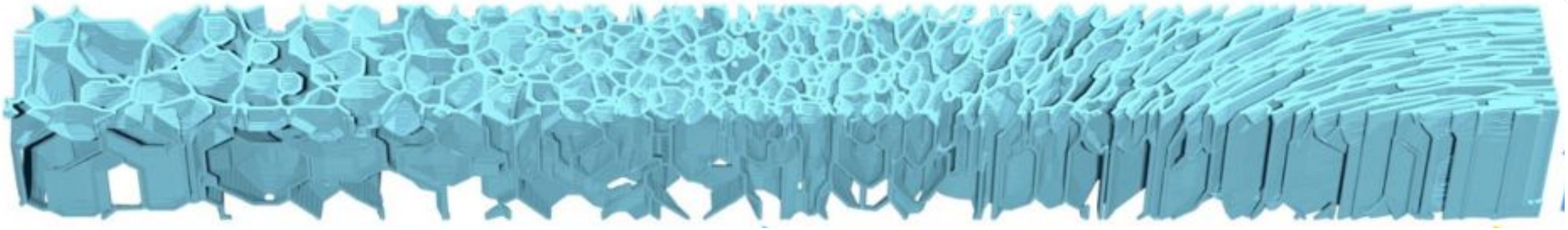


VU Computational Aspects of Digital Fabrication

3D Printable Microstructures from Polyhedral Voronoi Diagrams

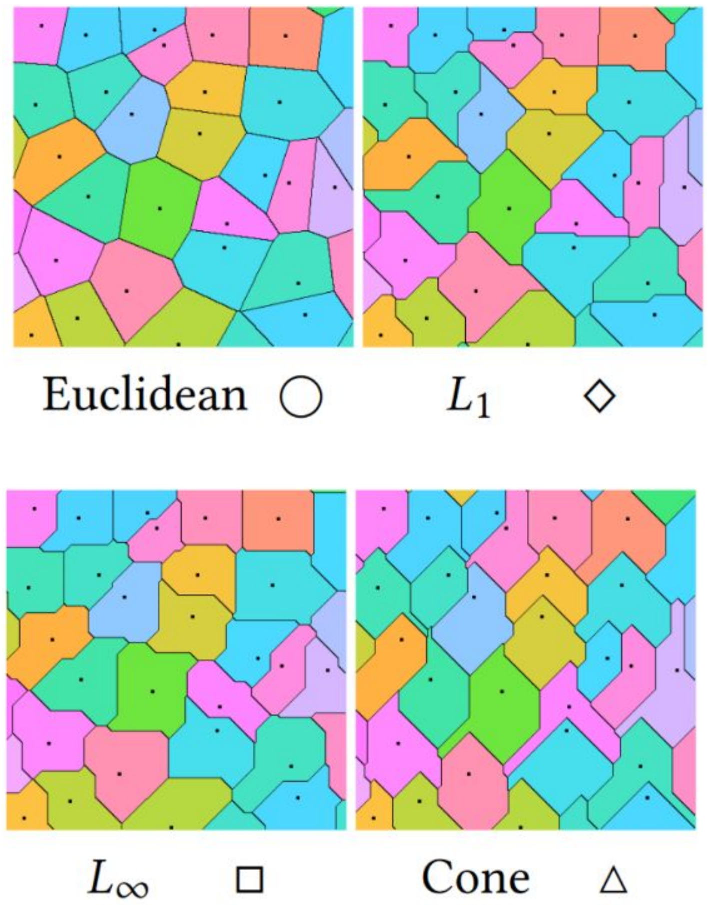
Jonas Prohaska / 01449302

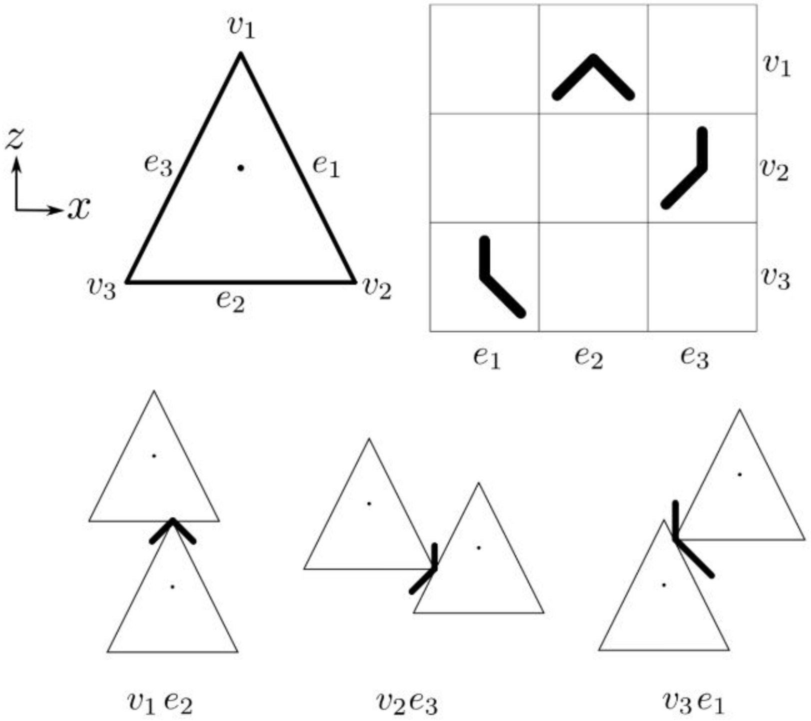
Marcel Jira / 11909464

Institute of Visual Computing & Human-Centered Technology, TU Wien, Austria

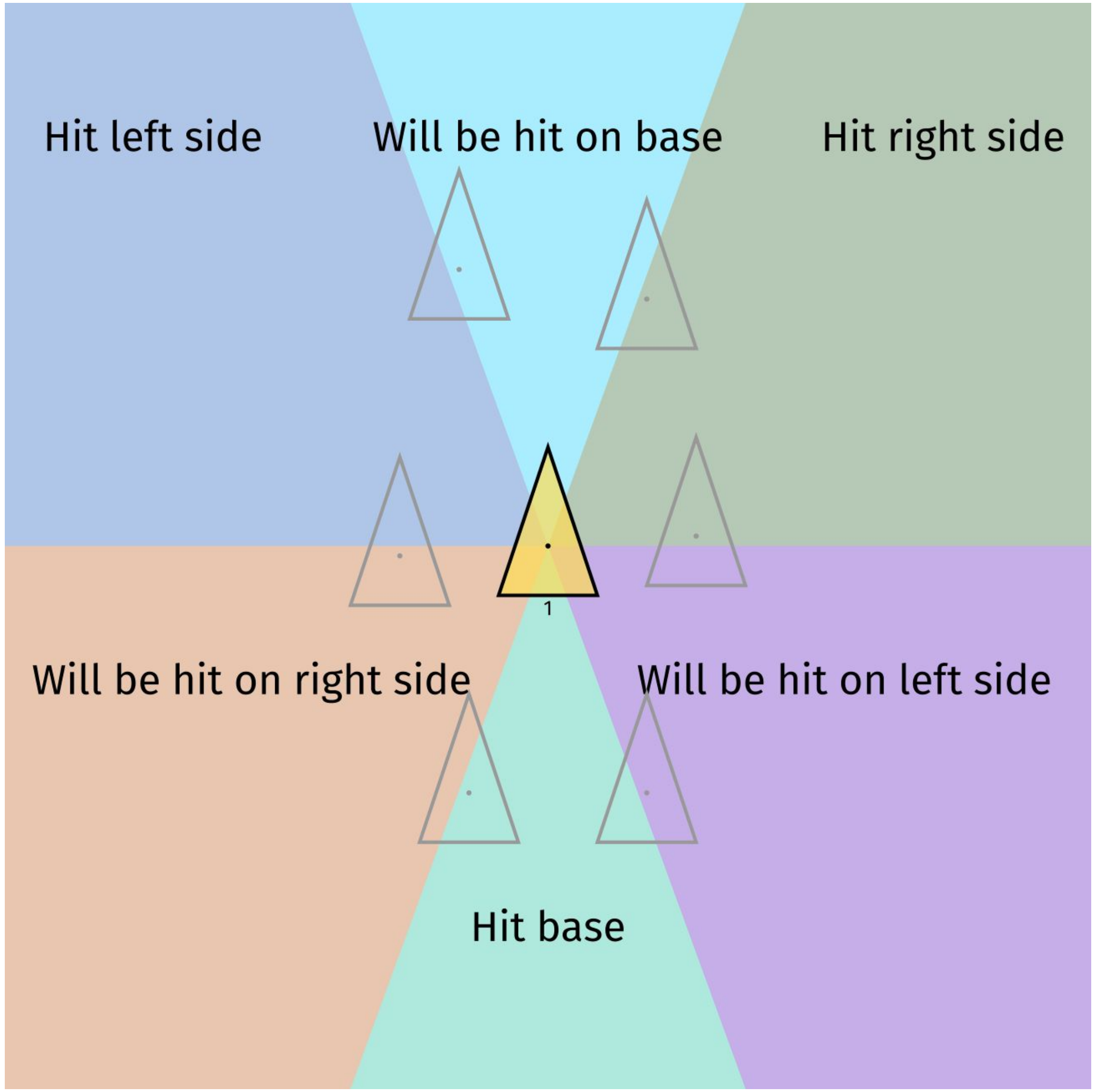
Idea (2D)

- Jonàs Martínez et al.: Polyhedral Voronoi diagrams for additive manufacturing

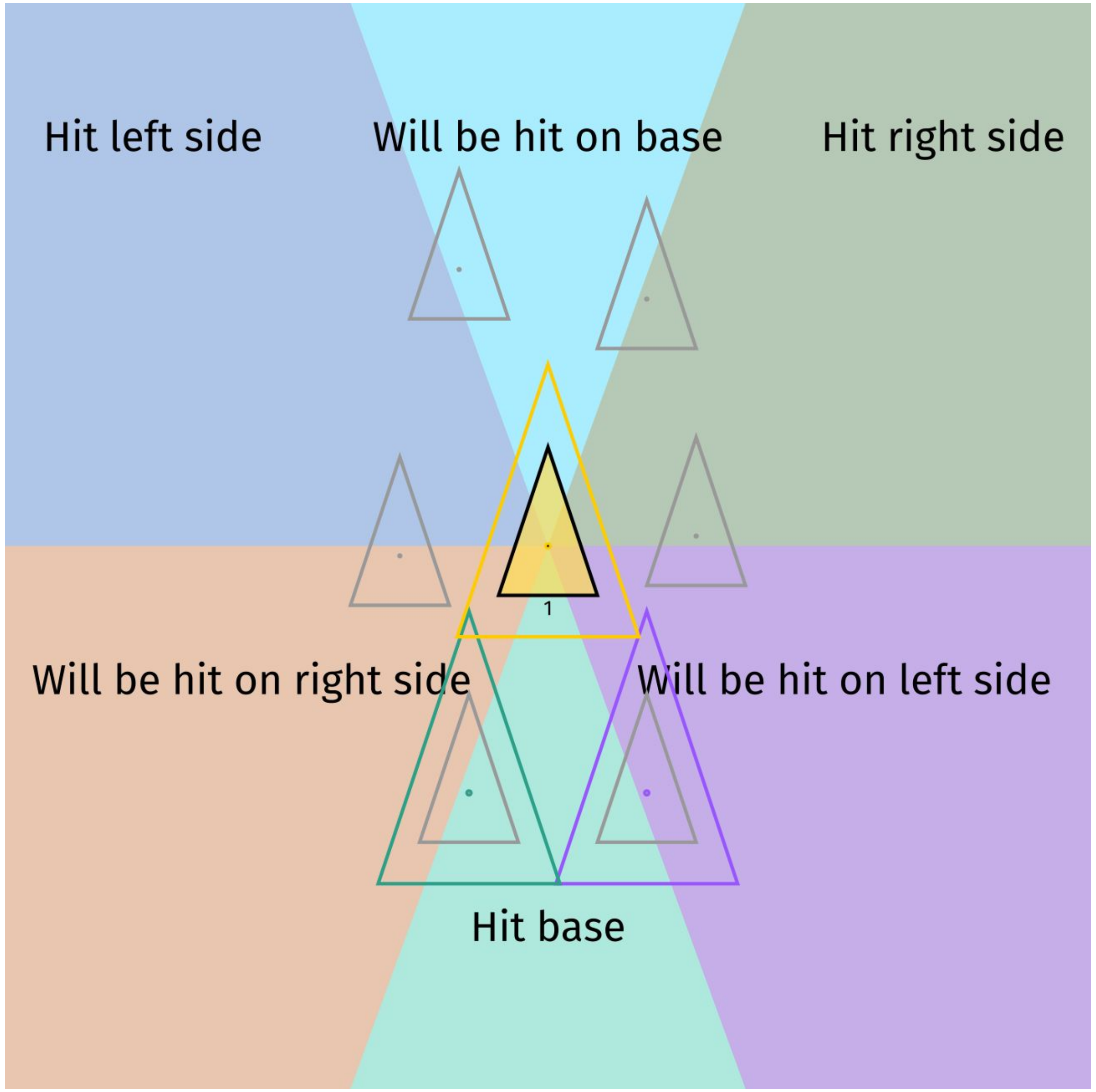




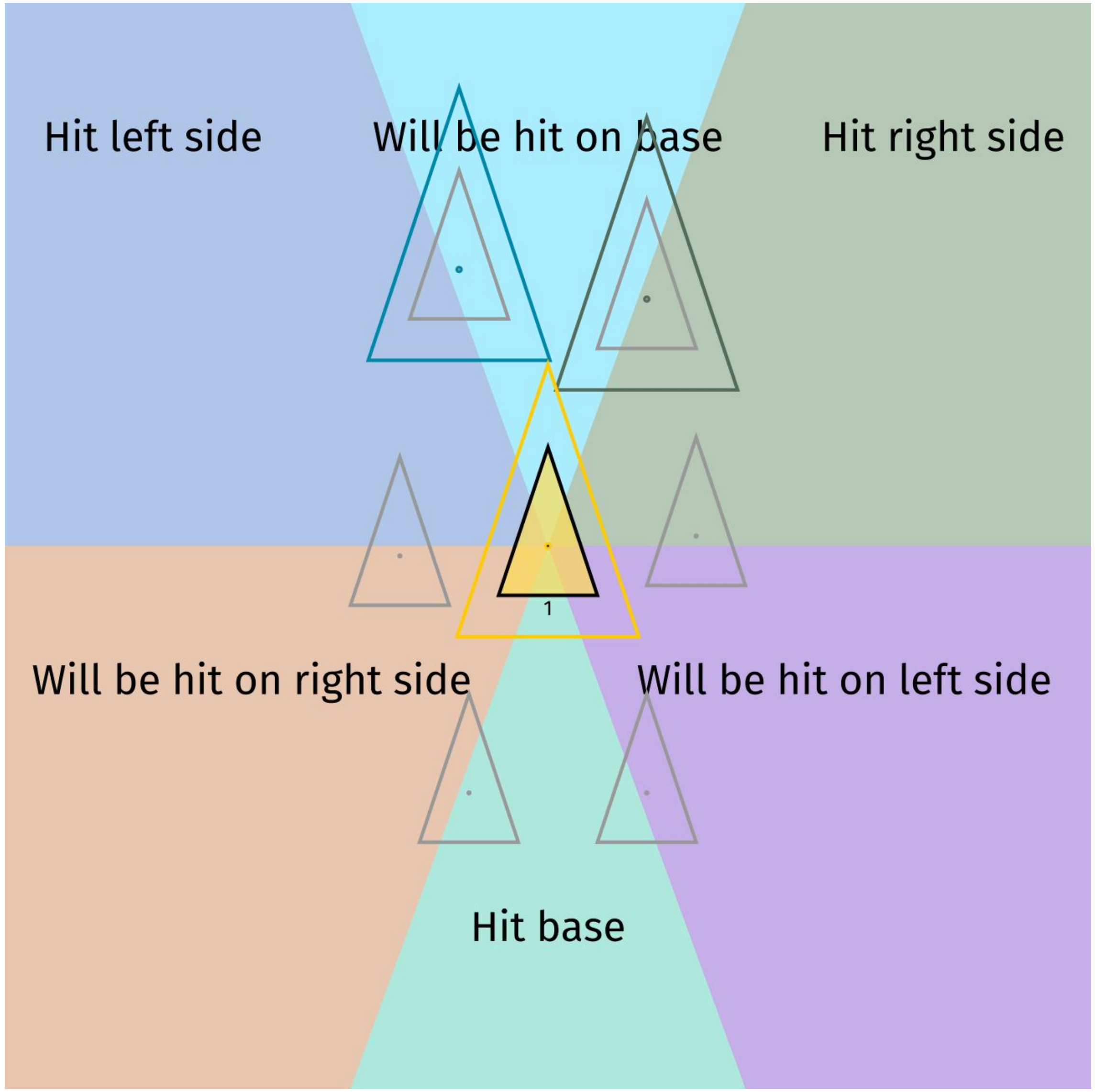
Collision Planning: HOW will the Triangles collide?



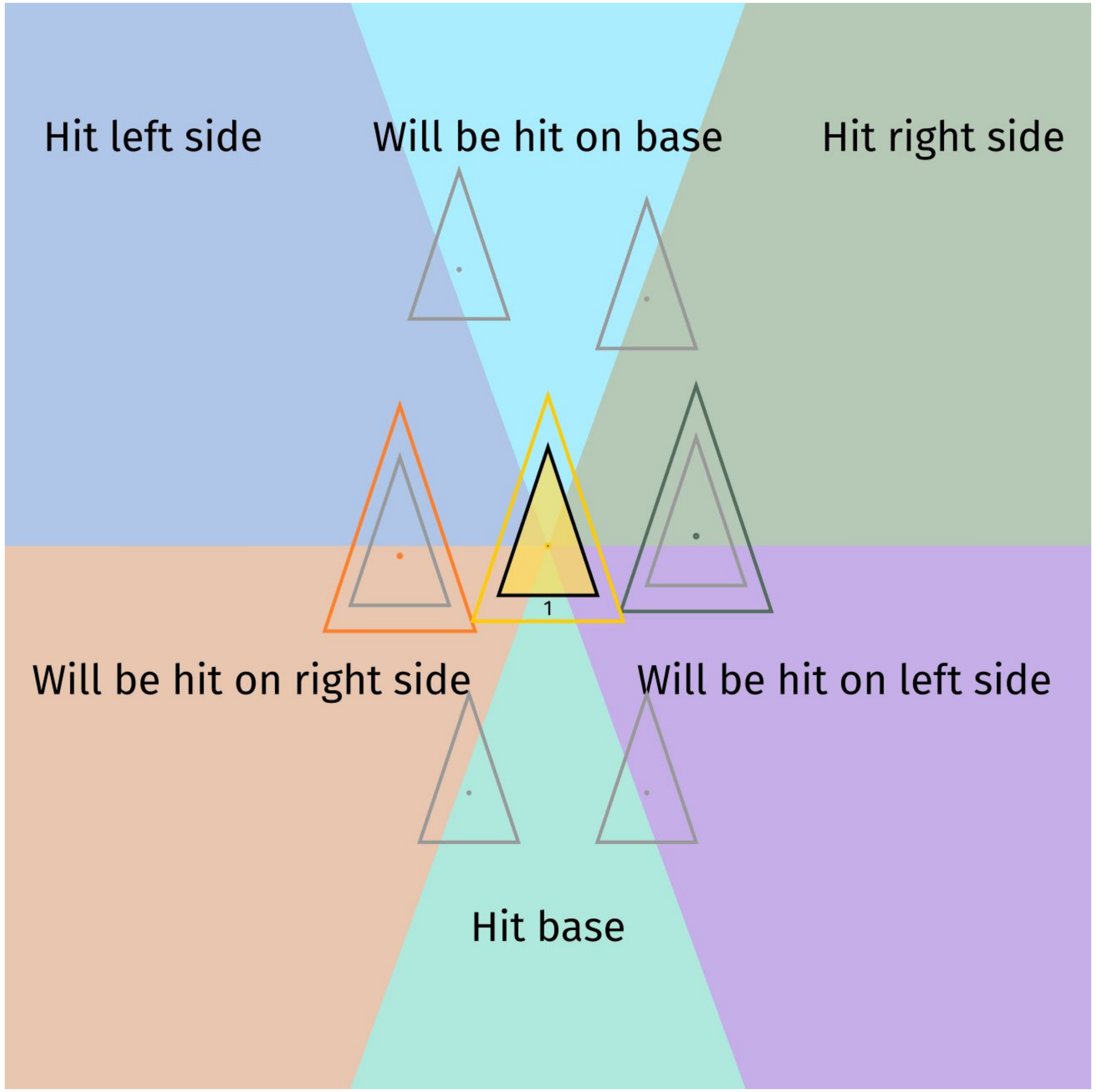
Collision Planning: HOW will the Triangles collide?



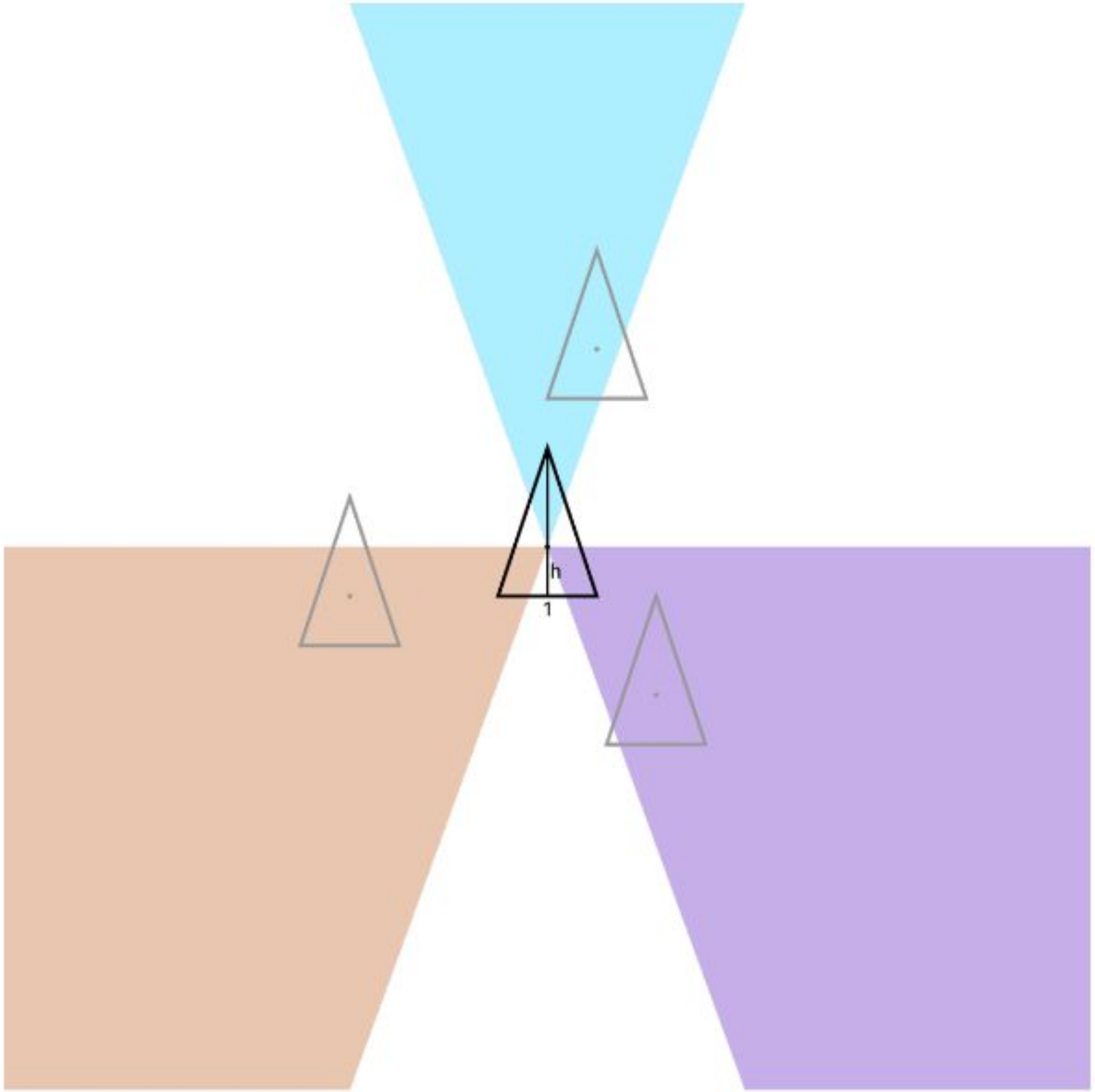
Collision Planning: HOW will the Triangles collide?



Collision Planning: HOW will the Triangles collide?

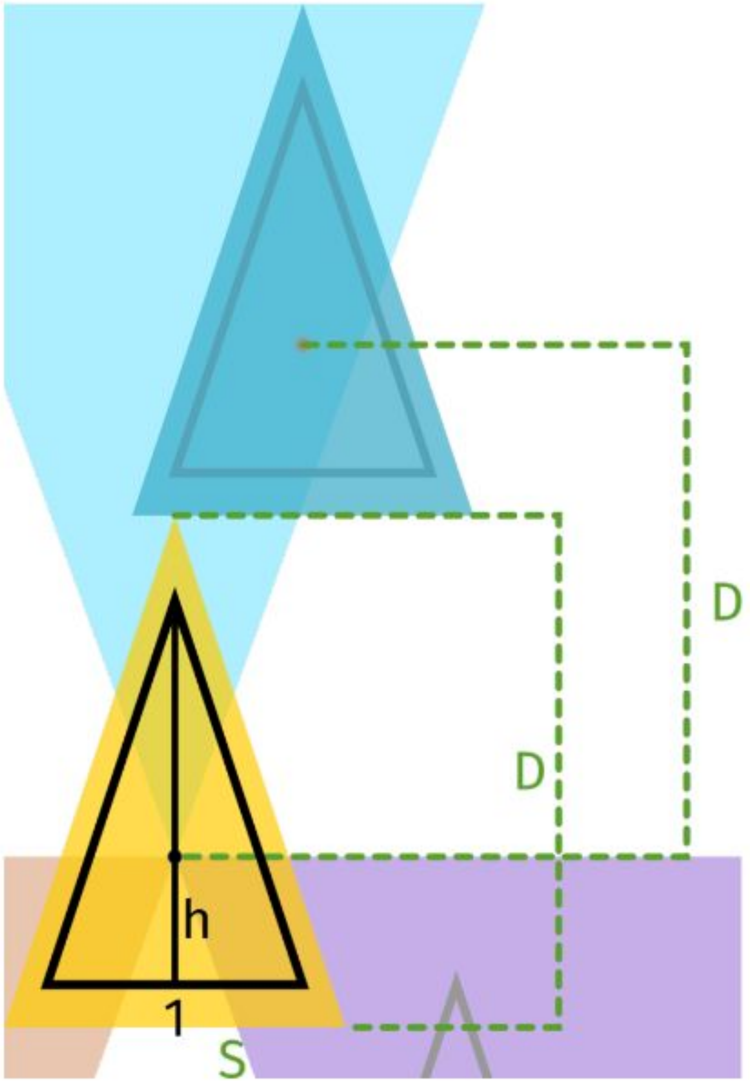


Collision Planning: WHERE will the Triangles collide?



Every triangle hits 3 other triangles
(or a bounding wall)

Collision Planning: WHERE will the Triangles collide?

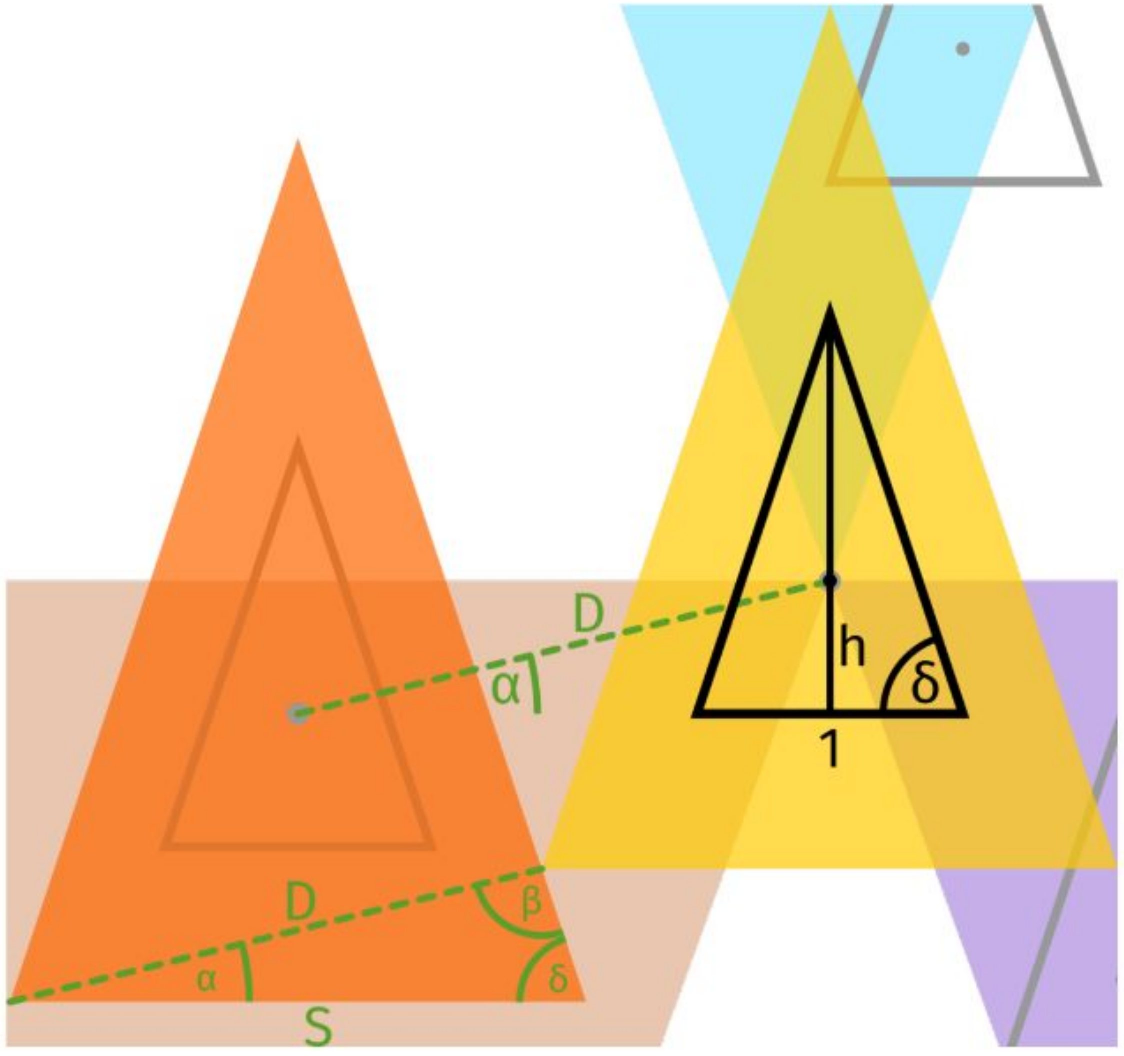


Base Hit Scale

$$\frac{D}{h} = \frac{S}{1}$$

$$S = \frac{D}{h}$$

Collision Planning: WHERE will the Triangles collide?



Side Hit Scale

$$\beta = 180 - (\alpha + \delta)$$

Law of Sines

$$\frac{D}{\sin\delta} = \frac{S}{\sin\beta}$$

$$S = \frac{D}{\sin\delta} \cdot \sin\beta$$

$$S = \frac{D}{\sin\delta} \cdot \sin[180 - (\alpha + \delta)]$$

Jonas Prohaska & Marcel Jira

Strategy to find collisions

- Create random points stored in a list points
- Create two lists of points
- For base = 1, find h and δ
  - pointsx = points sorted by x
  - pointsy = points sorted by y

Strategy to find collisions

- 4) Set currentScale = 1
- 5) In a loop, do the following
  - 1) Find minimal scale > currentScale for two triangles to collide, respecting
    - 1) Hitted triangle center is in hitting zone of hitting triangle center
    - 2) Hitting point has analyzed hitting zone = true
    - 3) For base hit, use use Base Hit Scale (Euclidean Distance)
    - 4) For side hit, use Side Hit Scale
  - 2) Store the following information in a list
    - Hitting point, Hitted point, Hitting side, Coordinates, (Scale factor)
  - 3) For the hitting point set respective hitting zone (top, left or right) to false
  - 4) Set currentScale = scale factor

Technology Approaches

- Tried with FreeCAD
  - Idea: Use FreeCAD sketcher tools to do the math for us
  - Pros: Works pretty good in theory, helps try out geometric ideas
  - Cons: Begins to struggle really fast, syntax is rather difficult
- Visual Debugging using a simple renderer in C++
  - Pros: Faster code, simpler math operations, custom GUI debugging
  - Cons: Building a custom renderer using C++/OpenGL
- Python using Plotly 3D graphs
  - Pros: Features 3D camera movements and tooltips out of the box, renders many thousand points flawlessly once loaded
  - Cons: Designed for plots, objects are per default distorted if too far between

Live Demo of Progress