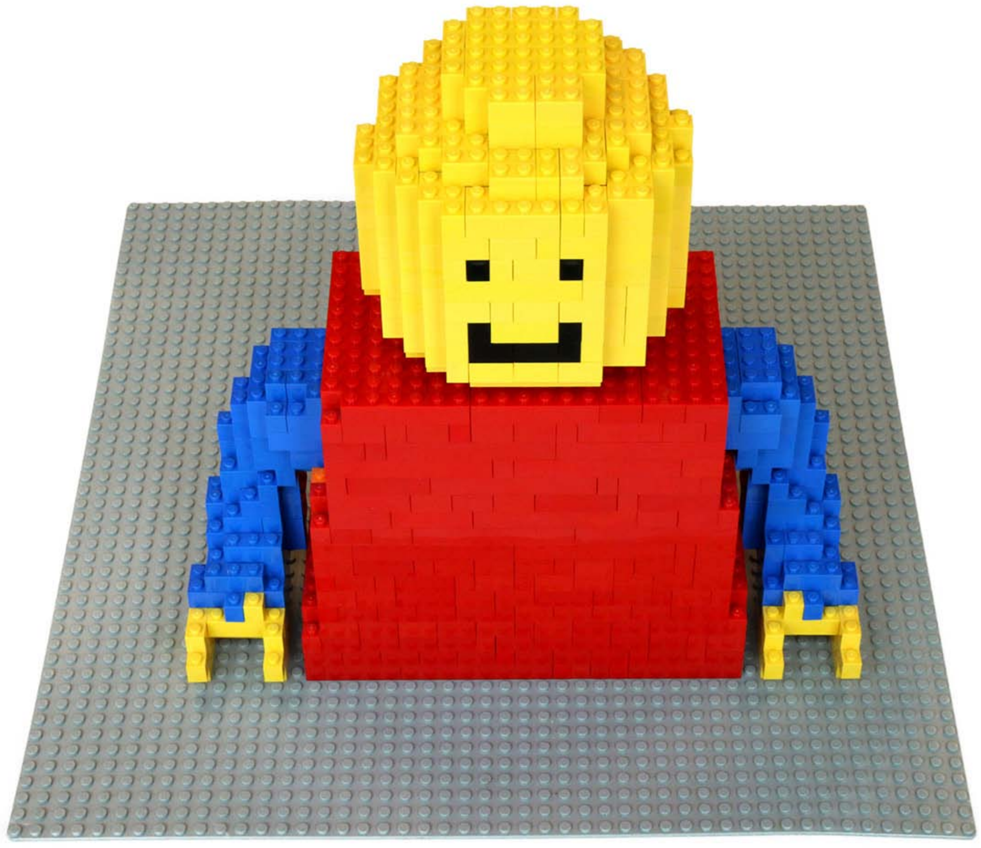


Lego Assembly

Author: Romain Testuz
Supervised by: Yuliy Schwartzburg

Previous work

- All the approaches are based on a variation of the following heuristic:

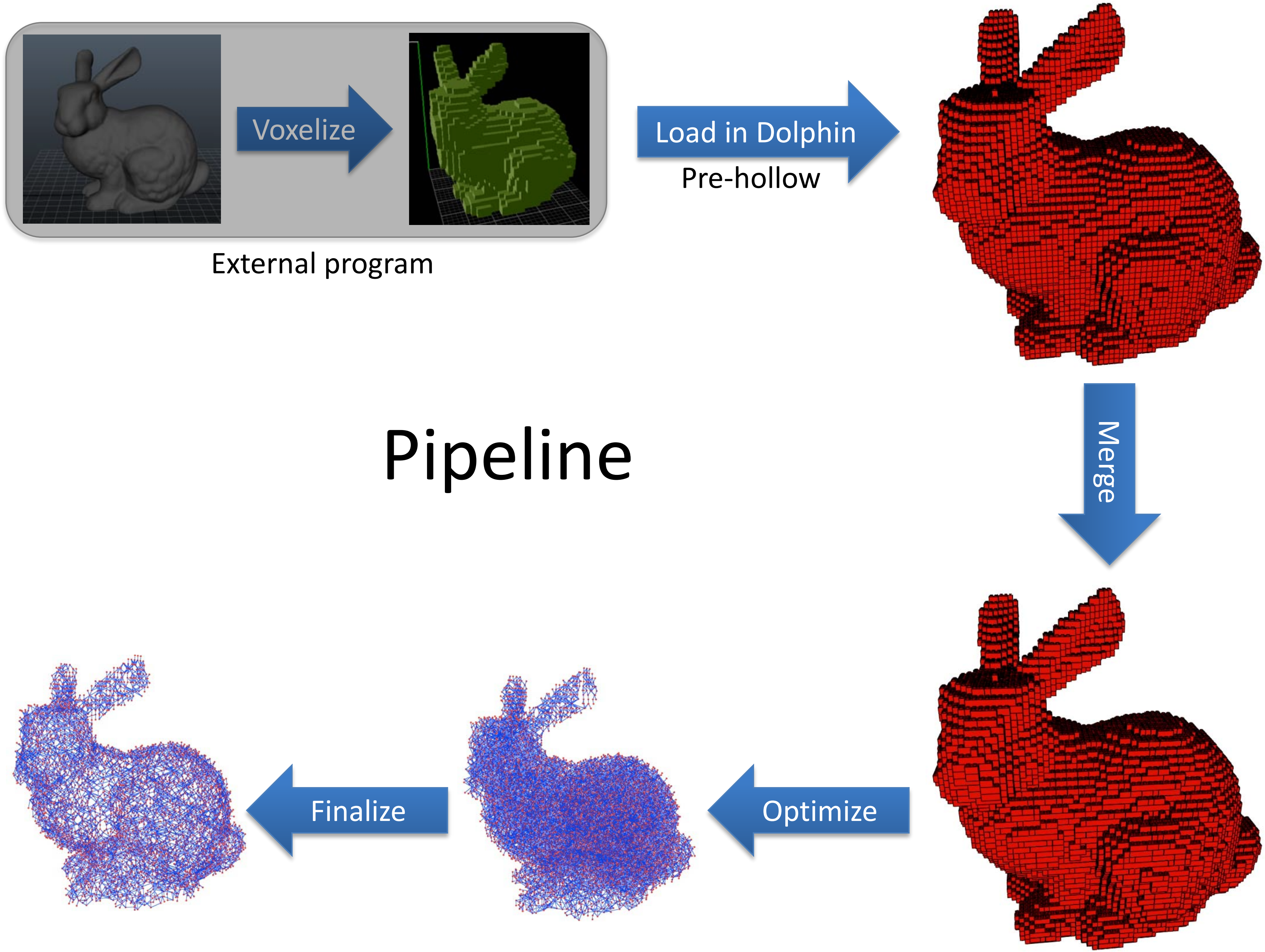
$$\begin{aligned} Fitness &= C_{numbricks} \times numbricks + C_{perpend} \times perpend \\ &+ C_{edge} \times edge + C_{uncovered} \times uncovered \\ &+ C_{otherbricks} \times otherbricks + C_{neighbour} \times neighbour, \end{aligned}$$

- They all try to prevent problems but they never actually try to find the problems to solve them.

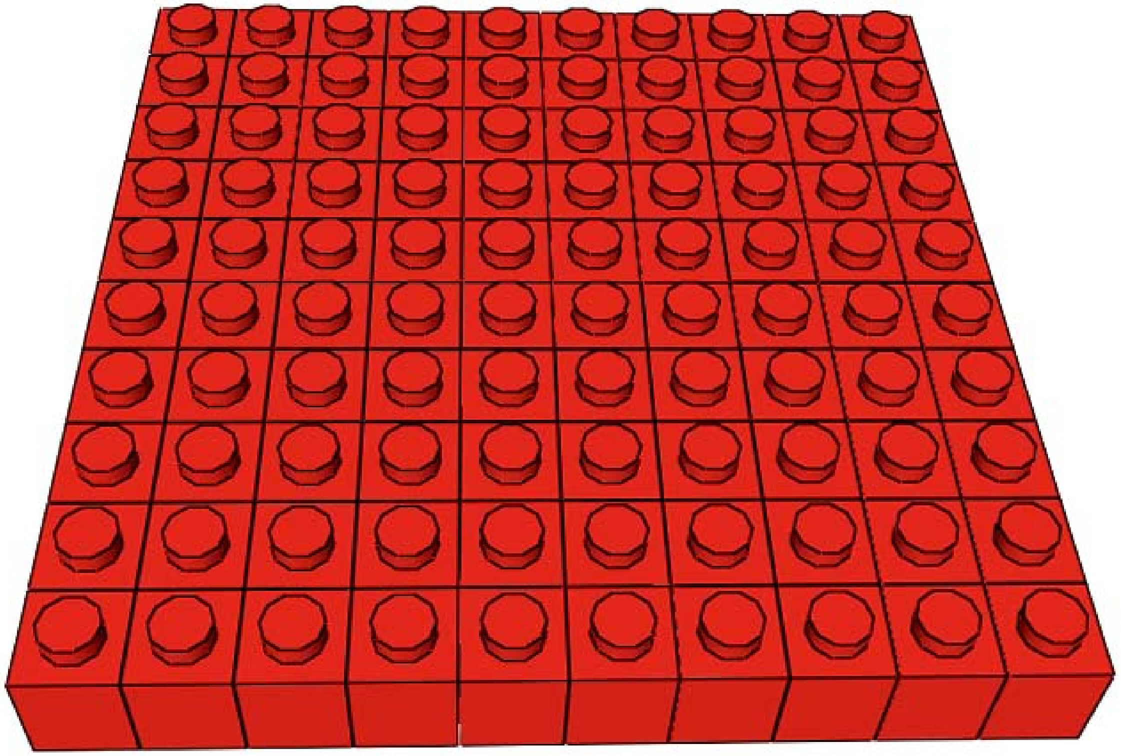
Our approach

- Our method will not try to prevent problems, it will correct them.
- At the end we know if there are problems and we can estimate their seriousness.

Pipeline

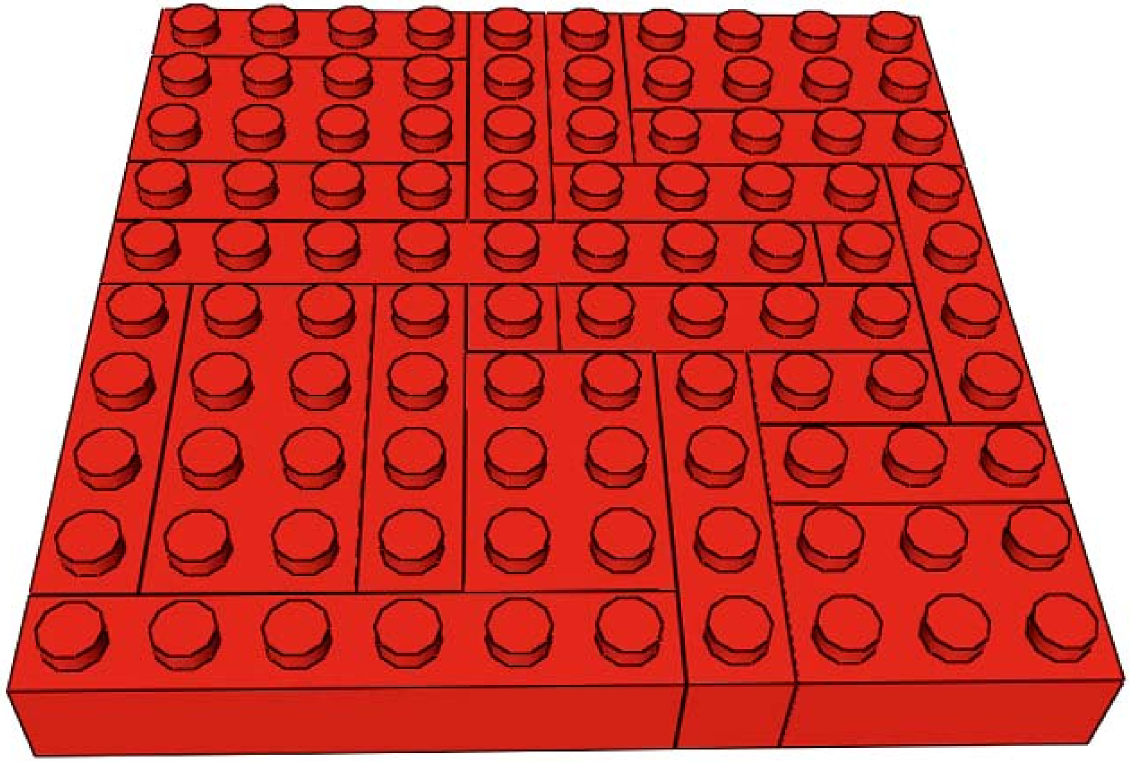

Voxelize
External program
Load in Dolphin
Pre-hollow
Merge
Optimize
Finalize

Merge



Cost function: favors the merges that will create the most connections

Merge

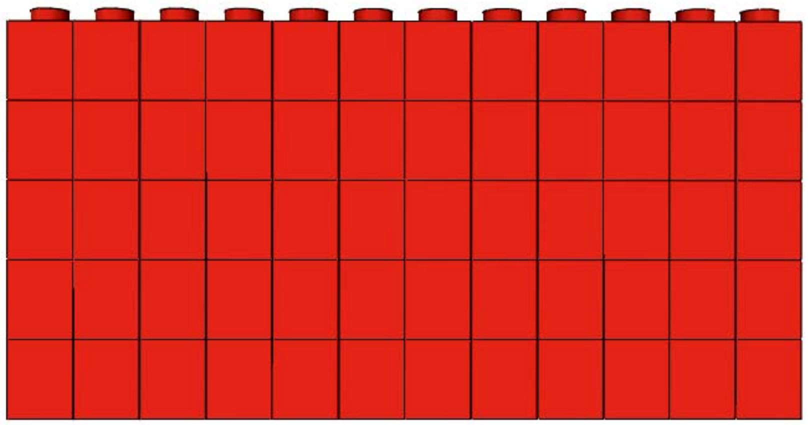


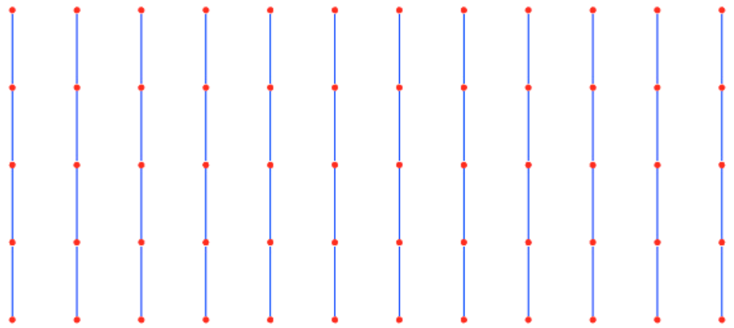
Cost function: favors the merges that will create the most connections

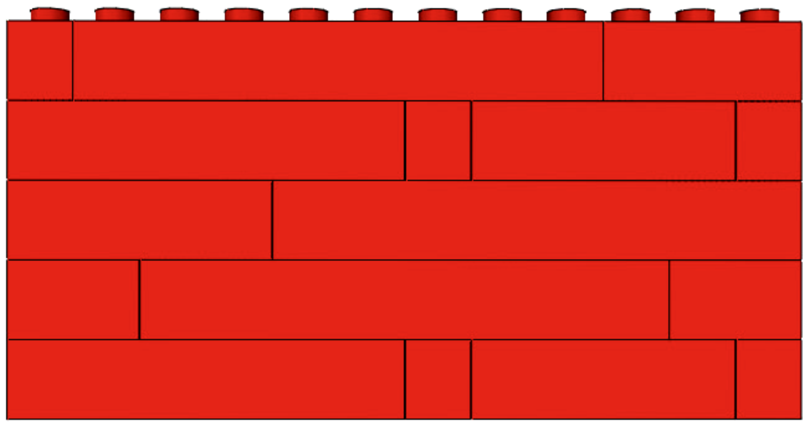
Connectivity graph

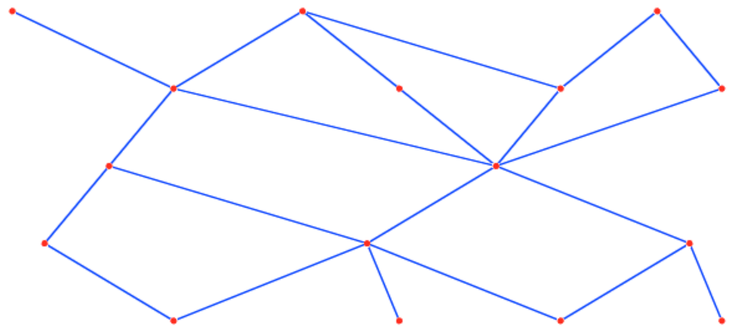
Vertex <=> brick

Edge <=> connection





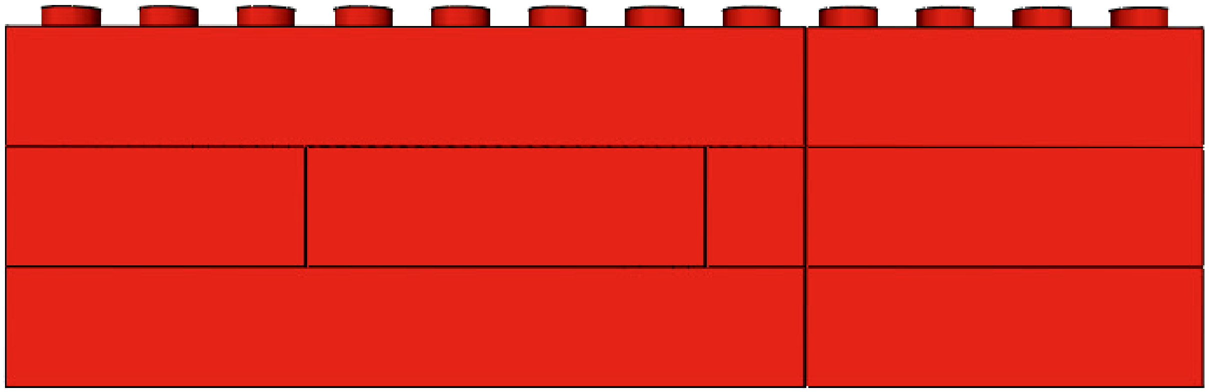


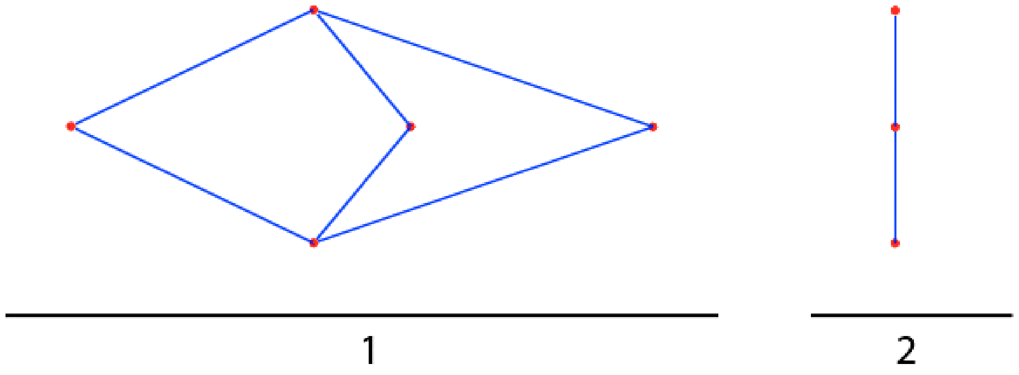


How to measure solidity?

1. All the bricks should be connected: **connected graph**
2. We do not want any brick to be the only connection between 2 large parts of the construction: **articulation points**

Connected components

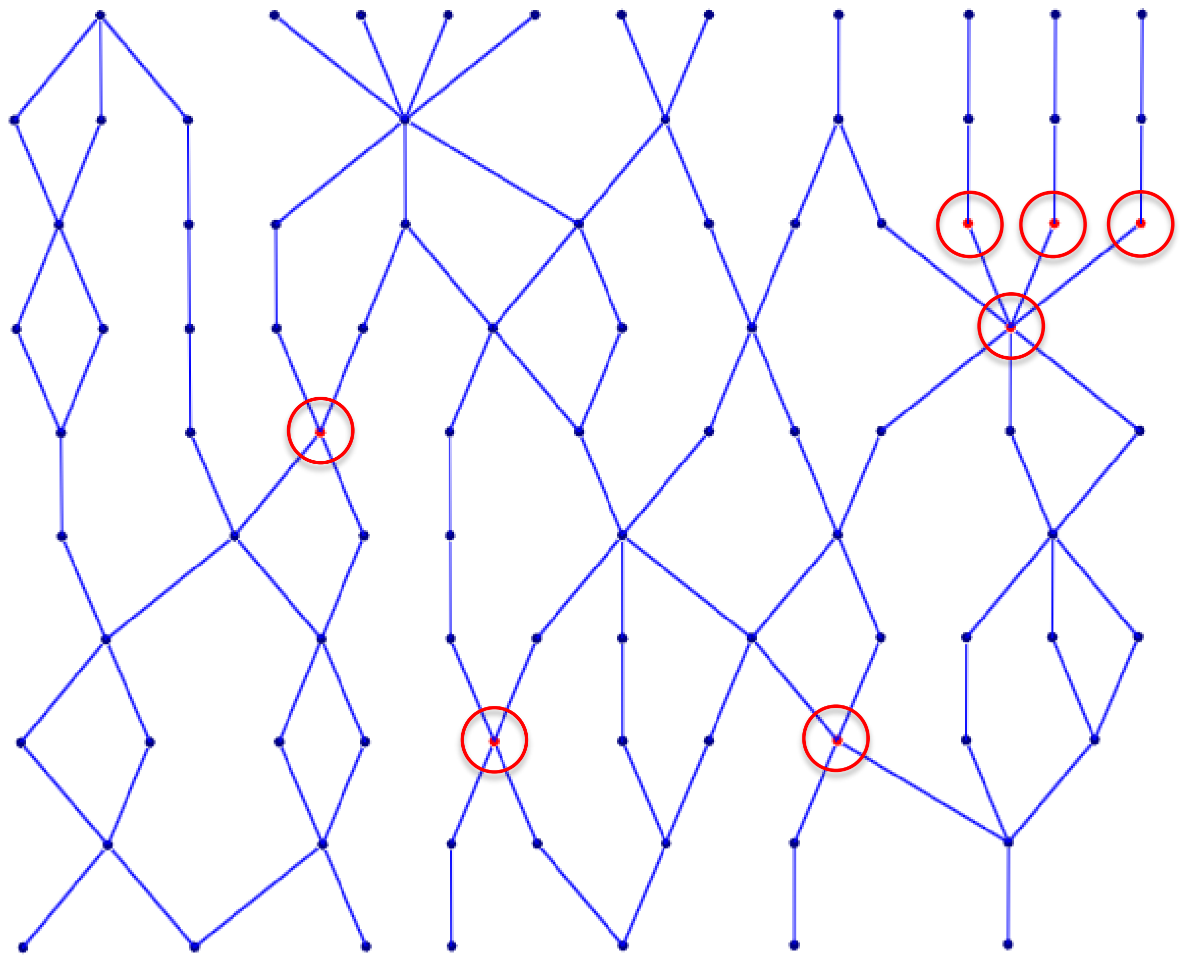




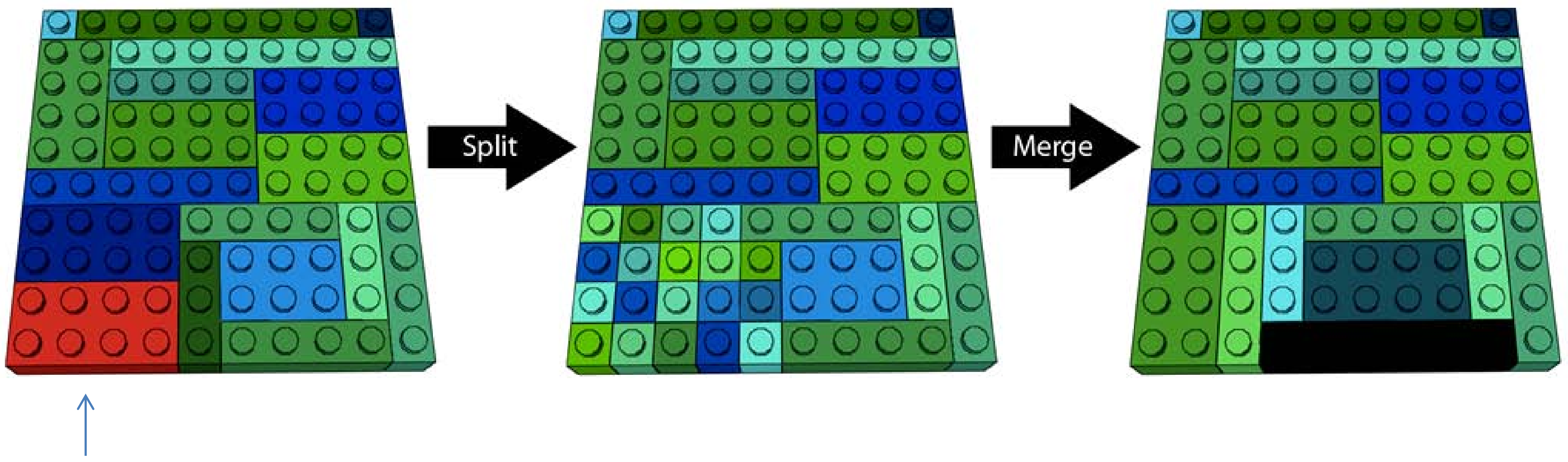
Bad articulation points

- **Articulation point**: a vertex that if removed increases the number of connected components
- **Bad articulation point**: an articulation point which connects 2 or more subgraphs of size greater than 1

“Bad” articulation points

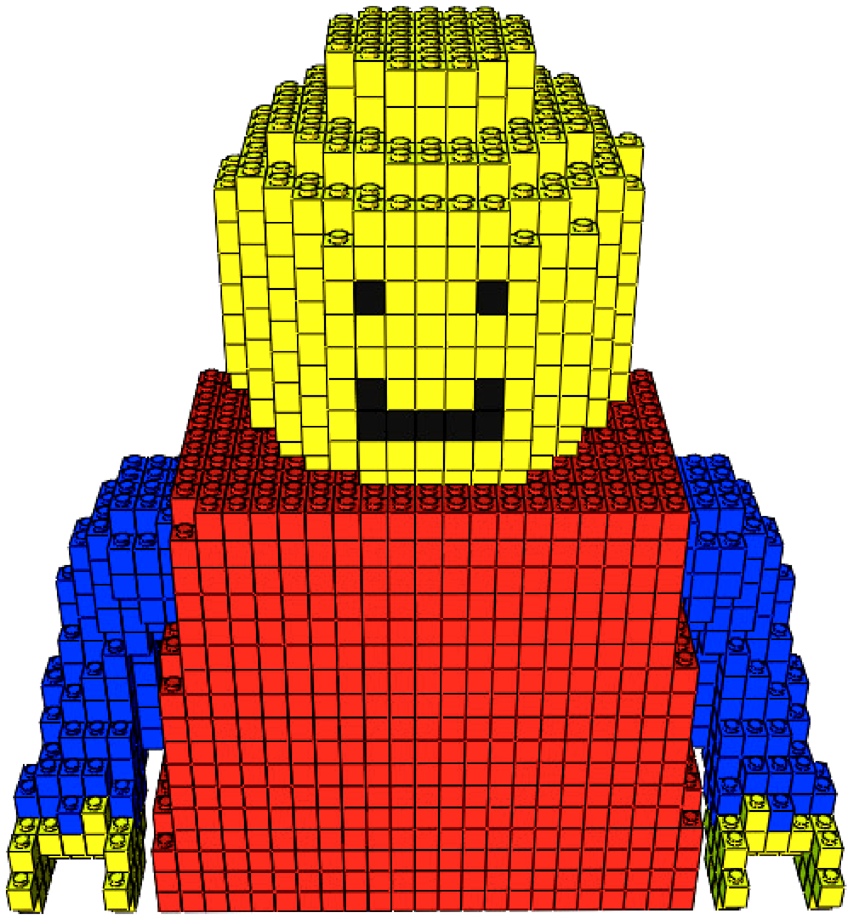


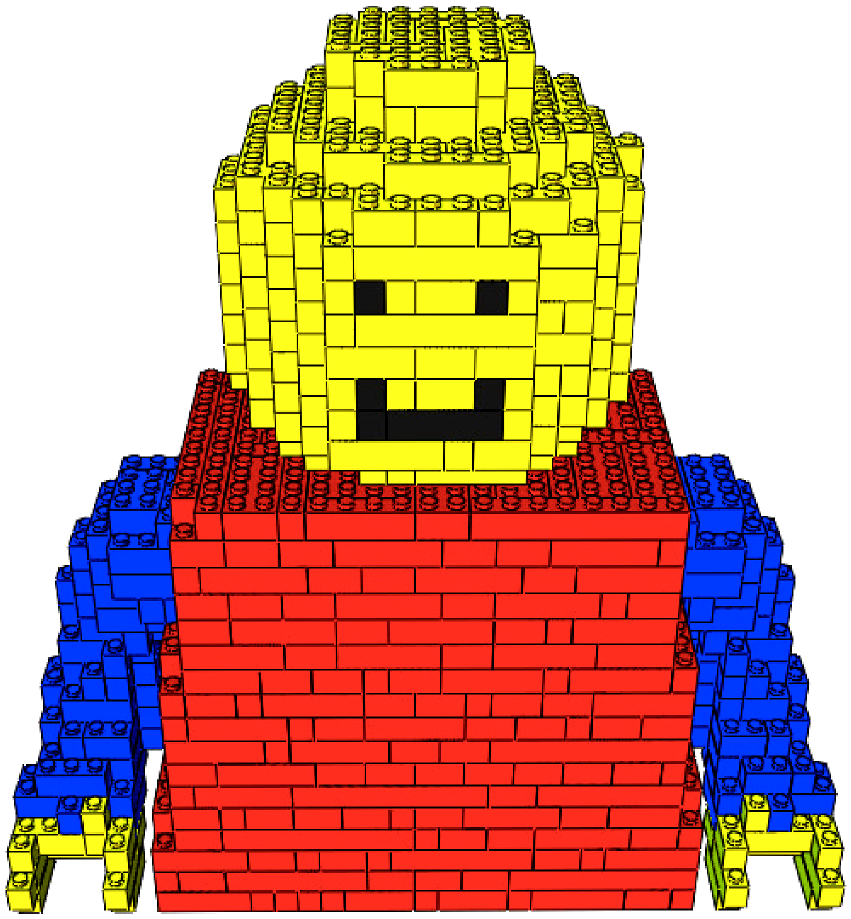
Removing connected components and bad articulation points



Bad articulation point

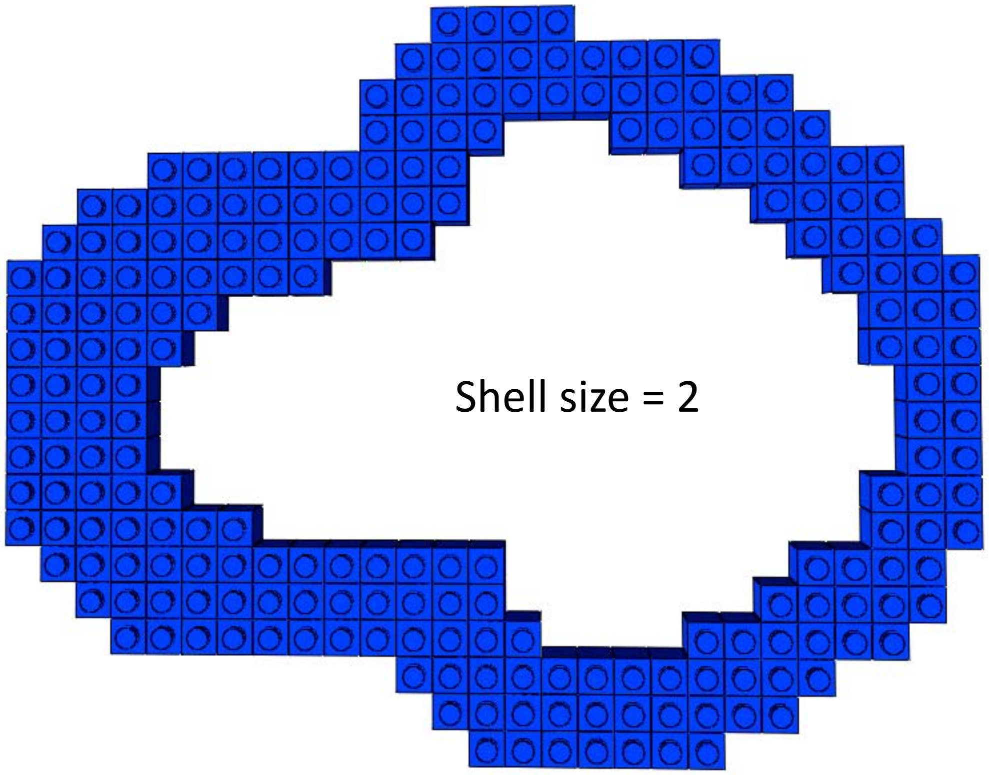
Using colors





Pre-hollow

- Reduces the number of bricks before optimization
- Very fast



Post-hollow

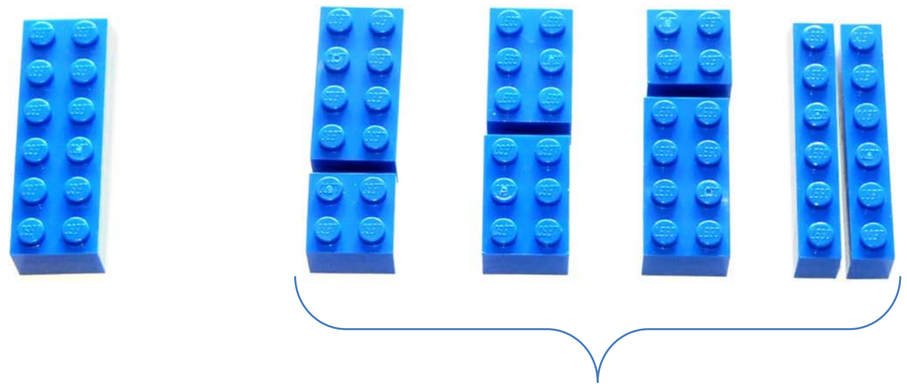
- Start when there is a single connected component and no more bad articulation point.
- Remove an inside brick only if it won’t introduce more connected components or bad articulation point.

Post-hollow

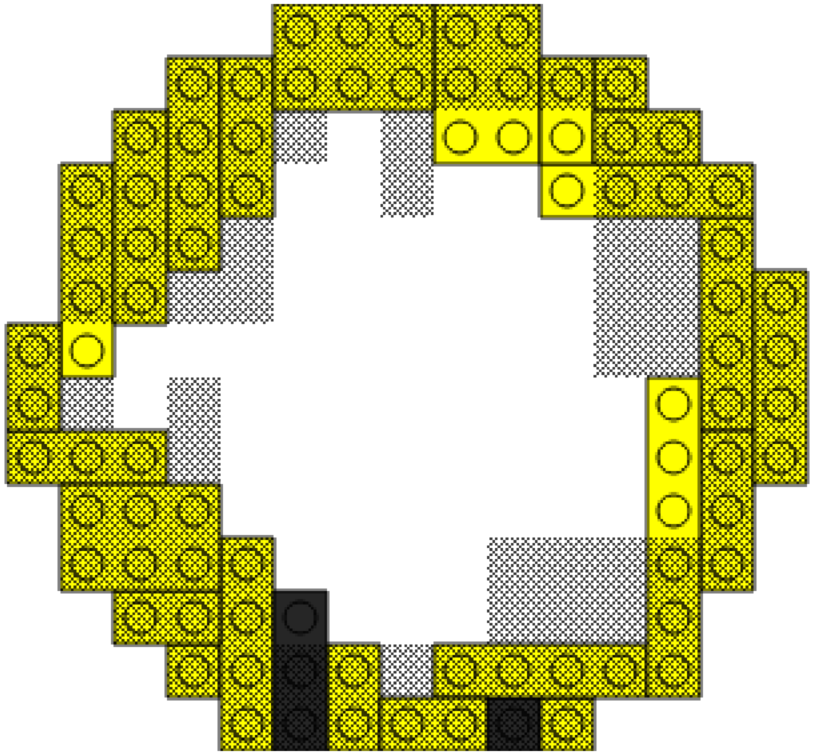
1. Select a random brick in the inside of the model.
2. Create a 3-ring subgraph of the connectivity graph centered on this brick.
3. Compute the connected components and the articulation points of this subgraph.
4. Remove the vertex corresponding to the current brick.
5. Recompute the connected components and the articulation points.
6. If there are more connected components or more articulation points in the second subgraph, the brick cannot be removed. Otherwise, the brick can be removed and the connectivity graph updated.

Satisfying brick type limit

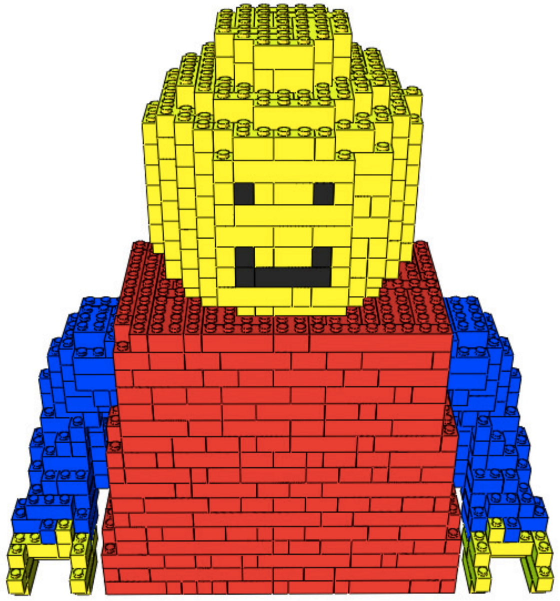
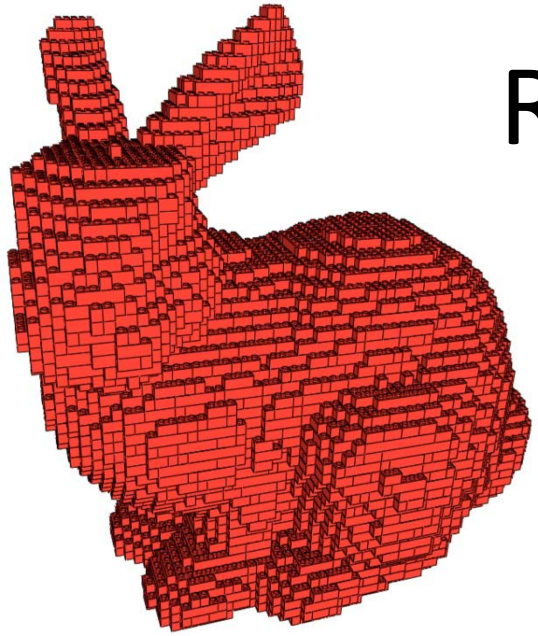
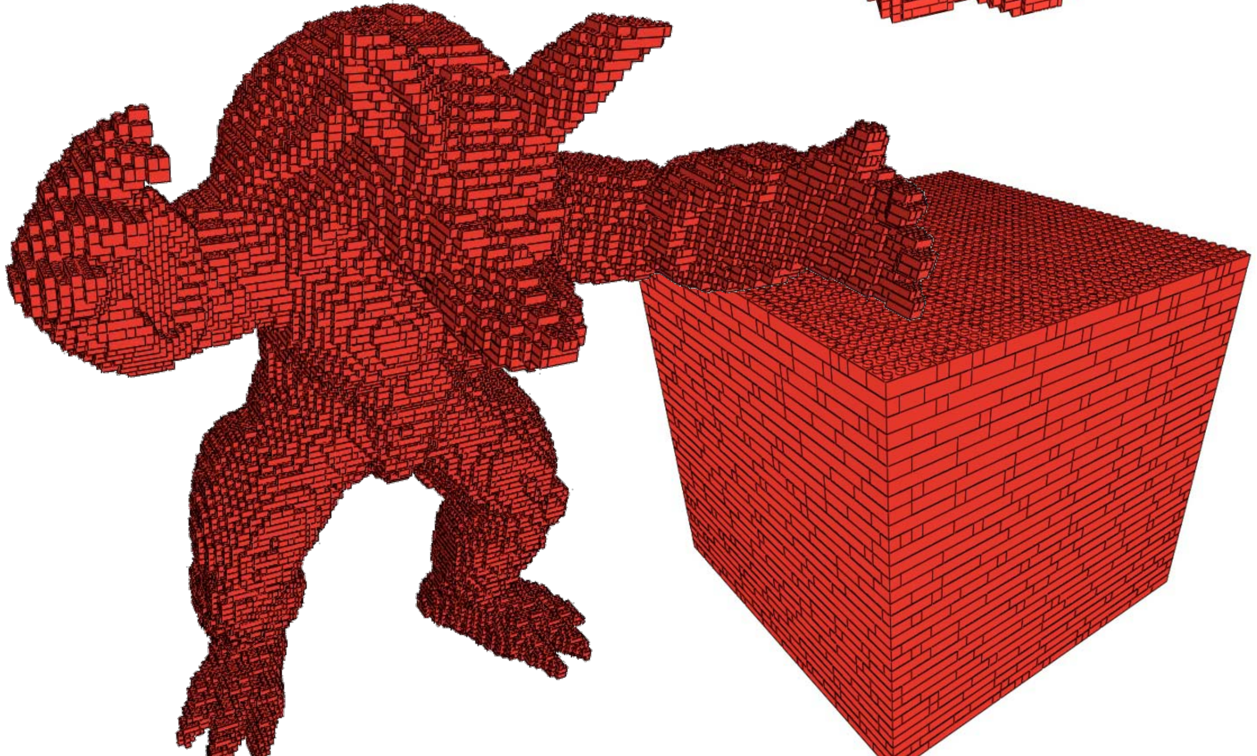
- Last step
- Similar to post-hollowing
- Select an extra brick and see if there is a cut that will not create weak points


The 4 cuts of a 2x6 brick

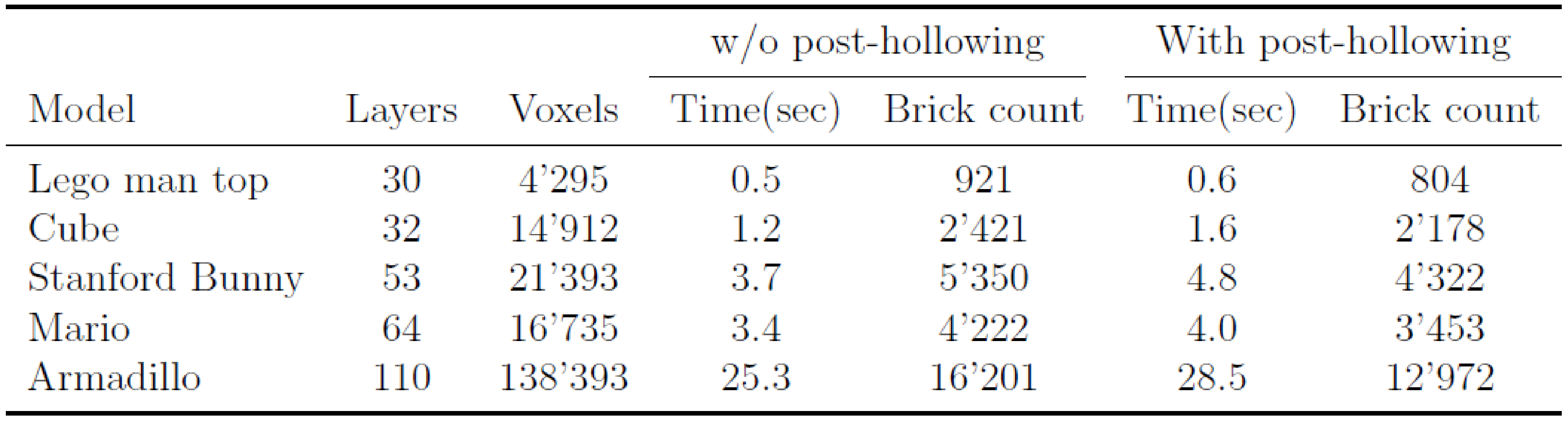
Instructions



Results

Results

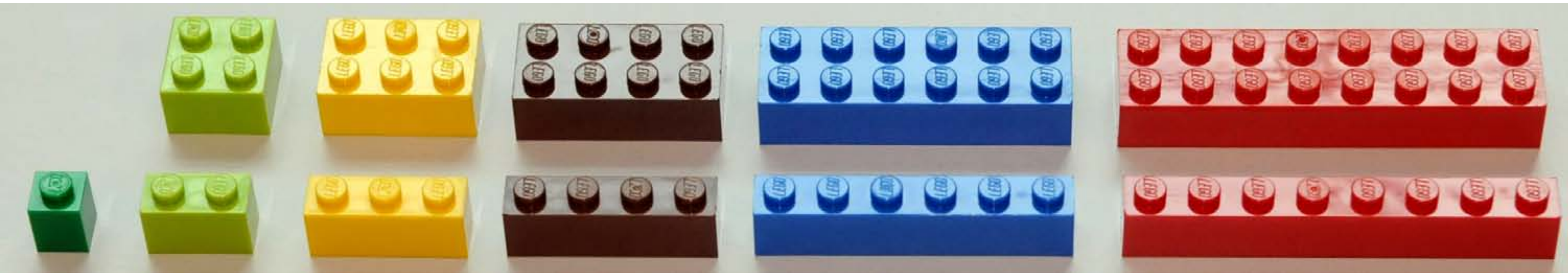


			w/o post-hollowing		With post-hollowing	
Model	Layers	Voxels	Time(sec)	Brick count	Time(sec)	Brick count
Lego man top	30	4'295	0.5	921	0.6	804
Cube	32	14'912	1.2	2'421	1.6	2'178
Stanford Bunny	53	21'393	3.7	5'350	4.8	4'322
Mario	64	16'735	3.4	4'222	4.0	3'453
Armadillo	110	138'393	25.3	16'201	28.5	12'972

- All of the above models have a single connected component and no bad articulation point.

Demo

Legal bricks



1 2 3 4 6 8

Can be changed except for the 1x1 and 1x2

Merging

1. Choose a random brick
2. Find the set of neighbours with which the brick can be merged
3. Select one of them (randomly or with a cost function) and merge
4. Goto 2. until there is no more mergeable neighbours
5. Goto 1