Cage-based deformations for implicit surfaces

Kevin Trancho⋆
Master 2 student in Computer sciences

at University Paris-Est Marne-la-Vallée
in Internship supervised by Loïc Barthe† and Pascal Romon‡

July 08, 2019

⋆
⋆ †
‡
⋆ †
⋆ ‡

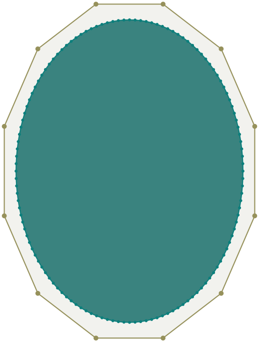
Topics of the presentation :

- Introduction to cage-based deformations.
  - Cages and interest as deformation.
  - Generalized barycentric coordinates in cages for deformation.
- How to deform an implicit surface with a cage.
  - Inverse position problem.
  - State of the art : Free-Form Deformations special type of cages and implicit surface deformation.
  - Our methods and flexible solving architecture.
  - Cage auto-intersection field-operator solving trials.
- Future work and interest for Implicit skinning.

Cage on surface

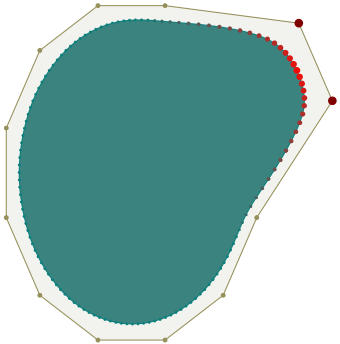
- Bounding simplification of a discrete **surface**.
- Define **control positions**.
- Allow **smooth deformations** of the surface.

Cage on surface



- Bounding simplification of a discrete **surface**.
- Define **control positions**.
- Allow **smooth deformations** of the surface.

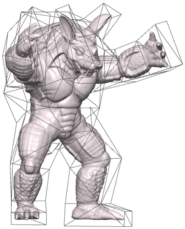
Cage on surface



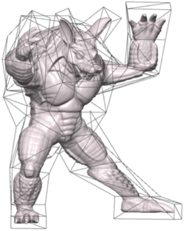
- Bounding simplification of a discrete **surface**.
- Define **control positions**.
- Allow **smooth deformations** of the surface.

Cage interest for animation

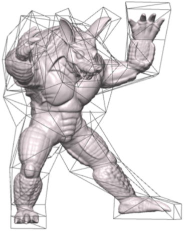
- Tool artists are familiar with.
- Allow smooth inside the cage and free deformations from control positions.
- Can be plug with a chosen barycentric coordinates system.
- Global space deformation.



(a) Bind pose

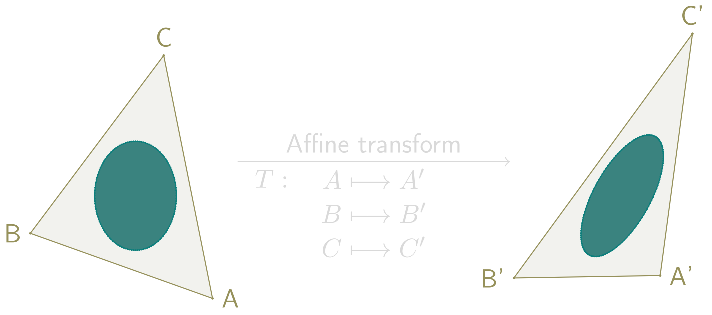


(b) Mean Value Coordinates



(c) Green Coordinates

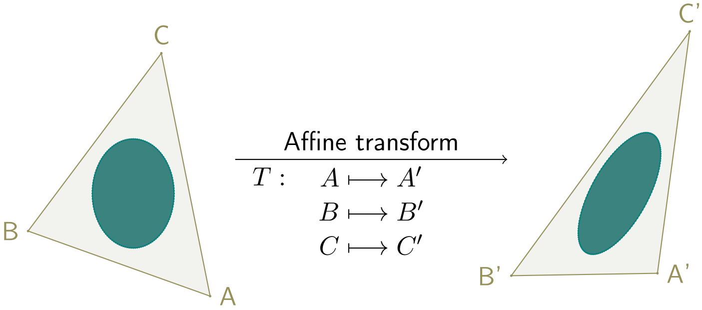
Barycentric coordinates and affine transform



$$P = \alpha A + \beta B + \gamma C$$
$$\alpha + \beta + \gamma = 1$$
$$\Rightarrow \alpha \vec{PA} + \beta \vec{PB} + \gamma \vec{PC} = 0$$

$$Q = T(P) = T(\alpha A + \beta B + \gamma C)$$
$$Q = \alpha T(A) + \beta T(B) + \gamma T(C)$$
$$Q = \alpha A' + \beta B' + \gamma C'$$

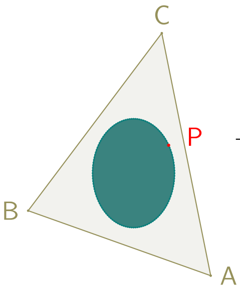
Barycentric coordinates and affine transform



$$P = \alpha A + \beta B + \gamma C$$
$$\alpha + \beta + \gamma = 1$$
$$\Rightarrow \alpha \vec{PA} + \beta \vec{PB} + \gamma \vec{PC} = 0$$

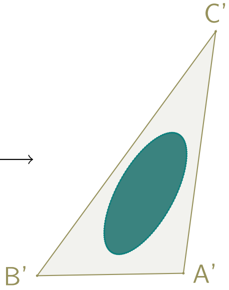
$$Q = T(P) = T(\alpha A + \beta B + \gamma C)$$
$$Q = \alpha T(A) + \beta T(B) + \gamma T(C)$$
$$Q = \alpha A' + \beta B' + \gamma C'$$

Barycentric coordinates and affine transform



Affine transform

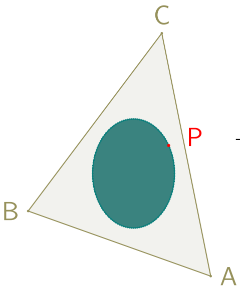
$$T: \quad A \longmapsto A' \quad B \longmapsto B' \quad C \longmapsto C'$$



$$P = \alpha A + \beta B + \gamma C$$
$$\alpha + \beta + \gamma = 1$$
$$\Rightarrow \alpha \vec{PA} + \beta \vec{PB} + \gamma \vec{PC} = 0$$

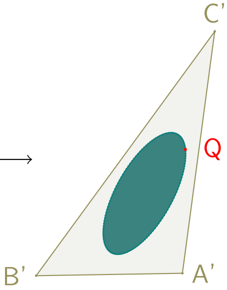
$$Q = T(P) = T(\alpha A + \beta B + \gamma C)$$
$$Q = \alpha T(A) + \beta T(B) + \gamma T(C)$$
$$Q = \alpha A' + \beta B' + \gamma C'$$

Barycentric coordinates and affine transform



Affine transform

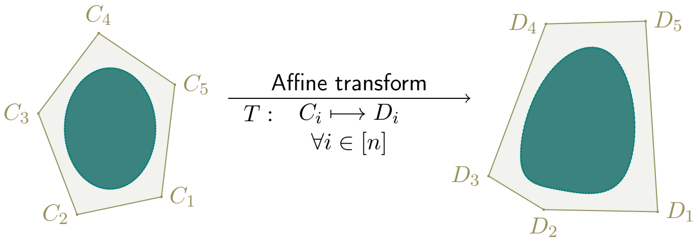
$$T: \quad A \longmapsto A'$$
$$B \longmapsto B'$$
$$C \longmapsto C'$$



$$P = \alpha A + \beta B + \gamma C$$
$$\alpha + \beta + \gamma = 1$$
$$\Rightarrow \alpha \vec{PA} + \beta \vec{PB} + \gamma \vec{PC} = 0$$

$$Q = T(P) = T(\alpha A + \beta B + \gamma C)$$
$$Q = \alpha T(A) + \beta T(B) + \gamma T(C)$$
$$Q = \alpha A' + \beta B' + \gamma C'$$

Generalized barycentric coordinates to cages



$$P = \sum_{i \in [n]} \alpha_i(P) C_i$$

$$\sum_{i \in [n]} \alpha_i(P) = 1$$

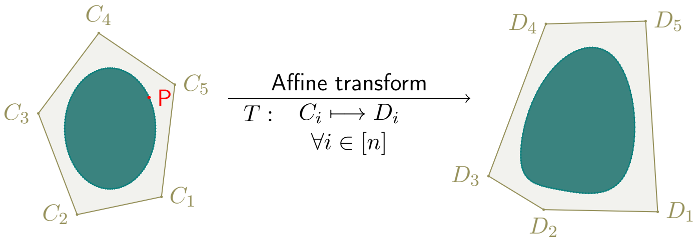
Binding step :
compute weights $(\alpha_i(P))_{i \in [n]}$

$$Q = T(P) = T(\sum_{i \in [n]} \alpha_i(P) C_i)$$

$$Q = \sum_{i \in [n]} \alpha_i(P) T(C_i)$$

$$Q = \sum_{i \in [n]} \alpha_i(P) D_i$$

Generalized barycentric coordinates to cages



$$P = \sum_{i \in [n]} \alpha_i(P) C_i$$

$$\sum_{i \in [n]} \alpha_i(P) = 1$$

Binding step :

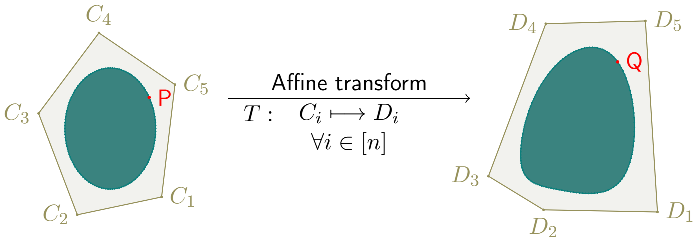
compute weights $(\alpha_i(P))_{i \in [n]}$

$$Q = T(P) = T(\sum_{i \in [n]} \alpha_i(P) C_i)$$

$$Q = \sum_{i \in [n]} \alpha_i(P) T(C_i)$$

$$Q = \sum_{i \in [n]} \alpha_i(P) D_i$$

Generalized barycentric coordinates to cages



$$P = \sum_{i\in[n]} \alpha_i(P)C_i$$

$$\sum_{i\in[n]} \alpha_i(P) = 1$$

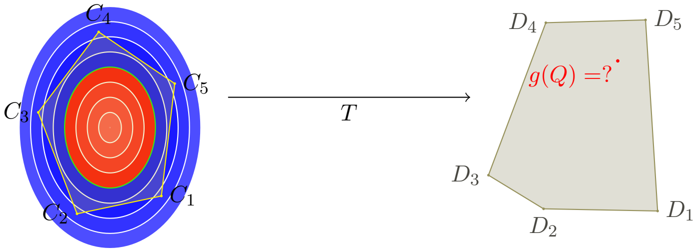
Binding step :
compute weights $(\alpha_i(P))_{i\in[n]}$

$$Q = T(P) = T(\sum_{i\in[n]} \alpha_i(P)C_i)$$

$$Q = \sum_{i\in[n]} \alpha_i(P)T(C_i)$$

$$Q = \sum_{i\in[n]} \alpha_i(P)D_i$$

Inverse position problem

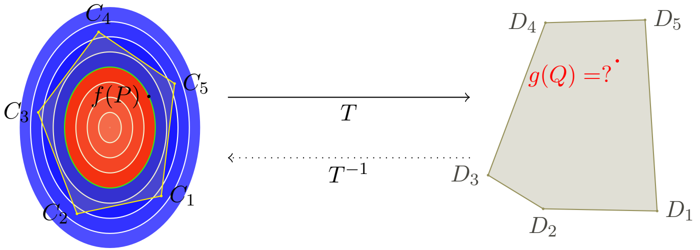


$$g(Q) = f(P)$$
$$g(Q) = f(T^{-1}(Q))$$
$$Q = \sum_{i \in [n]} \alpha_i(P) D_i$$

$$Q = \sum_{i \in [n]} \alpha_i(P)(C_i + \vec{u_i})$$
$$Q = P + \sum_{i \in [n]} \alpha_i(P) \vec{u_i}$$
$$P = Q - \sum_{i \in [n]} \alpha_i(P) \vec{u_i}$$

$\left\{ \overrightarrow{C_1 C_i} \right\}_{i \in [n] \setminus \{1\}}$ linearly dependent.

Inverse position problem

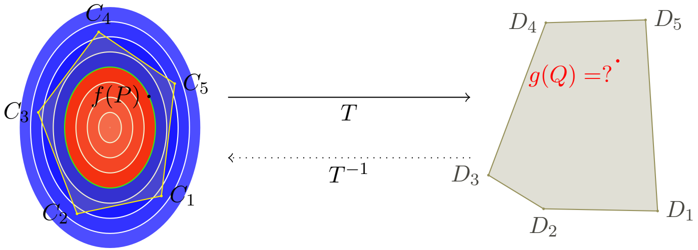


$$g(Q) = f(P)$$
$$g(Q) = f(T^{-1}(Q))$$
$$Q = \sum_{i \in [n]} \alpha_i(P) D_i$$

$$Q = \sum_{i \in [n]} \alpha_i(P)(C_i + \vec{u_i})$$
$$Q = P + \sum_{i \in [n]} \alpha_i(P)\vec{u_i}$$
$$P = Q - \sum_{i \in [n]} \alpha_i(P)\vec{u_i}$$

$\left\{\overrightarrow{C_1 C_i}\right\}_{i \in [n] \setminus \{1\}}$ linearly dependent.

Inverse position problem

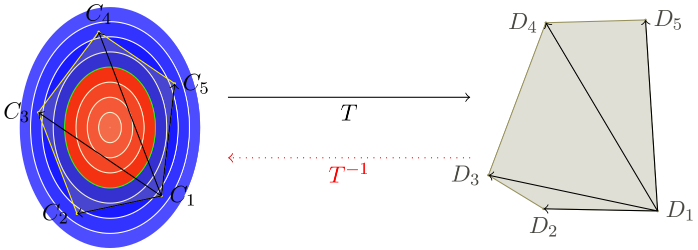


$$g(Q) = f(P)$$
$$g(Q) = f(T^{-1}(Q))$$
$$Q = \sum_{i \in [n]} \alpha_i(P) D_i$$

$$Q = \sum_{i \in [n]} \alpha_i(P)(C_i + \vec{u_i})$$
$$Q = P + \sum_{i \in [n]} \alpha_i(P)\vec{u_i}$$
$$P = Q - \sum_{i \in [n]} \alpha_i(P)\vec{u_i}$$

$\left\{\overrightarrow{C_1 C_i}\right\}_{i \in [n] \setminus \{1\}}$ linearly dependent.

Inverse position problem

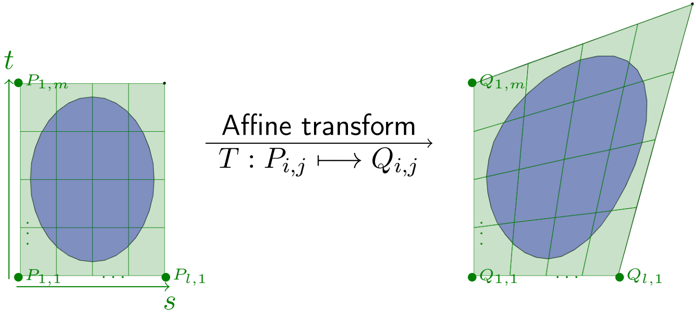


$$g(Q) = f(P)$$
$$g(Q) = f(T^{-1}(Q))$$
$$Q = \sum_{i \in [n]} \alpha_i(P) D_i$$

$$Q = \sum_{i \in [n]} \alpha_i(P)(C_i + \vec{u_i})$$
$$Q = P + \sum_{i \in [n]} \alpha_i(P)\vec{u_i}$$
$$\mathbf{P} = Q - \sum_{i \in [n]} \alpha_i(\mathbf{P})\vec{u_i}$$

$\left\{\overrightarrow{C_1 C_i}\right\}_{i \in [n] \setminus \{1\}}$ linearly dependent.

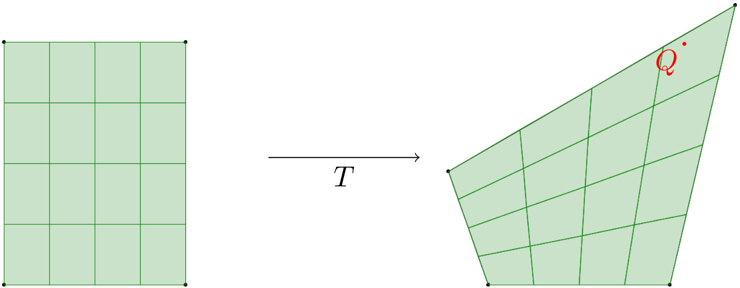
Free-Form-Deformations



(s,t) coordinates in $(P_{1,1}, \overrightarrow{P_{1,1}P_{l,1}}, \overrightarrow{P_{1,1}P_{1,m}})$.
$P_{i,j}$ and $Q_{i,j}$ bilinear interpolation of the parallelepipeds.

$$T(s,t) = \sum_{(i,j)\in[l]\times[m]} \binom{l}{i}\binom{m}{j} s^i (1-s)^{l-i} t^j (1-t)^{m-j} T(P_{i,j})$$

Free-Form-Deformations and inverse problem solving



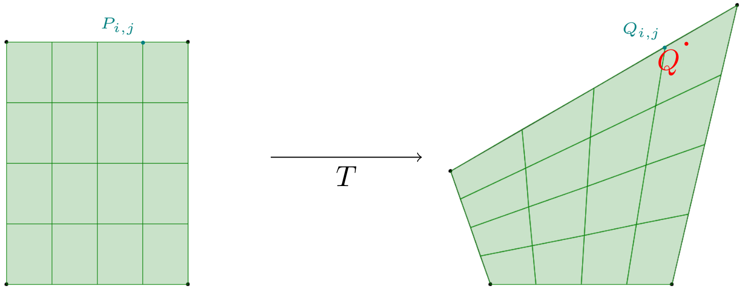
Get nearest $Q_{i,j}$,
hence $P_{i,j}$ equivalent.
Solve $T(s,t) - Q = 0$
using Newton method.

$$J_T(s,t) = \left(\frac{\partial}{\partial s}T(s,t), \frac{\partial}{\partial t}T(s,t)\right)$$

$$X_0 = \left(\frac{i}{l}, \frac{j}{m}\right)$$

$$X_{n+1} = X_n - {J_T}^{-1}(X_n)\,(T(X_n) - Q)$$

Free-Form-Deformations and inverse problem solving



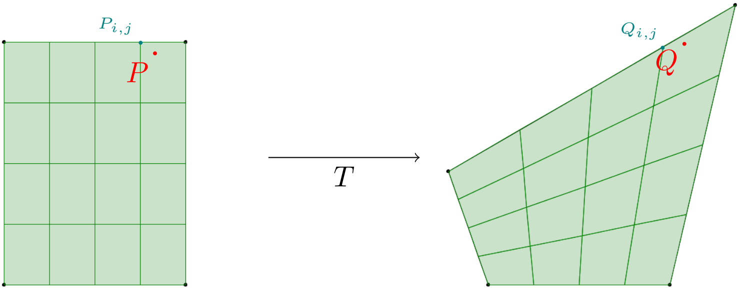
Get nearest $Q_{i,j}$,
hence $P_{i,j}$ equivalent.
Solve $T(s,t) - Q = 0$
using Newton method.

$$J_T(s,t) = \left(\frac{\partial}{\partial s}T(s,t), \frac{\partial}{\partial t}T(s,t)\right)$$

$$X_0 = \left(\frac{i}{l}, \frac{j}{m}\right)$$

$$X_{n+1} = X_n - {J_T}^{-1}(X_n)\,(T(X_n) - Q)$$

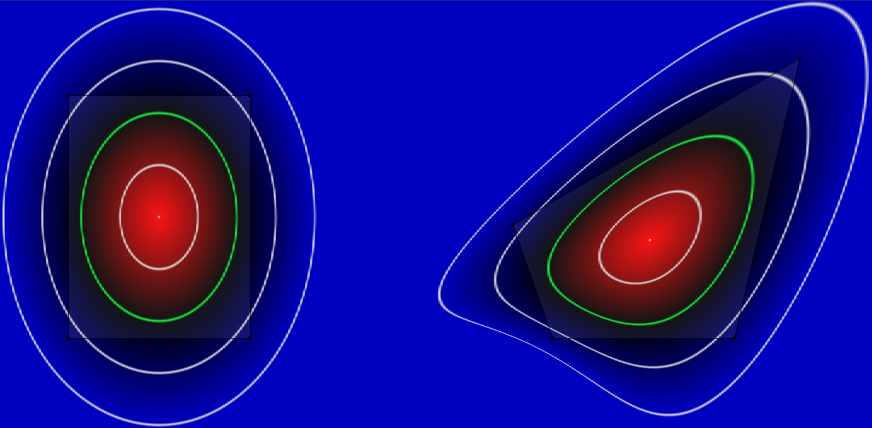
Free-Form-Deformations and inverse problem solving



Get nearest $Q_{i,j}$,
hence $P_{i,j}$ equivalent.
Solve $T(s,t) - Q = 0$
using Newton method.

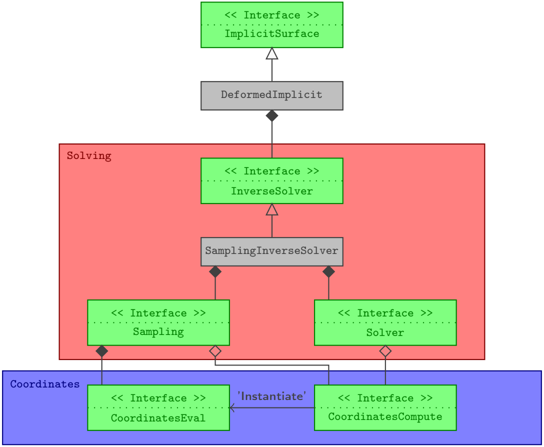
$$J_T(s,t) = \left(\frac{\partial}{\partial s}T(s,t), \frac{\partial}{\partial t}T(s,t)\right)$$
$$X_0 = \left(\frac{i}{l}, \frac{j}{m}\right)$$
$$X_{n+1} = X_n - {J_T}^{-1}(X_n)\,(T(X_n) - Q)$$

Free-Form-Deformations implicit surface example in 2D

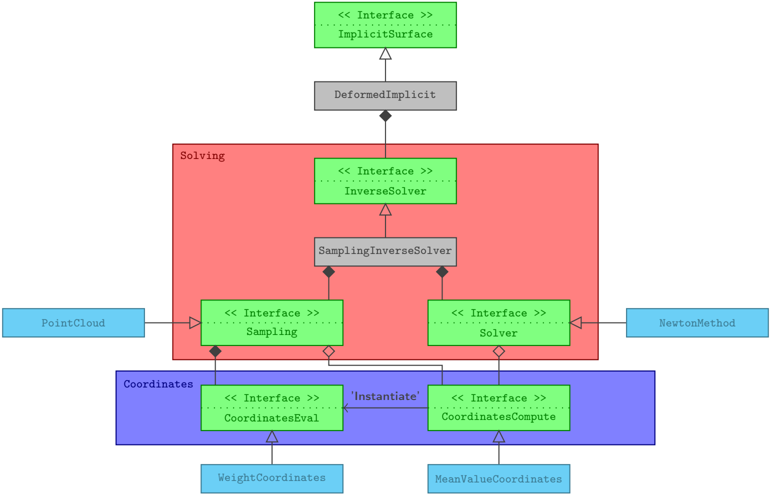


Example in 2D (render done in java using Processing IDE).

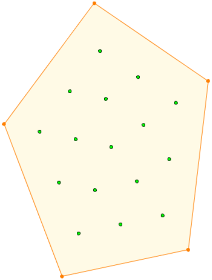
Our general architecture for inverse position solving

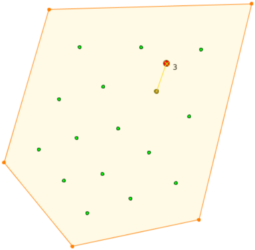


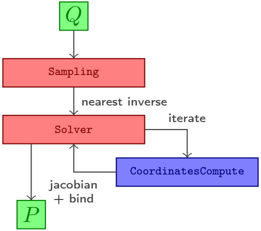
State of the art inspired idea : architecture



State of the art inspired idea : method







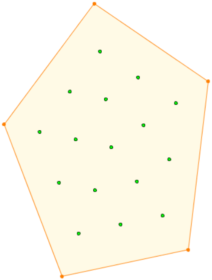
Solve $T(P) - Q = 0$:

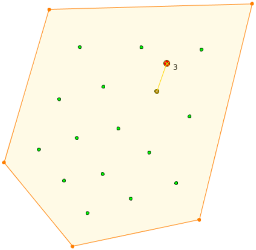
$J_T(x, y) = \left(\frac{\partial}{\partial x} T(x, y), \frac{\partial}{\partial y} T(x, y)\right)$

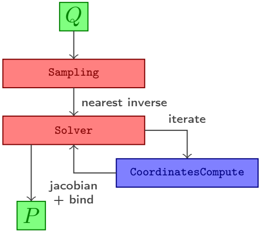
P_0 nearest sample.

$P_{n+1} = P_n - {J_T}^{-1}(P_n)\,(T(P_n) - Q)$

State of the art inspired idea : method







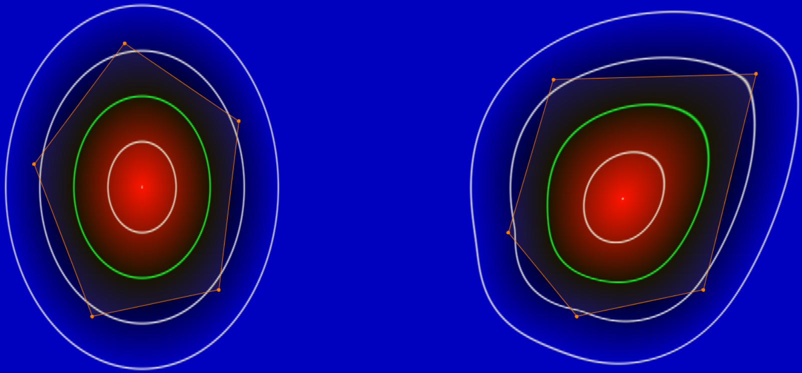
Solve $T(P) - Q = 0$:

$J_T(x,y) = \left(\frac{\partial}{\partial x}T(x,y), \frac{\partial}{\partial y}T(x,y)\right)$

P_0 nearest sample.

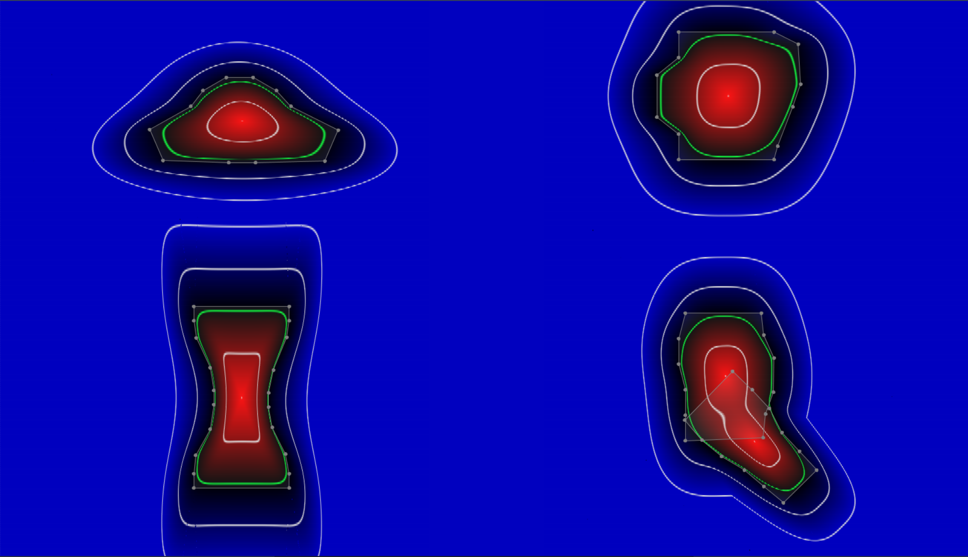
$P_{n+1} = P_n - {J_T}^{-1}(P_n)\,(T(P_n) - Q)$

State of the art inspired idea : result in 2D

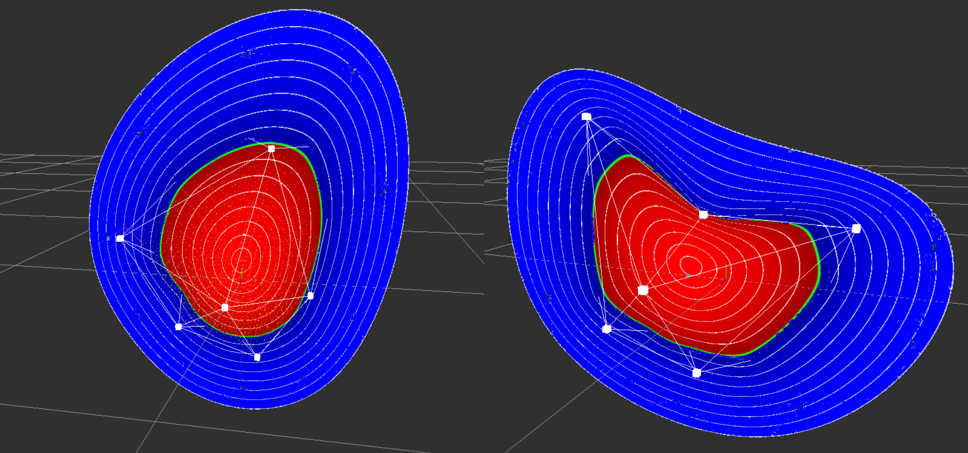


Example in 2D (render done in java using Processing IDE).

State of the art inspired idea : examples in 2D



State of the art inspired idea : result in 3D

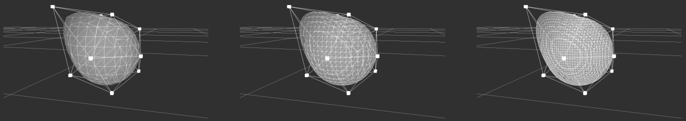


Example in 3D (render done in C++ using Radium Engine).

State of the art inspired : benchmark marching cubes

Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

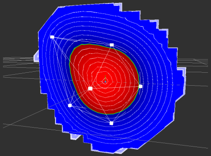
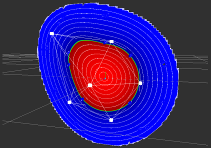
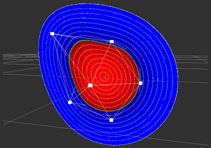
resolution	$16 \times 16 \times 16$ (2^{12})	$32 \times 32 \times 32$ (2^{15})	$64 \times 64 \times 64$ (2^{18})
time	$9.24 \cdot 10^8 ns(0.924s)$	$4.6 \cdot 10^9 ns(4.6s)$	$2.65 \cdot 10^{10} ns(26.5s)$
visualize			



Kd-tree and sampling update time : $2.4 \cdot 10^6 ns(2.4ms)$ in average.

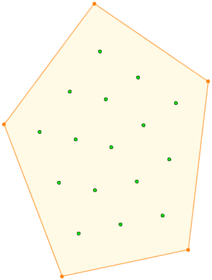
State of the art inspired : benchmark display planes

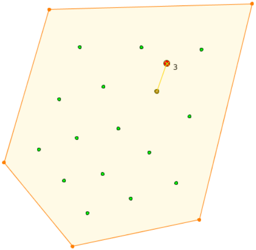
Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

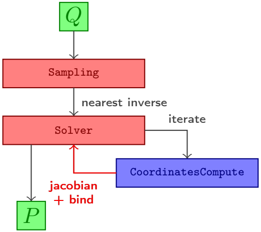
resolution	32×32 (2^{10})	128×128 (2^{14})	512×512 (2^{18})
time	$1.62 \cdot 10^8 ns (162ms)$	$2.4 \cdot 10^9 ns (2.4s)$	$3.82 \cdot 10^{10} ns (38.2s)$
visualize			

Kd-tree and sampling update time : $2.4 \cdot 10^6 ns (2.4ms)$ in average.

State of the art inspired idea : method







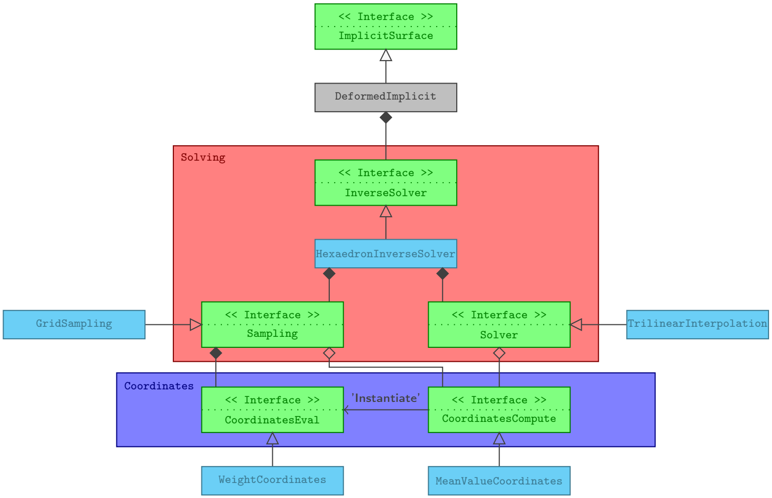
Solve $T(P) - Q = 0$:

$J_T(x, y) = \left(\frac{\partial}{\partial x} T(x, y), \frac{\partial}{\partial y} T(x, y)\right)$

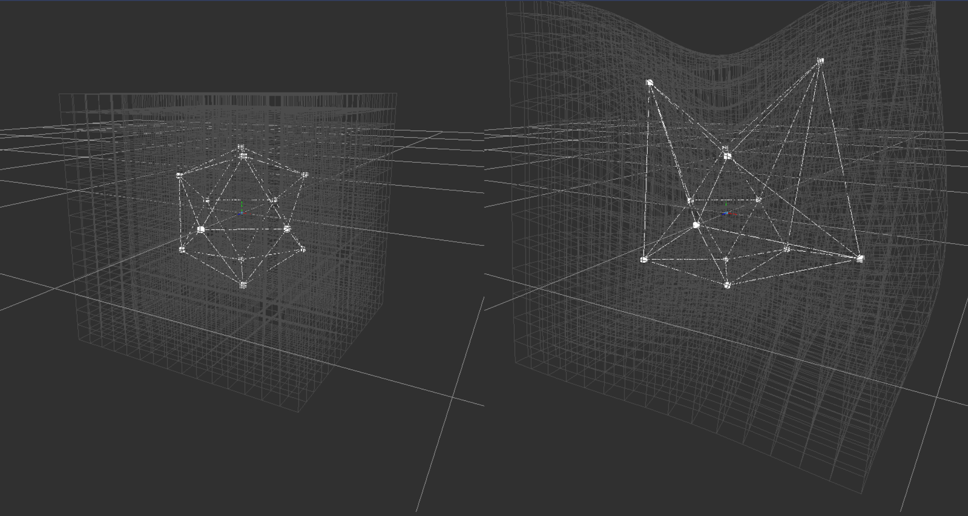
P_0 nearest sample.

$P_{n+1} = P_n - {J_T}^{-1}(P_n)\,(T(P_n) - Q)$

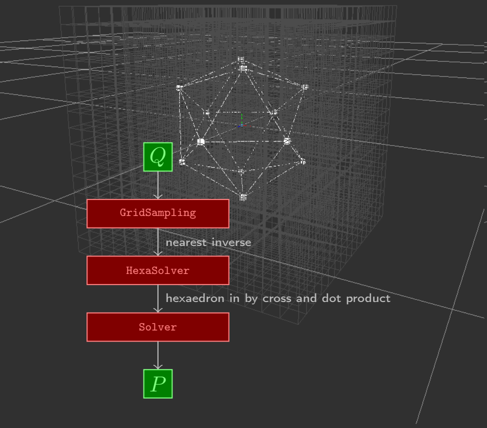
Bounding grid and trilinear approximation : architecture

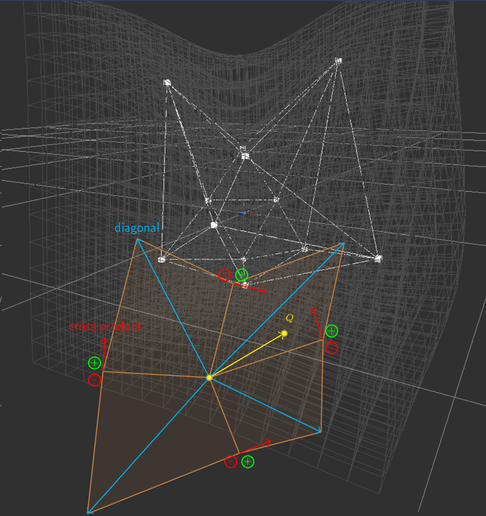


Bounding grid and trilinear approximation : method



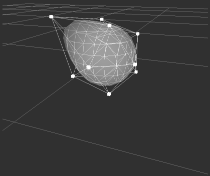
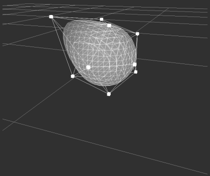
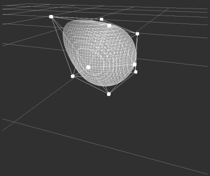
Bounding grid and trilinear approximation : method





Bounding grid nearest neighbor : benchmark marching cubes

Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

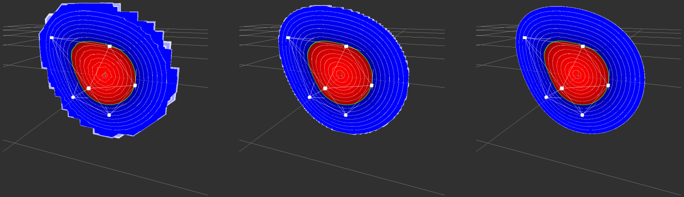
resolution	$16 \times 16 \times 16$ (2^{12})	$32 \times 32 \times 32$ (2^{15})	$64 \times 64 \times 64$ (2^{18})
time	$9.9 \cdot 10^7 ns (99ms)$	$4.1 \cdot 10^8 ns (410ms)$	$1.8 \cdot 10^9 ns (1.8s)$
visualize			

Kd-tree and sampling update time : $2.1 \cdot 10^7 ns (21ms)$ in average.

Bounding grid nearest neighbor : benchmark display planes

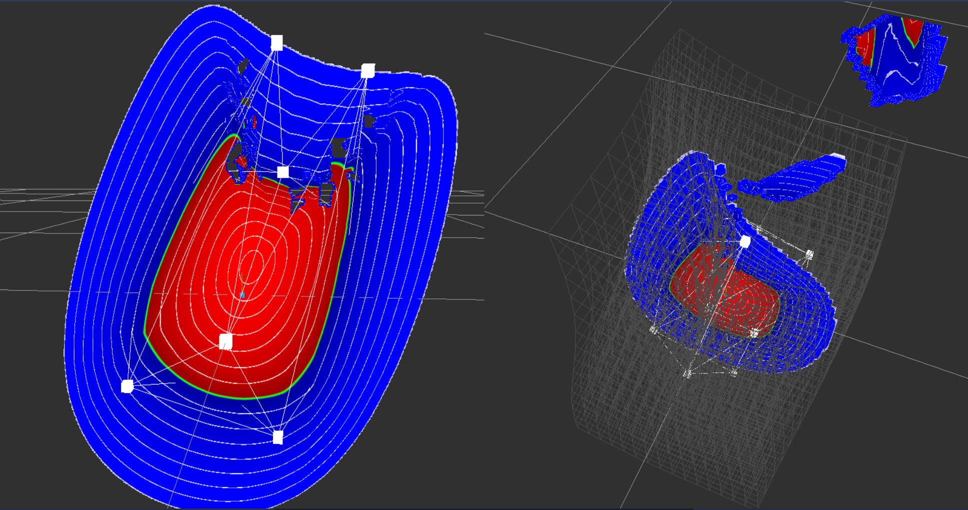
Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

resolution	$32 \times 32\ (2^{10})$	$128 \times 128\ (2^{14})$	$512 \times 512\ (2^{18})$
time	$7.9 \cdot 10^6 ns (7.9ms)$	$8 \cdot 10^7 ns (80ms)$	$1.29 \cdot 10^9 ns (1.29s)$
visualize			

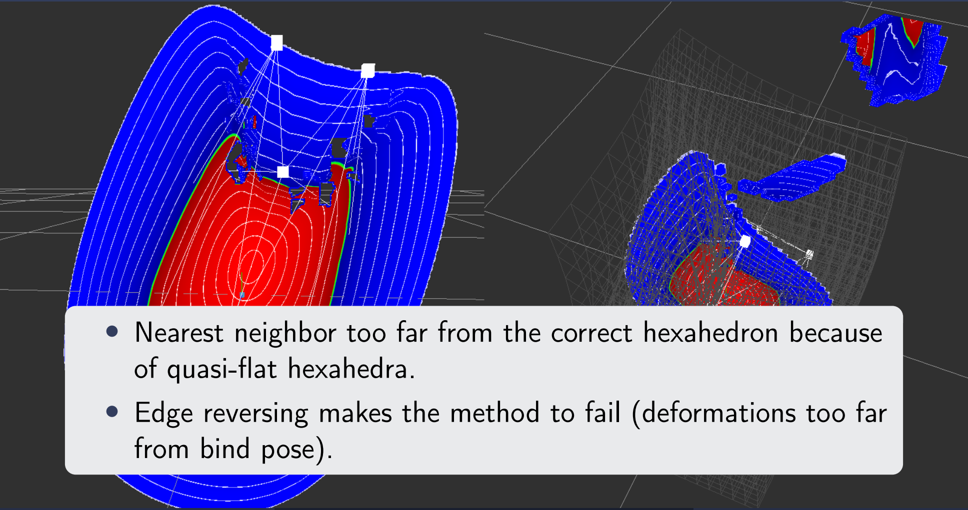


Kd-tree and sampling update time : $2.1 \cdot 10^7 ns (21ms)$ in average.

Bounding grid nearest neighbor : robustness to deformations problems



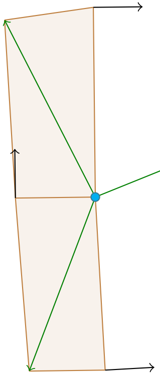
Bounding grid nearest neighbor : robustness to deformations problems



- Nearest neighbor too far from the correct hexahedron because of quasi-flat hexahedra.
- Edge reversing makes the method to fail (deformations too far from bind pose).

Grid reversal fail : example

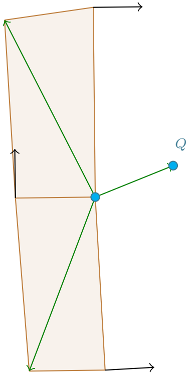
Correctly oriented :



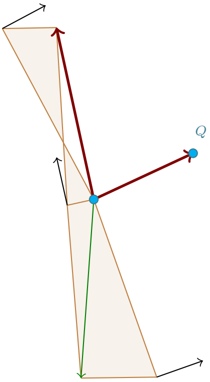
Edge reversed :

Grid reversal fail : example

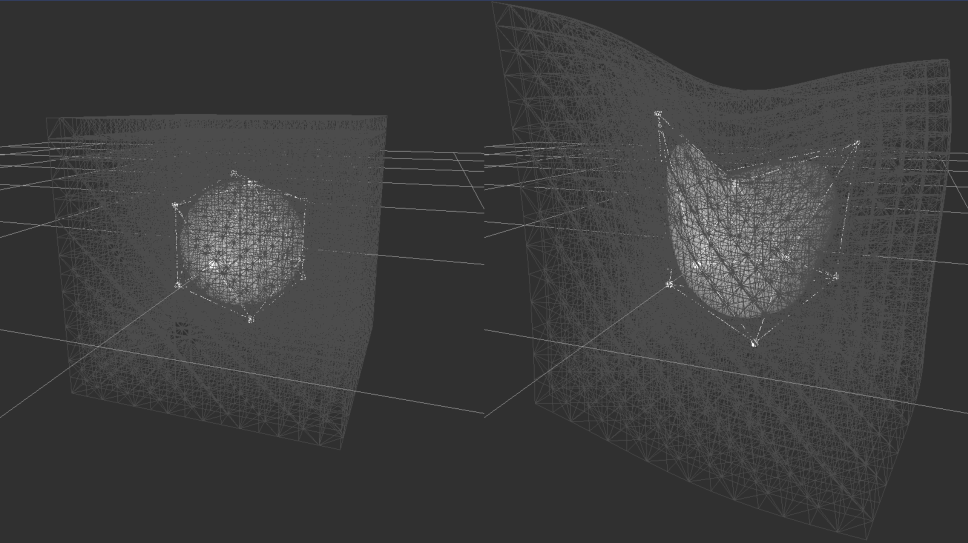
Correctly oriented :



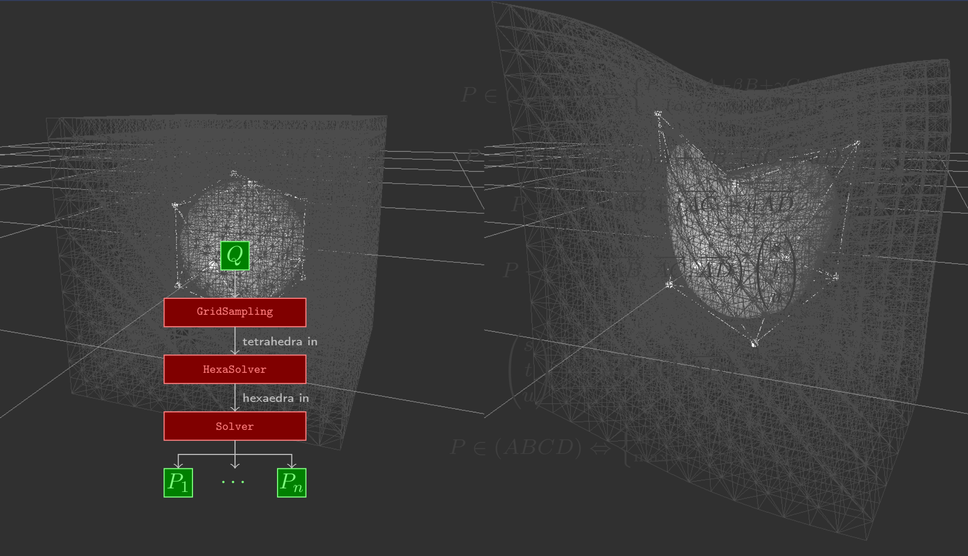
Edge reversed :



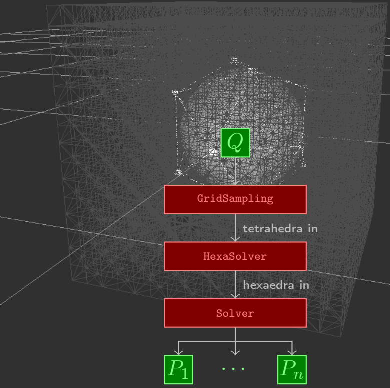
Bounding grid tetrahedral cut : method

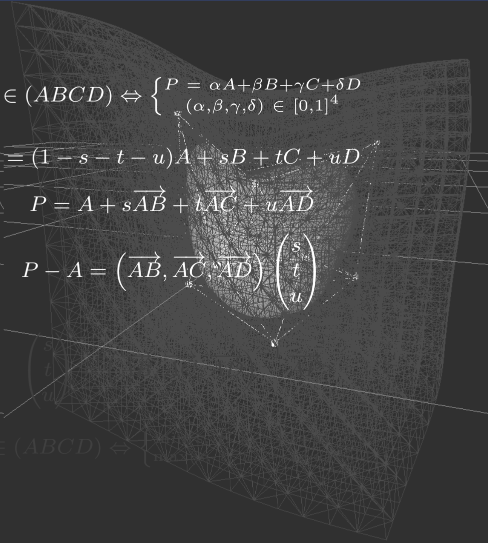


Bounding grid tetrahedral cut : method



Bounding grid tetrahedral cut : method





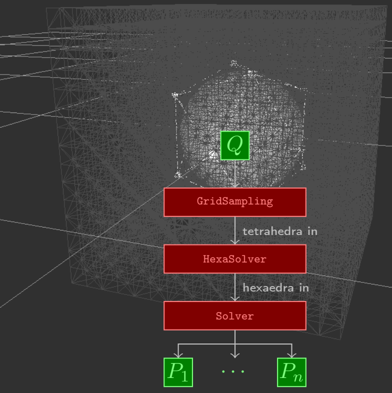
$$P \in (ABCD) \Leftrightarrow \begin{cases} P = \alpha A + \beta B + \gamma C + \delta D \\ (\alpha, \beta, \gamma, \delta) \in [0,1]^4 \end{cases}$$

$$P = (1 - s - t - u)A + sB + tC + uD$$

$$P = A + s\overrightarrow{AB} + t\overrightarrow{AC} + u\overrightarrow{AD}$$

$$P - A = \left(\overrightarrow{AB}, \overrightarrow{AC}, \overrightarrow{AD}\right)\begin{pmatrix} s \\ t \\ u \end{pmatrix}$$

Bounding grid tetrahedral cut : method



$$P \in (ABCD) \Leftrightarrow \begin{cases} P = \alpha A + \beta B + \gamma C + \delta D \\ (\alpha, \beta, \gamma, \delta) \in [0,1]^4 \end{cases}$$

$$P = (1 - s - t - u)A + sB + tC + uD$$

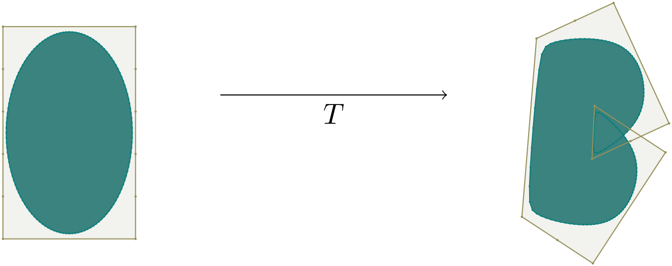
$$P = A + s\overrightarrow{AB} + t\overrightarrow{AC} + u\overrightarrow{AD}$$

$$P - A = \left(\overrightarrow{AB}, \overrightarrow{AC}, \overrightarrow{AD}\right) \begin{pmatrix} s \\ t \\ u \end{pmatrix}$$

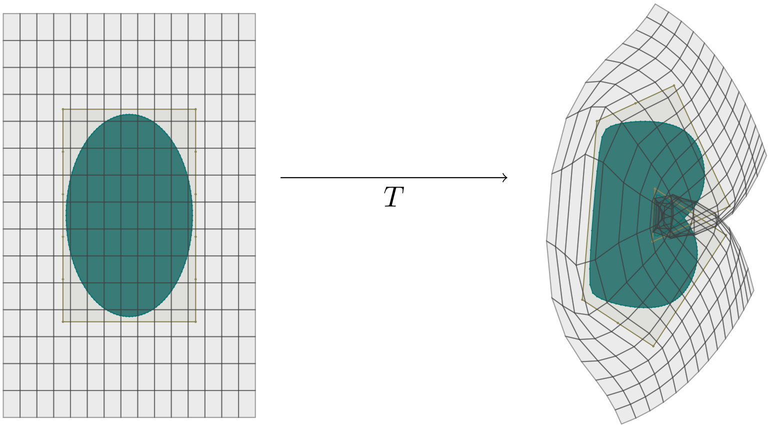
$$\begin{pmatrix} s \\ t \\ u \end{pmatrix} = \left(\overrightarrow{AB}, \overrightarrow{AC}, \overrightarrow{AD}\right)^{-1} \overrightarrow{AP}$$

$$P \in (ABCD) \Leftrightarrow \begin{cases} \min(s,\ t,\ u,\ s+t+u) \geq 0 \\ \max(s,\ t,\ u,\ s+t+u) \leq 1 \end{cases}$$

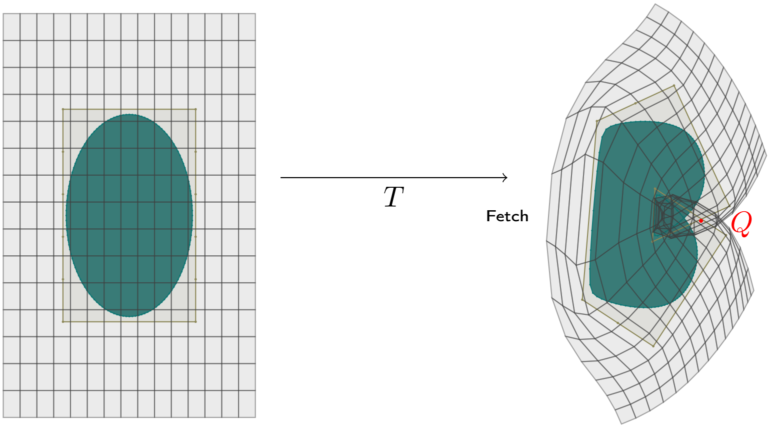
Multiple equivalents solving : example



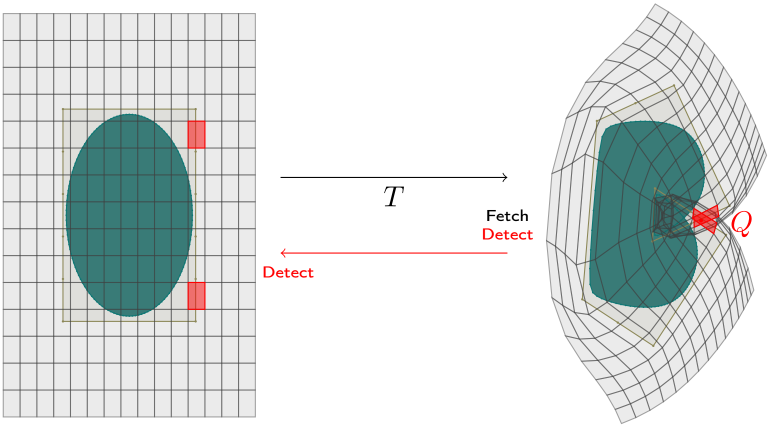
Multiple equivalents solving : example



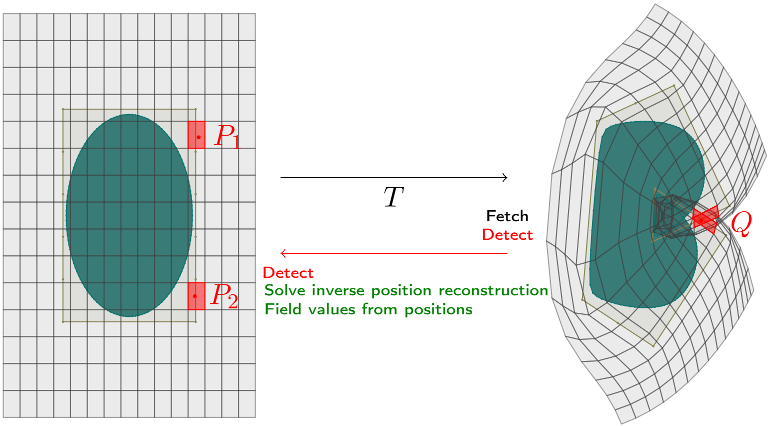
Multiple equivalents solving : example



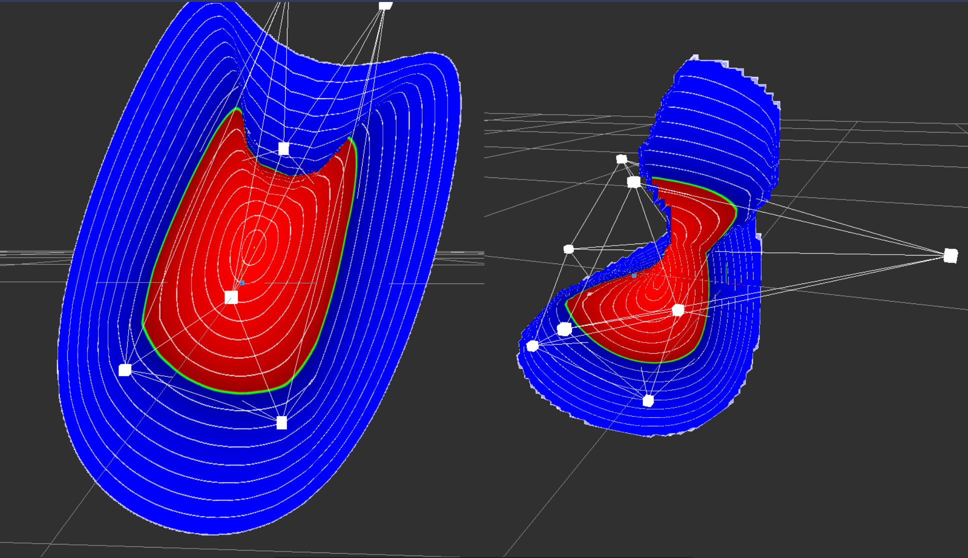
Multiple equivalents solving : example



Multiple equivalents solving : example

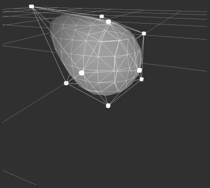
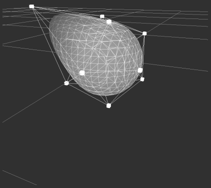
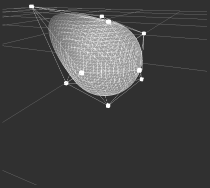


Bounding grid tetrahedral cut : more robust solution



Bounding grid tetrahedral cut : benchmark marching cubes

Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

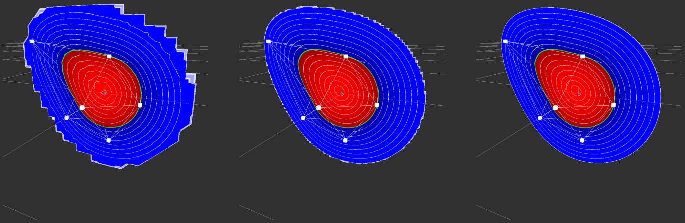
resolution	$16 \times 16 \times 16\ (2^{12})$	$32 \times 32 \times 32\ (2^{15})$	$64 \times 64 \times 64\ (2^{18})$
time	$8.5 \cdot 10^7 ns (85ms)$	$3.31 \cdot 10^8 ns (331ms)$	$1.37 \cdot 10^9 ns (1.37s)$
visualize			

BIH and sampling update time : $1.7 \cdot 10^7 ns (17ms)$ in average.

Bounding grid tetrahedral cut : benchmark display planes

Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

resolution	$32 \times 32\ (2^{10})$	$128 \times 128\ (2^{14})$	$512 \times 512\ (2^{18})$
time	$2.7 \cdot 10^7 ns (27ms)$	$6 \cdot 10^7 ns (60ms)$	$8.05 \cdot 10^8 ns (805ms)$
visualize			



BIH and sampling update time : $1.7 \cdot 10^7 ns (17ms)$ in average.

Benchmark overview

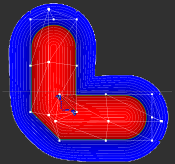
Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

Method :	State of the art inspired	Bounding grid nearest neighbor	Bounding grid tetrahedral cut
Update time	$2.4 \cdot 10^6 ns (2.4ms)$	$2.1 \cdot 10^7 ns (21ms)$	$1.7 \cdot 10^7 ns (17ms)$
Marching cubes :			
$16 \times 16 \times 16$ (2^{12})	$9.24 \cdot 10^8 ns (0.924s)$	$9.9 \cdot 10^7 ns (99ms)$	$8.5 \cdot 10^7 ns (85ms)$
$32 \times 32 \times 32$ (2^{15})	$4.6 \cdot 10^9 ns (4.6s)$	$4.1 \cdot 10^8 ns (410ms)$	$3.31 \cdot 10^8 ns (331ms)$
$64 \times 64 \times 64$ (2^{18})	$2.65 \cdot 10^{10} ns (26.5s)$	$1.8 \cdot 10^9 ns (1.8s)$	$1.37 \cdot 10^9 ns (1.37s)$
Display plane :			
32×32 (2^{10})	$1.62 \cdot 10^8 ns (162ms)$	$7.9 \cdot 10^6 ns (7.9ms)$	$2.7 \cdot 10^7 ns (27ms)$
128×128 (2^{14})	$2.4 \cdot 10^9 ns (2.4s)$	$8 \cdot 10^7 ns (80ms)$	$6 \cdot 10^7 ns (60ms)$
512×512 (2^{18})	$3.82 \cdot 10^{10} ns (38.2s)$	$1.29 \cdot 10^9 ns (1.29s)$	$8.05 \cdot 10^8 ns (805ms)$

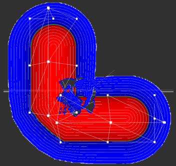
Multiple equivalent self-intersection results

Implicit blob capsule deformed by cage (26 vertices, 48 faces) :

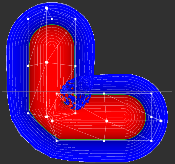
State of the art inspired
$1 \cdot 10^{11} ns(100s)$



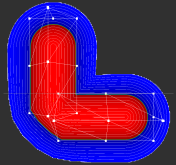
Bounding grid nearest neighbor
$1.4 \cdot 10^{9} ns(1.4s)$



Bounding grid tetrahedral detection
$6.93 \cdot 10^{8} ns(693ms)$



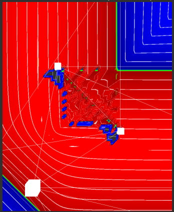
Bounding grid tetrahedral multiple detection
$7.17 \cdot 10^{8} ns(717ms)$



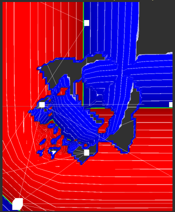
Multiple equivalent self-intersection results : zoom

Implicit blob capsule deformed by cage (26 vertices, 48 faces) :

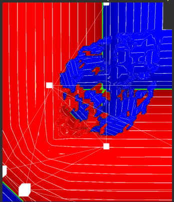
State of the art inspired
$1 \cdot 10^{11} ns (100s)$



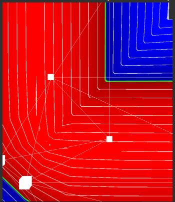
Bounding grid nearest neighbor
$1.4 \cdot 10^{9} ns (1.4s)$



Bounding grid tetrahedral detection
$6.93 \cdot 10^{8} ns (693ms)$



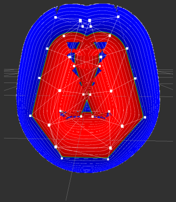
Bounding grid tetrahedral multiple detection
$7.17 \cdot 10^{8} ns (717ms)$



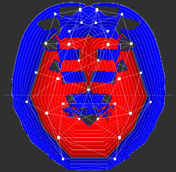
Multiple equivalent self-intersection and contact results

Implicit blob capsule deformed by cage (42 vertices, 80 faces) :

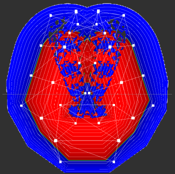
State of the art inspired
$1.58 \cdot 10^{11} ns (158s)$



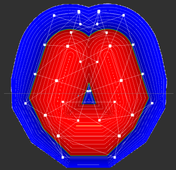
Bounding grid nearest neighbor
$2.2 \cdot 10^{9} ns (2.2s)$



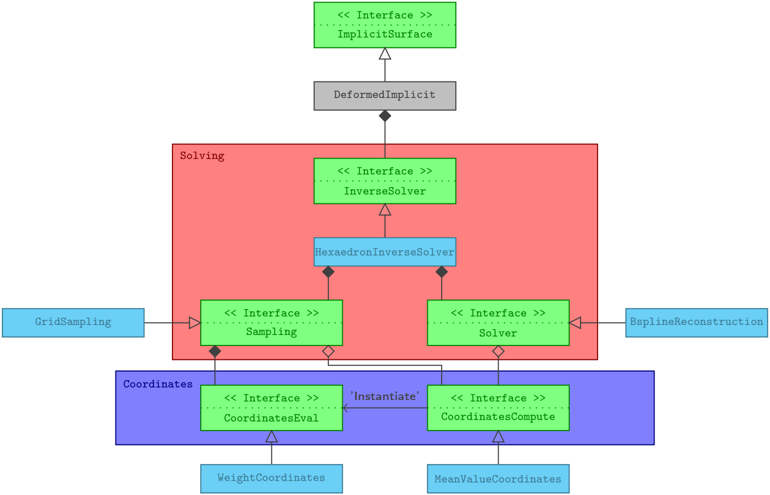
Bounding grid tetrahedral detection
$1.75 \cdot 10^{9} ns (1.75s)$



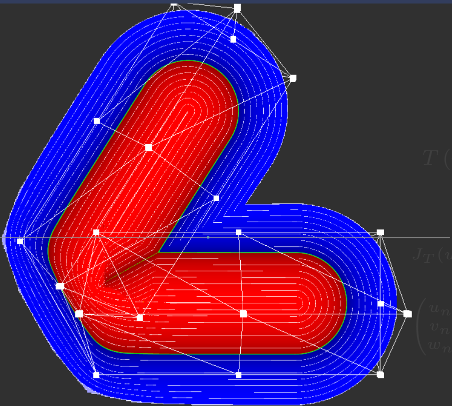
Bounding grid tetrahedral multiple detection
$1.67 \cdot 10^{9} ns (1.67s)$



Bounding grid B-spline reconstruction : architecture



B-spline reconstruction



Hexahedral detection.
Get starting (u_0, v_0, w_0).
Solve $T(u,v,w) - Q = 0$
using Newton method.

$$T(u,v,w) = \sum_{(i,j,k)\in[r]^3} \mathcal{N}_i^3(u)\,\mathcal{N}_j^3(v)\,\mathcal{N}_k^3(w)$$

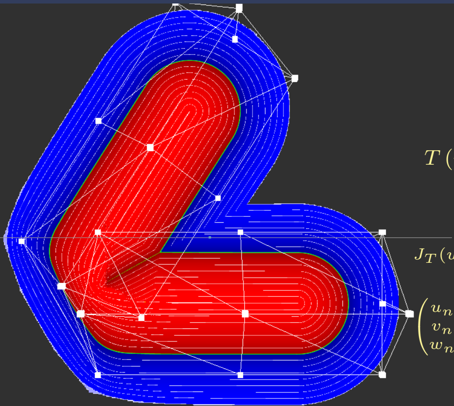
$$J_T(u,v,w) = \left(\frac{\partial}{\partial u}T(u,v,w), \frac{\partial}{\partial v}T(u,v,w), \frac{\partial}{\partial w}T(u,v,w)\right)$$

$$\begin{pmatrix} u_{n+1} \\ v_{n+1} \\ w_{n+1} \end{pmatrix} = \begin{pmatrix} u_n \\ v_n \\ w_n \end{pmatrix} - J_T^{-1}\left(\begin{pmatrix} u_n \\ v_n \\ w_n \end{pmatrix}\right)\left(T\left(\begin{pmatrix} u_n \\ v_n \\ w_n \end{pmatrix}\right) - Q\right)$$

Open nodal vector : $\left[0, 0, 0, \frac{1}{r-2}, \cdots, \frac{r-3}{r-2}, 1, 1, 1\right]$

Constrainst : $(u_n, v_n, w_n) \in (0,1)^3$

B-spline reconstruction



$$T(u,v,w) = \sum_{(i,j,k)\in[r]^3} \mathcal{N}_i^3(u)\,\mathcal{N}_j^3(v)\,\mathcal{N}_k^3(w)$$

$$J_T(u,v,w) = \left(\frac{\partial}{\partial u}T(u,v,w), \frac{\partial}{\partial v}T(u,v,w), \frac{\partial}{\partial w}T(u,v,w)\right)$$

$$\begin{pmatrix} u_{n+1} \\ v_{n+1} \\ w_{n+1} \end{pmatrix} = \begin{pmatrix} u_n \\ v_n \\ w_n \end{pmatrix} - {J_T}^{-1}\left(\begin{pmatrix} u_n \\ v_n \\ w_n \end{pmatrix}\right)\left(T\left(\begin{pmatrix} u_n \\ v_n \\ w_n \end{pmatrix}\right) - Q\right)$$

Hexahedral detection.
Get starting (u_0, v_0, w_0).
Solve $T(u,v,w) - Q = 0$
using Newton method.

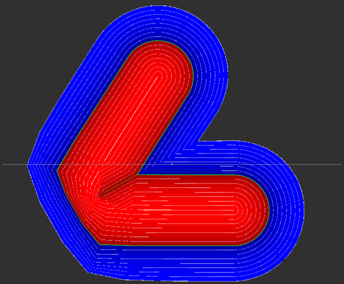
Open nodal vector : $\left[0, 0, 0, \frac{1}{r-2}, \cdots, \frac{r-3}{r-2}, 1, 1, 1\right]$

Constrainst : $(u_n, v_n, w_n) \in (0,1)^3$

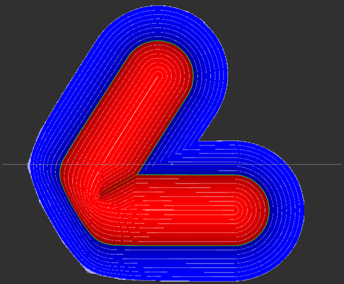
Reconstruction compare

Implicit blob capsule deformed by cage (26 vertices, 48 faces) :

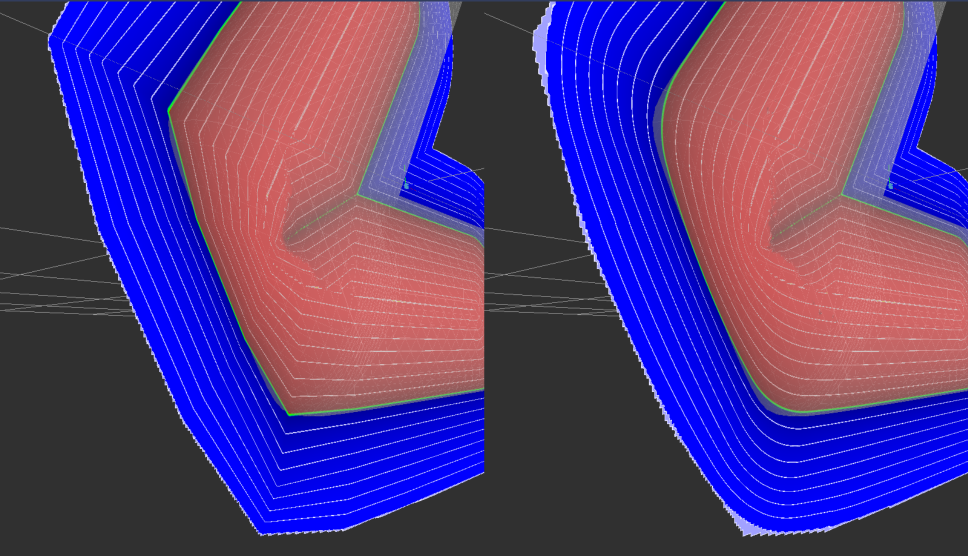
Trilinear reconstruction
$3.21 \cdot 10^{9} ns (3.2s)$



B-spline reconstruction
$1.19 \cdot 10^{11} ns (119s)$



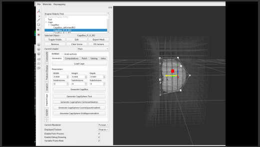
Reconstruction and coordinates system fitting



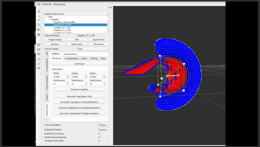
Space fold-over : movie time

Space fold-over solutions overview :

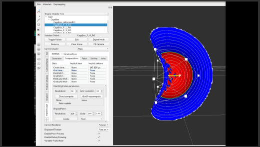
Bounding grid deform



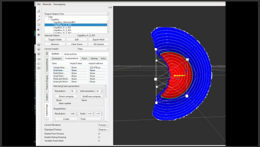
Bounding grid nearest neighbor



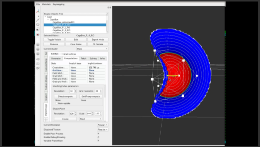
Bounding grid tetrahedral detection



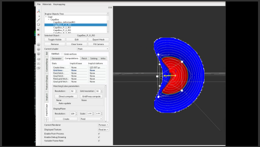
Bounding grid tetrahedral multiple detection solving (max)



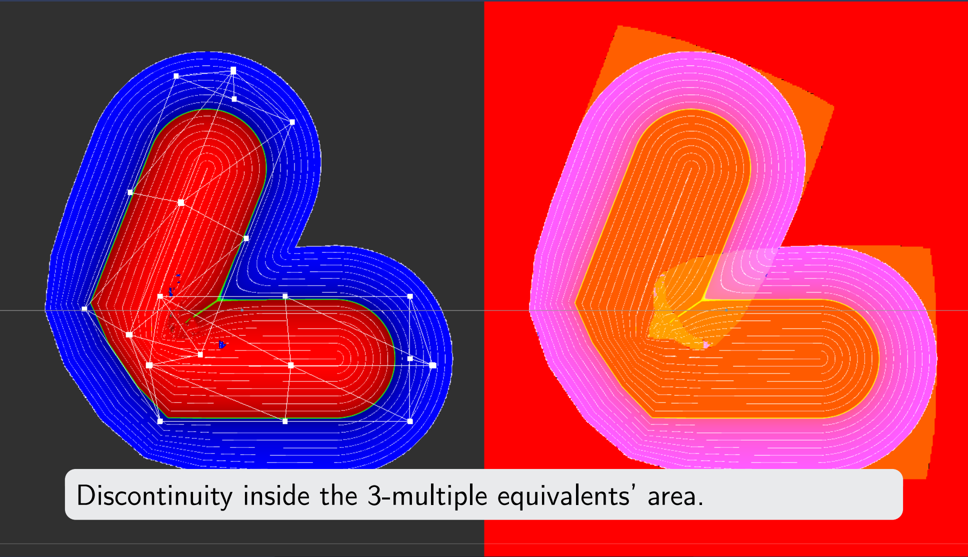
Bounding grid tetrahedral multiple detection solving (diff(max, min))



Bounding grid tetrahedral multiple and reversal detection solving (diff(space, reversed))

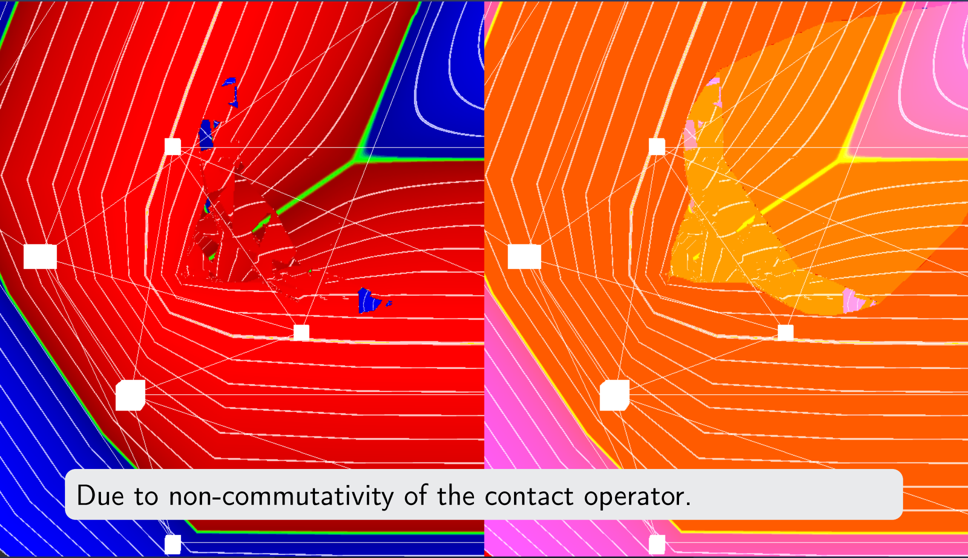


Multiple equivalents : order problem



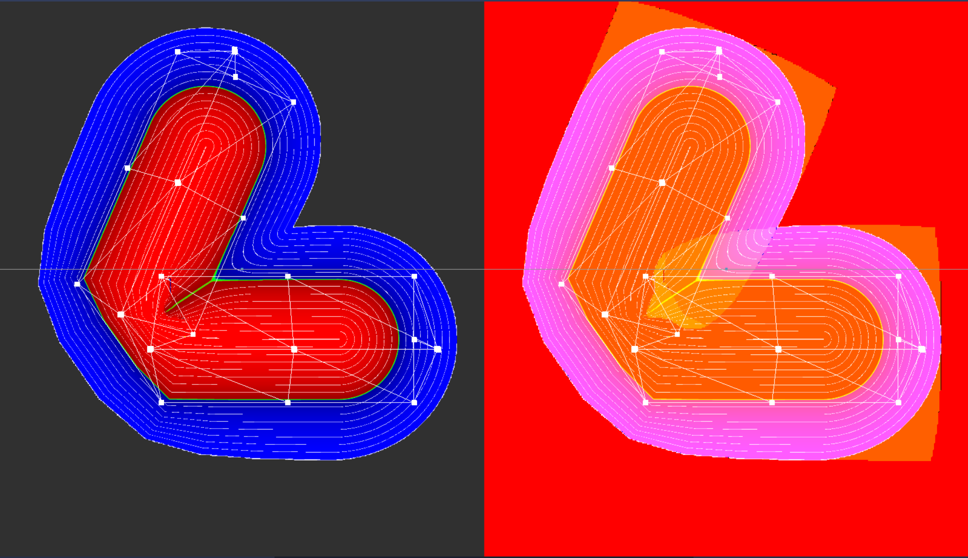
Discontinuity inside the 3-multiple equivalents' area.

Multiple equivalents : order problem (zoom)

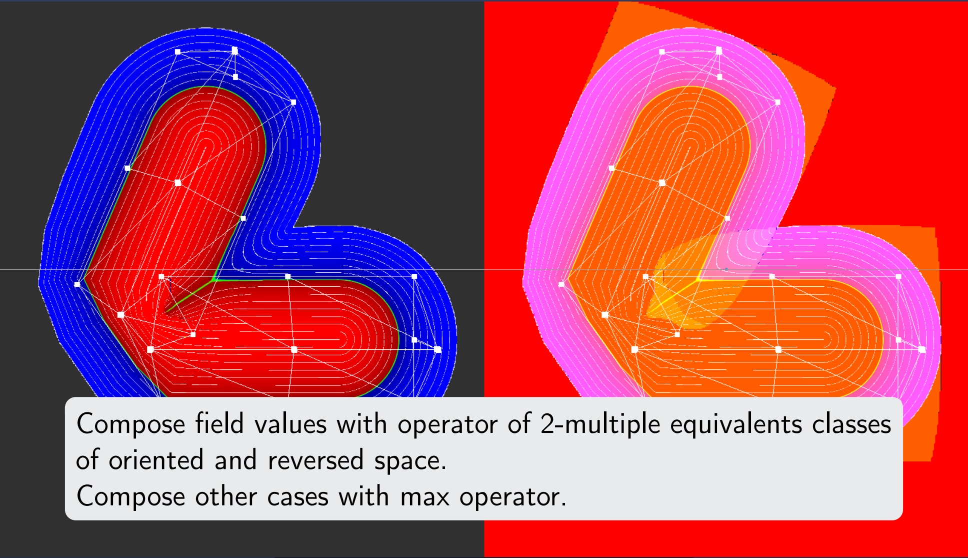


Due to non-commutativity of the contact operator.

Classification and dual composition

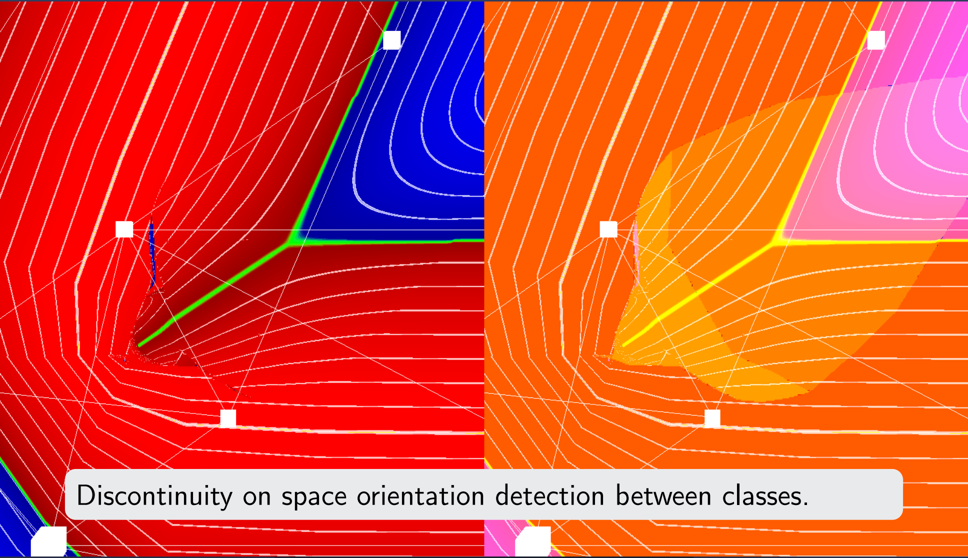


Classification and dual composition



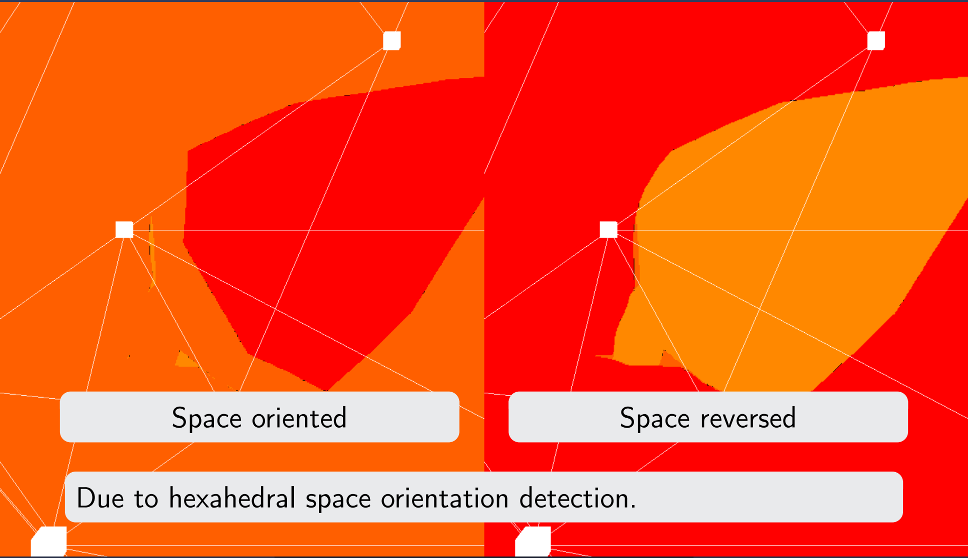
Compose field values with operator of 2-multiple equivalents classes of oriented and reversed space.
Compose other cases with max operator.

Classification and dual composition discontinuity

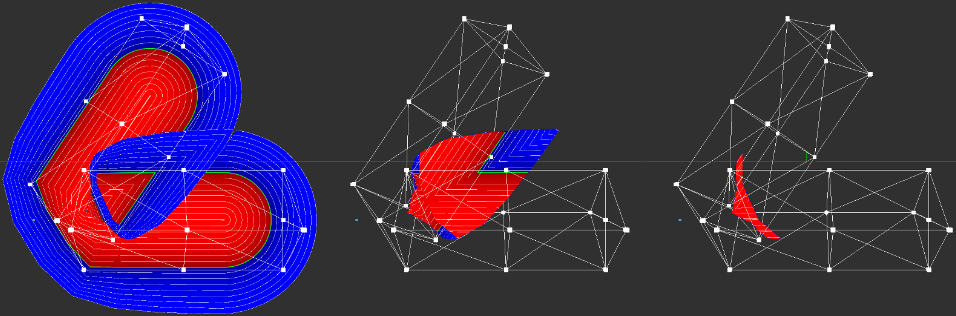


Discontinuity on space orientation detection between classes.

Classification and dual composition problem overview



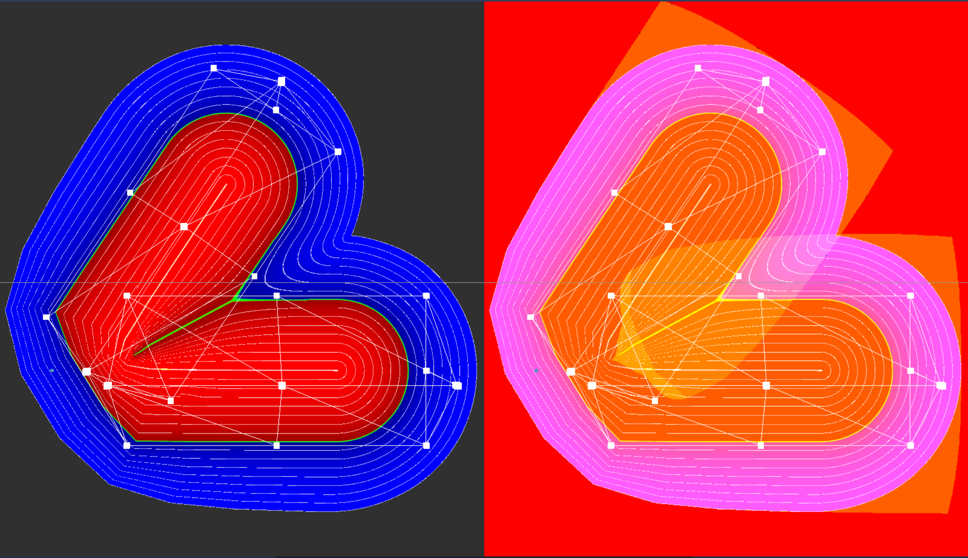
Implicit order : classes



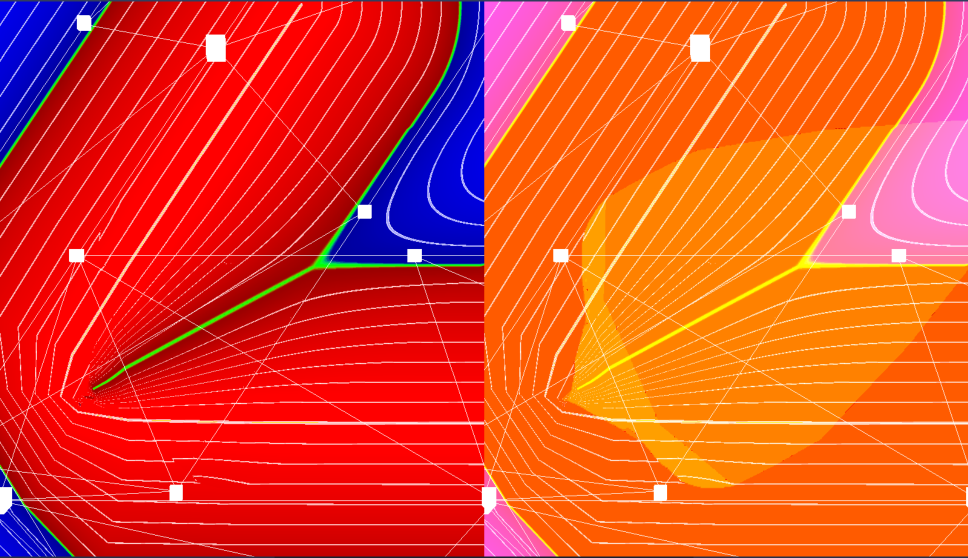
Minimal value | Median value | Maximal value

Sorting field value for each fetch.

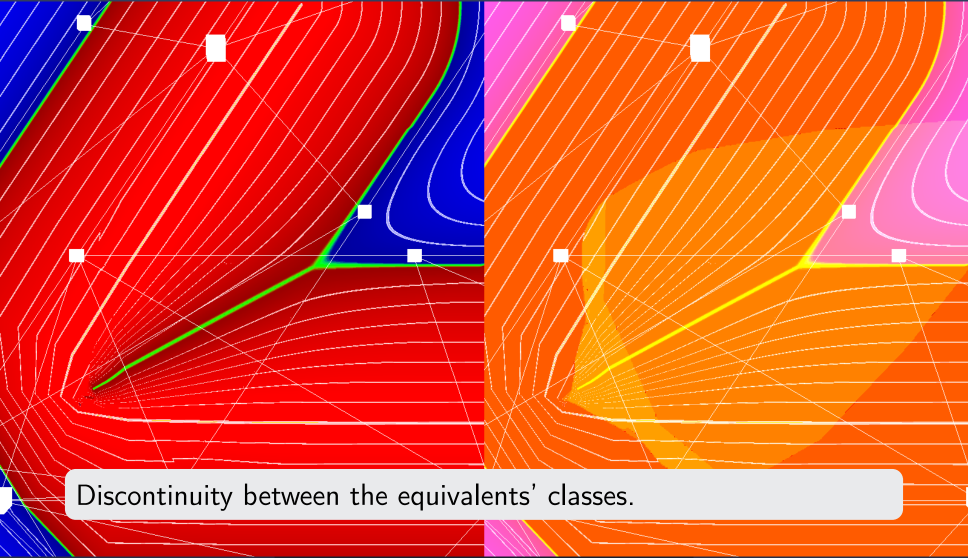
Implicit order : composition result



Implicit order : composition and discontinuities

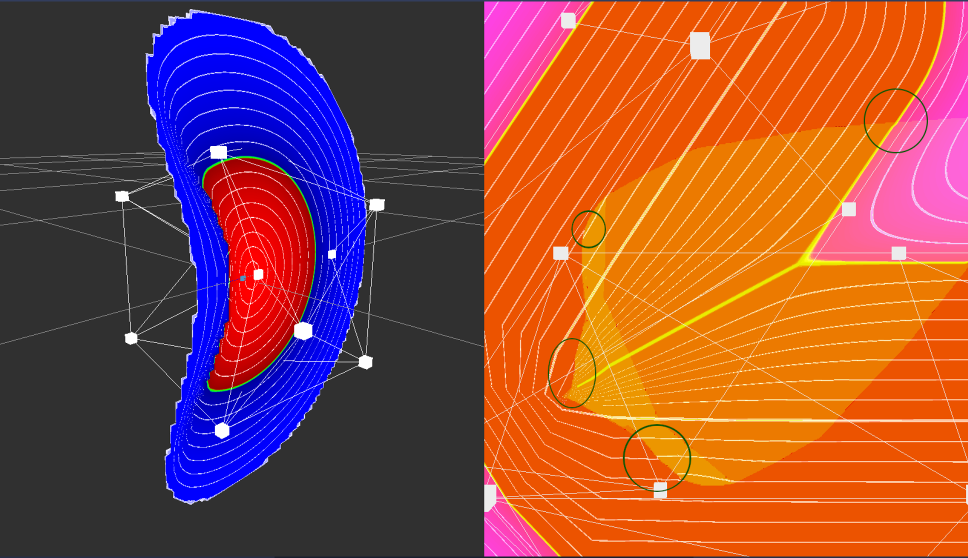


Implicit order : composition and discontinuities

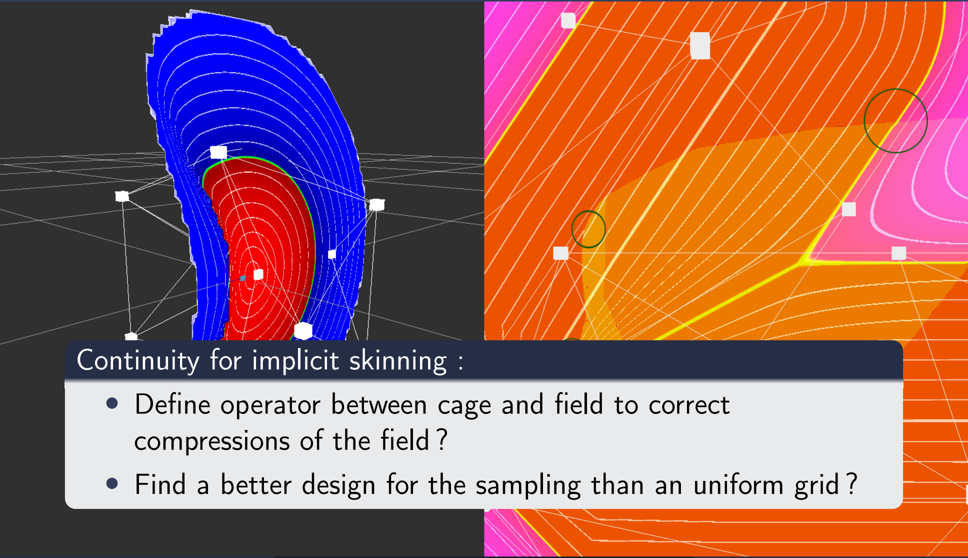


Discontinuity between the equivalents' classes.

Improve continuity of the method



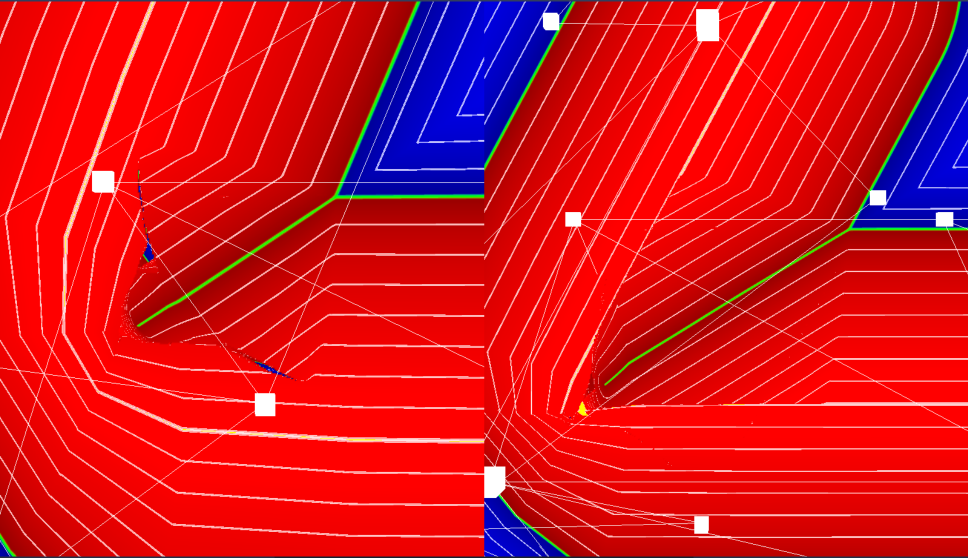
Improve continuity of the method



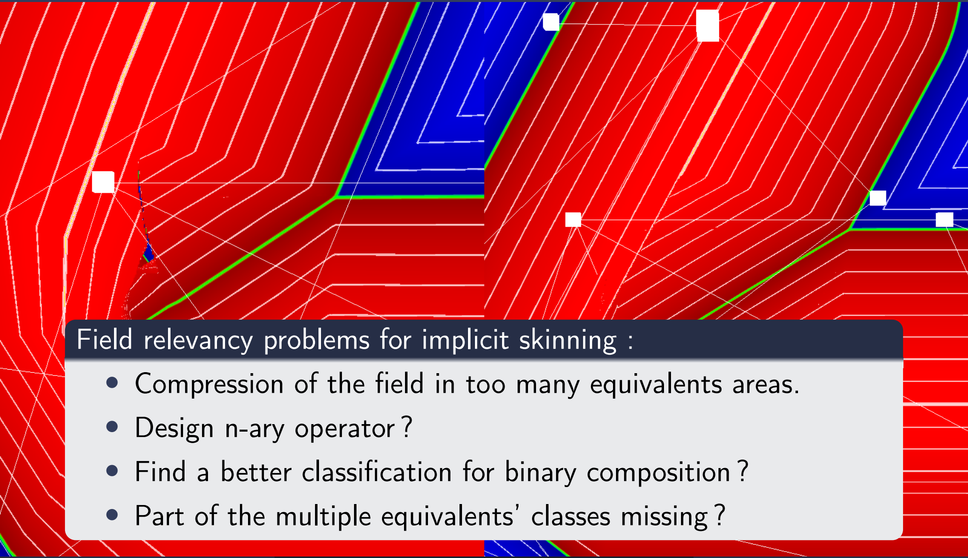
Continuity for implicit skinning :

- Define operator between cage and field to correct compressions of the field ?
- Find a better design for the sampling than an uniform grid ?

Limitation : Field reconstruction lack of relevancy



Limitation : Field reconstruction lack of relevancy



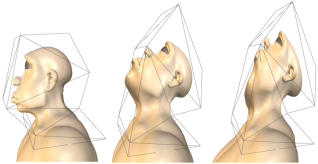
Field relevancy problems for implicit skinning :

- Compression of the field in too many equivalents areas.
- Design n-ary operator ?
- Find a better classification for binary composition ?
- Part of the multiple equivalents' classes missing ?

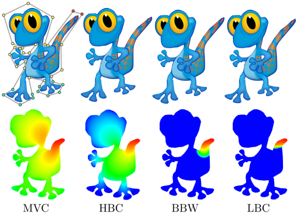
Test Green and Local barycentric coordinates systems

Test other barycentric coordinates system :

- Green coordinates for quasi-conformal transform in 3D.

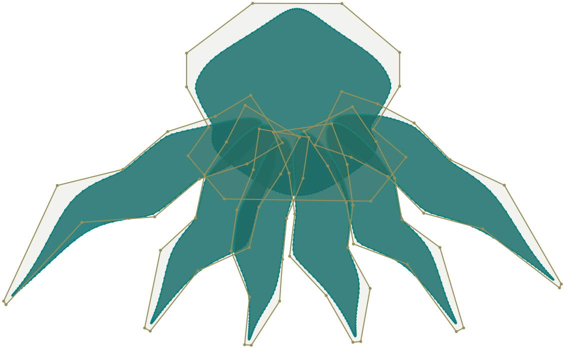


- Local barycentric coordinates for local transform of the space.



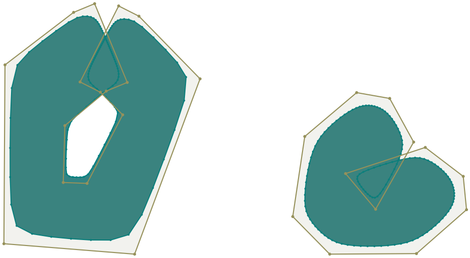
Integration to Implicit skinning

- Add smooth deformations.
- Allow free deformations for animators.



Correct mesh self-intersection

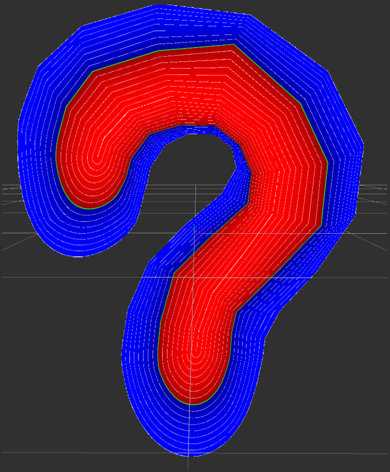
Case :	Self-contact	Self-intersection	· · ·
Solution :	contact operator	skinning operator	
Visualize :			



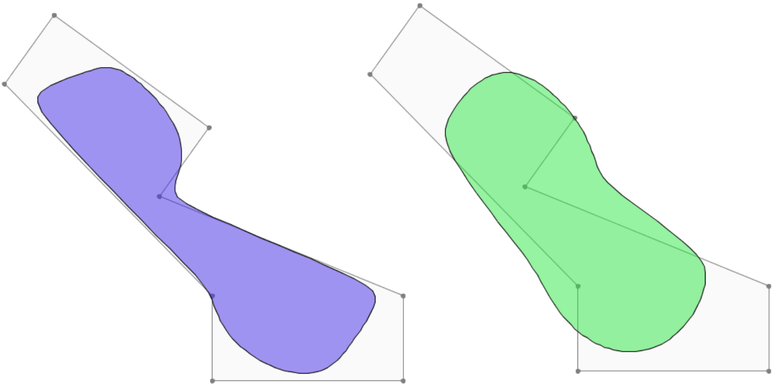
Conclusion

- Cages allow smooth and fast deformations we want to use to improve the Implicit skinning method.
- We propose a flexible plugin method to solve the inverse position problem and compute deformation of an implicit field.
- We are able to catch cases of multiple equivalents in original space to propose interesting solving of the implicit field in deformed space using composition operators.

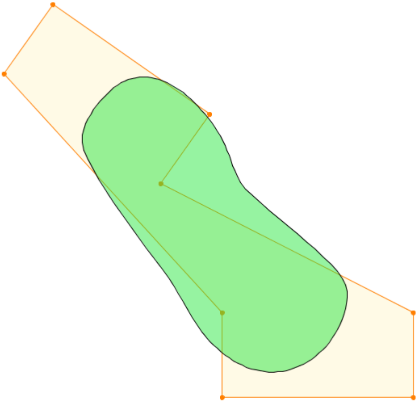
Questions ?



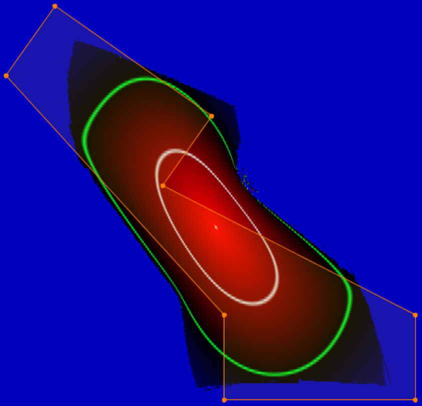
Green and MeanValue compare in 2d



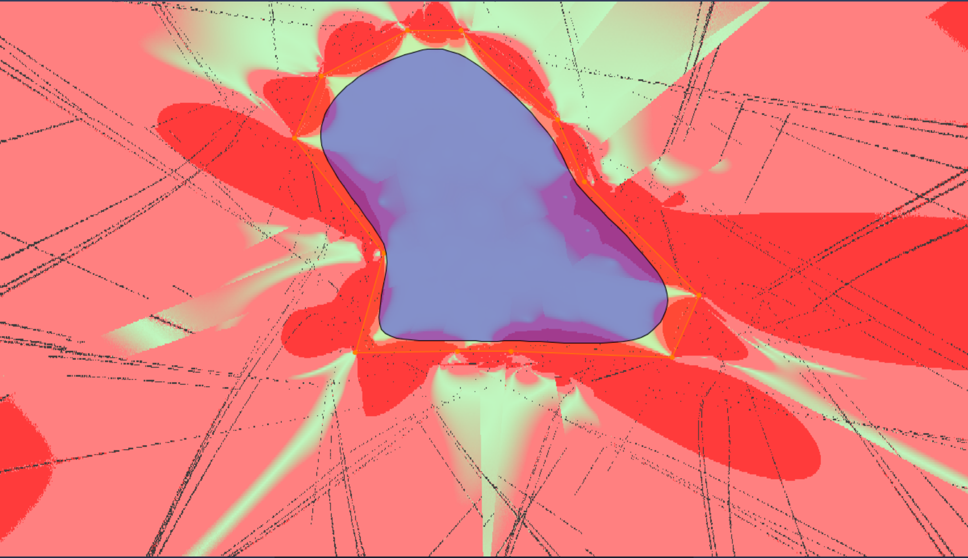
Green deform 2d



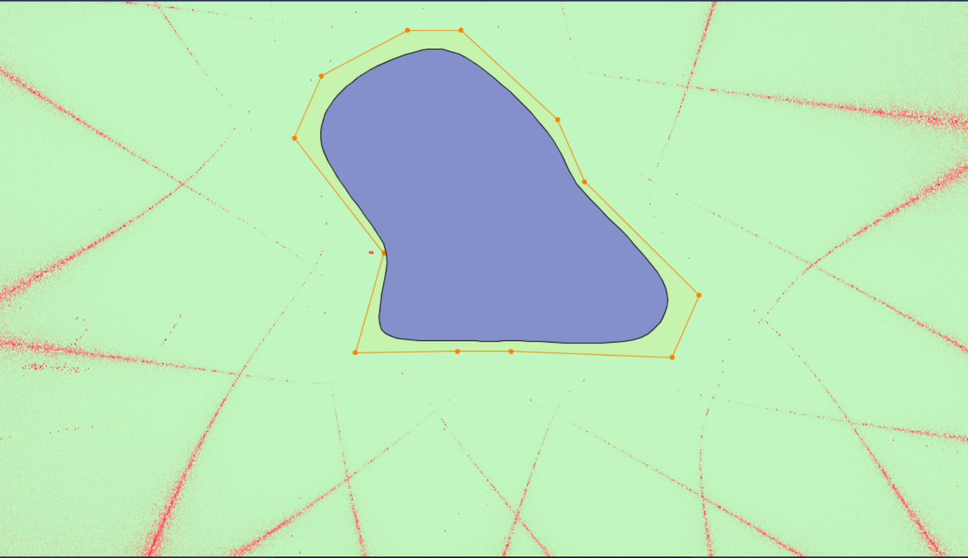
Green field 2d



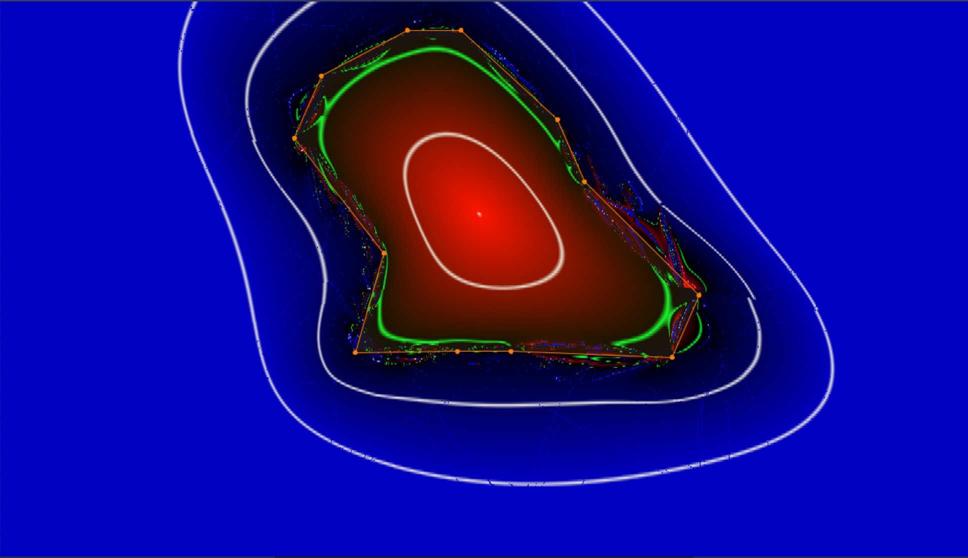
Using computed gradient : error



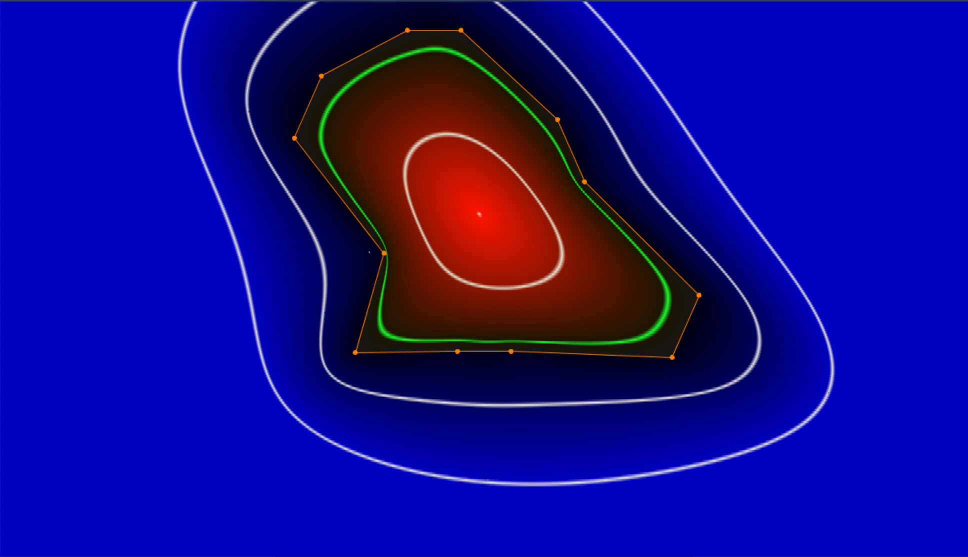
Using discrete gradient : error



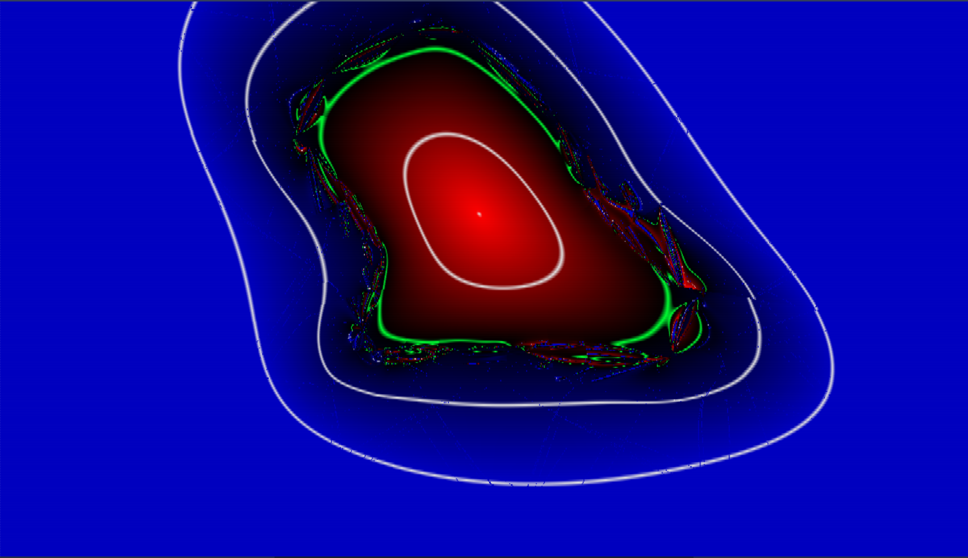
Using computed gradient : field



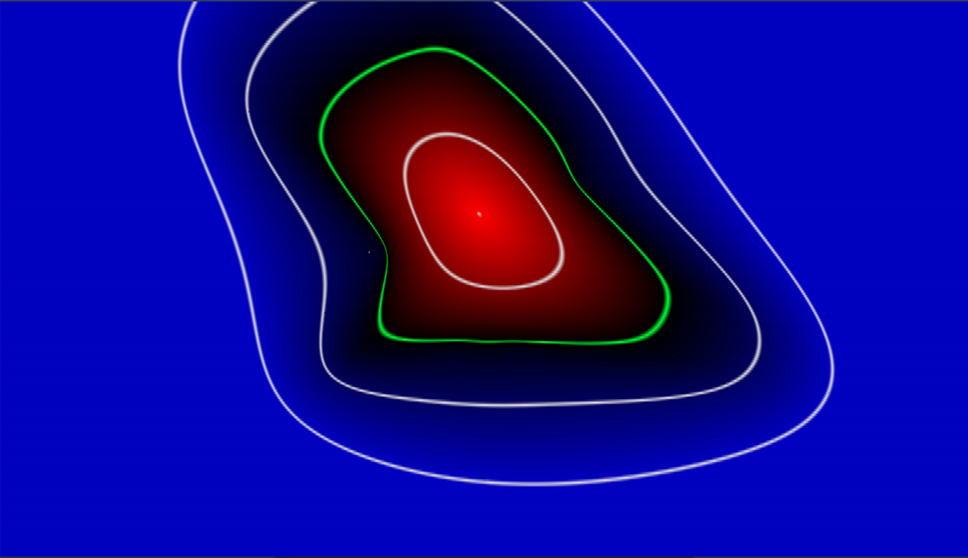
Using discrete gradient : field



Using computed gradient : field without cage



Using discrete gradient : field without cage



Gradient discontinuity on edges

$$a_i = C_i - P,\ b_i = C_{i+1} - P, \delta_i = \frac{\det(a_i, b_i)}{|\det(a_i, b_i)|} \frac{a_i \cdot b_i}{||a_i||||b_i||}$$

$$\omega_i = \frac{\tan\left(\frac{\delta_{i-1}}{2}\right) + \tan\left(\frac{\delta_i}{2}\right)}{||a_i||},\ P = \frac{\sum\limits_{i\in[n]} \omega_i C_i}{\sum\limits_{i\in[n]} \omega_i}$$

$$\frac{\partial}{\partial\gamma}\delta_i = \frac{\partial}{\partial\gamma} \frac{a_i \cdot b_i}{||a_i||||b_i||} = (a_i \cdot b_i) \frac{a_{i,\gamma}||b_i||^2 + b_{i,\gamma}||a_i||^2}{||a_i||^3||b_i||^3} - \frac{a_{i,\gamma} + b_{i,\gamma}}{||a_i||||b_i||}$$

$$\frac{\partial}{\partial\gamma} \tan\left(\frac{\delta_i}{2}\right) = \frac{1 + \tan^2\left(\frac{\delta_i}{2}\right)}{2} \sqrt{\frac{||a_i||||b_i||}{||a_i||||b_i|| - a_i \cdot b_i}} \frac{\partial}{\partial\gamma}\delta_i \frac{\det(a_i, b_i)}{|\det(a_i, b_i)|}$$

$$\frac{\partial}{\partial\gamma}\omega_i = \frac{\left(\frac{\partial}{\partial\gamma}\tan\left(\frac{\delta_{i-1}}{2}\right) + \frac{\partial}{\partial\gamma}\tan\left(\frac{\delta_i}{2}\right)\right)||a_i|| + \frac{a_{i,\gamma}}{||a_i||}\left(\tan\left(\frac{\delta_{i-1}}{2}\right) + \tan\left(\frac{\delta_i}{2}\right)\right)}{||a_i||^2}$$

$$\frac{\partial}{\partial\gamma}T(P) = \frac{\left(\sum \frac{\partial}{\partial\gamma}\omega_i T(C_i)\right)\left(\sum \omega_i\right) - \left(\sum \omega_i T(C_i)\right)\left(\sum \frac{\partial}{\partial\gamma}\omega_i\right)}{\left(\sum \omega_i\right)^2}$$

Gradient discontinuity on edges

$$a_i = C_i - P,\ b_i = C_{i+1} - P, \delta_i = \frac{\det(a_i, b_i)}{|\det(a_i, b_i)|} \frac{a_i \cdot b_i}{||a_i||||b_i||}$$

$$\omega_i = \frac{\tan\left(\frac{\delta_{i-1}}{2}\right) + \tan\left(\frac{\delta_i}{2}\right)}{||a_i||},\ P = \frac{\sum\limits_{i\in[n]} \omega_i C_i}{\sum\limits_{i\in[n]} \omega_i}$$

$$\frac{\partial}{\partial\gamma}\delta_i = \frac{\partial}{\partial\gamma}\frac{a_i \cdot b_i}{||a_i||||b_i||} = (a_i \cdot b_i)\frac{a_{i,\gamma}||b_i||^2 + b_{i,\gamma}||a_i||^2}{||a_i||^3||b_i||^3} - \frac{a_{i,\gamma} + b_{i,\gamma}}{||a_i||||b_i||}$$

$$\frac{\partial}{\partial\gamma}\tan\left(\frac{\delta_i}{2}\right) = \frac{1 + \tan^2\left(\frac{\delta_i}{2}\right)}{2}\sqrt{\frac{||a_i||||b_i||}{||a_i||||b_i|| - a_i \cdot b_i}}\frac{\partial}{\partial\gamma}\delta_i \frac{\det(a_i, b_i)}{|\det(a_i, b_i)|}$$

$$\frac{\partial}{\partial\gamma}\omega_i = \frac{\left(\frac{\partial}{\partial\gamma}\tan\left(\frac{\delta_{i-1}}{2}\right) + \frac{\partial}{\partial\gamma}\tan\left(\frac{\delta_i}{2}\right)\right)||a_i|| + \frac{a_{i,\gamma}}{||a_i||}\left(\tan\left(\frac{\delta_{i-1}}{2}\right) + \tan\left(\frac{\delta_i}{2}\right)\right)}{||a_i||^2}$$

$$\frac{\partial}{\partial\gamma}T(P) = \frac{\left(\sum \frac{\partial}{\partial\gamma}\omega_i T(C_i)\right)\left(\sum \omega_i\right) - \left(\sum \omega_i T(C_i)\right)\left(\sum \frac{\partial}{\partial\gamma}\omega_i\right)}{\left(\sum \omega_i\right)^2}$$

Gradient descent in Coordinates space

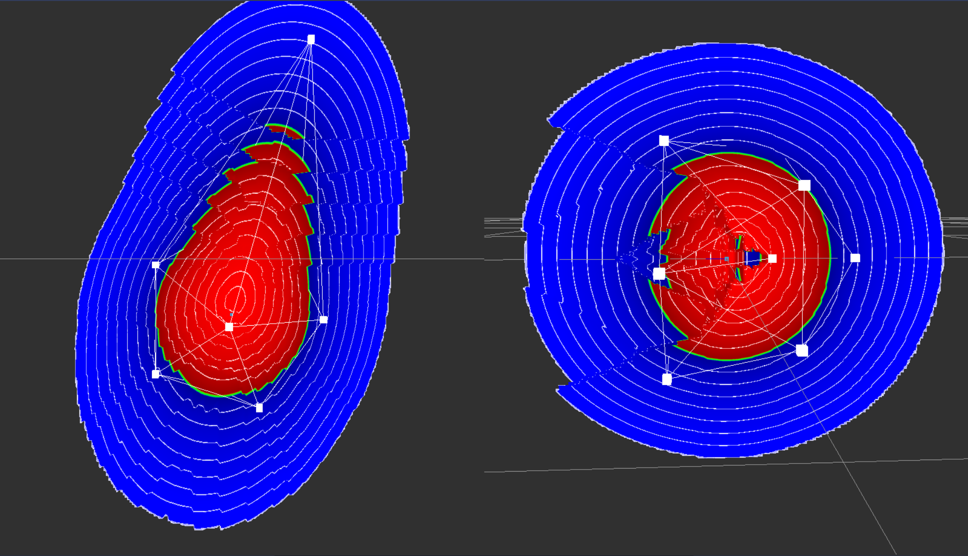
$$\text{Minimize } || \sum_{i \in [n]} \alpha_i T(C_i) - Q||^2.$$

$$\frac{\partial}{\partial \alpha_k} || \sum_{i \in [n]} \alpha_i T(C_i) - Q||^2 = 2T(C_k) \cdot \left(\sum_{i \in [n]} \alpha_i T(C_i) - Q \right)$$

$$\nabla || \sum_{i \in [n]} \alpha_i T(C_i) - Q||^2 = 2 \quad {}^t \left(T(C_1), T(C_2), \ldots, T(C_n) \right) \cdot \left(\sum_{i \in [n]} \alpha_i T(C_i) - Q \right)$$

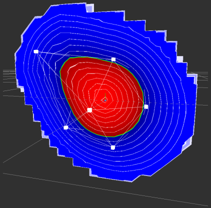
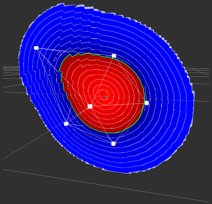
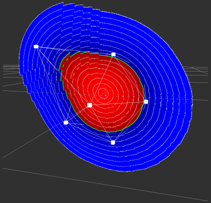
Do gradient descent.

Coordinates space gradient descent : result



Coordinates space gradient descent : benchmark display planes

Implicit blob sphere deformed by cage (12 vertices, 20 faces) :

resolution	32×32 (2^{10})	128×128 (2^{14})	512×512 (2^{18})
time	$1 \cdot 10^7 ns (10ms)$	$1.58 \cdot 10^8 ns (158ms)$	$2.4 \cdot 10^9 ns (2.4s)$
visualize			

Gradient descent in Coordinates space

Find $\{\mathfrak{C}_j\}_{j\in[l]}$ orthonormal basis of constraints from coordinates system.
Then do gradient descent on :

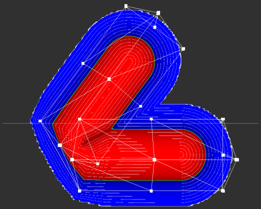
$$\nabla_{\mathfrak{C}}||\sum_{i\in[n]}\alpha_i T(C_i) - Q||^2 = \nabla||\sum_{i\in[n]}\alpha_i T(C_i) - Q||^2 - \sum_{j\in[l]}\frac{\nabla||\sum_{i\in[n]}\alpha_i T(C_i) - Q||^2 \cdot \mathfrak{C}_j}{\mathfrak{C}_j \cdot \mathfrak{C}_j}\mathfrak{C}_j$$

- Is constraints computation that relevant for time saving ?
- Nearest neighbor's like method : no multiple equivalent solving.
- Benchmark times near to Bounding grid methods.

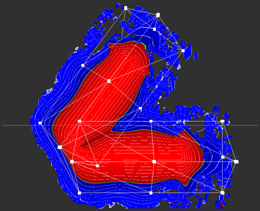
Reconstruction compare : fails

Implicit blob capsule deformed by cage (26 vertices, 48 faces) :

Trilinear reconstruction
$2.16 \cdot 10^8 ns (216ms)$



Lagrange reconstruction
$2.56 \cdot 10^{10} ns (25.6s)$



Bezier reconstruction
$8.35 \cdot 10^{10} ns (83.5s)$