

SAPIENZA
UNIVERSITÀ DI ROMA

Leonardo Rocchi

University of Birmingham
LAMS Seminars

Consiglio Nazionale delle Ricerche

March 25, 2017

A Level-Set based method for fixing overhangs in 3D printing

(a joint work with Emiliano Cristiani and Simone Cacace)

Outline

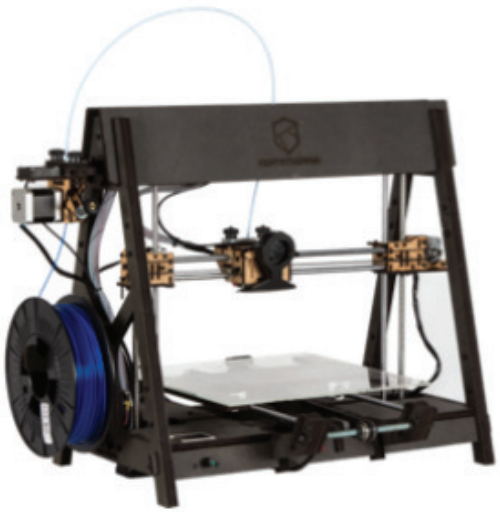
1. **Additive Manufacturing or 3D printing**
   - FDM technology
   - Main issues in 3D printing
2. **The Level-Set method**
   - Tracking an evolving surface
   - Motions along normal direction
   - Hamilton-Jacobi equations
3. **Using the Level-Set method for fixing overhangs**
   - Idea and scope
   - Printability
   - Construction of the vector field
4. **Numerical tests**

Fused Deposition Modeling

FDM is the most common technique used for 3D printing. It is an additive manufacturing technology involving material extrusion (it is sometimes also known as Fused Filament Fabrication (FFF))

▷ The creation of a 3D printed object is achieved by laying down successive layers of material until the object is created. Each of these layers can be seen as a thinly sliced horizontal cross-section of the object.

▷ FDM technology is clean, simply to use and office-friendly.





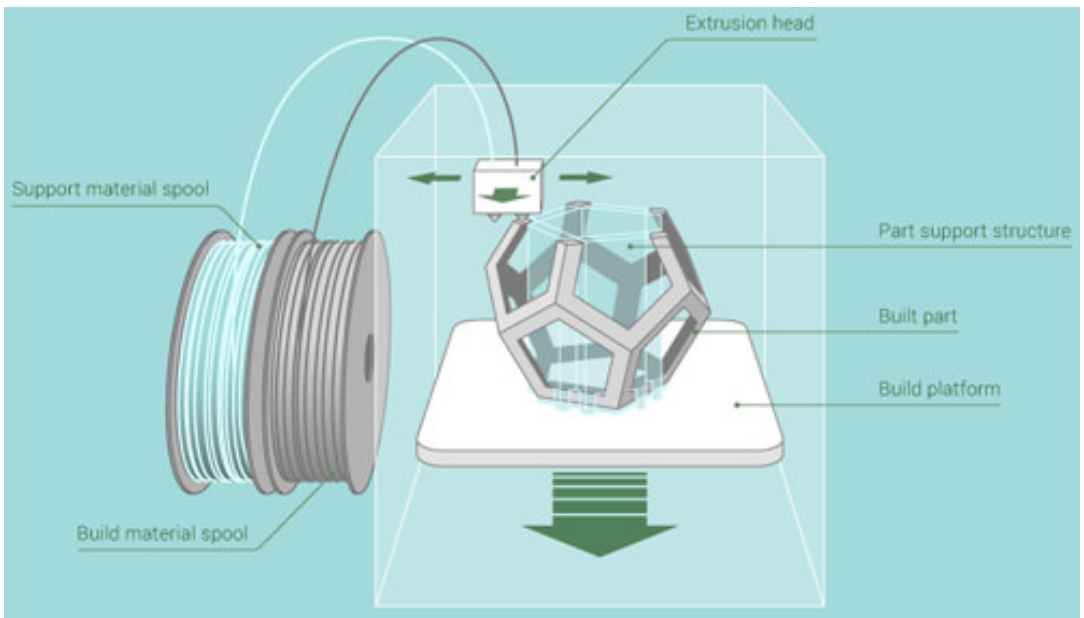


Fused Deposition Modeling

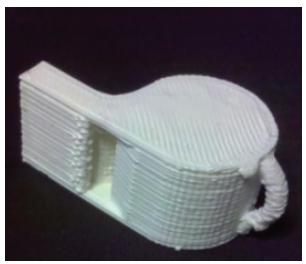
▷ The single layers are created by melting and extruding a plastic thermofilament through a nozzle (extruder), starting from the bottom layer up to the top one.

▷ Slicing resolution is important: the larger the number of slices the better the final printed result.

▷ Common materials: PLA (PolyLactid Acid) or ABS (Acrylonitrile Butadiene Styrene)











Principal steps in 3D printing

Principal steps in 3D printing via FDM

1. **CAD modelling:** generate a 3D model of the object by using any CAD software (e.g Blender, 123D, OpenSCAD).
   Alternatively: use a 3D scanner;
2. **.STL file:** the name stands for *Standard Tessellation Language*. This file contains information about the model and it can be (usually) obtained directly from the CAD software.
   Alternatively: look for your .stl file from some online database (e.g. thingiverse.com, pinshape.com);
3. **Pre-processing:** use any compatible *Slicer* to calibrate the printer settings (e.g. Cura, Slic3r). The final *G-code* produced is then read by the printer (position, orientation, extrusion path, supports);
4. **Printing**.

▷ Eventual **post-processing** step to remove any printed support.

Principal steps in 3D printing

Principal steps in 3D printing via FDM

1. **CAD modelling:** generate a 3D model of the object by using any CAD software (e.g Blender, 123D, OpenSCAD).
   Alternatively: use a 3D scanner;
2. **.STL file:** the name stands for *Standard Tessellation Language*. This file contains information about the model and it can be (usually) obtained directly from the CAD software.
   Alternatively: look for your .stl file from some online database (e.g. thingiverse.com, pinshape.com);
3. **Pre-processing:** use any compatible *Slicer* to calibrate the printer settings (e.g. Cura, Slic3r). The final *G-code* produced is then read by the printer (position, orientation, extrusion path, supports);
4. **Printing**.

▷ Eventual **post-processing** step to remove any printed support.

Principal steps in 3D printing

Principal steps in 3D printing via FDM

1. **CAD modelling:** generate a 3D model of the object by using any CAD software (e.g Blender, 123D, OpenSCAD).
   Alternatively: use a 3D scanner;
2. **.STL file:** the name stands for *Standard Tessellation Language*. This file contains information about the model and it can be (usually) obtained directly from the CAD software.
   Alternatively: look for your .stl file from some online database (e.g. thingiverse.com, pinshape.com);
3. **Pre-processing:** use any compatible *Slicer* to calibrate the printer settings (e.g. Cura, Slic3r). The final *G-code* produced is then read by the printer (position, orientation, extrusion path, supports);
4. **Printing**.

▷ Eventual **post-processing** step to remove any printed support.

Principal steps in 3D printing

Principal steps in 3D printing via FDM

1. **CAD modelling:** generate a 3D model of the object by using any CAD software (e.g Blender, 123D, OpenSCAD).
   Alternatively: use a 3D scanner;
2. **.STL file:** the name stands for *Standard Tessellation Language*. This file contains information about the model and it can be (usually) obtained directly from the CAD software.
   Alternatively: look for your .stl file from some online database (e.g. thingiverse.com, pinshape.com);
3. **Pre-processing:** use any compatible *Slicer* to calibrate the printer settings (e.g. Cura, Slic3r). The final *G-code* produced is then read by the printer (position, orientation, extrusion path, supports);
4. **Printing**.

▷ Eventual **post-processing** step to remove any printed support.

Principal steps in 3D printing

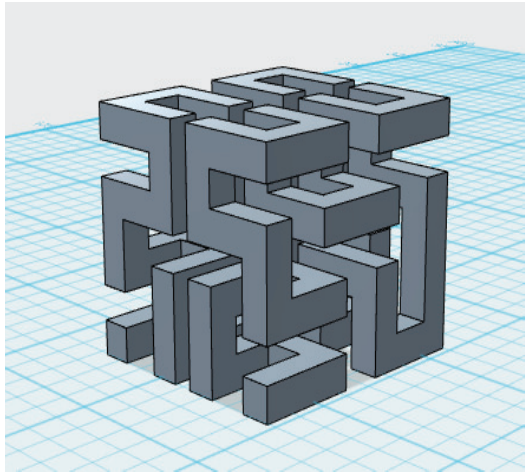
Principal steps in 3D printing via FDM

1. **CAD modelling:** generate a 3D model of the object by using any CAD software (e.g Blender, 123D, OpenSCAD).
   Alternatively: use a 3D scanner;
2. **.STL file:** the name stands for *Standard Tessellation Language*. This file contains information about the model and it can be (usually) obtained directly from the CAD software.
   Alternatively: look for your .stl file from some online database (e.g. thingiverse.com, pinshape.com);
3. **Pre-processing:** use any compatible *Slicer* to calibrate the printer settings (e.g. Cura, Slic3r). The final *G-code* produced is then read by the printer (position, orientation, extrusion path, supports);
4. **Printing**.

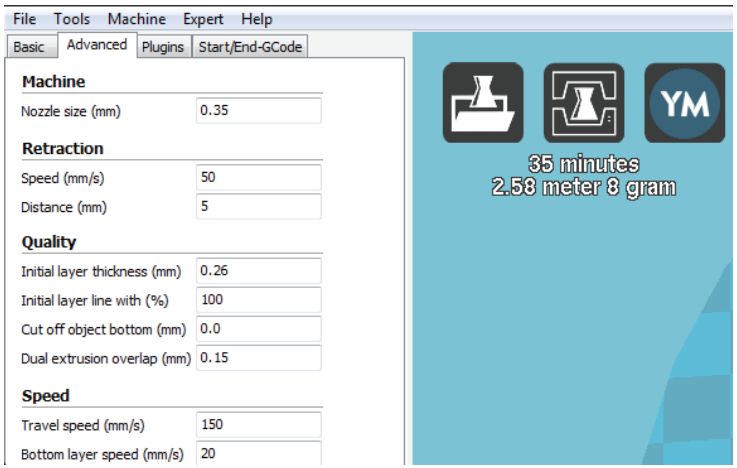
▷ Eventual **post-processing** step to remove any printed support.

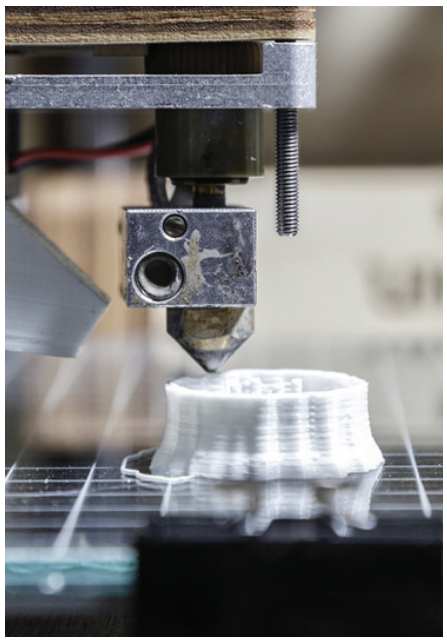
Principal steps in 3D printing via FDM

3D printing steps:



```
solid "nameOfTheSolid"
  facet normal -1 -0 0
    outer loop
      vertex -7.07035541534 2.92021799088 -5.74996852875
      vertex -7.07035541534 1.42340242863 -7.5273680687
      vertex -7.07035541534 1.11192011833 -5.70109224319
    endloop
  endfacet
  facet normal -1 1.87925490991e-06 1.09245002022e-06
    outer loop
      vertex -7.07035255432 3.90445995331 -4.61336088181
      vertex -7.07035255432 4.67988586426 -5.94726419449
      vertex -7.07035541534 2.92021799088 -5.74996852875
    endloop
  endfacet
```





3D printing applications

▷ **Medical applications**

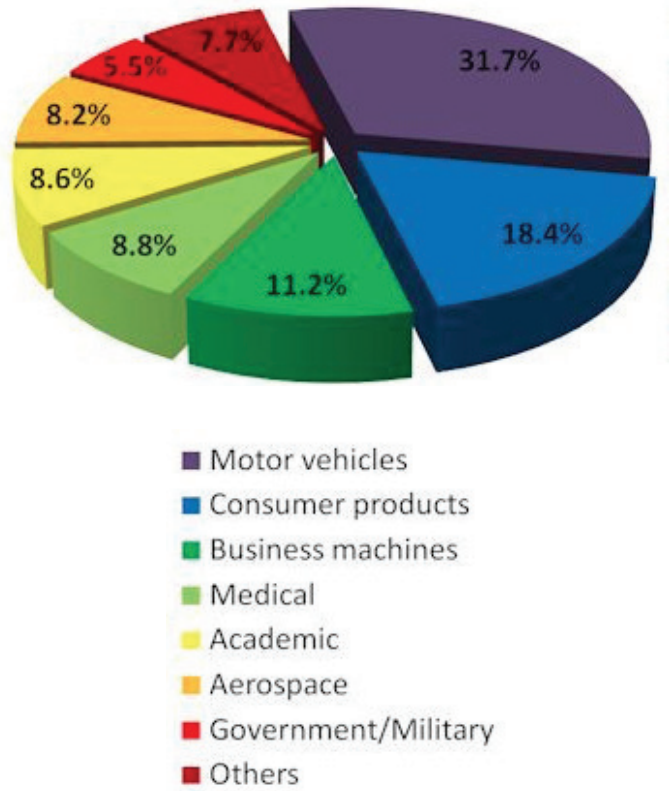
- 3D bio-printing using cells (printing of tiny strips of organs tissue);
- 3D printed prosthetics: detailed multi-material models reproducing real anatomy's organs to enhance surgical training.

▷ **Motor vehicles and aerospace applications**

- 3D printed of mechanical components: brackets, parts of engines, . . . (with plastic brackets Airbus save 650K Euros per year, and $\sim$ 2.5 Kg on a single aircraft.

▷ **Fashion**: 3D printed shoes, glasses and real clothing in flexible PLA.

▷ **Architecture**: urban planning and real-life models

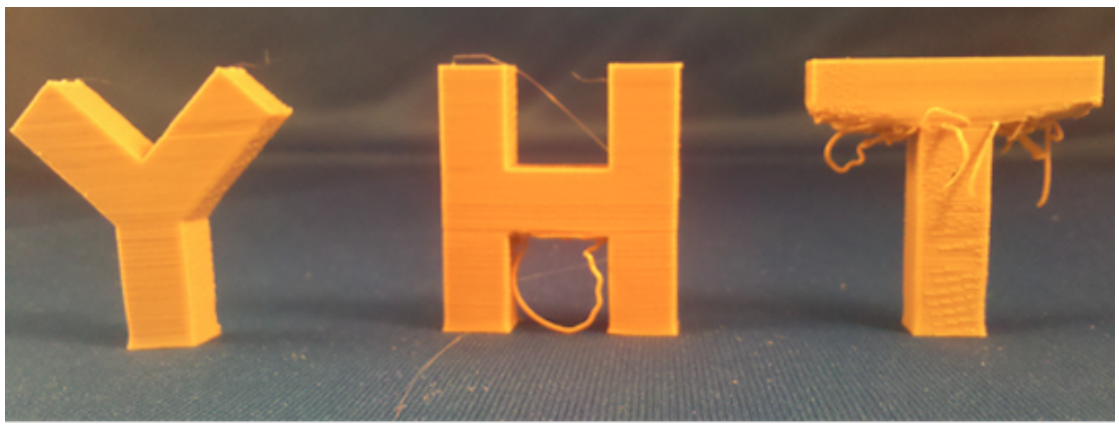


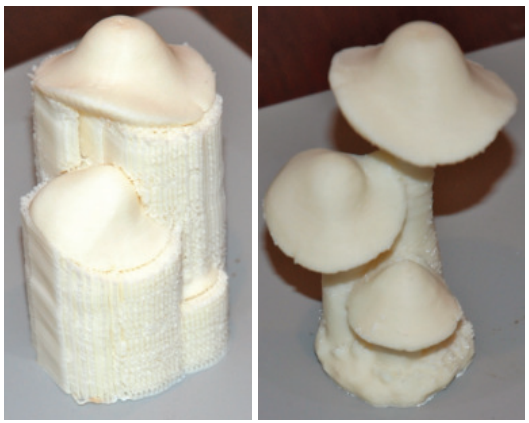
3D printing issues: bridges and overhangs

▷ Each layer of the object has to be deposited on top of an existing surface (i.e. the previously printed layers), otherwise the melted plastic solidifies "in the air".

FDM issues:

- **Bridges**: the filament is extruded between two points over empty space (the longer the distance, the more the bending);
- **Overhangs**: the filament of the upper layer protrudes a little over the lower layer (within a certain limit).





▷ If printer's settings don't manage to handle those issues, a **scaffolding structures** has to be inserted. Result: large printing time and more material wasted.

Outline

1 Additive Manufacturing or 3D printing
FDM technology
Main issues in 3D printing

2 The Level-Set method
Tracking an evolving surface
Motions along normal direction
Hamilton-Jacobi equations

3 Using the Level-Set method for fixing overhangs
Idea and scope
Printability
Construction of the vector field

4 Numerical tests

The Level-Set method

The Level-Set (LS) method is a **versatile** and **extensible** method introduced in 1988 by Osher & Sethian[1] for tracking the curvature-dependent evolving interfaces in the setting of Hamilton-Jacobi equations.

▷ Example: You can think of a fire-front propagation in a wildland;

Suppose we are given an evolving $(d-1)$-dimensional surface embedded in $\mathbb{R}^d$ transported by a given/known velocity vector field $\mathbf{v} : \mathbb{R}^d \to \mathbb{R}^d$.

▷ **Idea**: the surface is recovered by the level-set 0 of a scalar hypersurface (the **level-set function**) $\varphi : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}$.

▷ If $d = 2$, we are tracking the evolution of a 1-dimensional front in $\mathbb{R}^2$ space;
▷ If $d = 3$, we are tracking the evolution of a 2-dimensional surface in $\mathbb{R}^3$ space. . .

NOTE: if $d \geq 3$, it is not possible to display φ.

[1] Osher, Sethian, *J. Comput. Phys.*, 79(1):12-49, 1988

The Level-Set method

The Level-Set (LS) method is a **versatile** and **extensible** method introduced in 1988 by Osher & Sethian[1] for tracking the curvature-dependent evolving interfaces in the setting of Hamilton-Jacobi equations.

▷ Example: You can think of a fire-front propagation in a wildland;

Suppose we are given an evolving $(d-1)$-dimensional surface embedded in $\mathbb{R}^d$ transported by a given/known velocity vector field $\mathbf{v} : \mathbb{R}^d \to \mathbb{R}^d$.

▷ **Idea**: the surface is recovered by the level-set 0 of a scalar hypersurface (the **level-set function**) $\varphi : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}$.

▷ If $d = 2$, we are tracking the evolution of a 1-dimensional front in $\mathbb{R}^2$ space;
▷ If $d = 3$, we are tracking the evolution of a 2-dimensional surface in $\mathbb{R}^3$ space. . .

NOTE: if $d \geq 3$, it is not possible to display φ.

[1] Osher, Sethian, *J. Comput. Phys.*, 79(1):12-49, 1988

The Level-Set method

The Level-Set (LS) method is a **versatile** and **extensible** method introduced in 1988 by Osher & Sethian[1] for tracking the curvature-dependent evolving interfaces in the setting of Hamilton-Jacobi equations.

▷ Example: You can think of a fire-front propagation in a wildland;

Suppose we are given an evolving $(d-1)$-dimensional surface embedded in $\mathbb{R}^d$ transported by a given/known velocity vector field $\mathbf{v} : \mathbb{R}^d \to \mathbb{R}^d$.

▷ **Idea**: the surface is recovered by the level-set 0 of a scalar hypersurface (the **level-set function**) $\varphi : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}$.

▷ If $d = 2$, we are tracking the evolution of a 1-dimensional front in $\mathbb{R}^2$ space;
▷ If $d = 3$, we are tracking the evolution of a 2-dimensional surface in $\mathbb{R}^3$ space...

NOTE: if $d \geq 3$, it is not possible to display φ.

[1] Osher, Sethian, *J. Comput. Phys.*, 79(1):12-49, 1988

The Level-Set method

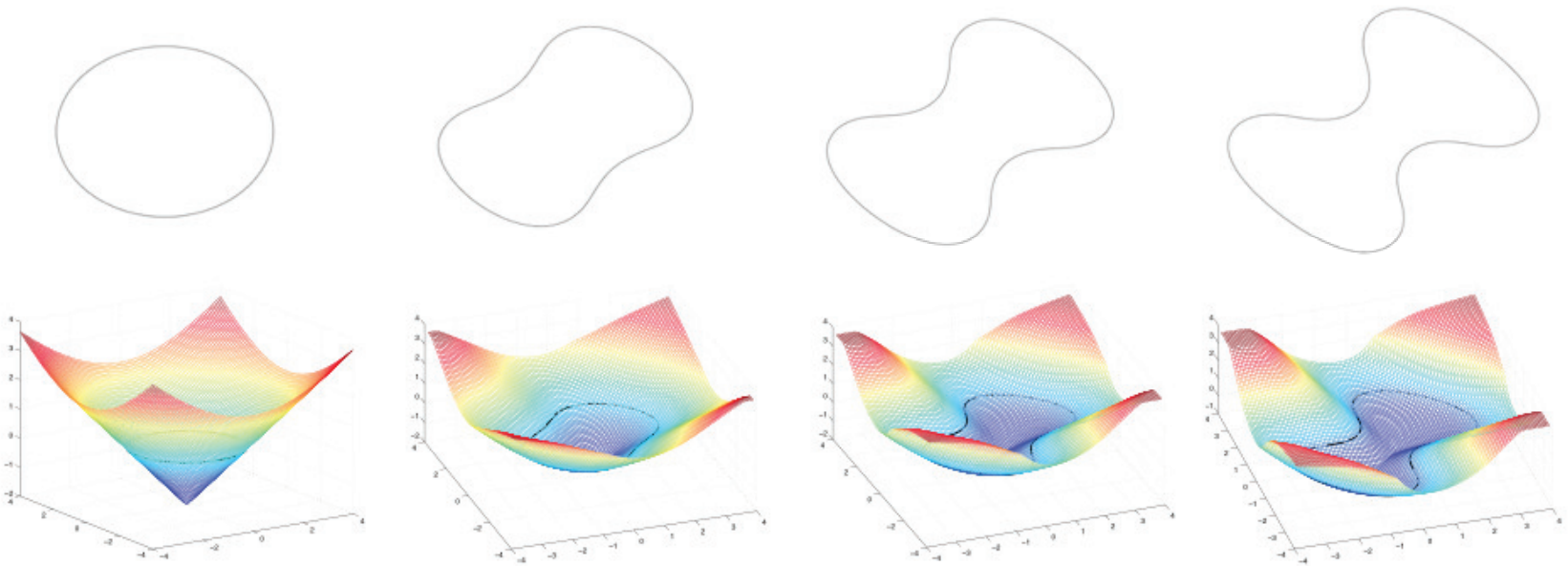
▷ **Example**: circular 1-dimensional front embedded in $\mathbb{R}^2$.

Consider the front Γ given by

$$\Gamma = \{(x, y) \in \mathbb{R}^2 \ : \ x^2 + y^2 = r^2, r > 0\},$$

and the corresponding level-set function $\varphi : \mathbb{R}^2 \to \mathbb{R}$ given by

$$\varphi = \varphi(x, y) := \sqrt{x^2 + y^2} - r.$$



The Level-Set function

General setting

▷ Let Σ_0 be a bounded closed surface in $\mathbb{R}^{d-1}$ at initial time $t = 0$, and let Ω_0 be the domain strictly contained in Σ_0, i.e. $\Sigma_0 = \partial\Omega_0$.
We want to track the unknown evolution Σ_t, $t > 0$, under the action of a given/known vector field $\mathbf{v}$, so that $\Sigma_t = \partial\Omega_t$.

Define the **level-set function** $\varphi : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that

$$\text{for all } t \geq 0, \quad \begin{cases} \varphi(t, \mathbf{x}) > 0, & \mathbf{x} \notin \overline{\Omega_t}, \\ \varphi(t, \mathbf{x}) = 0, & \mathbf{x} \in \Sigma_t, \\ \varphi(t, \mathbf{x}) < 0, & \mathbf{x} \in \Omega_t. \end{cases}$$

Then, we recover the surface by looking at the zero level-set of φ, i.e.

$$\Sigma_t = \{\mathbf{x} \in \mathbb{R}^d \, : \, \varphi(t, \mathbf{x}) = 0\}, \quad \text{for all } t \geq 0.$$

▷ A typical initialisation is $\varphi(0, \mathbf{x})$ equal to the signed distance function from the surface Σ_0.

The Level-Set function

General setting

▷ Let Σ_0 be a bounded closed surface in $\mathbb{R}^{d-1}$ at initial time $t = 0$, and let Ω_0 be the domain strictly contained in Σ_0, i.e. $\Sigma_0 = \partial\Omega_0$.
We want to track the unknown evolution Σ_t, $t > 0$, under the action of a given/known vector field $\mathbf{v}$, so that $\Sigma_t = \partial\Omega_t$.

Define the **level-set function** $\varphi : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that

$$\text{for all } t \geq 0, \quad \begin{cases} \varphi(t, \mathbf{x}) > 0, & \mathbf{x} \notin \overline{\Omega_t}, \\ \varphi(t, \mathbf{x}) = 0, & \mathbf{x} \in \Sigma_t, \\ \varphi(t, \mathbf{x}) < 0, & \mathbf{x} \in \Omega_t. \end{cases}$$

Then, we recover the surface by looking at the zero level-set of φ, i.e.

$$\Sigma_t = \{\mathbf{x} \in \mathbb{R}^d \,:\, \varphi(t, \mathbf{x}) = 0\}, \quad \text{for all } t \geq 0.$$

▷ A typical initialisation is $\varphi(0, \mathbf{x})$ equal to the signed distance function from the surface Σ_0.

The Level-Set equation

Let $\mathbf{x}(t)$ be the position at time t of a point $\mathbf{x} = (x_1, \ldots, x_d) \in \Sigma_t$, and impose that $\mathbf{x}(t)$ belongs to the zero level-set of φ, i.e. $\varphi(t, \mathbf{x}(t)) = 0$, for $t \geq 0$.

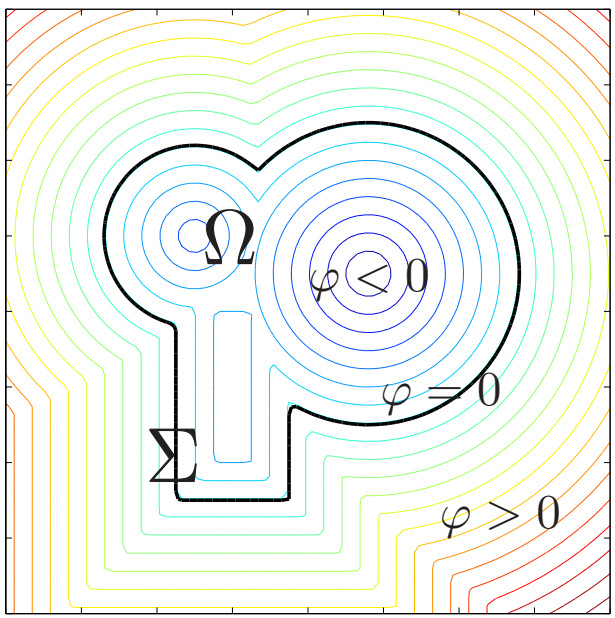
▷ The level-set function φ satisfies the following 1st order Hamilton-Jacobi equation

$$\partial_t \varphi(t, \mathbf{x}) + \mathbf{v} \cdot \nabla \varphi(t, \mathbf{x}) = 0 \qquad \forall\, \mathbf{x} \in \mathbb{R}^d,\ t \geq 0,$$

with a suitable initial condition $\varphi_0(\mathbf{x}) := \varphi(0, \mathbf{x})$.

▷ The evolution of φ regards all the possible level-sets that move together according to $\mathbf{v}$. Level-set 0 is just a choice.

▷ The vector field $\mathbf{v}$ can be a function of the level-set function φ, i.e. $\mathbf{v} = \mathbf{v}[\varphi](\mathbf{x})$.



Unit normal and mean curvature

Appealing feature: some geometrical properties of Σ_t are described by means of the corresponding level-set function, for example, the unit exterior unit normal $\hat{\mathbf{n}}$ and the (mean) curvature k

$$\hat{\mathbf{n}} = \frac{\nabla\varphi}{|\nabla\varphi|} \quad \text{and} \quad k = \operatorname{div}(\hat{\mathbf{n}}).$$

- If $\mathbf{v}[\varphi](t,\mathbf{x}) = v(\mathbf{x})\hat{\mathbf{n}}(t,\mathbf{x})$, for some scalar function v, then

$$\partial_t\varphi(t,\mathbf{x}) + v(\mathbf{x})|\nabla\varphi(t,\mathbf{x})| = 0 \qquad \forall\,\mathbf{x}\in\mathbb{R}^d,\ t\geq 0,$$

  describing the evolution of the front along its normal direction;

- If $\mathbf{v}[\varphi](t,\mathbf{x}) = -\operatorname{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right)\hat{\mathbf{n}}(t,\mathbf{x})$, then

$$\partial_t\varphi(t,\mathbf{x}) - \operatorname{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right)|\nabla\varphi(t,\mathbf{x})| = 0 \qquad \forall\,\mathbf{x}\in\mathbb{R}^d,\ t\geq 0,$$

  describing the motion under mean curvature.

Unit normal and mean curvature

Appealing feature: some geometrical properties of Σ_t are described by means of the corresponding level-set function, for example, the unit exterior unit normal $\hat{\mathbf{n}}$ and the (mean) curvature k

$$\hat{\mathbf{n}} = \frac{\nabla\varphi}{|\nabla\varphi|} \quad \text{and} \quad k = \operatorname{div}(\hat{\mathbf{n}}).$$

- If $\mathbf{v}[\varphi](t,\mathbf{x}) = v(\mathbf{x})\hat{\mathbf{n}}(t,\mathbf{x})$, for some scalar function v, then

$$\partial_t\varphi(t,\mathbf{x}) + v(\mathbf{x})|\nabla\varphi(t,\mathbf{x})| = 0 \qquad \forall\,\mathbf{x}\in\mathbb{R}^d,\ t\geq 0,$$

  describing the evolution of the front along its normal direction;

- If $\mathbf{v}[\varphi](t,\mathbf{x}) = -\operatorname{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right)\hat{\mathbf{n}}(t,\mathbf{x})$, then

$$\partial_t\varphi(t,\mathbf{x}) - \operatorname{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right)|\nabla\varphi(t,\mathbf{x})| = 0 \qquad \forall\,\mathbf{x}\in\mathbb{R}^d,\ t\geq 0,$$

  describing the motion under mean curvature.

Unit normal and mean curvature

Appealing feature: some geometrical properties of Σ_t are described by means of the corresponding level-set function, for example, the unit exterior unit normal $\hat{\mathbf{n}}$ and the (mean) curvature k

$$\hat{\mathbf{n}} = \frac{\nabla\varphi}{|\nabla\varphi|} \quad \text{and} \quad k = \operatorname{div}(\hat{\mathbf{n}}).$$

- If $\mathbf{v}[\varphi](t,\mathbf{x}) = v(\mathbf{x})\hat{\mathbf{n}}(t,\mathbf{x})$, for some scalar function v, then

$$\partial_t\varphi(t,\mathbf{x}) + v(\mathbf{x})|\nabla\varphi(t,\mathbf{x})| = 0 \qquad \forall\,\mathbf{x}\in\mathbb{R}^d,\ t\geq 0,$$

  describing the evolution of the front along its normal direction;

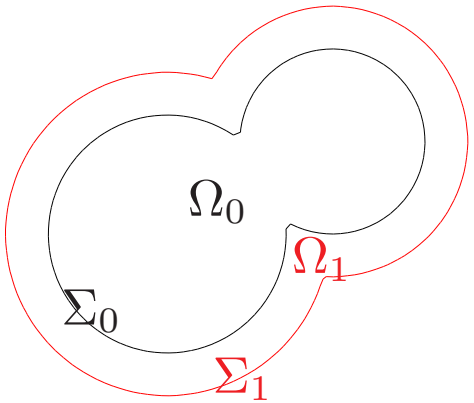
- If $\mathbf{v}[\varphi](t,\mathbf{x}) = -\operatorname{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right)\hat{\mathbf{n}}(t,\mathbf{x})$, then

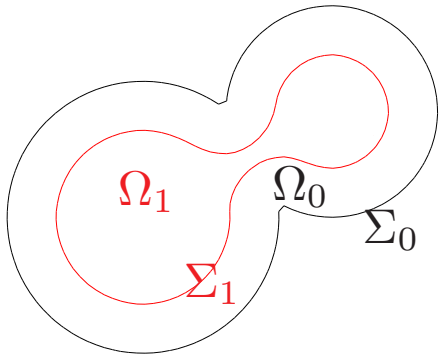
$$\partial_t\varphi(t,\mathbf{x}) - \operatorname{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right)|\nabla\varphi(t,\mathbf{x})| = 0 \qquad \forall\,\mathbf{x}\in\mathbb{R}^d,\ t\geq 0,$$

  describing the motion under mean curvature.

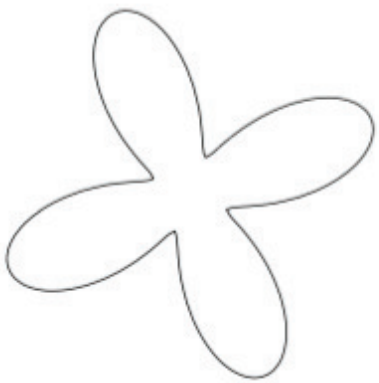
Unit normal and mean curvature

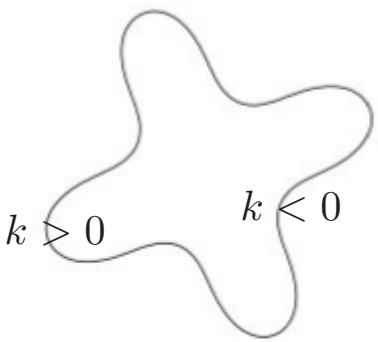
▷ **Monotone motion along the normal direction**: if $v(\mathbf{x}) > 0$ for every $\mathbf{x}$, then ww obtain an outward expansion of the surface ($\Omega_0 \subset \Omega_1$); if $v(\mathbf{x}) < 0$ we get in a contraction ($\Omega_0 \supset \Omega_1$).



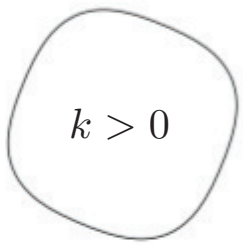


▷ **Motion under mean curvature**: if $k(\mathbf{x}) > 0$ the surface will move inward; if $k(\mathbf{x}) < 0$ the front will move outward.









Hamilton-Jacobi equations

▷ **General nonlinear second order Hamilton-Jacobi equation:**

$$\partial_t \varphi(t,\mathbf{x}) + H(t,\mathbf{x},\varphi(t,\mathbf{x}),\nabla\varphi(t,\mathbf{x}),\mathbf{H}\varphi(t,\mathbf{x})) = 0 \qquad \forall\, \mathbf{x}\in\mathbb{R}^d,\ t\geq 0,$$

where $\mathbf{H}\varphi \in \mathcal{S}_d$ is the Hessian matrix of φ, $\mathcal{S}_d$ is the set of symmetric $d\times d$ real matrices, and $H\colon \mathbb{R}^+\times\mathbb{R}^d\times\mathbb{R}\times\mathbb{R}^d\times\mathcal{S}_d\to\mathbb{R}$ is a continuous function called Hamiltonian;

- General LS equation (1st order): $H(t,\mathbf{x},\nabla\varphi) = \mathbf{v}\cdot\nabla\varphi$;
- Motion along normal (1st order): $H(t,\mathbf{x},\nabla\varphi) = v|\nabla\varphi|$;
- Motion under curvature (2nd order): $H(t,\mathbf{x},\nabla\varphi,\mathbf{H}\varphi) = -\mathrm{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right)|\nabla\varphi|$;

In general, for such equations there will not be unique classic solutions, no matter the regularity of H and initial data $\varphi(0,\mathbf{x})$.

▷ **Crandall & Lions**[2] in 1983 introduced the notion of **viscosity solutions**: they permit to select a unique weak solution among the existing infinitely many.

[2] Crandall, Lions, *Trans. Amer. Math. Soc.*, 43(167):1-19, 1983

Hamilton-Jacobi equations: assumptions

▷ Main assumptions on H for the **well-posedness** in the classical theory of viscosity solutions

- $H\colon [0,T] \times \Omega \times \mathbb{R} \times \mathbb{R}^d \times \mathcal{S}_d \to \mathbb{R}$ is **degenerate elliptic** if, for every t,

$$H(t, \mathbf{x}, r, p, \mathbf{X}) \leq H(t, \mathbf{x}, r, p, \mathbf{Y}), \quad \text{whenever } \mathbf{Y} \leq \mathbf{X} \tag{1}$$

  for every $\mathbf{X}, \mathbf{Y} \in \mathcal{S}_n$ ($\mathbf{Y} \leq \mathbf{X}$ means $\mathbf{X} - \mathbf{Y}$ is semipositive definite);
- $H\colon [0,T] \times \Omega \times \mathbb{R} \times \mathbb{R}^d \times \mathcal{S}_d \to \mathbb{R}$ is **proper** if, for every t, H is degenerate elliptic and

$$H(t, \mathbf{x}, r, p, \mathbf{X}) \leq H(t, \mathbf{x}, s, p, \mathbf{Y}), \quad \text{whenever } r \leq s \tag{2}$$

  that is, H is not decreasing with respect to r.

These hypothesis are the basis for the existence and uniqueness of viscosity solutions.

Hamilton-Jacobi equations: examples

▷ **First order equations**: $H(\mathbf{x}, r, p, \mathbf{X}) = H(\mathbf{x}, r, p)$. They are always degenerate elliptic and thus they are proper if and only if (2) holds.

- Motion along the normal: $H(\mathbf{x}, r, p, \mathbf{X}) = H(\mathbf{x}, p) = v(\mathbf{x})|p|$ is proper.

▷ **General 2nd order linear equations**:

$$H(\mathbf{x}, r, p, \mathbf{X}) = -\mathrm{tr}(A(\mathbf{x})\,\mathbf{X}) + \sum_{i=1}^{d} b_i(\mathbf{x}) p_i + c(\mathbf{x}) r - f(x)$$

Here, H is degenerate elliptic if and only if A is positive definite, and it is proper if also $c(\mathbf{x}) \geq 0$. Ex: Diffusion-reaction and Laplace equations ($A = I$).

▷ Motion under mean curvature: carrying out the differentiations yields

$$\partial_t \varphi - \Delta \varphi + \sum_{i,j=1}^{d} \partial^2_{x_i x_j} \varphi \, \partial_{x_i} \varphi \, \partial_{x_j} \varphi \, |\nabla \varphi|^{-2} = 0,$$

that can be rewritten as $\partial_t \varphi + H(t, \mathbf{x}, r, p, \mathbf{X}) = 0$, where for every t

$$H(x, r, p, \mathbf{X}) = H(x, p, \mathbf{X}) = -\mathrm{tr}\left(\left(\mathrm{Id} - \frac{p \otimes p}{|p|^2}\right)\mathbf{X}\right) = -\mathrm{tr}(P\mathbf{X}).$$

Then, H is degenerate elliptic if P is positive definite.

Hamilton-Jacobi equations: examples

▷ **First order equations**: $H(\mathbf{x}, r, p, \mathbf{X}) = H(\mathbf{x}, r, p)$. They are always degenerate elliptic and thus they are proper if and only if (2) holds.

- Motion along the normal: $H(\mathbf{x}, r, p, \mathbf{X}) = H(\mathbf{x}, p) = v(\mathbf{x})|p|$ is proper.

▷ **General 2nd order linear equations**:

$$H(\mathbf{x}, r, p, \mathbf{X}) = -\mathrm{tr}(A(\mathbf{x})\,\mathbf{X}) + \sum_{i=1}^{d} b_i(\mathbf{x}) p_i + c(\mathbf{x}) r - f(x)$$

Here, H is degenerate elliptic if and only if A is positive definite, and it is proper if also $c(\mathbf{x}) \geq 0$. Ex: Diffusion-reaction and Laplace equations ($A = I$).

▷ Motion under mean curvature: carrying out the differentiations yields

$$\partial_t \varphi - \Delta\varphi + \sum_{i,j=1}^{d} \partial^2_{x_i x_j} \varphi\, \partial_{x_i} \varphi\, \partial_{x_j} \varphi\, |\nabla \varphi|^{-2} = 0,$$

that can be rewritten as $\partial_t \varphi + H(t, \mathbf{x}, r, p, \mathbf{X}) = 0$, where for every t

$$H(x, r, p, \mathbf{X}) = H(x, p, \mathbf{X}) = -\mathrm{tr}\left(\left(\mathrm{Id} - \frac{p \otimes p}{|p|^2}\right)\mathbf{X}\right) = -\mathrm{tr}(P\mathbf{X}).$$

Then, H is degenerate elliptic if P is positive definite.

Hamilton-Jacobi equations: examples

▷ **First order equations**: $H(\mathbf{x}, r, p, \mathbf{X}) = H(\mathbf{x}, r, p)$. They are always degenerate elliptic and thus they are proper if and only if (2) holds.

- Motion along the normal: $H(\mathbf{x}, r, p, \mathbf{X}) = H(\mathbf{x}, p) = v(\mathbf{x})|p|$ is proper.

▷ **General 2nd order linear equations**:

$$H(\mathbf{x}, r, p, \mathbf{X}) = -\mathrm{tr}(A(\mathbf{x})\,\mathbf{X}) + \sum_{i=1}^{d} b_i(\mathbf{x}) p_i + c(\mathbf{x}) r - f(x)$$

Here, H is degenerate elliptic if and only if A is positive definite, and it is proper if also $c(\mathbf{x}) \geq 0$. Ex: Diffusion-reaction and Laplace equations ($A = I$).
▷ Motion under mean curvature: carrying out the differentiations yields

$$\partial_t \varphi - \Delta\varphi + \sum_{i,j=1}^{d} \partial^2_{x_i x_j} \varphi \, \partial_{x_i} \varphi \, \partial_{x_j} \varphi \, |\nabla \varphi|^{-2} = 0,$$

that can be rewritten as $\partial_t \varphi + H(t, \mathbf{x}, r, p, \mathbf{X}) = 0$, where for every t

$$H(x, r, p, \mathbf{X}) = H(x, p, \mathbf{X}) = -\mathrm{tr}\left(\left(\mathrm{Id} - \frac{p \otimes p}{|p|^2}\right)\mathbf{X}\right) = -\mathrm{tr}(P\mathbf{X}).$$

Then, H is degenerate elliptic if P is positive definite.

Outline

1 Additive Manufacturing or 3D printing
- FDM technology
- Main issues in 3D printing

2 The Level-Set method
- Tracking an evolving surface
- Motions along normal direction
- Hamilton-Jacobi equations

3 **Using the Level-Set method for fixing overhangs**
- Idea and scope
- Printability
- Construction of the vector field

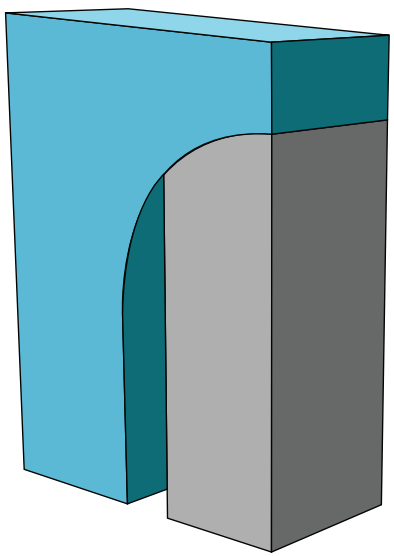
4 Numerical tests

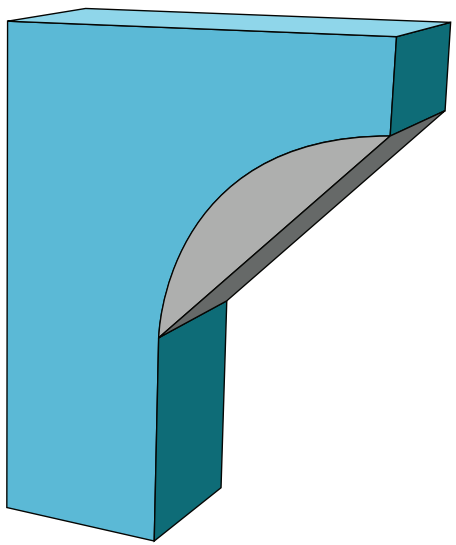
Fixing overhangs

▷ **Problem**: suppose the 3D object has some hanging parts which require supporting structures to be printed;

▷ **Goal**: get rid of scaffolding structures extended until the build plate;

▷ **Idea**: use the level-set method to "expand", or to "evolve" the hanging parts in order to make them fully printable *with no supports* (a much more smaller one!).





Fixing overhangs

Level-set formulation

Consider the object to be printed as a 3D domain $\Omega_0 \subset \mathbb{R}^3$ and let $\Sigma_0 = \partial\Omega_0$ be the surface of the object which is not fully printable because of some hanging parts

Construct an *ad hoc* vector field $\mathbf{v}\colon \mathbb{R}^3 \to \mathbb{R}^3$, such that $\mathbf{v}(x) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$ and let $\varphi : \mathbb{R}^+ \times \mathbb{R}^3 \to \mathbb{R}$ be the level-set function such that $\Sigma_t = \{\mathbf{x} \in \mathbb{R}^3 \,:\, \varphi(t, \mathbf{x}) = 0\}$, for each $t \geq 0$.

We will then solve:

$$\begin{aligned} \partial_t \varphi(t, \mathbf{x}) + \widetilde{v}(\mathbf{x})|\nabla\varphi(t, \mathbf{x})| = 0 \qquad & t > 0,\ \mathbf{x} \in \mathbb{R}^3, \\ \varphi(t, \mathbf{x}) = \varphi_0(\mathbf{x}) \qquad & t = 0,\ \mathbf{x} \in \mathbb{R}^3. \end{aligned}$$

▷ **The surface Σ_t will evolve until an optimal surface Σ_* is reached;**

▷ **The final object Ω_* will be printed and the difference $\Omega_* \setminus \Omega_0$ removed.**

Object printability

▷ Let $\hat{\mathbf{g}} = (0, 0, -1)$ be the unit gravity vector;
▷ Let $\hat{\mathbf{n}}(\mathbf{x})$ the unit exterior normal to the surface Σ at the point $\mathbf{x}$;
▷ Let $\theta(\hat{\mathbf{n}}(\mathbf{x})) := \arccos(\hat{\mathbf{g}} \cdot \hat{\mathbf{n}}(\mathbf{x}))$ be the angle between $\hat{\mathbf{g}}$ and $\hat{\mathbf{n}}$ at the point $\mathbf{x}$.

Definition (Printability)

Given a limit angle $\bar{\alpha} \in (0, \pi/2)$, a point $\mathbf{x} \in \Sigma$ is said to be

$$\begin{cases} \text{unprintable} & \text{if } \theta \in [0, \bar{\alpha}) \cup (2\pi - \bar{\alpha}, 2\pi] \\ \text{safe} & \text{if } \theta \in [\pi/2, 3\pi/2] \\ \text{modifiable} & \text{otherwise} \end{cases}$$

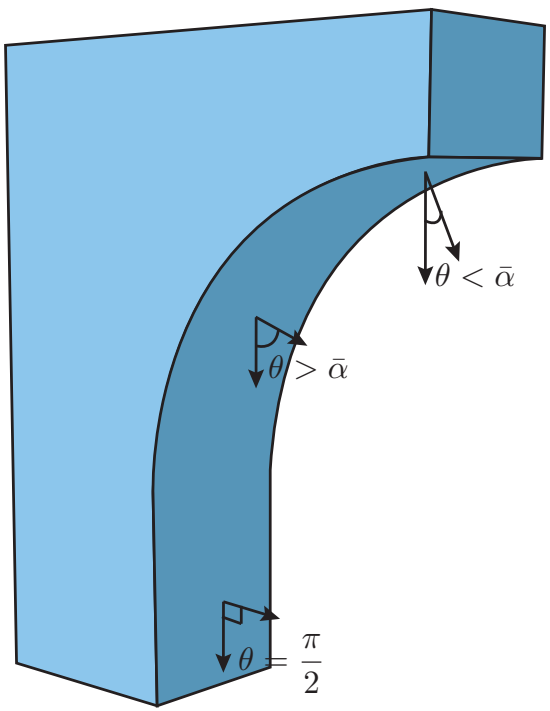
▷ Safe and modifiable points compose the surface's parts printable without any support;
▷ It is convenient to move also modifiable points in order to make printable the unprintable ones;
▷ Generally, $\bar{\alpha} = \pi/4$ (45-degree rule), though it depends on 3D printer setting and properties.

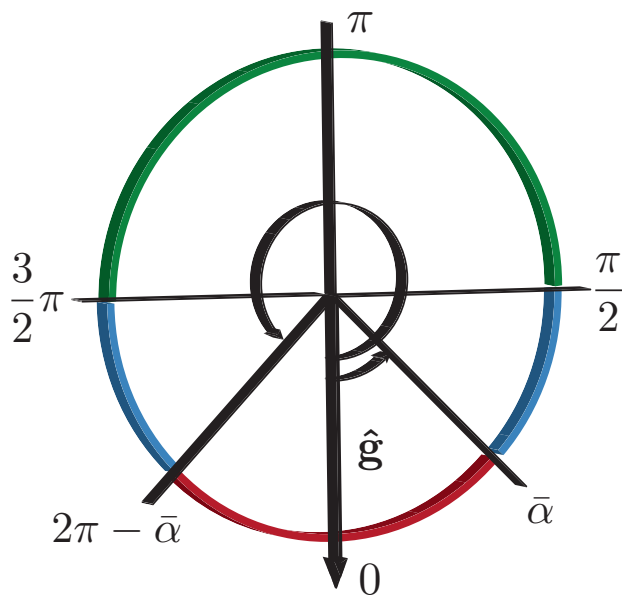
Object printability

Example

▷ Unprintable points = red region;

▷ Printable points = $\begin{cases} \text{Modifiable} = \text{blue region} \\ \text{Safe} = \text{green region} \end{cases}$





Vector Field $\mathbf{v}(\mathbf{x}) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$

Let $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$, $\hat{\mathbf{n}} = (n_1, n_2, n_3) \in \mathbb{R}^3$ and $z_{\min}$ and $z_{\max}$ be the minimum and the maximum height of the object, respectively.

- **Positivity:** we need to guarantee that $\Omega_0 \subseteq \Omega_t \subset \Omega_*$ for all $t > 0$. Thus, we impose $\widetilde{v}(\mathbf{x}) \geq 0$. Moreover, $\widetilde{v} = 0$ when $z \leq z_{\min}$;
- **Unprintable points:** we consider the term

$$v_1(\hat{\mathbf{n}}; \bar{\alpha}) := \left(\cos(\theta(\hat{\mathbf{n}})) - \cos(\bar{\alpha})\right)^+$$

  which lets the unprintable points move outward and stops them as far as they become printable;
- **Lower the faster:** the unprintable lower points have to move faster respect to the higher ones. We introduce a term $(z_{\max} - z)$;
- **Modifiable points:** if necessary, modifiable are moved outward by the curvature:

$$v_2(\mathbf{x}) := k(\mathbf{x})^-;$$

- **Blockage of safe points:** safe points have to be excluded from the evolution. They are characterised by $n_3 \geq 0$;

Vector Field $\mathbf{v}(\mathbf{x}) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$

Let $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$, $\hat{\mathbf{n}} = (n_1, n_2, n_3) \in \mathbb{R}^3$ and $z_{\min}$ and $z_{\max}$ be the minimum and the maximum height of the object, respectively.

- **Positivity:** we need to guarantee that $\Omega_0 \subseteq \Omega_t \subset \Omega_*$ for all $t > 0$. Thus, we impose $\widetilde{v}(\mathbf{x}) \geq 0$. Moreover, $\widetilde{v} = 0$ when $z \leq z_{\min}$;
- **Unprintable points:** we consider the term

$$v_1(\hat{\mathbf{n}}; \bar{\alpha}) := \left(\cos(\theta(\hat{\mathbf{n}})) - \cos(\bar{\alpha})\right)^+$$

  which lets the unprintable points move outward and stops them as far as they become printable;
- **Lower the faster:** the unprintable lower points have to move faster respect to the higher ones. We introduce a term $(z_{\max} - z)$;
- **Modifiable points:** if necessary, modifiable are moved outward by the curvature:

$$v_2(\mathbf{x}) := k(\mathbf{x})^-;$$

- **Blockage of safe points:** safe points have to be excluded from the evolution. They are characterised by $n_3 \geq 0$;

Vector Field $\mathbf{v}(\mathbf{x}) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$

Let $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$, $\hat{\mathbf{n}} = (n_1, n_2, n_3) \in \mathbb{R}^3$ and $z_{\min}$ and $z_{\max}$ be the minimum and the maximum height of the object, respectively.

- **Positivity:** we need to guarantee that $\Omega_0 \subseteq \Omega_t \subset \Omega_*$ for all $t > 0$. Thus, we impose $\widetilde{v}(\mathbf{x}) \geq 0$. Moreover, $\widetilde{v} = 0$ when $z \leq z_{\min}$;
- **Unprintable points:** we consider the term

$$v_1(\hat{\mathbf{n}}; \bar{\alpha}) := \left(\cos(\theta(\hat{\mathbf{n}})) - \cos(\bar{\alpha})\right)^+$$

  which lets the unprintable points move outward and stops them as far as they become printable;
- **Lower the faster:** the unprintable lower points have to move faster respect to the higher ones. We introduce a term $(z_{\max} - z)$;
- **Modifiable points:** if necessary, modifiable are moved outward by the curvature:

$$v_2(\mathbf{x}) := k(\mathbf{x})^-;$$

- **Blockage of safe points:** safe points have to be excluded from the evolution. They are characterised by $n_3 \geq 0$;

Vector Field $\mathbf{v}(\mathbf{x}) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$

Let $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$, $\hat{\mathbf{n}} = (n_1, n_2, n_3) \in \mathbb{R}^3$ and $z_{\min}$ and $z_{\max}$ be the minimum and the maximum height of the object, respectively.

- **Positivity:** we need to guarantee that $\Omega_0 \subseteq \Omega_t \subset \Omega_*$ for all $t > 0$. Thus, we impose $\widetilde{v}(\mathbf{x}) \geq 0$. Moreover, $\widetilde{v} = 0$ when $z \leq z_{\min}$;
- **Unprintable points:** we consider the term

$$v_1(\hat{\mathbf{n}}; \bar{\alpha}) := \left(\cos(\theta(\hat{\mathbf{n}})) - \cos(\bar{\alpha})\right)^+$$

  which lets the unprintable points move outward and stops them as far as they become printable;
- **Lower the faster:** the unprintable lower points have to move faster respect to the higher ones. We introduce a term $(z_{\max} - z)$;
- **Modifiable points:** if necessary, modifiable are moved outward by the curvature:

$$v_2(\mathbf{x}) := k(\mathbf{x})^-;$$

- **Blockage of safe points:** safe points have to be excluded from the evolution. They are characterised by $n_3 \geq 0$;

Vector Field $\mathbf{v}(\mathbf{x}) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$

Let $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$, $\hat{\mathbf{n}} = (n_1, n_2, n_3) \in \mathbb{R}^3$ and $z_{\min}$ and $z_{\max}$ be the minimum and the maximum height of the object, respectively.

- **Positivity:** we need to guarantee that $\Omega_0 \subseteq \Omega_t \subset \Omega_*$ for all $t > 0$. Thus, we impose $\widetilde{v}(\mathbf{x}) \geq 0$. Moreover, $\widetilde{v} = 0$ when $z \leq z_{\min}$;
- **Unprintable points:** we consider the term

$$v_1(\hat{\mathbf{n}}; \bar{\alpha}) := \left(\cos(\theta(\hat{\mathbf{n}})) - \cos(\bar{\alpha})\right)^+$$

  which lets the unprintable points move outward and stops them as far as they become printable;
- **Lower the faster:** the unprintable lower points have to move faster respect to the higher ones. We introduce a term $(z_{\max} - z)$;
- **Modifiable points:** if necessary, modifiable are moved outward by the curvature:

$$v_2(\mathbf{x}) := k(\mathbf{x})^-;$$

- **Blockage of safe points:** safe points have to be excluded from the evolution. They are characterised by $n_3 \geq 0$;

Vector Field $\mathbf{v}(\mathbf{x}) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$ (cont.)

By putting all the together we define the vector field $\mathbf{v}(\mathbf{x}) := \widetilde{v}(\mathbf{x})\hat{\mathbf{n}}$ where

$$\widetilde{v}(\mathbf{x}) = \widetilde{v}(\mathbf{x}, \hat{\mathbf{n}}, k; \bar{\alpha}) := \begin{cases} C_1(z_{\max} - z)v_1(\hat{\mathbf{n}}; \bar{\alpha}) + C_2 v_2(\mathbf{x}) & n_3 < 0 \text{ and } z > z_{\min} \\ 0 & \text{otherwise,} \end{cases}$$

and $C_1, C_2 > 0$ (models parameters).

▷ The velocity $\widetilde{v}(\mathbf{x})$ has to be computed for each $t \geq 0$;

▷ By changing C_2, it is decided the (smoothing) contribution given by the movement of the curvature;

▷ The evolution has to be stopped as soon as all surface's points are printable (i.e., safe or modifiable); then, the final object $\Omega_* \supset \Omega_0$ is achieved.

▷ By construction, the surface will always evolve towards a fully-printable object.

Outline

1 **Additive Manufacturing or 3D printing**
- FDM technology
- Main issues in 3D printing

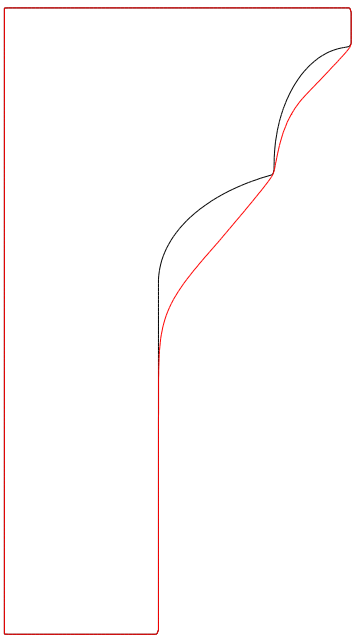
2 **The Level-Set method**
- Tracking an evolving surface
- Motions along normal direction
- Hamilton-Jacobi equations

3 **Using the Level-Set method for fixing overhangs**
- Idea and scope
- Printability
- Construction of the vector field

4 **Numerical tests**

From 2D to 3D via extrusion

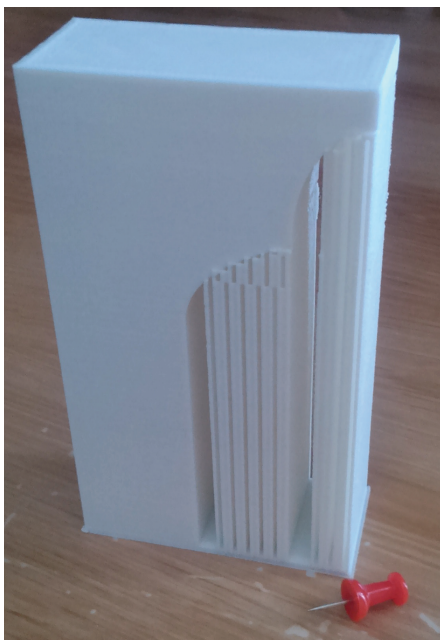
▷ (a) Initial contour Σ_0 (black) and the optimised one Σ_* (red) after the evolution;
▷ (b) Printed model with the proposed support;
▷ (c) Printed model with the support generated by Cura v15.04.2;
▷ (d) Tree-like support generated by Autodesk Meshmixer v2.9.



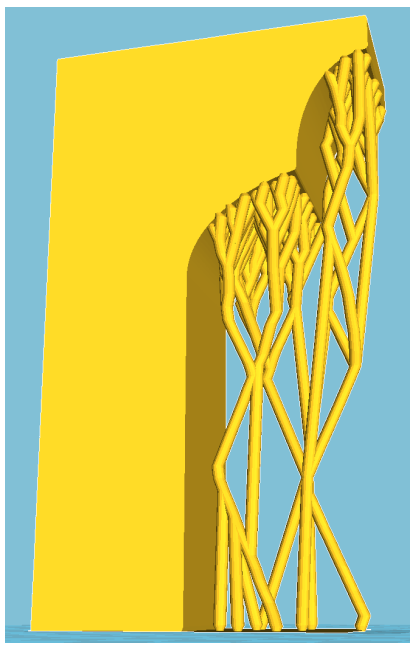
(a)



(b)



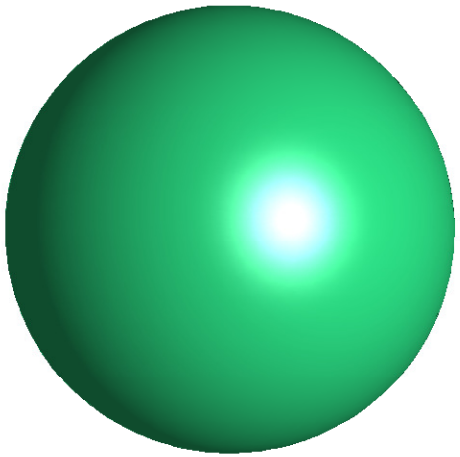
(c)

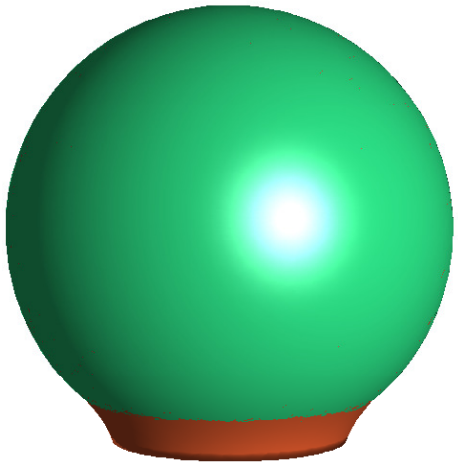


(d)

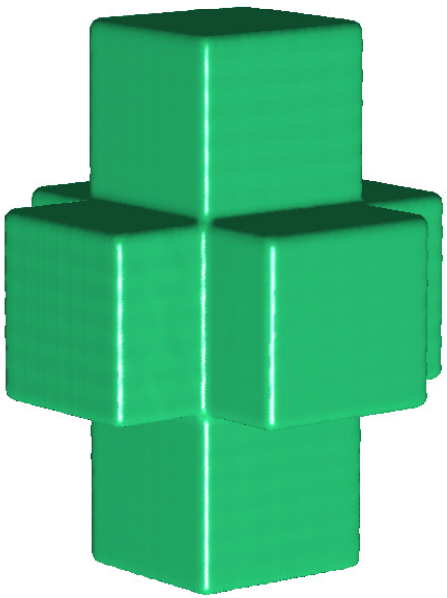
▷ Printed by a **sharebot NG**; PLA filament (1.75mm); slice resolution 0.2mm.

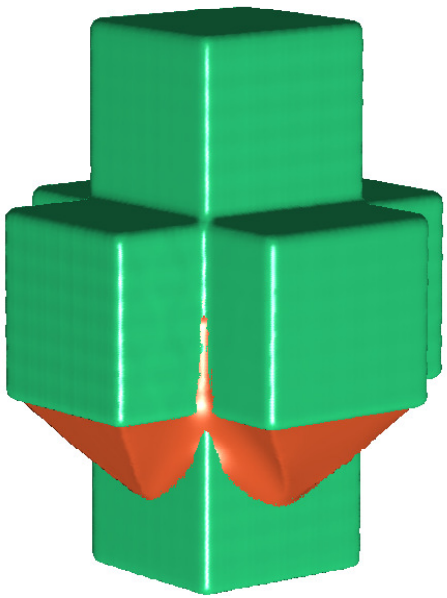
Simple examples: a sphere and a cross





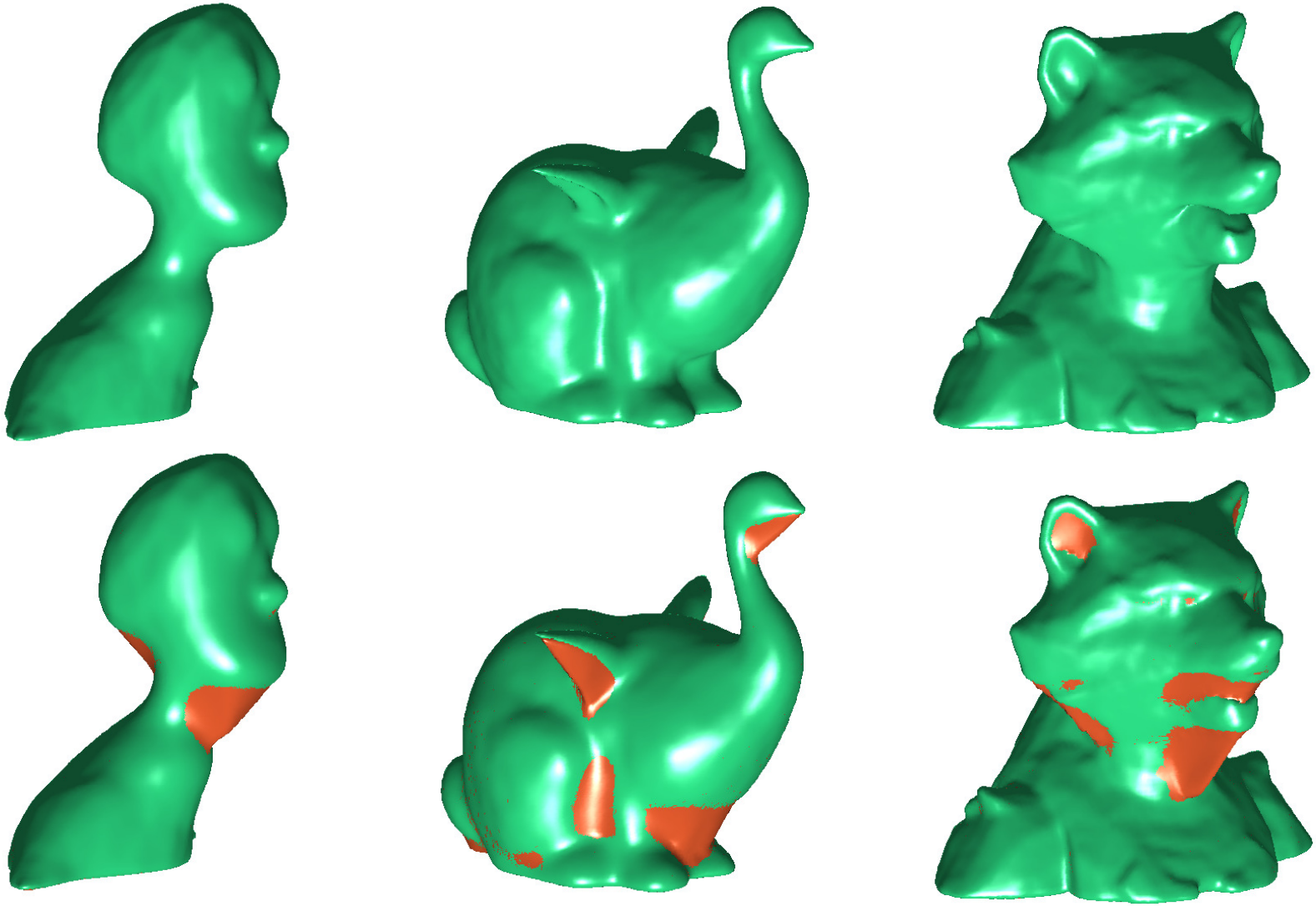
$$C_1 = 0.7$$
$$C_2 = 0.3$$





$$C_1 = 1.5$$
$$C_2 = 0.5$$

Funny examples: gremlin, bunny-goose, and raccoon

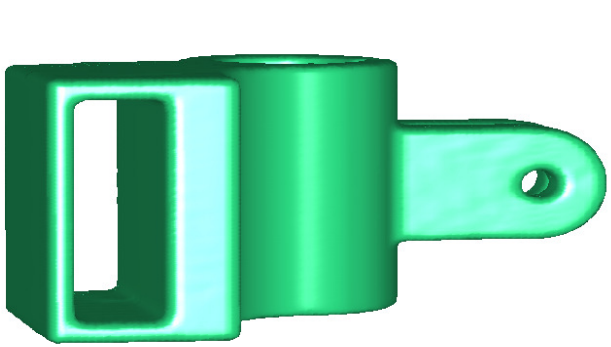
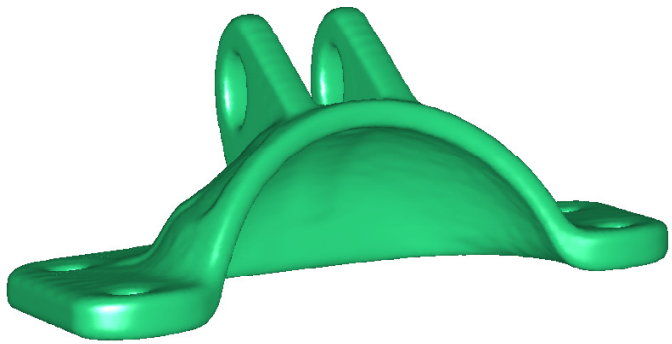
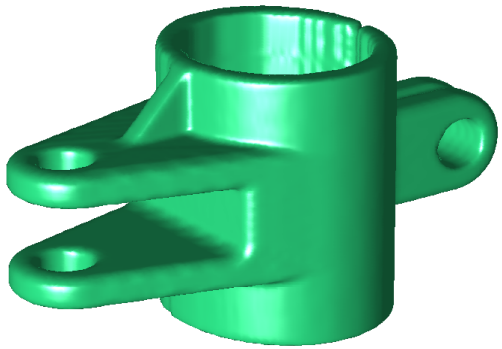
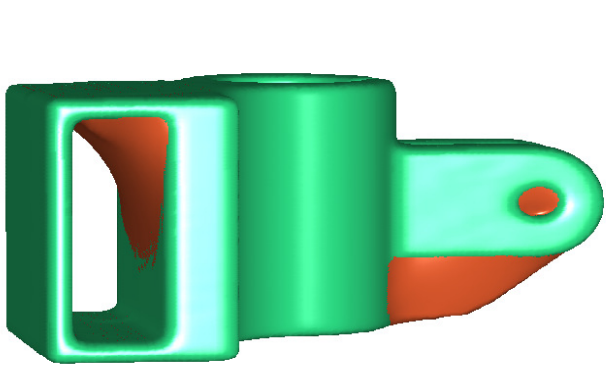
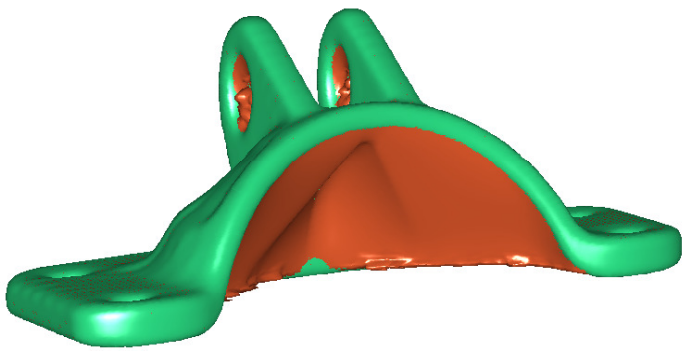
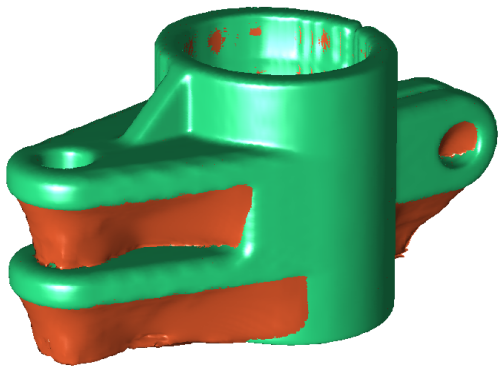


$C_1 = 0.8$
$C_2 = 0.8$

$C_1 = 1.8$
$C_2 = 0.6$

$C_1 = 0.6$
$C_2 = 1.2$

Serious examples: brackets

$C_1 = 3.5$
$C_2 = 0.9$

$C_1 = 1.2$
$C_2 = 1.6$

$C_1 = 5.6$
$C_2 = 0.3$