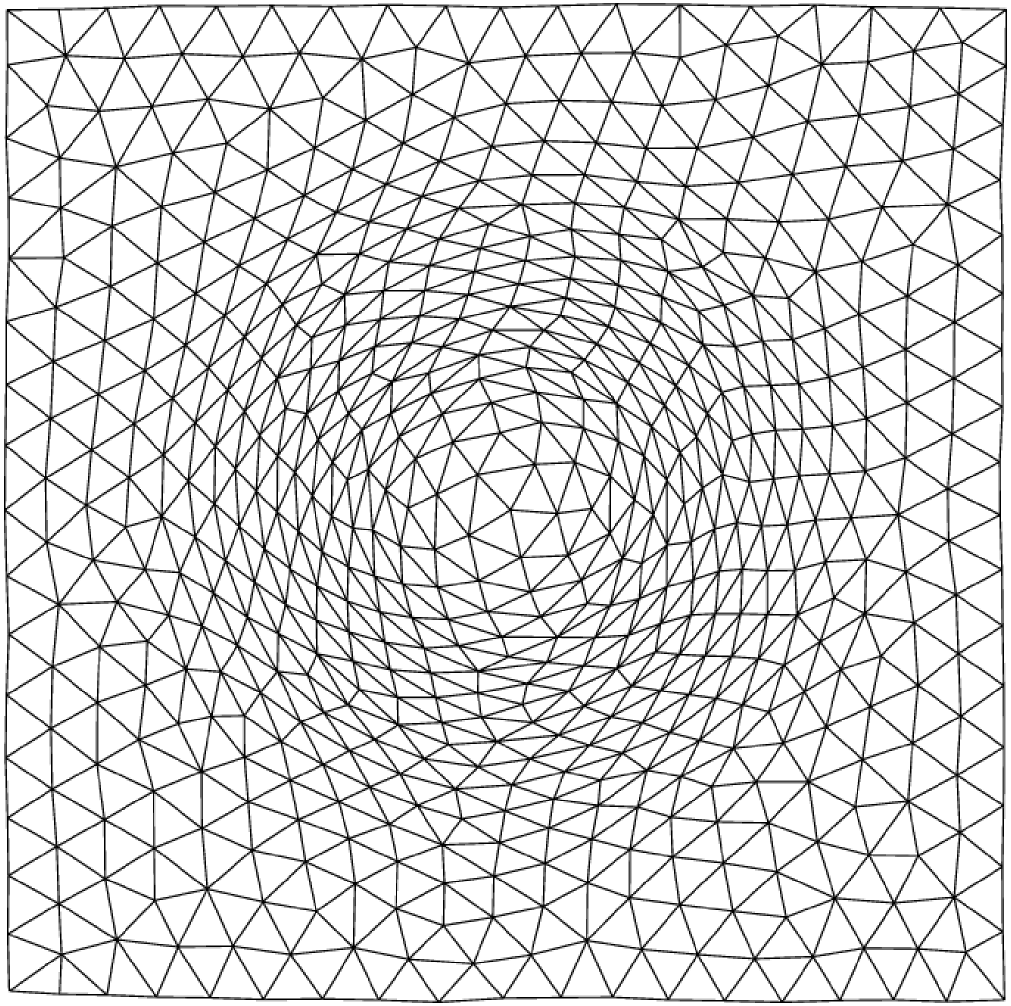


Part. 3. Anisotropy



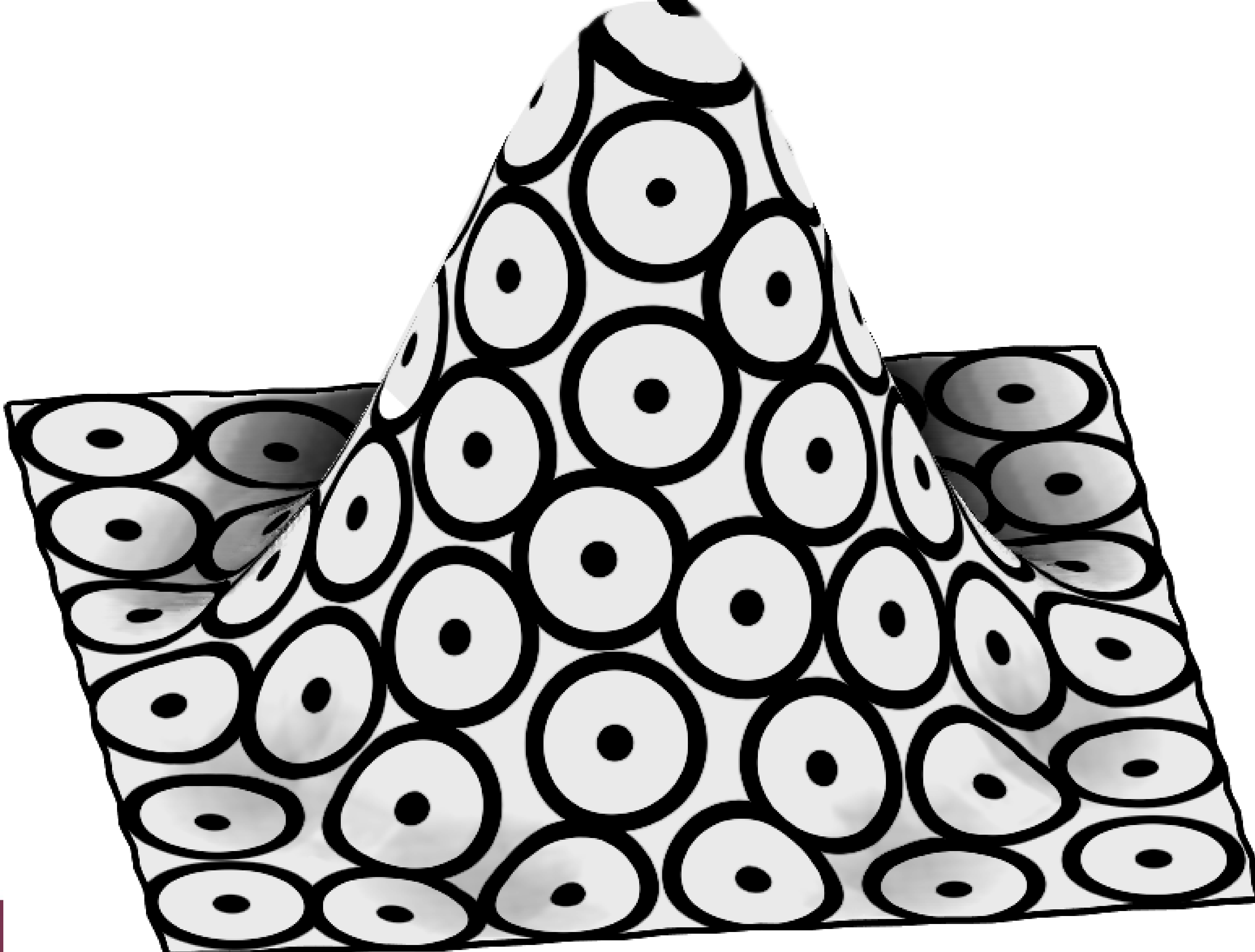
The result: triangles are "deformed" by the anisotropy.

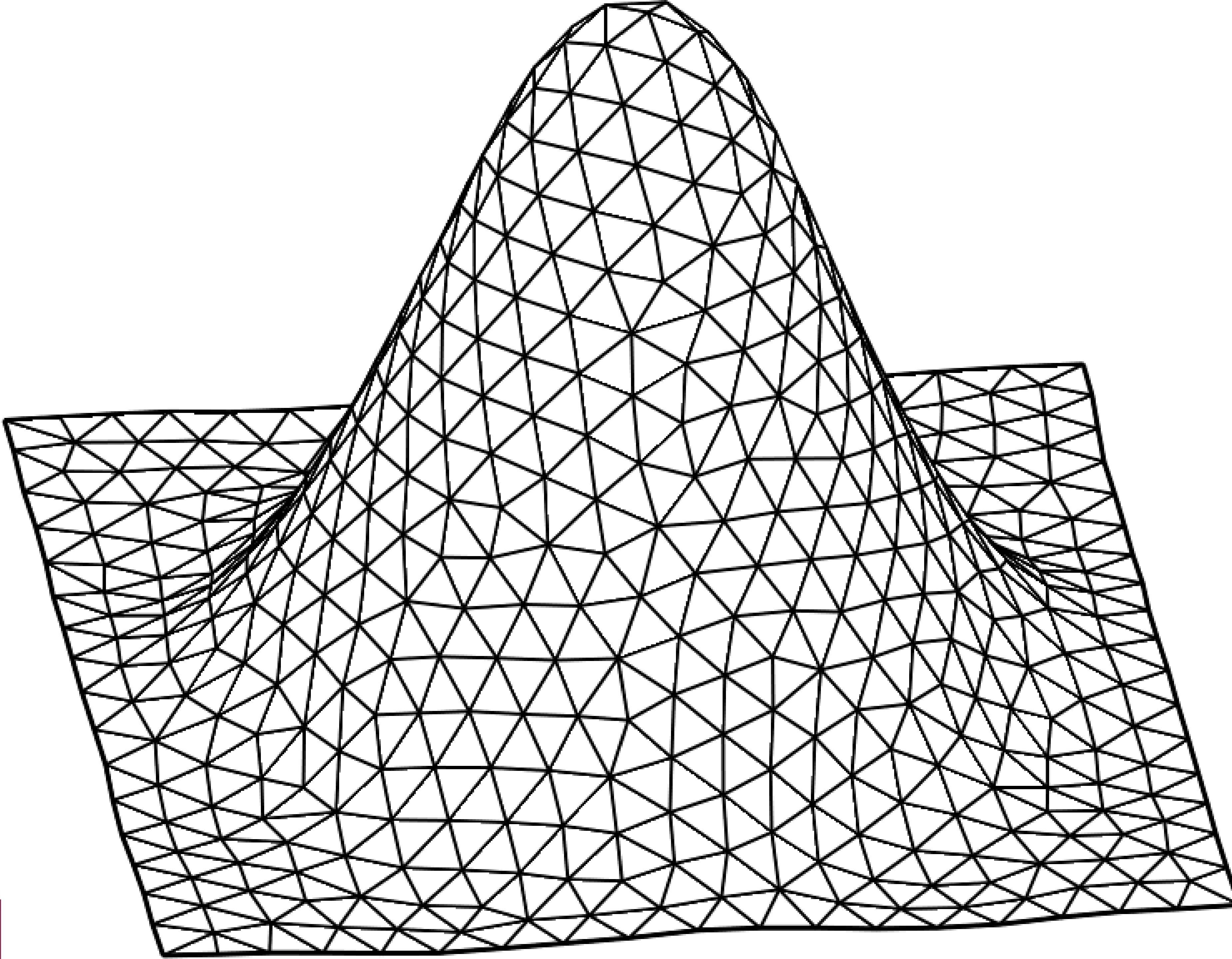
Part. 3. Anisotropy



The result: triangles are "deformed" by the anisotropy.

*Q: How to compute an **Anisotropic** Centroidal Voronoi Tessellation ?*





Part. 3 Journey in the 6^{th} dimension

The key idea

This example:

Anisotropic mesh in 2d ⟺ Isotropic mesh in 3d

Part. 3 Journey in the 6th dimension

The key idea

This example:

Anisotropic mesh in 2d $\Longleftrightarrow$ Isotropic mesh in 3d

Replace **anisotropy** with **additional dimensions**

Part. 3 Journey in the 6th dimension

The key idea

Replace **anisotropy** with **additional dimensions**

Note: more dimensions may be needed

Part. 3 Journey in the 6^{th} dimension

The key idea

Replace **anisotropy** with **additional dimensions**

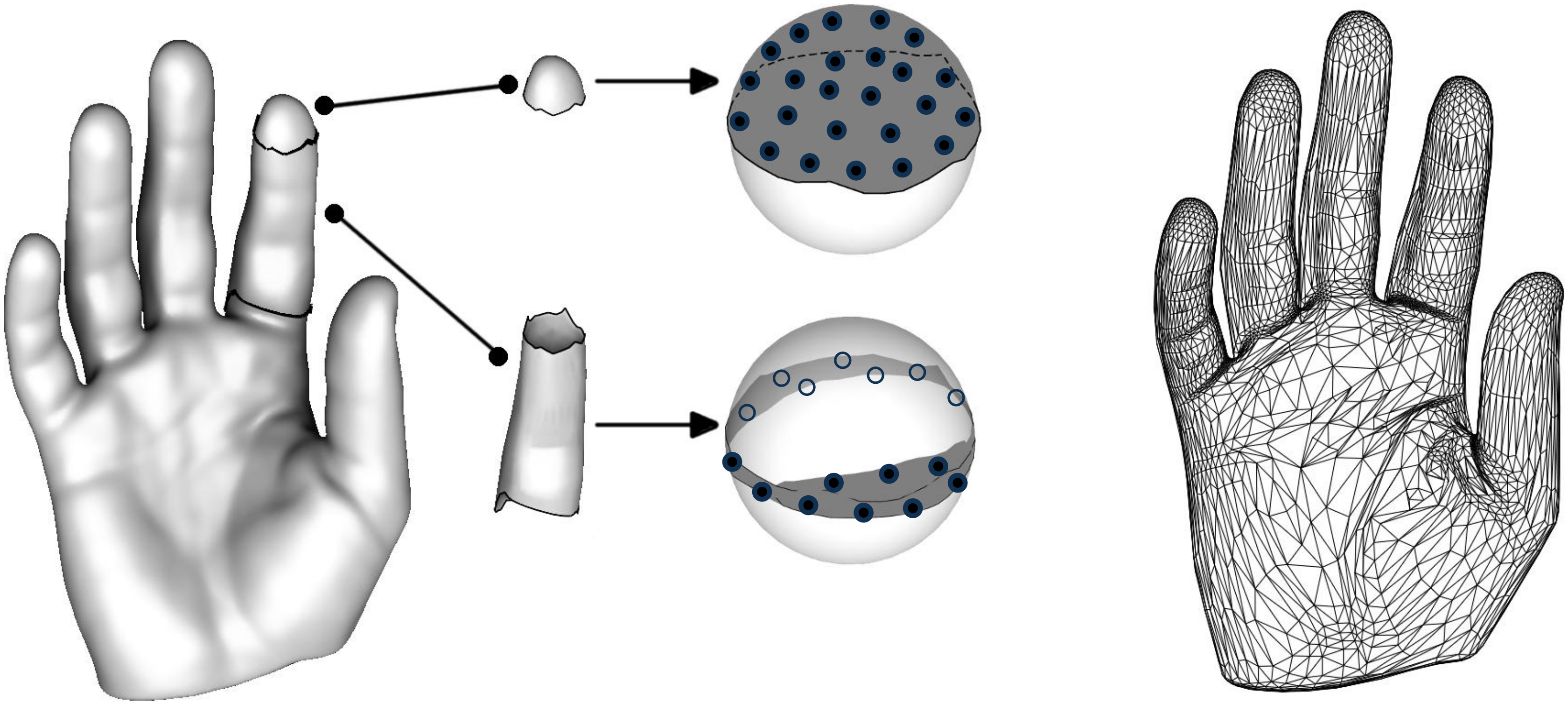
Note: more dimensions may be needed
How many ?
John Nash's isometric embedding theorem:

Maximum: depending on desired smoothness
C^1 : $2n$ [Nash-Kuiper]
C^k : bounded by $n(3n+11)/2$ [Nash, Nash-Moser]

Part. 3 Journey in the 6^{th} dimension

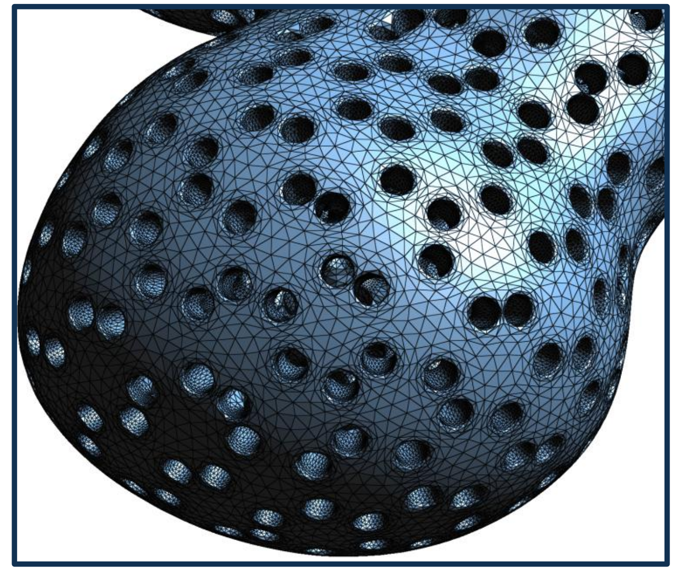
A 6d embedding for curvature-adapted meshing

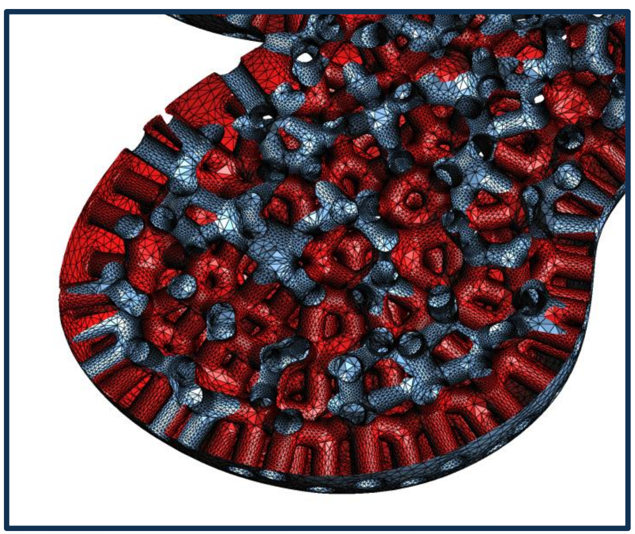
The Gauss map: $\begin{bmatrix} x \\ y \\ z \end{bmatrix} \rightarrow \begin{bmatrix} N_x \\ N_y \\ N_z \end{bmatrix}$



Part. 3 Journey in the 6th dimension

A 6d embedding for curvature-adapted meshing

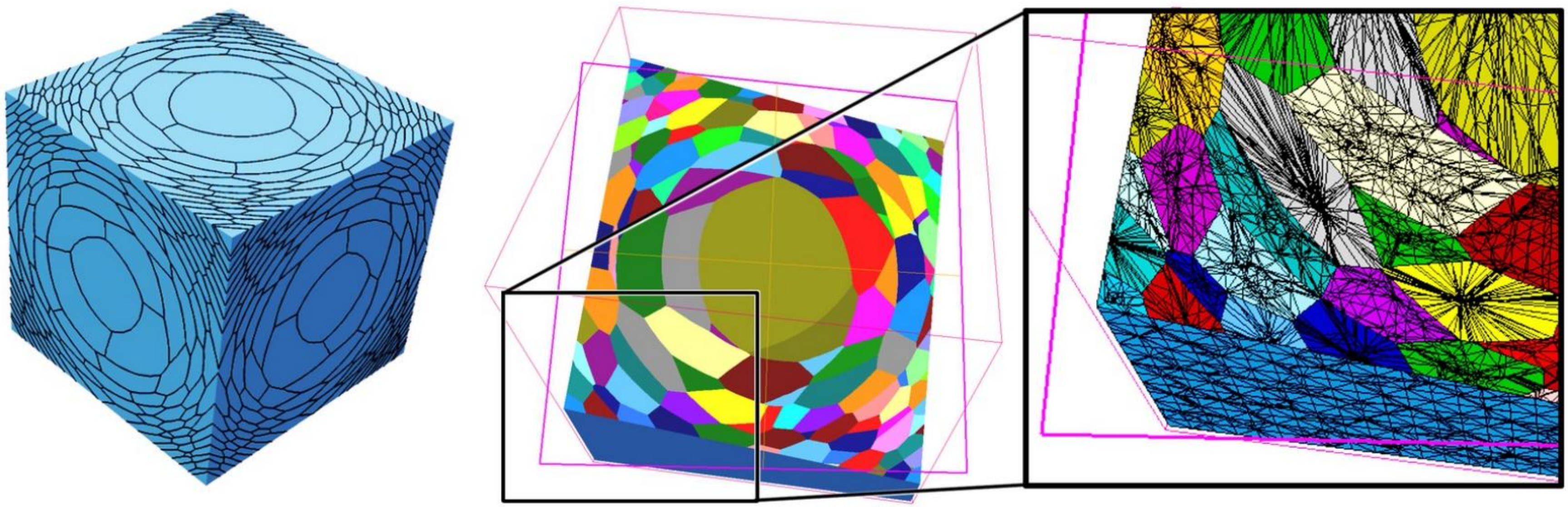




Vorpaline meshing software
ERC “Proof of Concept”

Part. 3 Journey in the 6th dimension

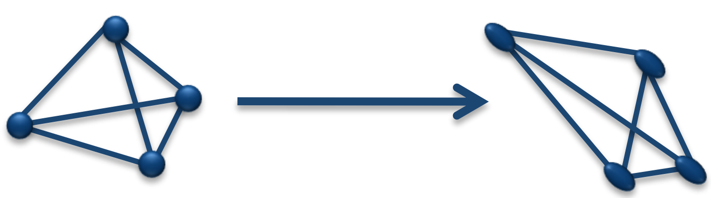
Anisotropy through high-dim. embedding



3D anisotropic Voronoi diagram and anisotropic Vector Quantization

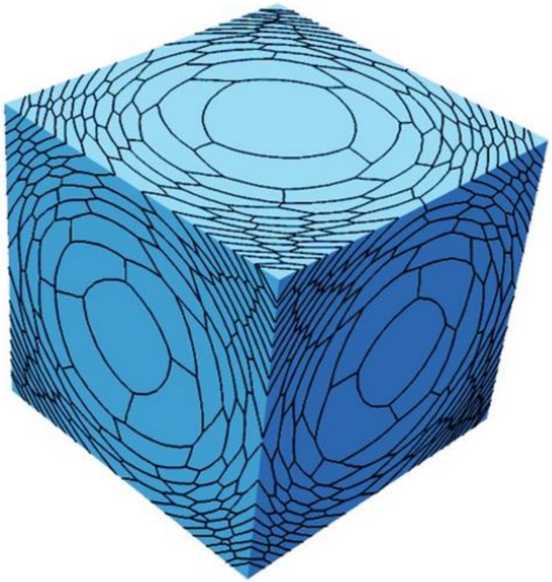
Anisotropy represented by a background mesh embedded in 6D

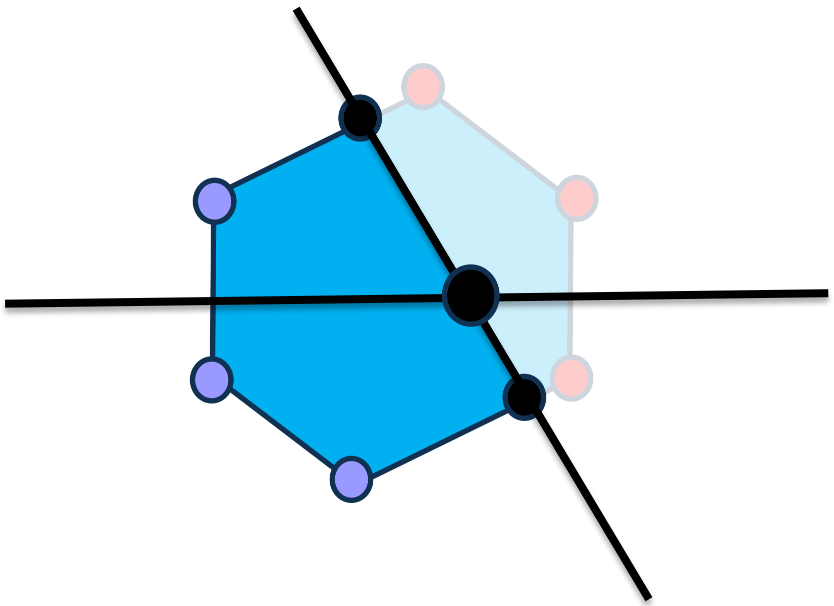
$$\mathbf{G_t = J_t^{\mathit{t}} J_t}$$



Part. 3 Journey in the 6th dimension

New predicates





$$
\begin{aligned}
side_1(\mathbf{p}_1,\mathbf{p}_2,\mathbf{q}) &= Sign(d^2(\mathbf{p}_2,\mathbf{q}) - d^2(\mathbf{p}_1,\mathbf{q}))\\
side_2(\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3,\mathbf{q}_1,\mathbf{q}_2) &= side_1(\mathbf{p}_1,\mathbf{p}_2,\mathbf{q}) \quad \text{where} \quad \mathbf{q} = \Pi(\mathbf{p}_1,\mathbf{p}_3)\cap[\mathbf{q}_1\mathbf{q}_2]\\
side_3(\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3,\mathbf{p}_4,\mathbf{q}_1,\mathbf{q}_2,\mathbf{q}_3) &= side_1(\mathbf{p}_1,\mathbf{p}_2,\mathbf{q}) \quad \text{where} \quad \mathbf{q} = \Pi(\mathbf{p}_1,\mathbf{p}_3)\cap\Pi(\mathbf{p}_1,\mathbf{p}_4)\cap\Delta(\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3)\\
side_4(\mathbf{p}_1,\ldots\mathbf{p}_5,\mathbf{q}_1,\ldots\mathbf{q}_4) &= side_1(\mathbf{p}_1,\mathbf{p}_2,\mathbf{q}) \ , \quad \mathbf{q} = \Pi(\mathbf{p}_1,\mathbf{p}_3)\cap\Pi(\mathbf{p}_1,\mathbf{p}_4)\cap\Pi(\mathbf{p}_1\mathbf{p}_5)\cap tet(\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3\mathbf{q}_4)
\end{aligned}
$$

Part. 3 Journey in the 6th dimension

New predicates

```
Sign side2(
    point p0, point p1, point p2,
    point q0, point q1
) {

    scalar l1 = sq_dist(p1,p0) ;
    scalar l2 = sq_dist(p2,p0) ;

    scalar a10 = 2*dot_at(p1,q0,p0);
    scalar a11 = 2*dot_at(p1,q1,p0);
    scalar a20 = 2*dot_at(p2,q0,p0);
    scalar a21 = 2*dot_at(p2,q1,p0);

    scalar Delta = a11 - a10 ;
    scalar DeltaLambda0 = a11 - l1 ;
    scalar DeltaLambda1 =  l1 - a10 ;
    scalar r =
        Delta*l2-a20*DeltaLambda0-a21*DeltaLambda1 ;

    Sign Delta_sign = sign(Delta) ;
    Sign r_sign     = sign(r) ;

    generic_predicate_result(Delta_sign*r_sign) ;

    begin_sos3(p0,p1,p2)
       sos(p0, Sign(Delta_sign*sign(Delta-a21+a20)))
       sos(p1, Sign(Delta_sign*sign(a21-a20)))
       sos(p2, NEGATIVE)
    end_sos
}
```

Part. 3 Journey in the 6^{th} dimension

New predicates

```
Sign side2(
    point p0, point p1, point p2,
    point q0, point q1
) {

    scalar l1 = sq_dist(p1,p0) ;
    scalar l2 = sq_dist(p2,p0) ;

    scalar a10 = 2*dot_at(p1,q0,p0);
    scalar a11 = 2*dot_at(p1,q1,p0);
    scalar a20 = 2*dot_at(p2,q0,p0);
    scalar a21 = 2*dot_at(p2,q1,p0);

    scalar Delta = a11 - a10 ;
    scalar DeltaLambda0 = a11 - l1 ;
    scalar DeltaLambda1 =  l1 - a10 ;
    scalar r =
        Delta*l2-a20*DeltaLambda0-a21*DeltaLambda1 ;

    Sign Delta_sign = sign(Delta) ;
    Sign r_sign     = sign(r) ;

    generic_predicate_result(Delta_sign*r_sign) ;

    begin_sos3(p0,p1,p2)
       sos(p0, Sign(Delta_sign*sign(Delta-a21+a20)))
       sos(p1, Sign(Delta_sign*sign(a21-a20)))
       sos(p2, NEGATIVE)
    end_sos
}
```

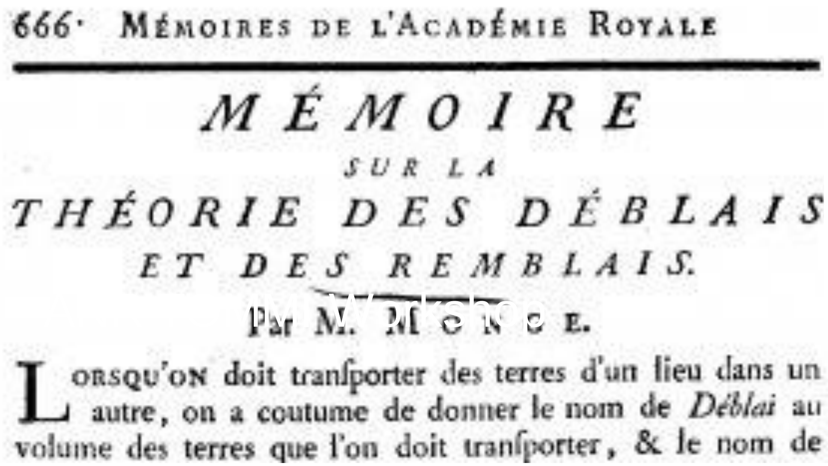
Automatically generated

```
Sign side2_exact_SOS(
    const double* p0,const double* p1,const double* p2,
    const double* q0,const double* q1,
    coord_index_t dim
) {
    const expansion& l1 = expansion_sq_dist(p1,p0,dim);
    const expansion& l2 = expansion_sq_dist(p2,p0,dim);
    const expansion& a10 = expansion_dot_at(p1,q0,p0,dim).scale_fast(2.0);
    const expansion& a11 = expansion_dot_at(p1,q1,p0,dim).scale_fast(2.0);
    const expansion& a20 = expansion_dot_at(p2,q0,p0,dim).scale_fast(2.0);
    const expansion& a21 = expansion_dot_at(p2,q1,p0,dim).scale_fast(2.0);
    const expansion& Delta = expansion_diff(a11,a10);
    Sign Delta_sign = Delta.sign(); vor_assert(Delta_sign != ZERO);

    const expansion& DeltaLambda0 = expansion_diff(a11,l1);
    const expansion& DeltaLambda1 = expansion_diff(l1,a10);
    const expansion& r0 = expansion_product(Delta,l2);
    const expansion& r1 = expansion_product(a20,DeltaLambda0).negate();
    const expansion& r2 = expansion_product(a21,DeltaLambda1).negate();
    const expansion& r = expansion_sum3(r0,r1,r2);
    Sign r_sign = r.sign();

    if(r_sign == ZERO) {
        const double* p_sort[3];
        p_sort[0] = p0;
        p_sort[1] = p1;
        p_sort[2] = p2;
        std::sort(p_sort,p_sort + 3);
        for(index_t i = 0; i < 3; ++i) {
            if(p_sort[i] == p0) {
                const expansion& z1 = expansion_diff(Delta,a21);
                const expansion& z = expansion_sum(z1,a20);
                Sign z_sign = z.sign();
                if(z_sign != ZERO) { return Sign(Delta_sign * z_sign); }
            }
            if(p_sort[i] == p1) {
                const expansion& z = expansion_diff(a21,a20);
                Sign z_sign = z.sign();
                if(z_sign != ZERO) { return Sign(Delta_sign * z_sign); }
            }
            if(p_sort[i] == p2) {
                return NEGATIVE;
            }
        }
        vor_assert_not_reached;
    }
    return Sign(Delta_sign * r_sign);
}
```

Predicate Construction Toolkit [PCK] – make it easy for everybody

Part. 3 Optimal Transport

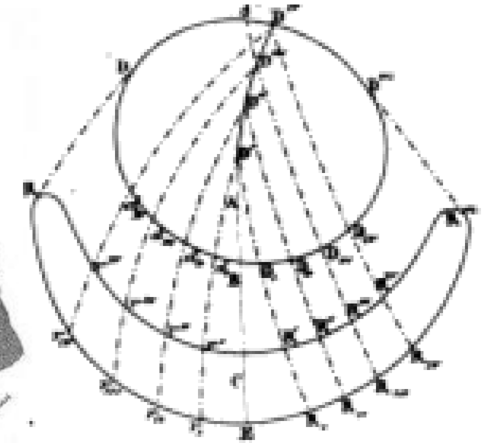
Gaspard Monge - 1784

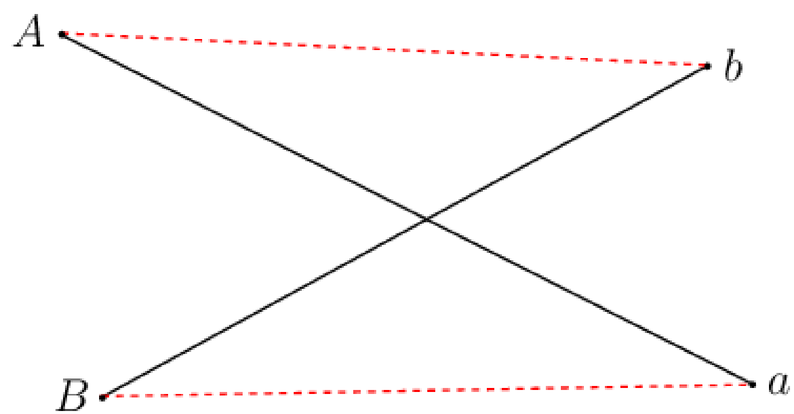

666. MÉMOIRES DE L'ACADÉMIE ROYALE

MÉMOIRE
SUR LA
THÉORIE DES DÉBLAIS
ET DES REMBLAIS.

Par M. MONGE.

Lorsqu'on doit tranſporter des terres d'un lieu dans un autre, on a coutume de donner le nom de *Déblai* au volume des terres que l'on doit tranſporter, & le nom de





Part. 3 Optimal Transport – some references

A Multiscale Approach to Optimal Transport,
Quentin Mérigot, Computer Graphics Forum, 2011

Variational Principles for Minkowski Type Problems, Discrete Optimal Transport, and Discrete Monge-Ampere Equations
Xianfeng Gu, Feng Luo, Jian Sun, S.-T. Yau, ArXiv 2013

Minkowski-type theorems and least-squares clustering
AHA! (Aurenhammer, Hoffmann, and Aronov), SIAM J. on math. ana. 1998

Topics on Optimal Transportation, 2003
Optimal Transport Old and New, 2008
Cédric Villani

Yann Brénier, Jean-David Benamou

Part. 3 Optimal Transport – Monge's problem

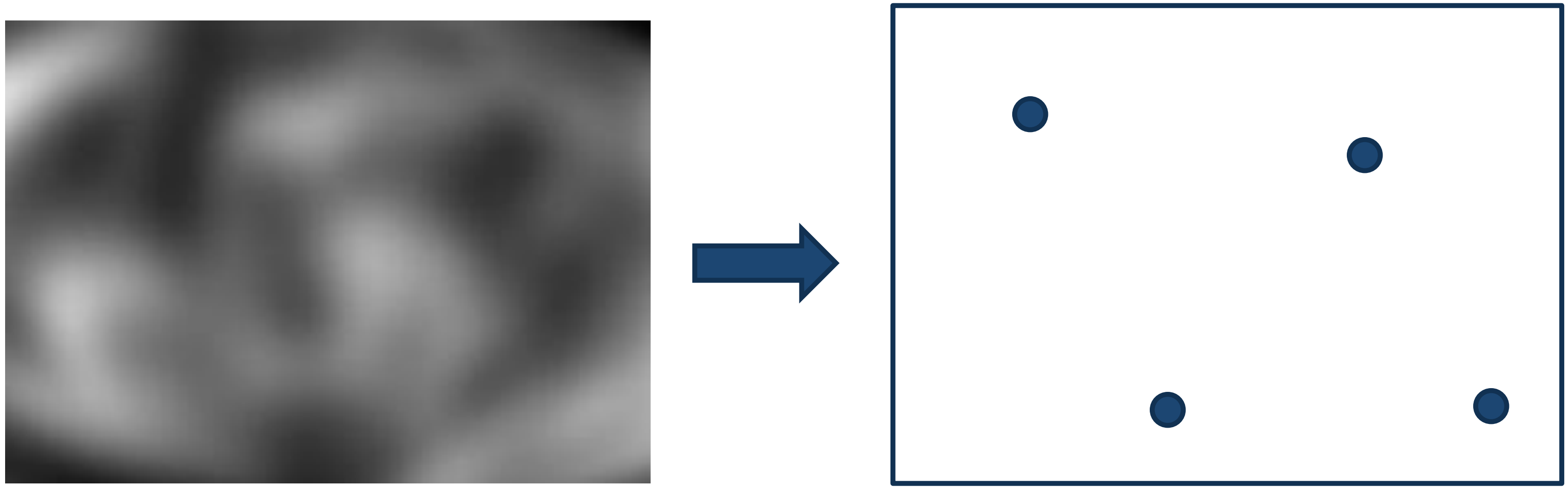
μ

ν

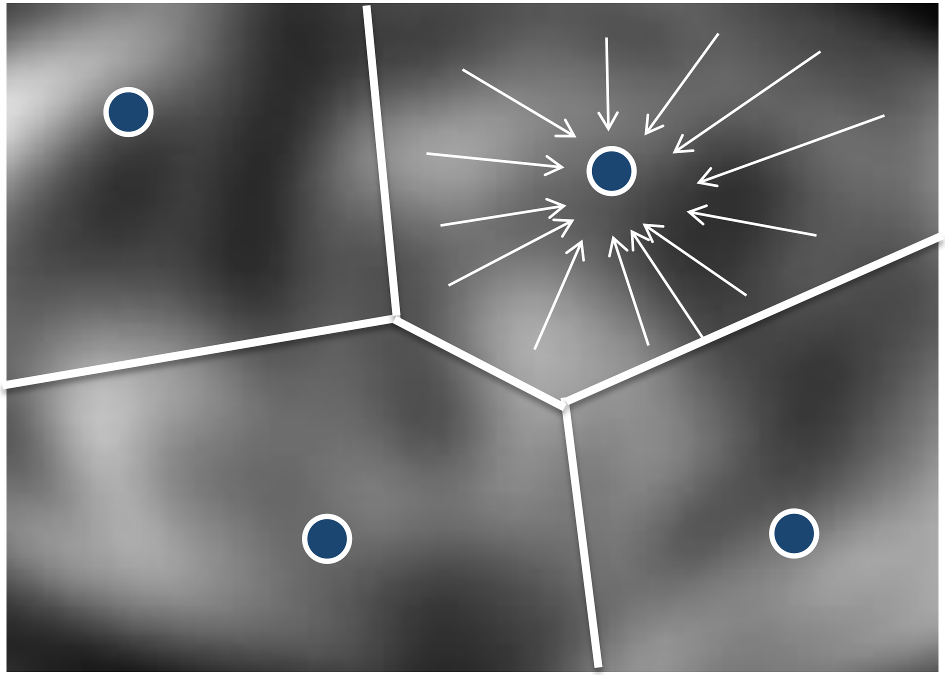
Monge's problem:

Find a transport map T that minimizes $C(T) = \int_{\Omega} || x - T(x) ||^2 \, d\mu(x)$

Part. 3 Optimal Transport – semi-discrete

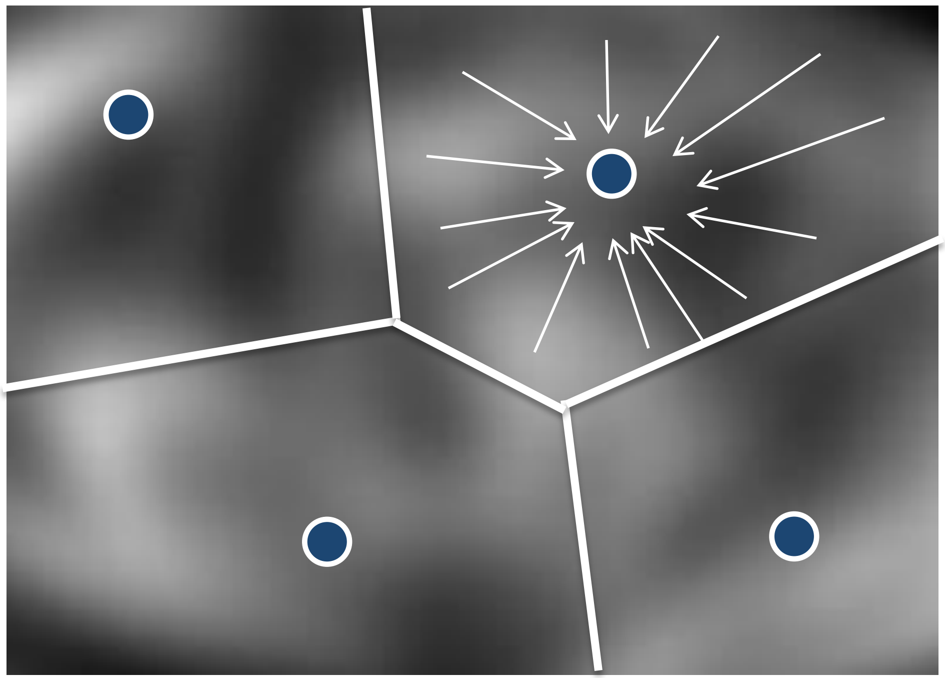


Part. 3 Optimal Transport – semi-discrete



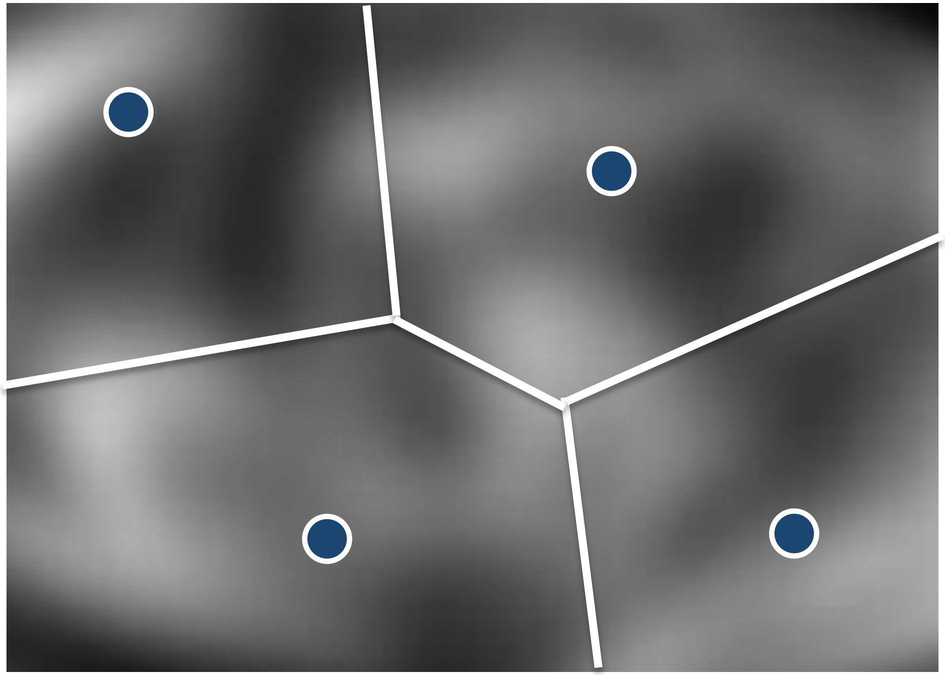
The pre-images of the Diracs define a partition of Ω

Part. 3 Optimal Transport – semi-discrete



The pre-images of the Diracs define a partition of Ω
This partition is a **power diagram** (more on this below)

Part. 3 Optimal Transport – the AHA paper



Theorem **[Aurenhammer, Hoffmann, Aronov 98], [Brenier91]**:

given a measure μ with density, a set of points (S), a set of positive coefficients λ such that $\sum \lambda_i = \int d\mu(x)$, it is possible to find the weights w such that the map $T_S{}^W$ is an optimal transport map between μ and $\nu = \sum \lambda_i \, \delta(s_i)$

Given the points (S), one can find the weights (w) such that $\int_{Pow(si)} d\mu(x) = \lambda_i$

Part. 3 Optimal Transport – the algorithm

The [AHA] paper summary:

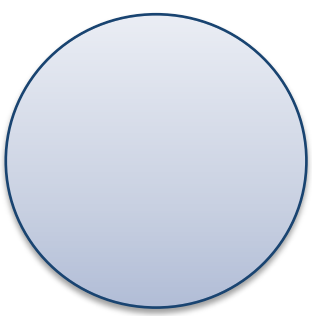
- The optimal weights minimize a convex function
- The gradient of this convex function is easy to compute

Note: the weight w(s) correspond to the Kantorovich potential $\psi(x)$
(solves a “discrete Monge-Ampere” equation)

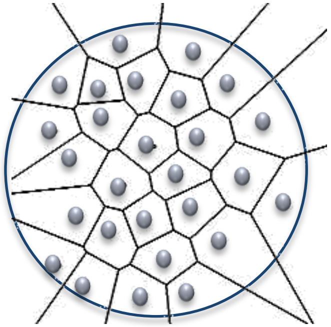
The algorithm:

Summary:

The algorithm computes the weights w_i such that the power cells associated with the Diracs correspond to the preimages of the Diracs.



μ



ν

Part. 3 Optimal Transport – the algorithm

The [AHA] paper summary:

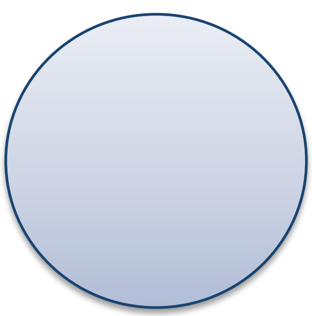
- The optimal weights minimize a convex function
- The gradient of this convex function is easy to compute

Note: the weight w(s) correspond to the Kantorovich potential $\psi(x)$
(solves a “discrete Monge-Ampere” equation)

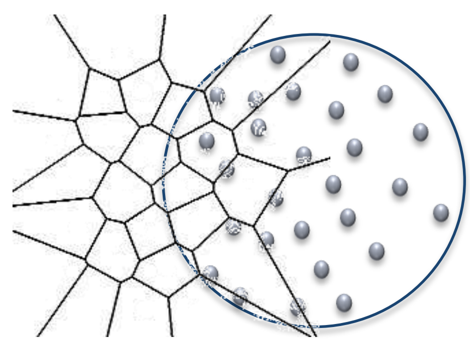
The algorithm:

Summary:

The algorithm computes the weights w_i such that the power cells associated with the Diracs correspond to the preimages of the Diracs.



μ



ν

Part. 3 Optimal Transport – the algorithm

The [AHA] paper summary:

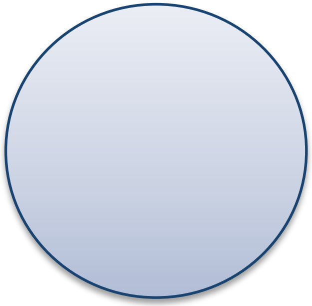
• The optimal weights minimize a convex function
• The gradient of this convex function is easy to compute

Note: the weight w(s) correspond to the Kantorovich potential $\psi(x)$
(solves a "discrete Monge-Ampere" equation)

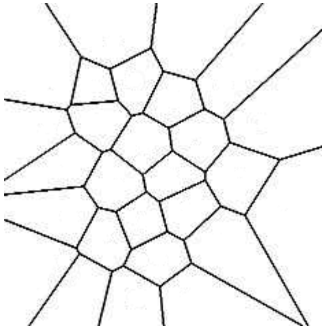
The algorithm:

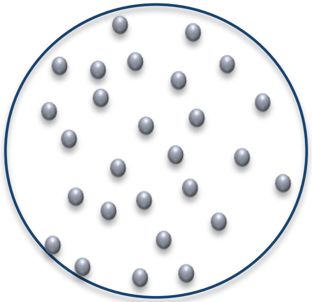
Summary:

The algorithm computes the weights w_i such that the power cells associated with the Diracs correspond to the preimages of the Diracs.



μ





ν

Part. 3 Optimal Transport – the algorithm

The [AHA] paper summary:

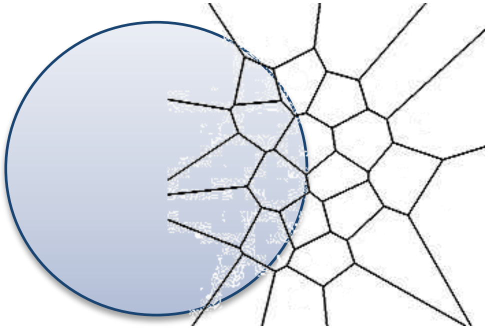
- The optimal weights minimize a convex function
- The gradient of this convex function is easy to compute

Note: the weight w(s) correspond to the Kantorovich potential $\psi(x)$
(solves a “discrete Monge-Ampere” equation)

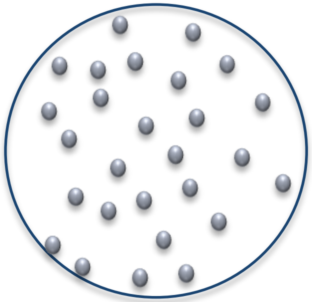
The algorithm:

Summary:

The algorithm computes the weights w_i such that the power cells associated with the Diracs correspond to the preimages of the Diracs.



μ



ν

Part. 3 Optimal Transport – the algorithm

The [AHA] paper summary:

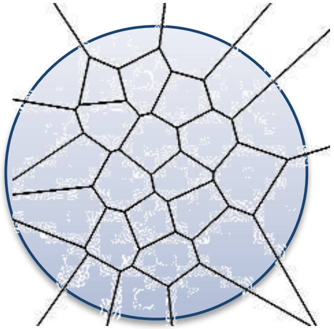
- The optimal weights minimize a convex function
- The gradient of this convex function is easy to compute

Note: the weight w(s) correspond to the Kantorovich potential $\psi(x)$
(solves a "discrete Monge-Ampere" equation)

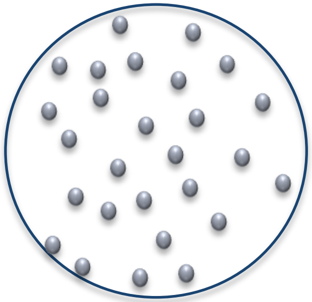
The algorithm:

Summary:

The algorithm computes the weights w_i such that the power cells associated with the Diracs correspond to the preimages of the Diracs.



μ



ν

<u>Wait a minute:</u>

This means that one can move (and possibly deform) a power diagram simply by changing the weights ?

Wait a minute:

This means that one can move (and possibly deform) a power diagram simply by changing the weights ?

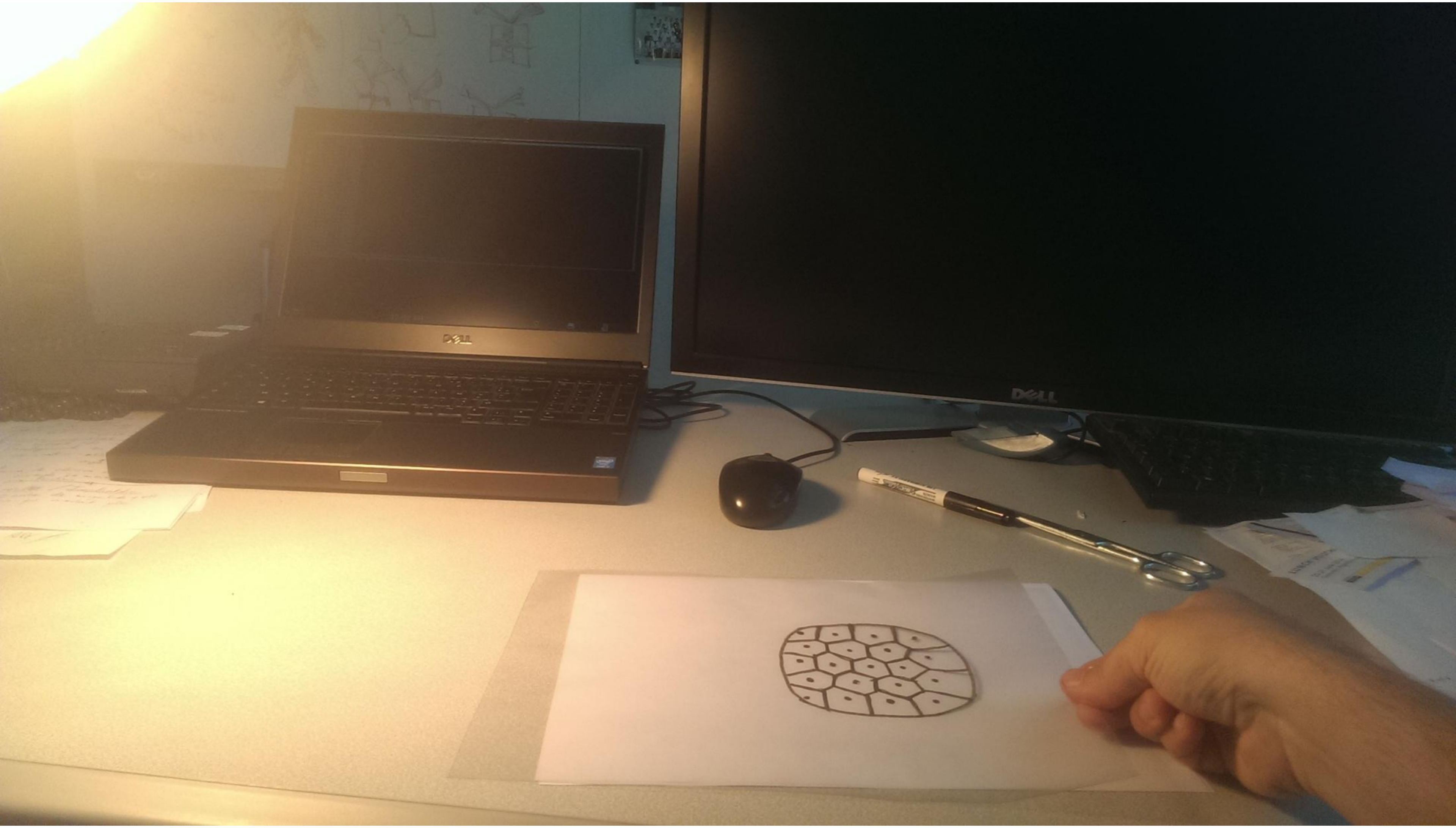
Reminder: Power diagram in 2d = intersection between Voronoi diagram in 3d and $\mathbb{R}^2$

$h_i = \mathrm{sqrt}(w_{max} - w_i)$

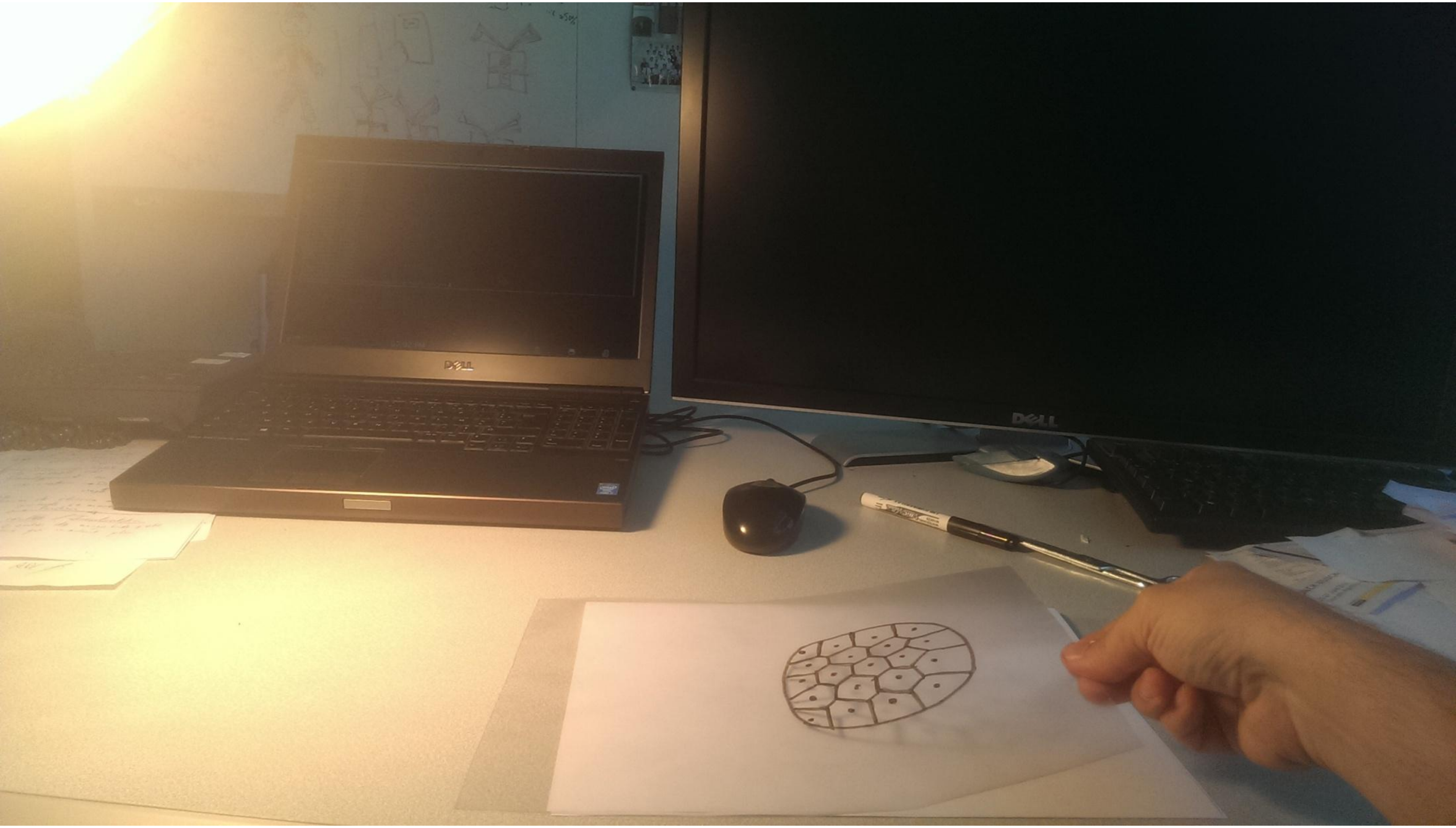
Height of point i

Weight of point i

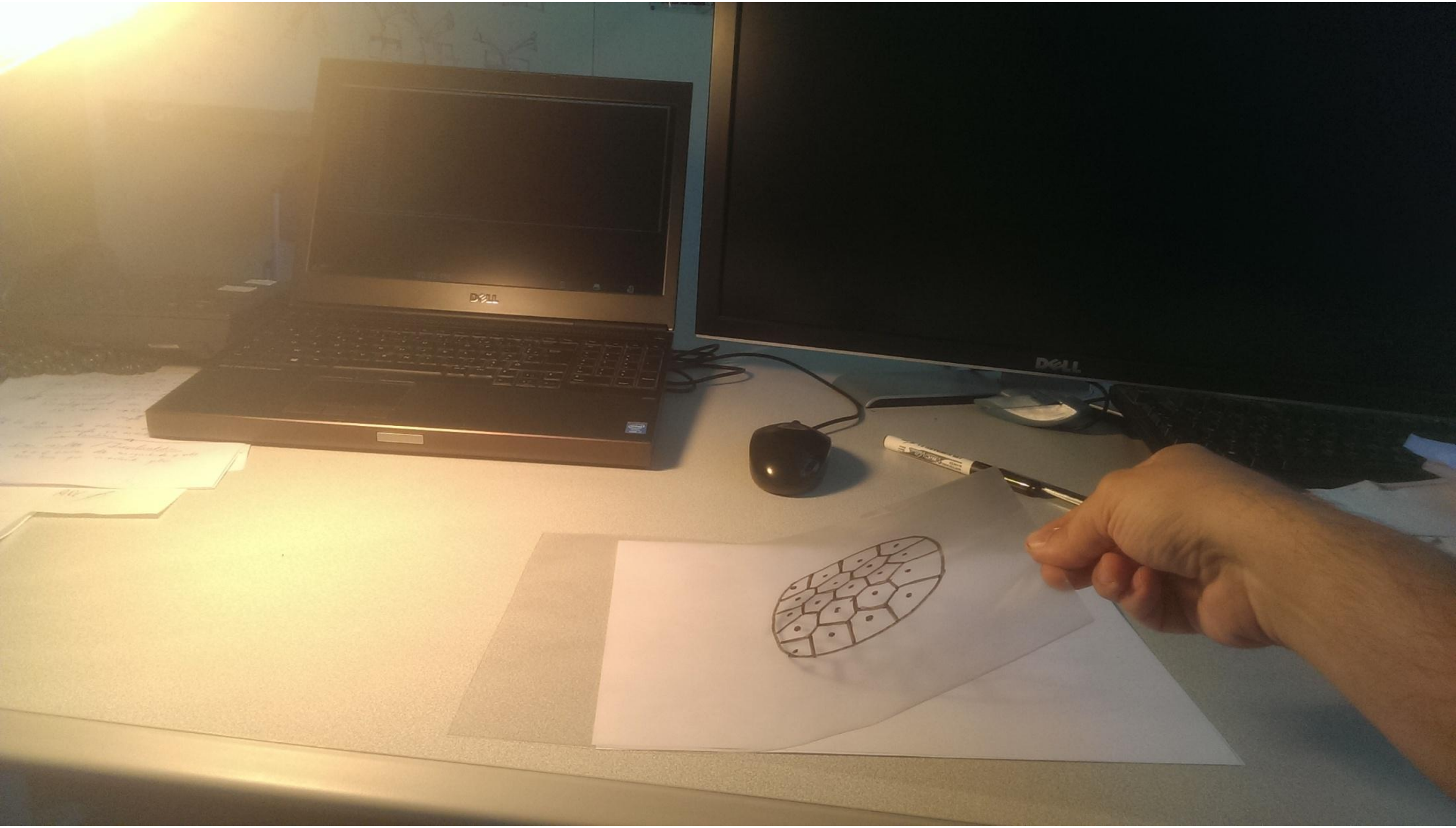
Part. 3 Power Diagrams & Transport



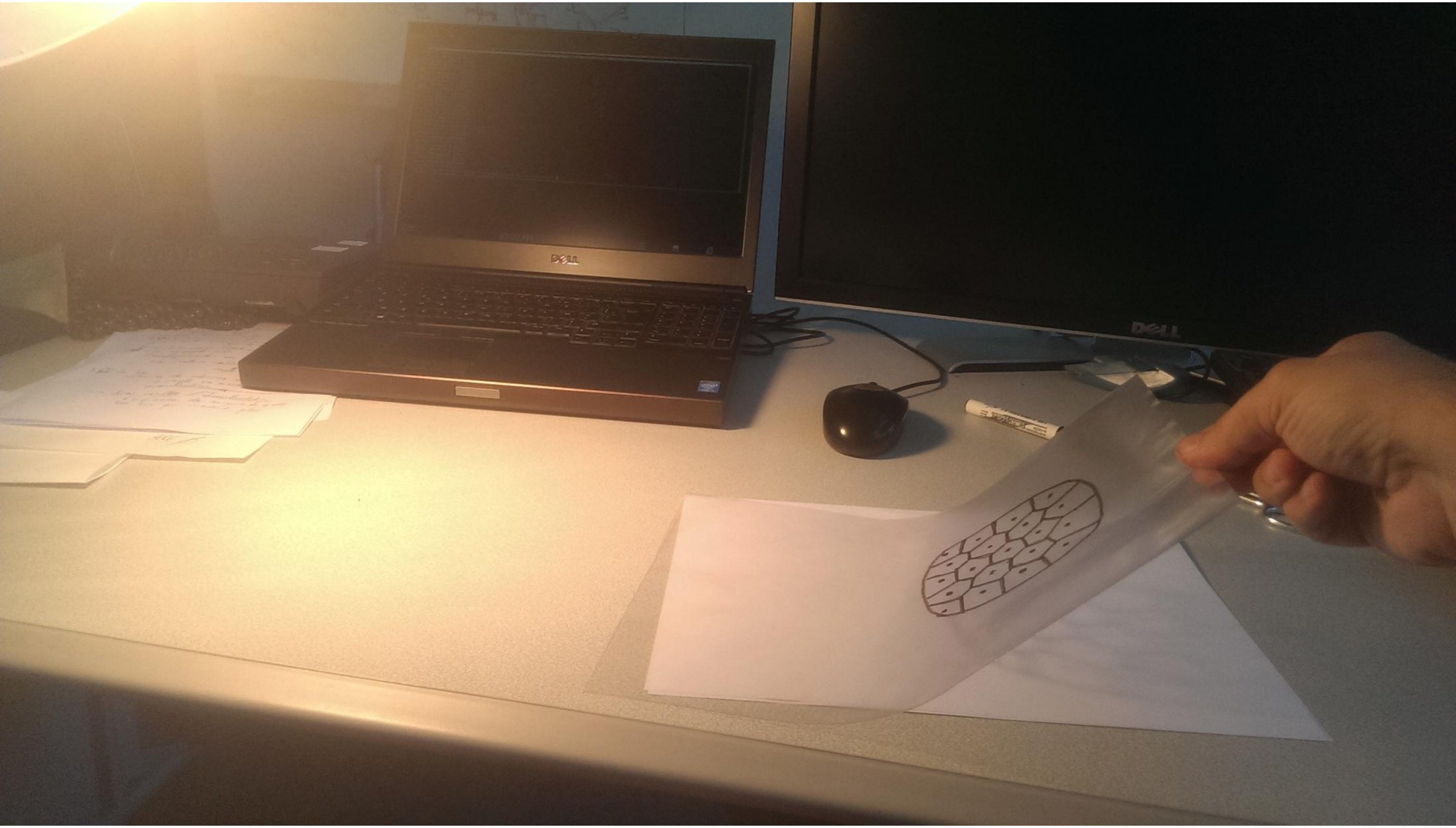
Part. 3 Power Diagrams & Transport



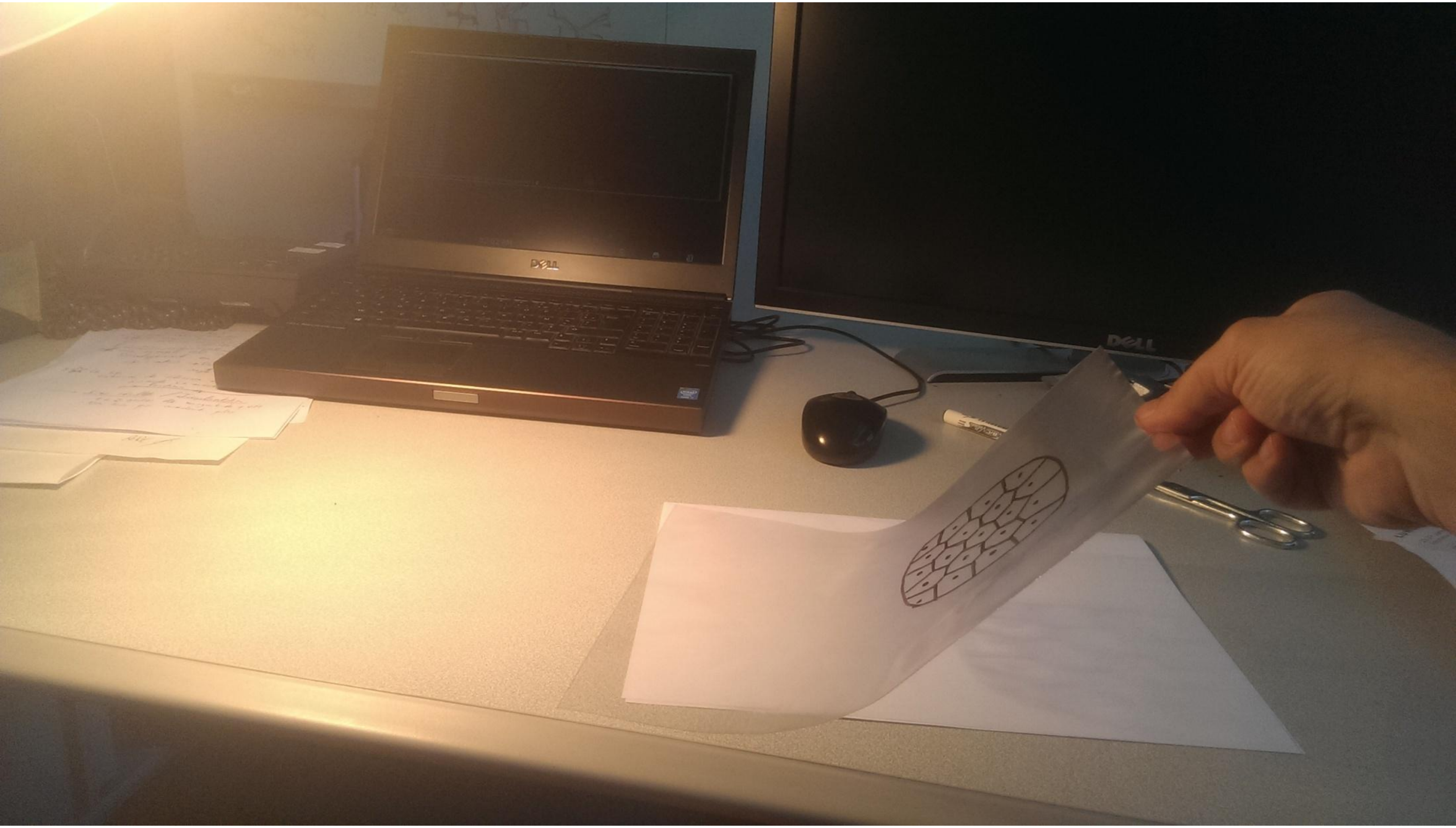
Part. 3 Power Diagrams & Transport



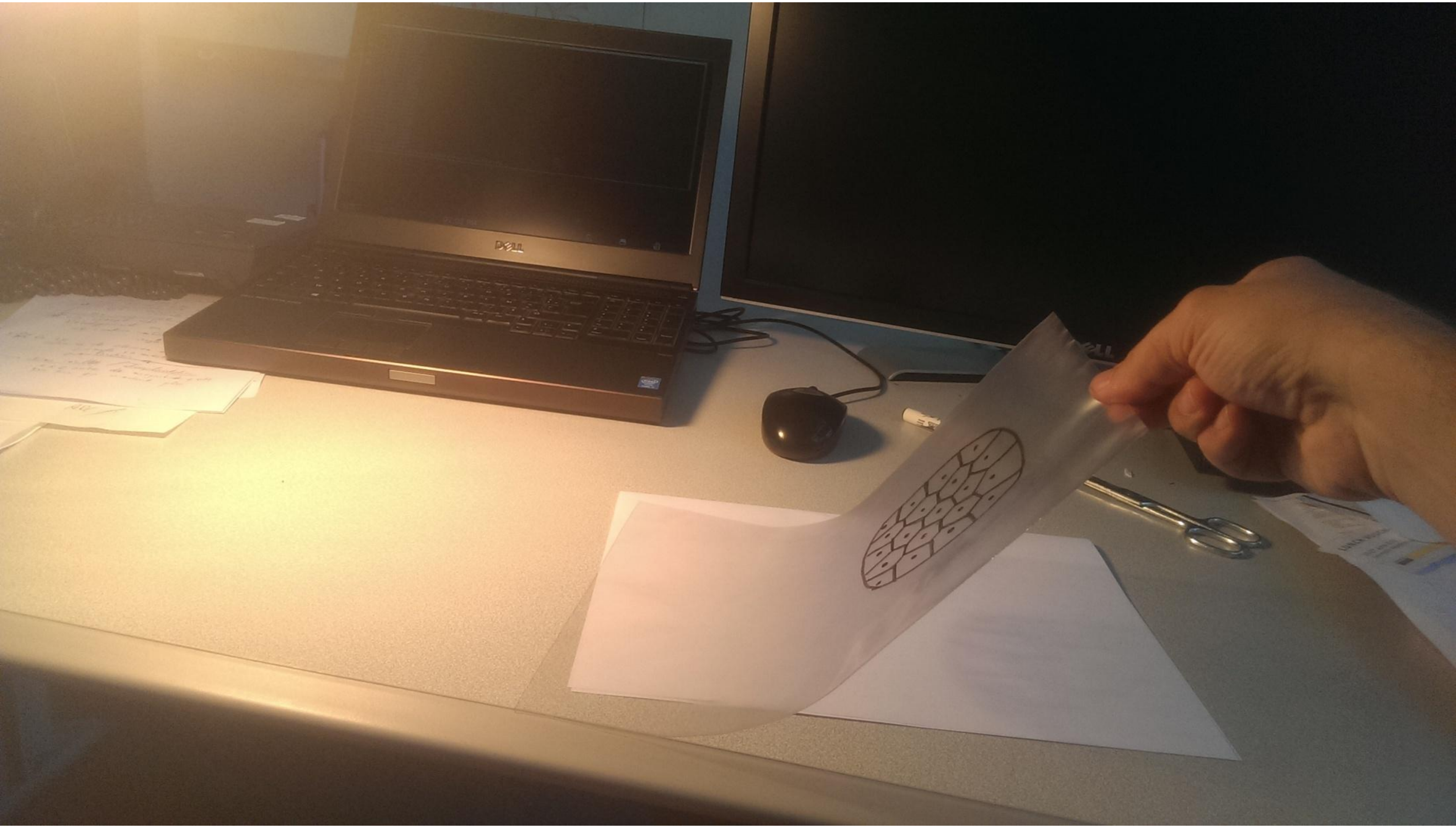
Part. 3 Power Diagrams & Transport



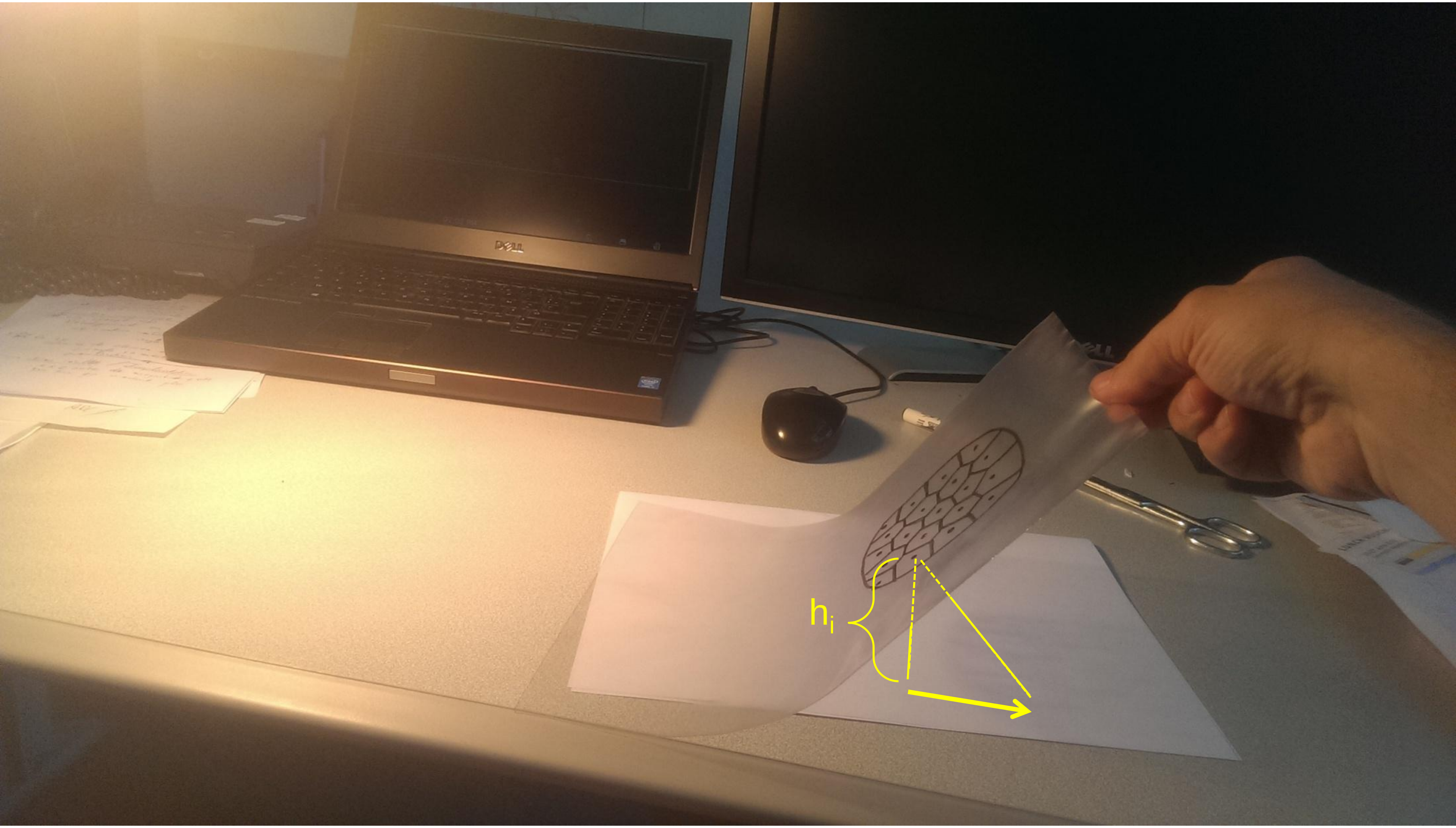
Part. 3 Power Diagrams & Transport



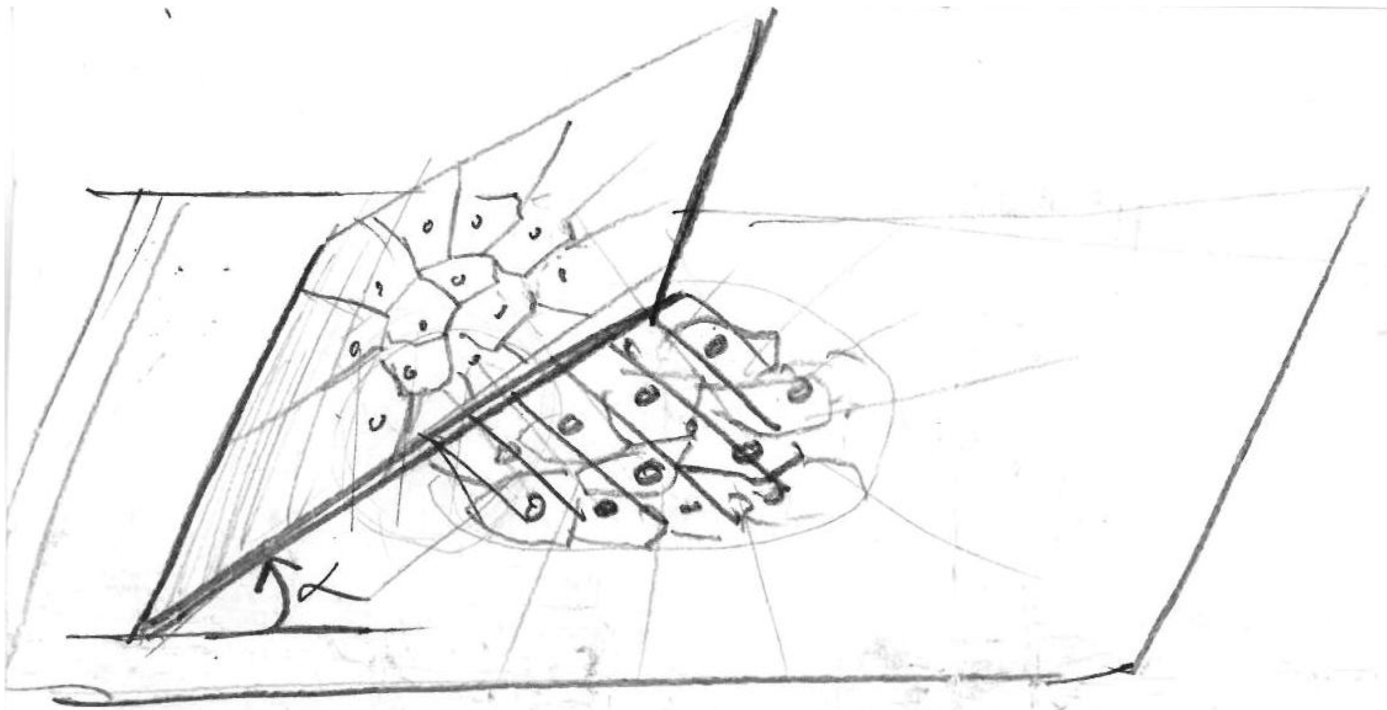
Part. 3 Power Diagrams & Transport



Part. 3 Power Diagrams & Transport



Part. 3 Power Diagrams & Transport

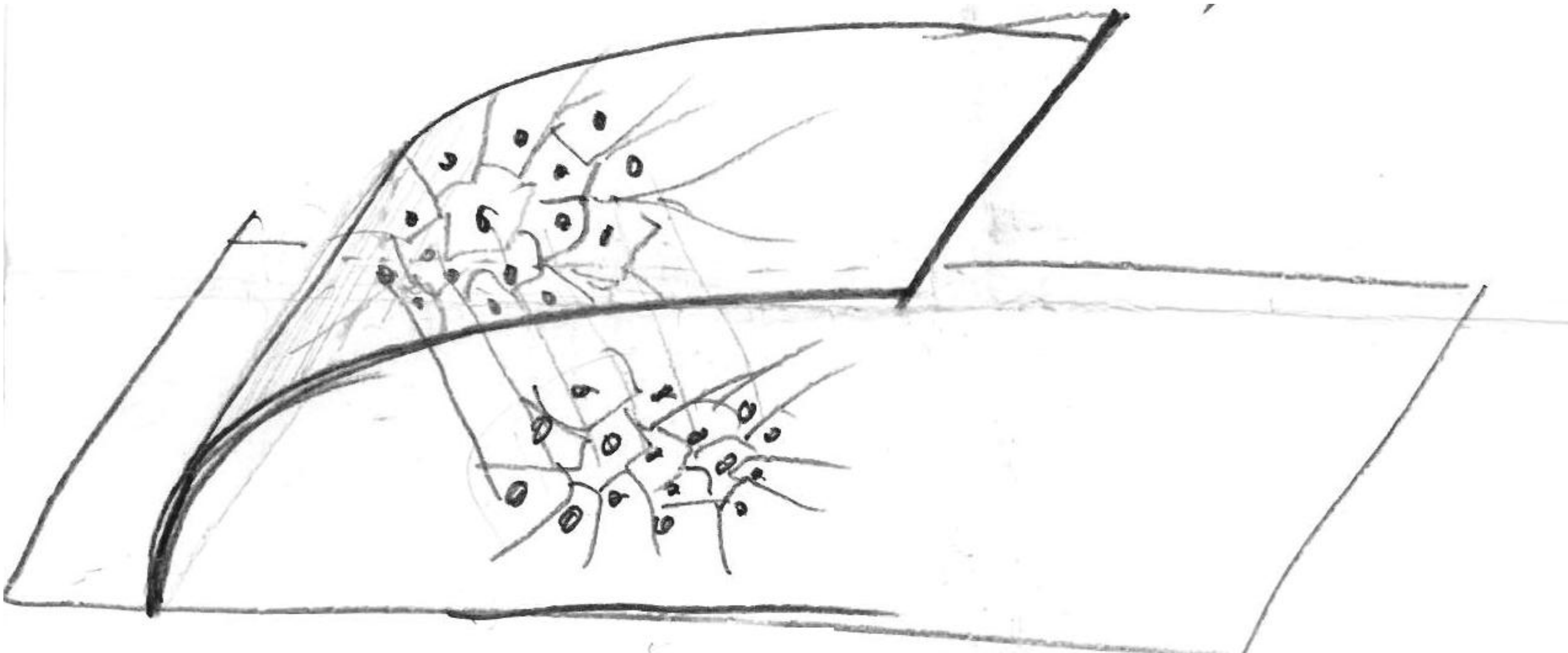


Translating a Voronoi diagram.

1st Try: linear lifting

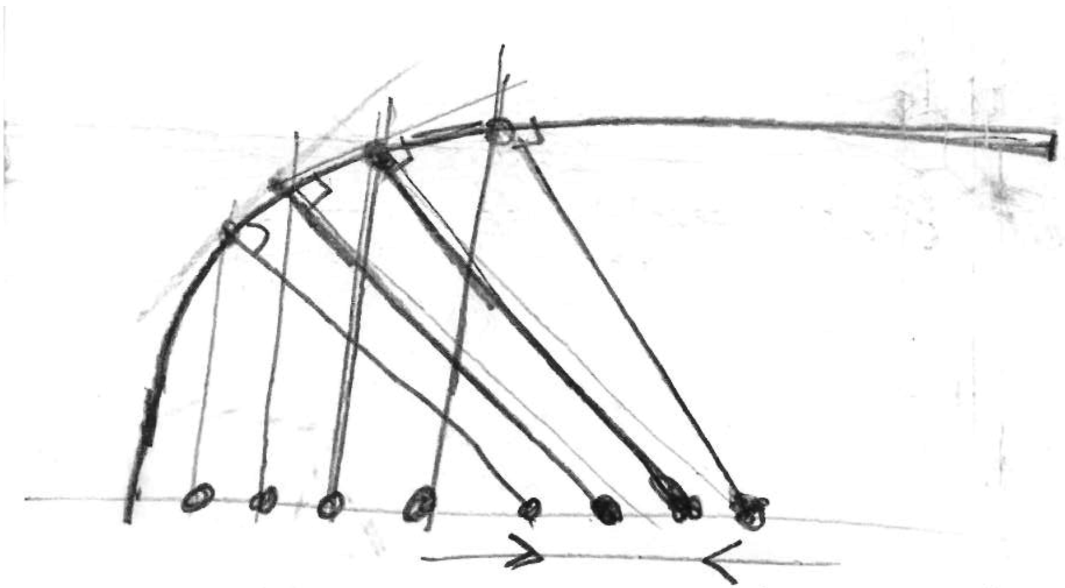
(FAIL: scales by 1/cos(α))

Part. 3 Power Diagrams & Transport



2nd Try : Curved lifting

Part. 3 Power Diagrams & Transport



"converging beams" can compensate the $1/\cos(\alpha)$ expansion by "re-concentrating" the points

Part. 3 Power Diagrams & Transport

(2)

$$d^2(p_i, q) \Big|_{-w_i}^{+h_i^2} < d^2(p_j, q) \overset{+h_i^2}{-w_j} \quad \forall j$$

$$d^2(p_i, q-T) < d^2(p_j, q-T) \quad \forall j$$

$$(p_i - q + T)^2 < (p_j - q + T)^2 \quad \forall j$$

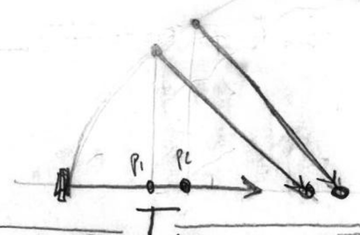
$$d^2(p_i, q) + 2T\cdot(p_i - q) + T^2 < d^2(p_j, q) + 2T(p_j - q) + T^2 \quad \forall j$$

$$d^2(p_i, q) + 2T\cdot p_i < d^2(p_j, q) + 2T\cdot p_j$$

$$w_i \Big|^{h_i^2} = -2T\cdot p_i \quad + cte$$

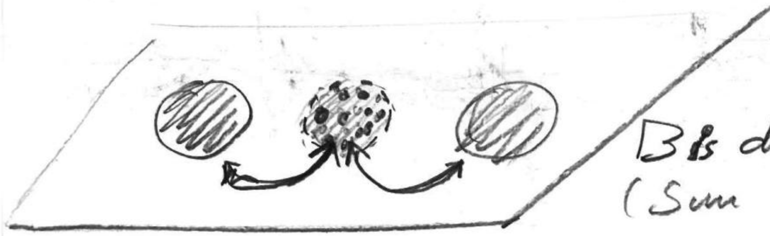
$$h_i^2 = (2T\cdot p_i + Cte)$$

$$\boxed{h_i = \sqrt{2(T\cdot p_i - \min_i(T\cdot p_i))}}$$

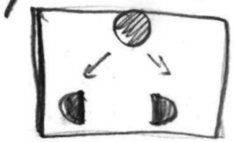


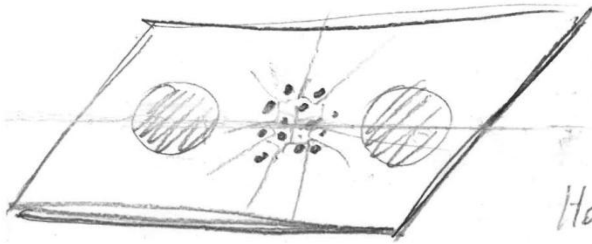
Translation d'un diagramme de Voronoi sectionnel - Relèvement en racine carrée -

Part. 3 Power Diagrams and Transport

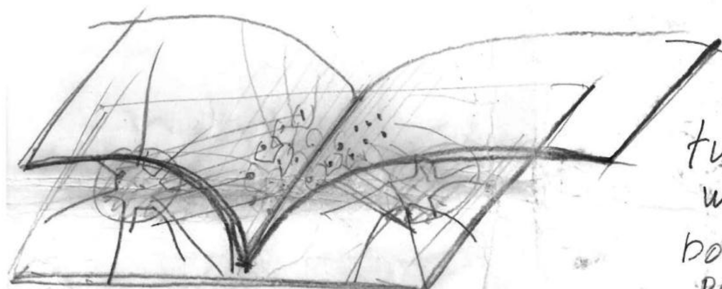


B is discretized (Sum of Diracs)





Voronoi diagram of B samples - How to "back displace" it onto A?



Lifting on two "square root wings" translates both halves of B points into the two blobs of A

Solving for the OTM ($T(x,y)$ vector field) is equivalent to solve for the "square root wings" ($h(x,y)$ scalar function) → Simpler, Unconstrained

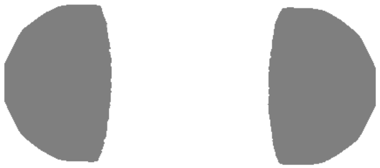
Ref - Name of eqn.

Part. 3 Optimal Transport – 2D example

Numerical Experiment: *Splitting a disk*

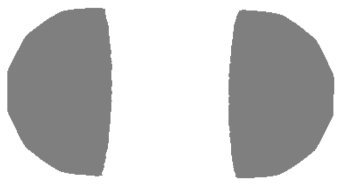
Part. 3 Optimal Transport – 2D example

Numerical Experiment: *Splitting a disk*



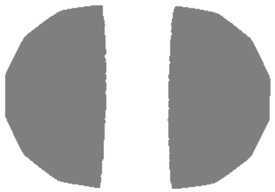
Part. 3 Optimal Transport – 2D example

Numerical Experiment: *Splitting a disk*



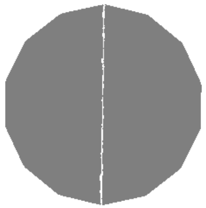
Part. 3 Optimal Transport – 2D example

Numerical Experiment: *Splitting a disk*



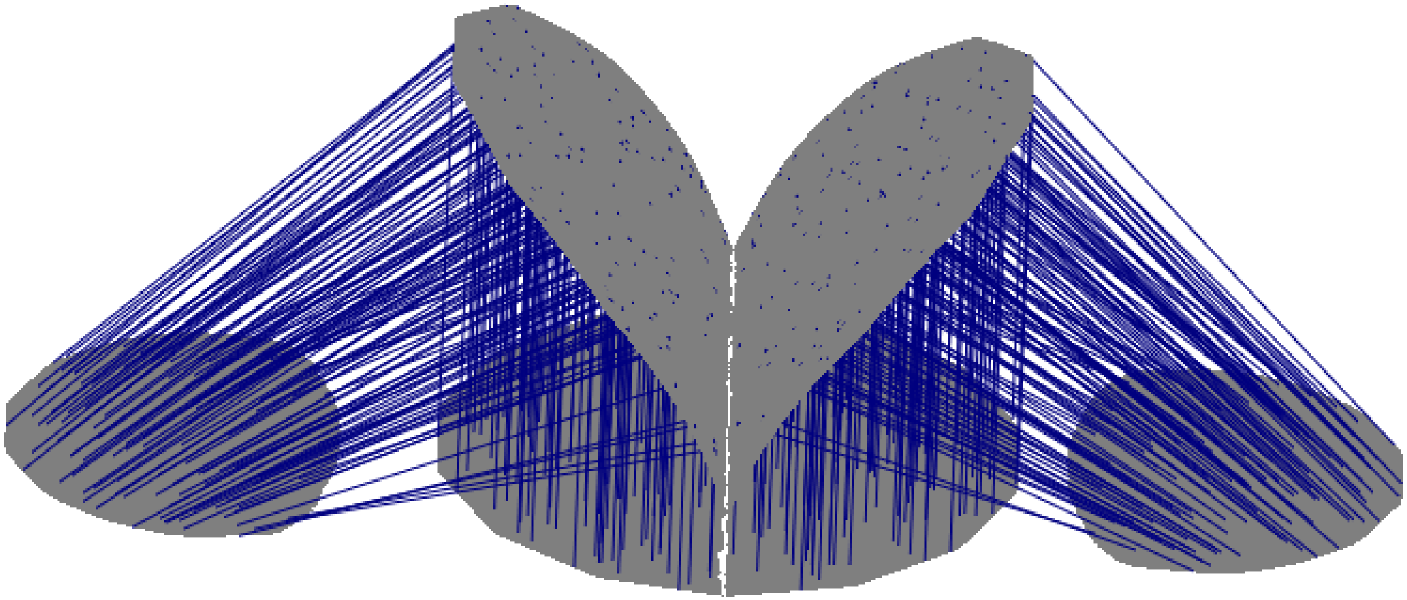
Part. 3 Optimal Transport – 2D example

Numerical Experiment: *Splitting a disk*

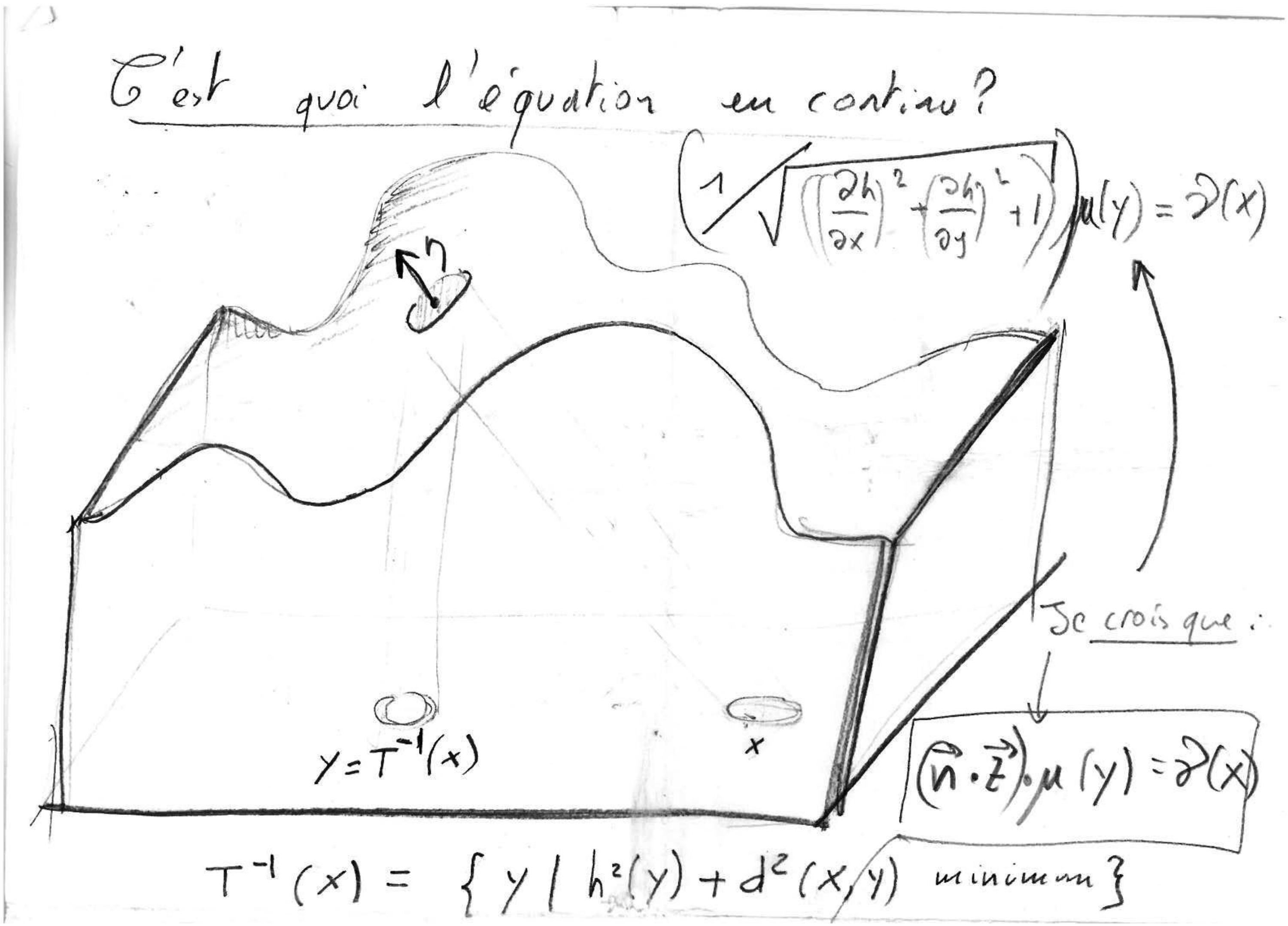


Part. 3 Optimal Transport – 2D example

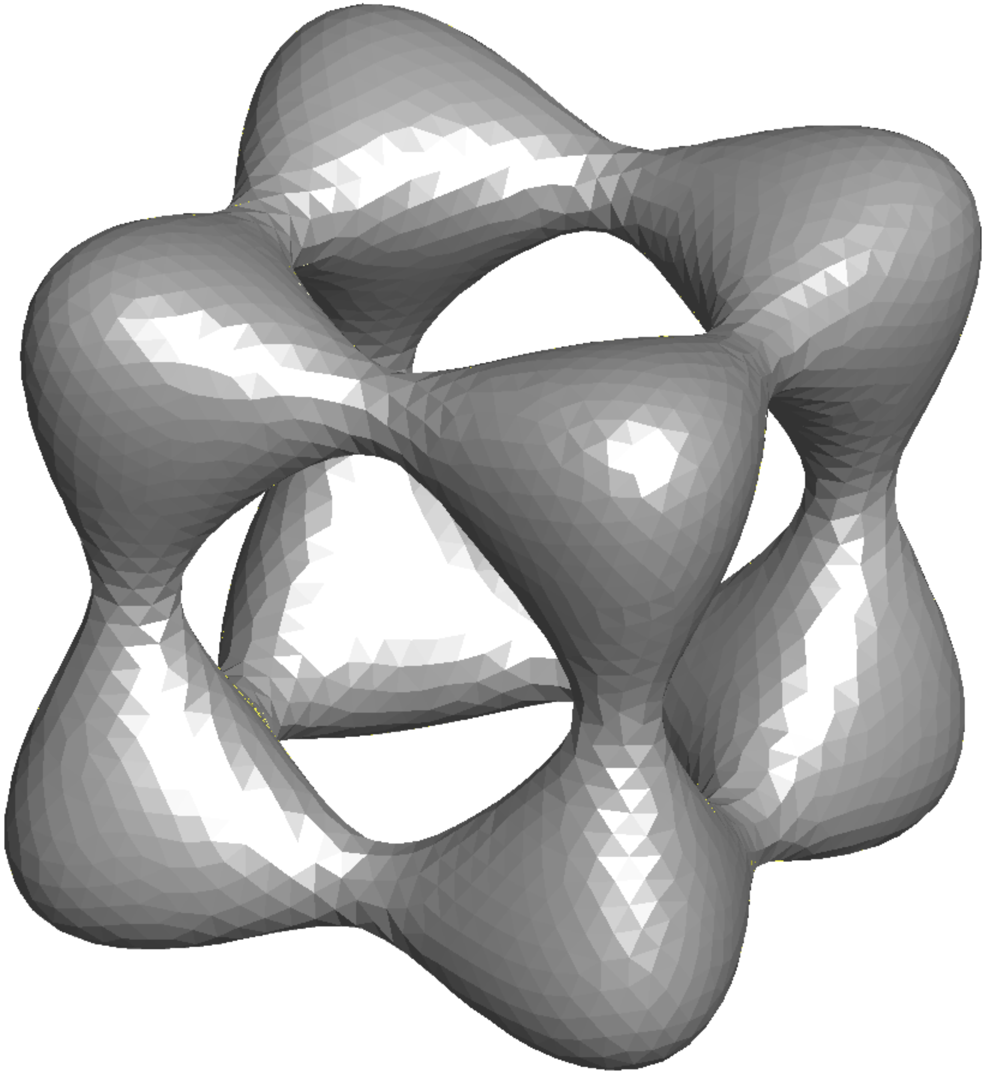
Numerical Experiment: *Splitting a disk*



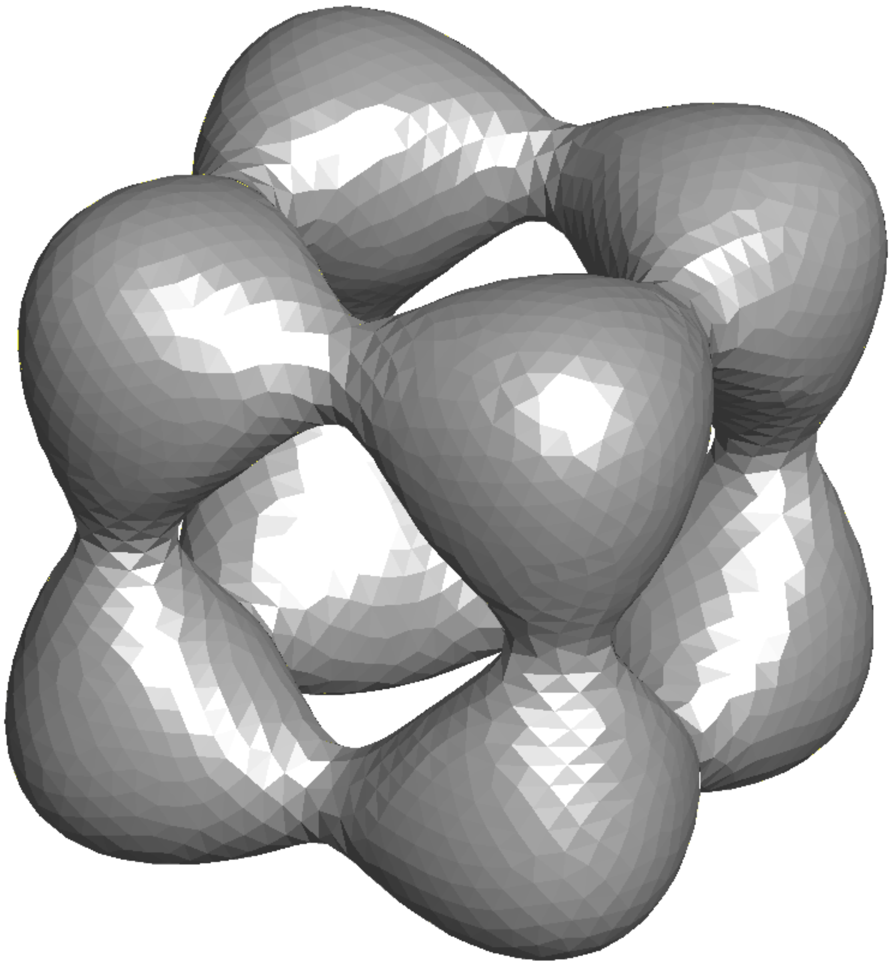
Part. 3 Power Diagrams & Transport



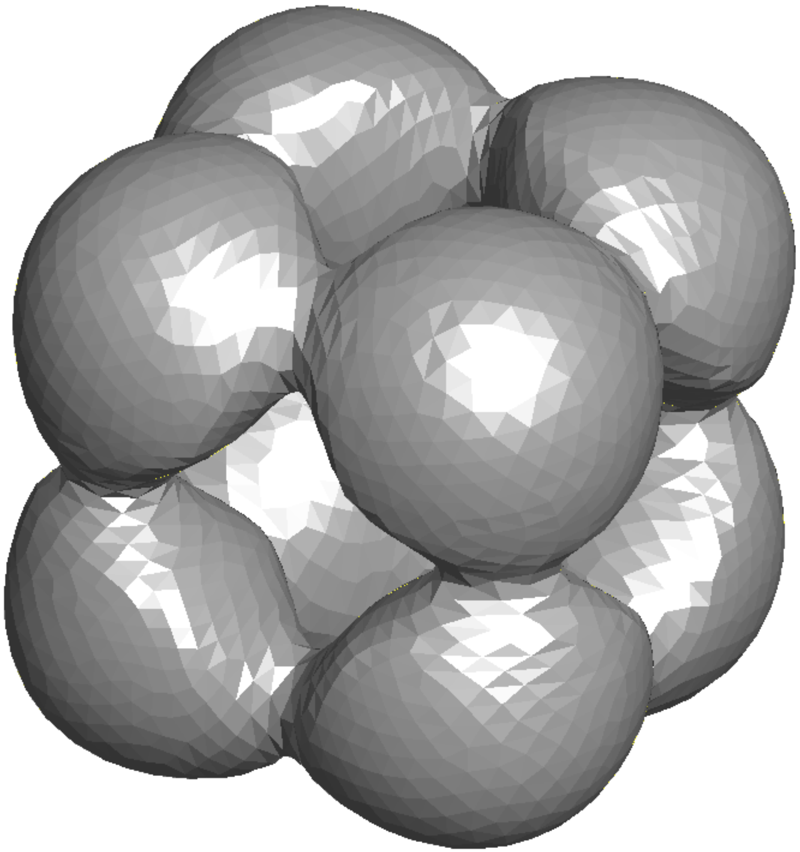
Part. 3 Towards FWD (Fast Wasserstein Distance)



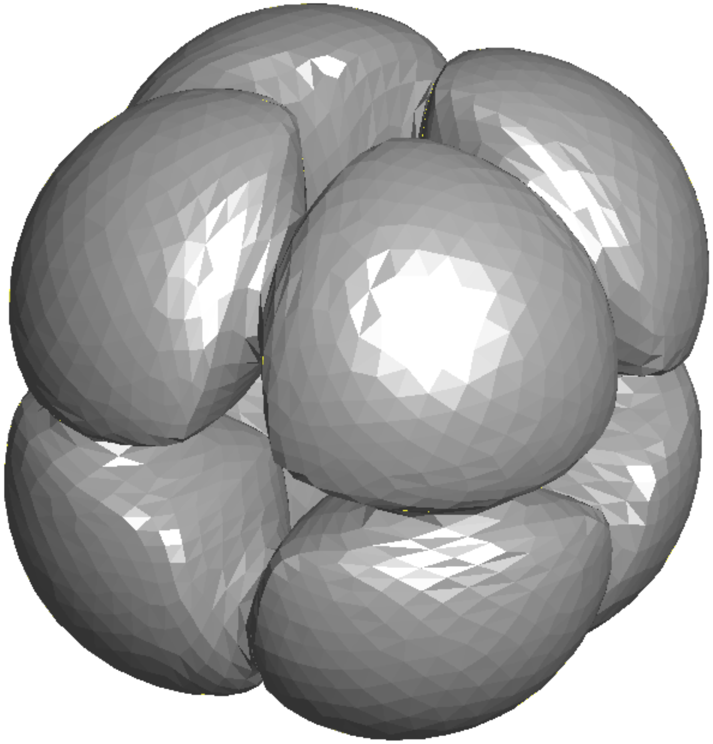
Part. 3 Towards FWD (Fast Wasserstein Distance)



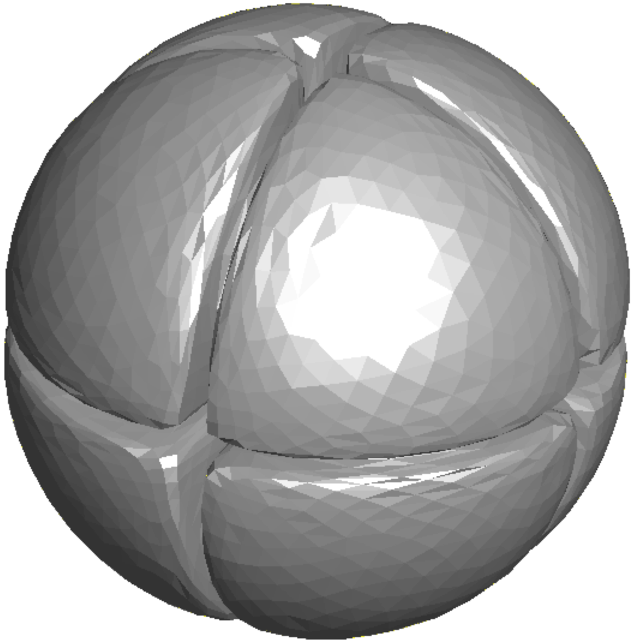
Part. 3 Towards FWD (Fast Wasserstein Distance)



Part. 3 Towards FWD (Fast Wasserstein Distance)



Part. 3 Towards FWD (Fast Wasserstein Distance)



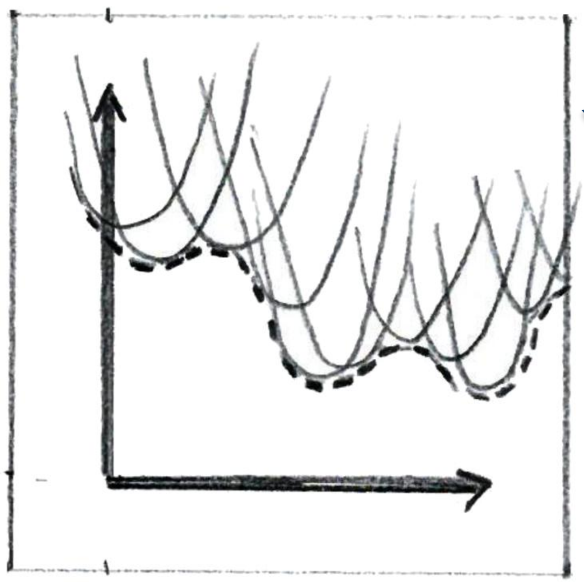
Part. 3 Relation with Vector Quantization

Observation 8. *The quantization noise power $\hat{Q}(\hat{Y})$ computed in $\mathbb{R}^{d+1}$ corresponds to the term $f_{T_W}(W)$ of the function maximized by the weight vector that defines a semi-discrete optimal transport map plus the constant $w_M \mu(\Omega)$.*

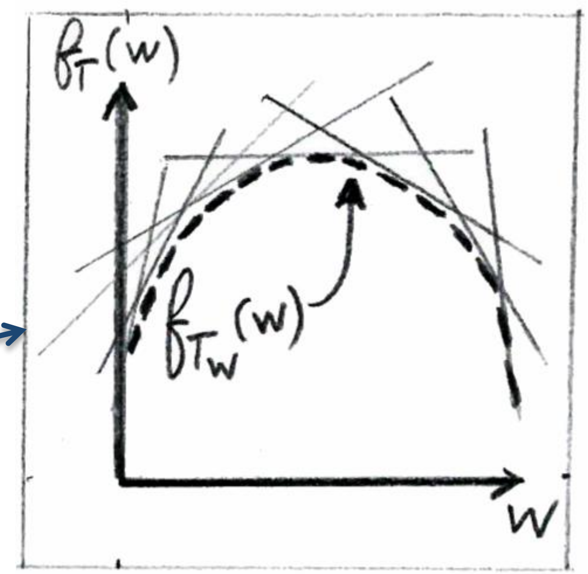
Proof.

$$\begin{aligned}
\hat{Q}(\hat{Y}) &= \sum_i \int_{\mathrm{Vor}(\hat{y}_i)\cap\mathbb{R}^d} \|\hat{x} - \hat{y}_i\|^2 d\mu \\
&= \sum_i \int_{\mathrm{Pow}_W(y_i)} \|x - y_i\|^2 - w_i + w_M d\mu \\
&= f_{T_W}(W) + w_M \mu(\Omega)
\end{aligned}$$

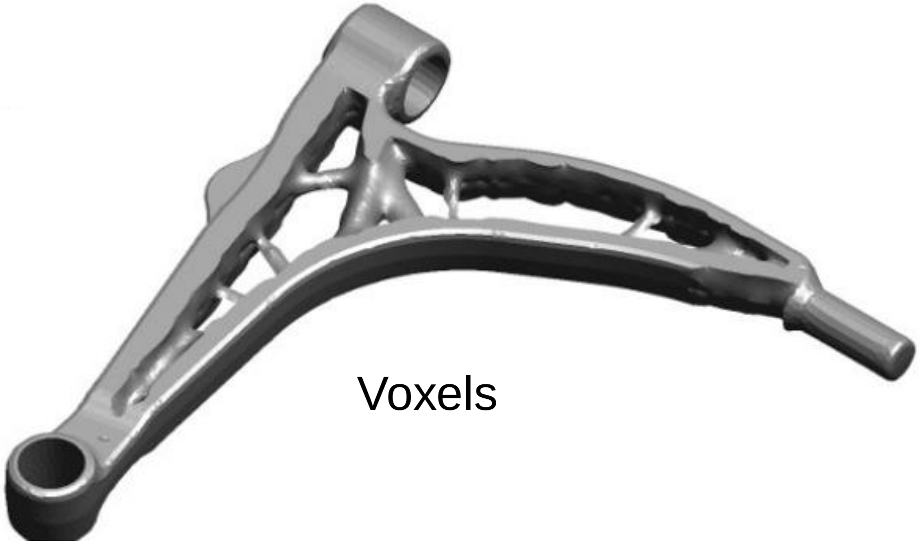
$\square$

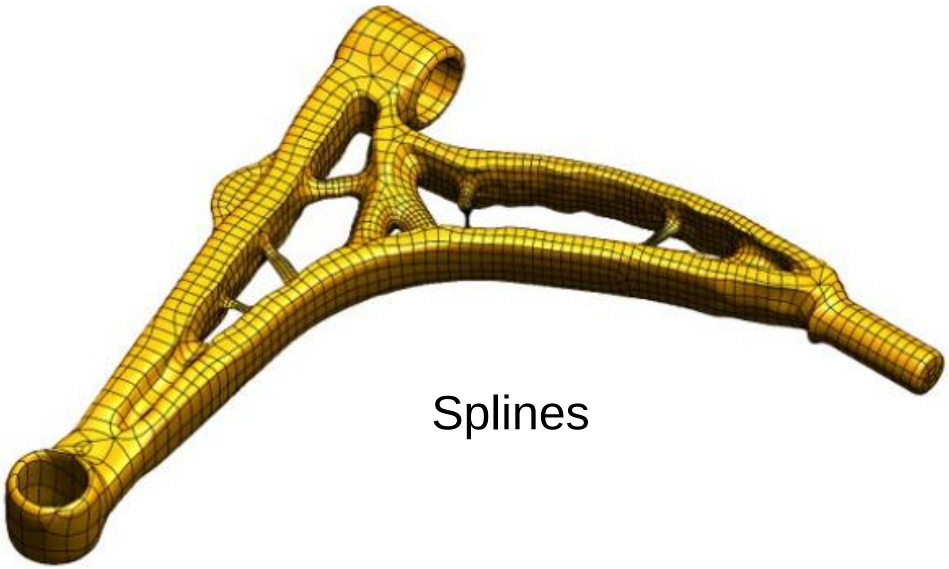


$$\hat{y}_i = \begin{pmatrix} \boxed{\begin{matrix} y_{i,1} \\ y_{i,2} \\ \vdots \\ y_{i,d} \end{matrix}} \\ \boxed{h_i = \sqrt{w_M - w_i}} \end{pmatrix}$$



Part. 3 Self Organizing Optimal Transport Maps





4

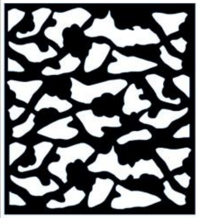
Future Works

Part. 4 Future Works in Fabrication

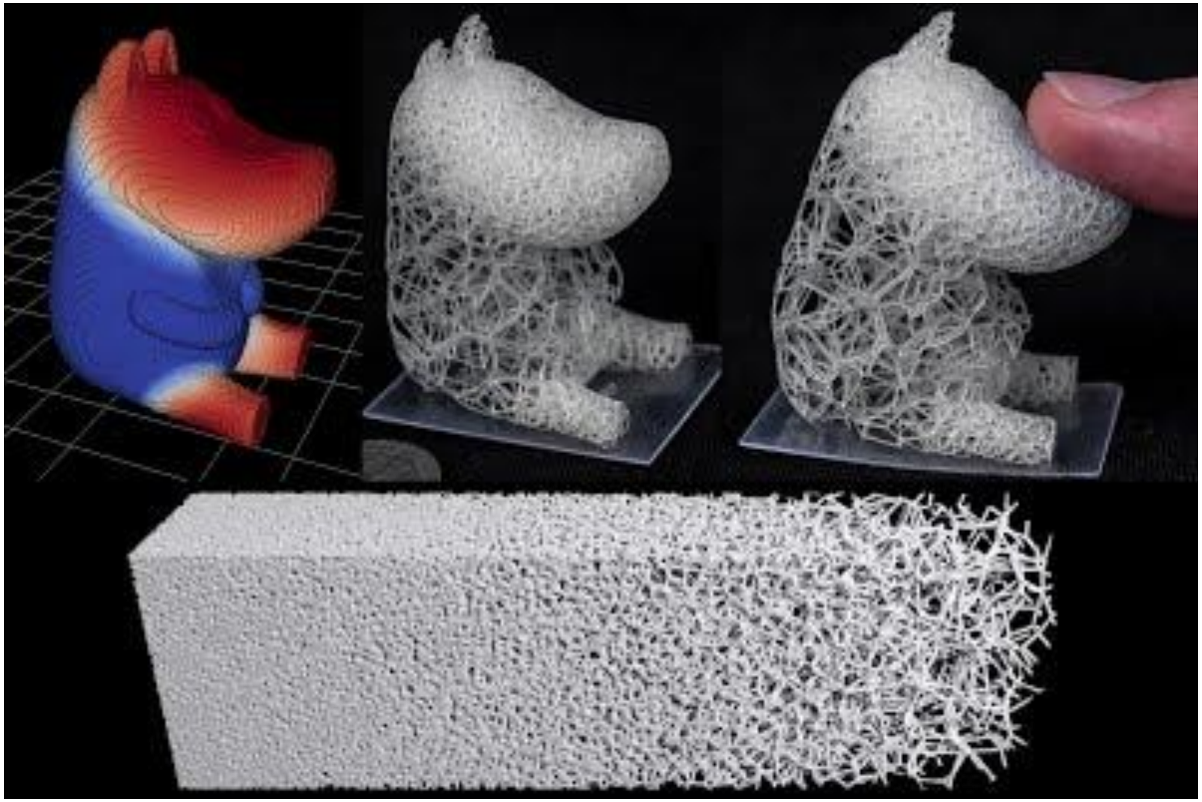
Guiding principles:

(1) Make it easy for everybody !

(2) Integrate more and more fabrication constraints in modeling



Part. 4 Future Works in Fabrication

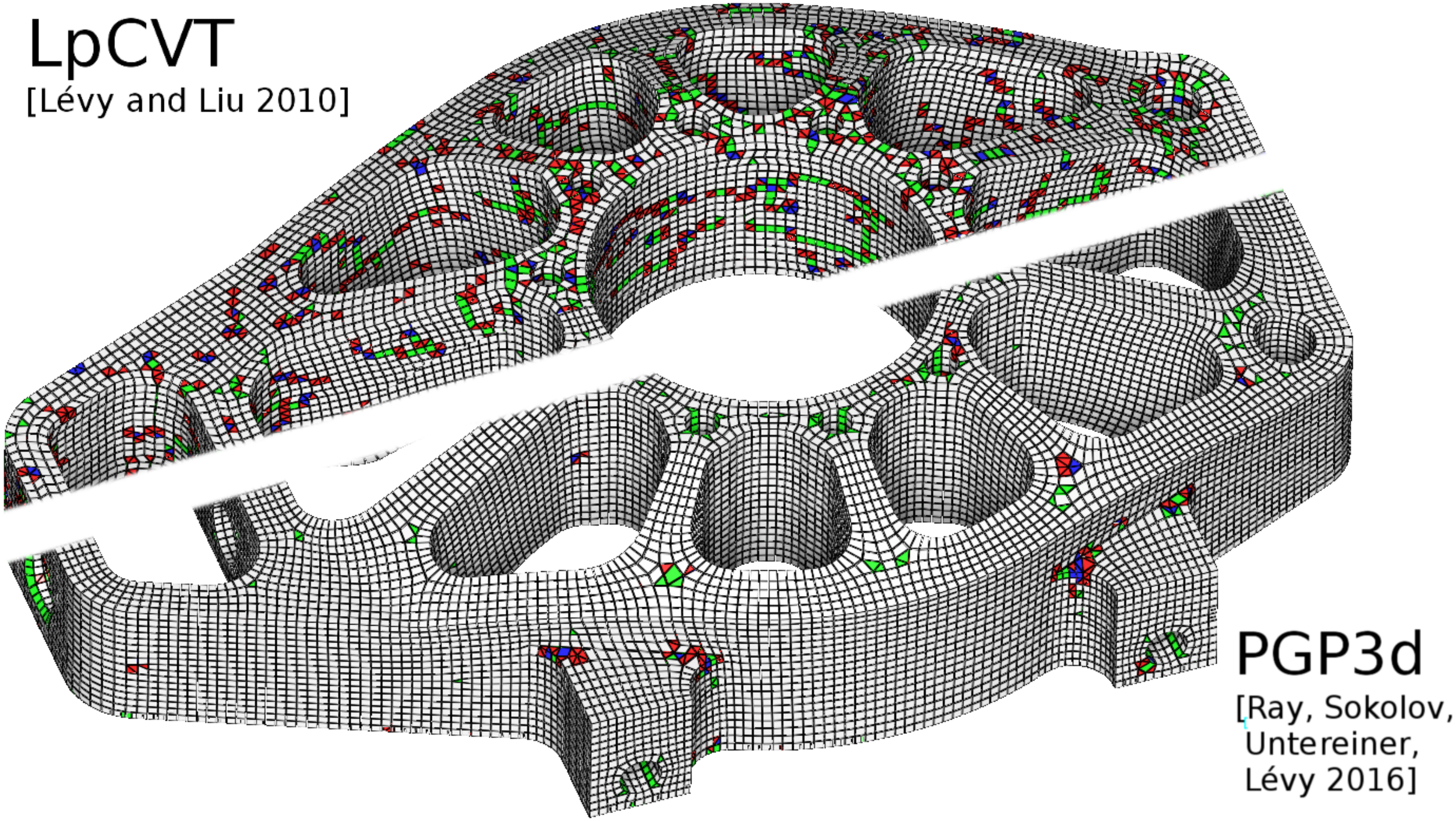


[ACM SIGGRAPH 2016]

Part. 4 Future Works in Applied Mathematics

Discrete Elements – from Equations to Programs

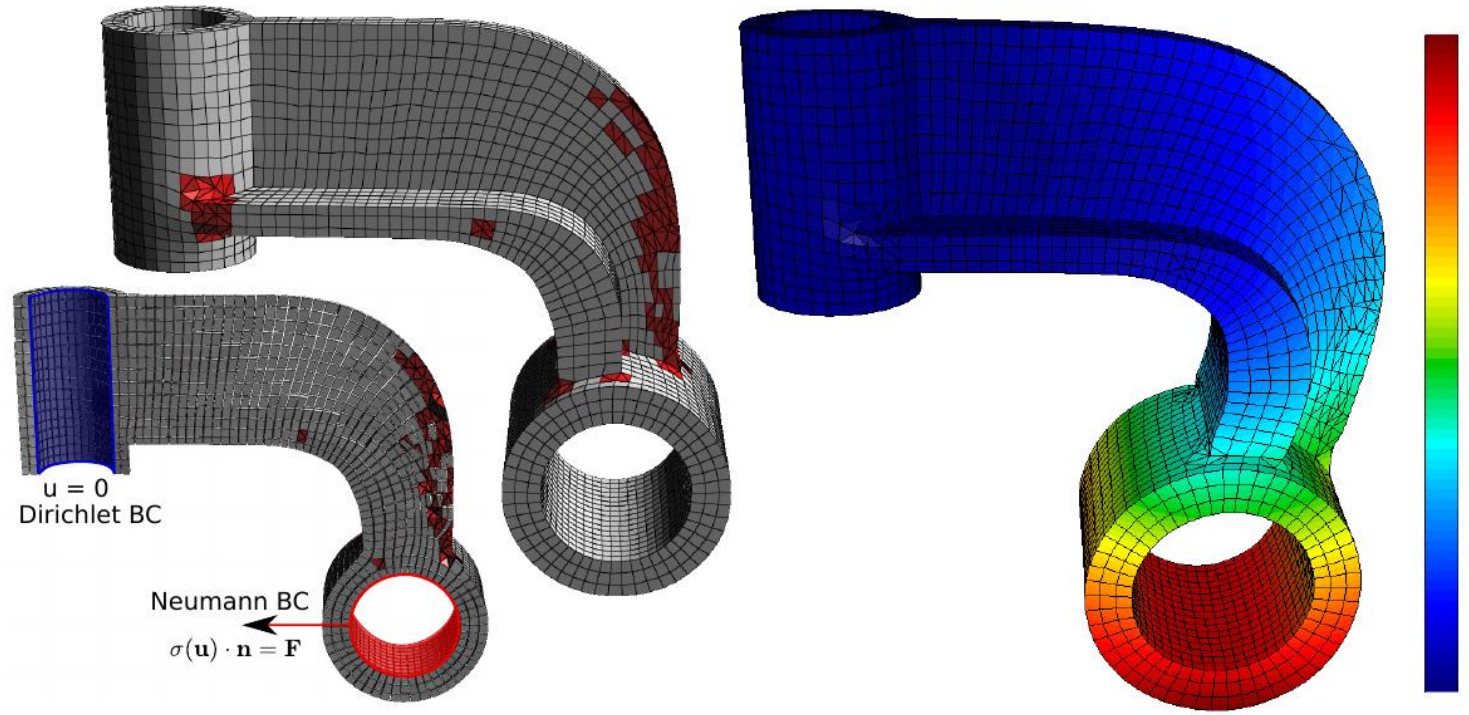
Short term: ***Hex-dominant meshing***



Part. 4 Future Works in Applied Mathematics

Discrete Elements – from Equations to Programs

*Short term: **Hex-dominant meshing***

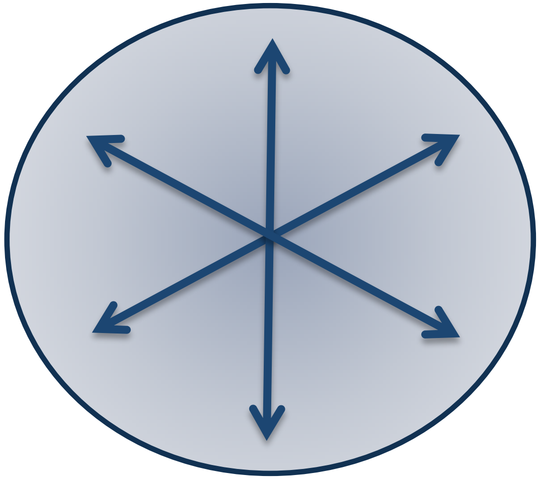


Finite Elements function basis for non-conforming meshes (submitted)

Part. 4 Future Works in Applied Mathematics

Optimization of frame fields for hex-dominant meshing

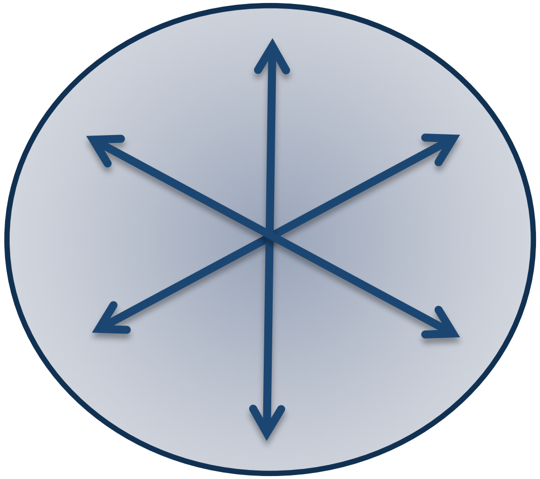
How to interpolate frame fields ?



Part. 4 Future Works in Applied Mathematics

Optimization of frame fields for hex-dominant meshing

How to interpolate frame fields ?



A natural idea:

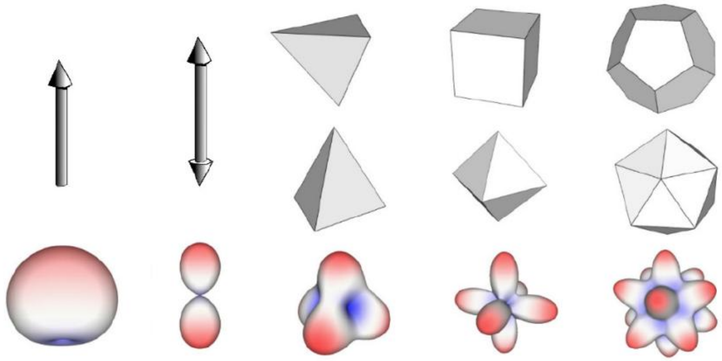
Frame field = 8 Dirac masses on the shere

Optimal Transport for interpolation, barycenters …

Part. 4 Future Works in Applied Mathematics

Optimization of frame fields for hex-dominant meshing

How to interpolate frame fields ?



A natural idea:

Frame field = 8 Dirac masses on the shere

Optimal Transport for interpolation, barycenters …

Not smooth enough

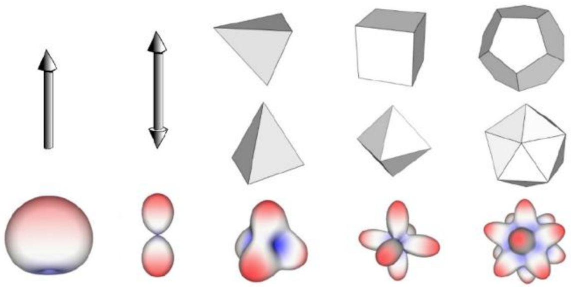
Use “smoothed” version, with functions that has the same symmetries.

Symmetries of platonic solids reproduced with sums of Spherical Harmonics.

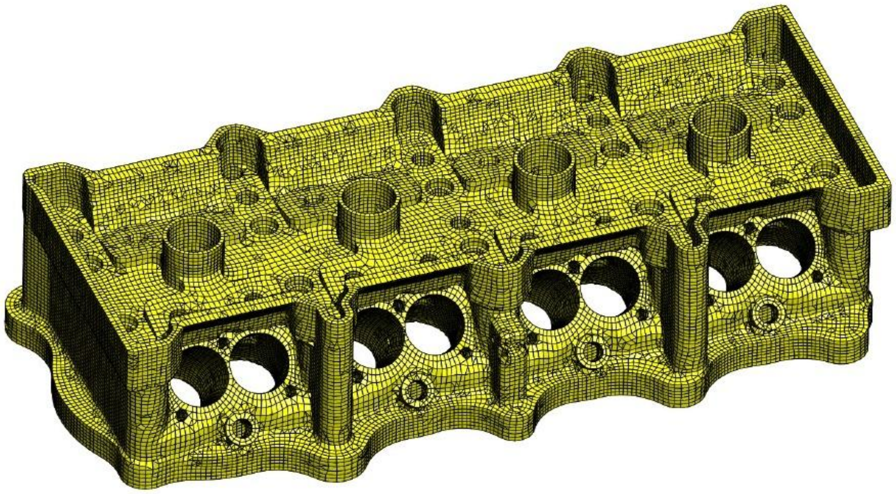
Part. 4 Future Works in Applied Mathematics

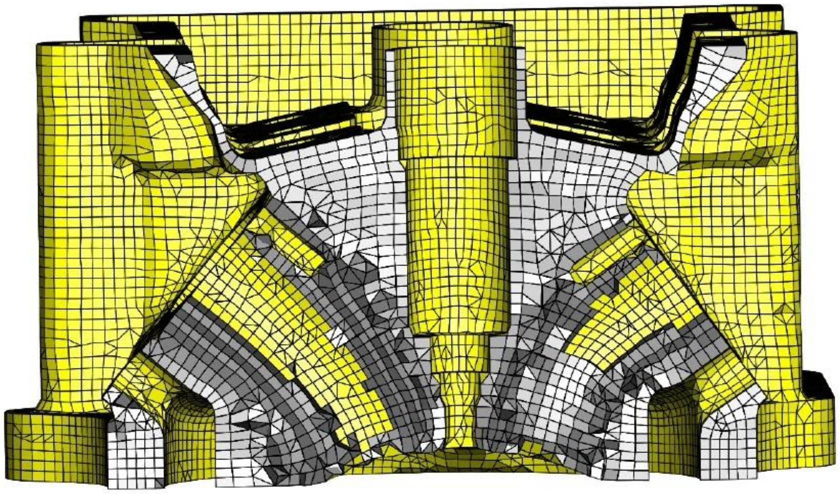
Optimization of frame fields for hex-dominant meshing

How to interpolate frame fields ?



First results are encouraging
(scales-up well)





[ACM Transactions on Graphics 2016]

Part. 4 Future Works in Applied Mathematics

*Longer term**:** from the principle of least action to optimal transport*

JKO scheme (Jordan, Kinderlehrer, Otto)
Benamou, Carlier, Merigot, Oudet arXiv 1408.4536

EXPLORAGRAM project (INRIA exploratory project)
MAGA project (ANR project – submitted)

Part. 4 Future Works in Applied Mathematics

Geometric Predicates: *How can we easily translate* geometric predicates *into computer programs ? How can we* certify *their validity ? Can we invent* programming tools *?*

Source PCK file (using my current version)

```
Sign side2(
    point p0, point p1, point p2,
    point q0, point q1
) {

    scalar l1 = sq_dist(p1,p0) ;
    scalar l2 = sq_dist(p2,p0) ;

    scalar a10 = 2*dot_at(p1,q0,p0);
    scalar a11 = 2*dot_at(p1,q1,p0);
    scalar a20 = 2*dot_at(p2,q0,p0);
    scalar a21 = 2*dot_at(p2,q1,p0);

    scalar Delta = a11 - a10 ;
    scalar DeltaLambda0 = a11 - l1 ;
    scalar DeltaLambda1 =  l1 - a10 ;
    scalar r =
        Delta*l2-a20*DeltaLambda0-a21*DeltaLambda1 ;

    Sign Delta_sign = sign(Delta) ;
    Sign r_sign     = sign(r) ;

    generic_predicate_result(Delta_sign*r_sign) ;

    begin_sos3(p0,p1,p2)
       sos(p0, Sign(Delta_sign*sign(Delta-a21+a20)))
       sos(p1, Sign(Delta_sign*sign(a21-a20)))
       sos(p2, NEGATIVE)
    end_sos
}
```

Source PCK file (using the tools that I plan to develop)

```
Sign side2(point p0, point p1, point p2, point q0, point q1) {

    scalar w0 = 0.0;
    scalar w1 = 0.0;
    scalar w2 = 0.0;

    sos_perturbation(wi, pi, pow(epsilon,i));

    Plane P1 = weighted_bisector(p0,w0,p1,w1);
    Point q = intersection(P1, segment(q0,q1));

    return Sign(sq_dist(q,p0) + w0 - sq_dist(q,p1) - w1);
}
```

sqrt(), root_of() … Voronoi diagram of Segments in 3d doable ?

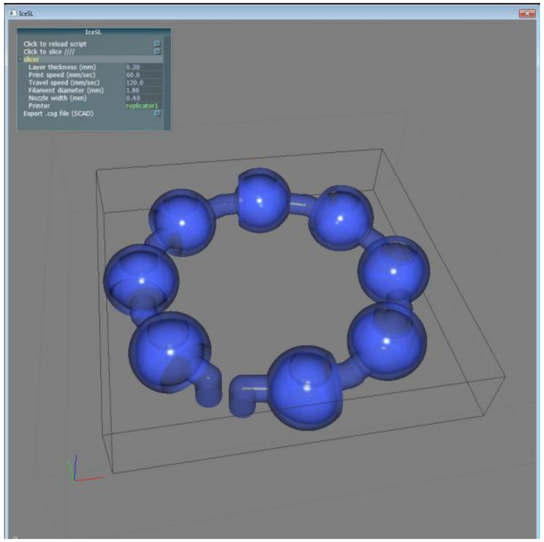
Highlights

ERC StG GOODSHAPE – Optimal Sampling

ERC PoC VORPALINE – Remeshing Software

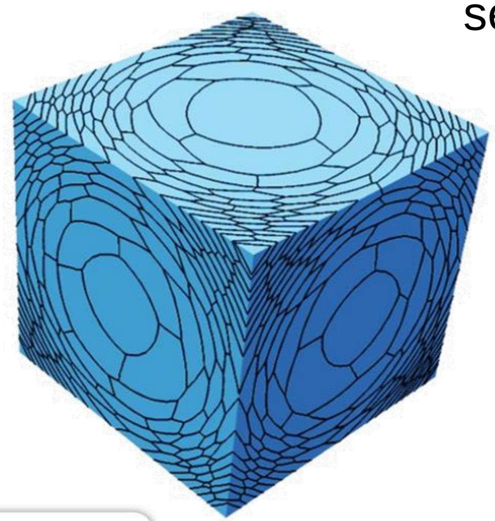
ERC StG SHAPEFORGE – 3D printing made easy

ERC PoC ICEXL – 3D printing – scaling up

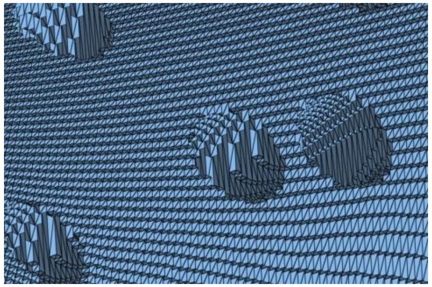


IceSL software – Fast CSG modeler, language, driver for 3d printers …

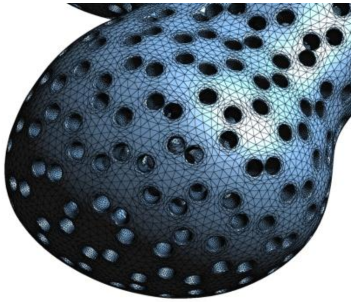
First algorithm that computes aniso. Voro. diagram and
semi-discrete Optimal Transport in 3d (+ Predicate Cons. Kit)



Integration of research results in ALICE !



SHAPEFORGE - Dexels



GOODSHAPE/VORPALINE

Thank you !