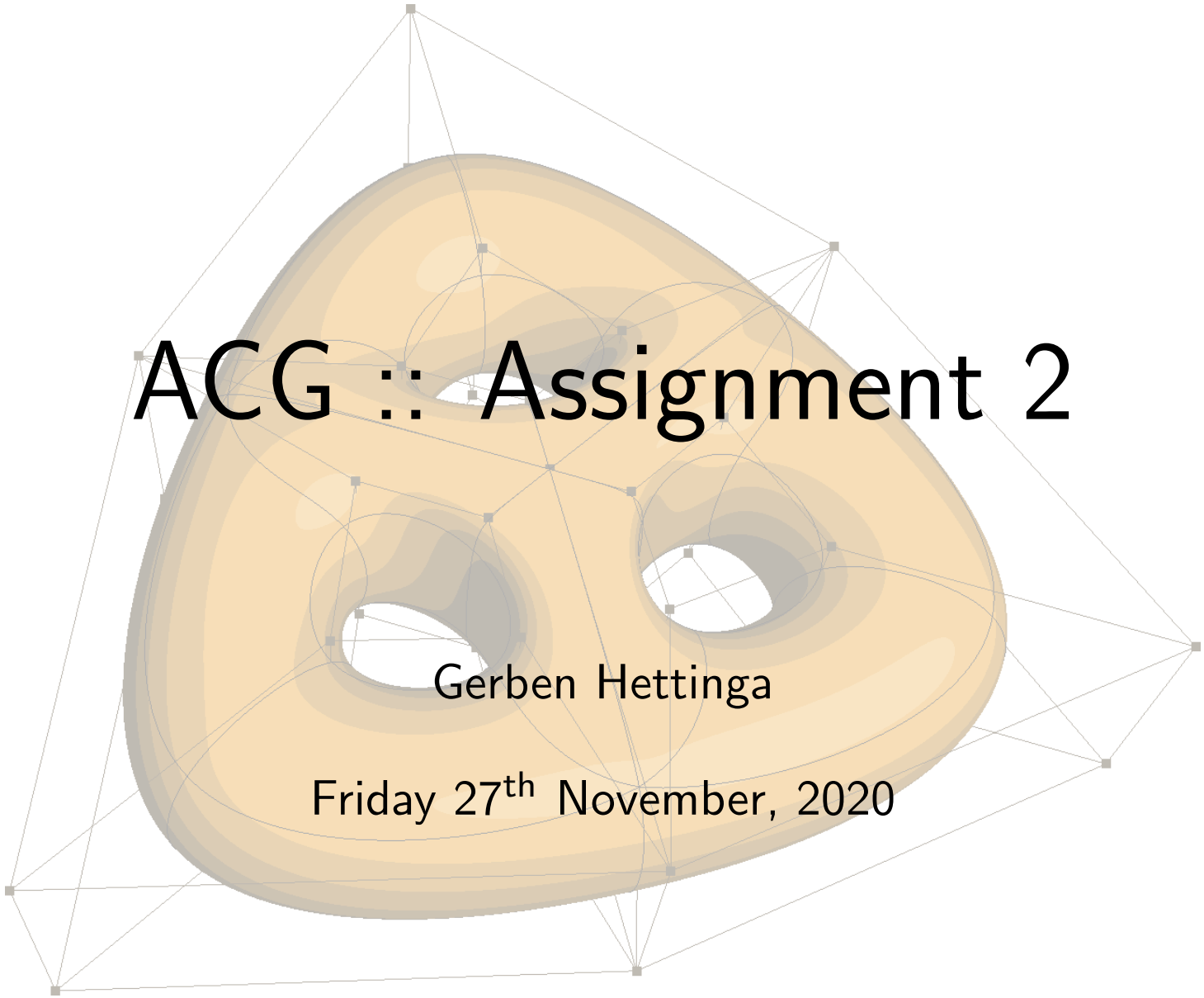


ACG :: Assignment 2

Gerben Hettinga

Friday 27th November, 2020

Previous lecture

- Questions, discussion, ...

Assignment

MA Add boundary rules for Loop subdivision

Assignment

MA Add boundary rules for Loop subdivision

MA Implement reflection lines or isophotes

- Example

Assignment

MA Add boundary rules for Loop subdivision

MA Implement reflection lines or isophotes

- Example

AF Vertex selection

Assignment

MA Add boundary rules for Loop subdivision

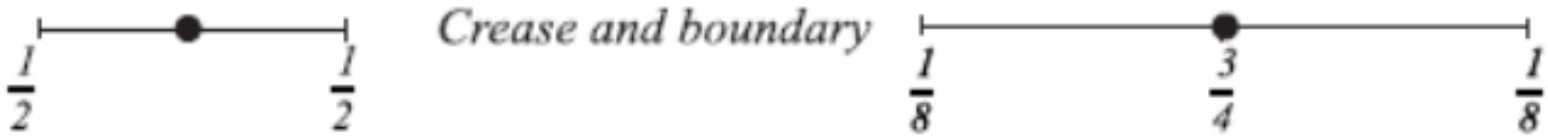
MA Implement reflection lines or isophotes

- Example

AF Vertex selection

B Gaussian curvature or edge selection

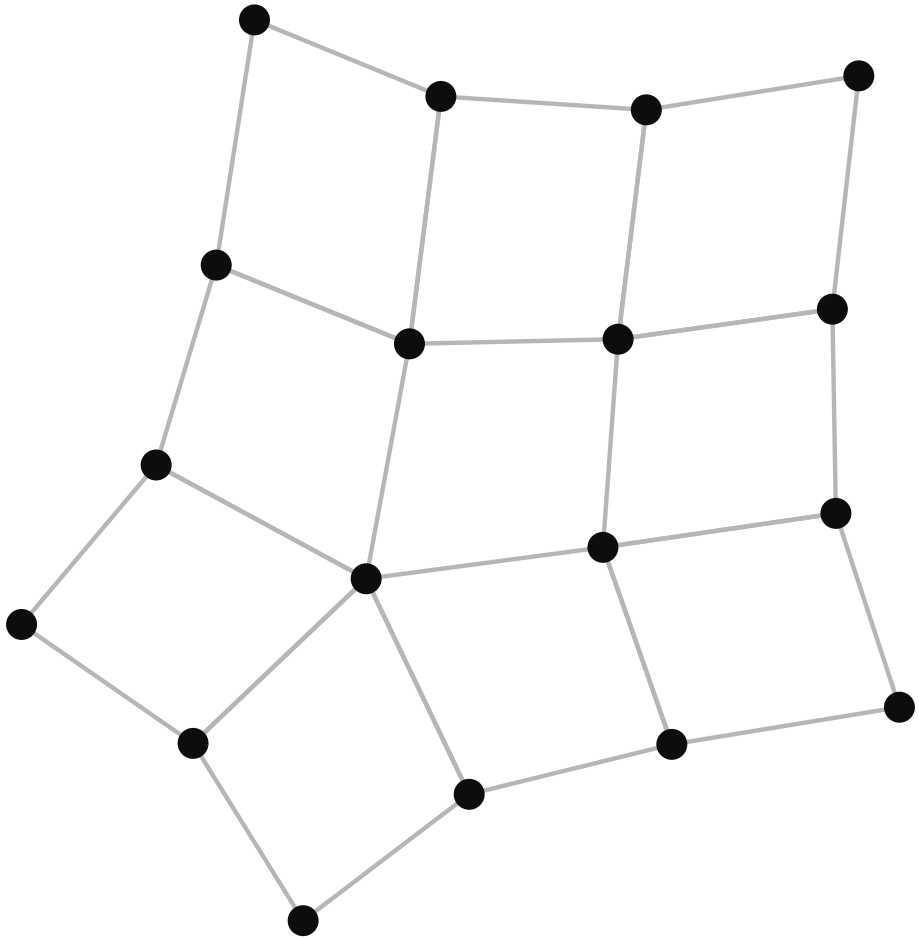
Boundary Rules



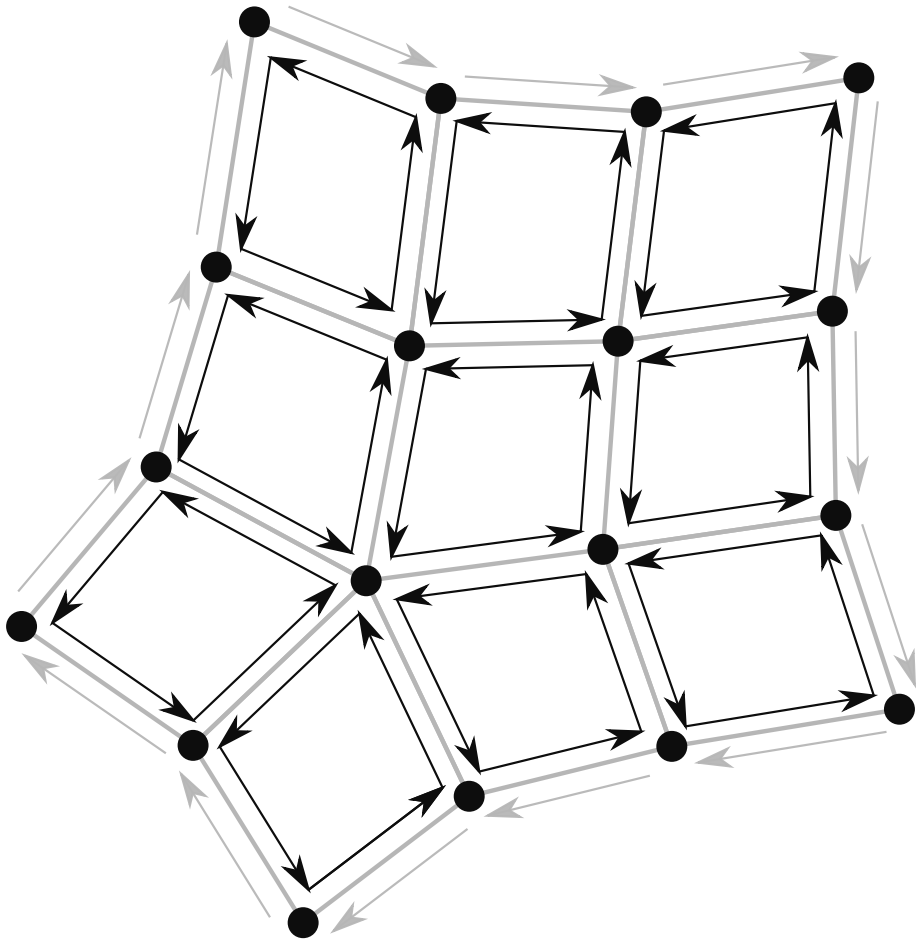
a. *Masks for odd vertices* b. *Masks for even vertices*

- The half-edge data structure is not correctly set up for meshes with a boundary!
- These are the curve rules from the last lab session

Half-Edge data structure

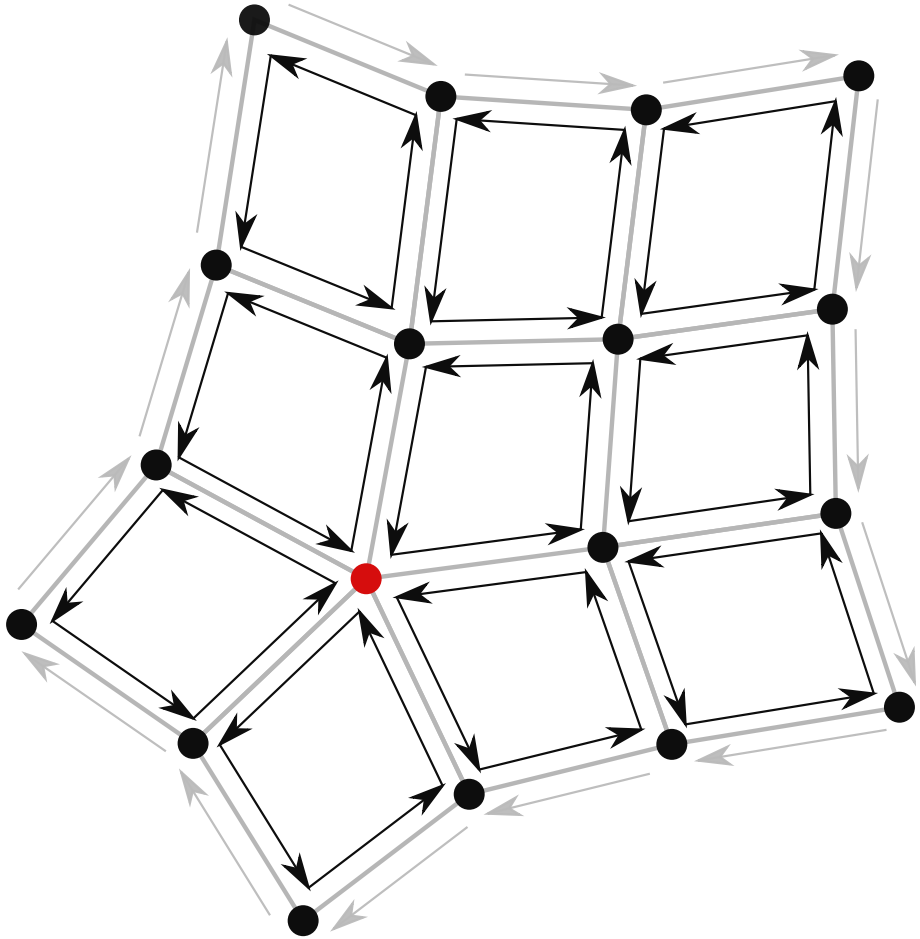


Half-Edge data structure



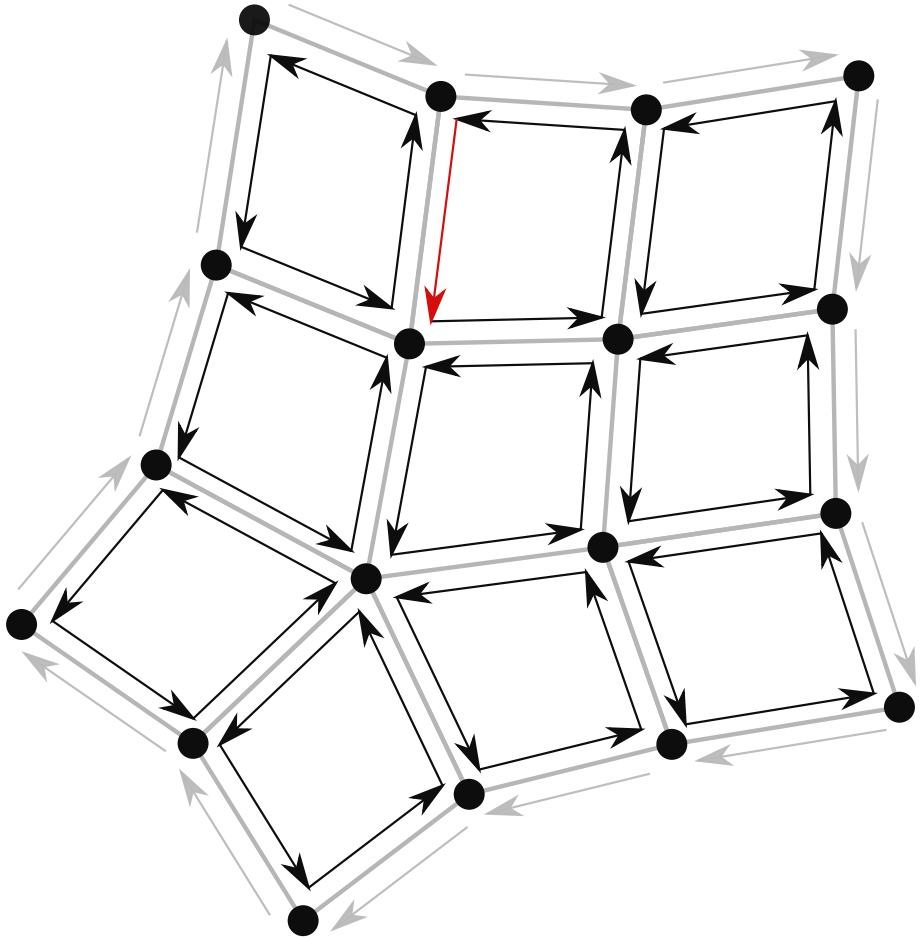
Half-Edge data structure

- Vertices



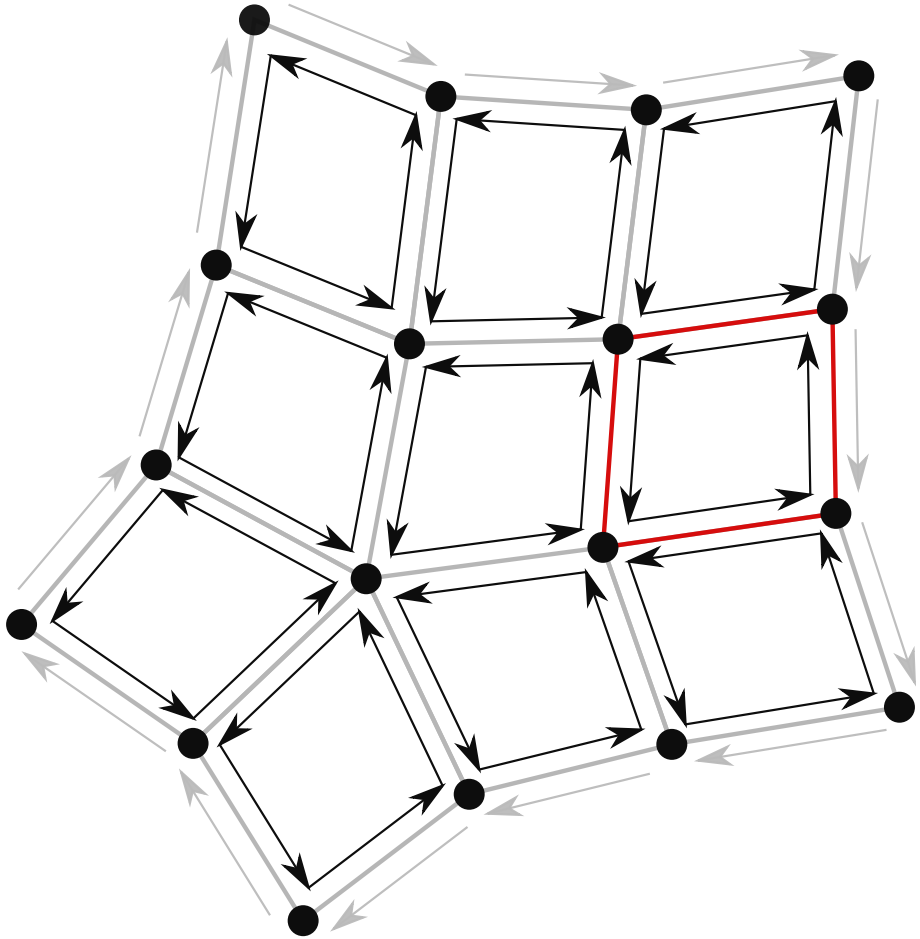
Half-Edge data structure

- Vertices
- HalfEdges



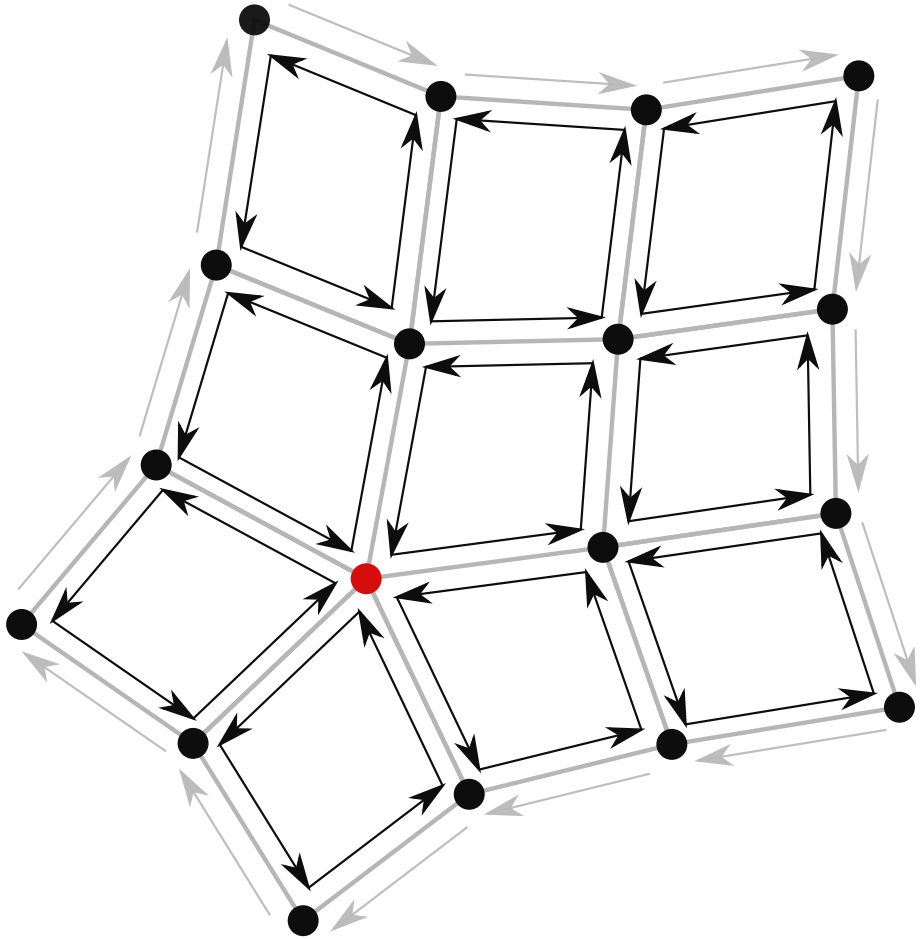
Half-Edge data structure

- Vertices
- HalfEdges
- Faces



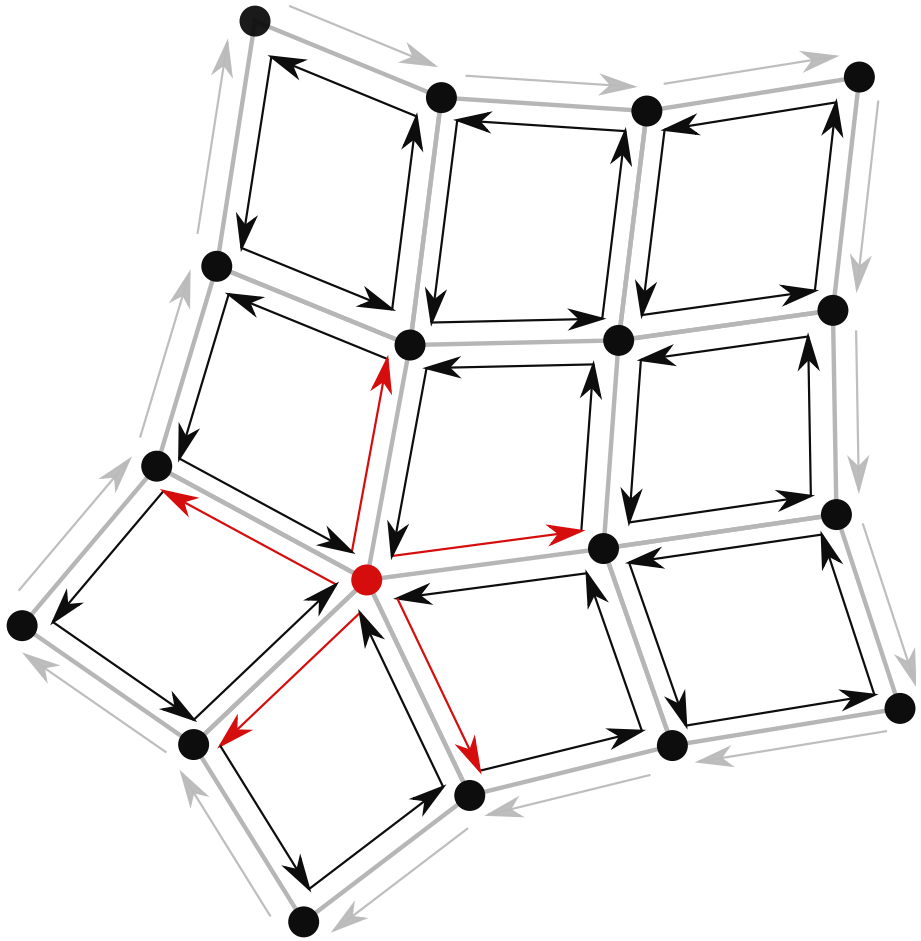
Half-Edge data structure :: Vertex

- x, y and z
- Out (HalfEdge*)
- Valency



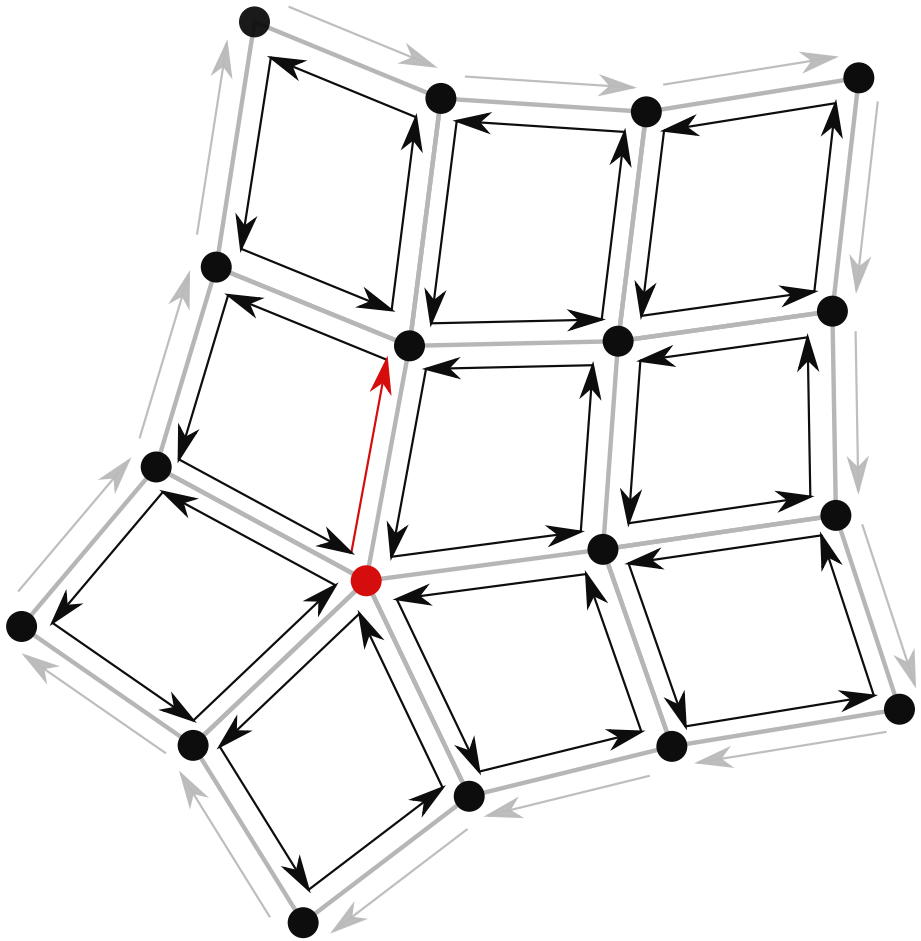
Half-Edge data structure :: Vertex

- x, y and z
- Out (HalfEdge*)
- Valency



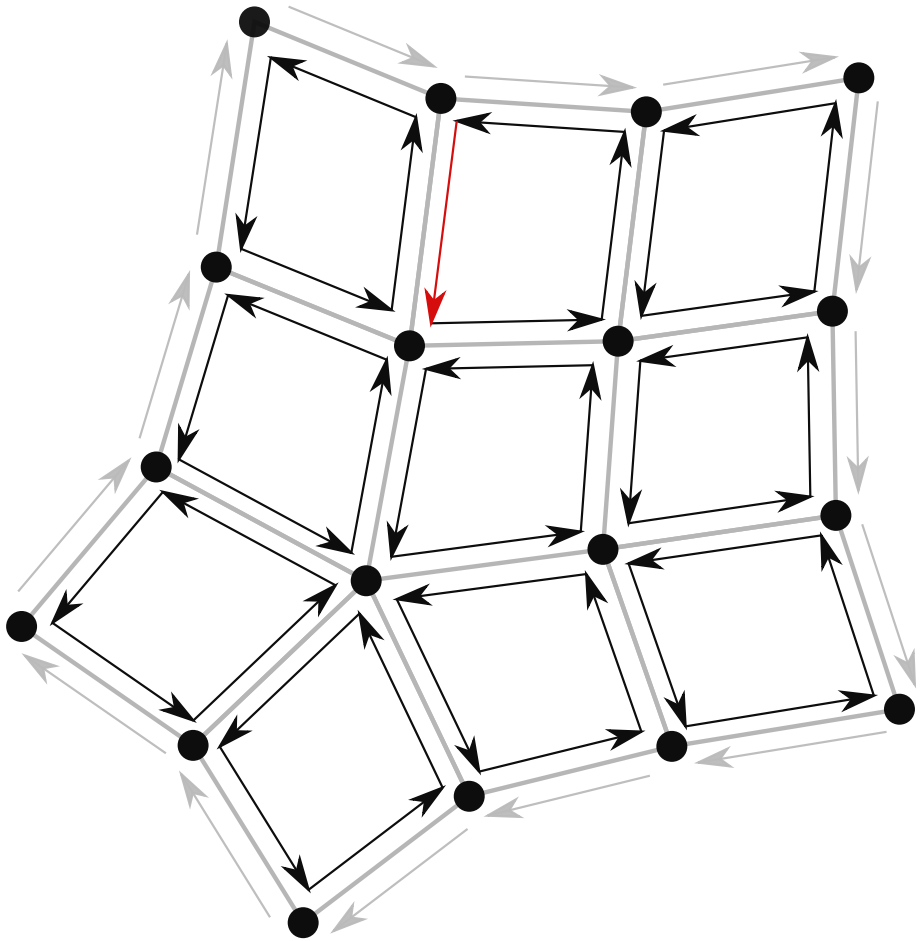
Half-Edge data structure :: Vertex

- x, y and z
- Out (HalfEdge*)
- Valency



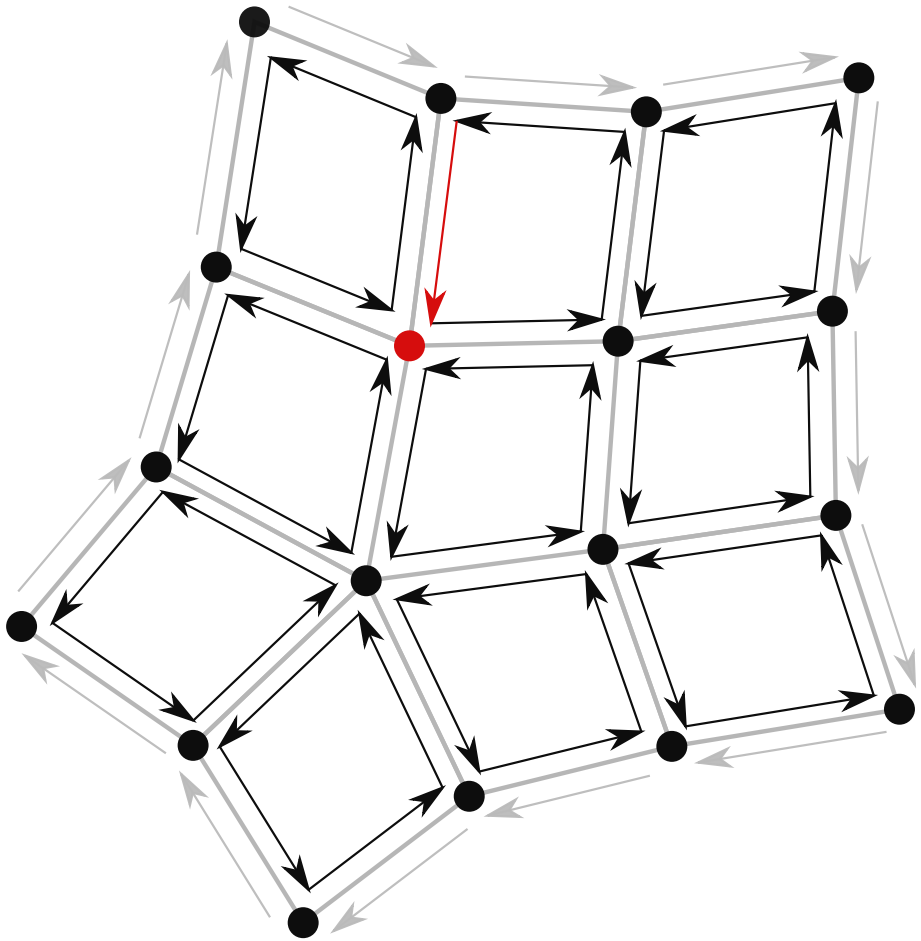
Half-Edge data structure :: HalfEdge

- Target (Vertex*)
- Next (HalfEdge*)
- Prev (HalfEdge*)
- Twin (HalfEdge*)
- Polygon (Face*)



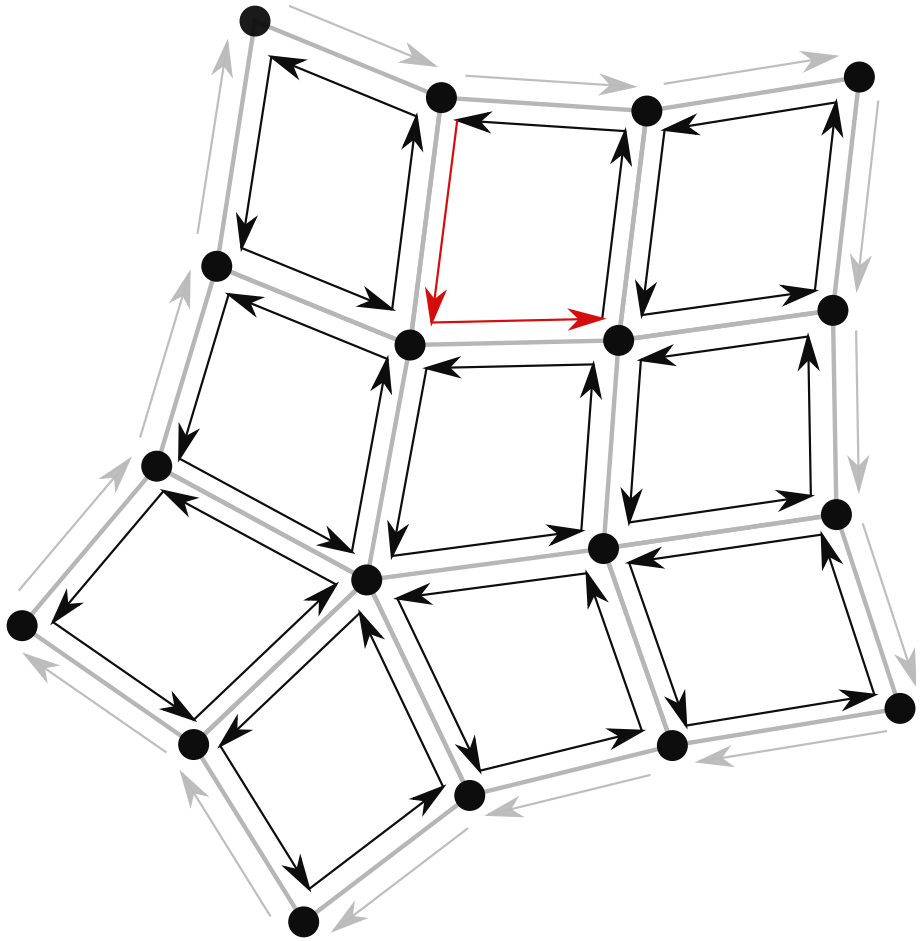
Half-Edge data structure :: HalfEdge

- Target (Vertex*)
- Next (HalfEdge*)
- Prev (HalfEdge*)
- Twin (HalfEdge*)
- Polygon (Face*)



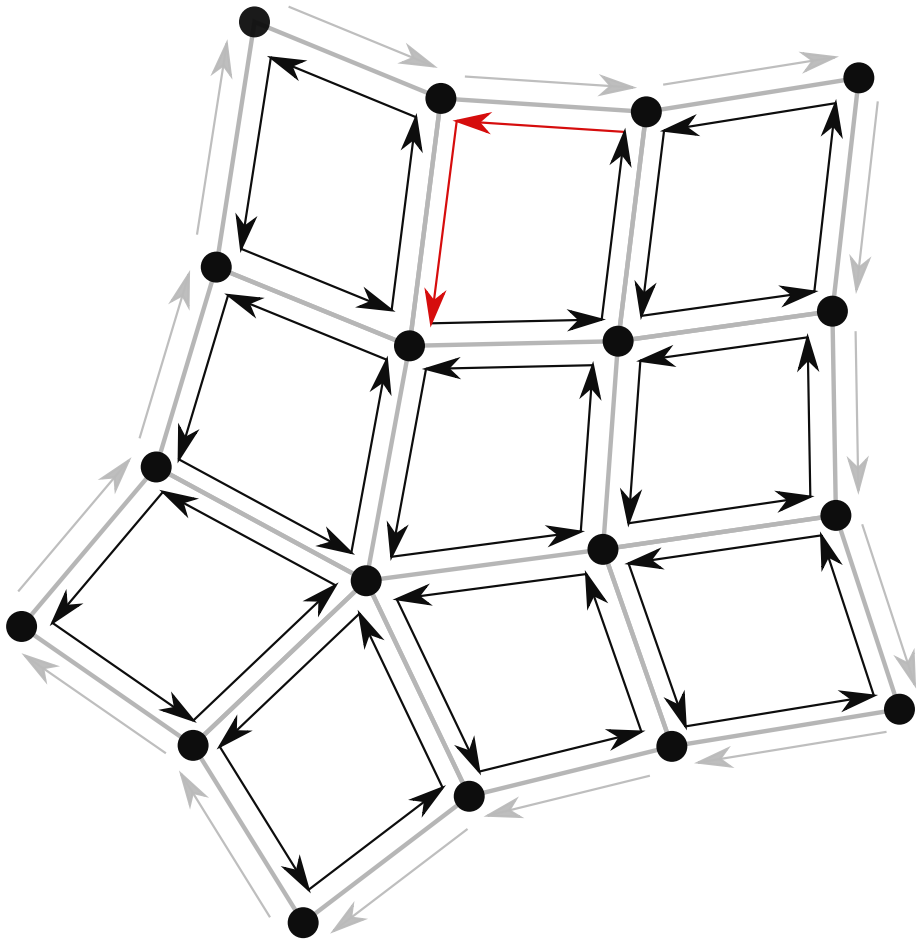
Half-Edge data structure :: HalfEdge

- Target (Vertex*)
- Next (HalfEdge*)
- Prev (HalfEdge*)
- Twin (HalfEdge*)
- Polygon (Face*)



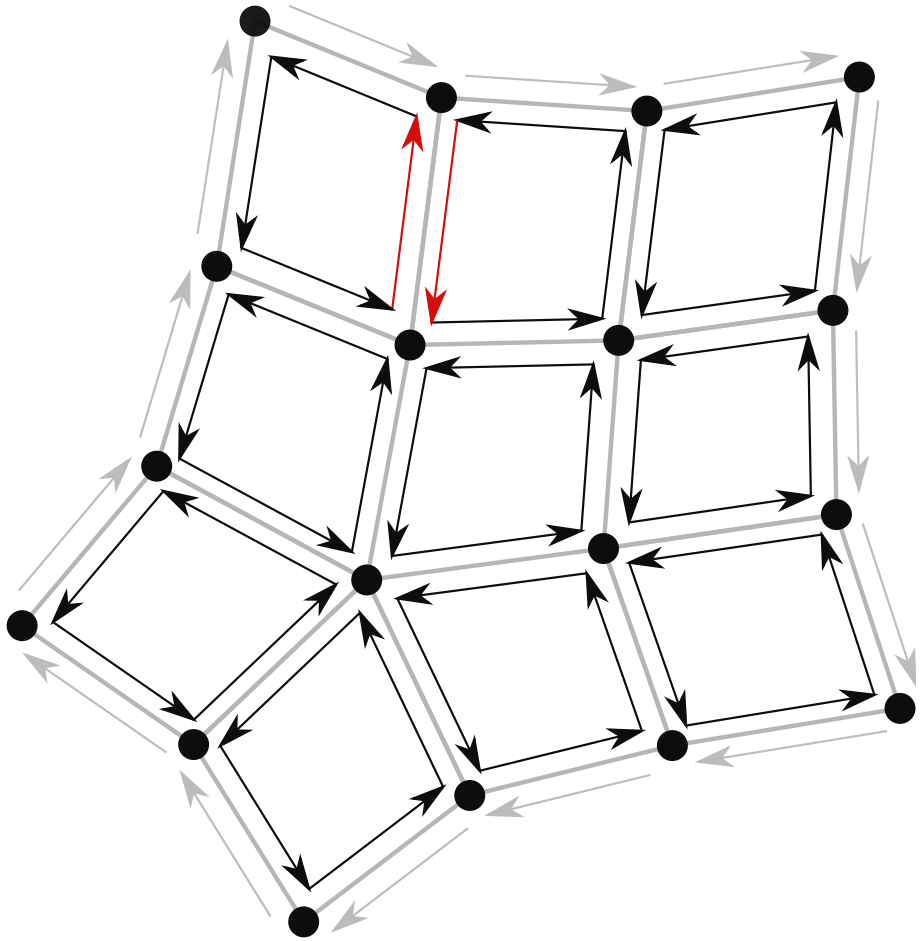
Half-Edge data structure :: HalfEdge

- Target (Vertex*)
- Next (HalfEdge*)
- Prev (HalfEdge*)
- Twin (HalfEdge*)
- Polygon (Face*)



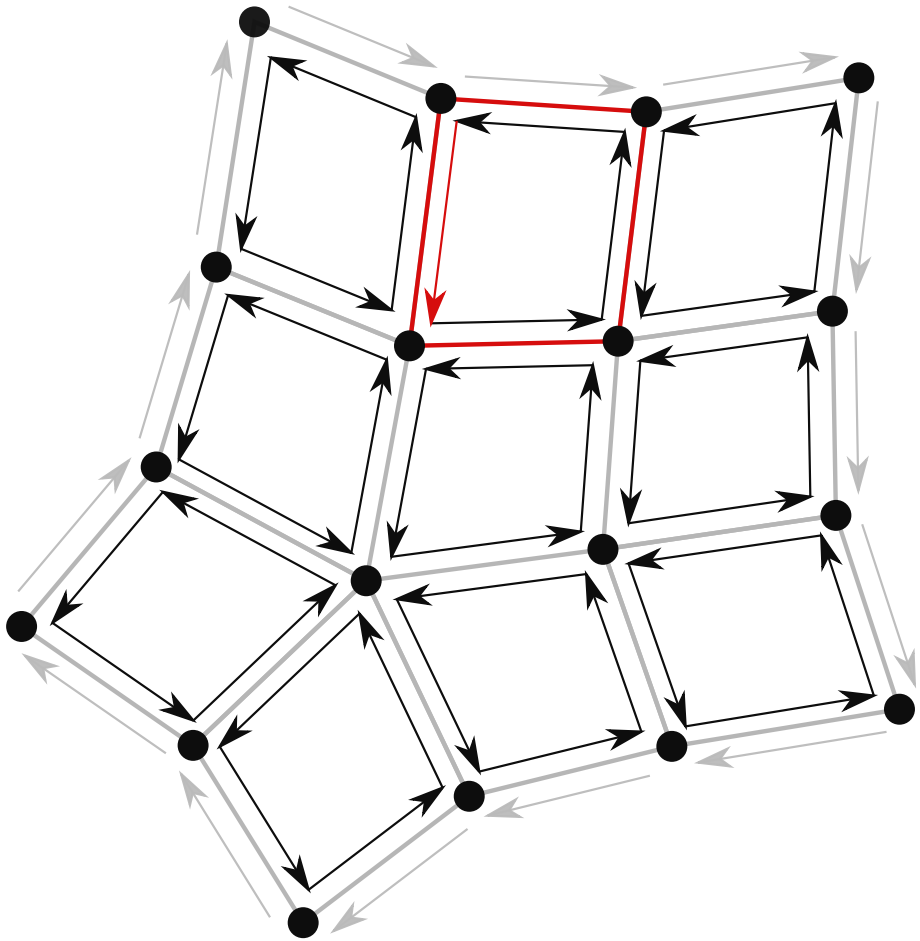
Half-Edge data structure :: HalfEdge

- Target (Vertex*)
- Next (HalfEdge*)
- Prev (HalfEdge*)
- Twin (HalfEdge*)
- Polygon (Face*)



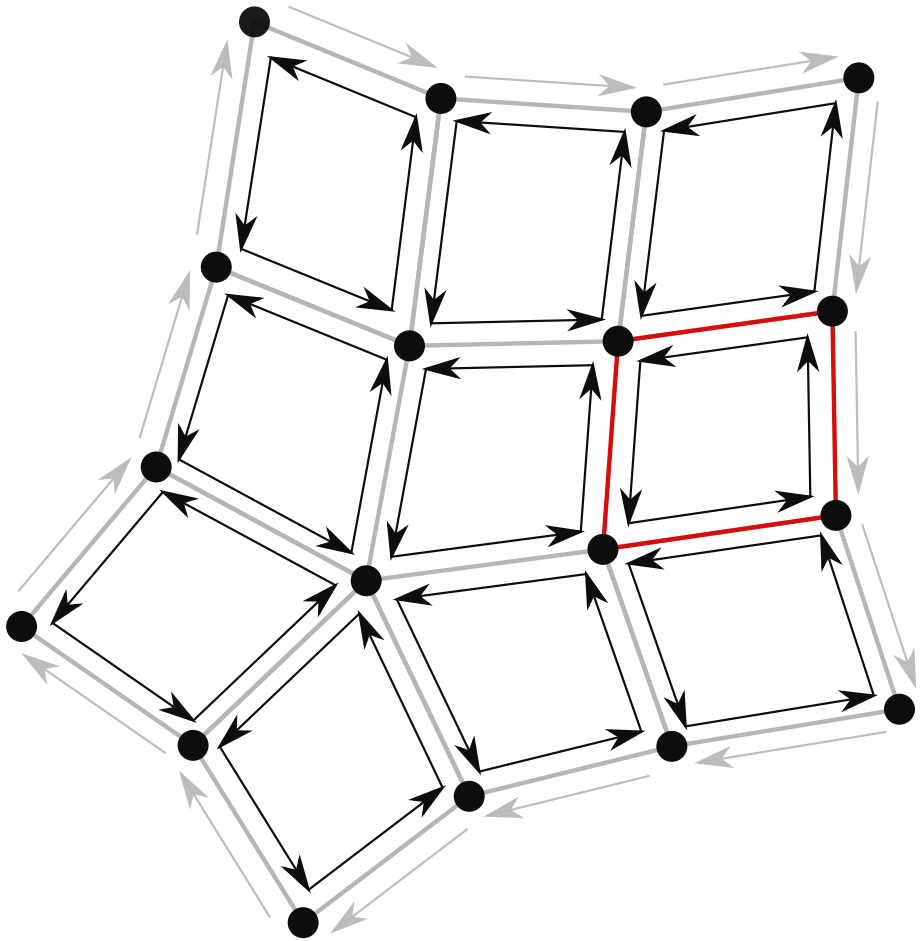
Half-Edge data structure :: HalfEdge

- Target (Vertex*)
- Next (HalfEdge*)
- Prev (HalfEdge*)
- Twin (HalfEdge*)
- Polygon (Face*)



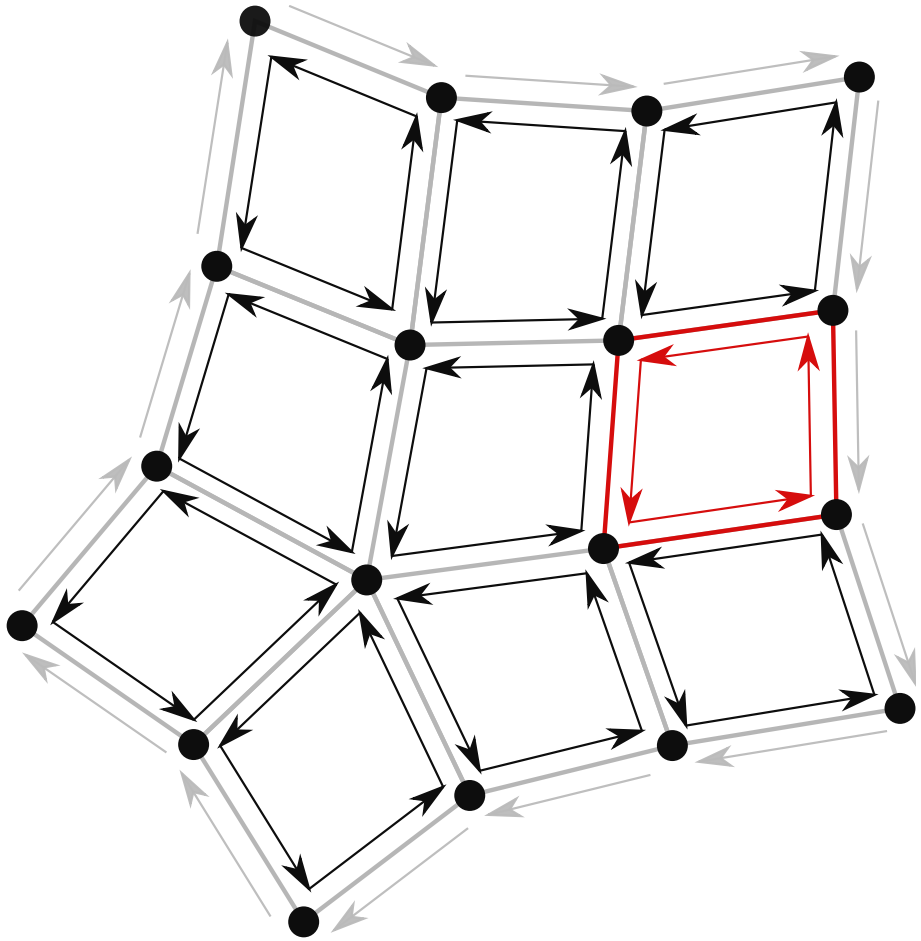
Half-Edge data structure :: Face

- Side (HalfEdge*)
- Valency



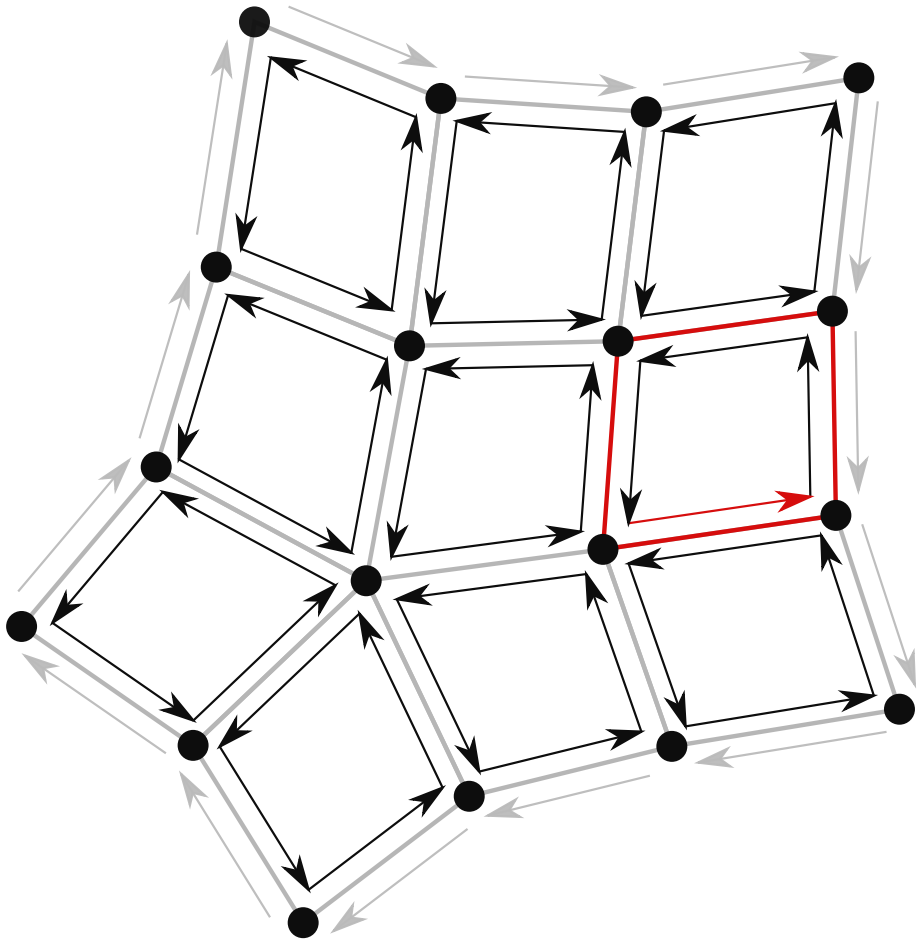
Half-Edge data structure :: Face

- Side (HalfEdge*)
- Valency



Half-Edge data structure :: Face

- Side (HalfEdge*)
- Valency



Half-Edge data structure :: Boundaries

- Do boundary HalfEdges have Twins?

Half-Edge data structure :: Boundaries

- Do boundary HalfEdges have Twins? Yes (this is a choice).

Half-Edge data structure :: Boundaries

- Do boundary HalfEdges have Twins? Yes (this is a choice).
  - Loop of HalfEdges at a boundary (CCW inside, CW outside)

Half-Edge data structure :: Boundaries

- Do boundary HalfEdges have Twins? Yes (this is a choice).
  - Loop of HalfEdges at a boundary (CCW inside, CW outside)
- Do those Twin HalfEdges have a Polygon?

Half-Edge data structure :: Boundaries

- Do boundary HalfEdges have Twins? Yes (this is a choice).
  - Loop of HalfEdges at a boundary (CCW inside, CW outside)
- Do those Twin HalfEdges have a Polygon? No.

Half-Edge data structure :: Tips

- Don't try to think about the structure, but draw it out!

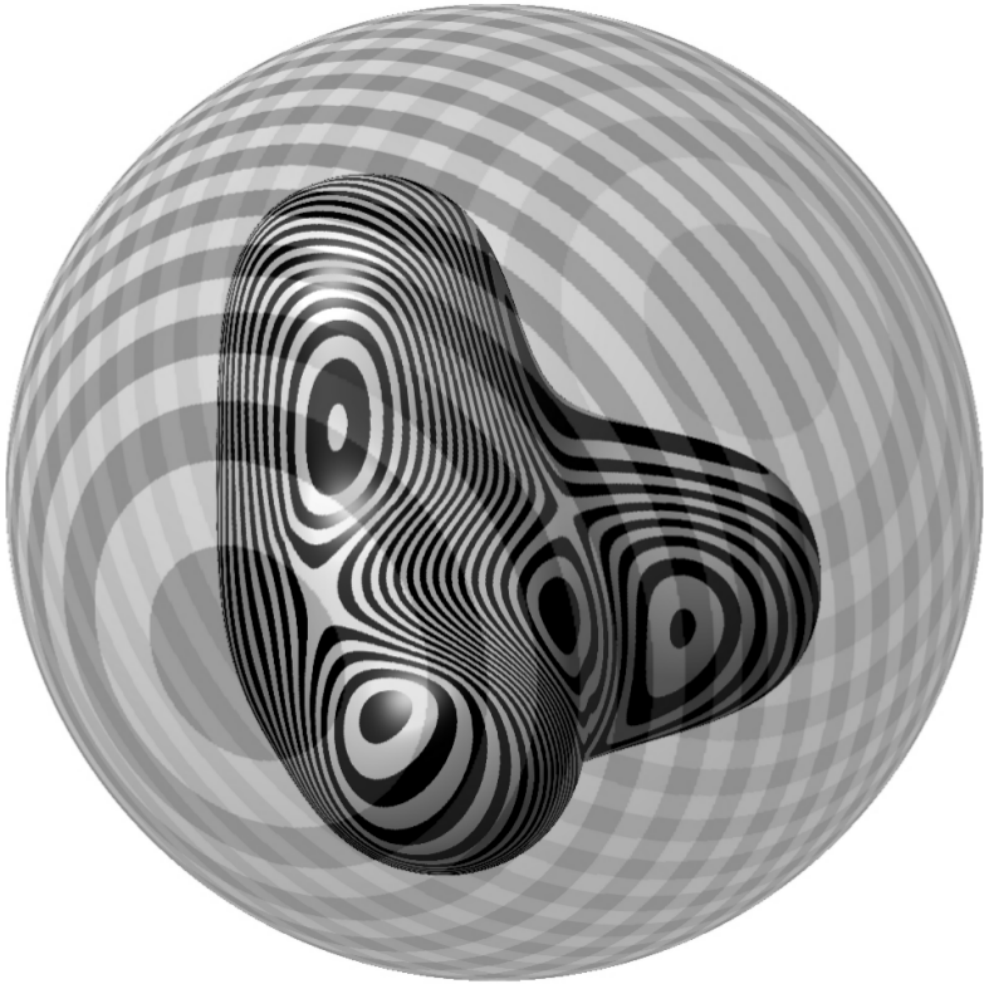
Half-Edge data structure :: Tips

- Don't try to think about the structure, but draw it out!
- Iteration around vertex: `HalfEdge* he = out;`
  - `he = twin->next;`
  - `he = prev->twin;`

Half-Edge data structure :: Tips

- Don't try to think about the structure, but draw it out!
- Iteration around vertex: `HalfEdge* he = out;`
  - `he = twin->next;`
  - `he = prev->twin;`
- Iteration around face: `HalfEdge* he = side;`
  - `he = he->next;`
  - `he = he->prev;`

Reflection lines (interpretation)



Selection

- Also referred to as picking. Input is a pair of (x, y) coordinates in Window/Screen space.

Selection

- Also referred to as picking. Input is a pair of (x, y) coordinates in Window/Screen space.
- Window/Screen $\rightarrow$ NDC $\rightarrow$ Clipping
  - Invert the viewport transformation and multiply by the right value for w. This is not trivial!

Selection

- Also referred to as picking. Input is a pair of (x, y) coordinates in Window/Screen space.
- Window/Screen $\rightarrow$ NDC $\rightarrow$ Clipping
  - Invert the viewport transformation and multiply by the right value for w. This is not trivial!
- Clipping $\rightarrow$ Camera/Eye $\rightarrow$ World ($\rightarrow$ Object)
  - Simply invert the Projection and ModelView matrices

Miscellaneous OpenGL/GLSL

- Reminder: GLSL provides built-in functions such as clamp(), length(), dot(), cross() and many more. See https://www.opengl.org/sdk/docs/man/ (look for GLSL Functions in the left panel).

Miscellaneous OpenGL/GLSL

- Reminder: GLSL provides built-in functions such as clamp(), length(), dot(), cross() and many more. See https://www.opengl.org/sdk/docs/man/ (look for GLSL Functions in the left panel).

- Custom GLSL syntax highlighting, see e.g. http://renderingpipeline.com/2013/12/glsl-syntax-highlighting-for-opengl-4-4/