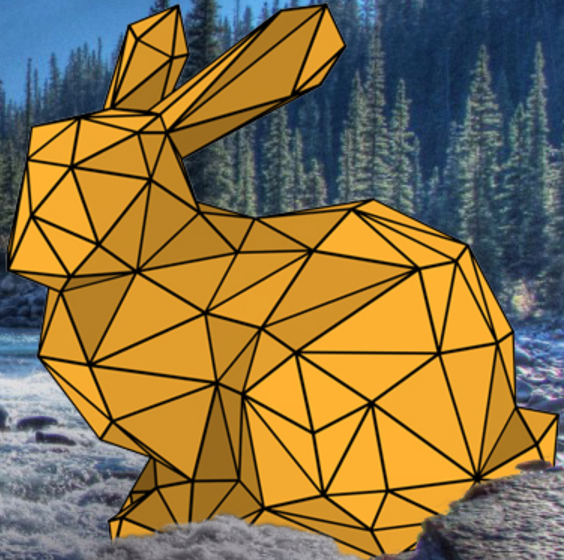
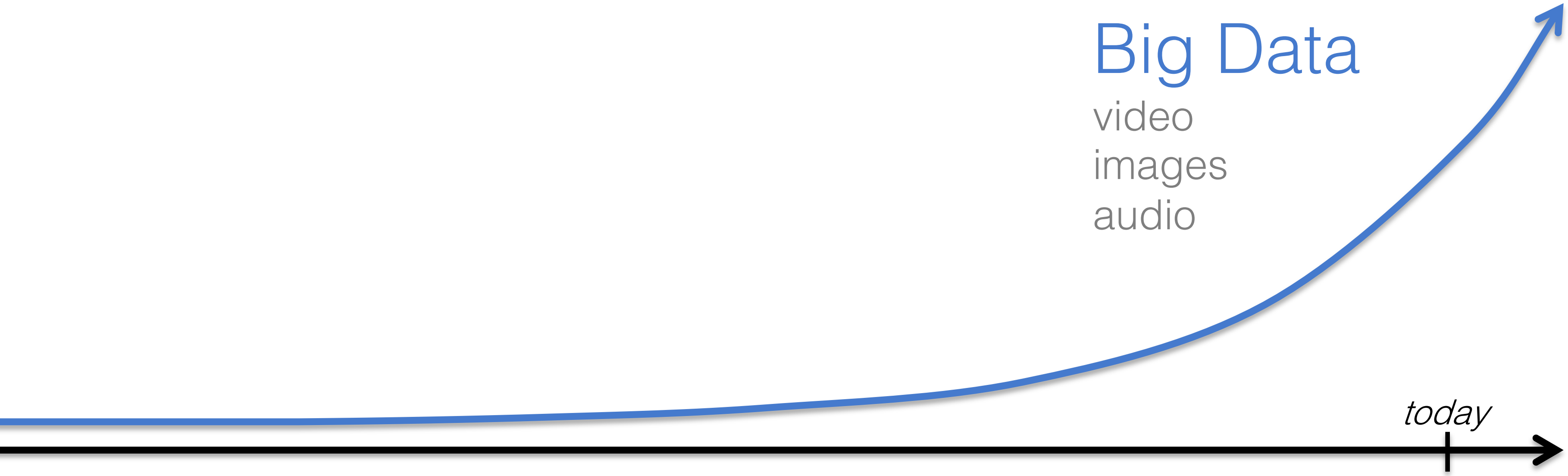
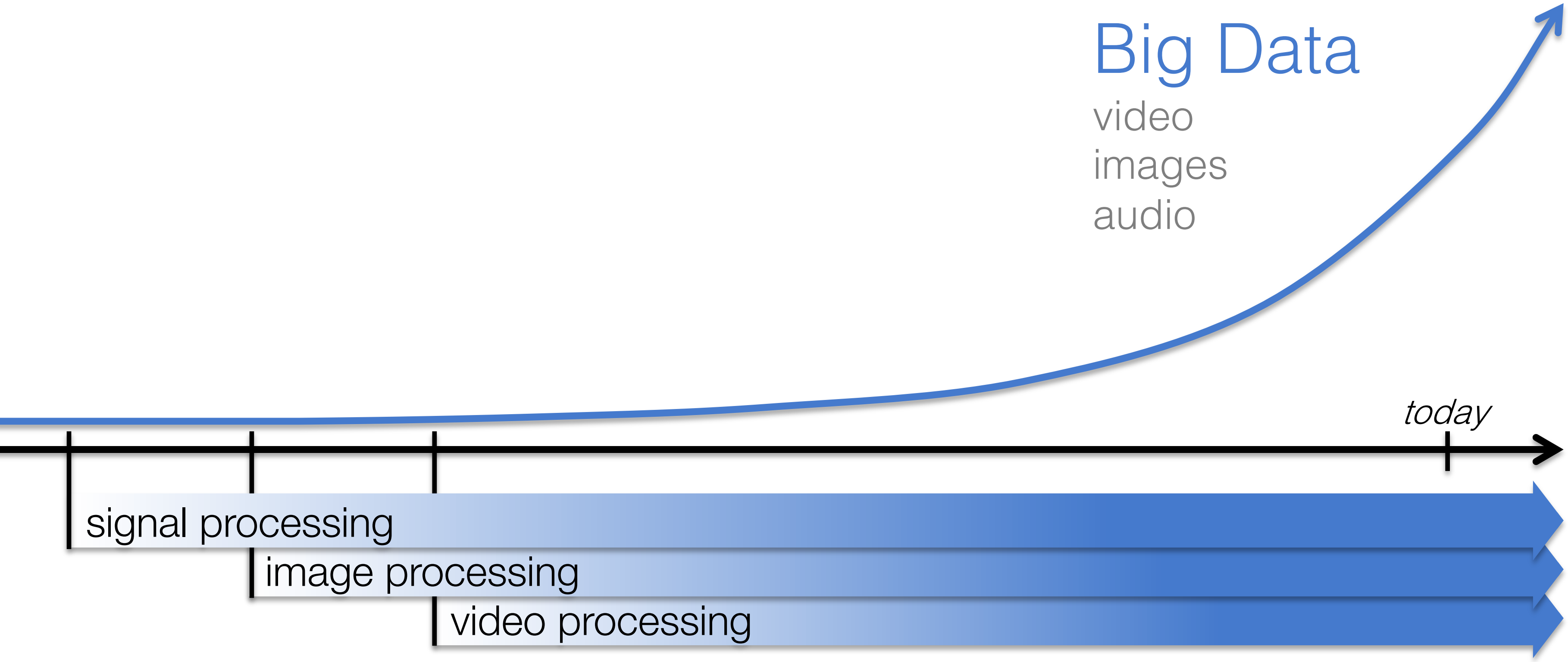
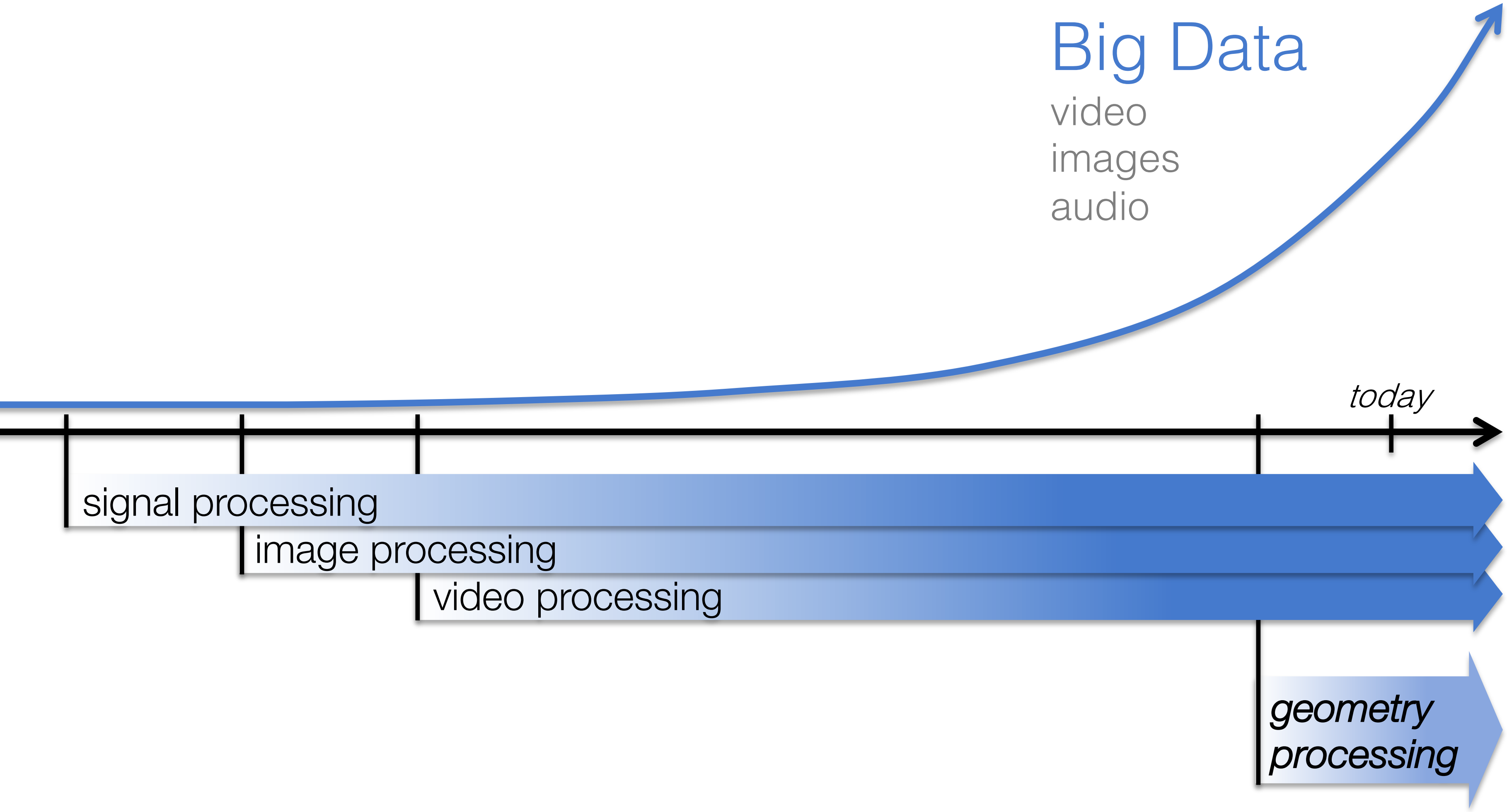
Geometry Processing in the Wild

Alec Jacobson
Canada Research Chair
University of Toronto



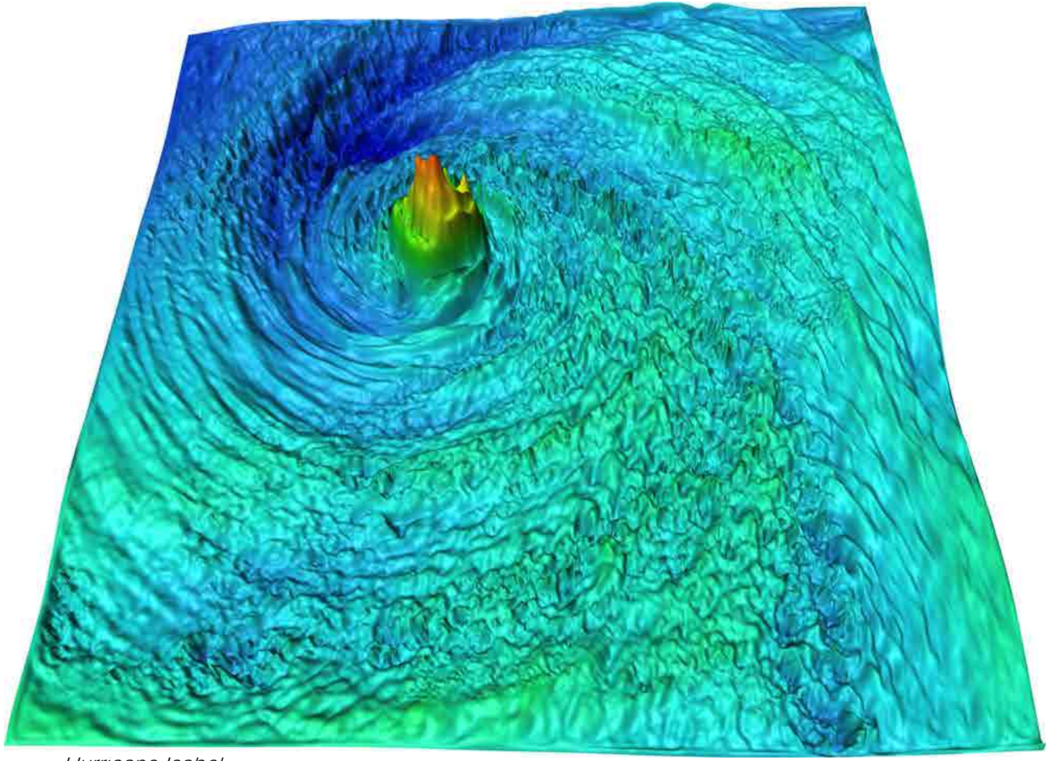

Big Data
video
images
audio
today


Big Data
video
images
audio
today
signal processing
image processing
video processing

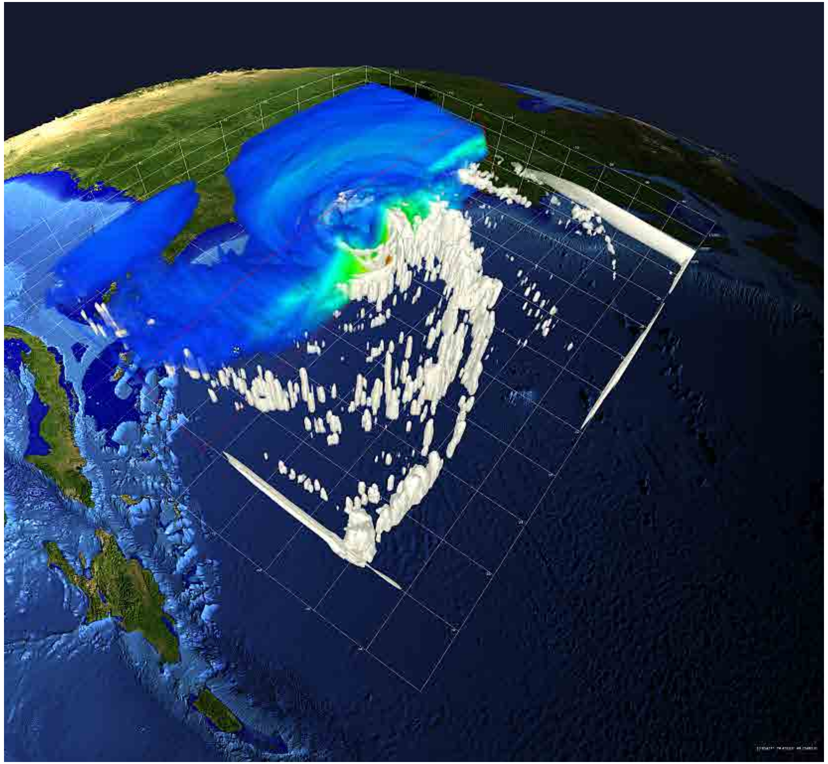


Big Data
video
images
audio
today
signal processing
image processing
video processing
geometry processing

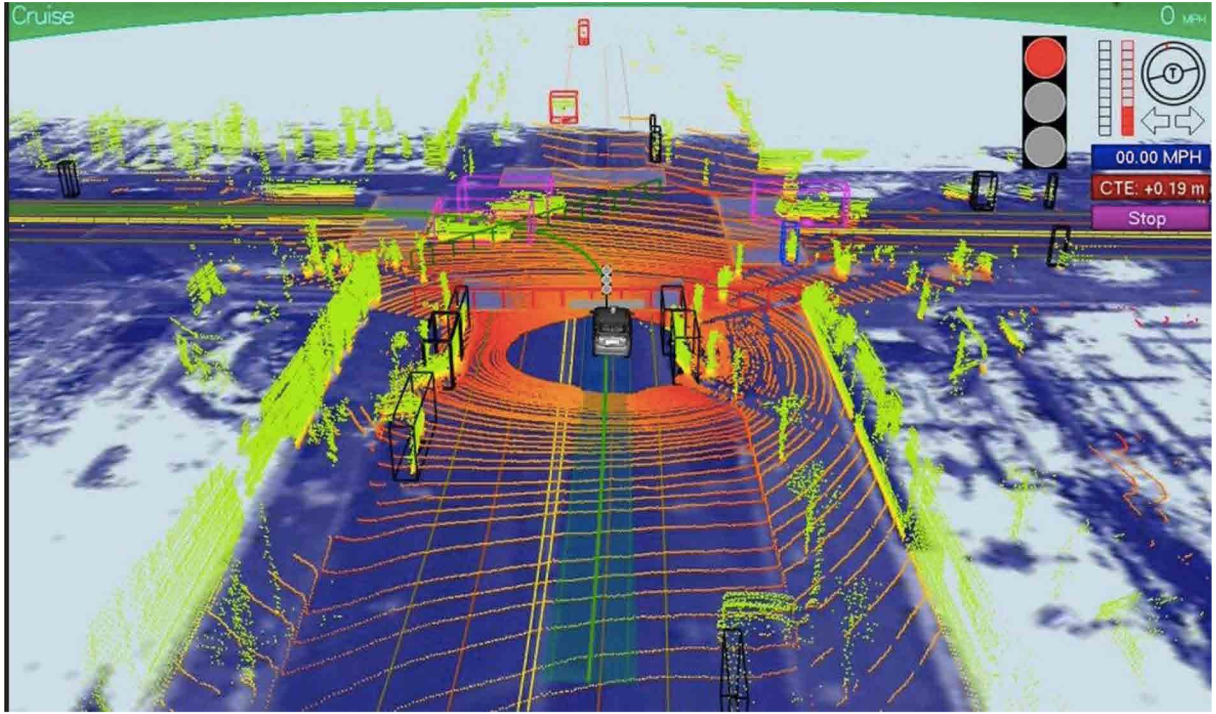
Geometric data are everywhere



Hurricane Isabel



Geometric data are everywhere



Google Self-Driving Car

The New York Times

Self-Driving Uber Car Kills Pedestrian in Arizona, Where Robots Roam

A woman crossing Mill Avenue at its intersection with Curry Road in Tempe, Ariz., on Monday. A pedestrian was struck and killed by a self-driving Uber vehicle at the intersection a night earlier. Caitlin O'Hara for The New York Times

By Daisuke Wakabayashi

March 19, 2018

1921

Leer en español

SAN FRANCISCO — Arizona officials saw opportunity when Uber and other companies began testing driverless cars a few years ago. Promising to keep oversight light, they invited the companies to test their robotic vehicles on the state's roads.

Then on Sunday night, an autonomous car operated by Uber — and with an emergency backup driver behind the wheel — struck and killed a woman on a street in Tempe, Ariz. It was believed to be the first pedestrian death associated with self-driving technology. The

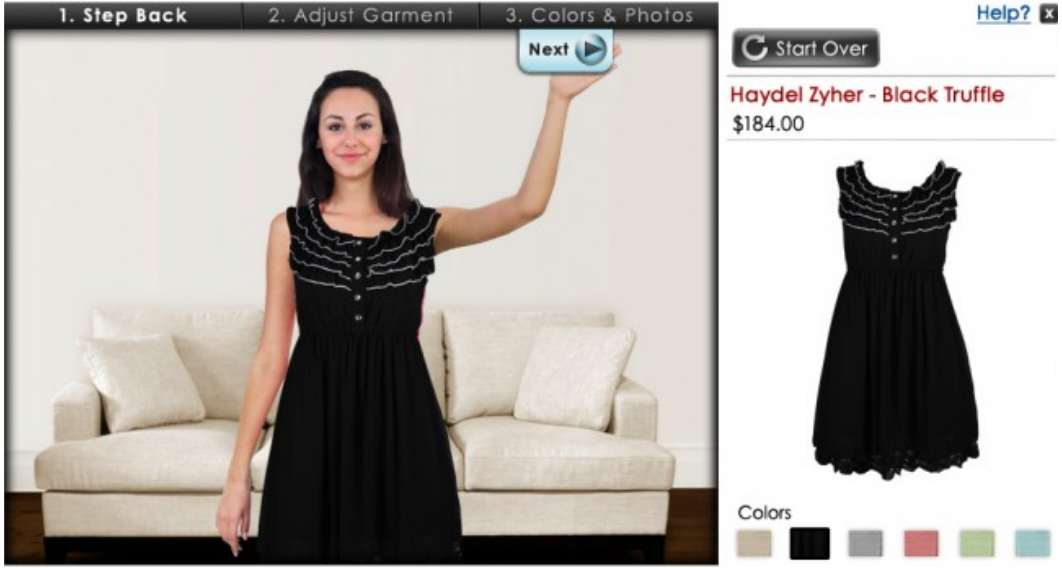
Geometric data are everywhere

Siemens Medical

R Loganathan

Geometric data are everywhere

Intel RealSense D415



Zugara

Geometric data are everywhere

thealternativelimbproject.com

New Balance

Geometric data are everywhere

Calvino Noir

Red Dead Redemption

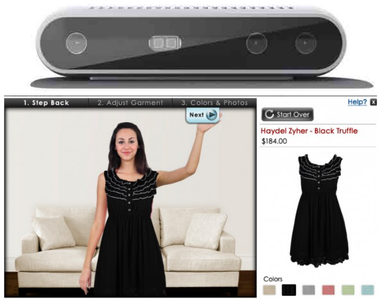
Geometric data are everywhere

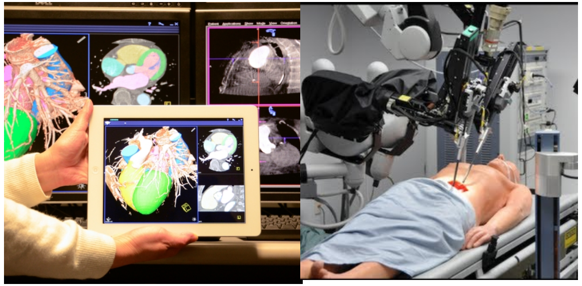
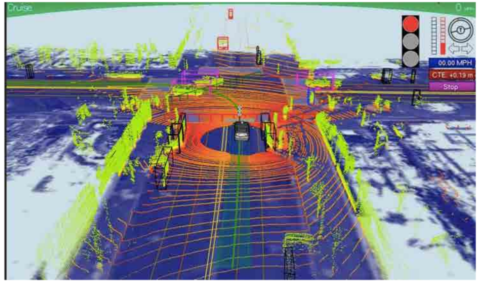
Ugly Betty

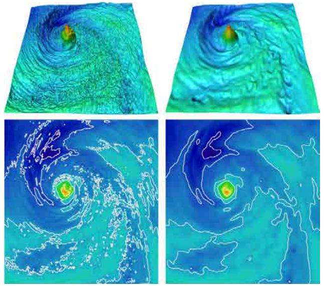
El secreto de sus ojos

All require collecting, processing and using geometric data

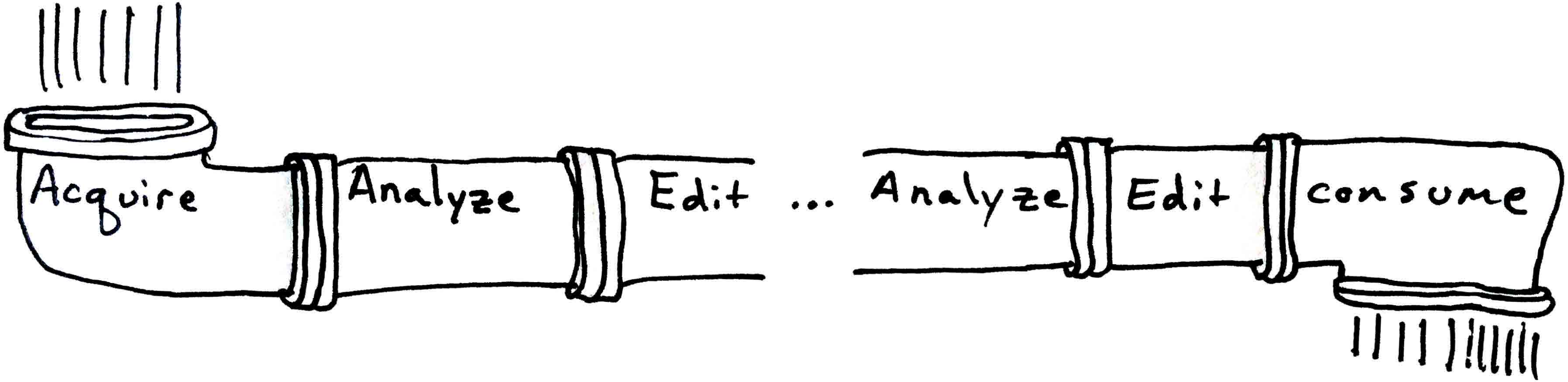


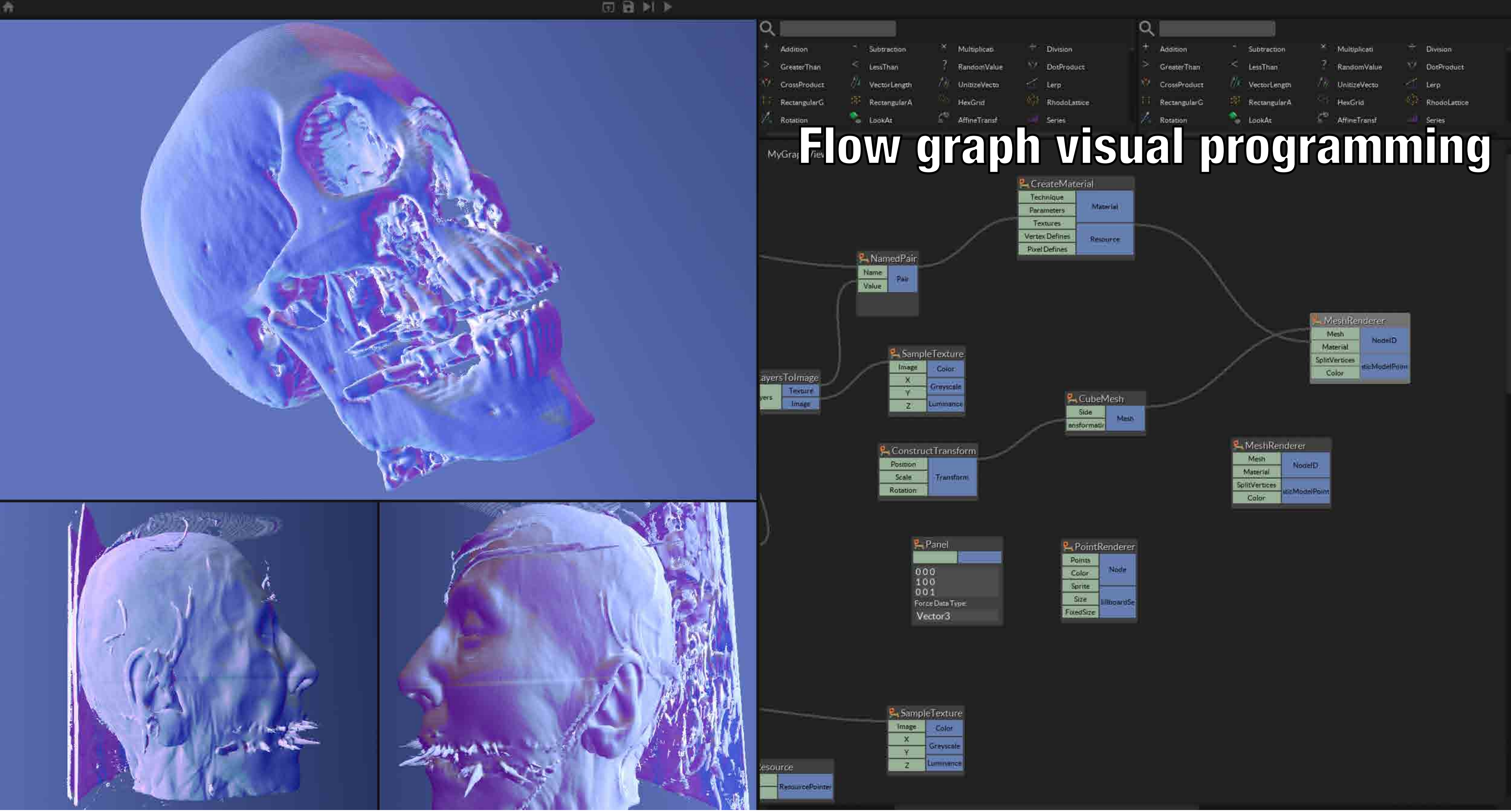





Traditionally we think of the geometry processing *pipeline*…

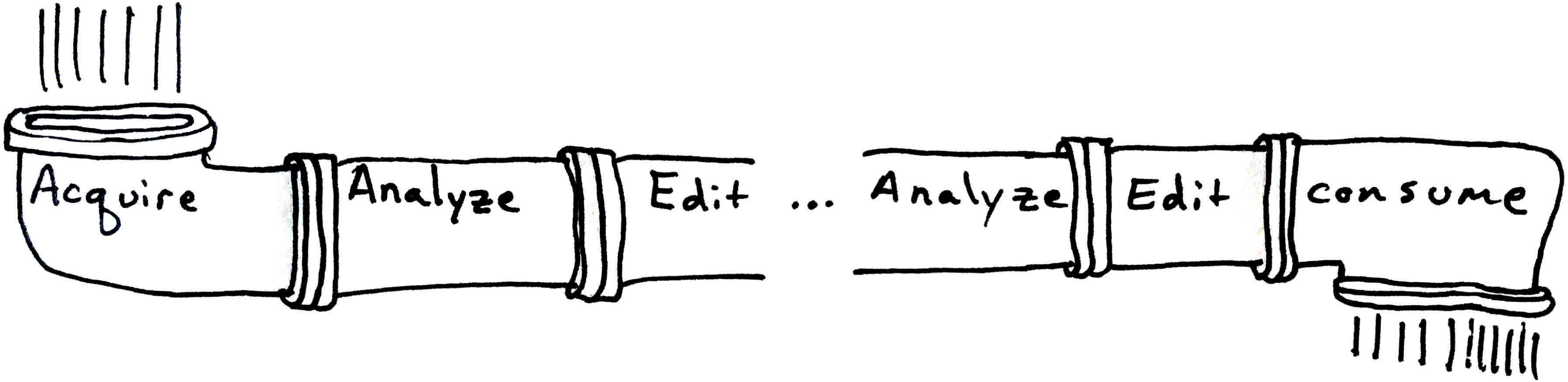


Flow graph visual programming

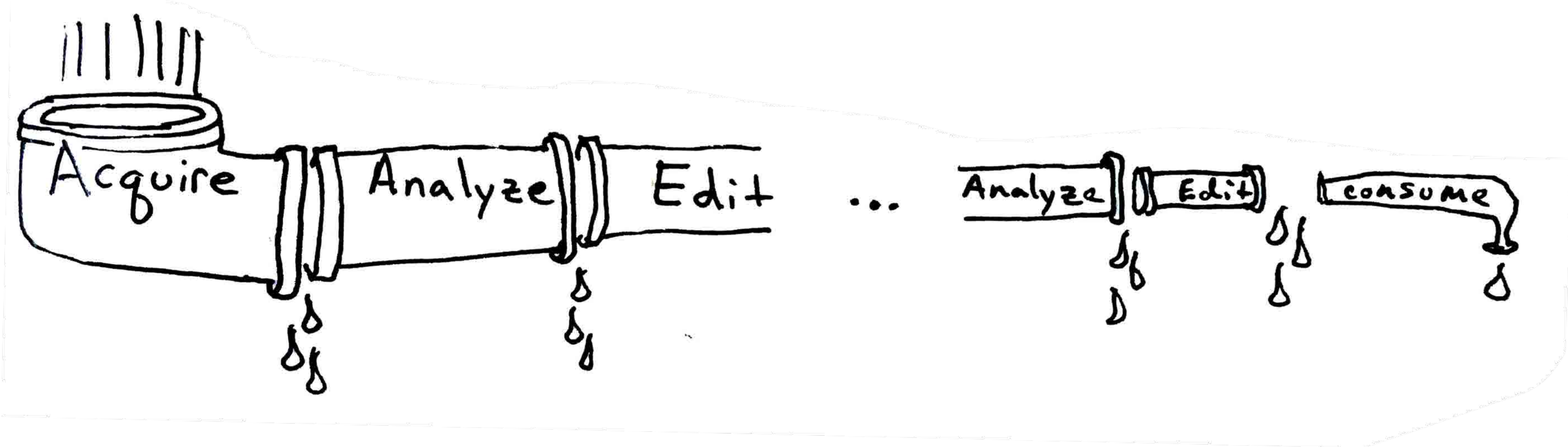


iogram by Mesh Inc. in Toronto

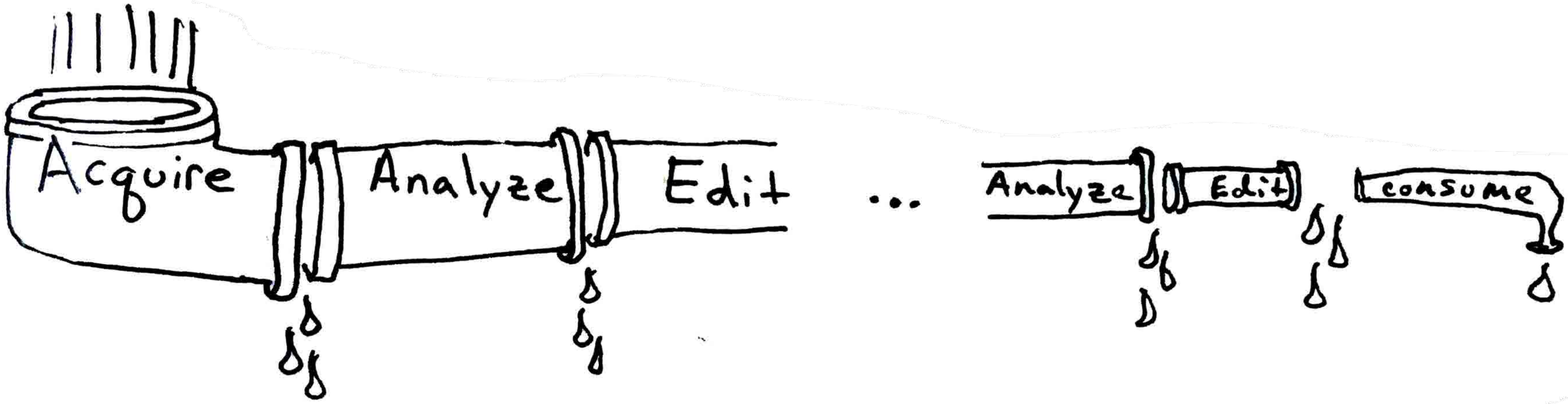
Traditionally we think of the geometry processing *pipeline*…



… in the *wild* this pipeline is leaky

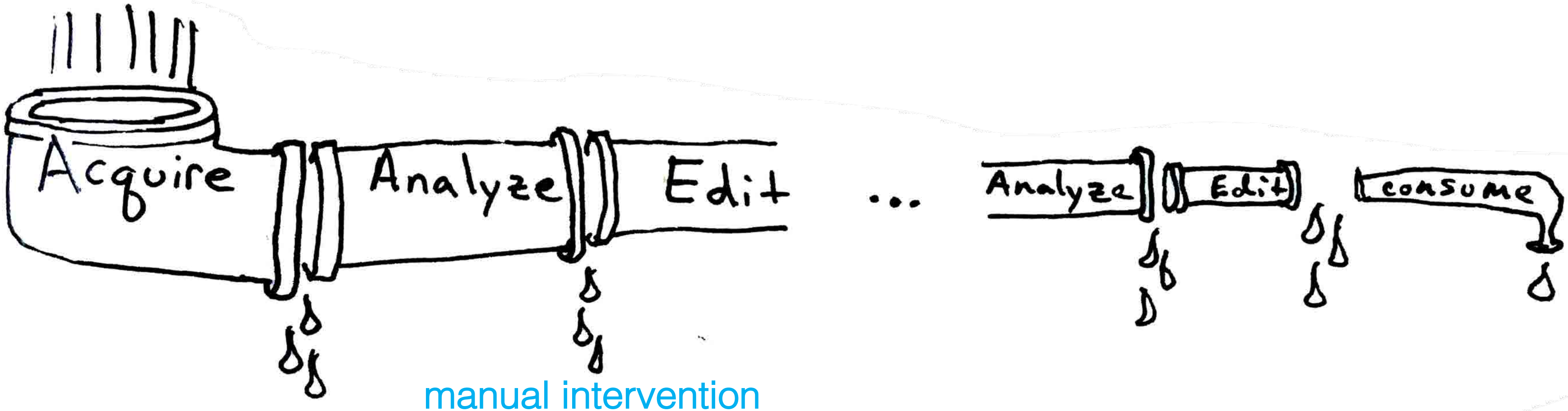


… in the *wild* this pipeline is leaky



lossy conversion

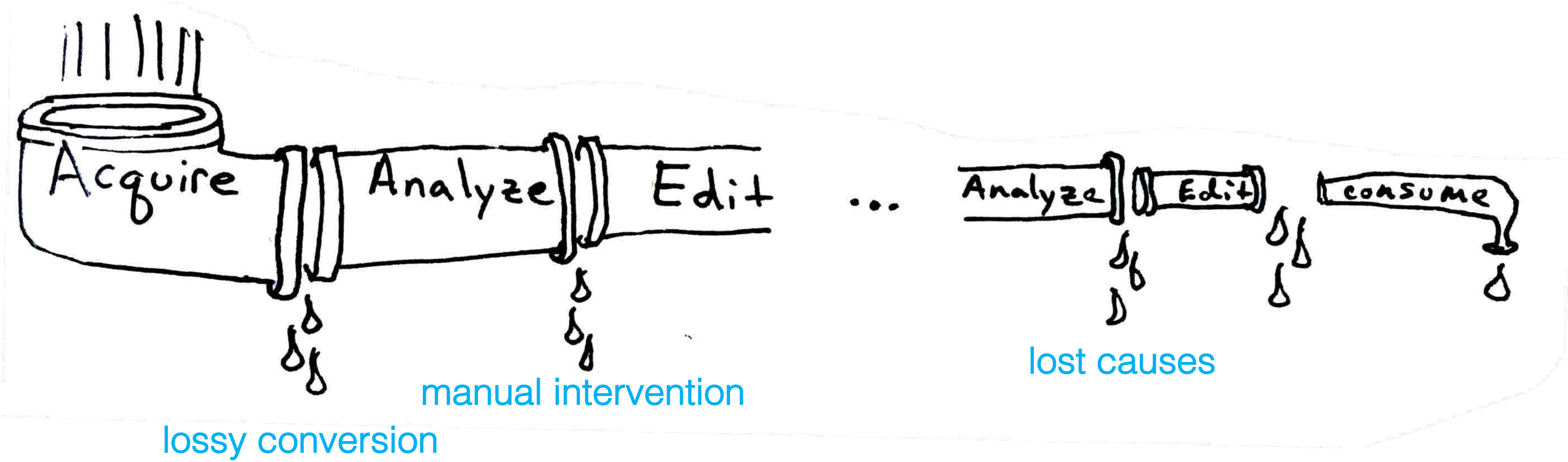
… in the *wild* this pipeline is leaky



manual intervention

lossy conversion

… in the *wild* this pipeline is leaky

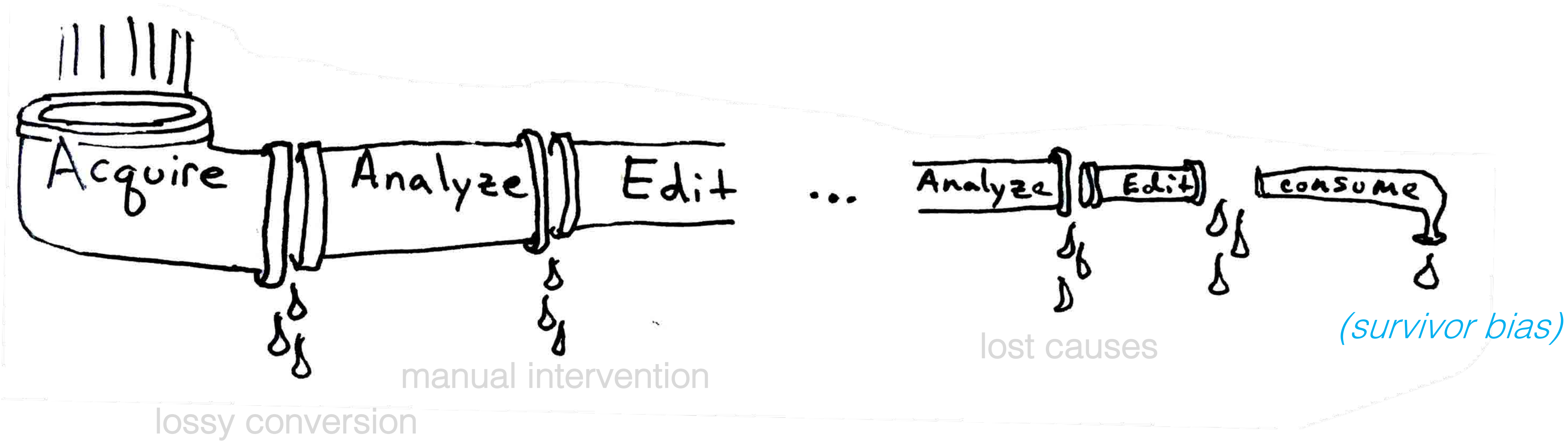


lost causes

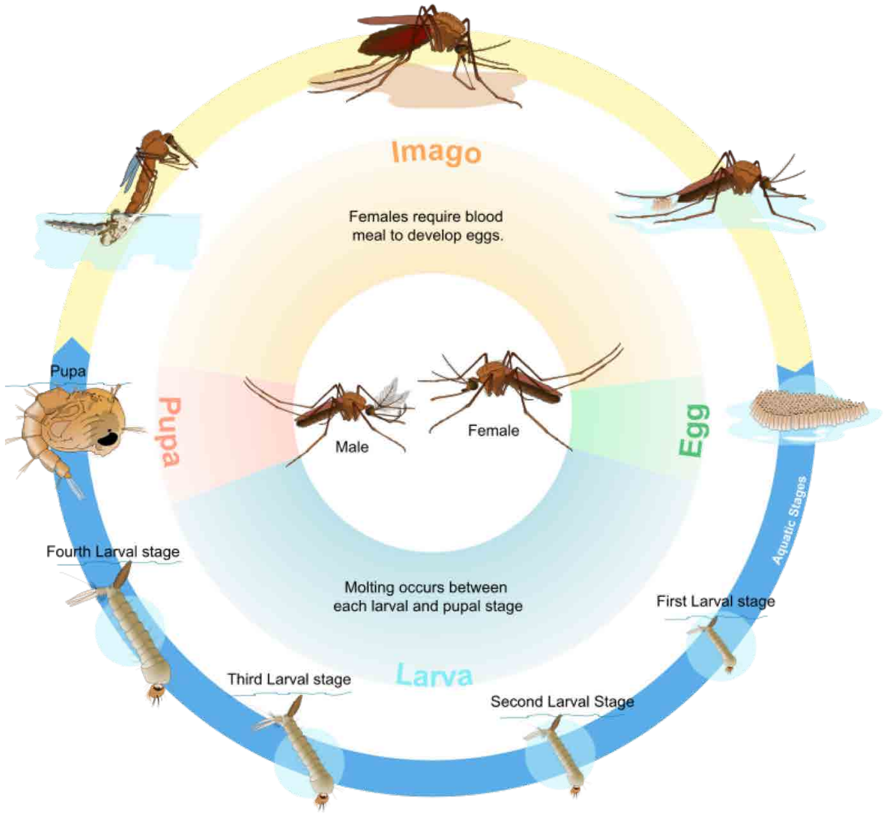
manual intervention

lossy conversion

… in the *wild* this pipeline is leaky

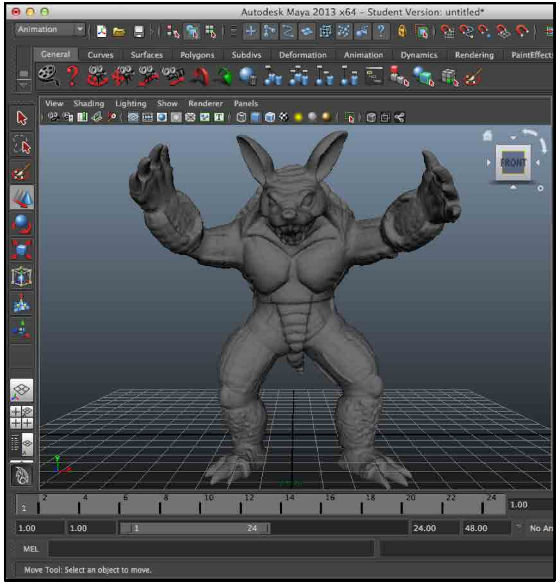


Geometry Processing is biology



Geometry processing studies the *life of a shape*

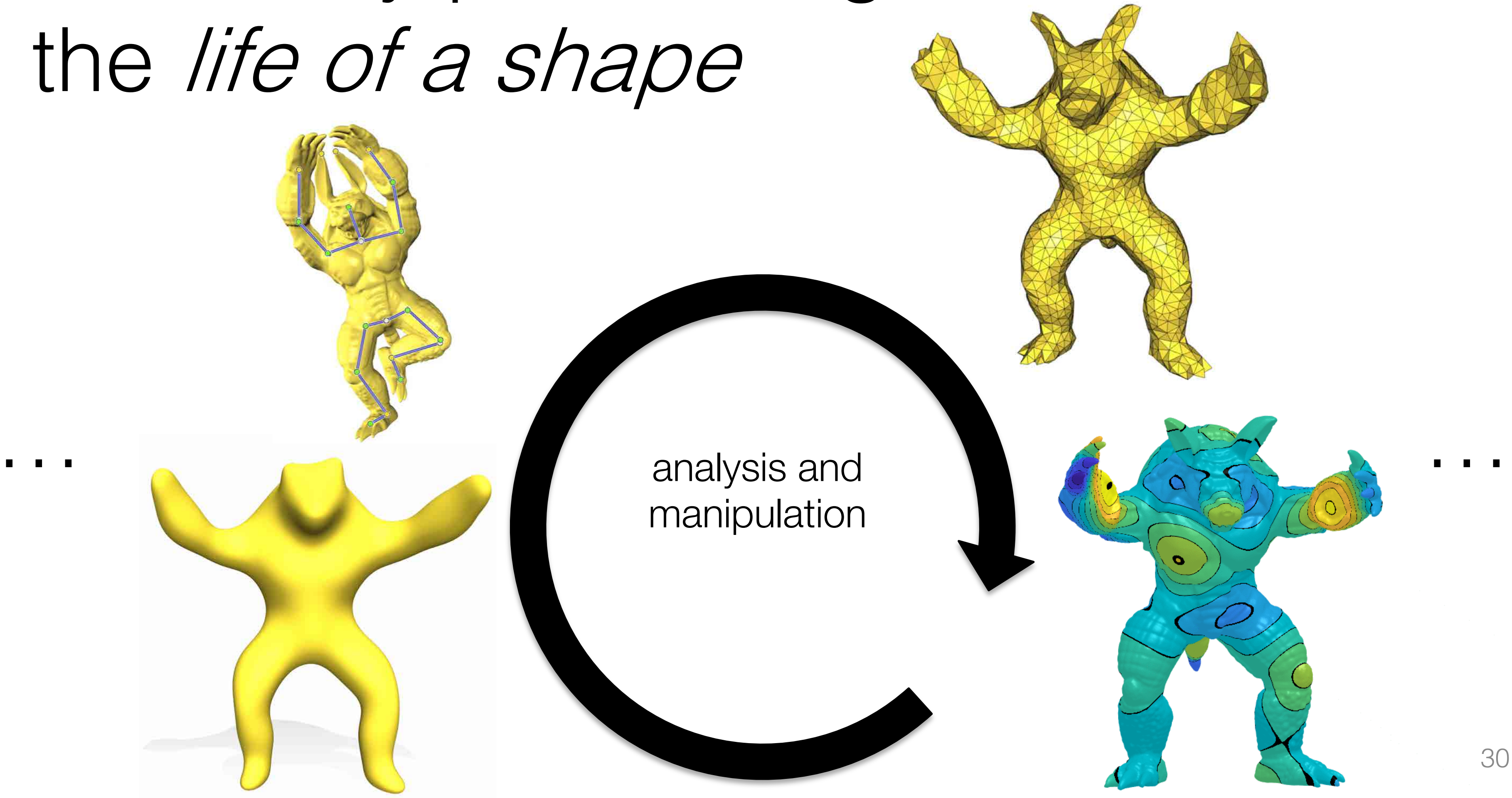
birth



. . .

e.g., scan of a physical object or modeling in Maya

Geometry processing studies the *life of a shape*



Geometry processing studies the *life of a shape*

consumption

. . .

animation

Geometry processing studies the *life of a shape*

consumption

…

3d printing

We should be suspicious of the conventional assumptions

“we can use any type or resolution of discretization any time we want”

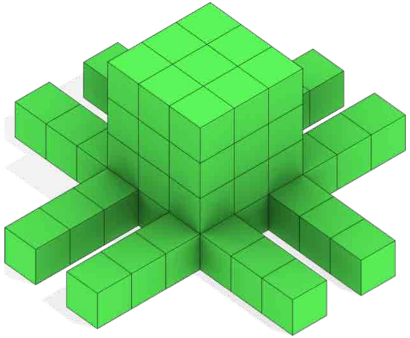
“geometric domain is given with certainty”

Conventional assumptions make problems well-posed and simpler

Conventional assumptions make problems well-posed and simpler

"Construct a coarse-to-fine
multigrid hierarchy for an octopus 🐙 shape"

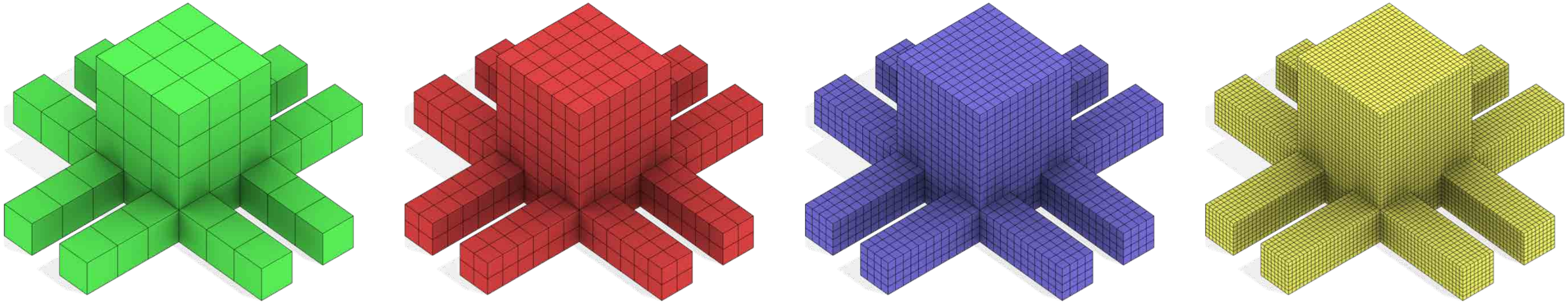
Conventional assumptions make problems well-posed and simpler



assume geometry is
representable on coarsest level

"Construct a coarse-to-fine
multigrid hierarchy for an octopus 🐙 shape"

Conventional assumptions make problems well-posed and simpler

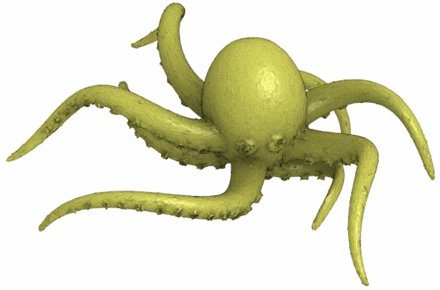


"Construct a coarse-to-fine
multigrid hierarchy for an octopus 🐙 shape"

In the *wild*, geometry is high-resolution and messy

“Construct a coarse-to-fine multigrid hierarchy for an octopus 🐙 shape”

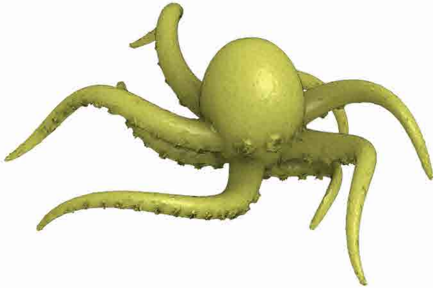
need to construct coarse domains *given* a fine domain



“Nested Cages” [Sacht, Vouga, & J. 2015]

In the *wild*, geometry is high-resolution and messy

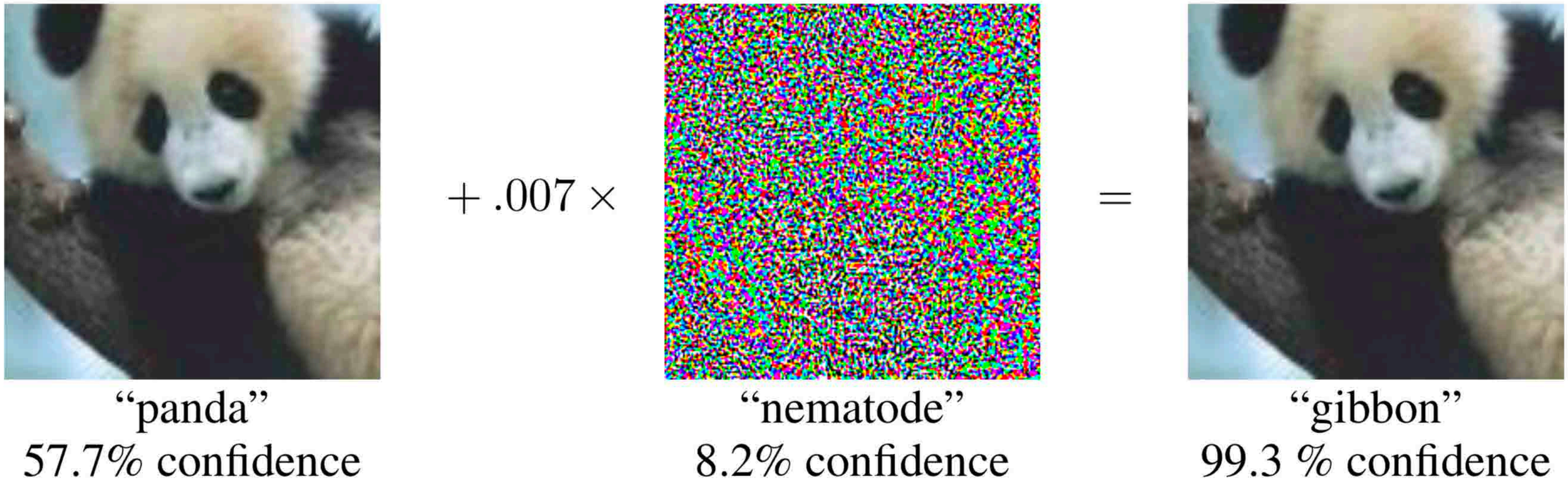
“Construct a coarse-to-fine
multigrid hierarchy for an octopus 🐙 shape”



“Nested Cages” [Sacht, Vouga, & J. 2015]

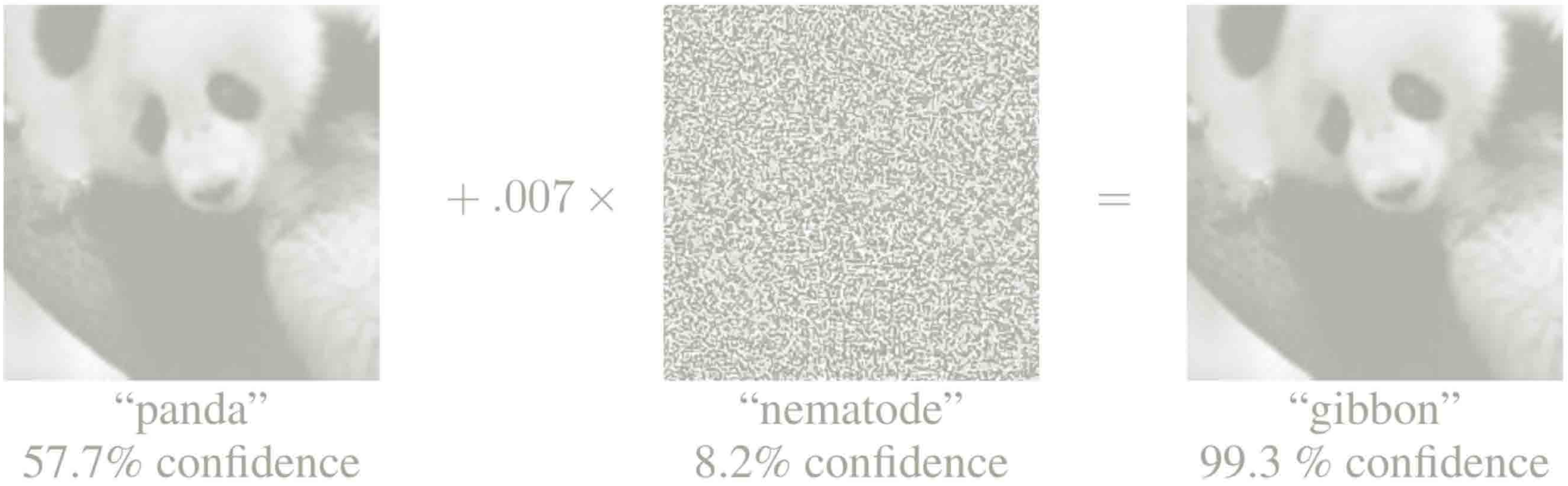
Images are often easier to work with than geometry…

Images are often easier to work with than geometry…



Source: [Goodfellow et al. 2015]

Images are often easier to work with than geometry…



“Most adversarial example research today is based on a specific *toy game* in the context of visual object recognition.”
-- Goodfellow, *three years later*

Geometry and light are the primary degrees of freedom behind images

Adversarial Geometry

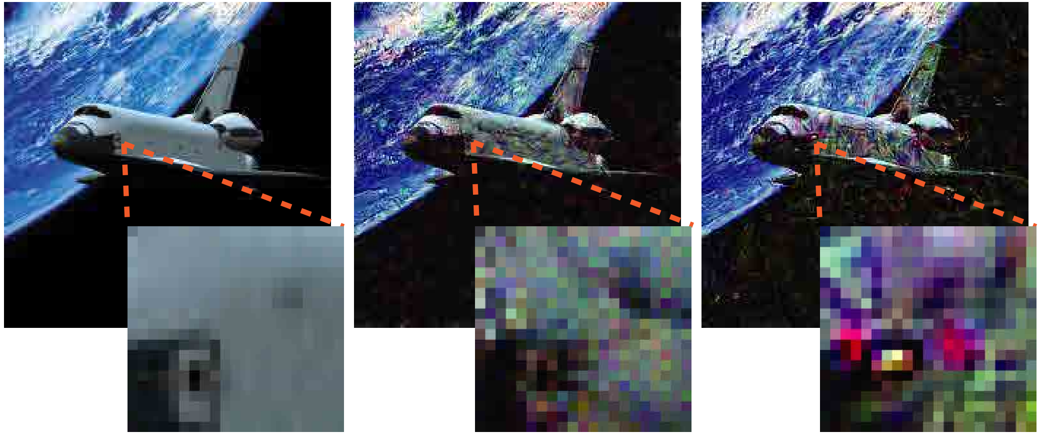
street sign 98%

mailbox 83%

"Beyond Pixel Norm-Balls: Parametric Adversaries using an Analytically Differentiable Renderer"
[Liu, Tao, Chun-Liang Li, Nowrouzezahrai, & J. 2019]

Geometry and light are the primary degrees of freedom behind images

original image | one-step pixel [Goodfellow 14] | multi-step pixel [Moosavi Dezfooli 16]



"Beyond Pixel Norm-Balls: Parametric Adversaries using an Analytically Differentiable Renderer"
[Liu, Tao, Chun-Liang Li, Nowrouzezahrai, & J. 2019]

Geometry and light are the primary degrees of freedom behind images

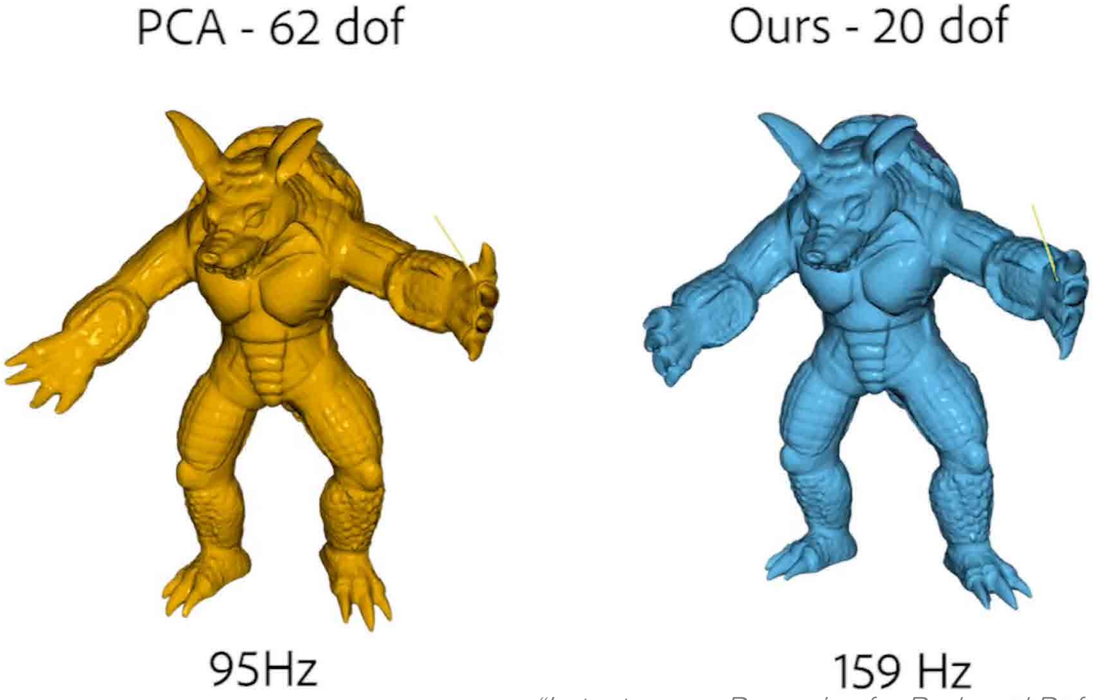
original image	one-step pixel [Goodfellow 14]	multi-step pixel [Moosavi Dezfooli 16]	**parametric (lighting)**	**parametric (geometry)**

"Beyond Pixel Norm-Balls: Parametric Adversaries using an Analytically Differentiable Renderer"
[Liu, Tao, Chun-Liang Li, Nowrouzezahrai, & J. 2019]

Linear models are straightforward and simple to work with…

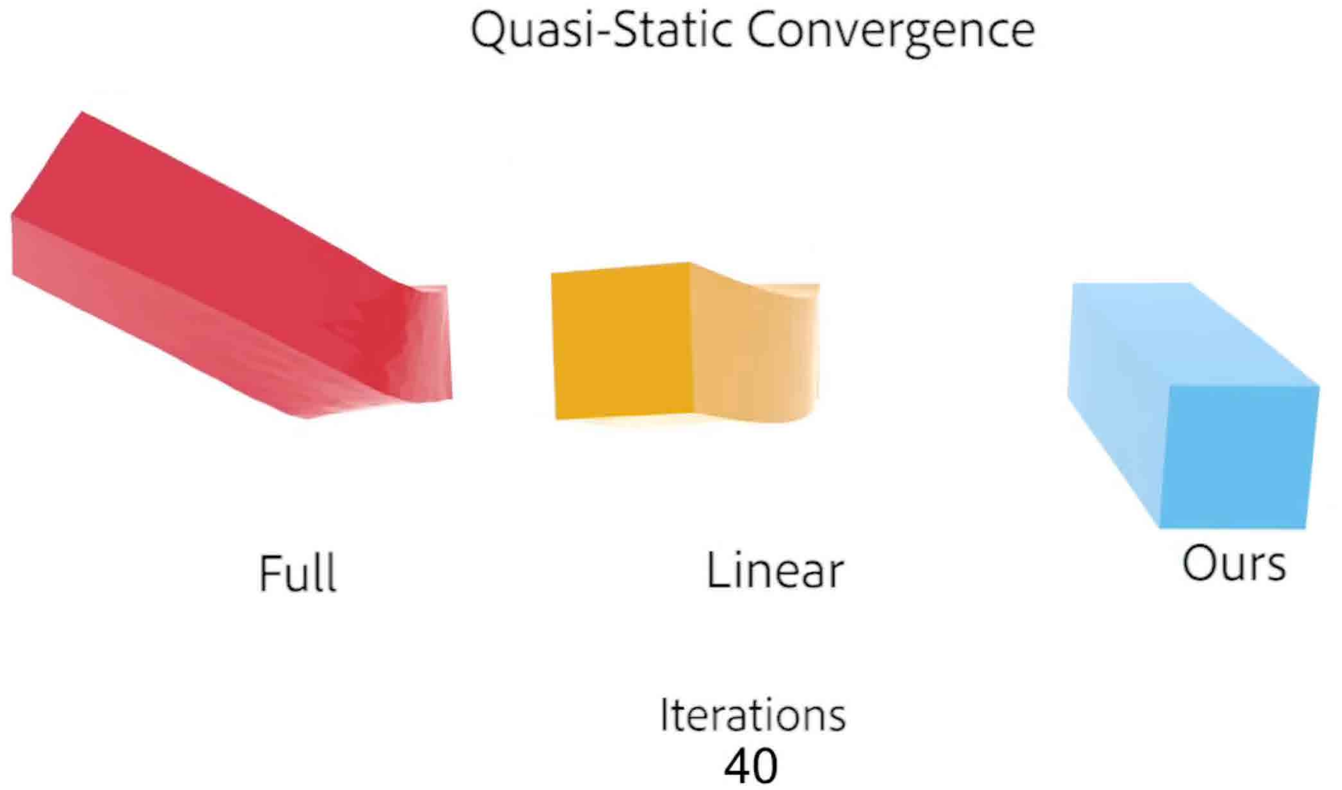
"Latent-space Dynamics for Reduced Deformable Simulation"
[Fulton, Modi, Duvenaud, Levin, J. 2019]

… but ultimately an inefficient use of data



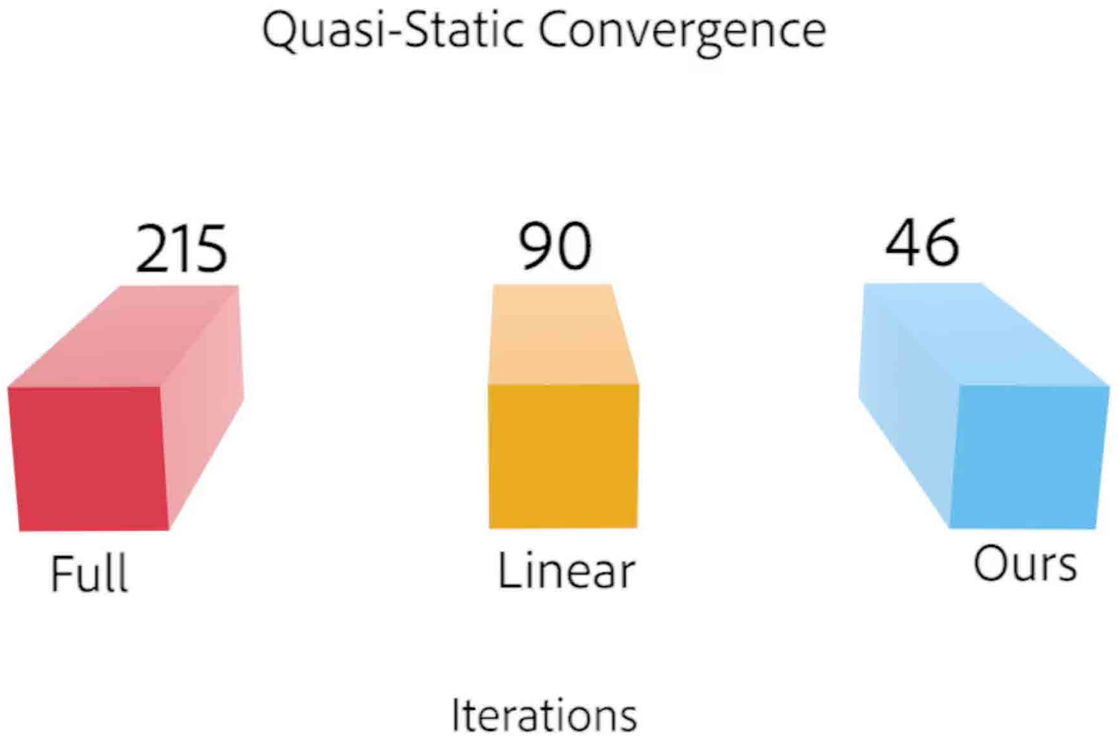
"Latent-space Dynamics for Reduced Deformable Simulation"
[Fulton, Modi, Duvenaud, Levin, J. 2019]

… but ultimately an inefficient use of data



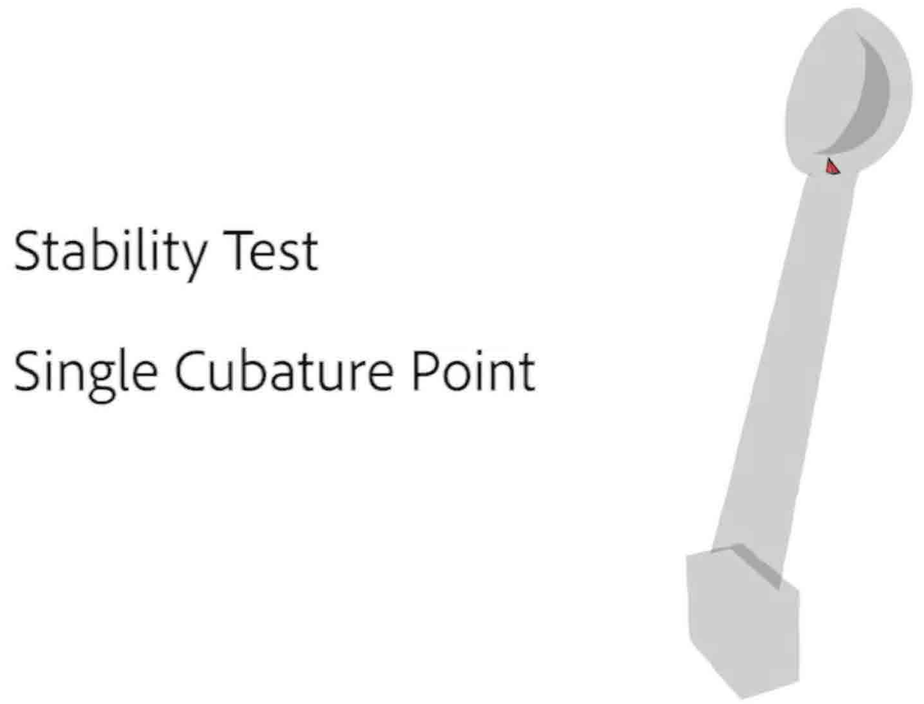
"Latent-space Dynamics for Reduced Deformable Simulation"
[Fulton, Modi, Duvenaud, Levin, J. 2019]

… but ultimately an inefficient use of data



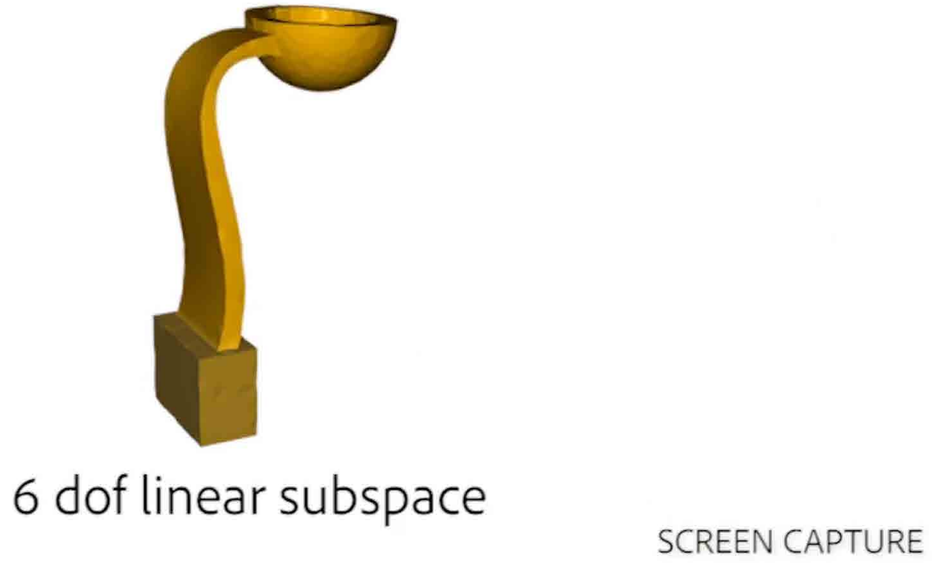
"Latent-space Dynamics for Reduced Deformable Simulation"
[Fulton, Modi, Duvenaud, Levin, J. 2019]

… but ultimately an inefficient use of data



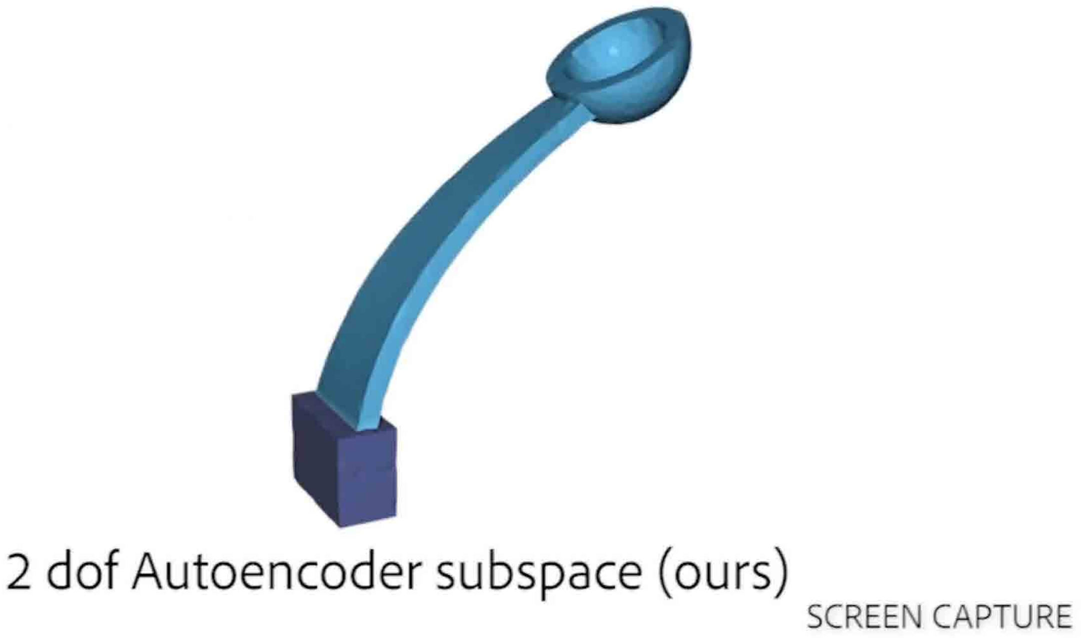
"Latent-space Dynamics for Reduced Deformable Simulation"
[Fulton, Modi, Duvenaud, Levin, J. 2019]

… but ultimately an inefficient use of data



"Latent-space Dynamics for Reduced Deformable Simulation"
[Fulton, Modi, Duvenaud, Levin, J. 2019]

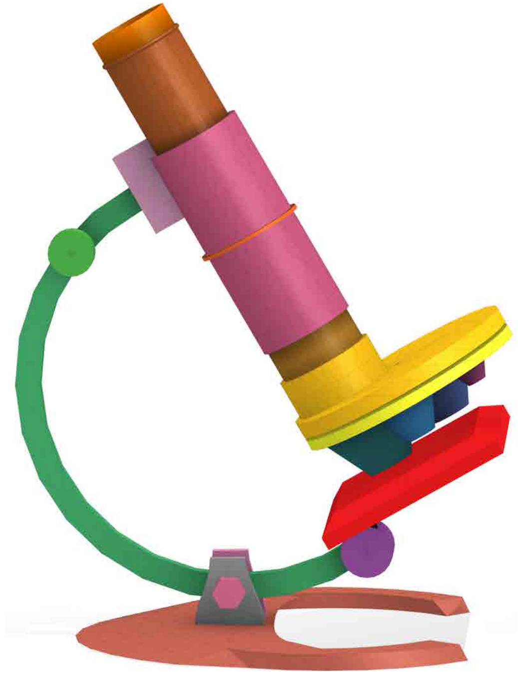
… but ultimately an inefficient use of data



"Latent-space Dynamics for Reduced Deformable Simulation"
[Fulton, Modi, Duvenaud, Levin, J. 2019]

Isolating subproblems from full pipeline can make robustness harder

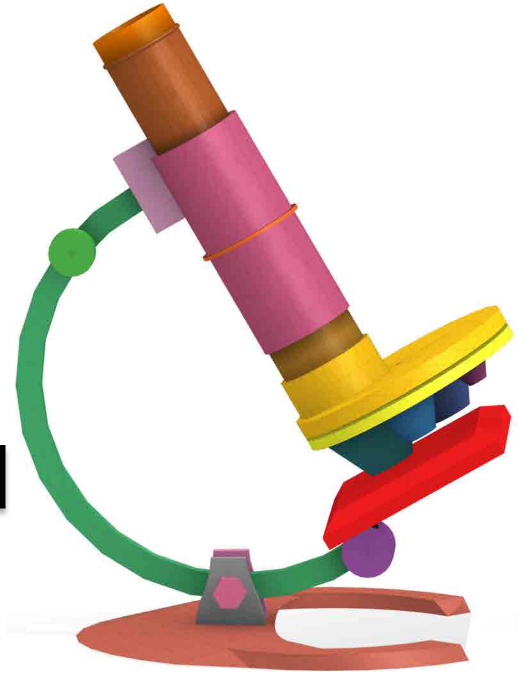
Isolating subproblems from full pipeline can make robustness harder



Solve heat equation on this microscope modeled as union of 24 overlapping parts

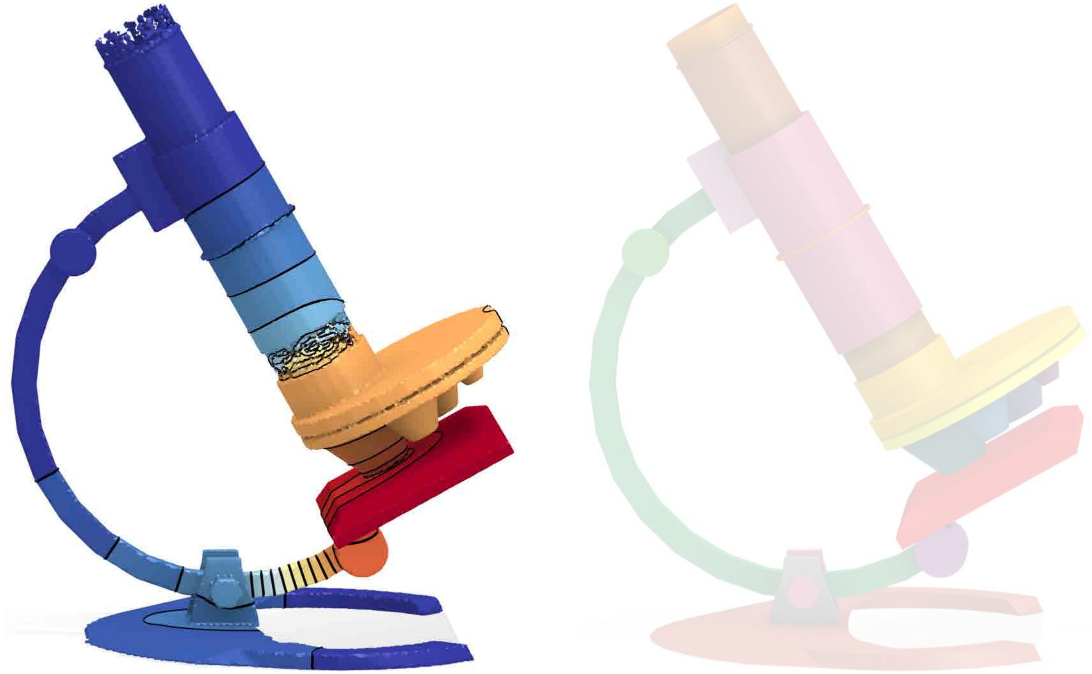
Isolating subproblems from full pipeline can make robustness harder

Discretize volume and
use finite element method



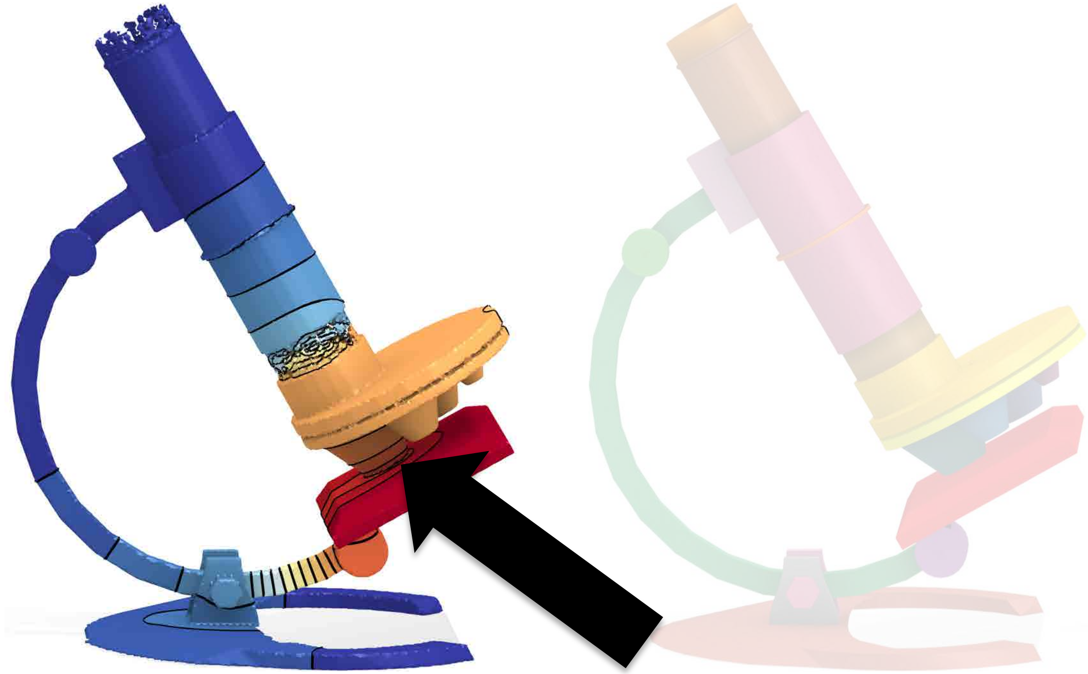
Solve heat equation on
this microscope
modeled as union of 24
overlapping parts

Isolating subproblems from full pipeline can make robustness harder



Solve heat equation on this microscope modeled as union of 24 overlapping parts

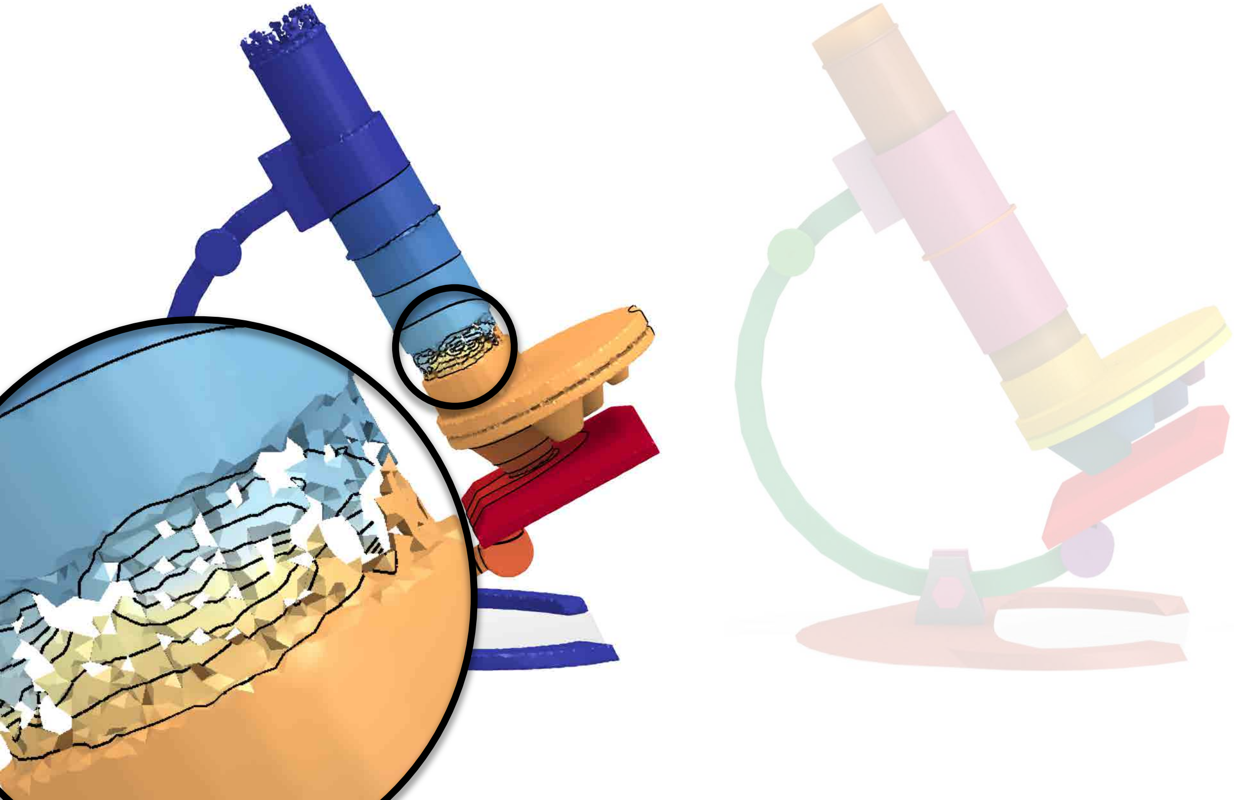
Isolating subproblems from full pipeline can make robustness harder



Solve heat equation on this microscope modeled as union of 24 overlapping parts

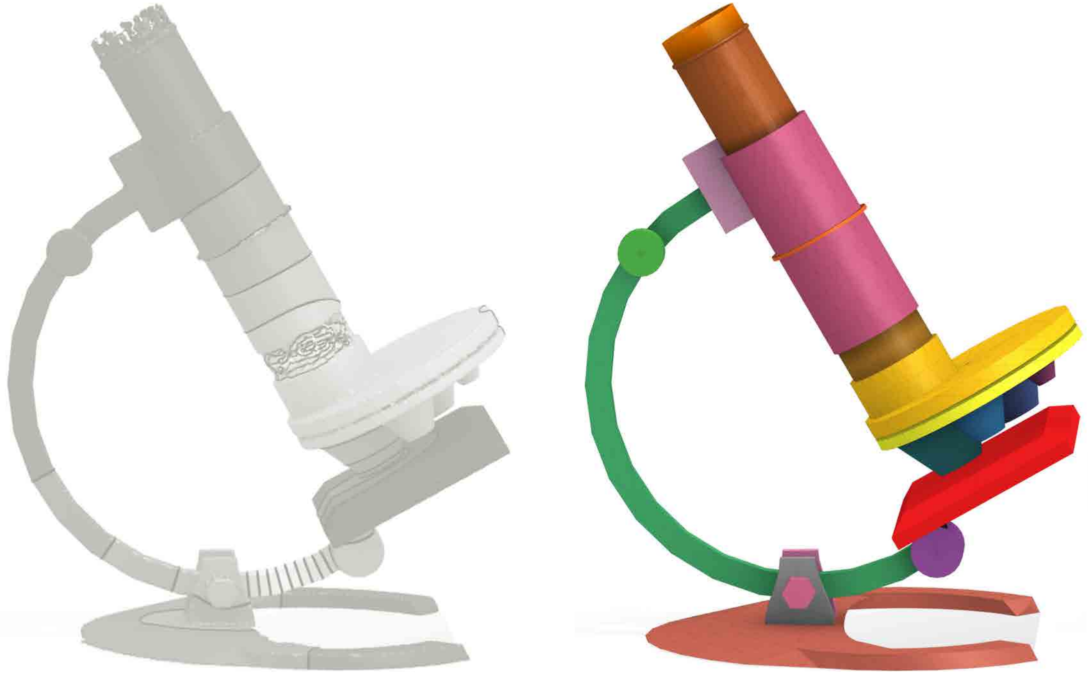
close-but-separate parts fused together

Isolating subproblems from full pipeline can make robustness harder



Solve heat equation on this microscope modeled as union of 24 overlapping parts

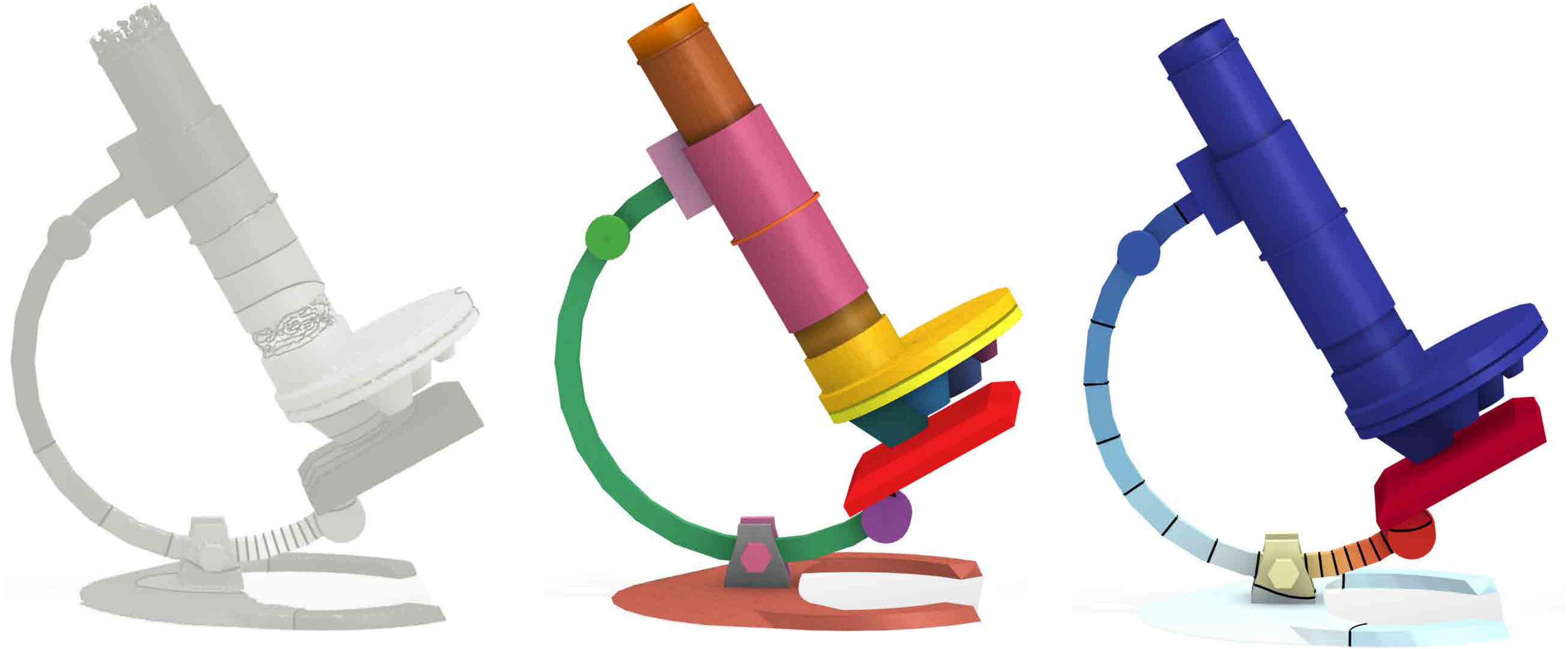
Isolating subproblems from full pipeline can make robustness harder



Discretize *each* volume
then solve coupled PDE

"Solid Geometry Processing on Deconstructed Domains"
[Sellán, Cheng, Ma, Dembowski, & J. 2019]

Isolating subproblems from full pipeline can make robustness harder



"Solid Geometry Processing on Deconstructed Domains"
[Sellán, Cheng, Ma, Dembowski, & J. 2019]

Return to fundamental questions about shapes and consider generalization to *messy* shapes

Return to fundamental questions about shapes and consider generalization to *messy* shapes

Am I inside or outside of a shape?

Return to fundamental questions about shapes and consider generalization to *messy* shapes

Am I inside or outside of a shape?

What is the combination of two shapes?

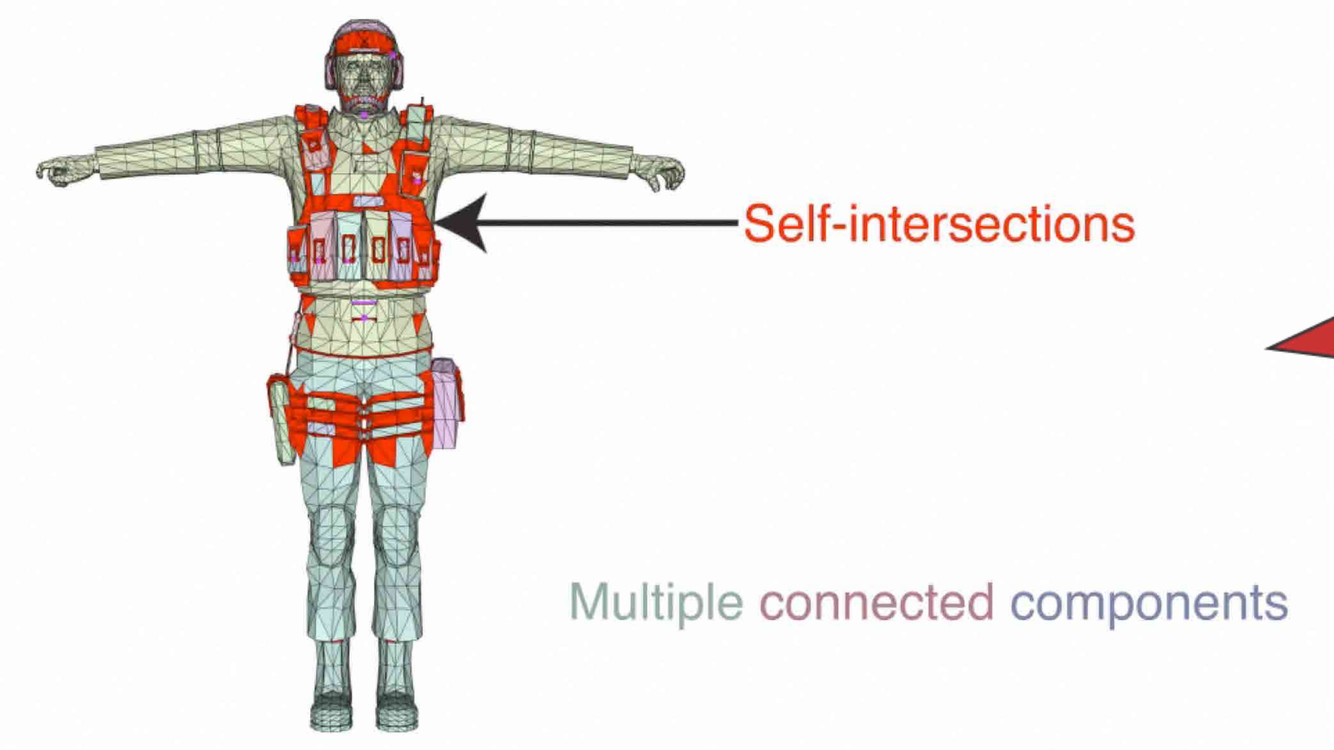
Return to fundamental questions about shapes and consider generalization to *messy* shapes

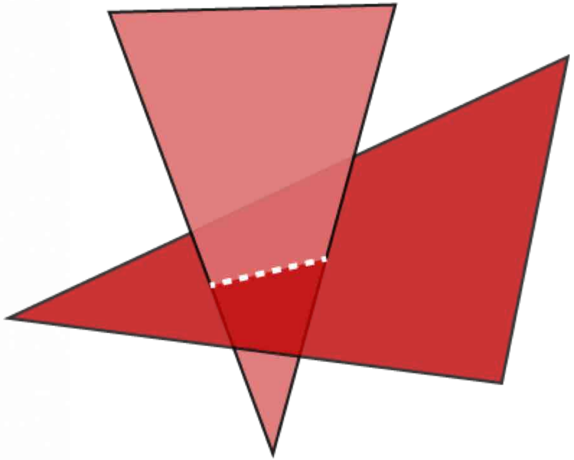
Am I inside or outside of a shape?

What is the combination of two shapes?

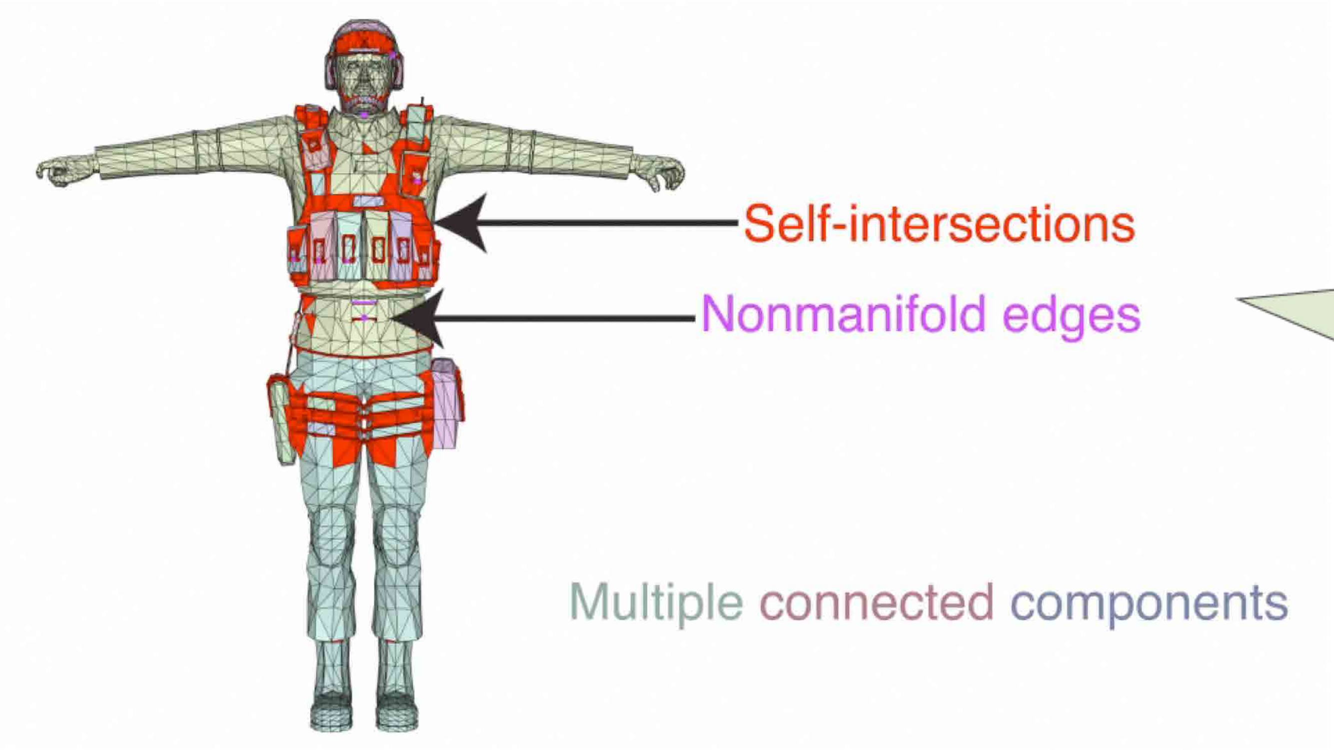
Given a solid object's surface,
can I represent the volume?

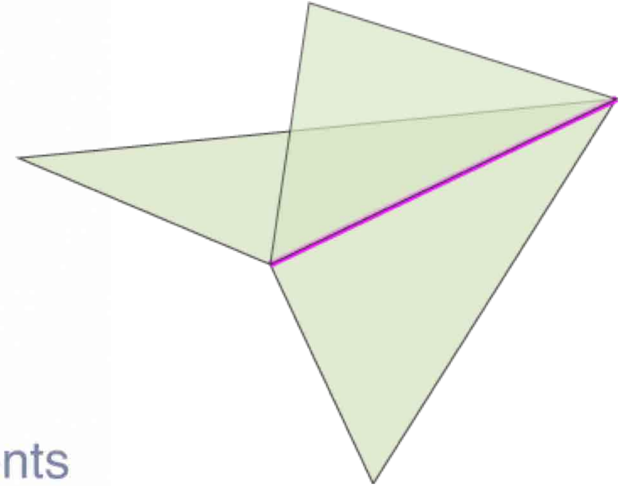
Good enough for visualization does not imply good enough for admit geometric computation



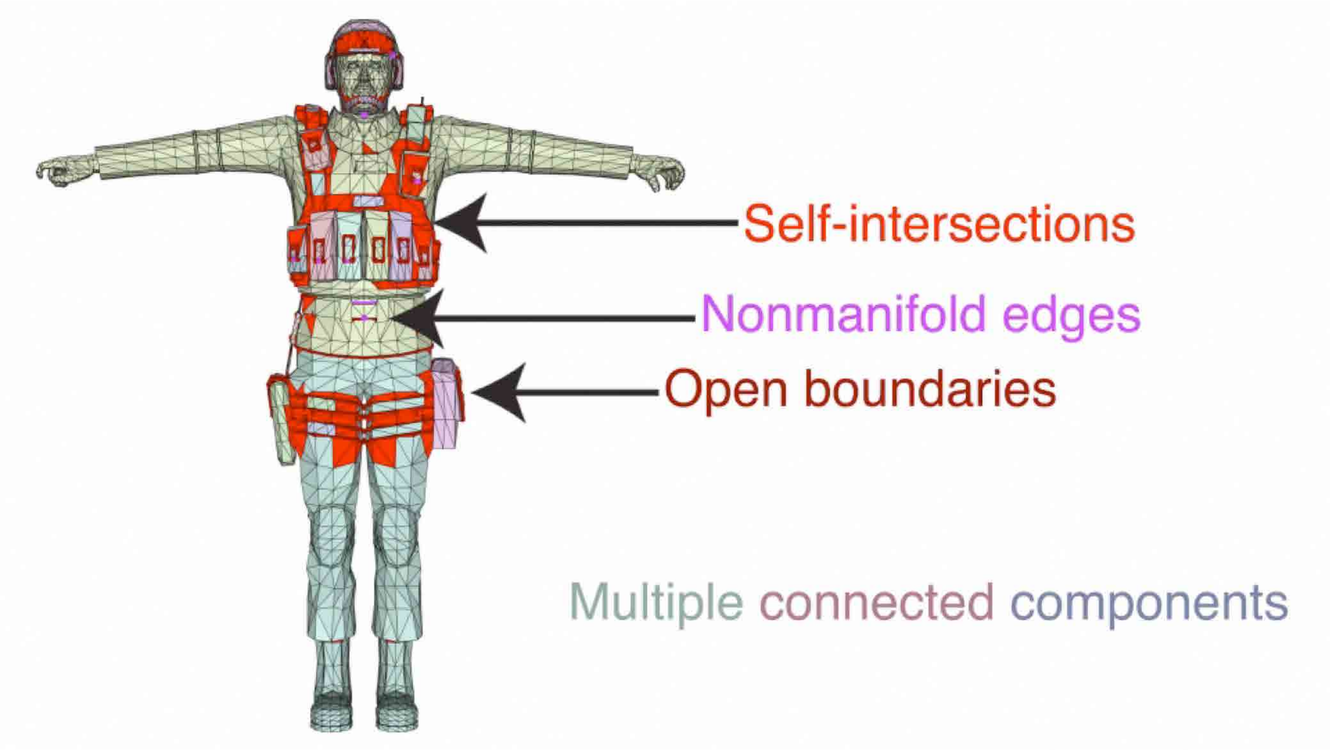


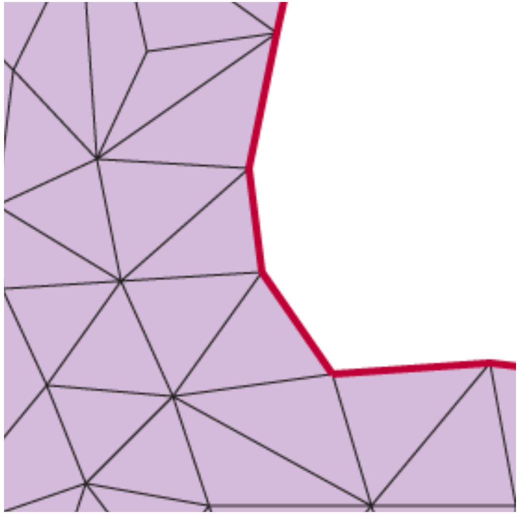
Good enough for visualization does not imply good enough for admit geometric computation





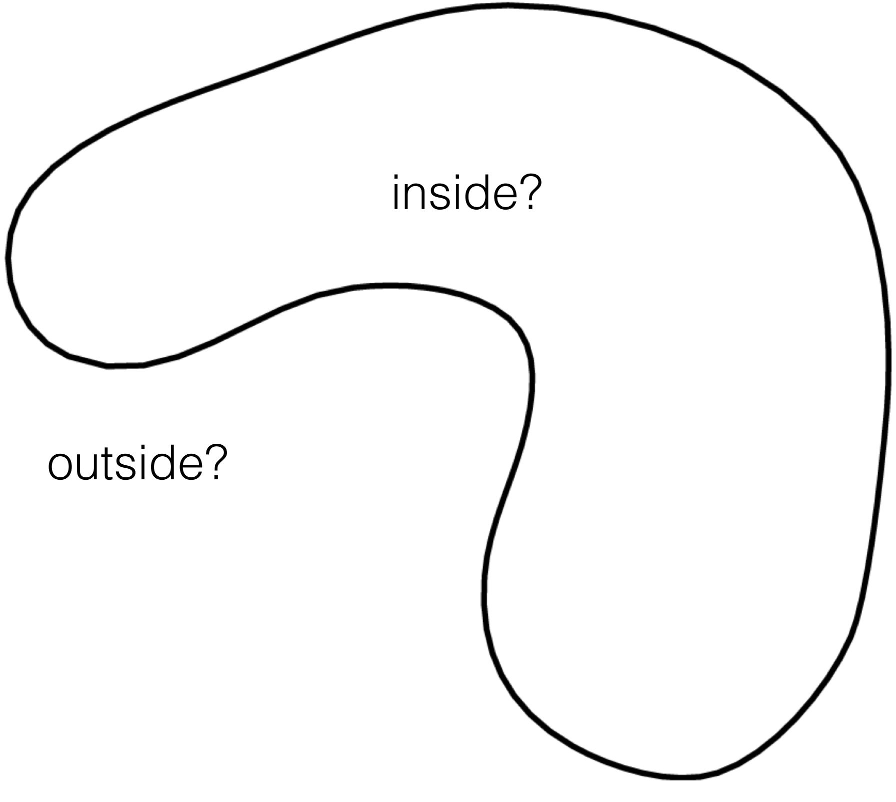
Good enough for visualization does not imply good enough for admit geometric computation



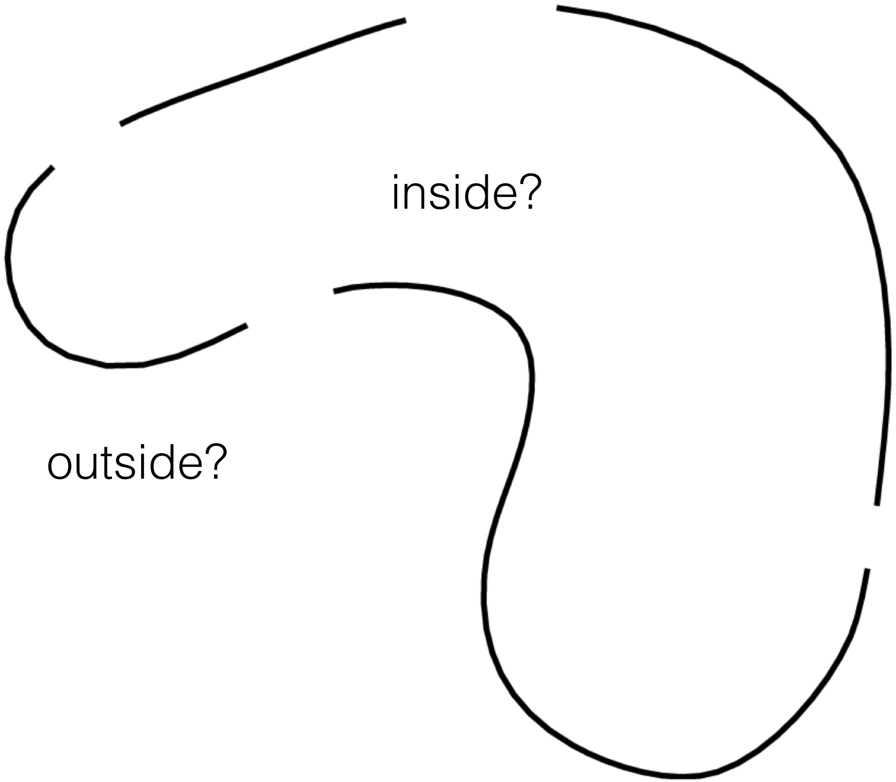


Fatal error
command exited with status 1

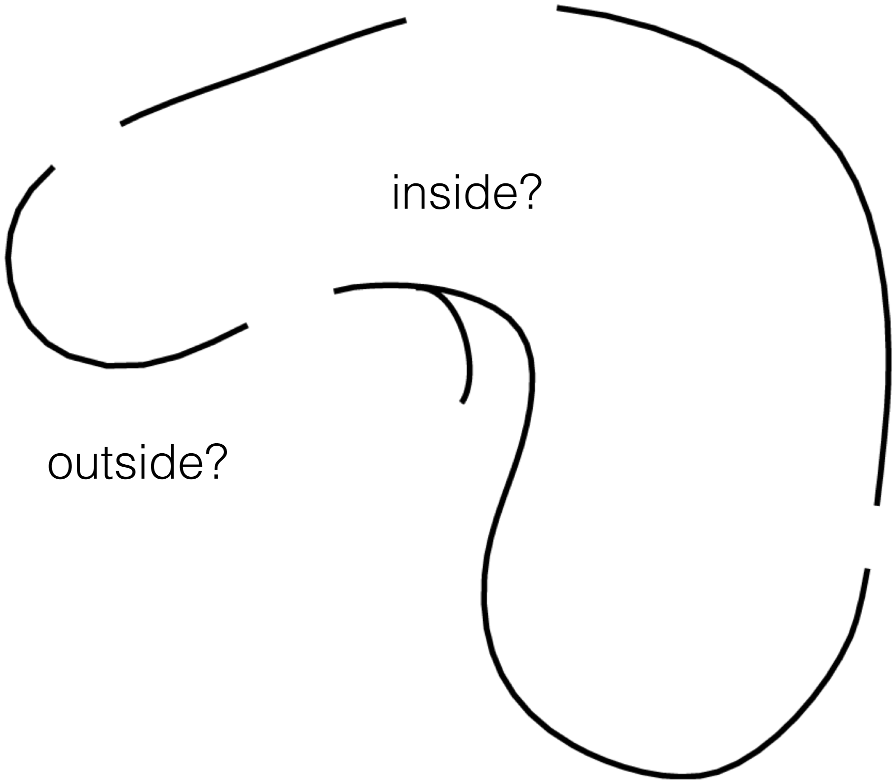
Adapt traditional algorithms and theory to work even in the presence of messy data



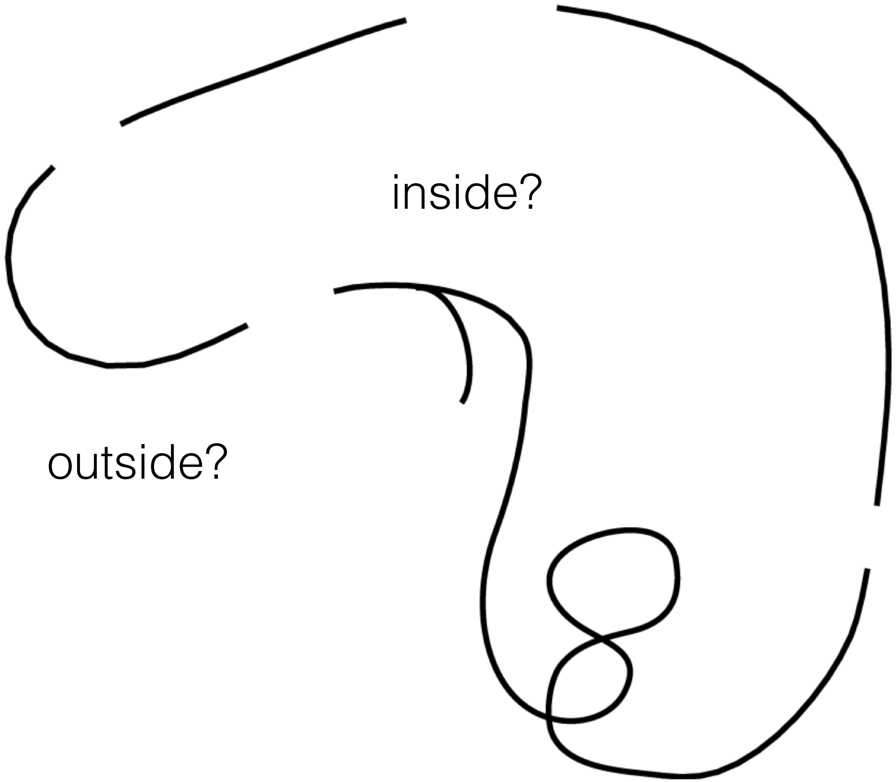
Adapt traditional algorithms and theory to work even in the presence of messy data



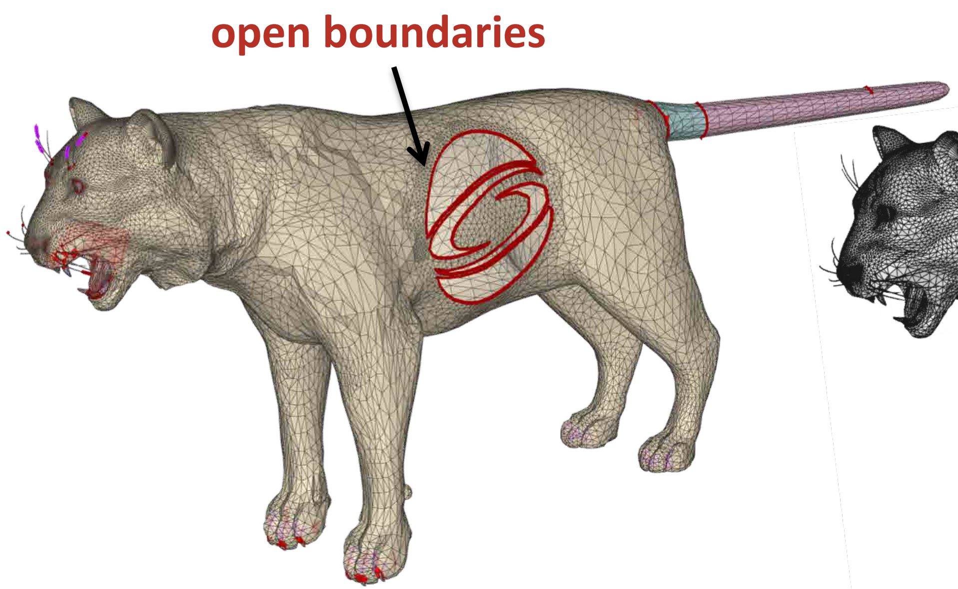
Adapt traditional algorithms and theory to work even in the presence of messy data



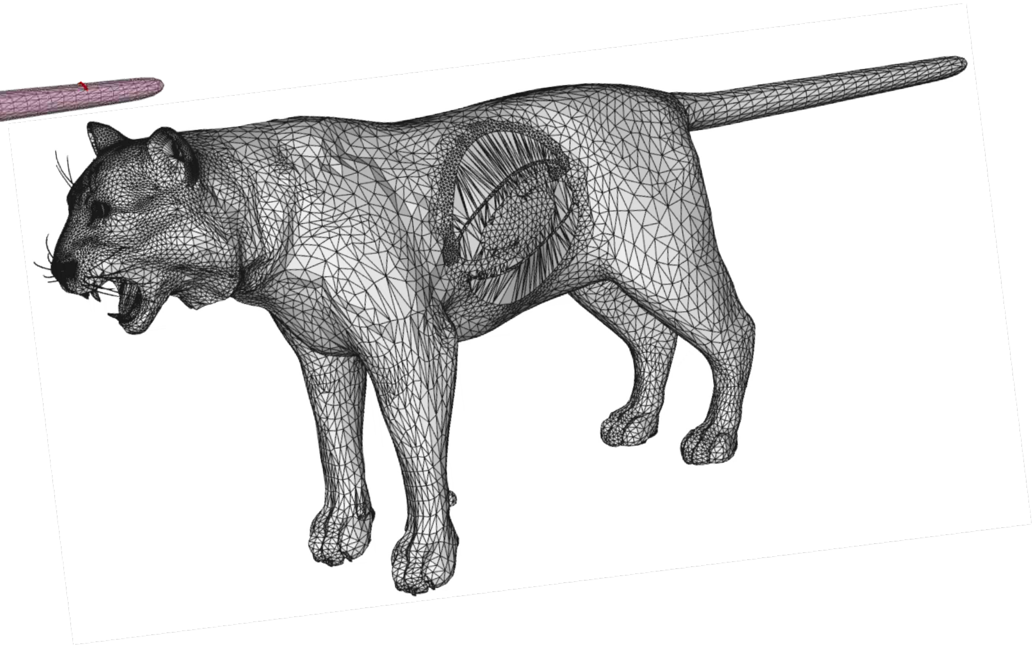
Adapt traditional algorithms and theory to work even in the presence of messy data



Determining insideness is *fundamental*

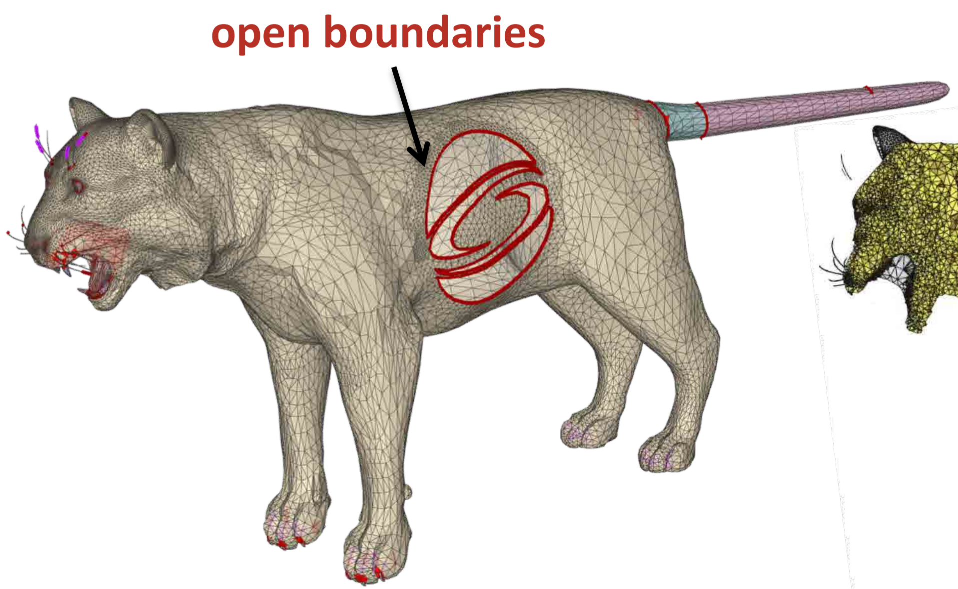


“messy” input mesh

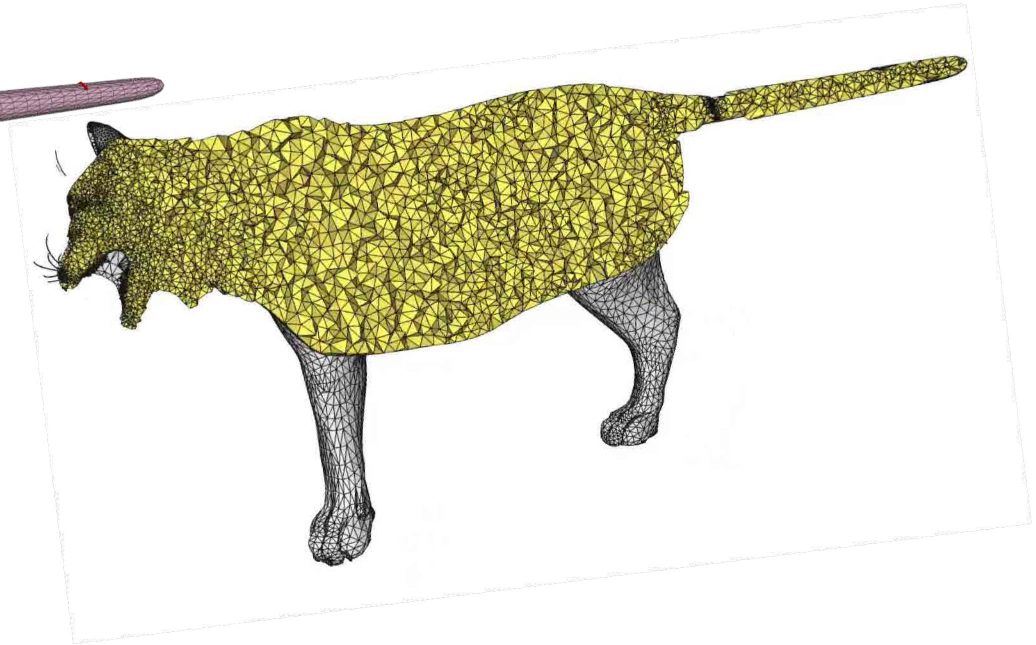


volume discretization

Determining insideness is *fundamental*

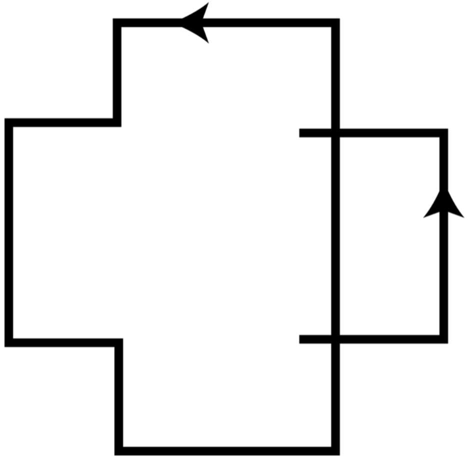


“messy” input mesh

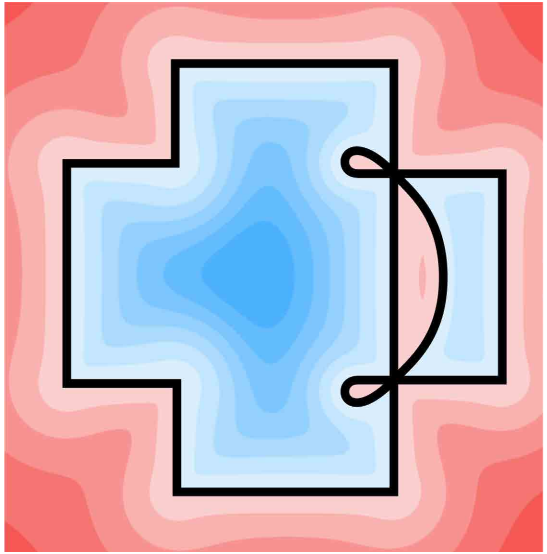


volume discretization

Previous solutions are unsatisfactory

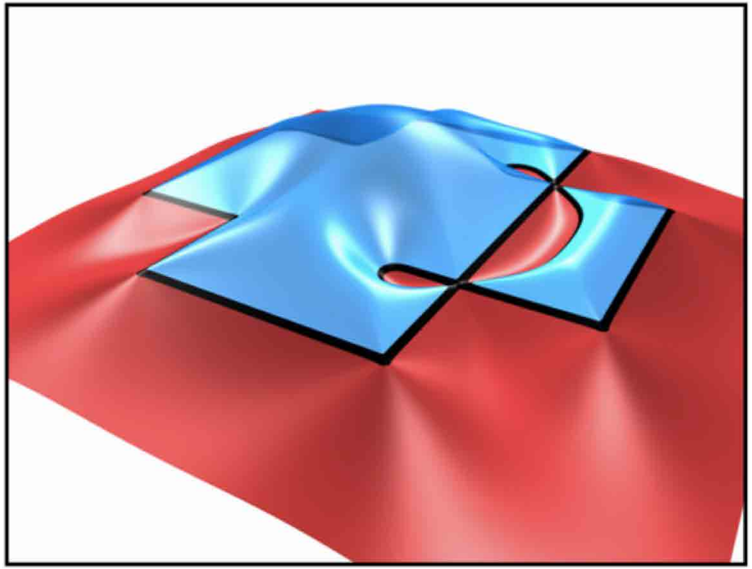


Input



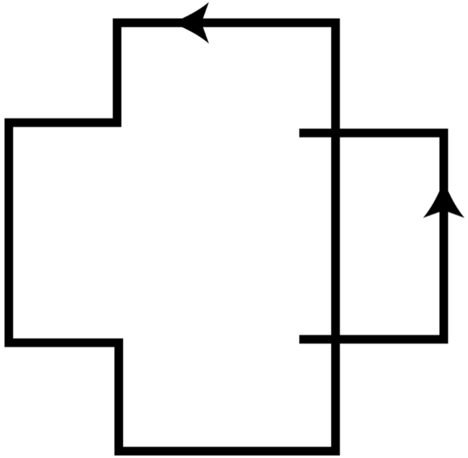
Interpolating function

[Shen et al. 2004]

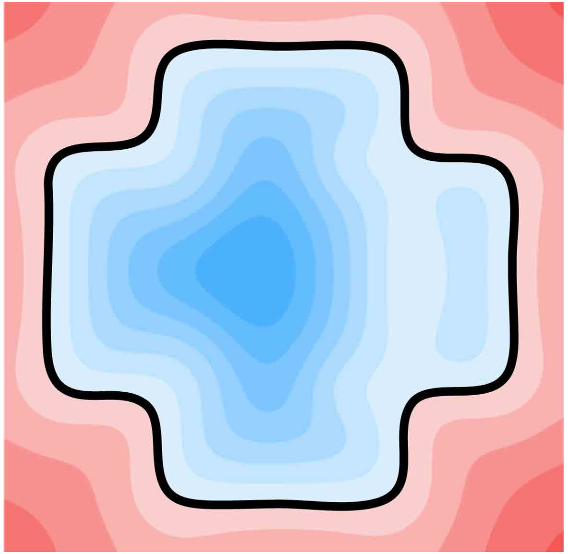


Height field visualization

Previous solutions are unsatisfactory

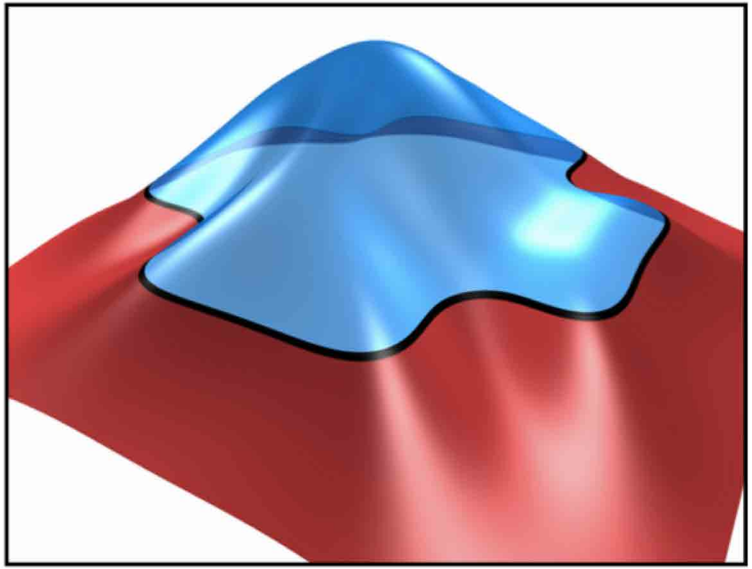


Input



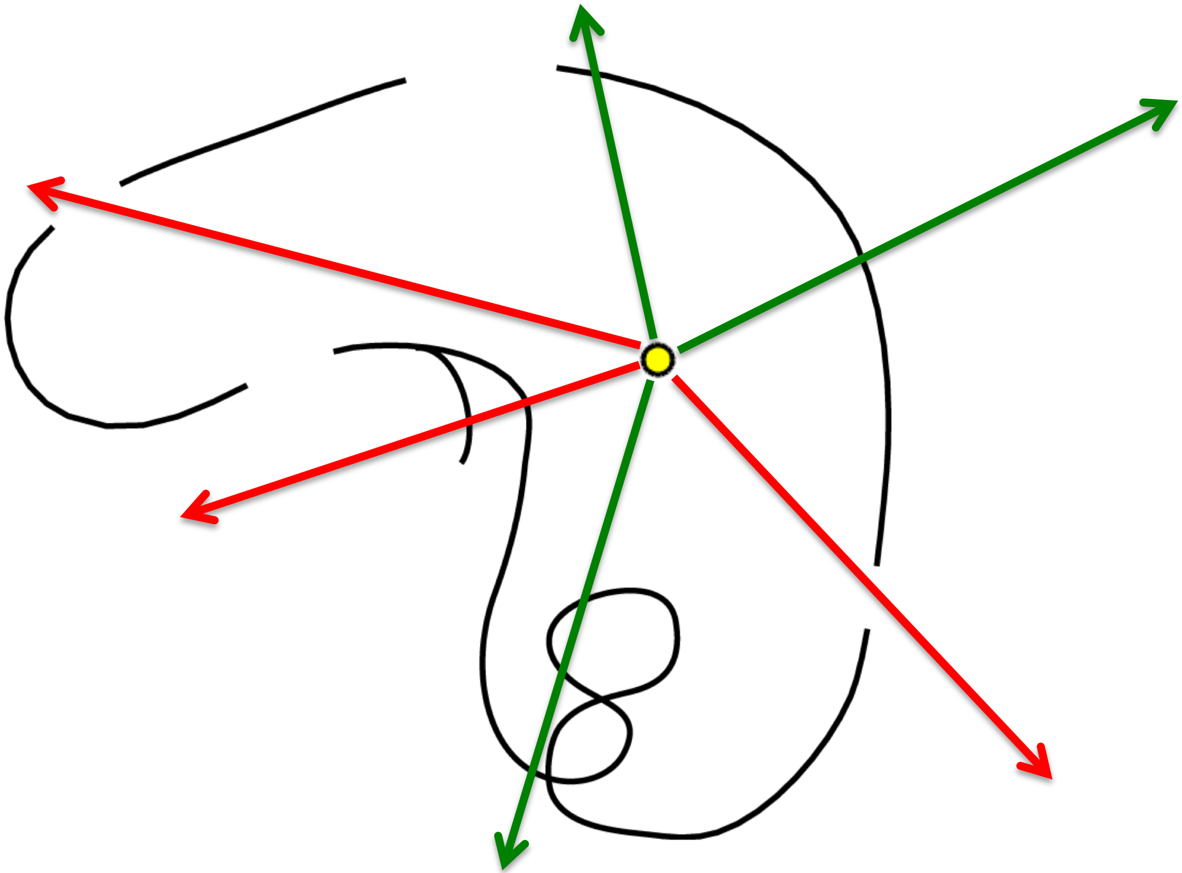
Approximating function

[Shen et al. 2004]



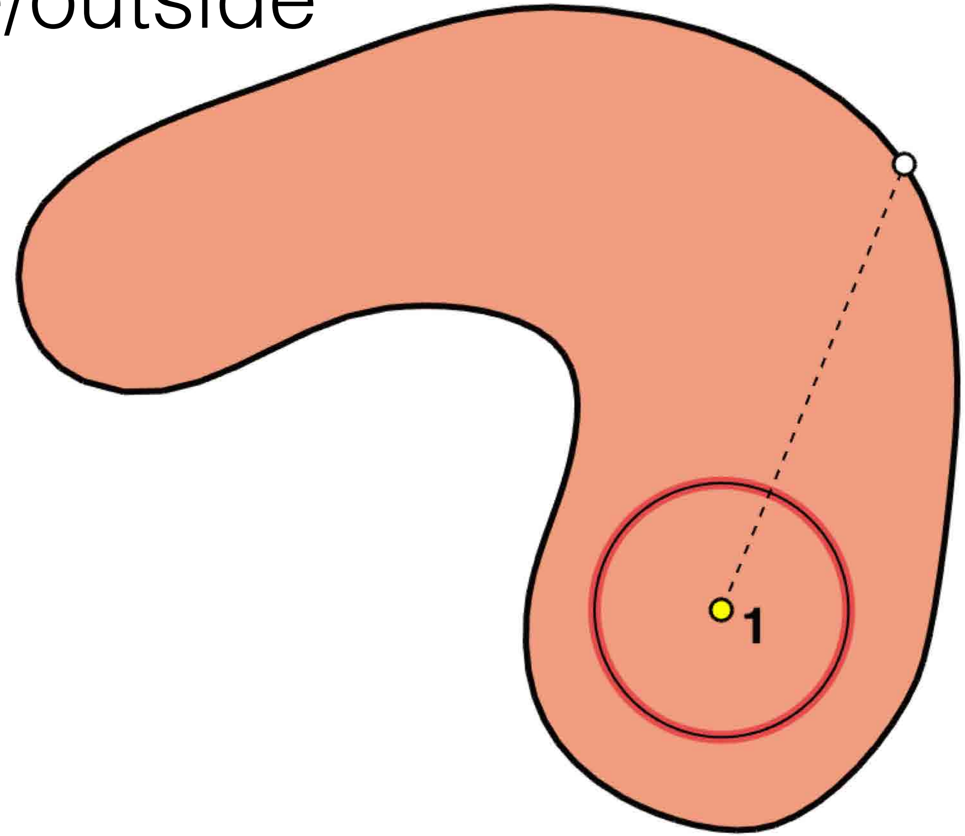
Height field visualization

Previous solutions are unsatisfactory



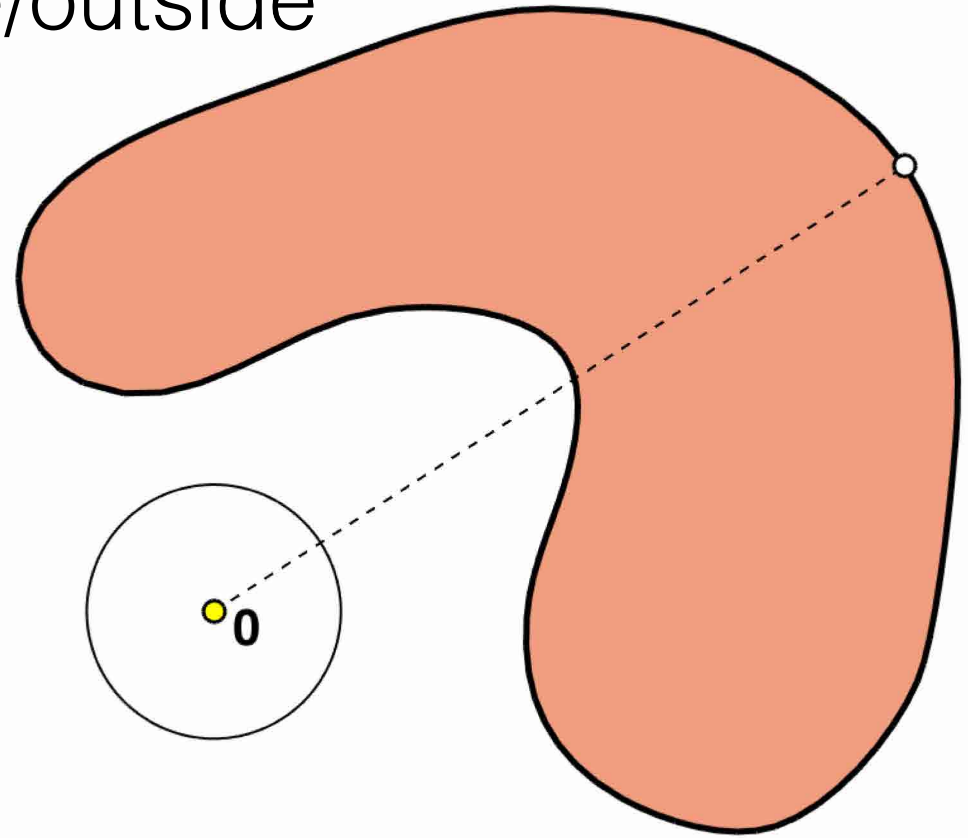
For “clean” shapes, classic winding number indicates inside/outside

$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_C d\theta$$



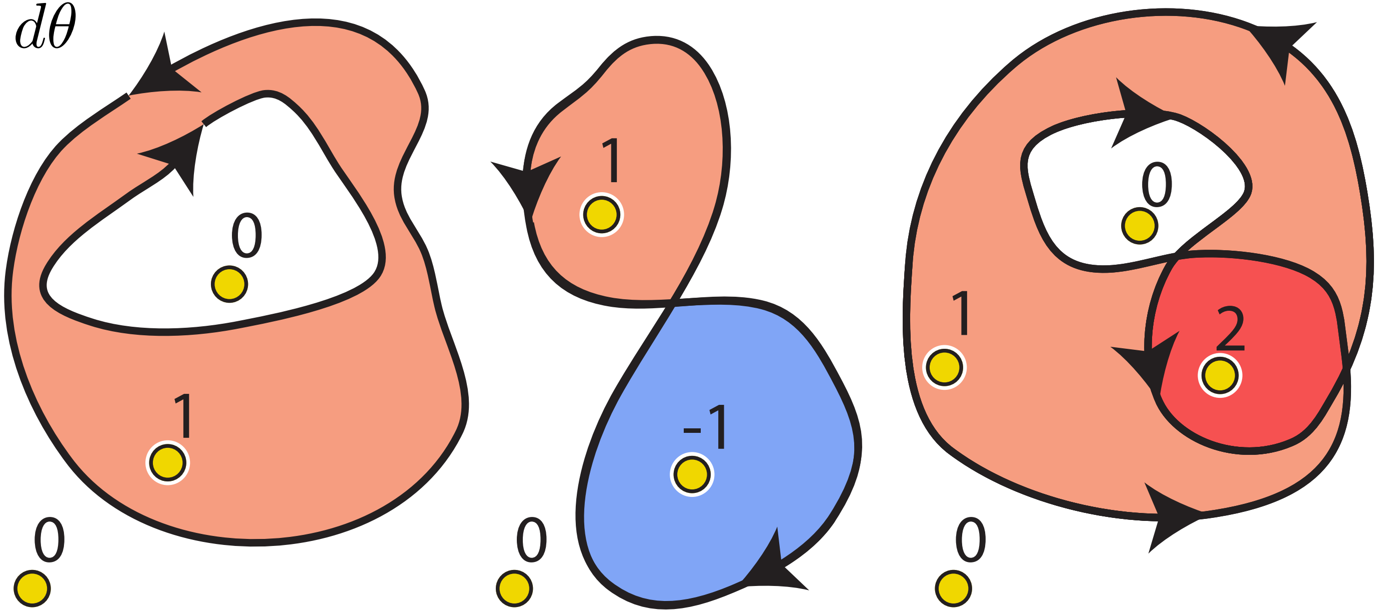
For “clean” shapes, classic winding number indicates inside/outside

$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_{\mathcal{C}} d\theta$$



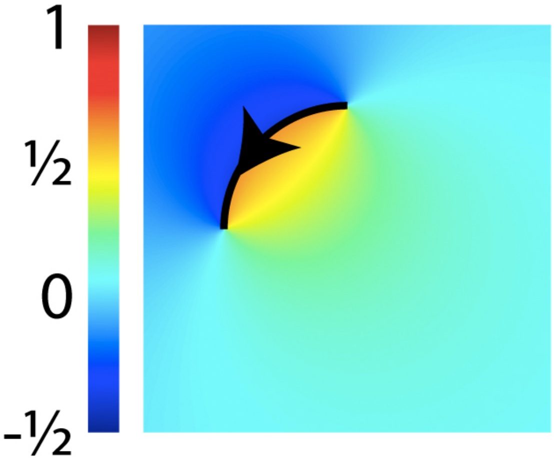
Classic winding number already handles a wide variety of shapes

$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_C d\theta$$



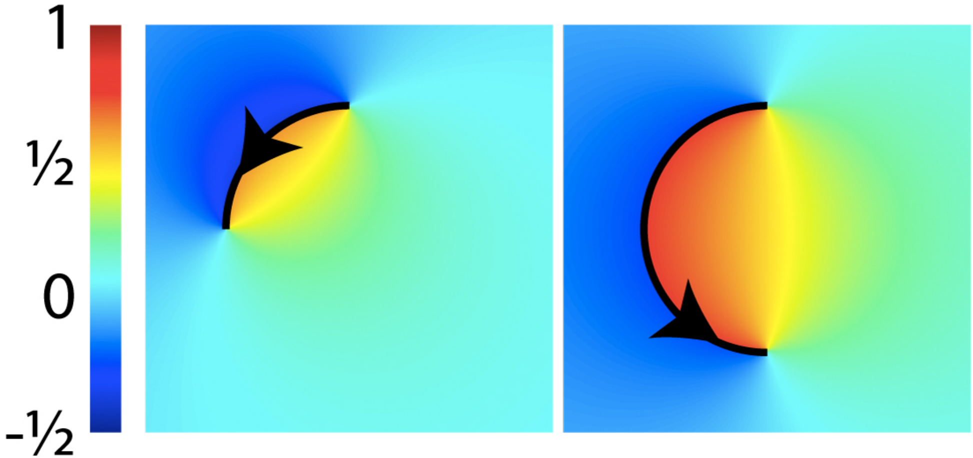
What happens if the shape is open?

$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_C d\theta$$



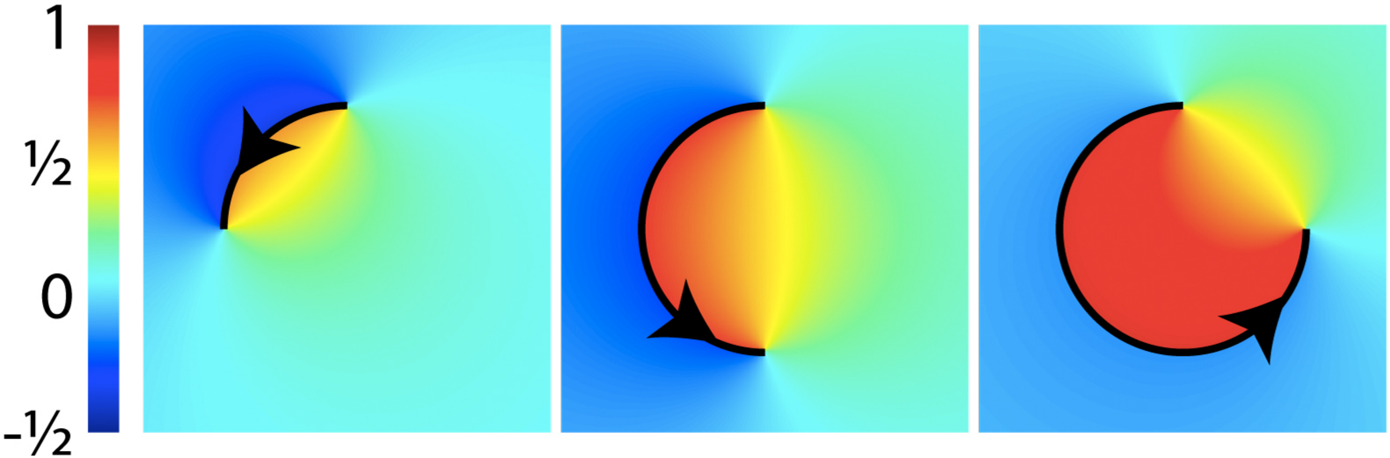
What happens if the shape is open?

$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_{\mathcal{C}} d\theta$$



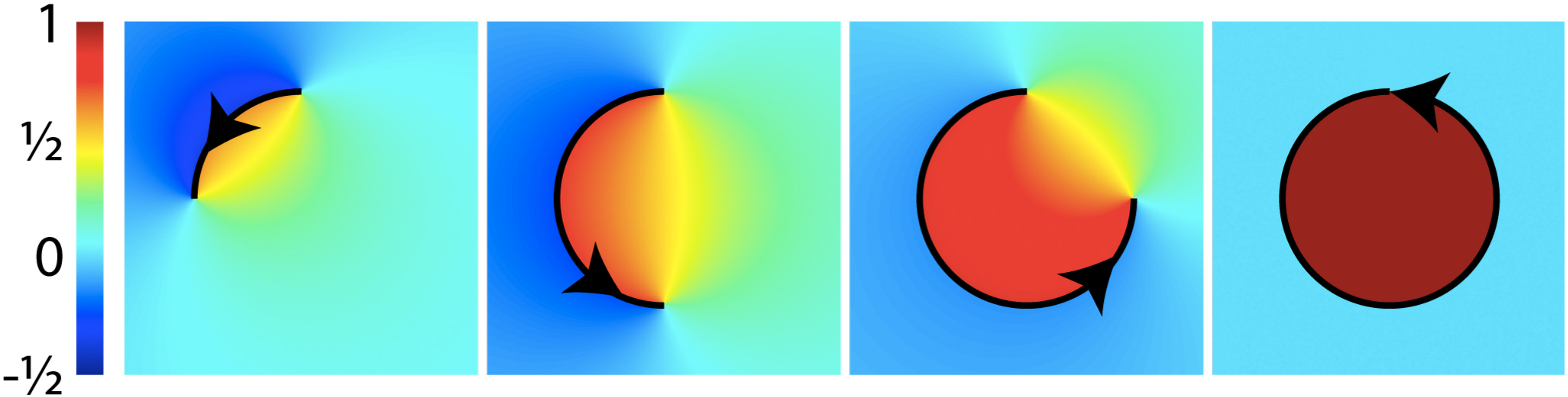
What happens if the shape is open?

$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_C d\theta$$



What happens if the shape is open?

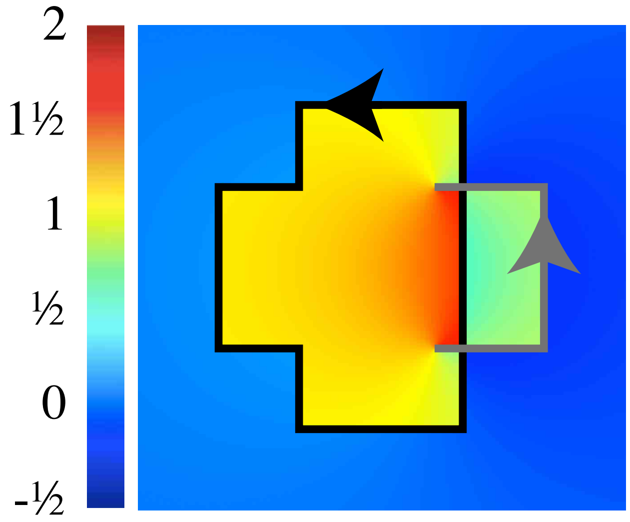
$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_C d\theta$$



Gracefully tends toward perfect indicator

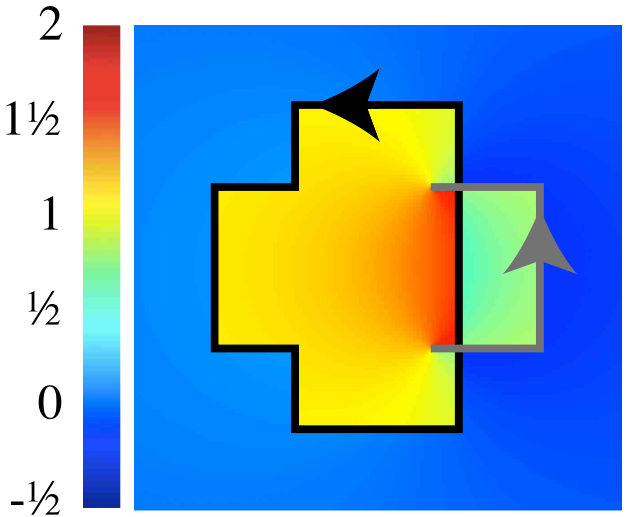
What if the shape is self-intersecting?

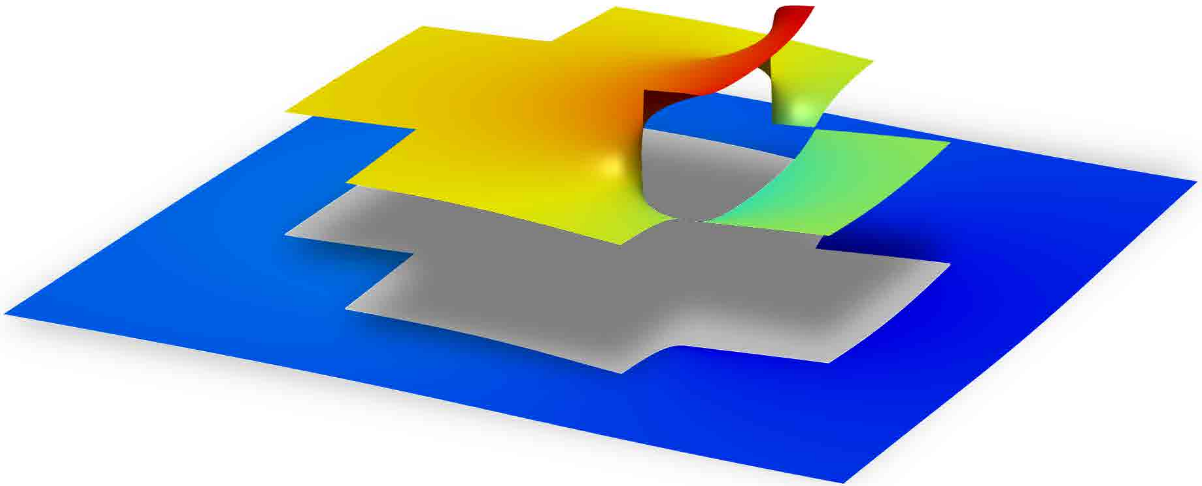
$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_{\mathcal{C}} d\theta$$



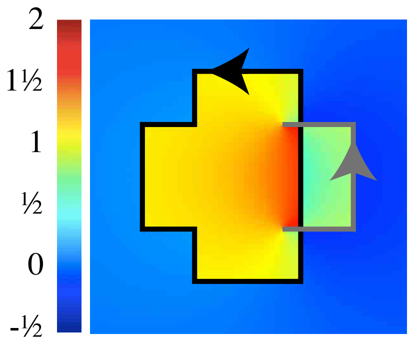
Winding number jumps across boundaries, otherwise harmonic!

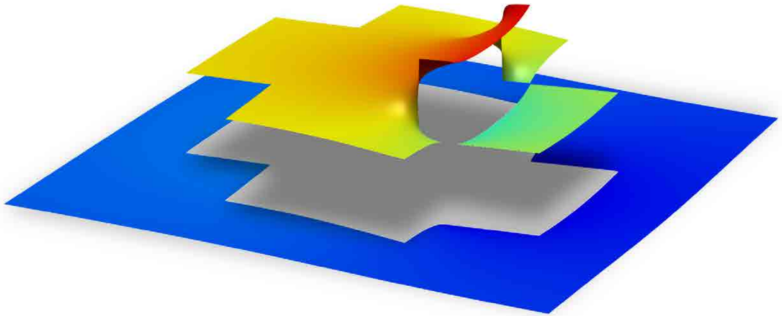
$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_{\mathcal{C}} d\theta \qquad\qquad \Delta w = 0$$

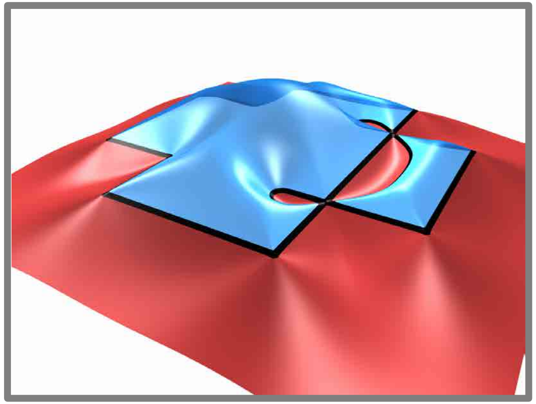




Other interpolating implicit functions may contain "surprise" oscillations





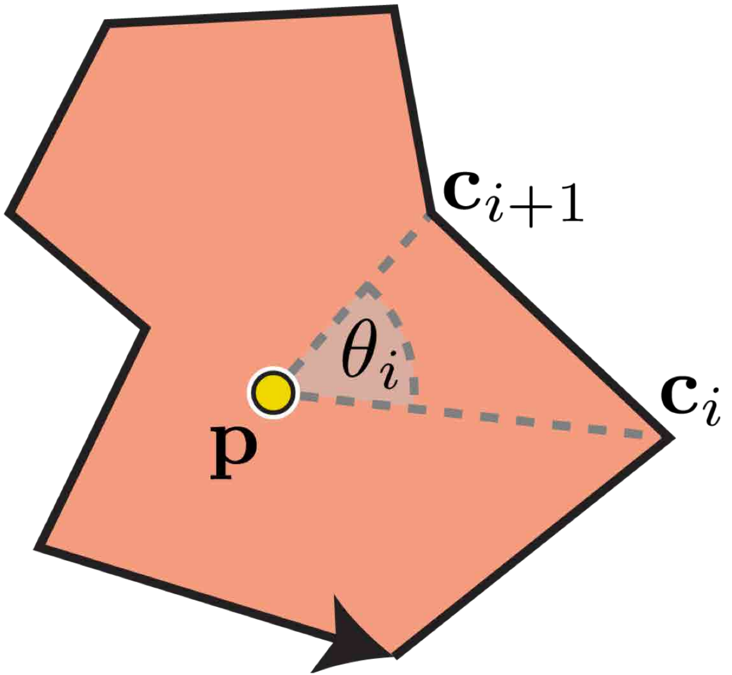


[Shen et al. 2004]

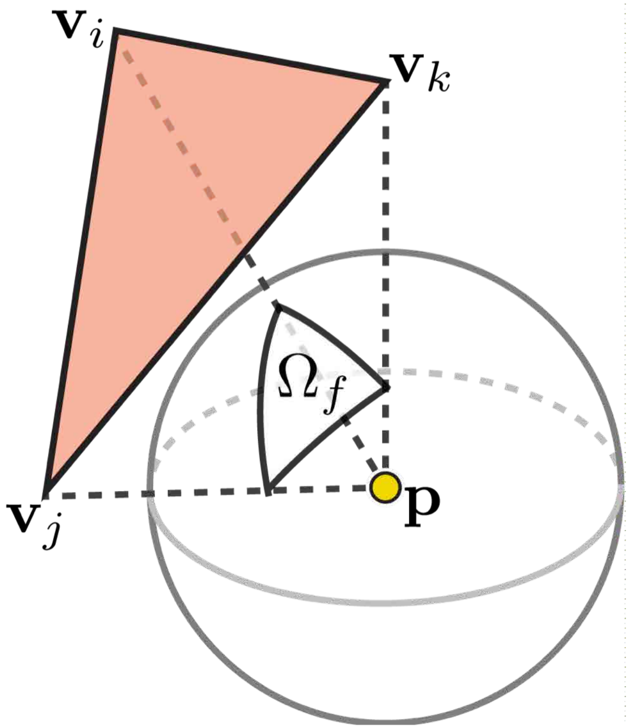
Discretization is simple and exact

$$w(\mathbf{p}) = \frac{1}{2\pi} \oint_{\mathcal{C}} d\theta$$

$$w(\mathbf{p}) = \frac{1}{2\pi} \sum_{i=1}^{n} \theta_i$$



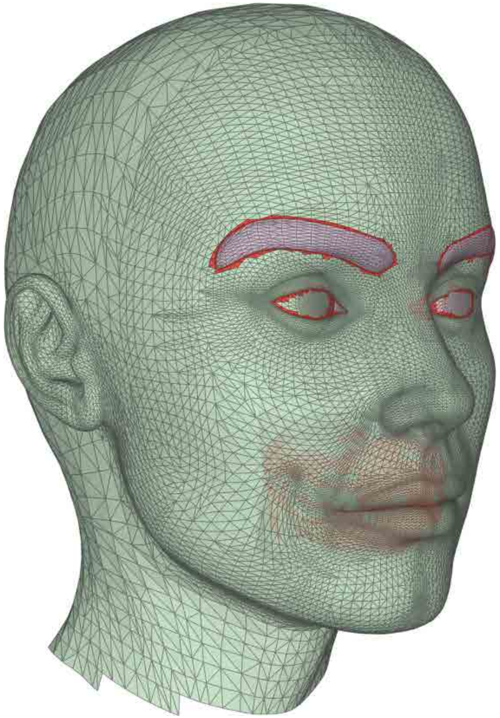
Generalizes elegantly to 3D via solid angle

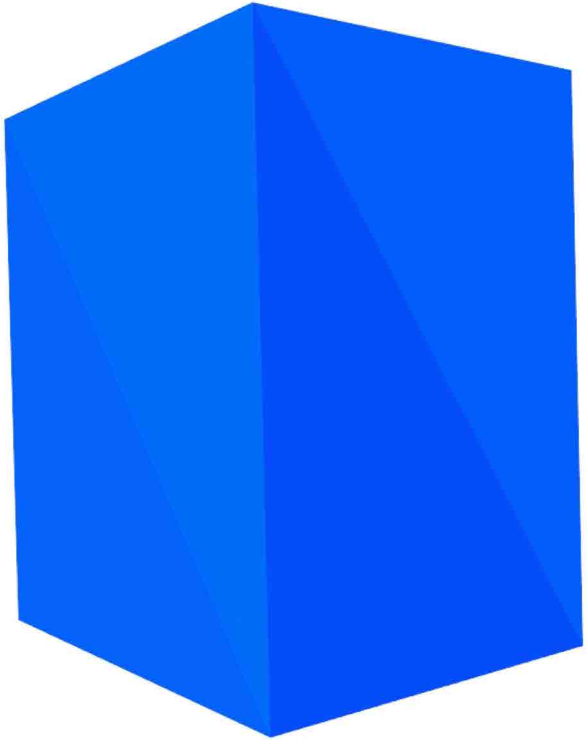


$$w(\mathbf{p}) = \frac{1}{4\pi} \iint_{\mathcal{S}} \sin(\phi) d\theta d\phi$$

$$w(\mathbf{p}) = \frac{1}{4\pi} \sum_{f=1}^{m} \Omega_f$$

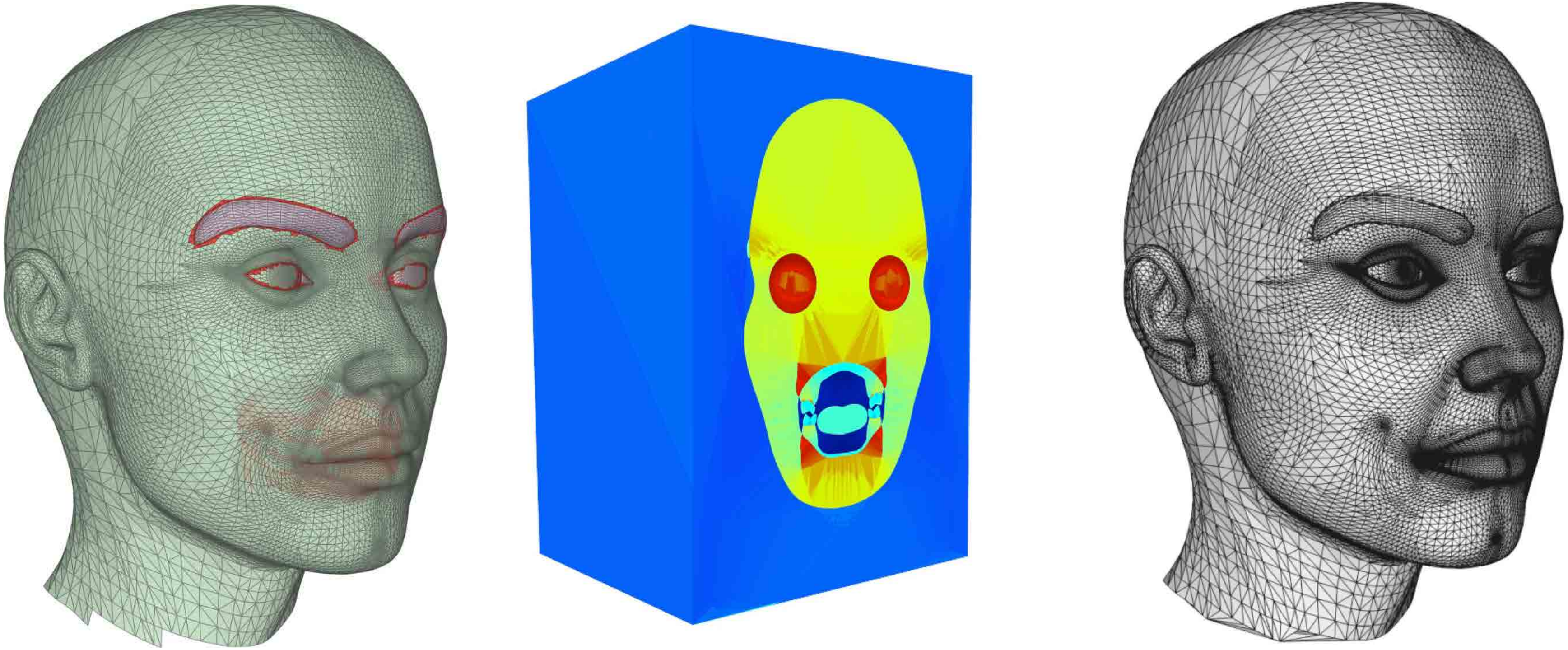
I adapt traditional algorithms and theory to work even in the presence of messy data





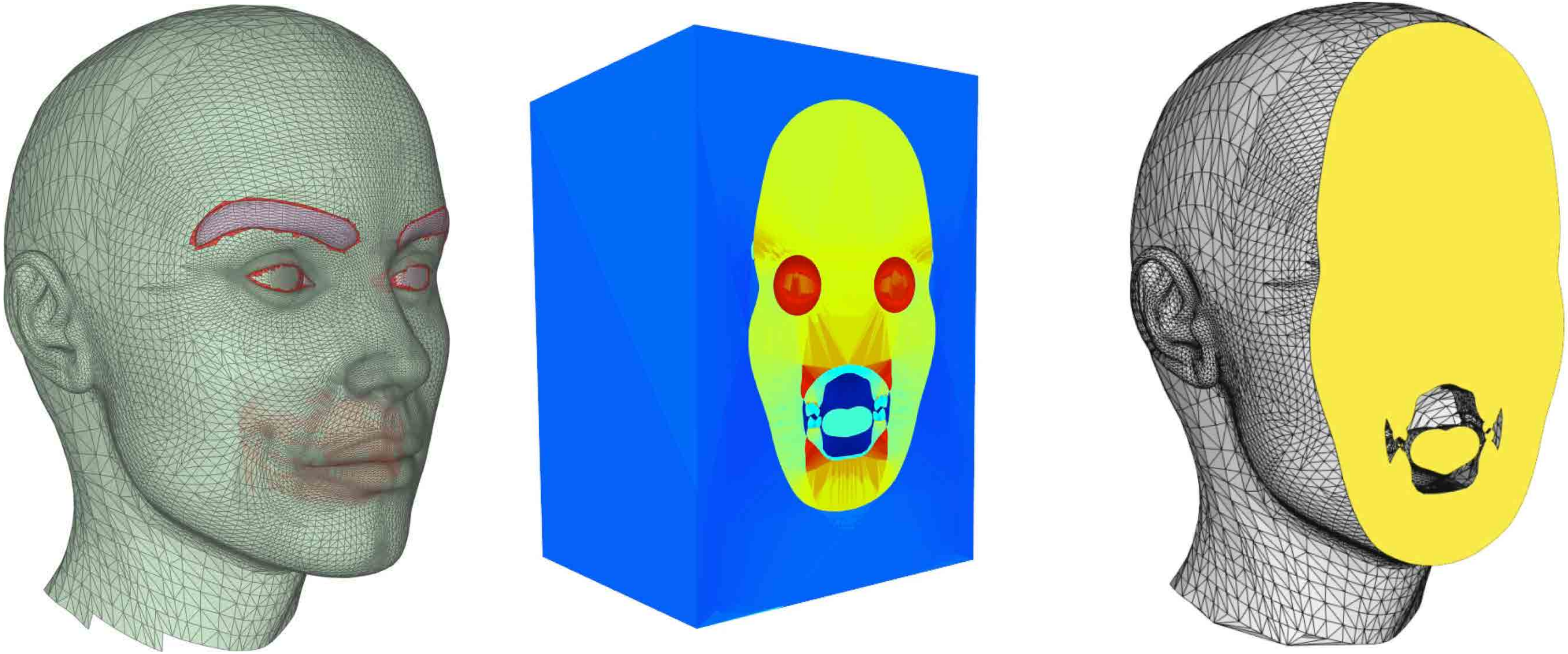
"Robust Inside-Outside Segmentation using Generalized Winding Numbers"
[J., Kavan, & Sorkine-Hornung 2013]

Enables volumetric discretization, in turn enables better physics, rendering, …



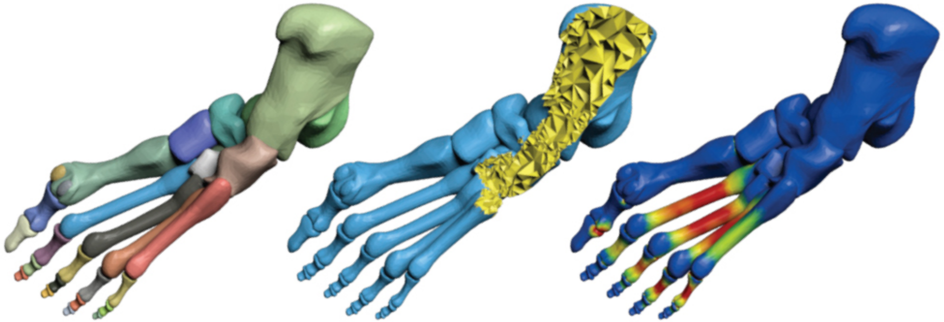
"Robust Inside-Outside Segmentation using Generalized Winding Numbers"
[J., Kavan, & Sorkine-Hornung 2013]

Enables volumetric discretization, in turn enables better physics, rendering, …



"Robust Inside-Outside Segmentation using Generalized Winding Numbers"
[J., Kavan, & Sorkine-Hornung 2013]

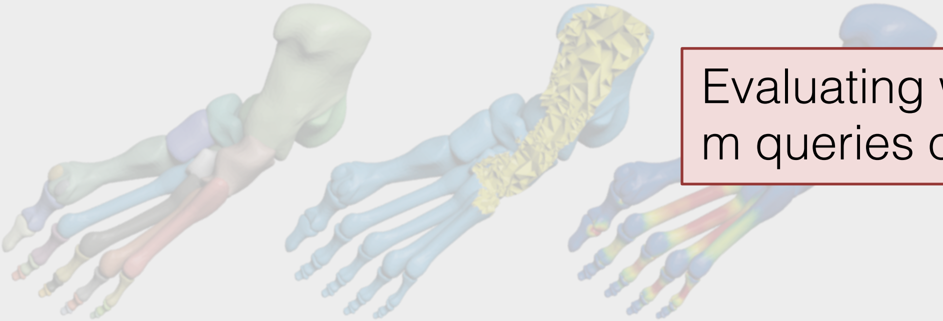
Robust inside/outside enables high-level processing

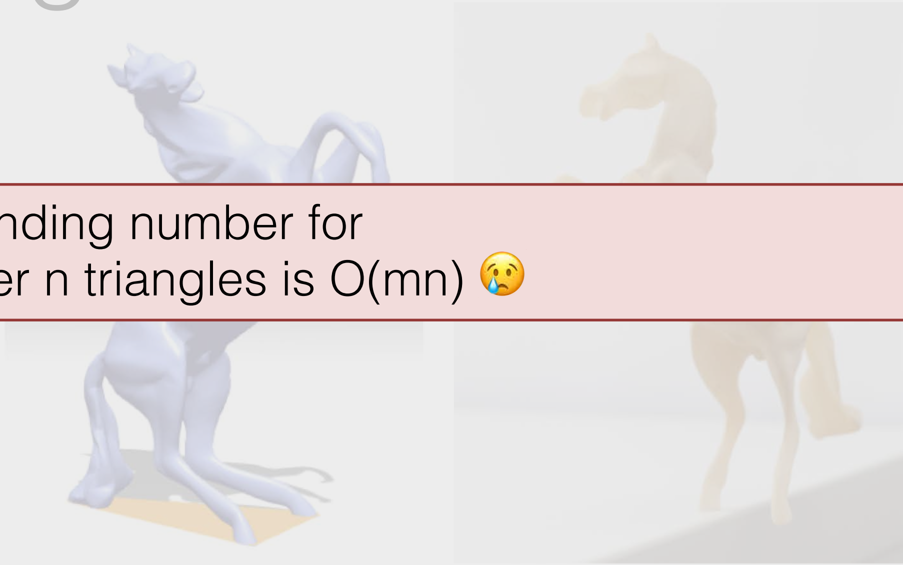


Structural analysis

Functionality-driven deformation

Robust inside/outside enables high-level processing





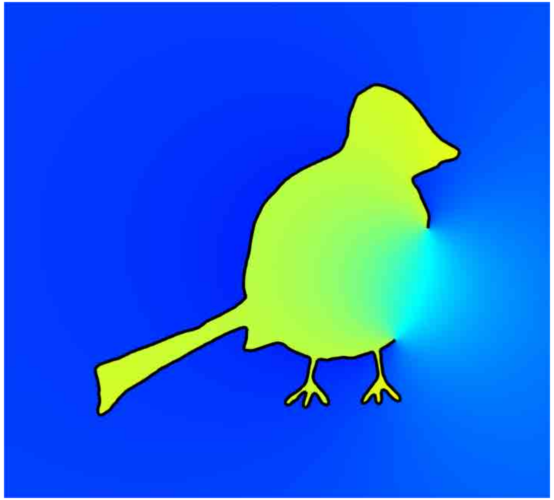
Evaluating winding number for
m queries over n triangles is O(mn) 😢

Structural analysis

Functionality-driven deformation

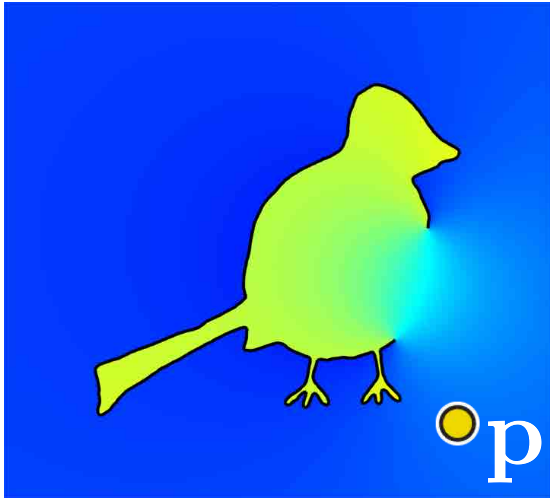
Exploit interesting fact for speedup

$\mathcal{C}$



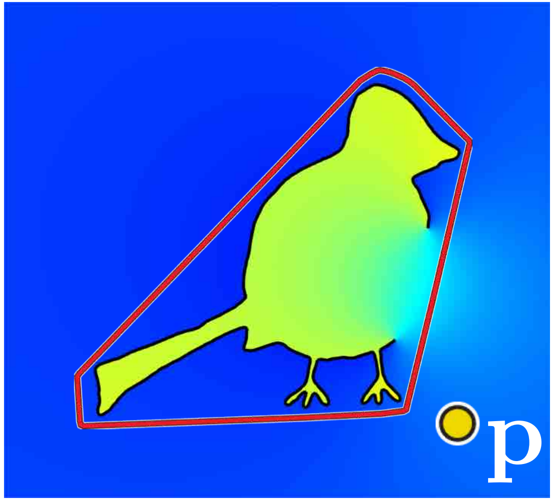
Exploit interesting fact for speedup

$\mathcal{C}$



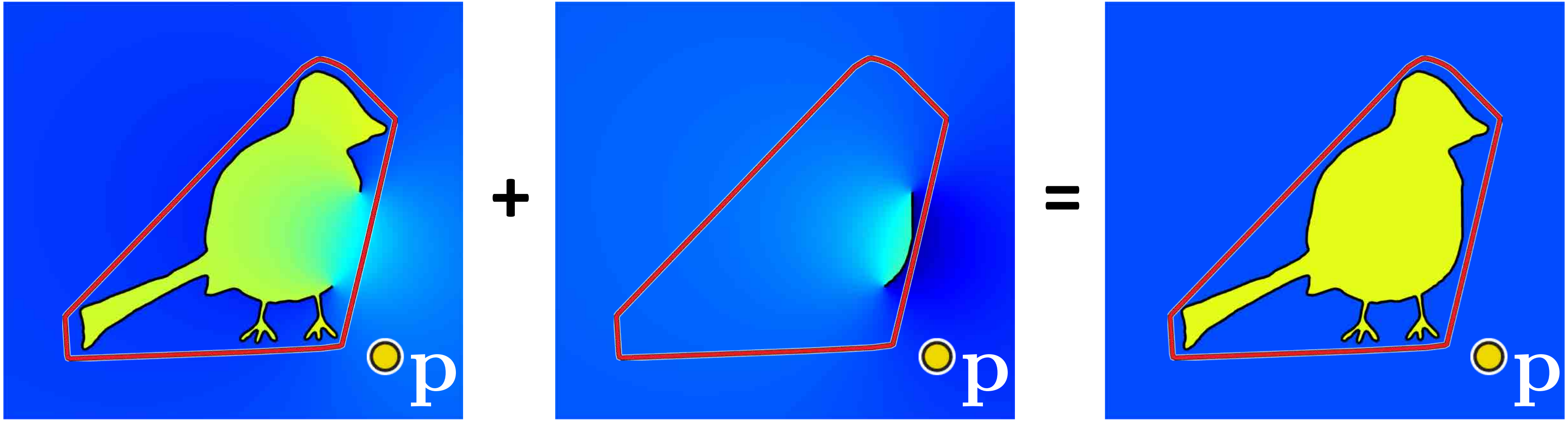
Exploit interesting fact for speedup

$\mathcal{C}$



Exploit interesting fact for speedup

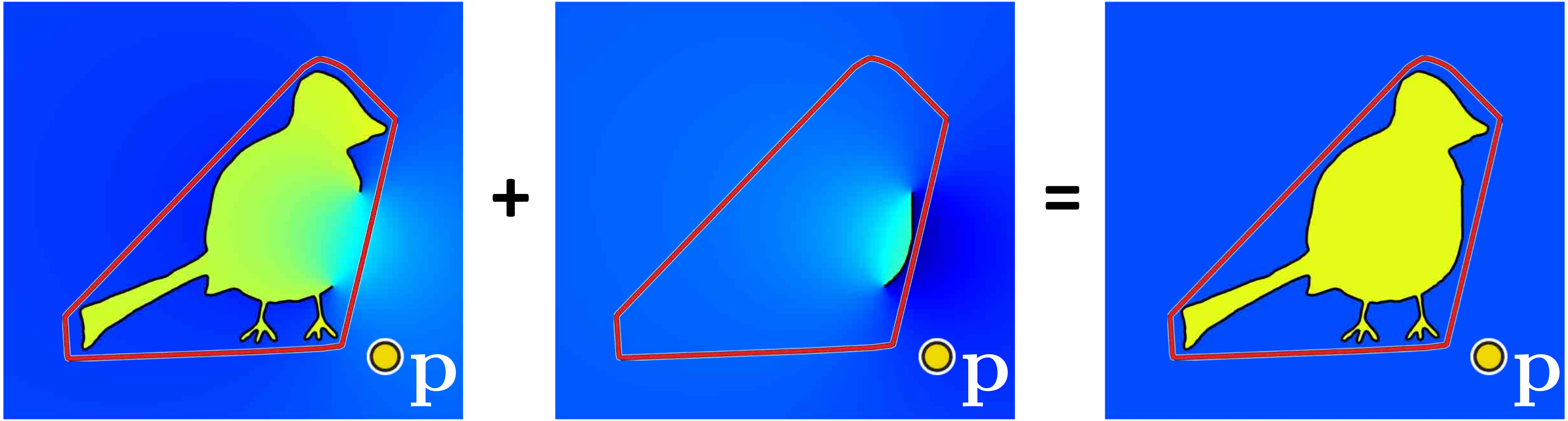
$\mathcal{C}$ $\qquad$ $\bar{\mathcal{C}}$ $\qquad$ $\mathcal{C} \cup \bar{\mathcal{C}}$



$$w_{\mathcal{C}\cup\bar{\mathcal{C}}}(\mathbf{p}) = 0$$

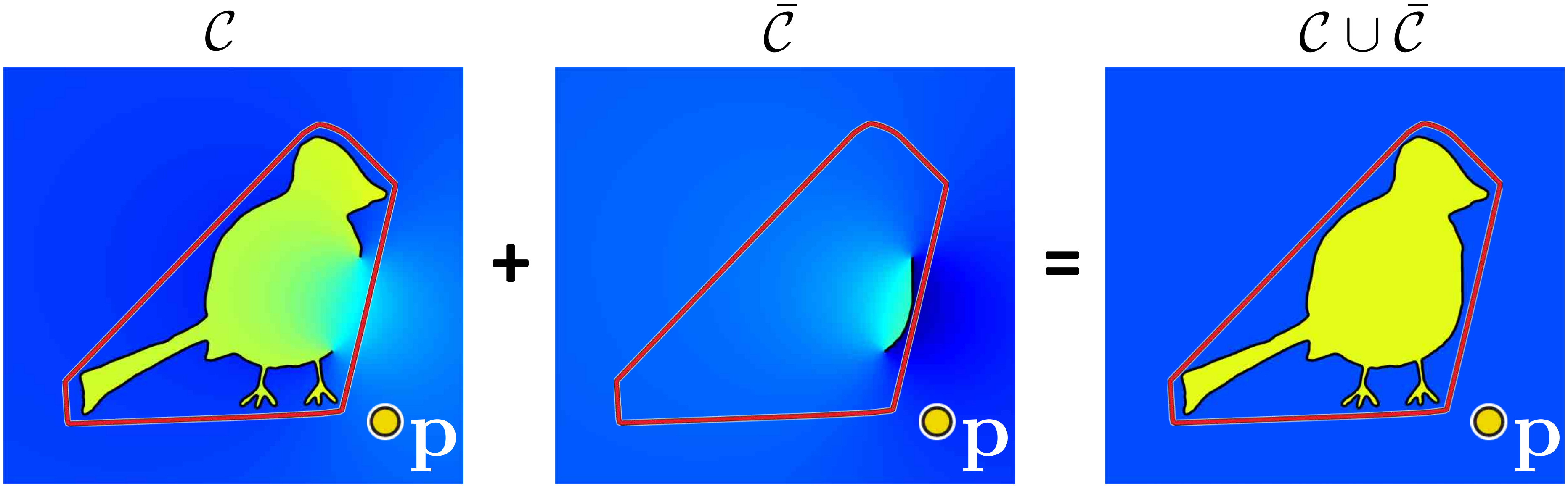
Exploit interesting fact for speedup

$\mathcal{C}$ $\quad\quad$ $\bar{\mathcal{C}}$ $\quad\quad$ $\mathcal{C} \cup \bar{\mathcal{C}}$



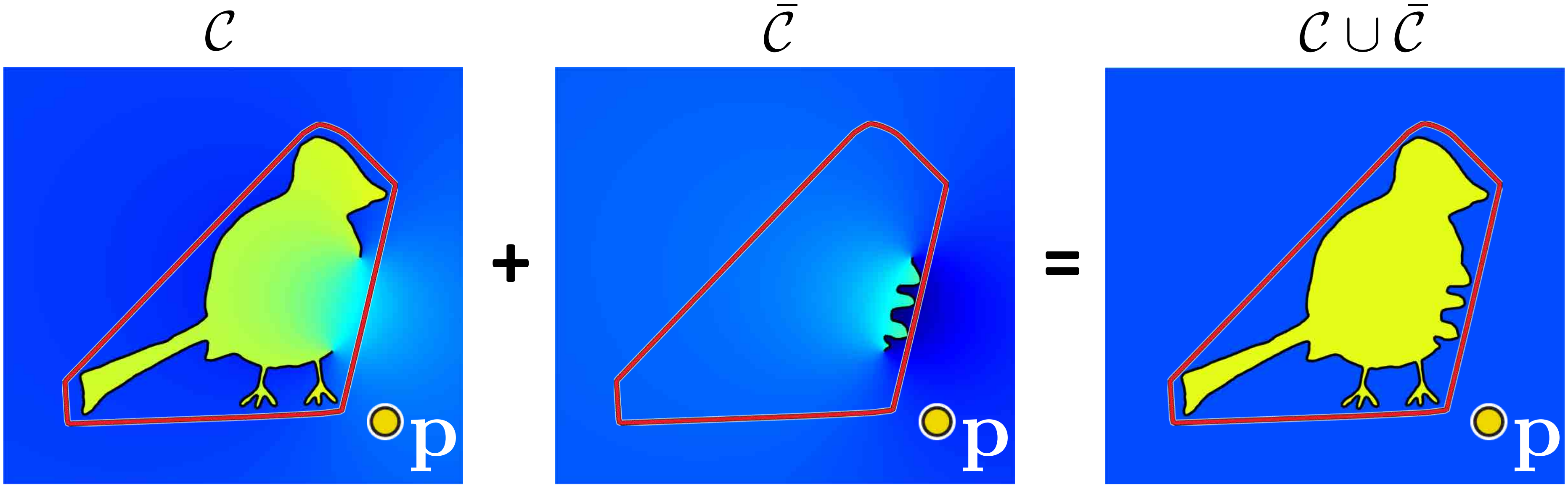
$$w_{\mathcal{C}}(\mathbf{p}) + w_{\bar{\mathcal{C}}}(\mathbf{p}) = w_{\mathcal{C}\cup\bar{\mathcal{C}}}(\mathbf{p}) = 0$$

Exploit interesting fact for speedup



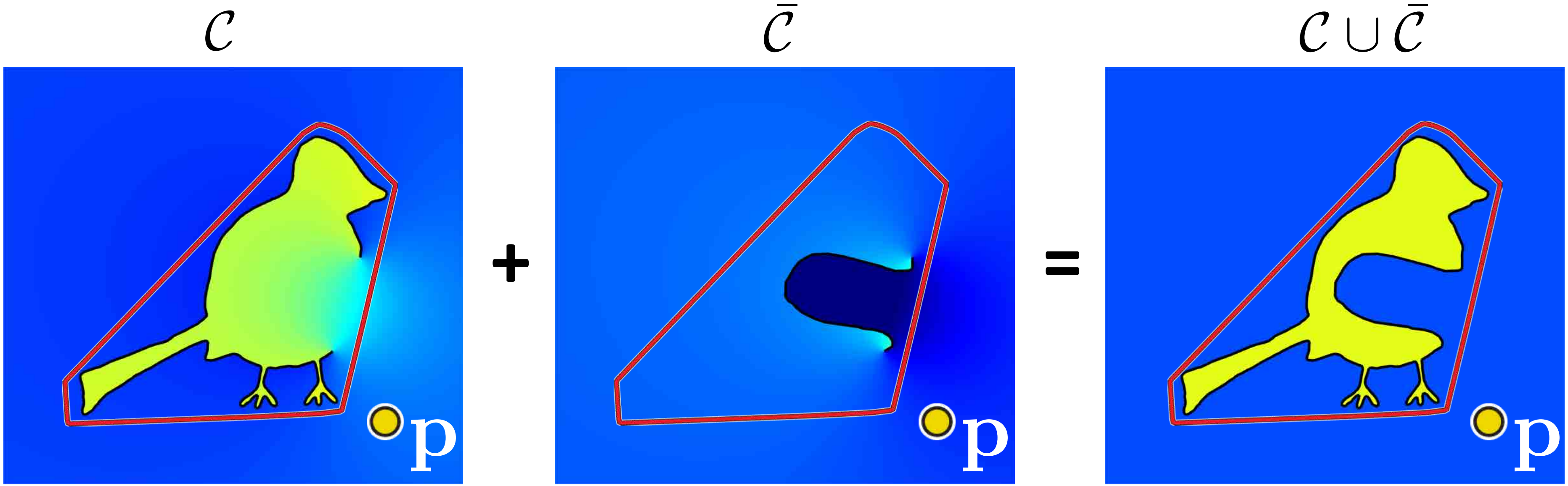
$$w_{\mathcal{C}}(\mathbf{p}) = -w_{\bar{\mathcal{C}}}(\mathbf{p})$$

Exploit interesting fact for speedup



$$w_{\mathcal{C}}(\mathbf{p}) = -w_{\bar{\mathcal{C}}}(\mathbf{p})$$

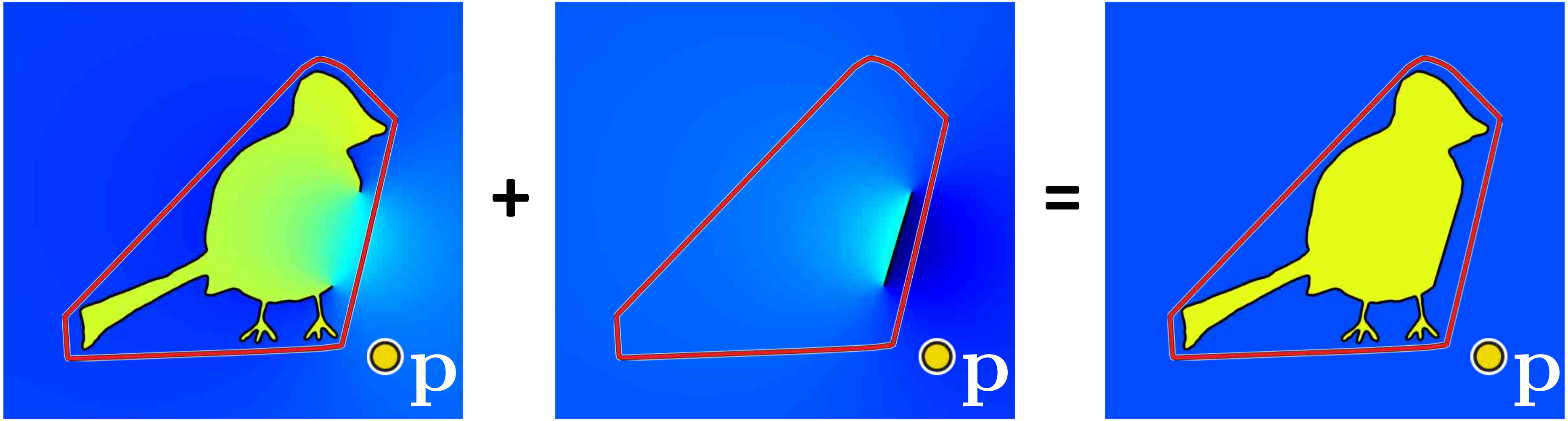
Exploit interesting fact for speedup



$$w_{\mathcal{C}}(\mathbf{p}) = -w_{\bar{\mathcal{C}}}(\mathbf{p})$$

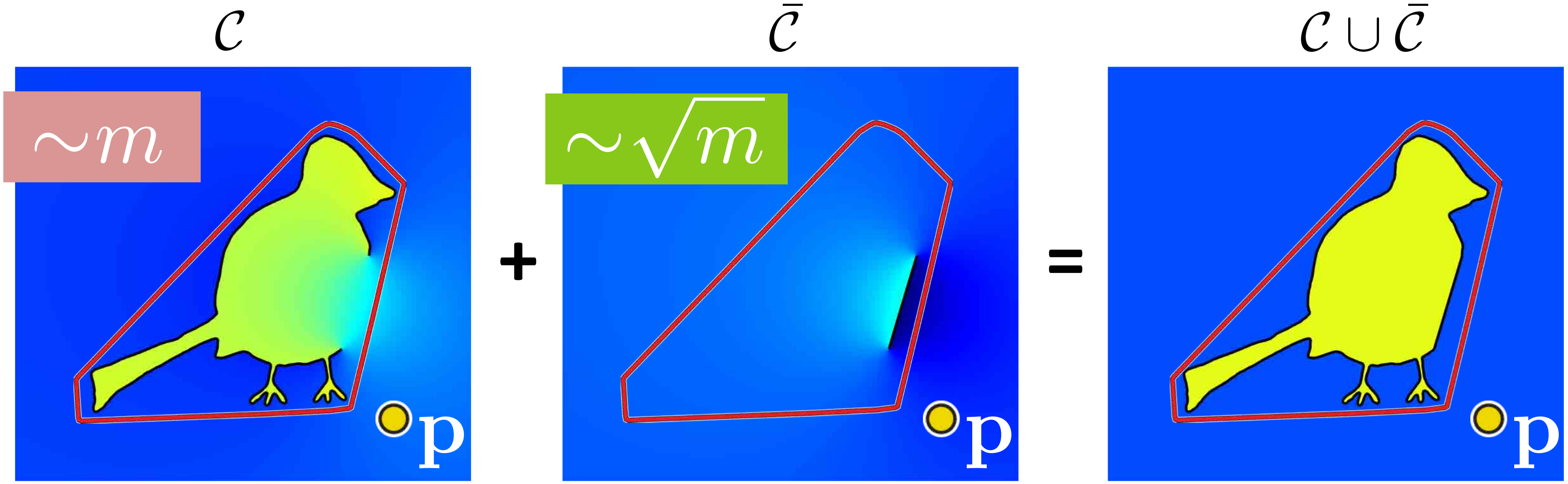
Exploit interesting fact for speedup

$\mathcal{C}$ $\qquad\qquad\qquad$ $\bar{\mathcal{C}}$ $\qquad\qquad\qquad$ $\mathcal{C} \cup \bar{\mathcal{C}}$



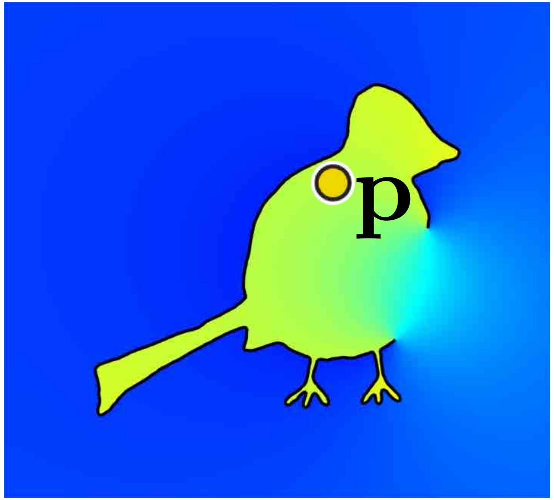
$$w_{\mathcal{C}}(\mathbf{p}) = -w_{\bar{\mathcal{C}}}(\mathbf{p})$$

Exploit interesting fact for speedup

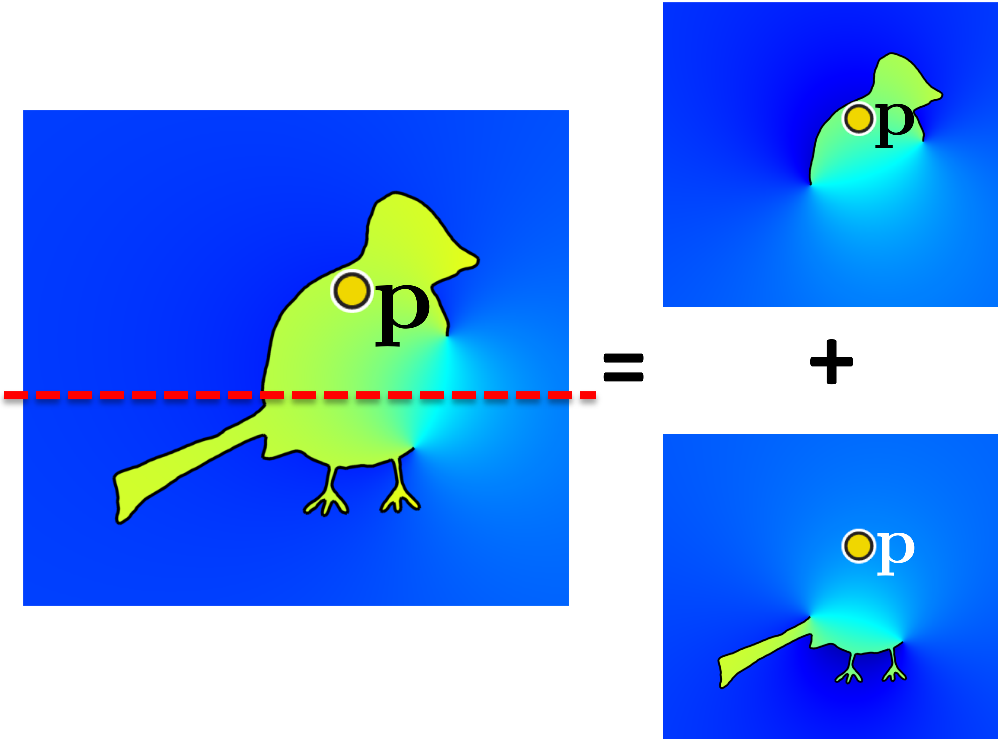


$$w_{\mathcal{C}}(\mathbf{p}) = -w_{\bar{\mathcal{C}}}(\mathbf{p})$$

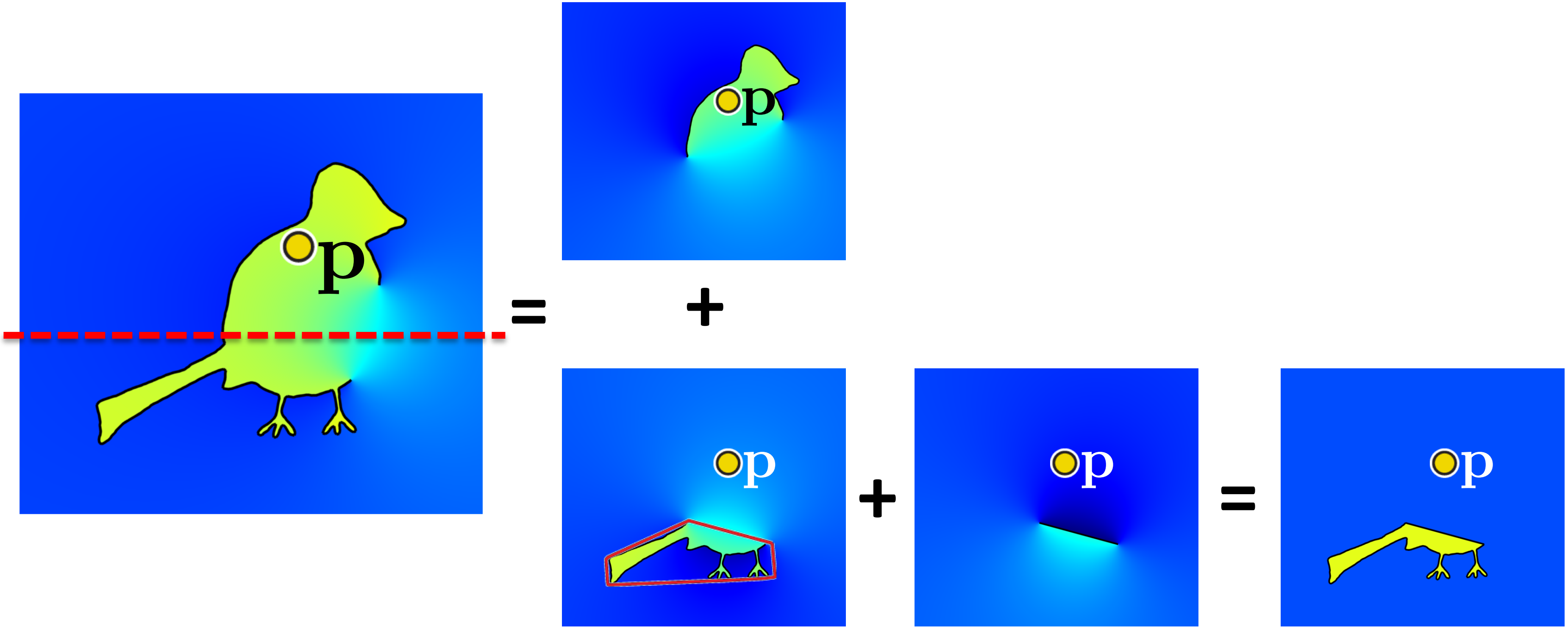
Divide and conquer!



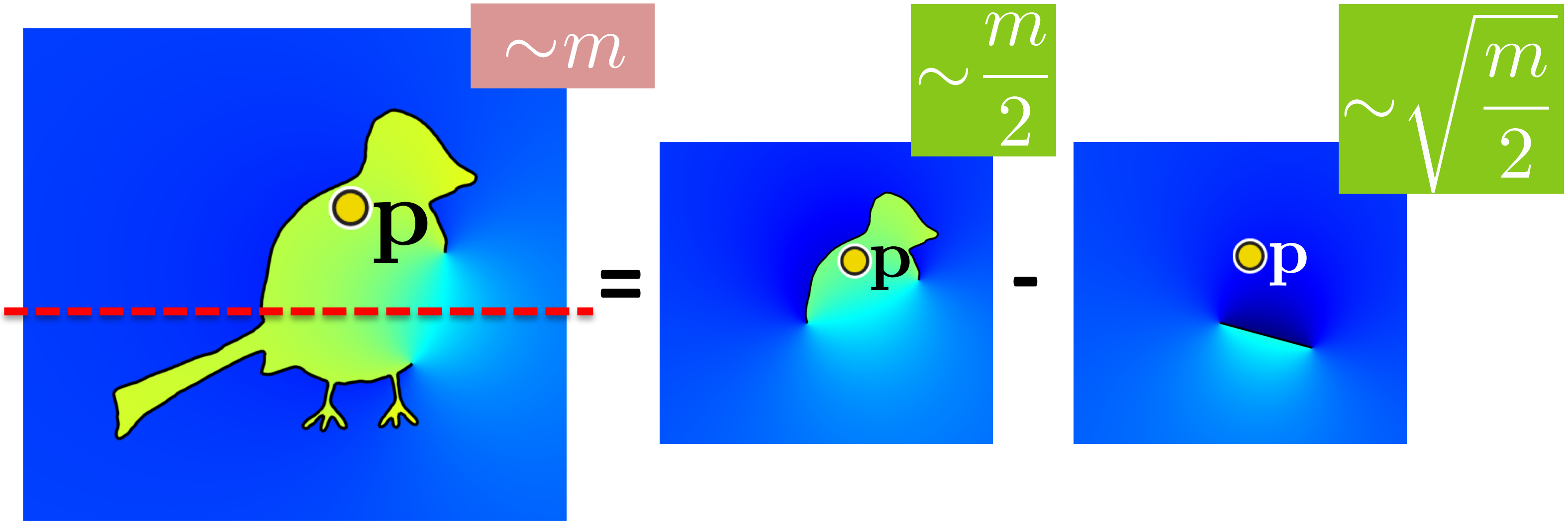
Divide and conquer!



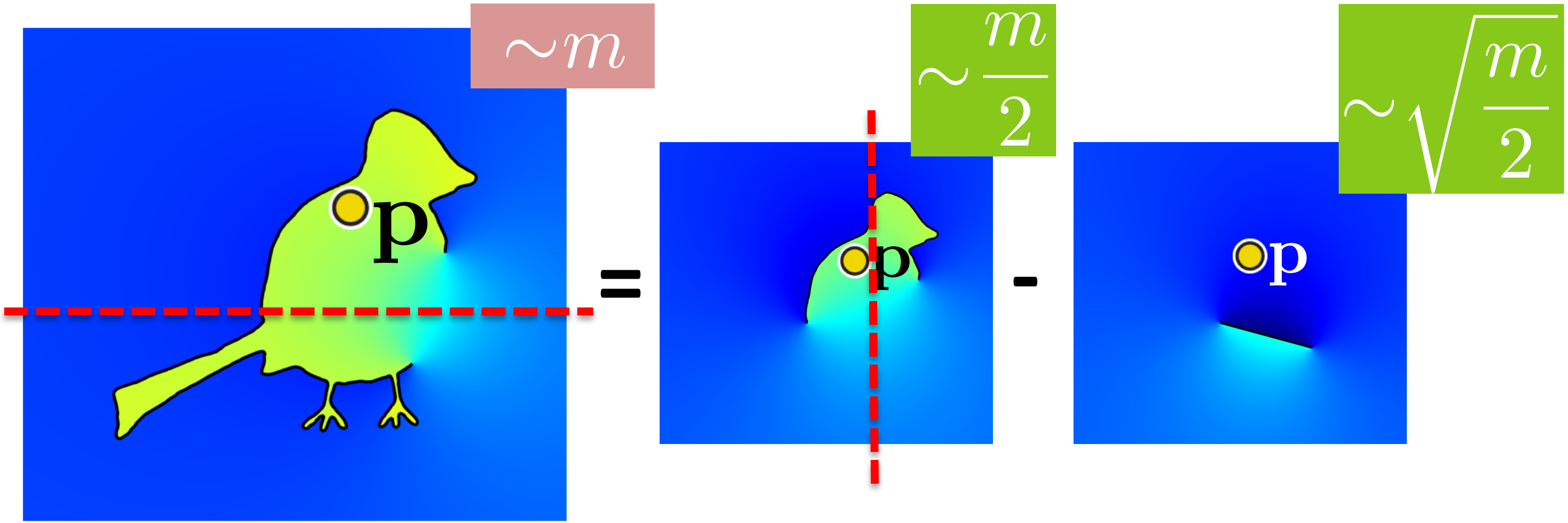
Divide and conquer!



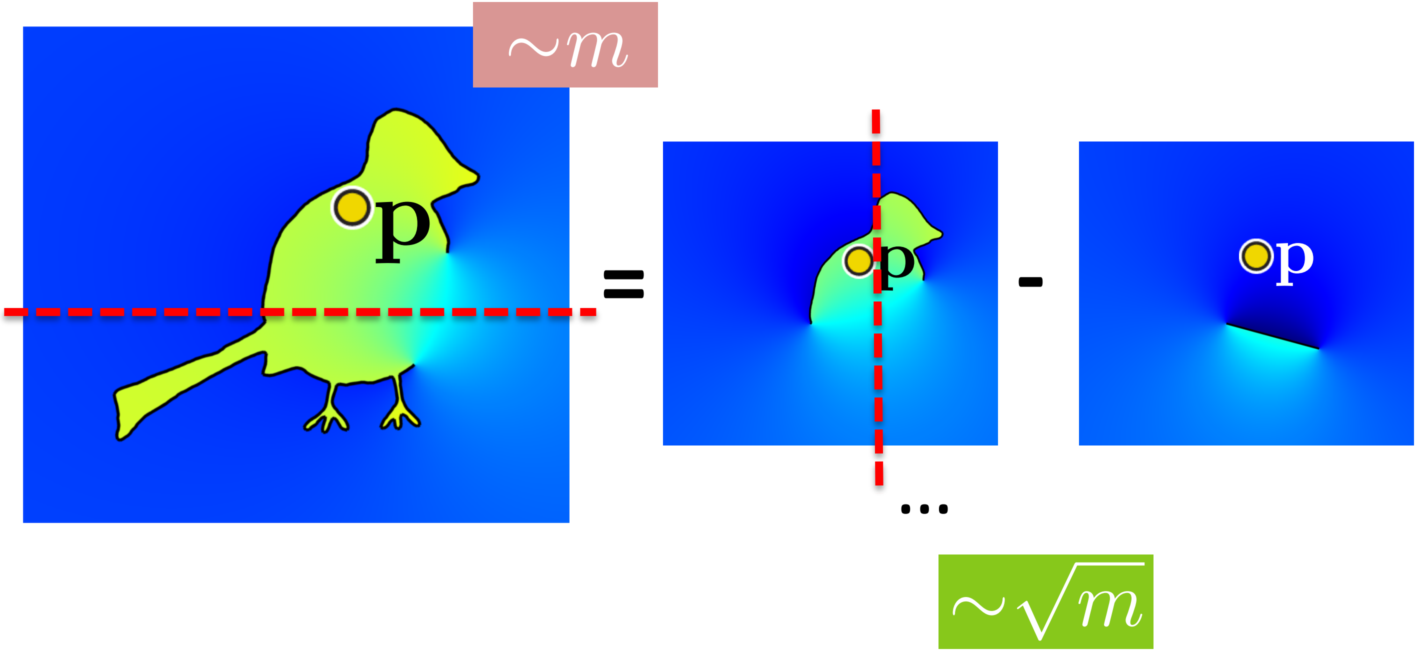
Divide and conquer!



Divide and conquer!

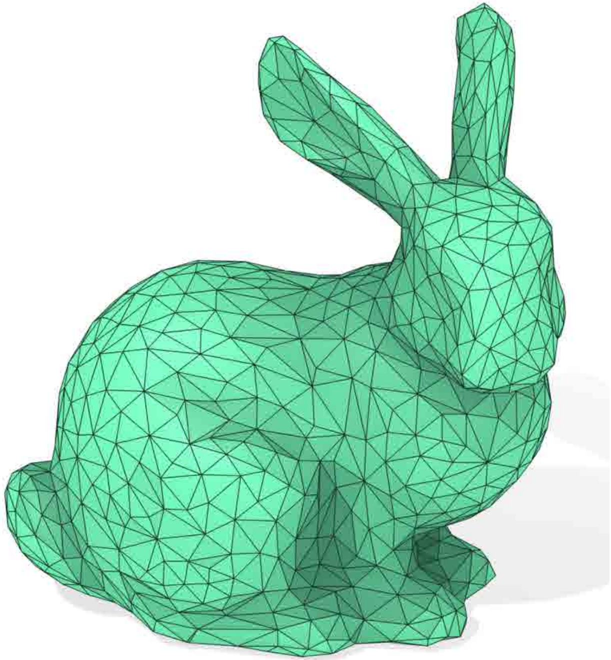


Divide and conquer!



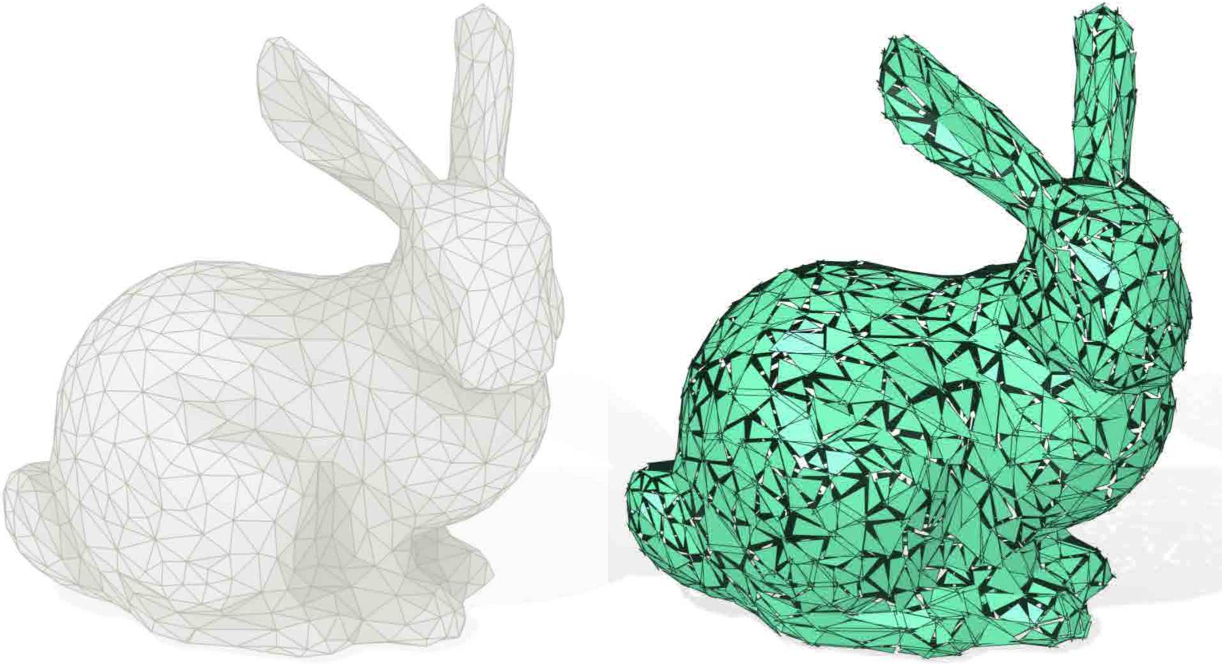
Trouble in paradise!

Trick depends on size of boundary being << size of mesh



For the imaginary “clean” watertight mesh this is true

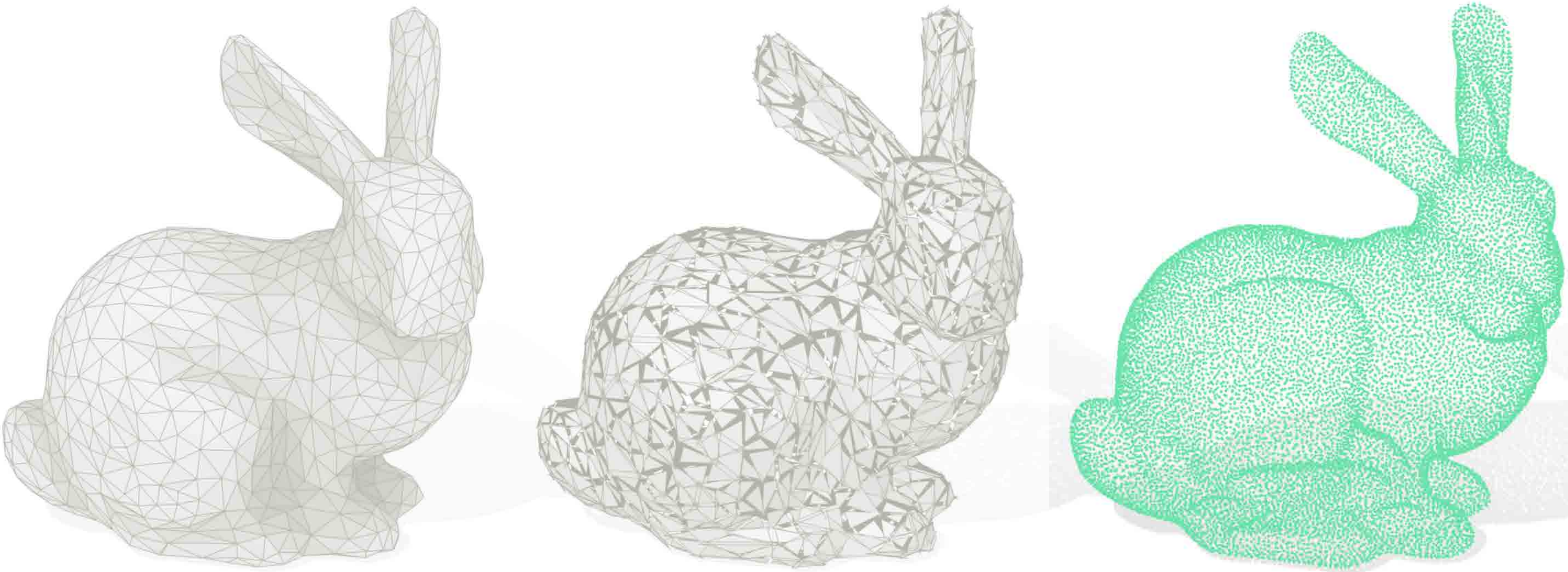
Trouble in paradise!



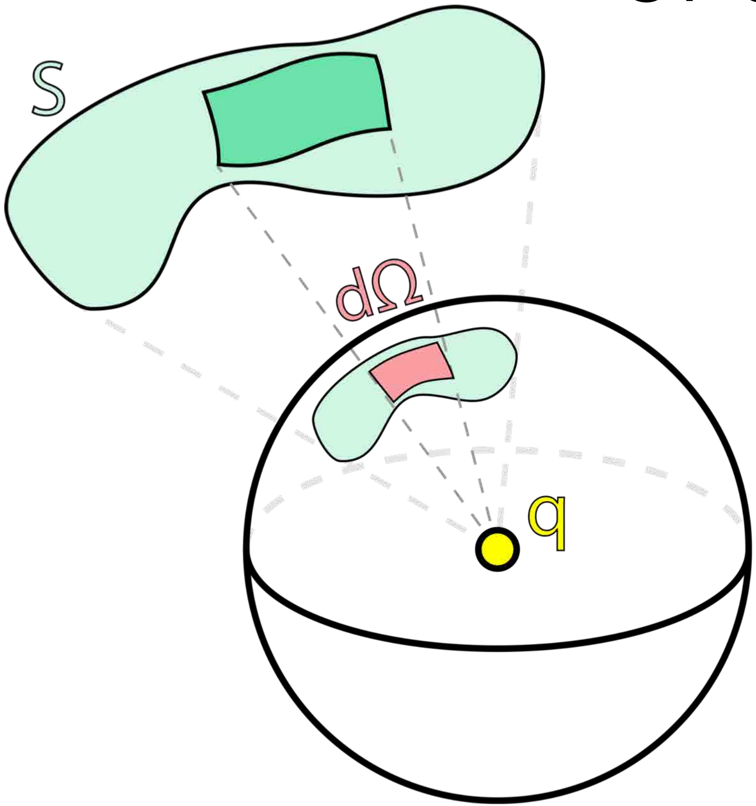
For disconnect “soup”,
in the worst case,
size of boundary >= mesh

Trouble in paradise!

For point clouds,
there is *no connectivity*
"all boundary"

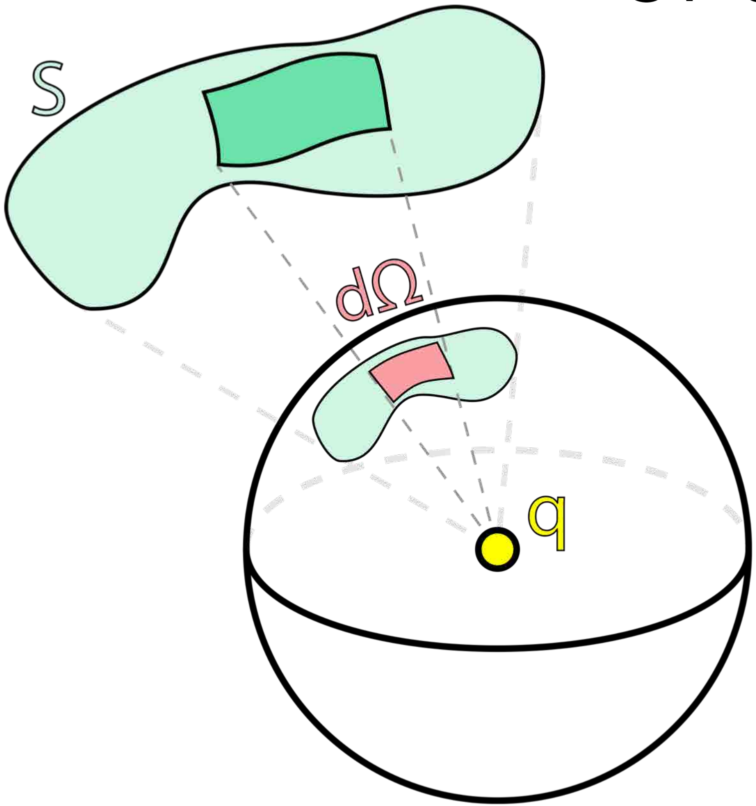


Winding number is sum of solid angles in 3D



$$w_S(\mathbf{q}) = \frac{1}{4\pi}\int_S d\Omega$$

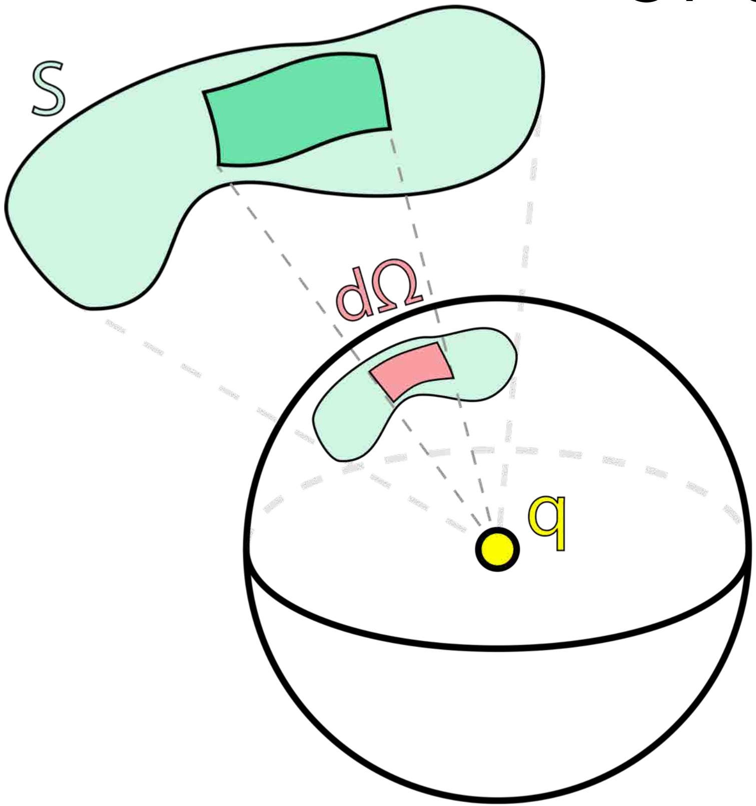
Winding number is sum of solid angles in 3D



$$w_S(\mathbf{q}) = \frac{1}{\boxed{4\pi}} \int_S d\Omega$$

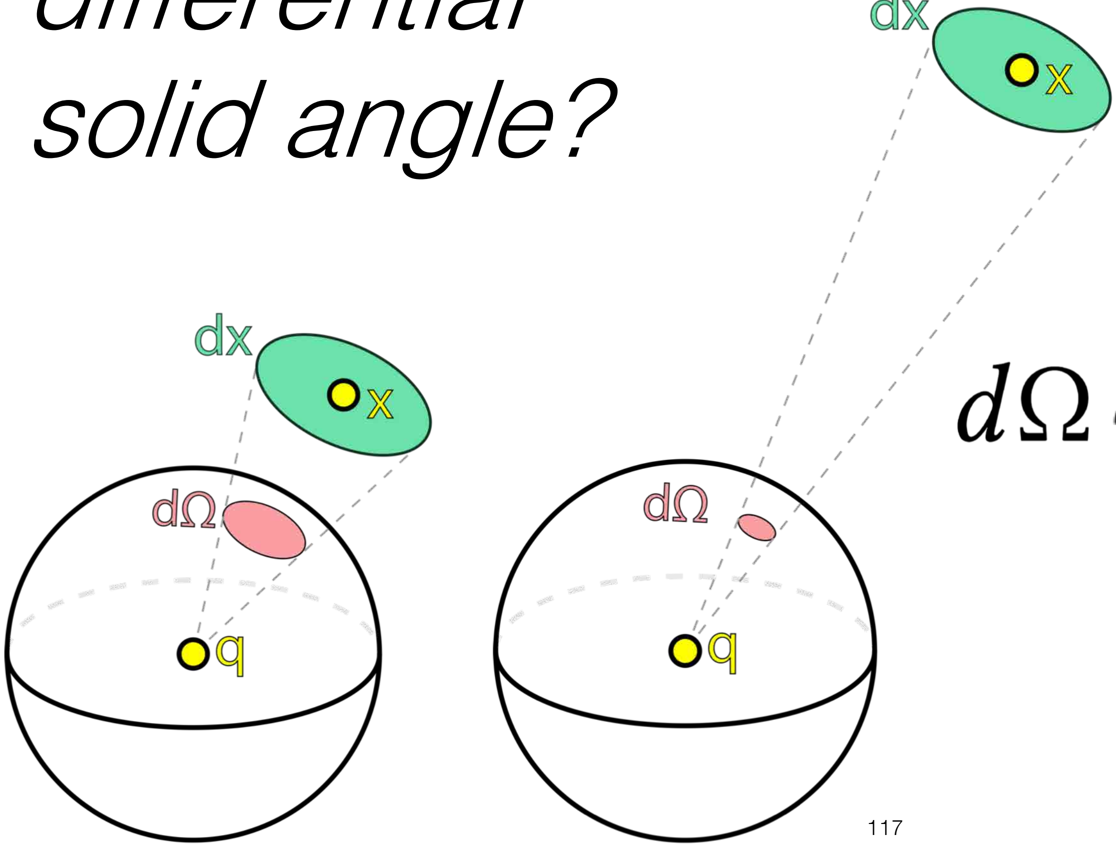
(surface area of unit sphere)

Winding number is sum of solid angles in 3D



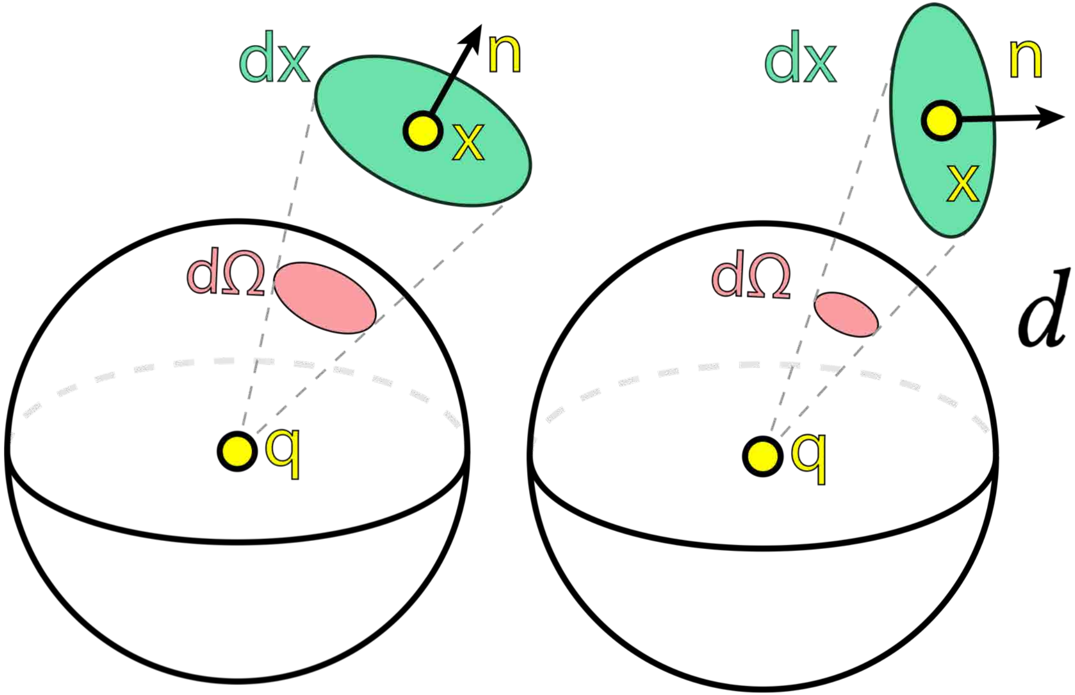
$$w_S(\mathbf{q}) = \frac{1}{4\pi} \int_S \boxed{d\Omega}$$

What can we say about this *differential solid angle?*



$$d\Omega \sim \frac{1}{\|\mathbf{x}-\mathbf{q}\|^2}\, d\mathbf{x}$$

What can we say about this *differential solid angle?*



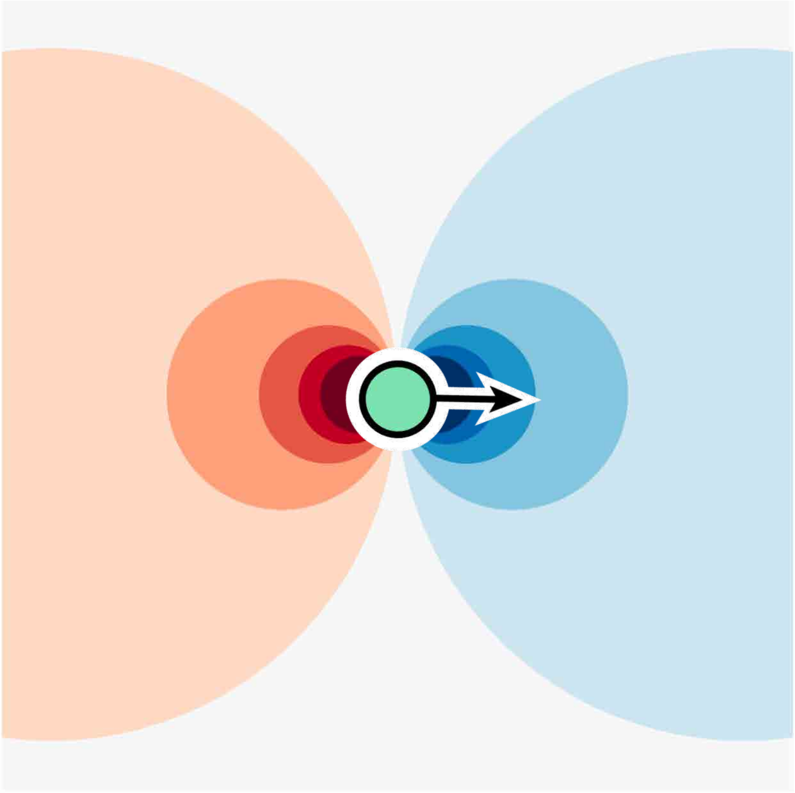
$$d\Omega \sim \mathbf{n} \cdot \frac{\mathbf{x} - \mathbf{q}}{\|\mathbf{x} - \mathbf{q}\|} \, d\mathbf{x}$$

differential solid angle is a dipole

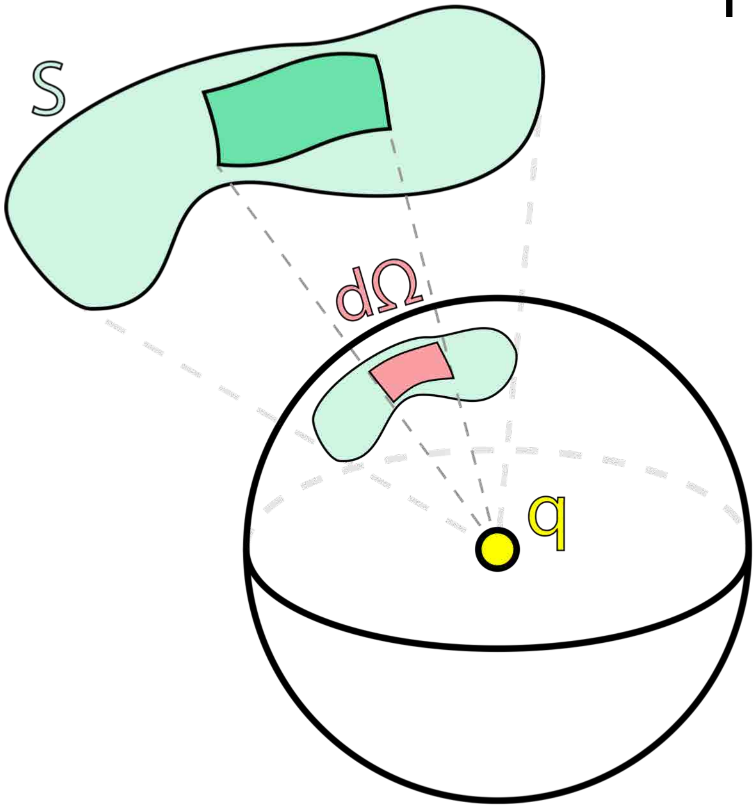
$$d\Omega = \frac{1}{\|\mathbf{x}-\mathbf{q}\|^2}\,\mathbf{n}\cdot\frac{\mathbf{x}-\mathbf{q}}{\|\mathbf{x}-\mathbf{q}\|}\,d\mathbf{x}$$

differential solid angle is a dipole

$$d\Omega = \frac{(\mathbf{x} - \mathbf{q}) \cdot \mathbf{n}}{\|\mathbf{x} - \mathbf{q}\|^3} d\mathbf{x}$$

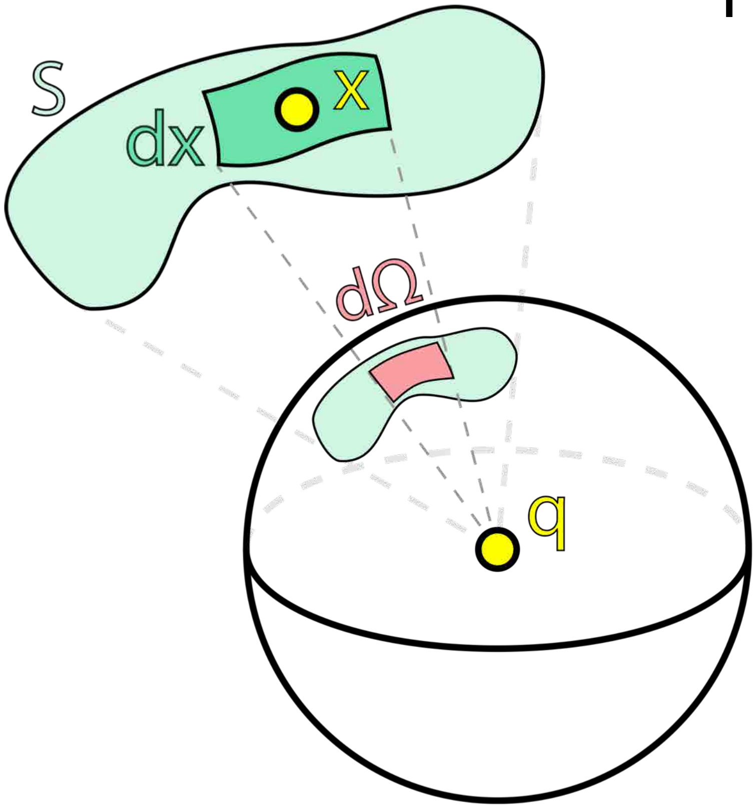


Winding number integrates dipoles on surface



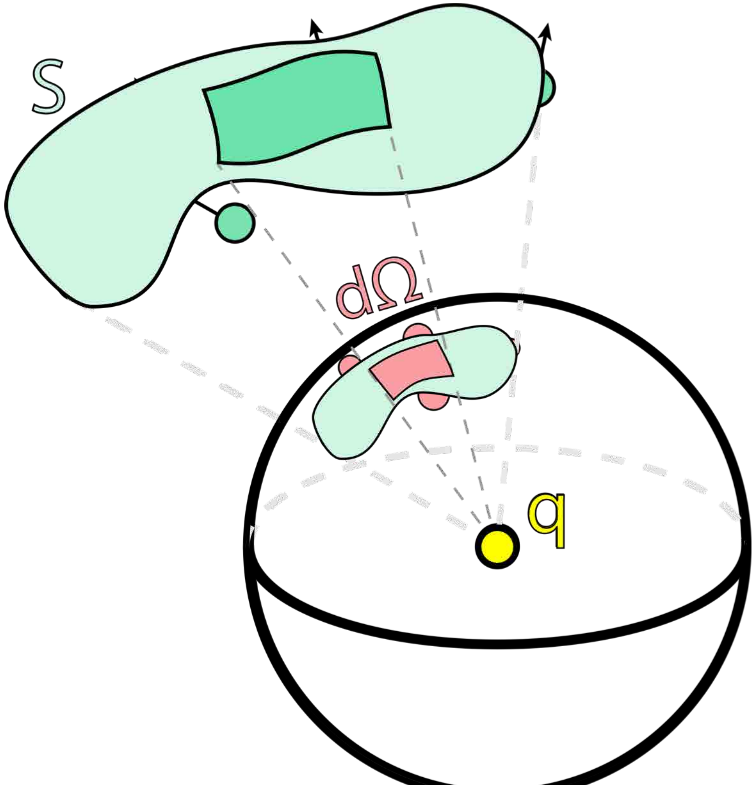
$$w_S(\mathbf{q}) = \frac{1}{4\pi} \int_S d\Omega$$

Winding number integrates dipoles on surface



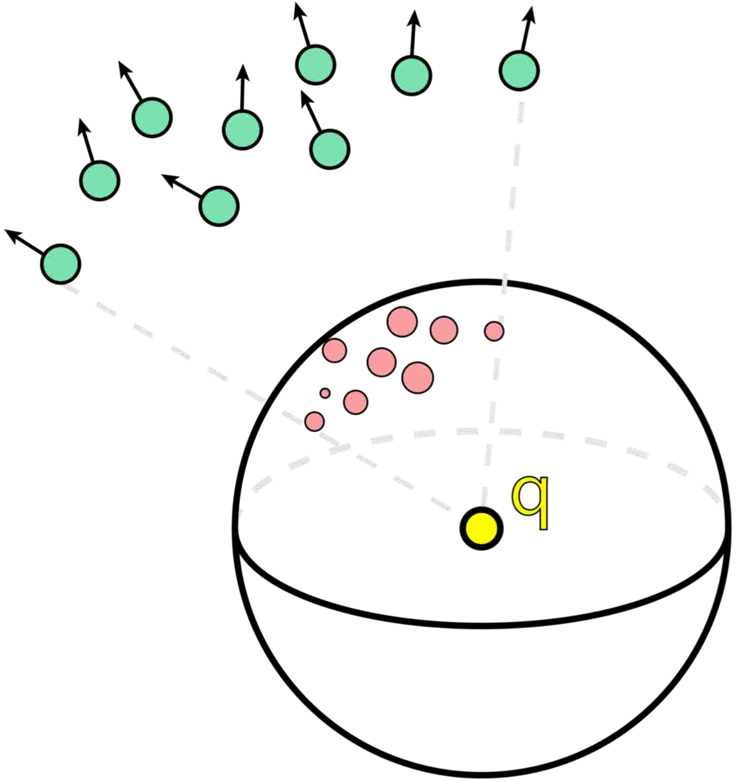
$$w_S(\mathbf{q}) = \frac{1}{4\pi}\int_S \frac{(\mathbf{x}-\mathbf{q})\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{q}\|^3}d\mathbf{x}$$

Winding number integrates dipoles on surface



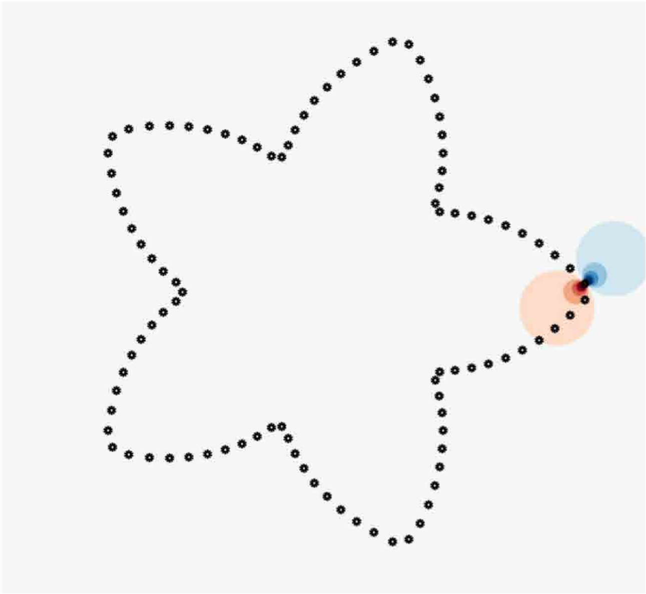
$$w_S(\mathbf{q}) = \frac{1}{4\pi} \int_S \frac{(\mathbf{x}-\mathbf{q})\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{q}\|^3} d\mathbf{x}$$

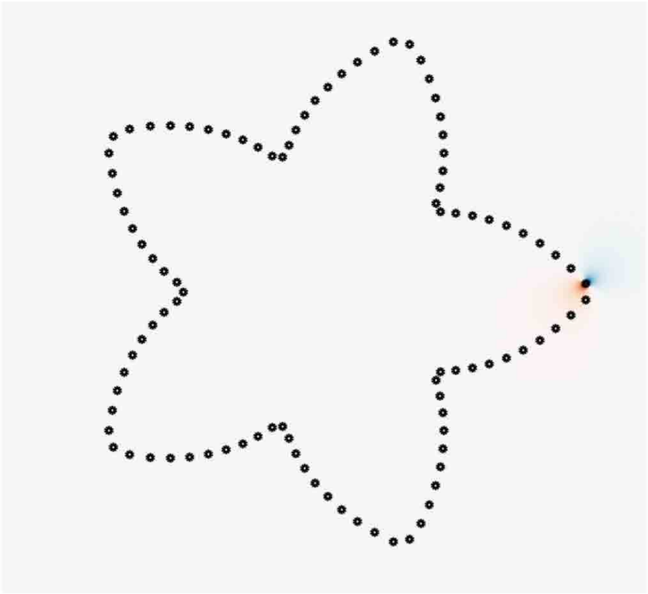
Winding number sums discrete *cloud* of dipoles



$$\sum_{i}^{n} \boxed{a_i} \frac{(\mathbf{x}_i - \mathbf{q}) \cdot \mathbf{n}_i}{\|\mathbf{x}_i - \mathbf{q}\|^3}$$

area associated with *i*-th point



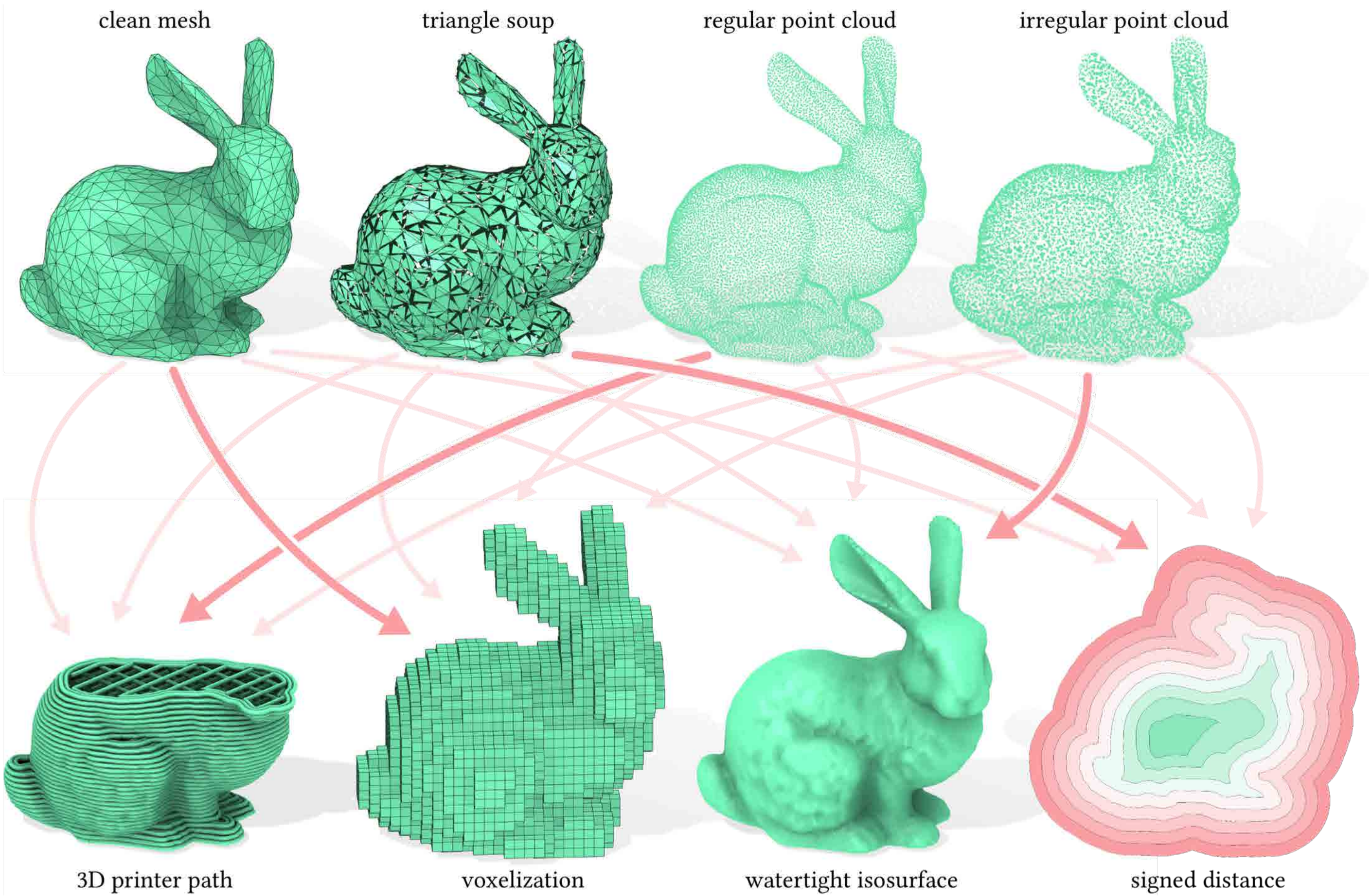


$$\Delta w = 0$$

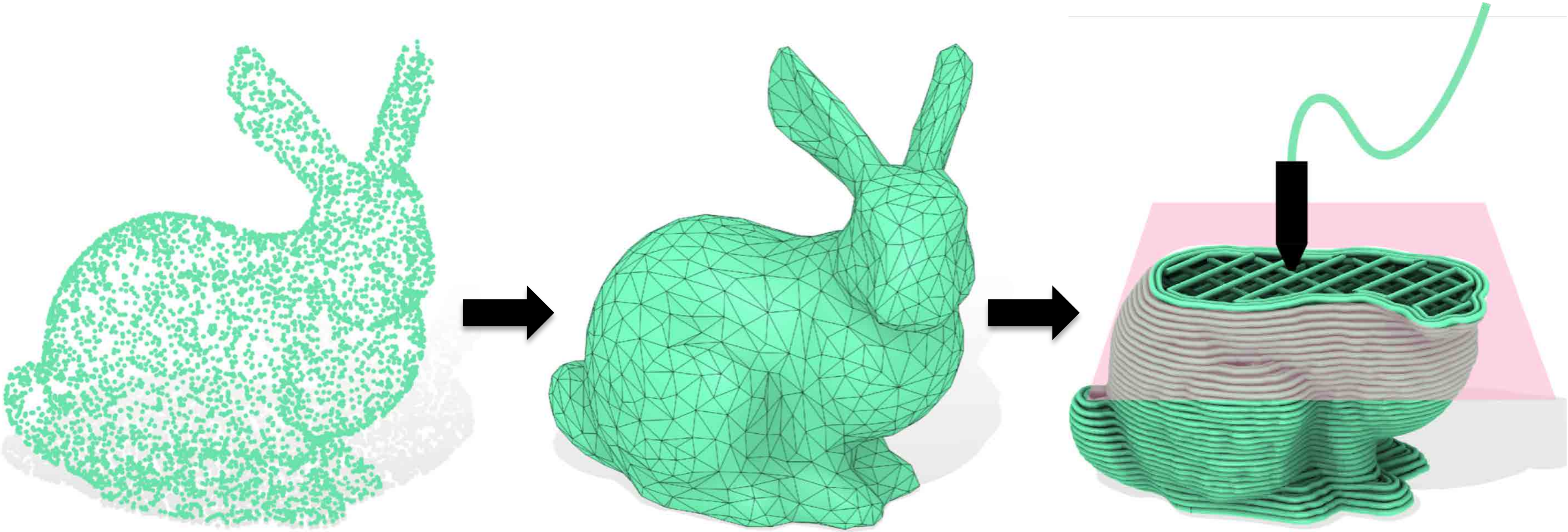
Winding number for point clouds!

$$w_S(\mathbf{q}) = \frac{1}{4\pi} \sum_{i}^{n} a_i \frac{(\mathbf{x}_i - \mathbf{q}) \cdot \mathbf{n}_i}{\|\mathbf{x}_i - \mathbf{q}\|^3}$$

Winding number further generalizes to point clouds

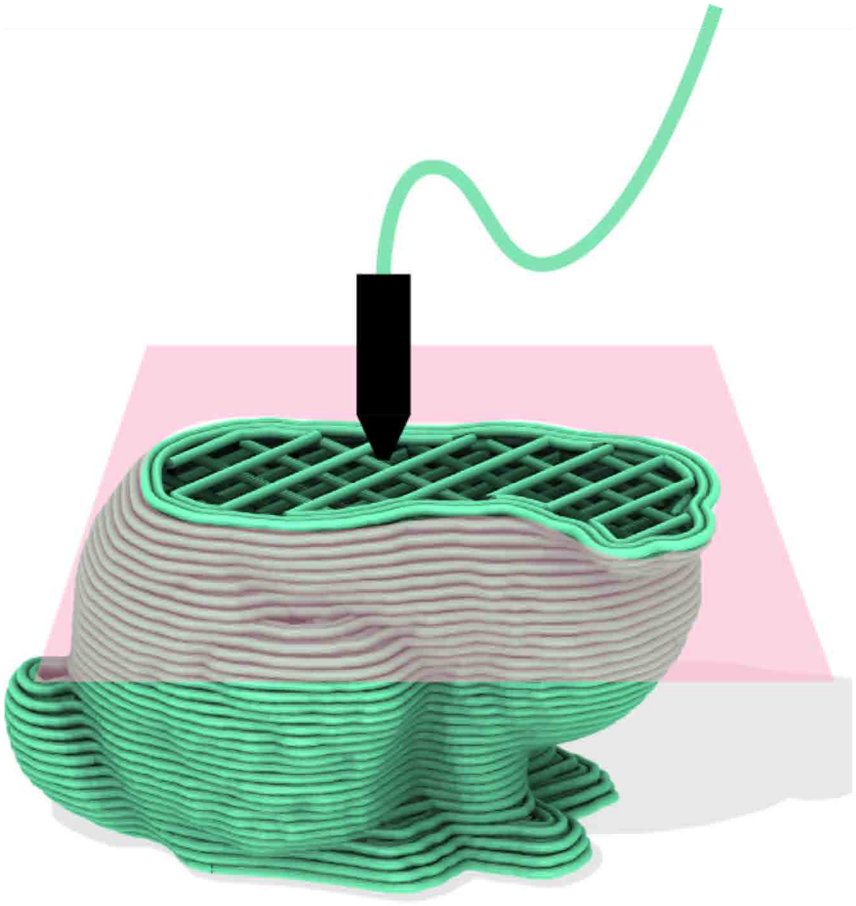


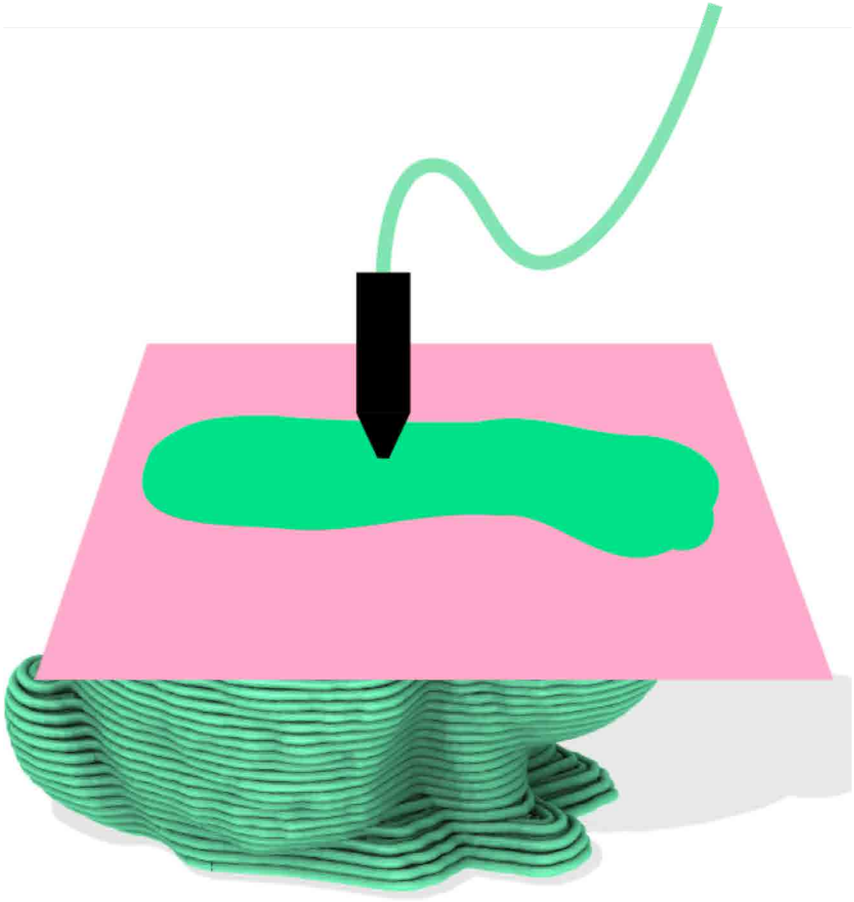
Typical 3D printing process requires `.stl` (triangle mesh) or 3D voxel image



Point Cloud — Surface Mesh (`.stl`) — Slice + Toolpath

"Fast Winding Numbers for Soups and Clouds" [Barill, Dickson, Schmidt, Levin, & J. 2018]

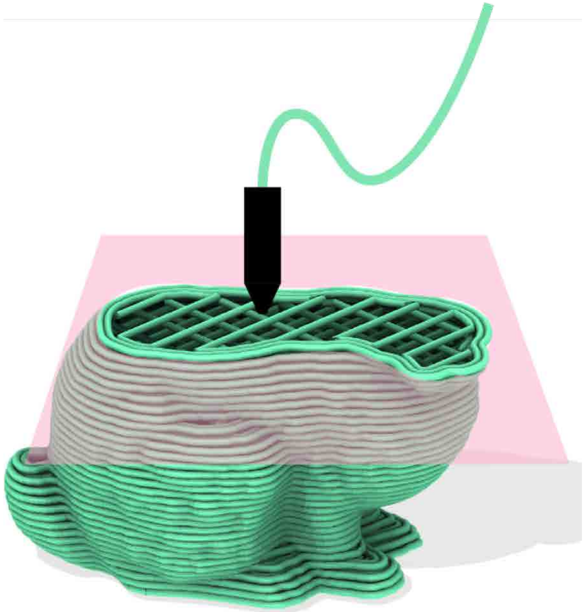




Winding number enables direct printing of point clouds

Point Cloud

Slice winding number

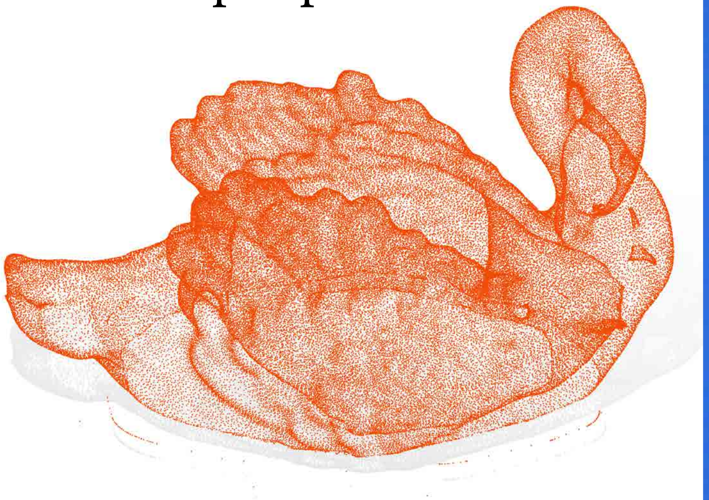


Toolpath

"Fast Winding Numbers for Soups and Clouds" [Barill, Dickson, Schmidt, Levin, & J. 2018]

Winding number enables direct printing of point clouds

input point cloud



“Fast Winding Numbers for Soups and Clouds” [Barill, Dickson, Schmidt, Levin, & J. 2018]

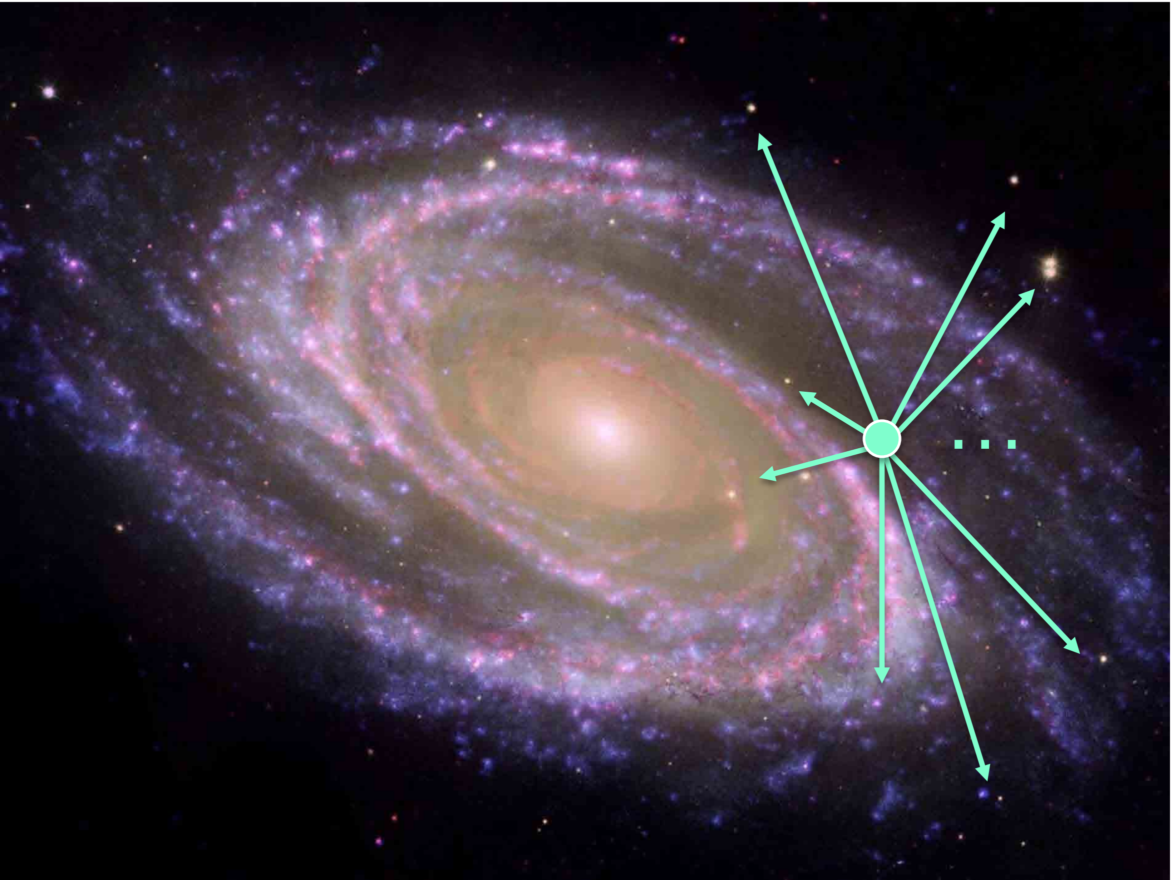
Winding number for point clouds!

$$w_S(\mathbf{q}) = \frac{1}{4\pi}\sum_{i}^{n} a_i \frac{(\mathbf{x}_i - \mathbf{q})\cdot \mathbf{n}_i}{\|\mathbf{x}_i - \mathbf{q}\|^3}$$

Evaluating winding number for
m queries over n points is still O(mn) 😢

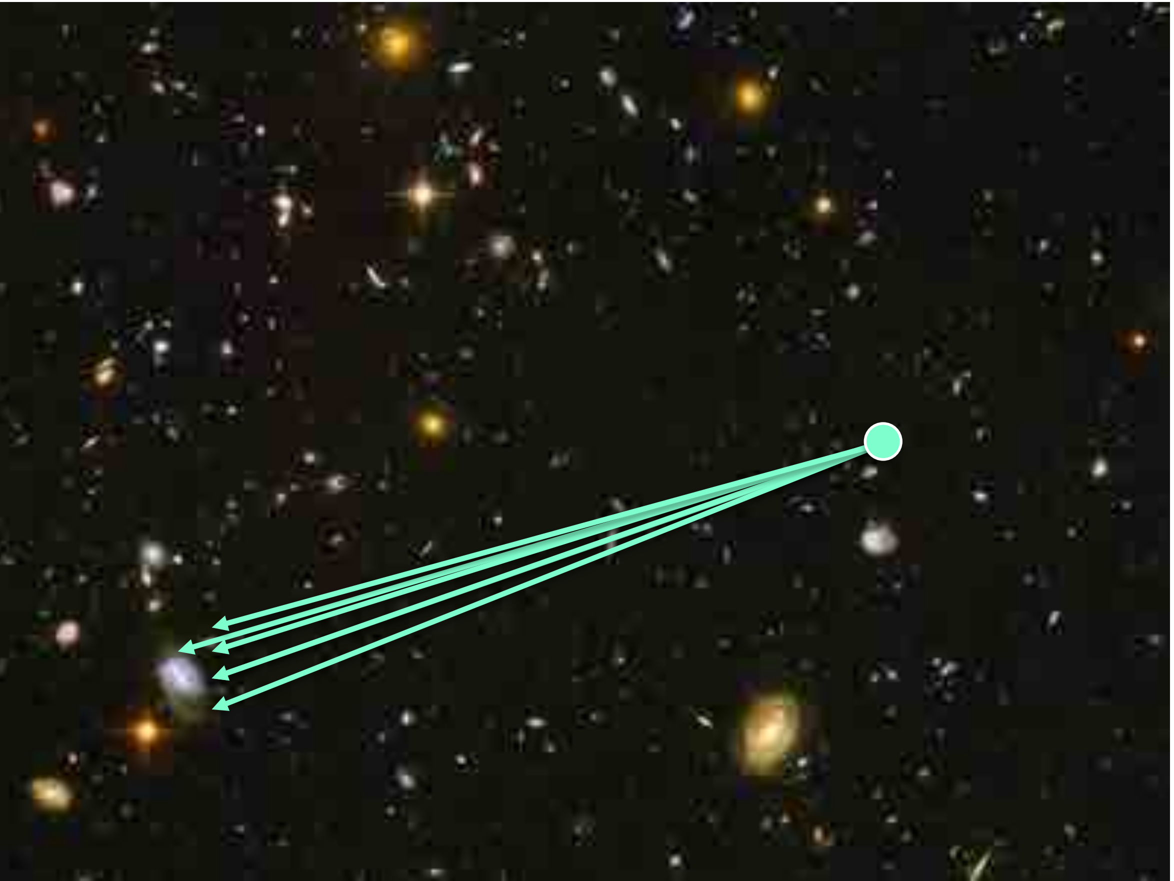
Borrow insight from astrophysics:

Borrow insight from astrophysics:



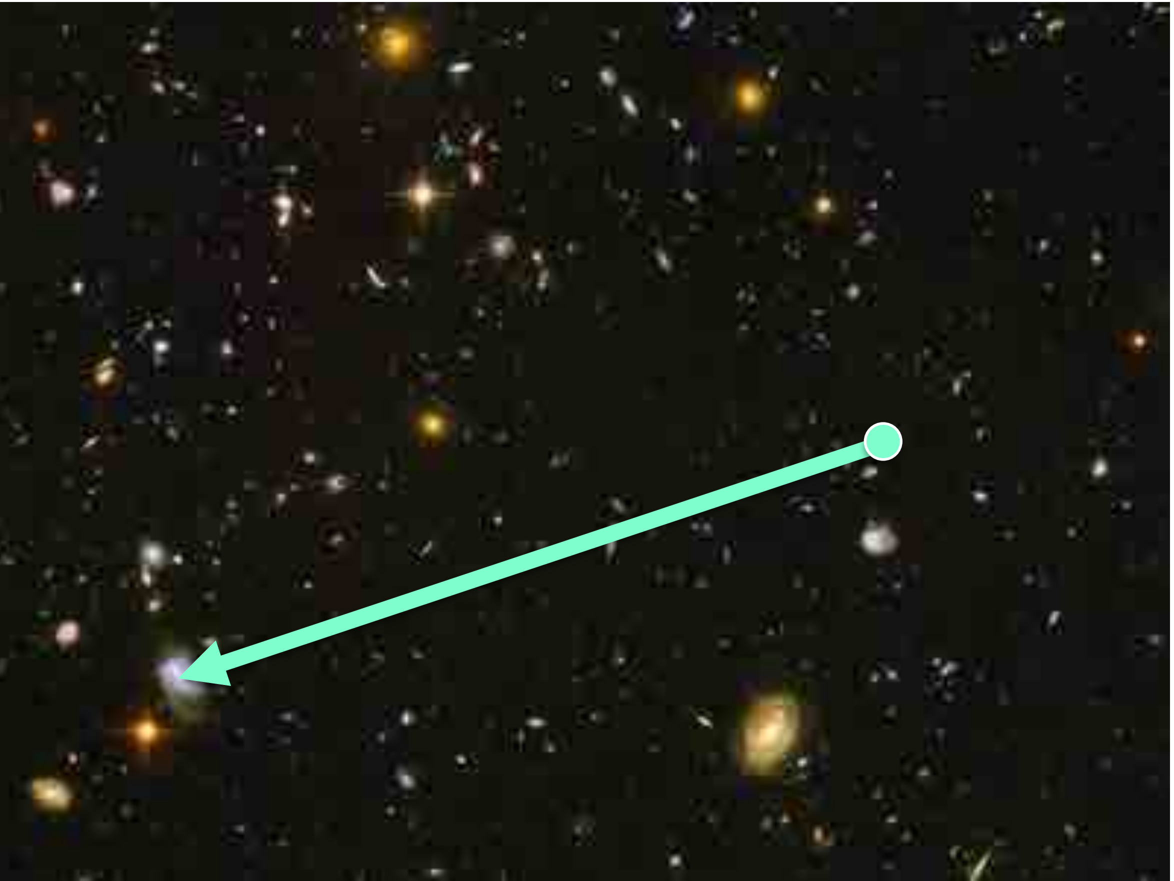
Gravitational forces of nearby bodies may be interesting…

Borrow insight from astrophysics:



... but force of many far away objects is well approximated by a single (big) object

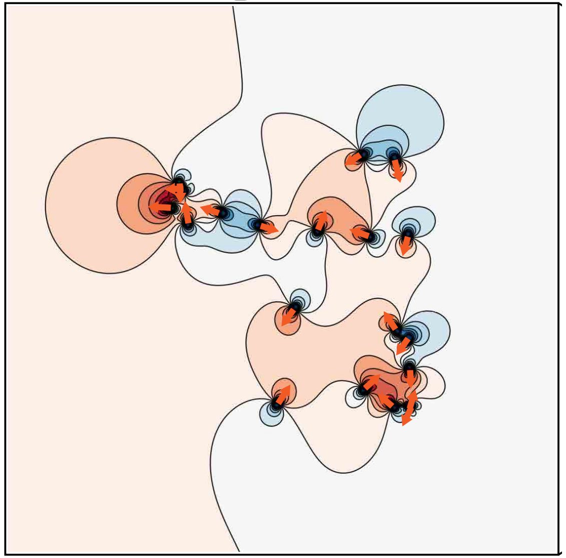
Borrow insight from astrophysics:



... but force of many far away objects is well approximated by a single (big) object

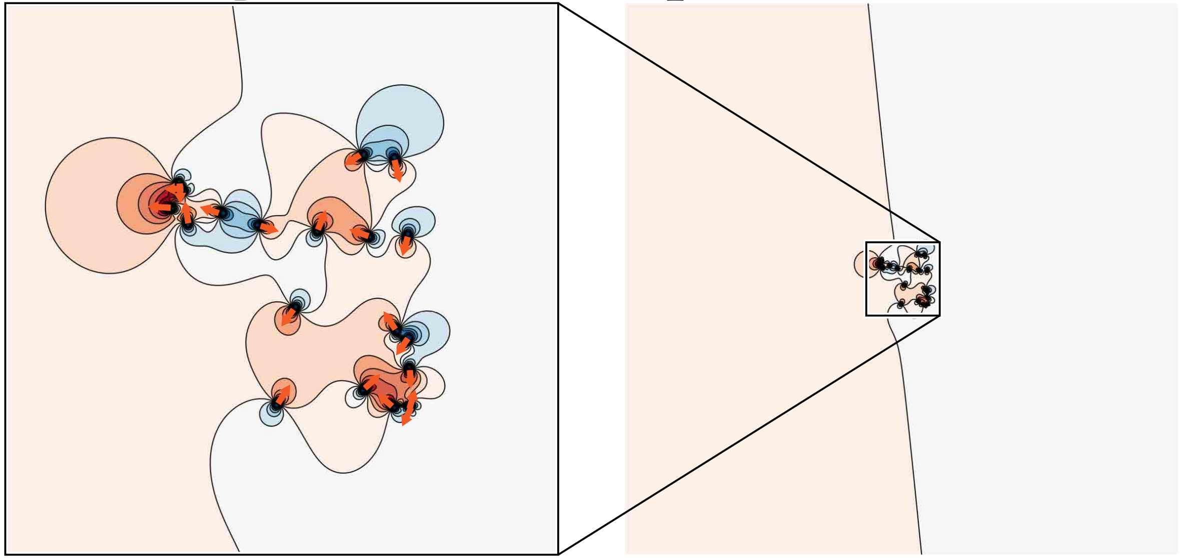
Winding number of many far away points looks just like that of single point

Winding number of 20 points



Winding number of many far away points looks just like that of single point

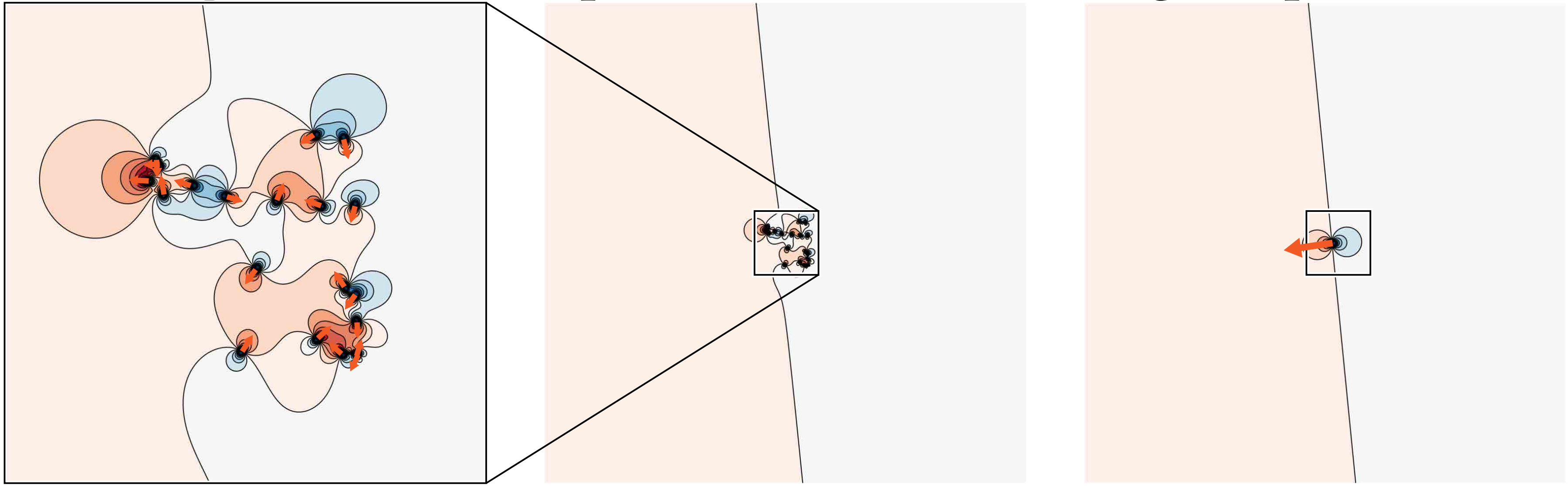
Winding number of 20 points



Winding number of many far away points looks just like that of single point

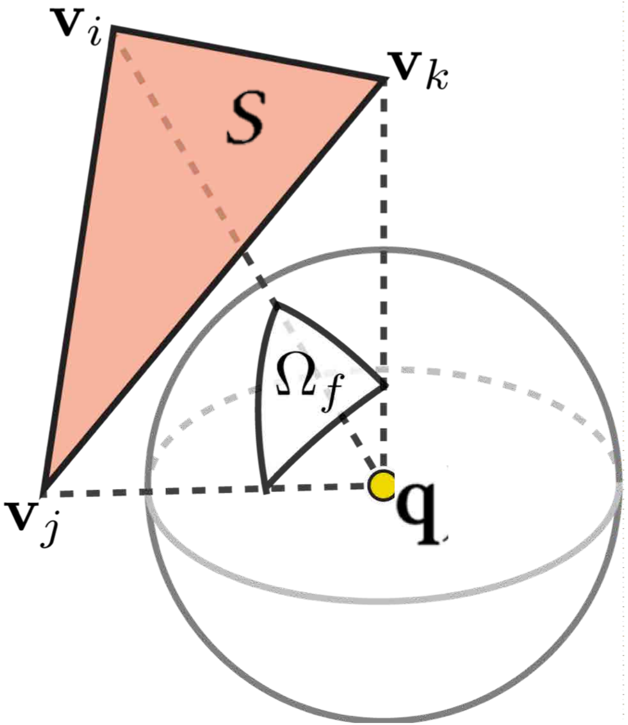
Winding number of 20 points

Single representative



Precomputation:
Throw points into bounding volume hierarchy (e.g., octree)
Query evaluation:
Use cell representative if far enough, otherwise recursive call on children

Immediately extends to triangles



$$w_S(\mathbf{q}) = \int_S \frac{(\mathbf{x}-\mathbf{q})\cdot\hat{\mathbf{n}}}{4\pi\|\mathbf{x}-\mathbf{q}\|^3}\, dx$$

Just integrate point winding number over triangle…

... "representative point" of a bunch of triangles is :

q: area-weighted barycenters
n: area-weighted normals
a: total area

Higher-order accuracy is possible

$$
\begin{aligned}
w(\mathbf{q}) \approx\ & \left(\sum_{t=1}^{m} \int_t \hat{\mathbf{n}}_t \; dA\right) \cdot \nabla G(\mathbf{q}, \tilde{\mathbf{p}}) && (18)\\
&+ \left(\sum_{t=1}^{m} \int_t (\mathbf{x} - \tilde{\mathbf{p}}) \otimes \hat{\mathbf{n}}_t \; dA\right) \cdot \nabla^2 G(\mathbf{q}, \tilde{\mathbf{p}}) && (19)\\
&+ \frac{1}{2}\left(\sum_{t=1}^{m} \int_t (\mathbf{x} - \tilde{\mathbf{p}}) \otimes ((\mathbf{x} - \tilde{\mathbf{p}}) \otimes \hat{\mathbf{n}}_t)\; dA\right) \cdot \nabla^3 G(\mathbf{q}, \tilde{\mathbf{p}}) && (20)\\
&+ \frac{1}{2}\left(\sum_{t=1}^{m} \int_t (\mathbf{x} - \tilde{\mathbf{p}}) \otimes (\mathbf{x} - \tilde{\mathbf{p}}) \otimes \hat{\mathbf{n}}_t)\; dA\right) \cdot \nabla^3 G(\mathbf{q}, \tilde{\mathbf{p}}) && (21)\\
&+ \ \textit{higher order terms} =: w(\mathbf{q}). && (22)
\end{aligned}
$$

Higher-order accuracy is possible

"Single representative" actually corresponds to first-order Taylor expansion…

We can take any order, see paper for up to 3rd order derivations.

Higher-order accuracy is possible

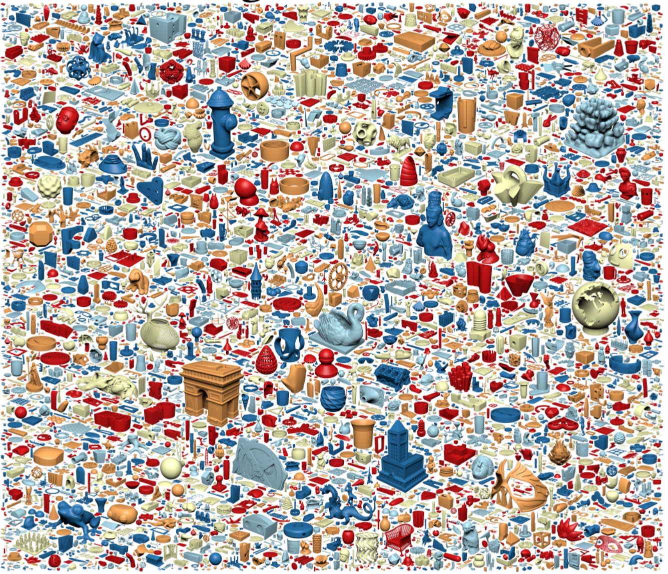
"Single representative" actually corresponds to first-order Taylor expansion…

We can take any order, see paper for up to 3rd order derivations.

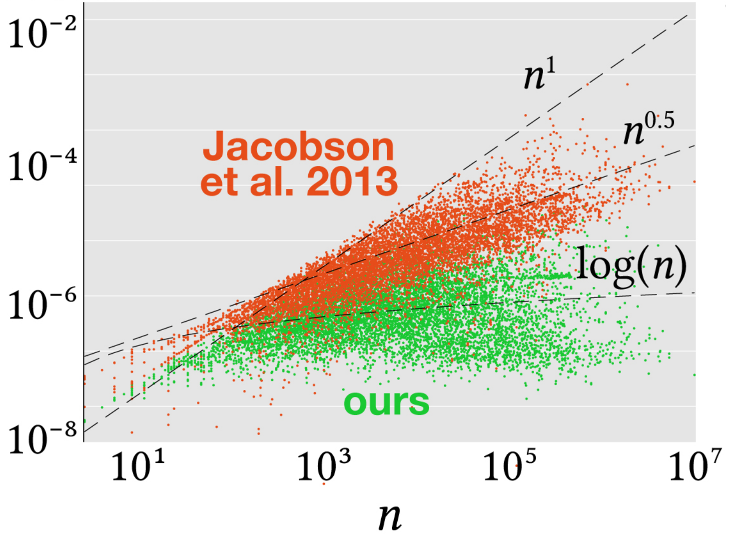
Fast Winding Number for Soups and Clouds
Gavin Barill, Neil Dickson, Ryan Schmidt, David I.W. Levin, Alec Jacobson
ACM SIGGRAPH 2018

This time, really asymptotically faster

Thingi10k dataset

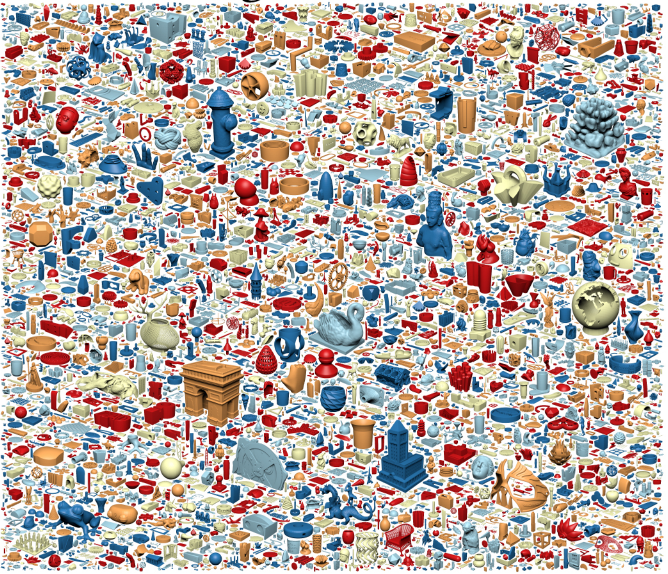


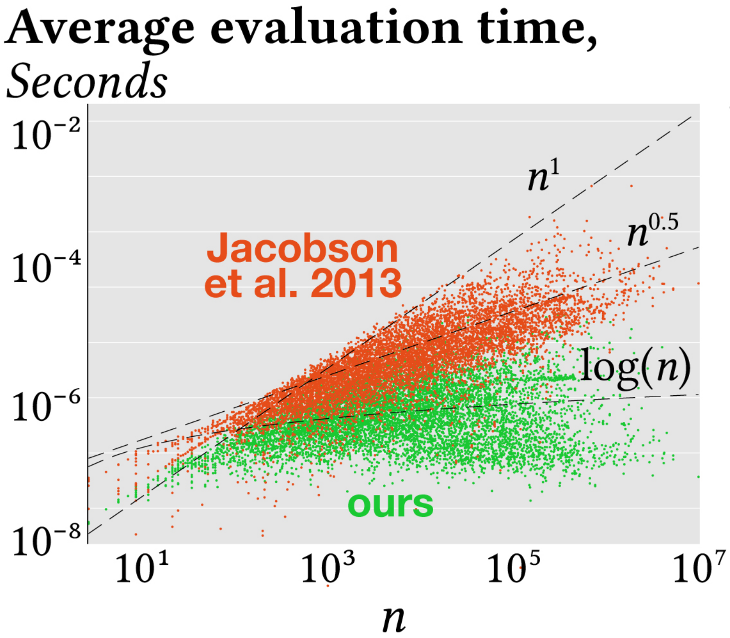
Average evaluation time,
Seconds

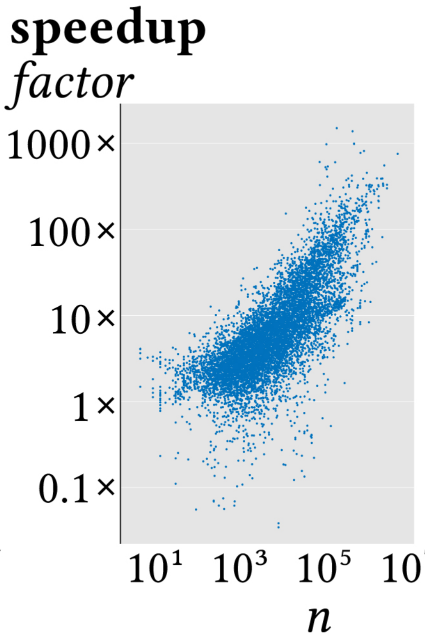


This time, really asymptotically faster

Thingi10k dataset

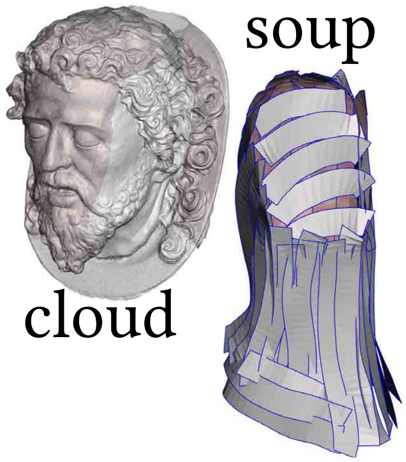






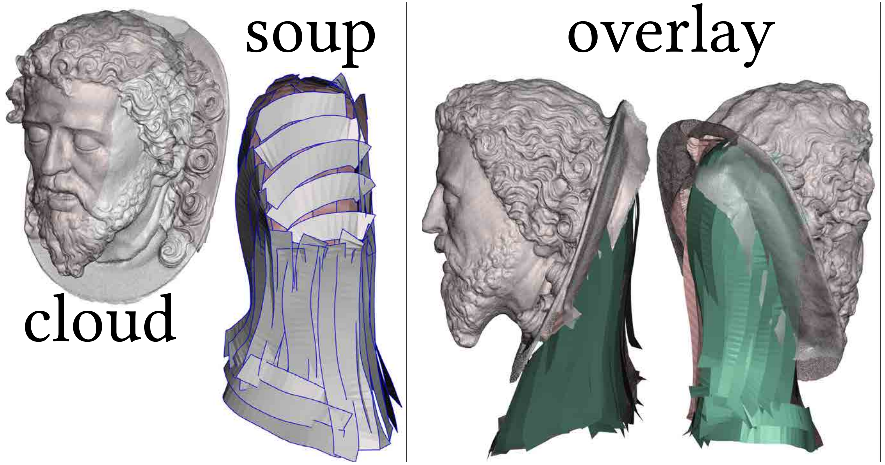
VR painters don't need/want to care about underlying data structures

VR painters don't need/want to care about underlying data structures



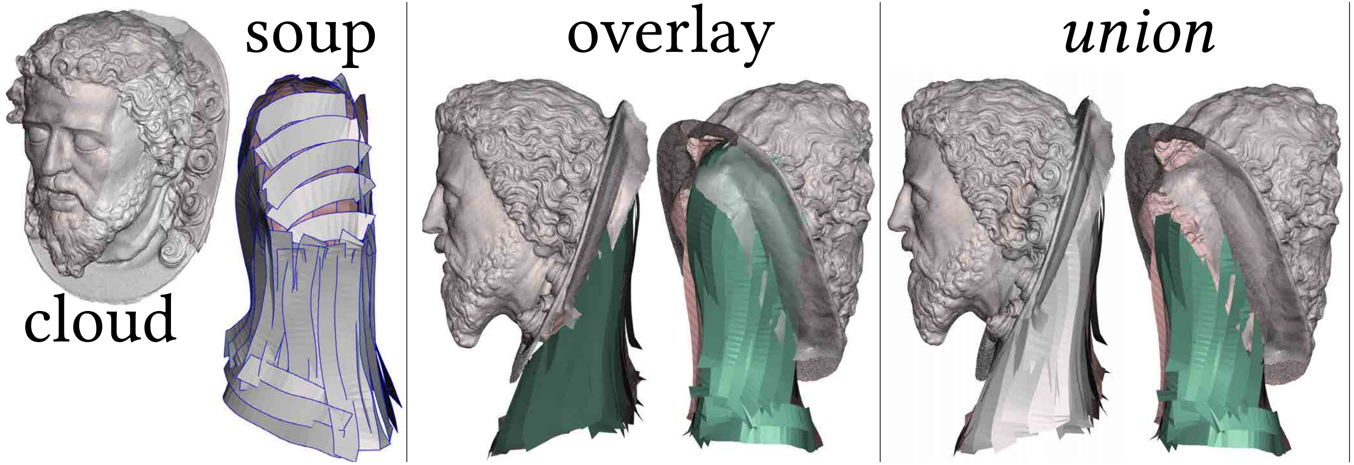
"Fast Winding Numbers for Soups and Clouds" [Barill, Dickson, Schmidt, Levin, & J. 2018]

VR painters don't need/want to care about underlying data structures



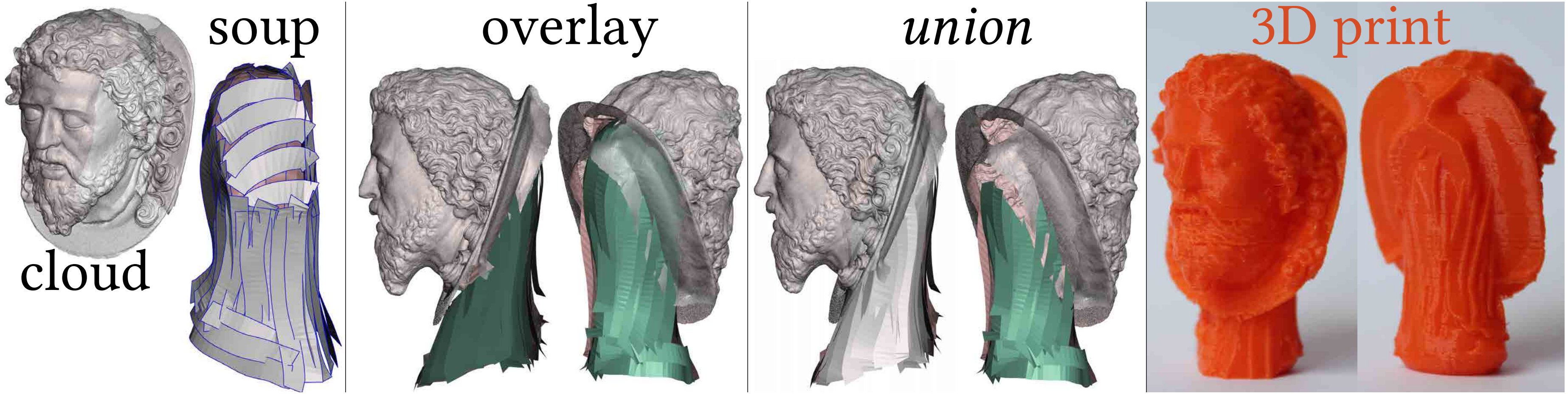
“Fast Winding Numbers for Soups and Clouds” [Barill, Dickson, Schmidt, Levin, & J. 2018]

VR painters don’t need/want to care about underlying data structures



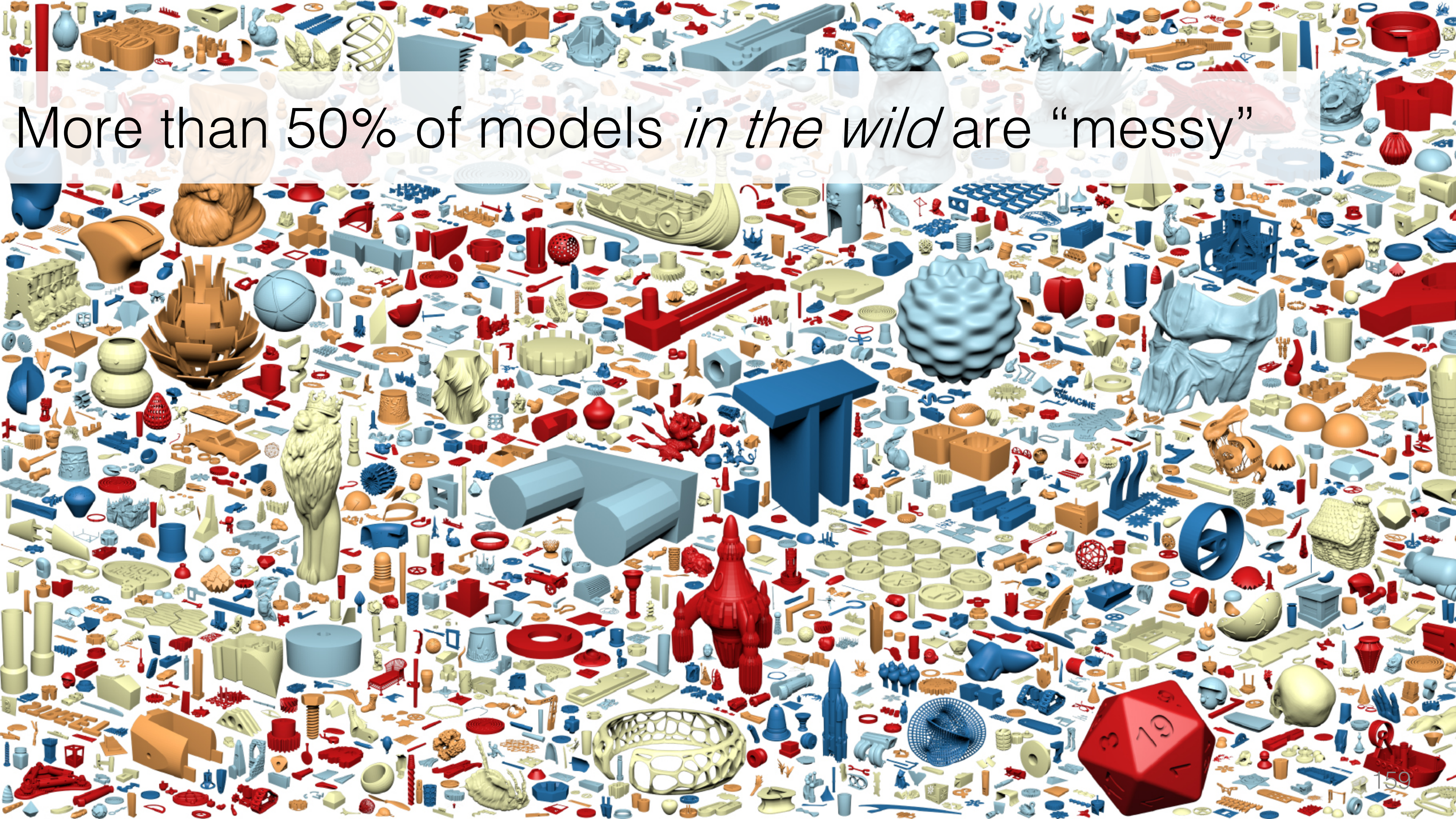
“Fast Winding Numbers for Soups and Clouds” [Barill, Dickson, Schmidt, Levin, & J. 2018]

VR painters don’t need/want to care about underlying data structures

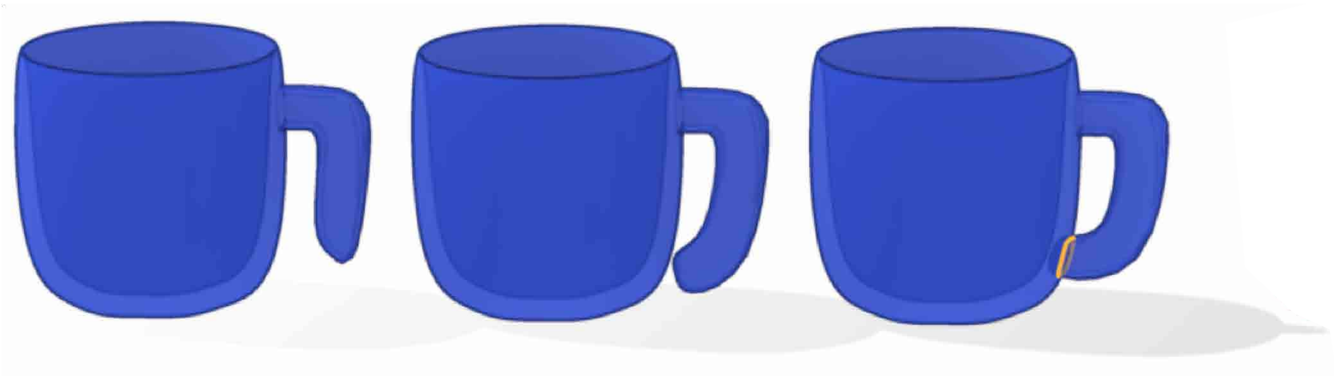


“Fast Winding Numbers for Soups and Clouds” [Barill, Dickson, Schmidt, Levin, & J. 2018]

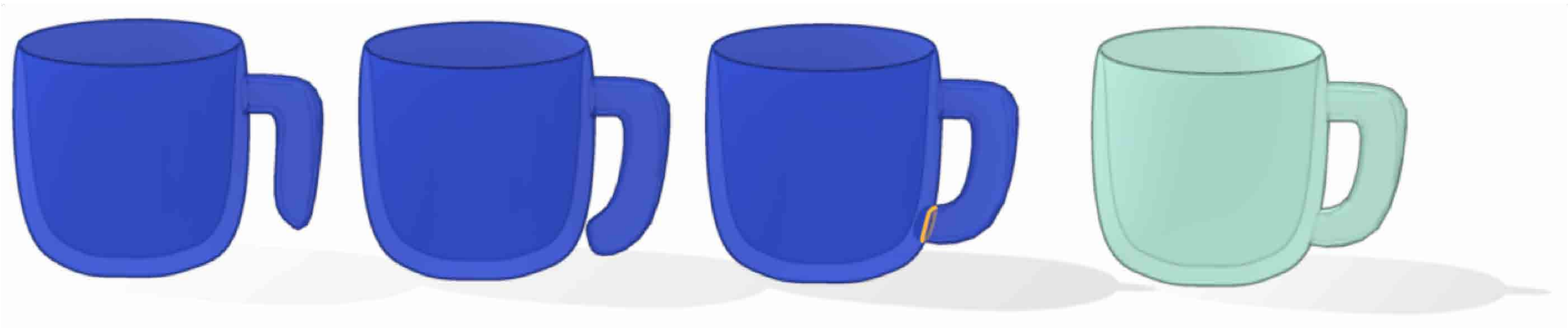
More than 50% of models *in the wild* are “messy”



“Artifacts” or “defects” are actually symptoms of friendly UIs



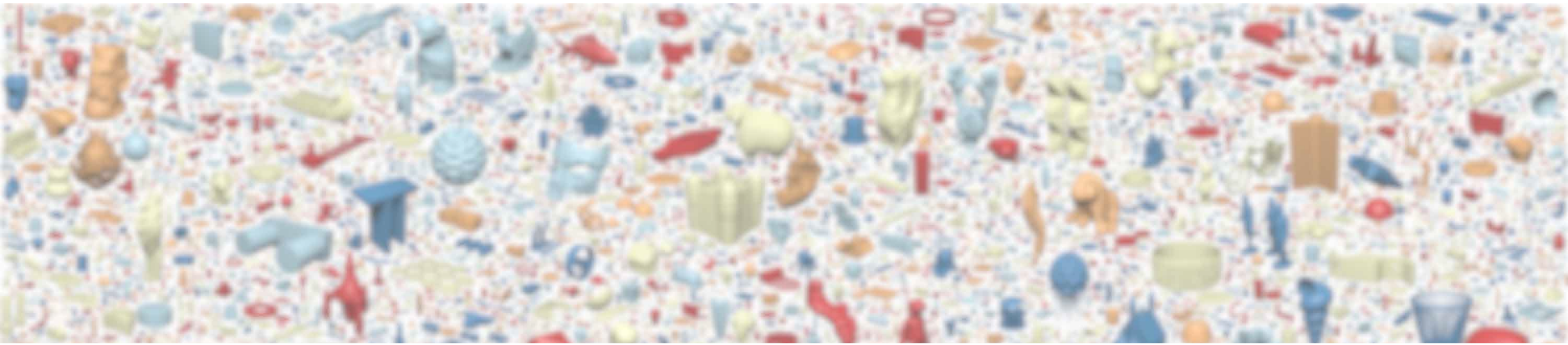
“Artifacts” or “defects” are actually symptoms of friendly UIs



Generalized winding number, as a *concept*, helps classify shapes

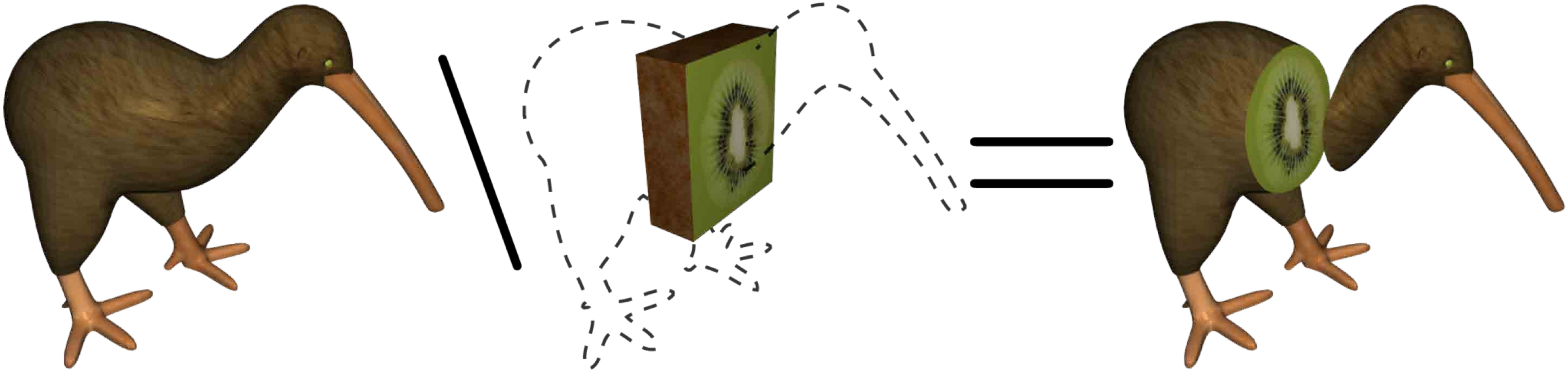
49% "clean" by standard metrics

86% piecewise-constant winding number



"Thingi10K: A Dataset of 10,000 3D-Printing Models" [Zhou & J. 2016]

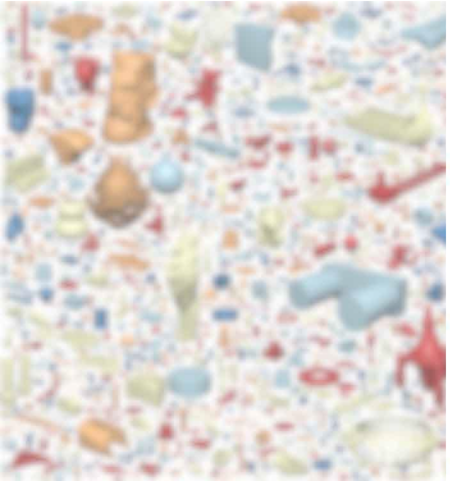
Novel Boolean algorithm accepts all piecewise-constant winding number meshes

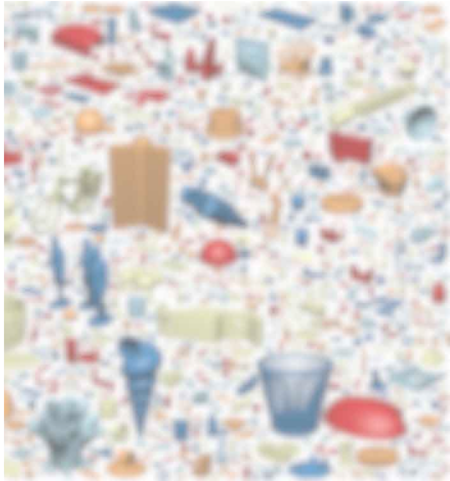


"Mesh Arrangements for Solid Geometry" [Zhou, Grinspun, Zorin, & J. 2016]

Sagrada Familia, Gaudí (1852-1926)

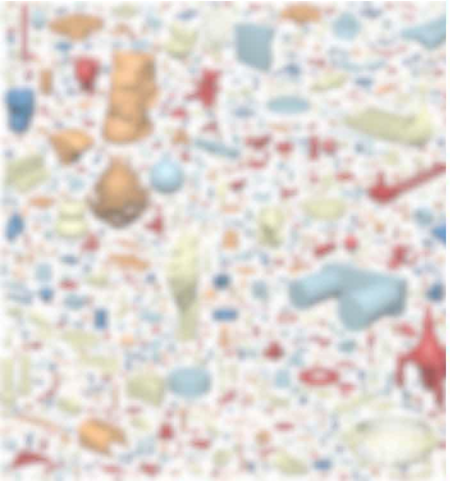
Careful preconditions, postconditions ensure robustness, validated empirically

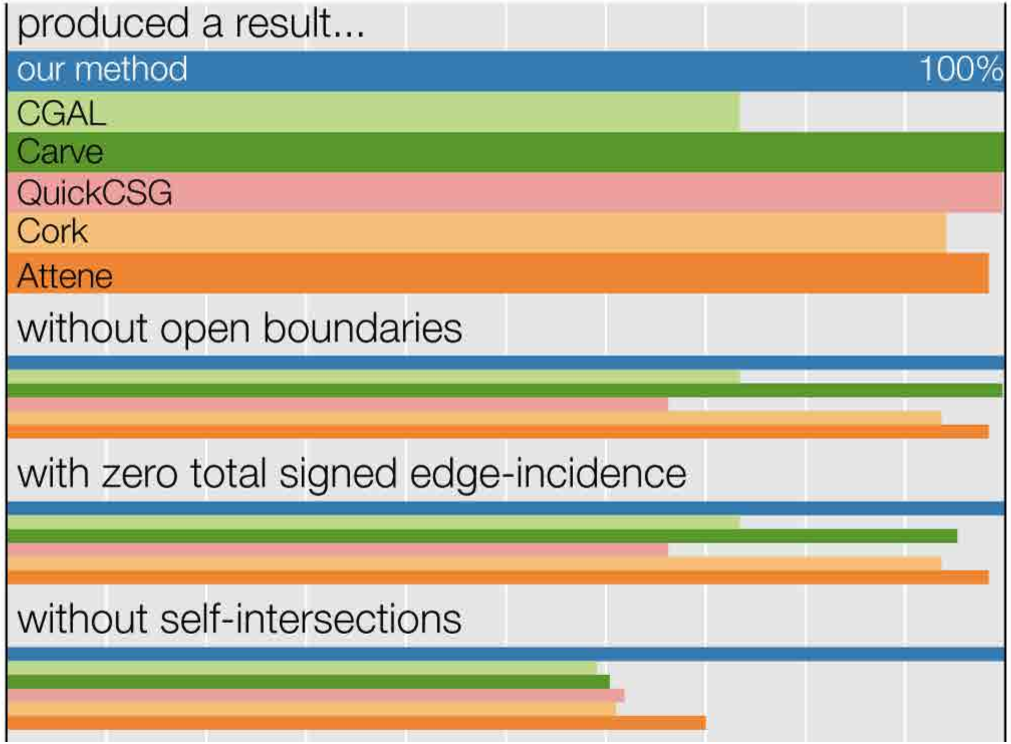


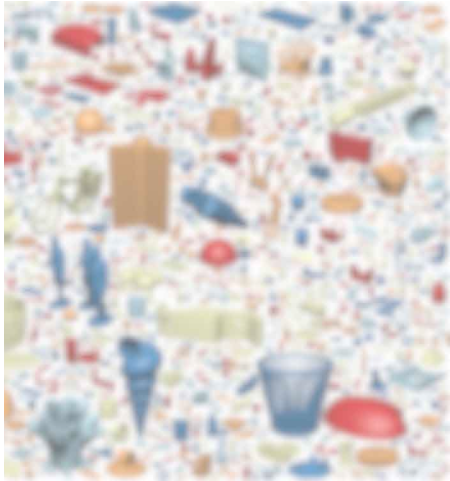


"Mesh Arrangements for Solid Geometry" [Zhou, Grinspun, Zorin, & J. 2016]

Careful preconditions, postconditions ensure robustness, validated empirically

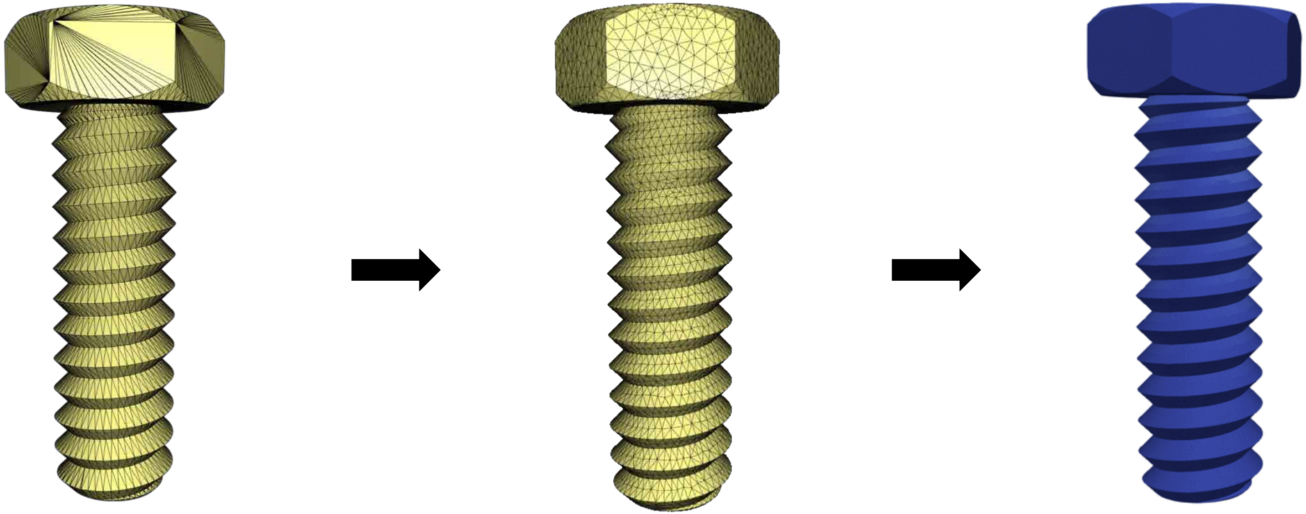






"Mesh Arrangements for Solid Geometry" [Zhou, Grinspun, Zorin, & J. 2016]

Simulation depends on tetrahedral meshing



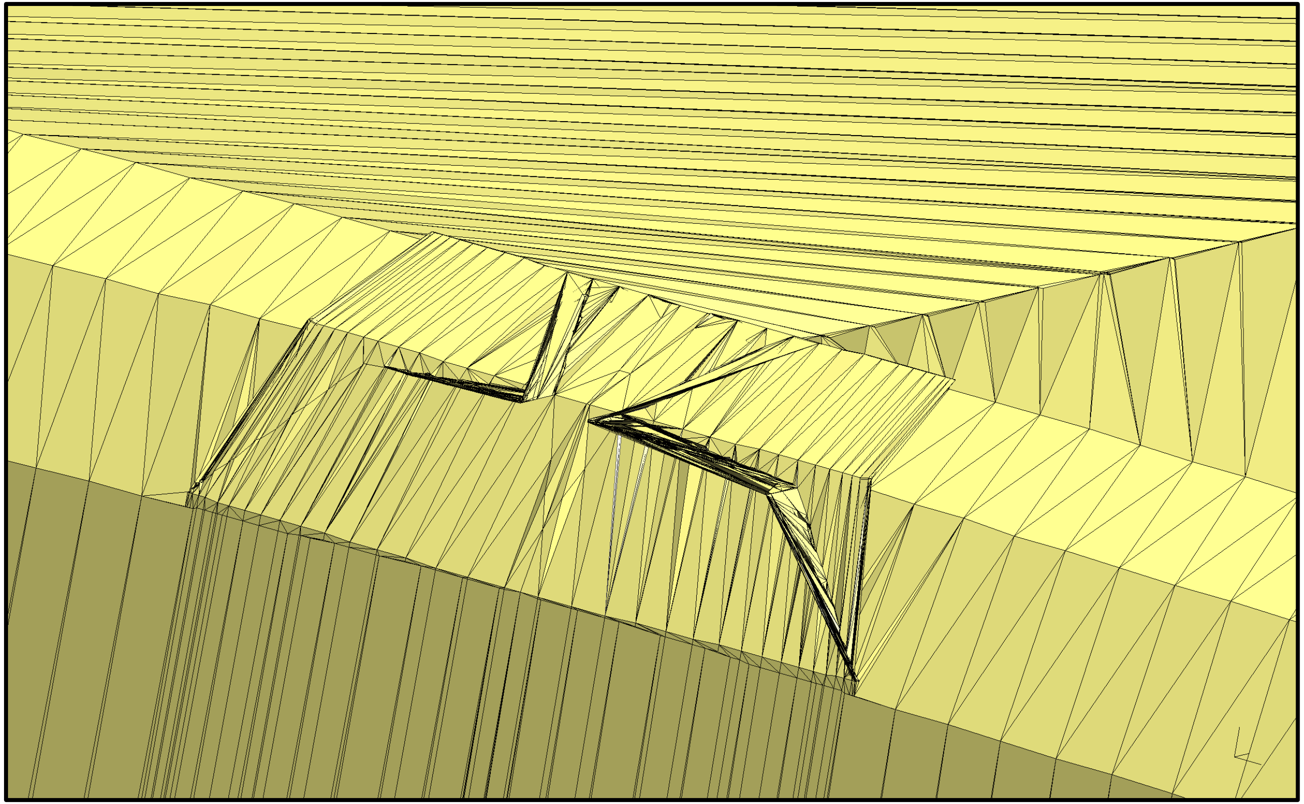
Surface Representation
(From Thingi10k)

Volumetric Representation
(Generated by TetWild)

Physical Simulation

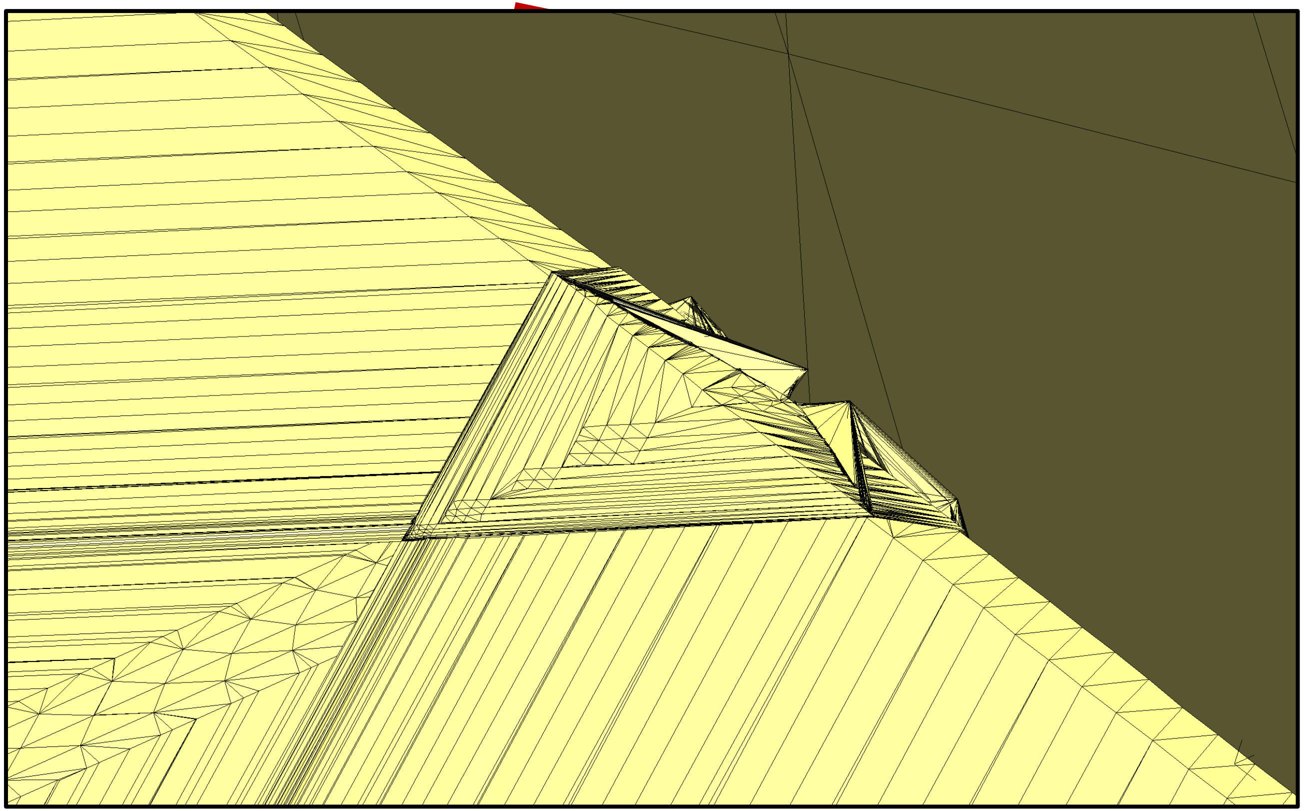
"Tetrahedral Meshing in the Wild" [Hu, Zhou, Gao, J., Zorin, Panozzo 2018]

Why is tetrahedral meshing hard?



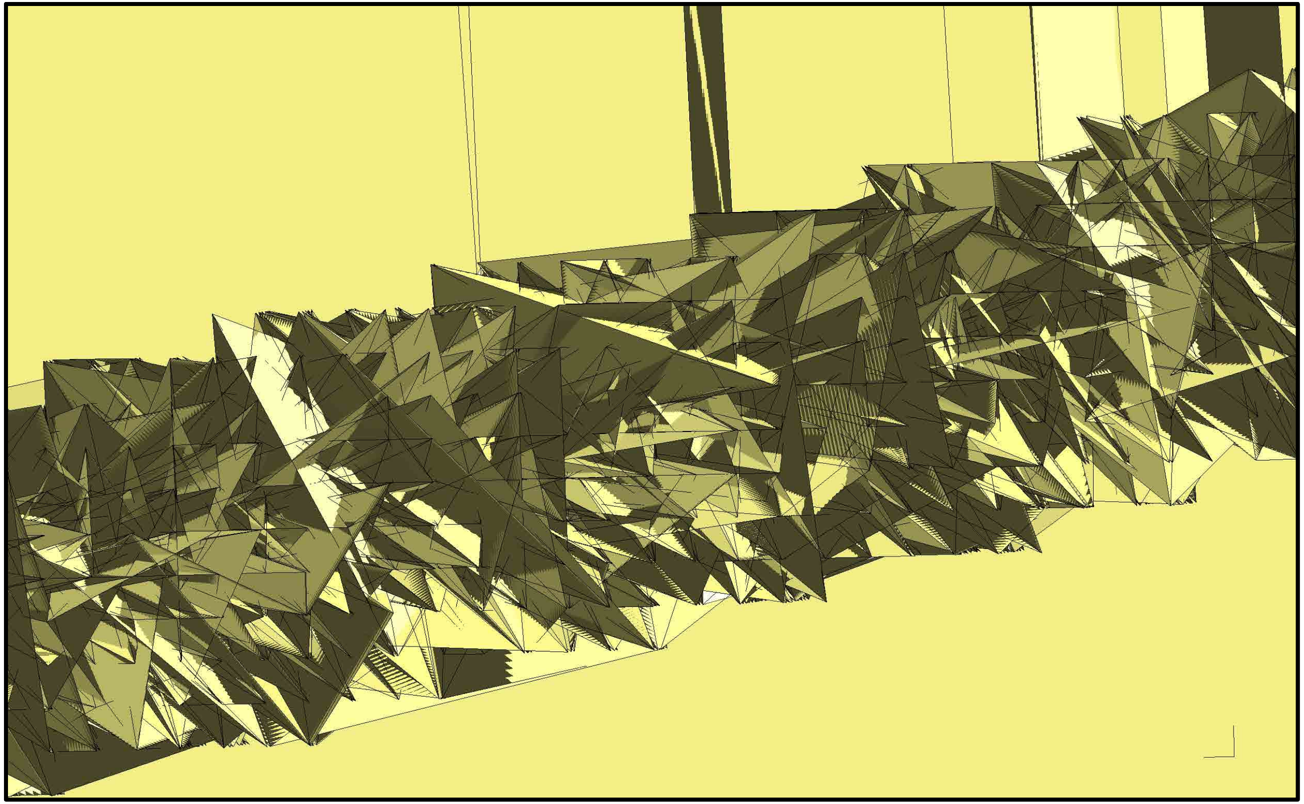
“Tetrahedral Meshing in the Wild” [Hu, Zhou, Gao, J., Zorin, Panozzo 2018]

Why is tetrahedral meshing hard?



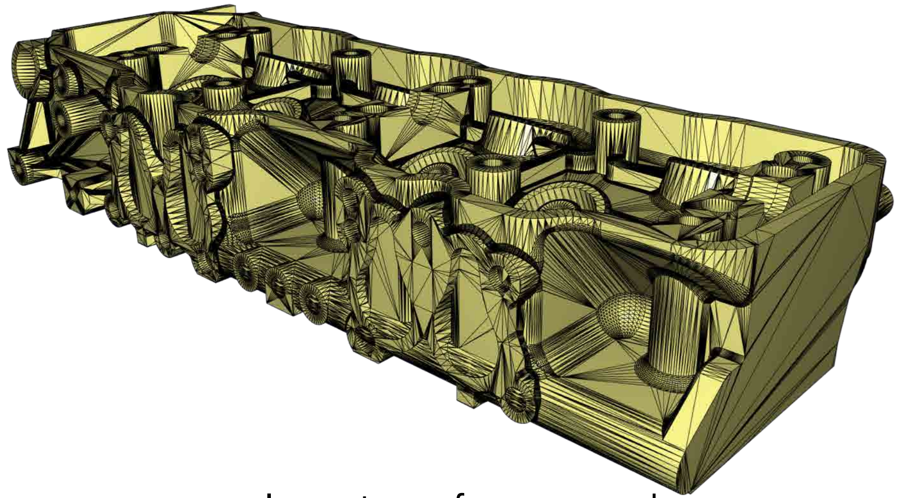
“Tetrahedral Meshing in the Wild” [Hu, Zhou, Gao, J., Zorin, Panozzo 2018]

Why is it a Hard Problem?



"Tetrahedral Meshing in the Wild" [Hu, Zhou, Gao, J., Zorin, Panozzo 2018]

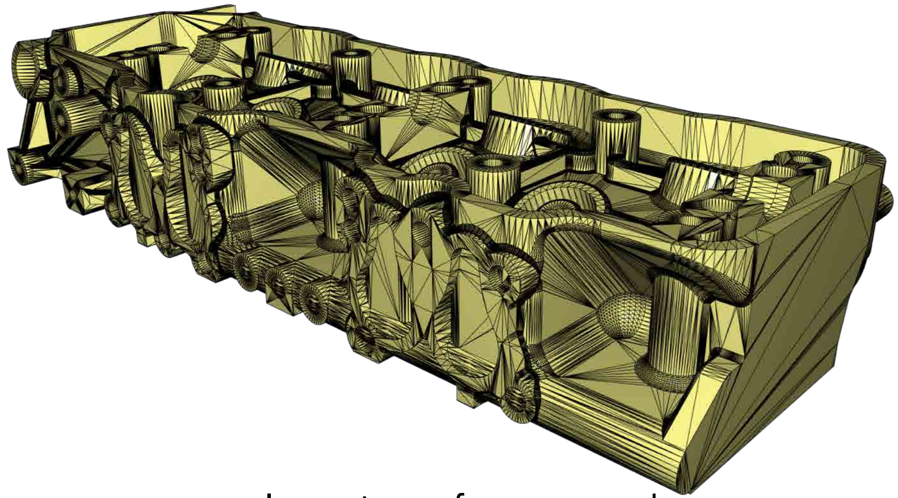
Winding number and small *epsilon* protect against nasty faceted CAD models



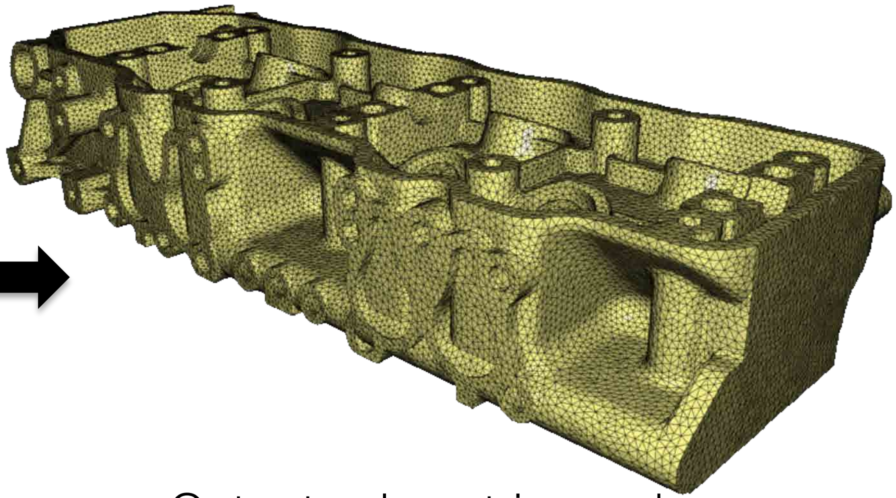
Input surface mesh

"Tetrahedral Meshing in the Wild" [Hu, Zhou, Gao, J., Zorin, Panozzo 2018]

Winding number and small *epsilon* protect against nasty faceted CAD models



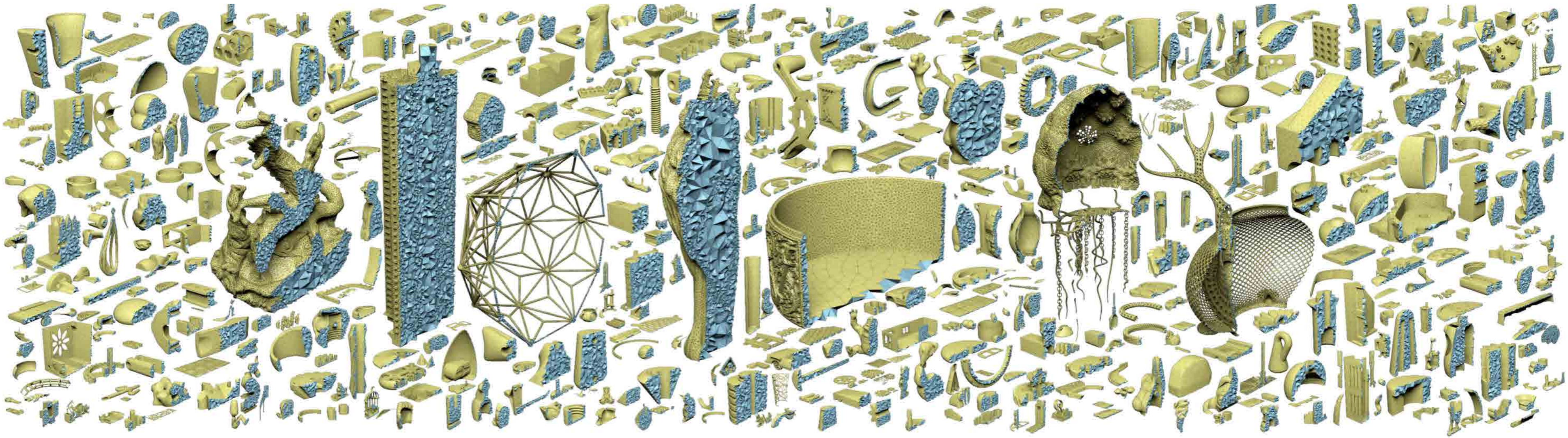
Input surface mesh



Output volumetric mesh

"Tetrahedral Meshing in the Wild" [Hu, Zhou, Gao, J., Zorin, Panozzo 2018]

Validate in the wild on 10,000 models

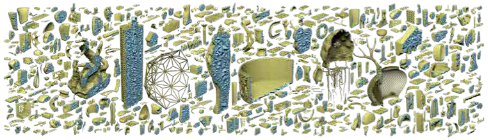


mesh entire convex hull
conform to surface up to small *epsilon*
extract interior via generalized winding number

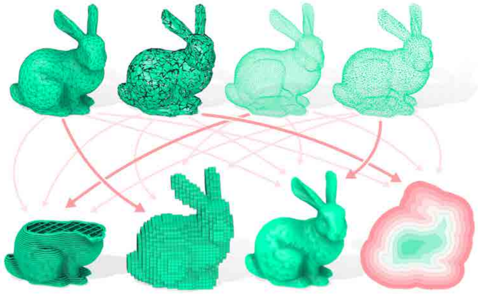
"Tetrahedral Meshing in the Wild" [Hu, Zhou, Gao, J., Zorin, Panozzo 2018]

We're dedicated to open software…

libigl
c++ geometry processing library
https://github.com/libigl/libigl

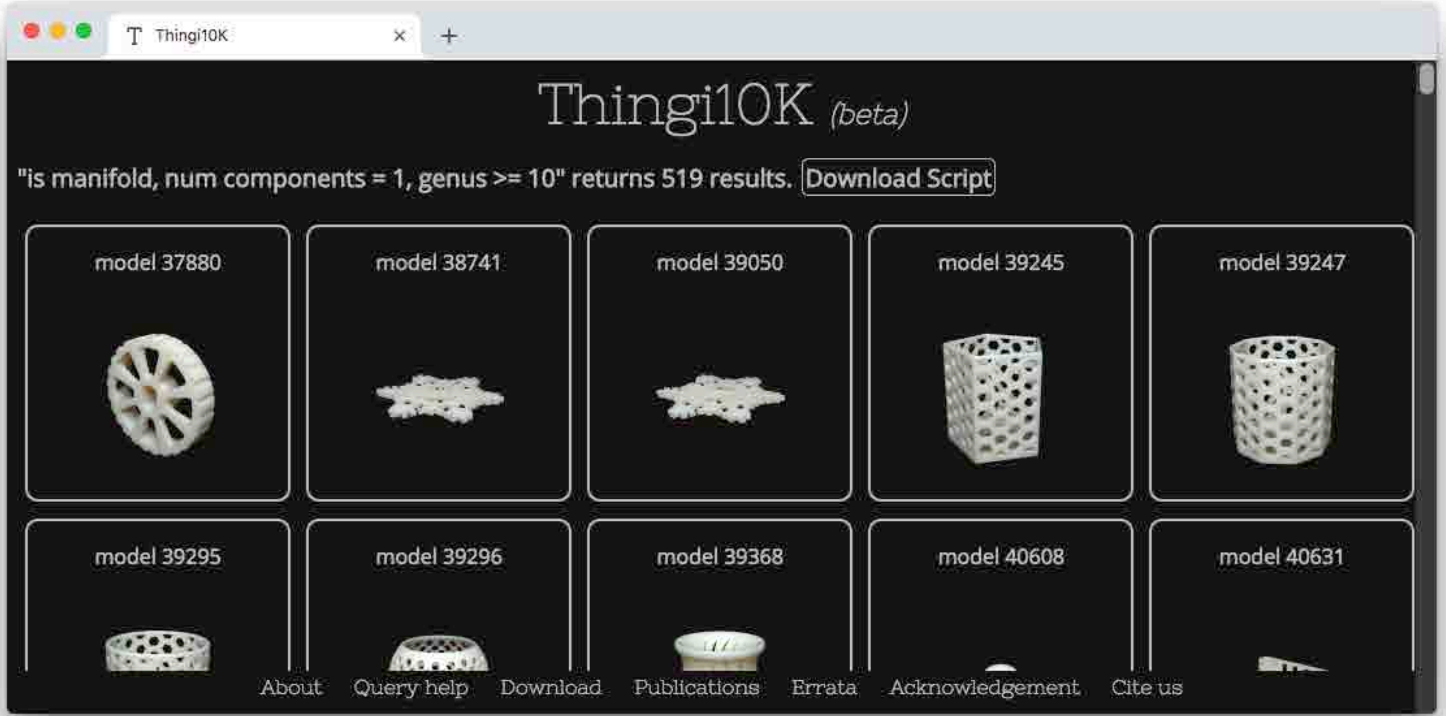


tetwild
c++ tetrahedral meshing
https://github.com/Yixin-Hu/TetWild



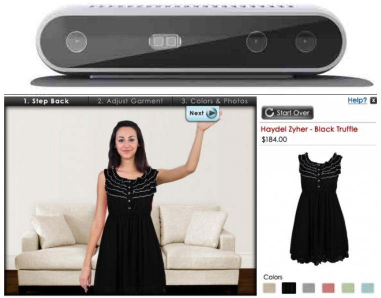
fast winding number
c++ robust inside/outside testing, voxelization
https://github.com/GavinBarill/fast-winding-number-soups

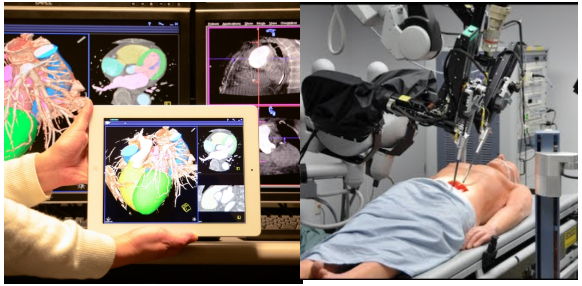
…and open data

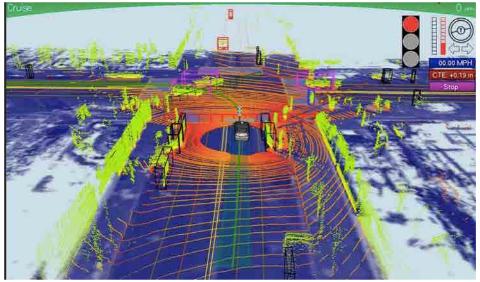


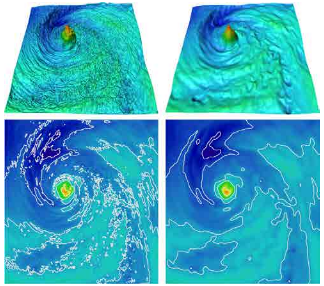
thingi10k
http://ten-thousand-models.appspot.com

Possibilities are everywhere



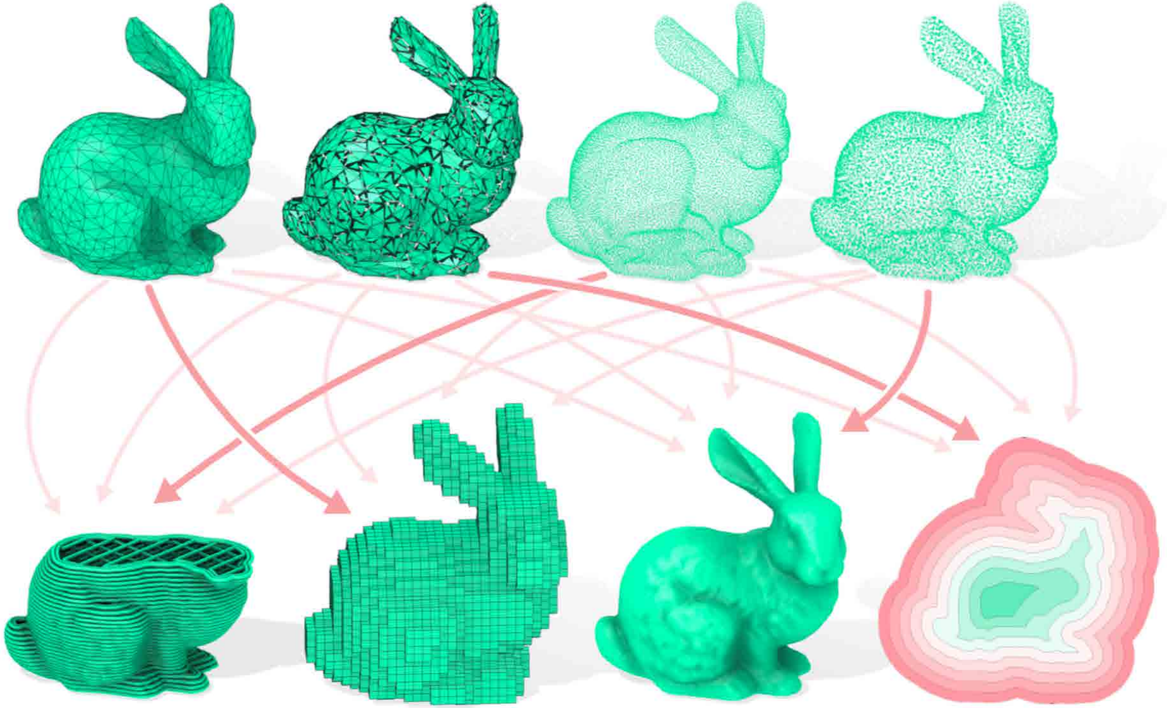






Concluding thoughts

Question common assumptions

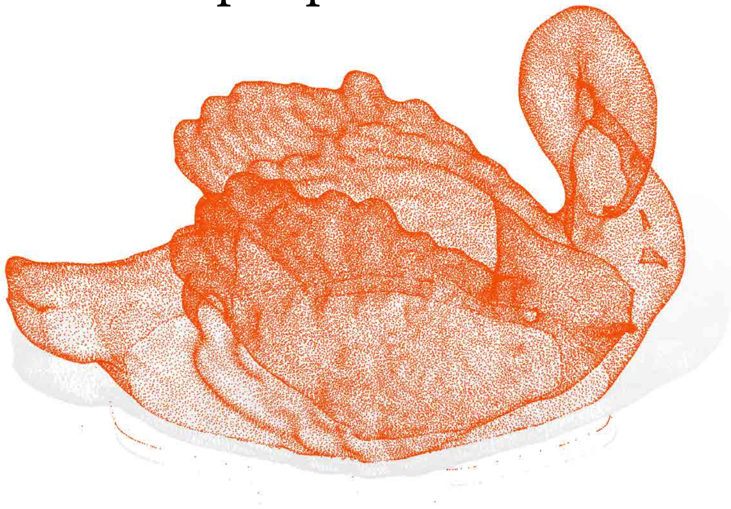


Concluding thoughts

Question common assumptions

Reach across entire pipeline

input point cloud



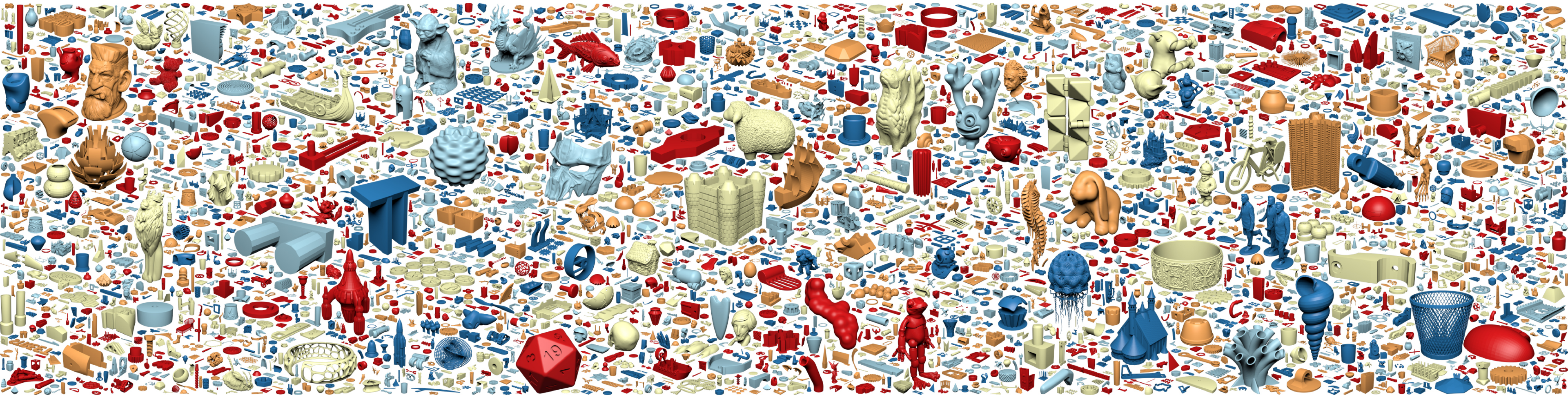
3D printed point cloud

Concluding thoughts

Question common assumptions

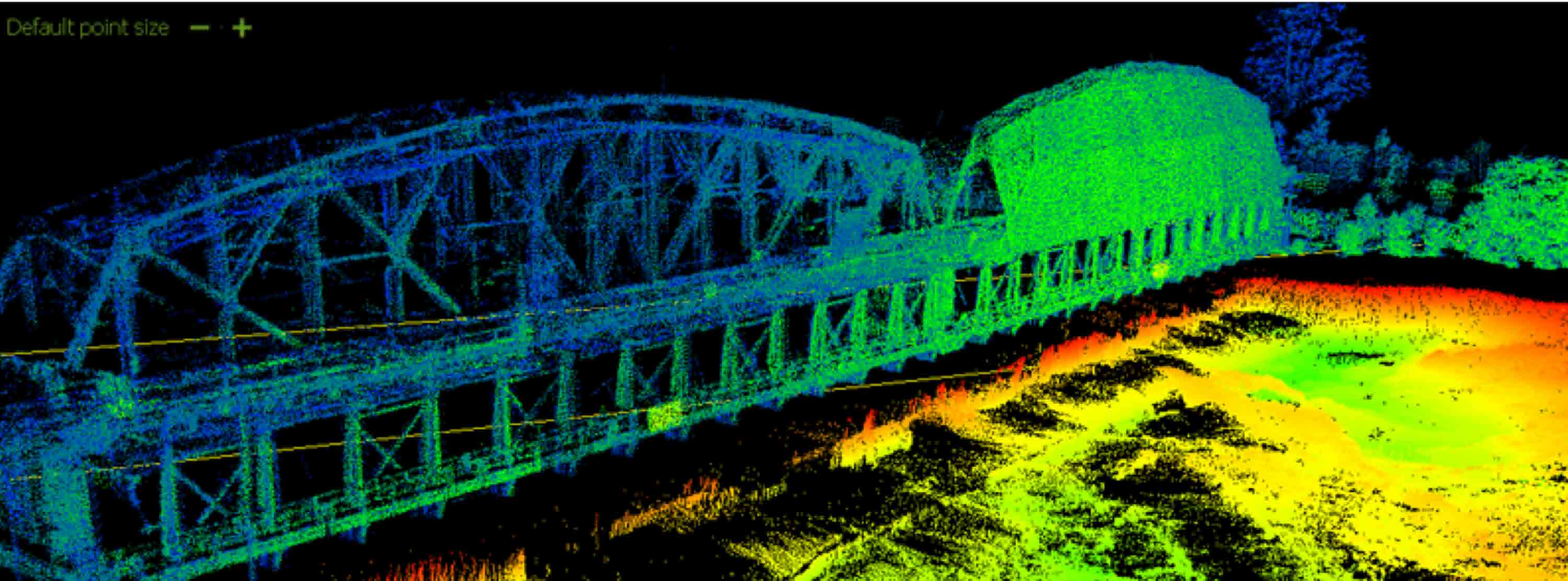
Reach across entire pipeline

Large-scale validation



Future directions

"Full stack" geometry processing

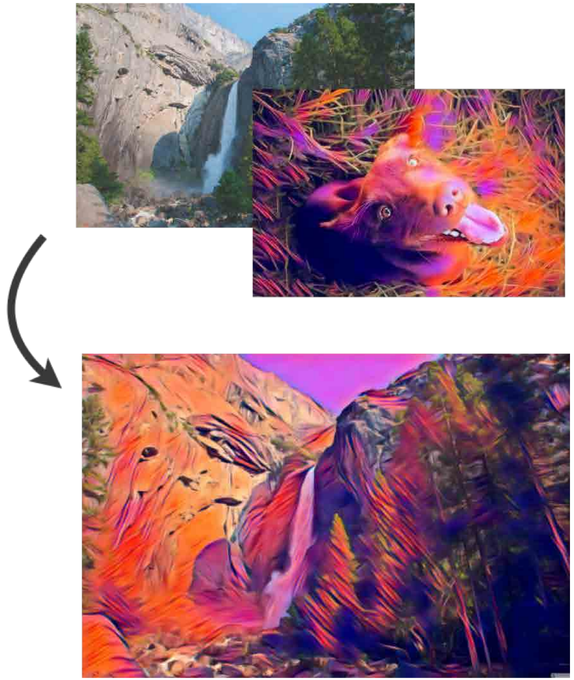


Future directions

Geometric learning *beyond* classification

Future directions

Geometric learning *beyond* classification



Future directions

Geometric learning *beyond* classification

"PAPARAZZI: Surface Editing by way of Multi-View Image Processing" [Liu, Tao, & J. 2018]

Acknowledgments…

Chun-Liang Li, Daniele Panozzo, David Levin, Denis Zorin, Derek Liu, Derek Nowrouzezahrai, Eitan Grinspun, Etienne Vouga, Gavin Barill, Herng Yi Cheng, Hsueh-Ti Derek Liu, Karan Singh, Ladislav Kavan, Leonardo Sacht, Michael Tao, Mitchell Dembowski, Neil Dickson, Olga Sorkine-Hornung, Qingnan Zhou, Rinat Abdrashitov, Ryan Schmidt, Silvia Sellán, Xifeng Gao, Yixin Hu, Yuming Ma

NSERC, Connaught Funds,
Canada Research Chairs Program, Fields Institute

Autodesk, Adobe, MESH Consultants

Geometry Processing in the Wild

Alec Jacobson
Canada Research Chair
University of Toronto
jacobson@cs.toronto.edu