

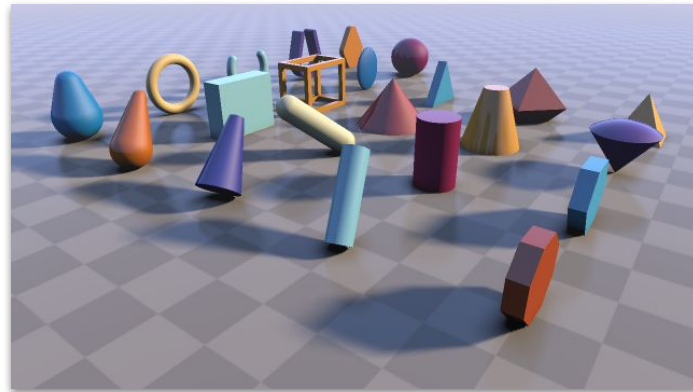
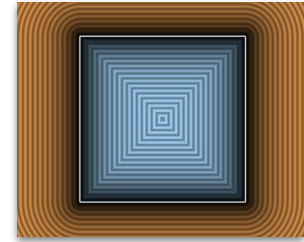
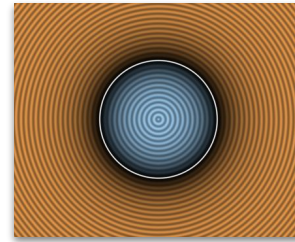
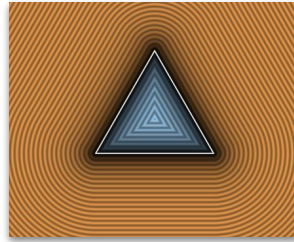
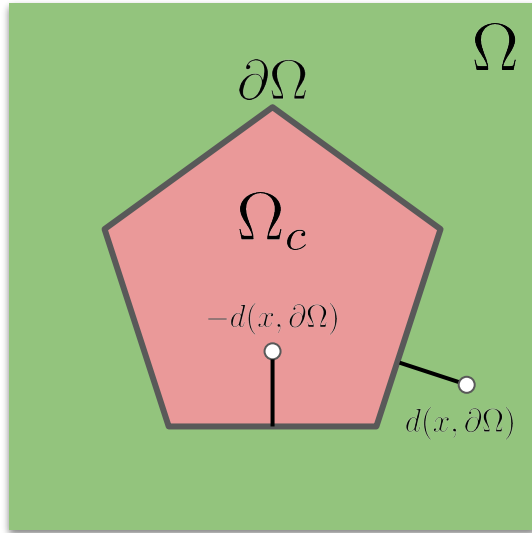
Local Optimization for Robust Signed Distance Field Collision (2019)

Miles Macklin, Kenny Erleben, Matthias Müller, Nuttapong Chentanez, Stefan Jeschke, Zack Corse



Signed Distance Function

$$\psi(x) = \begin{cases} d(x, \partial\Omega) & , \text{if } x \in \Omega \\ -d(x, \partial\Omega) & , \text{if } x \in \Omega_c \end{cases}$$



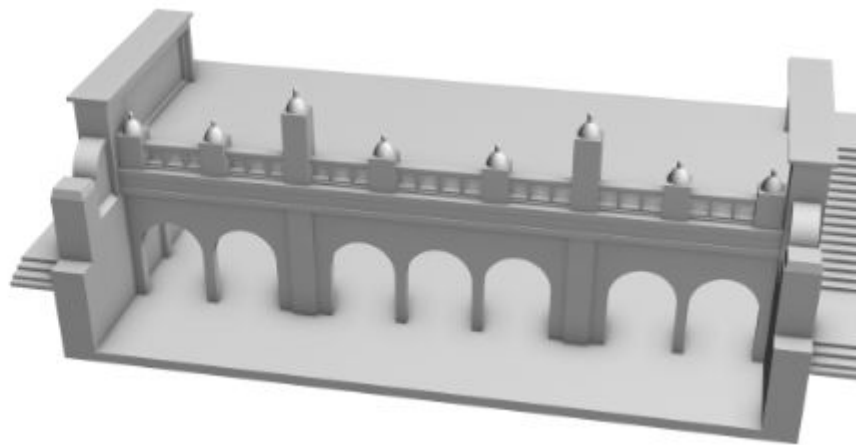
Some use cases

Massively Parallel Rendering of Complex Closed-Form Implicit Surfaces (2020)

Matthew J. Keeter



SDF GPU Rendering



Controllable rendering quality

Improved Alpha-Tested Magnification for Vector Textures and Special Effects (2007)

Chris Green



4096x4096
(binary)



64x64
(sdf)



In Game

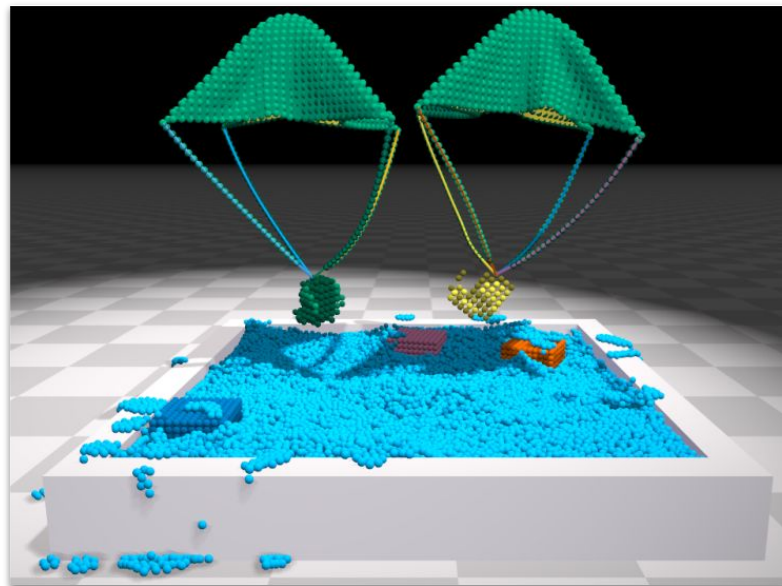
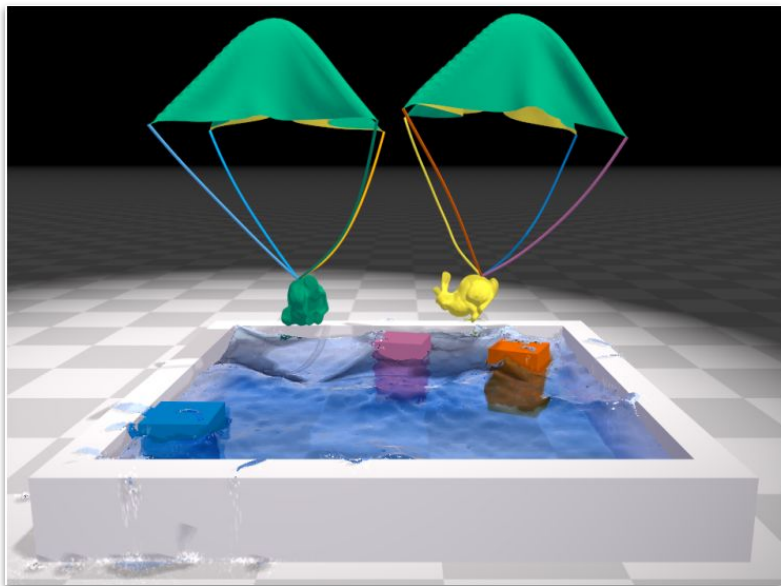
Keep sharp features
with low resolution

Offer lot of control

Can't represent too
complex texture ?

Unified Particle Physics for Real-Time Applications (2014)

Miles Macklin Matthias Müller Nuttapong Chentanez Tae-Yong Kim

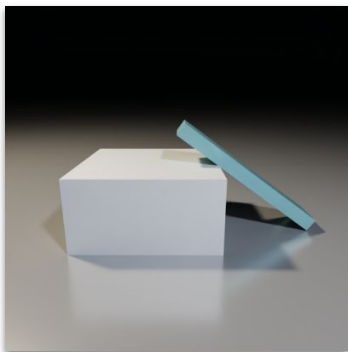
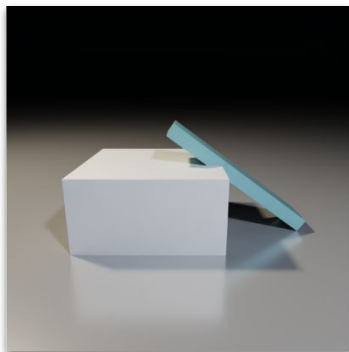
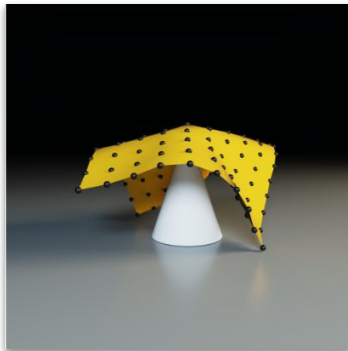


Collisions with SDF (kinda)
with gradient sampling to avoid interlocking

Back to collisions

Collisions in simulation

Point Based vs Optimisation



Handle precisely collision

(avoid interpenetration)

Gives valid positions

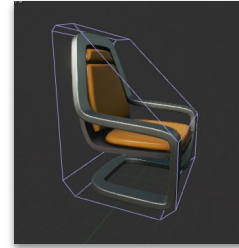
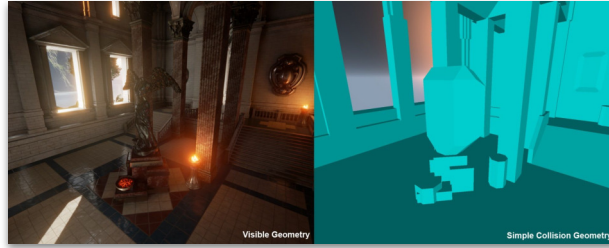
Can handle complex shapes

Low Cost (if possible)



Methods

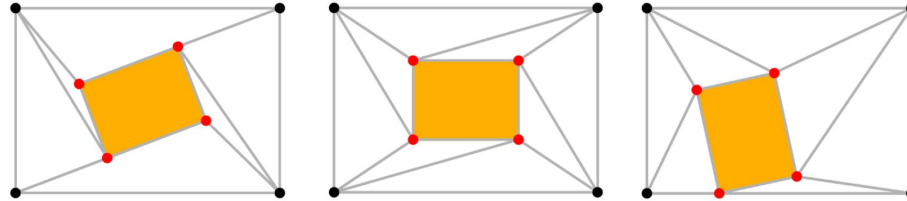
Primitives



Fast and Easy

Poor Shape Representation

Air Meshes

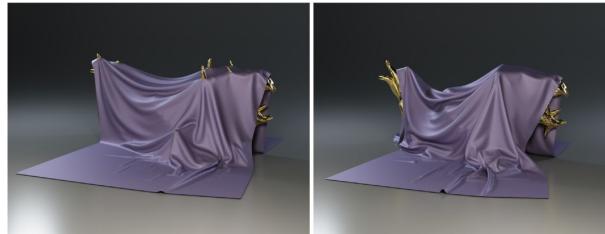


Air Meshes for Robust Collision Handling (2015)

Robust Collision

Static Data Structure

SDF



Very Precise

A bit tricky to use

Collisions With SDF

$$y = x - \nabla \phi(x) \phi(x)$$

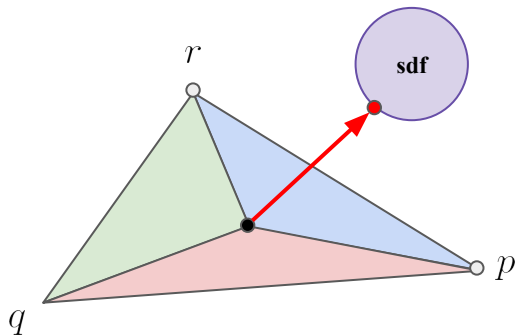
$$n = \nabla \phi(x)$$

Face Contact

$$\arg \min_{u,v,w} \psi(up + vq + wr)$$

$$\boxed{u} + \boxed{v} + \boxed{w} = 1$$

$$u, v, w \geq 0$$



Projected Gradient Descent

$$\mathbf{d} = \begin{bmatrix} \frac{\partial \phi}{\partial u} & \frac{\partial \phi}{\partial v} & \frac{\partial \phi}{\partial w} \end{bmatrix}^T = \begin{bmatrix} \nabla \phi(\mathbf{x}) \mathbf{p} \\ \nabla \phi(\mathbf{x}) \mathbf{q} \\ \nabla \phi(\mathbf{x}) \mathbf{r} \end{bmatrix}$$

$$\mathbf{c}_{i+1} \leftarrow \mathbb{P}(\mathbf{c}_i - \alpha \mathbf{d}) \quad \mathbf{c} = [u, v, w]^T$$

or

Frank-Wolfe

$$\underset{\mathbf{s}_i}{\operatorname{argmin}} \quad \mathbf{s}_i^T \nabla \phi(\mathbf{x}_i) \quad \text{s.t.} \quad \mathbf{s}_i \in \mathcal{D}$$

$$\boxed{s_i \in \{r, q, p\}}$$

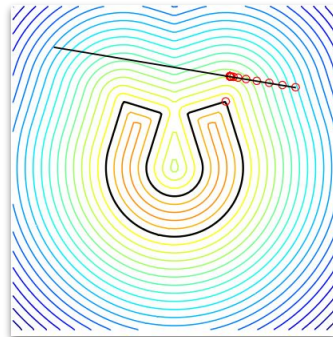
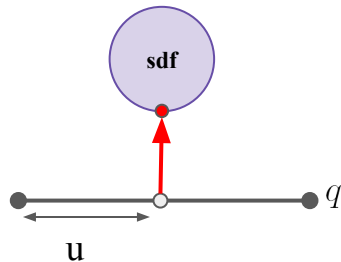
$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \alpha(\mathbf{s}_i - \mathbf{x}_i) \quad \alpha = \frac{2}{i+2}$$

Collisions With SDF

Edge Contact

$$\underset{u}{\operatorname{argmin}} \quad \phi(up + (1 - u)q)$$

$$\text{s.t.} \quad u \geq 0 \quad u \leq 1$$



or

Projected Gradient Descent

or

Frank-Wolfe

Golden-section Search (easy)

$$d = \frac{\sqrt{5} - 1}{2}(b - a) \quad \begin{array}{l} u_a = a + d \\ u_b = b - d \end{array}$$

$$x_a = u_a p + (1 - u_a) q$$

$$x_b = u_b p + (1 - u_b) q$$

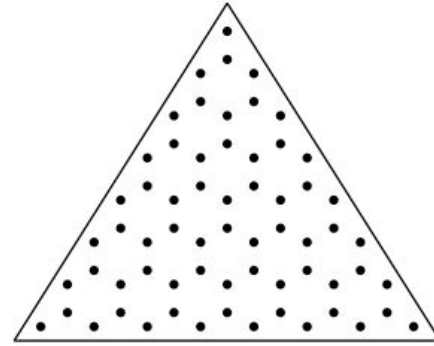
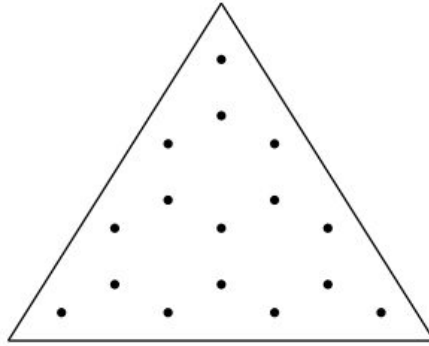
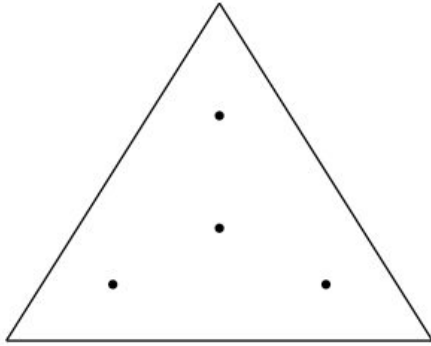
$$\phi(x_a) > \phi(x_b) \quad \Bigg| \quad \phi(x_a) < \phi(x_b)$$

$$b = x_1$$

$$a = x_2$$

Repeat

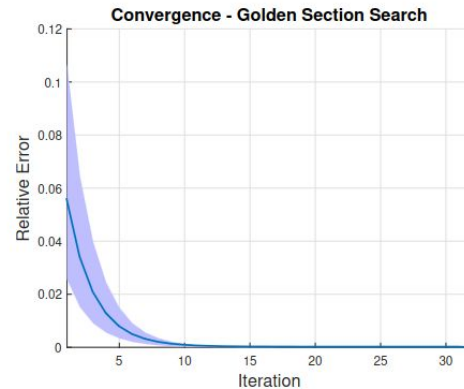
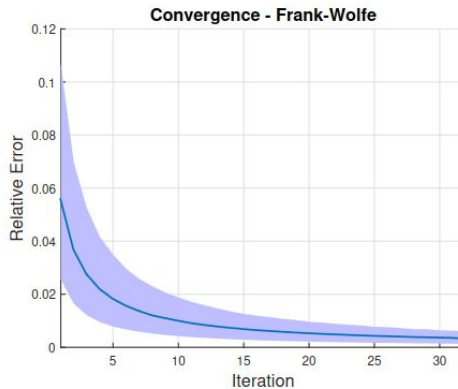
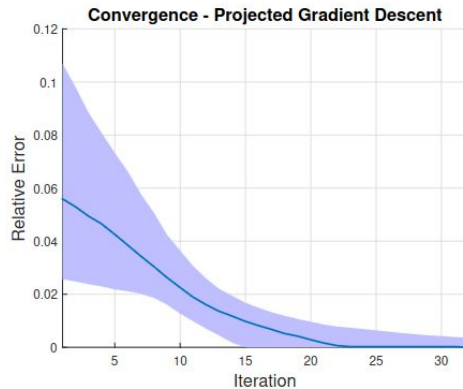
Point Based VS Optimisation



More test in practice

Miss Sharp Features

Convergence Rate ?

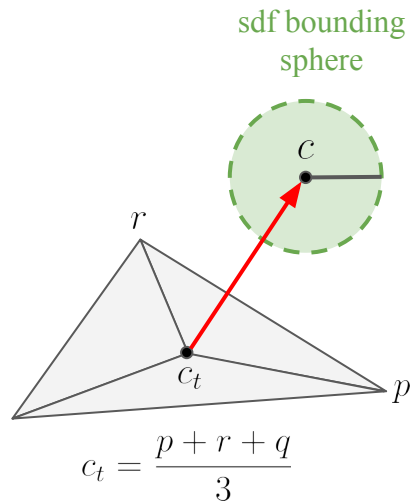


Place randomly a segment 2D segment in domain then do minimisation with a SDF
Relative error generated with high resolution brute-force sampling over the element (line/triangle)

Example	SDF	Points	Elements	Simple	PGD	FW	GS
Unit	#	#	#	μs	μs	μs	μs
Dragon	256^3	20,000	39,402	20	85	77	-
Net	256^3	51,161	81,194	336	373	321	-
Teapot	128^3	5,281	6,962	112	123	115	-
Shells	512^3	635,020	1,269,880	401	445	437	-
Rope	256^3	4224	4096	98	102	99	99
Analytic	-	10,000	19,502	48	48	48	-

Some details

“Culling” Phase



No collisions if
 $||c_t - c|| - radius > 0$

No BVH, Grid, KD Tree, Etc ...

Dynamic Objects

$$\phi_d = \phi_m(g(\mathbf{x}_s))$$

Trivial for rigid bodies
(We have a global transform)

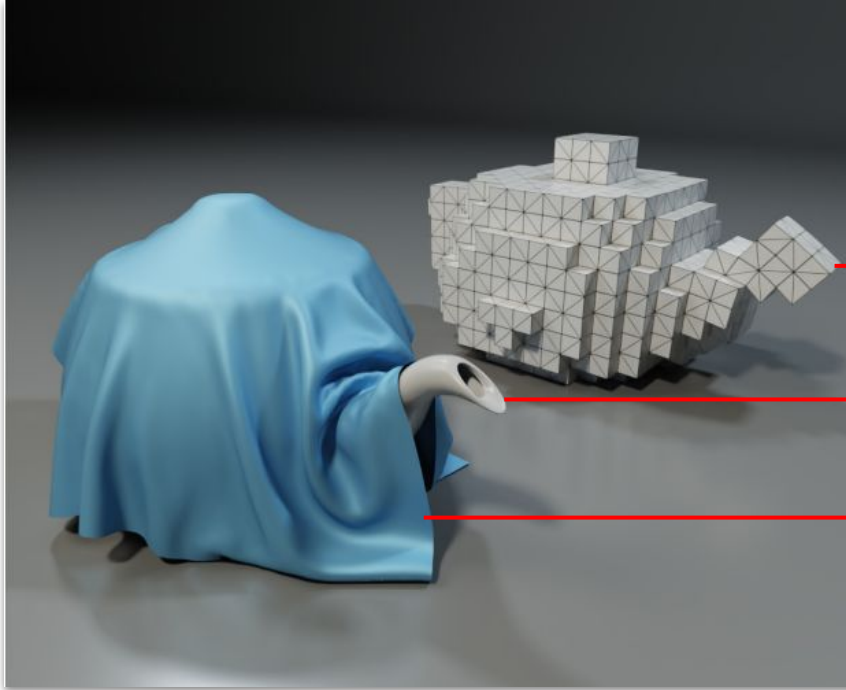
FEM Tetra

$$\mathbf{x}_m = g(\mathbf{x}_s) = \mathbf{M}\mathbf{D}^{-1}(\mathbf{x}_s - \mathbf{v}_{0s}) + \mathbf{v}_{0m}$$

$$\boxed{\mathbf{M}} = \begin{bmatrix} \mathbf{v}_{1m} - \mathbf{v}_{0m} & \mathbf{v}_{2m} - \mathbf{v}_{0m} & \mathbf{v}_{3m} - \mathbf{v}_{0m} \end{bmatrix}$$
$$\boxed{\mathbf{D}} = \begin{bmatrix} \mathbf{v}_{1s} - \mathbf{v}_{0s} & \mathbf{v}_{2s} - \mathbf{v}_{0s} & \mathbf{v}_{3s} - \mathbf{v}_{0s} \end{bmatrix}$$

Computed FEM in simulation anyway

Some details



Deformed Mesh with FEM

Deformed SDF

Cloth collide with Deformed SDF

Conclusion

Crazy Fast with limited use of
accelerating structures

Precise and can handle sharp features

GPU Friendly

Signed Field Function

Need a SDF representation of the mesh

Hard to get SDF gradient
(generally precomputed)

Signed Field Function

Collision with [Volume $\Leftrightarrow$ Cloth] and [Volume $\Leftrightarrow$ Volume], but does [Cloth $\Leftrightarrow$ Cloth] works ?

J'ai oublié le reste

Video

Bibliography

[Miles Macklin ; Local Optimization for Robust Signed Distance Field Collision](#)

[Matthew J. Keeter : Massively parallel rendering of complex closed-form implicit surfaces](#)

[Chris Green : Improved Alpha-Tested Magnification for Vector Textures and Special Effects](#)

[Matthias Muller : Air Meshes for Robust Collision Handling](#)