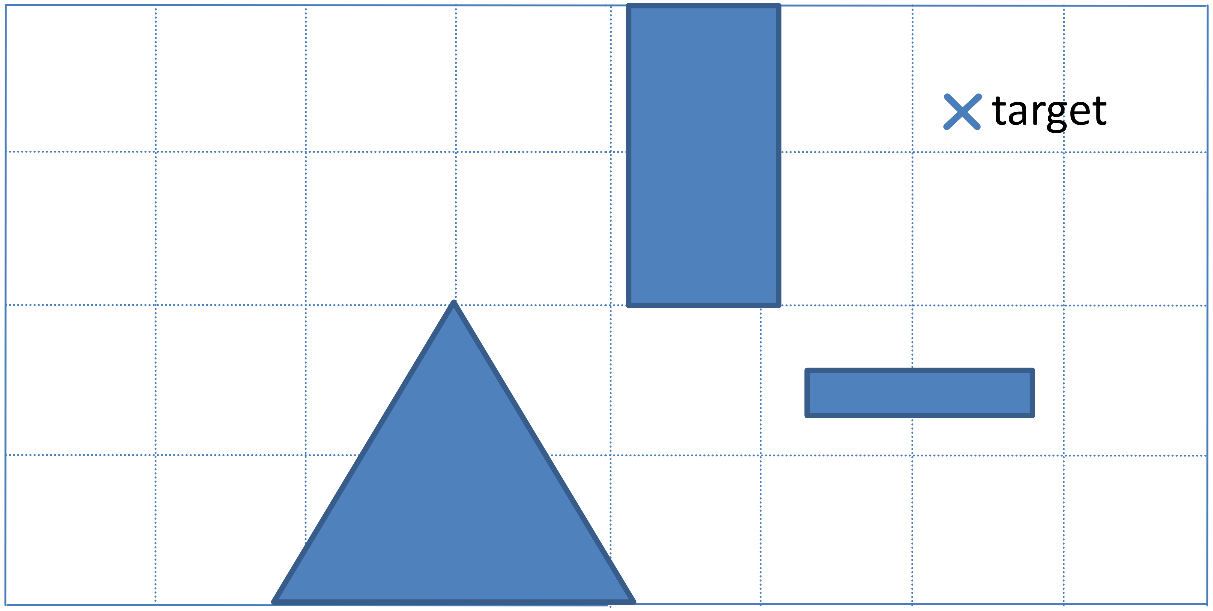
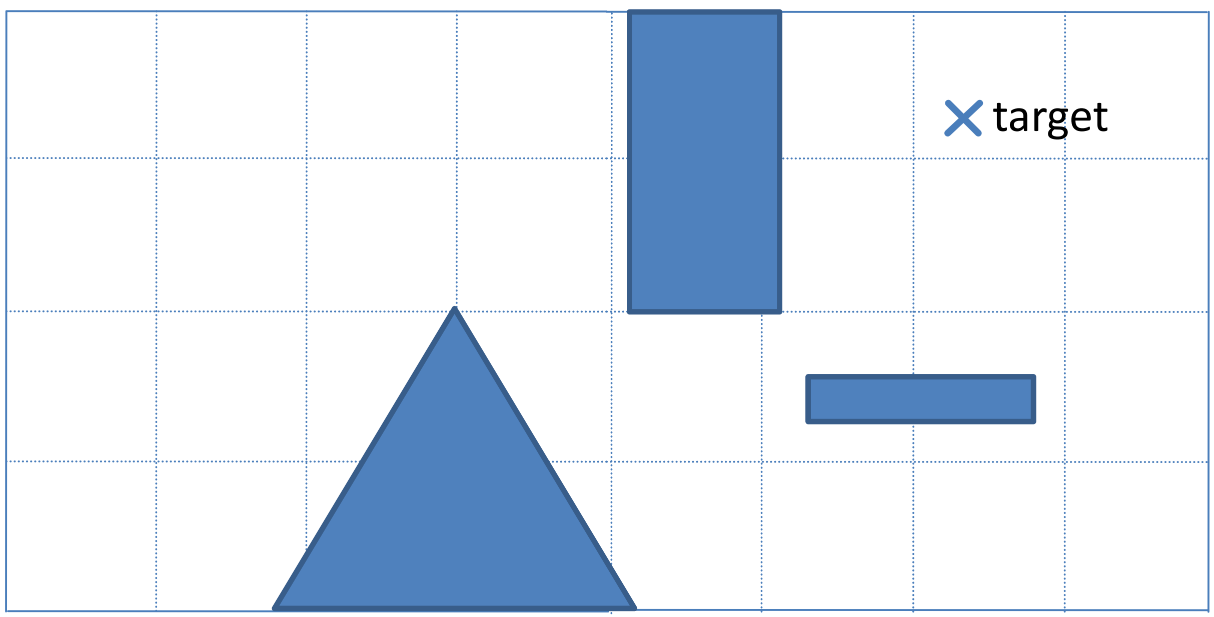
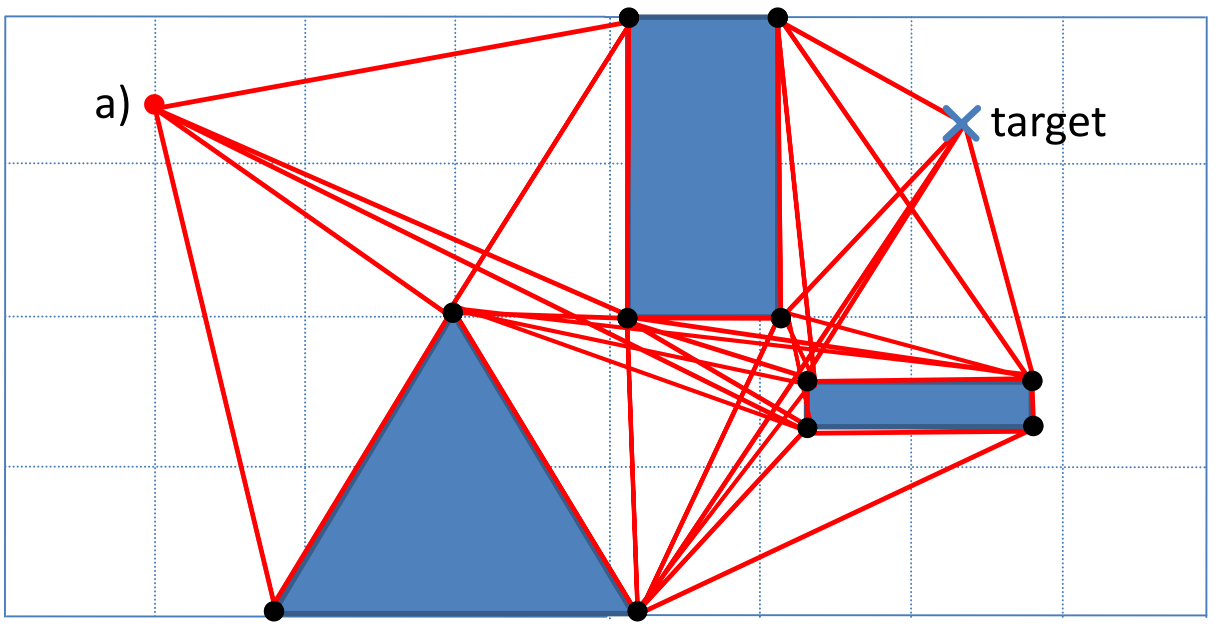

target


target

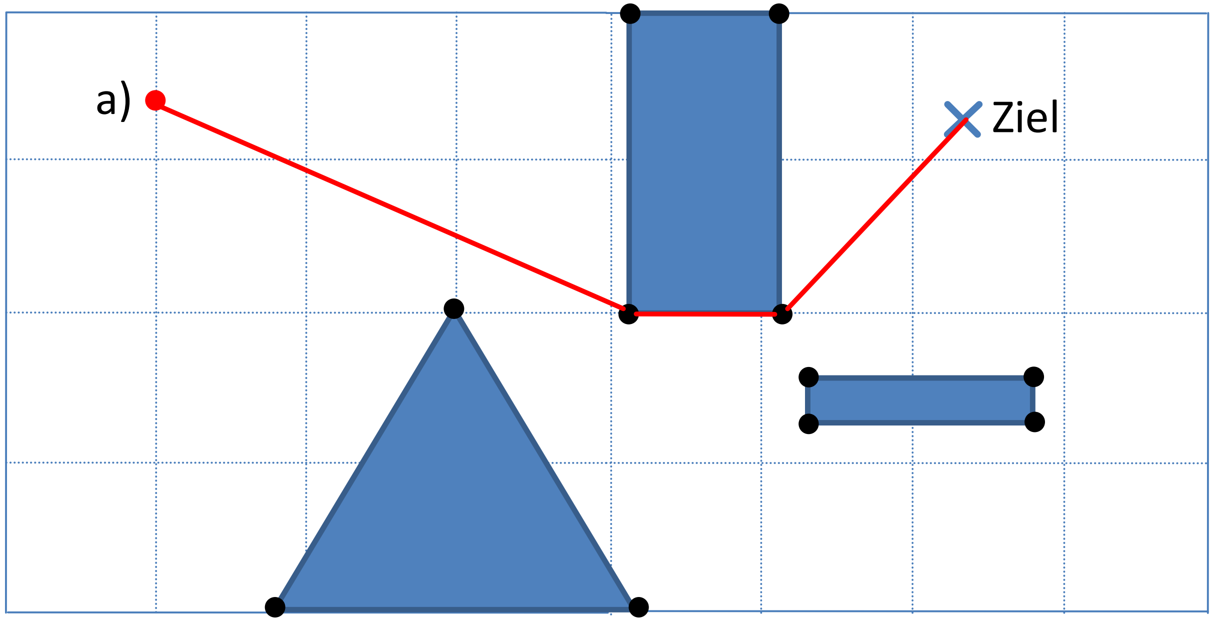
Visibility graph:

- Nodes are corner points of all obstacles, start and target
- An edge between a pair of nodes (a,b) exists if and only if a and b can see another, i.e. if the line [a,b] is not interrupted by an obstacle. Edges of polygons are edges in the visibility graph, too.



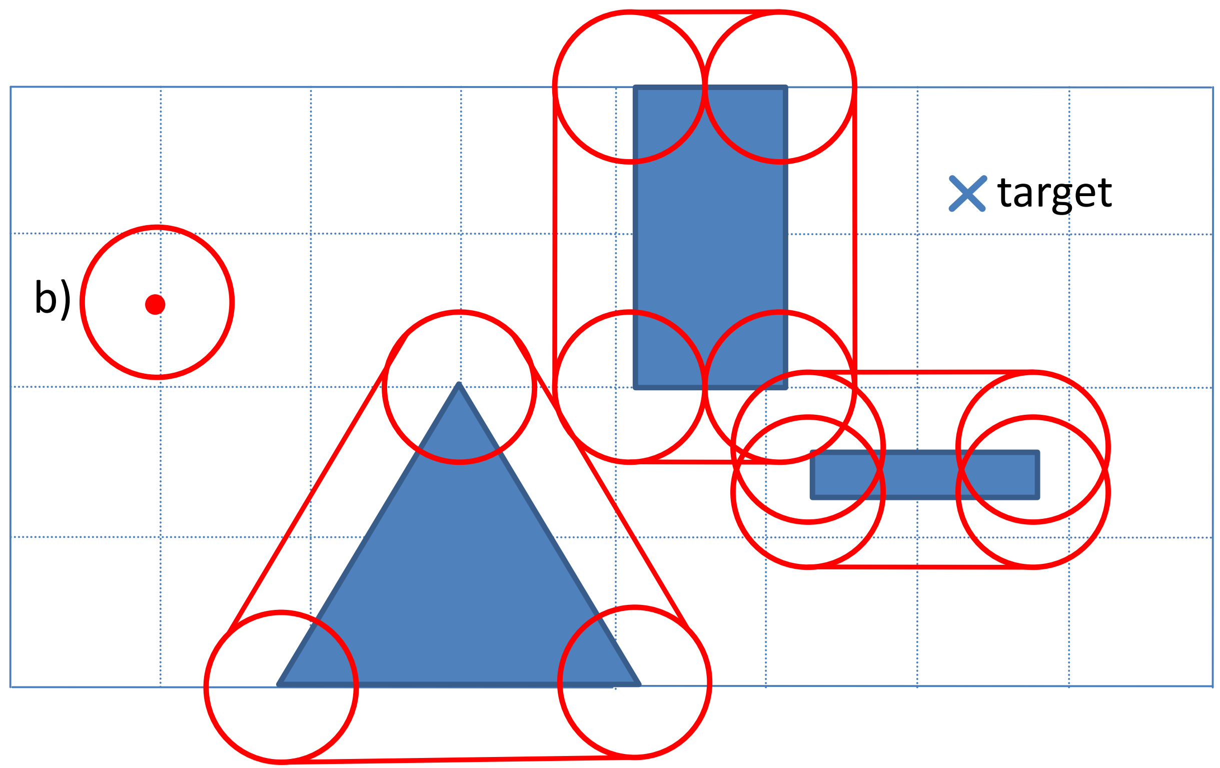
Visibility graph:

- Edge costs correspond to the Euclidean distance
- Shortest path search in the visibility graph (e.g. with Dijkstra) delivers shortest path between start and target

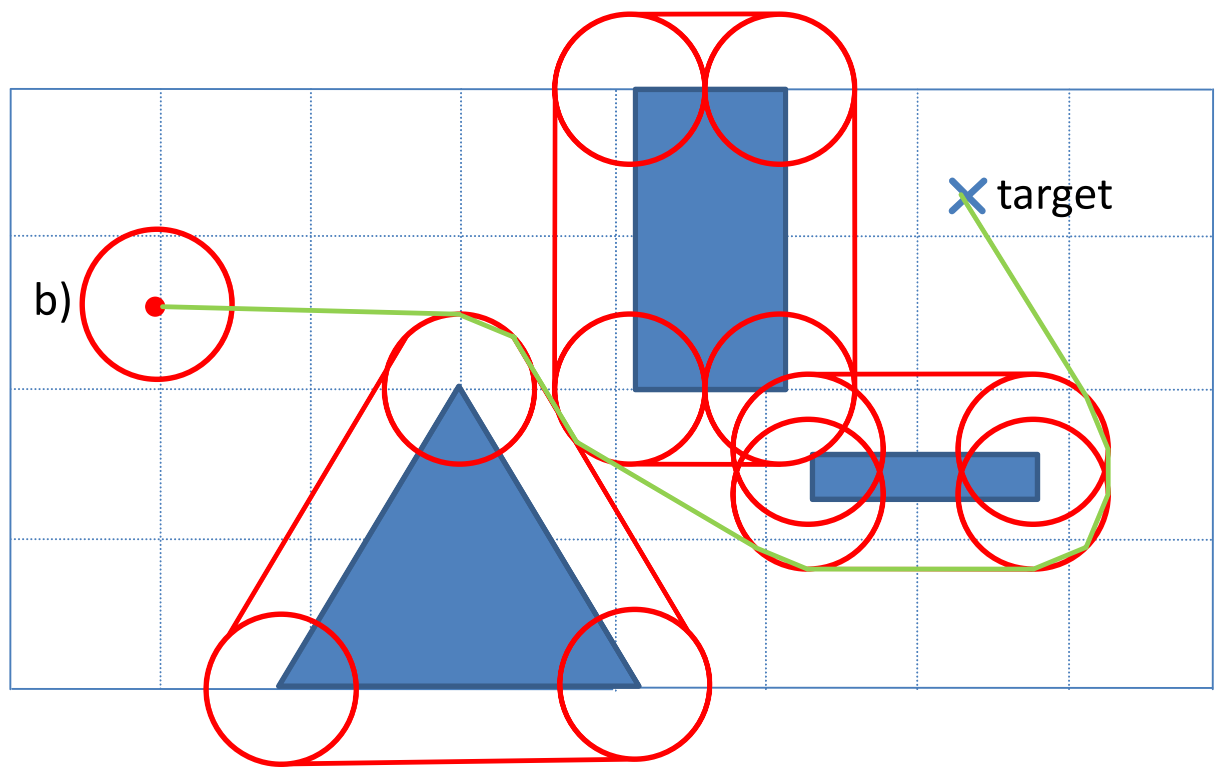


For extended objects which are invariant to point reflection:

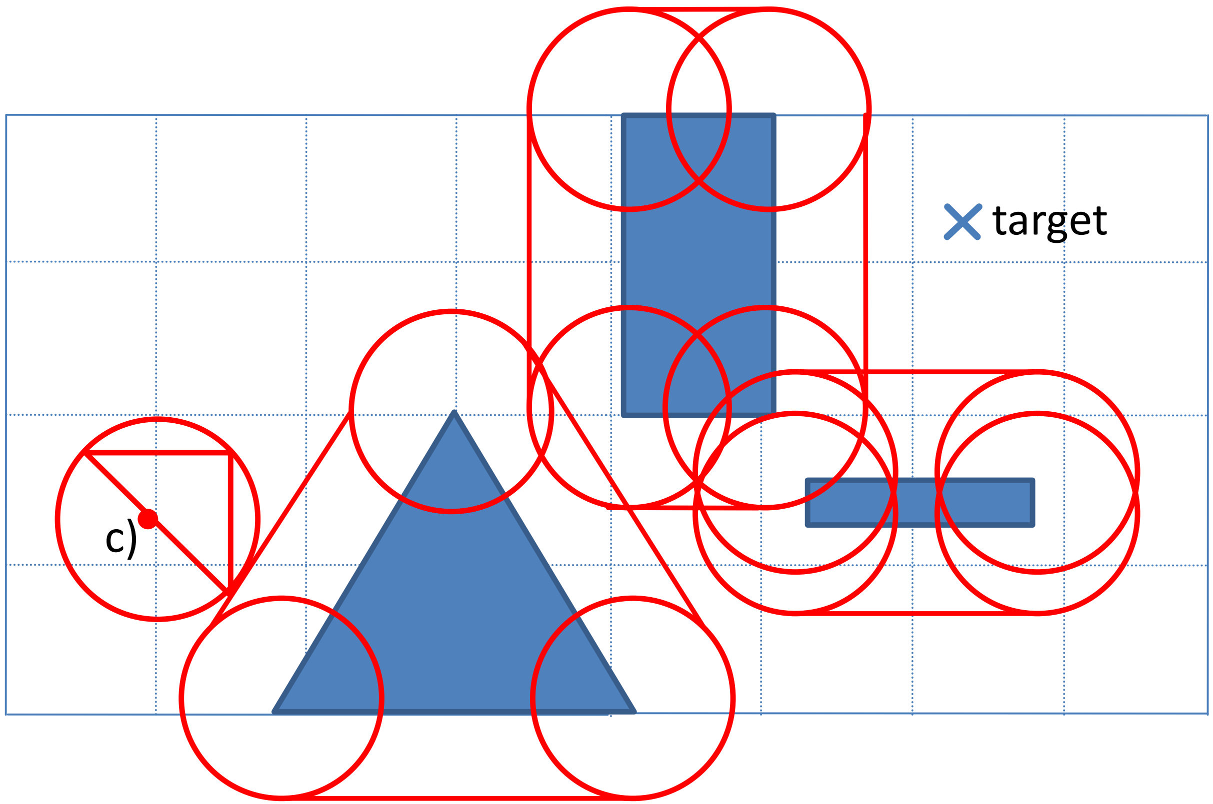
- Pick centroid of the object and expand all obstacles by the corresponding Minkowski-sum
- Like that the path finding of the extended object can be reduced to the path finding of a point: the shortest path of a circle through the obstacles is equivalent to the shortest path of the center of the circle through the expanded obstacles.
- Problem: Circles have infinite many angles which means that the visibility graph is not computable precisely.
- Solution: Approximate the circle with a polygon which is invariant to point reflection (e.g. hexagon, octagon)



Not computable, but obvious solution:

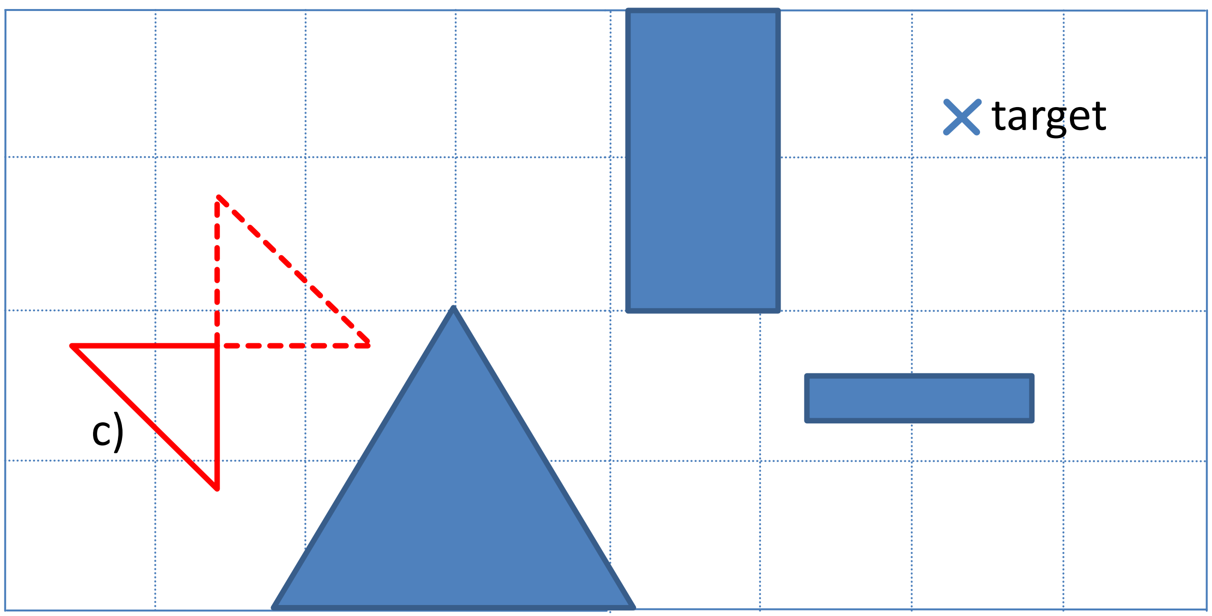


Annotation:
The triangle is not invariant to point reflection-
If approximated with a circle, there would be no path to the target.

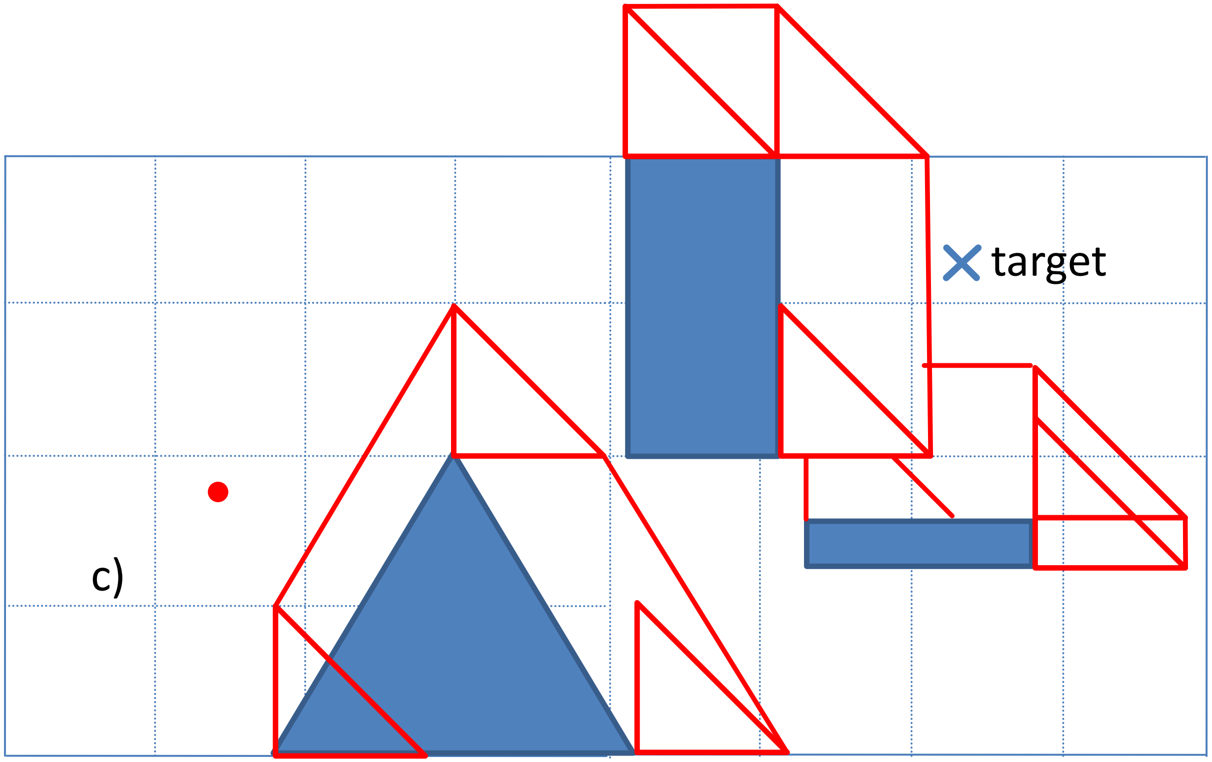


Approach from robotics (motion planning) which may be too expensive for MMOs:

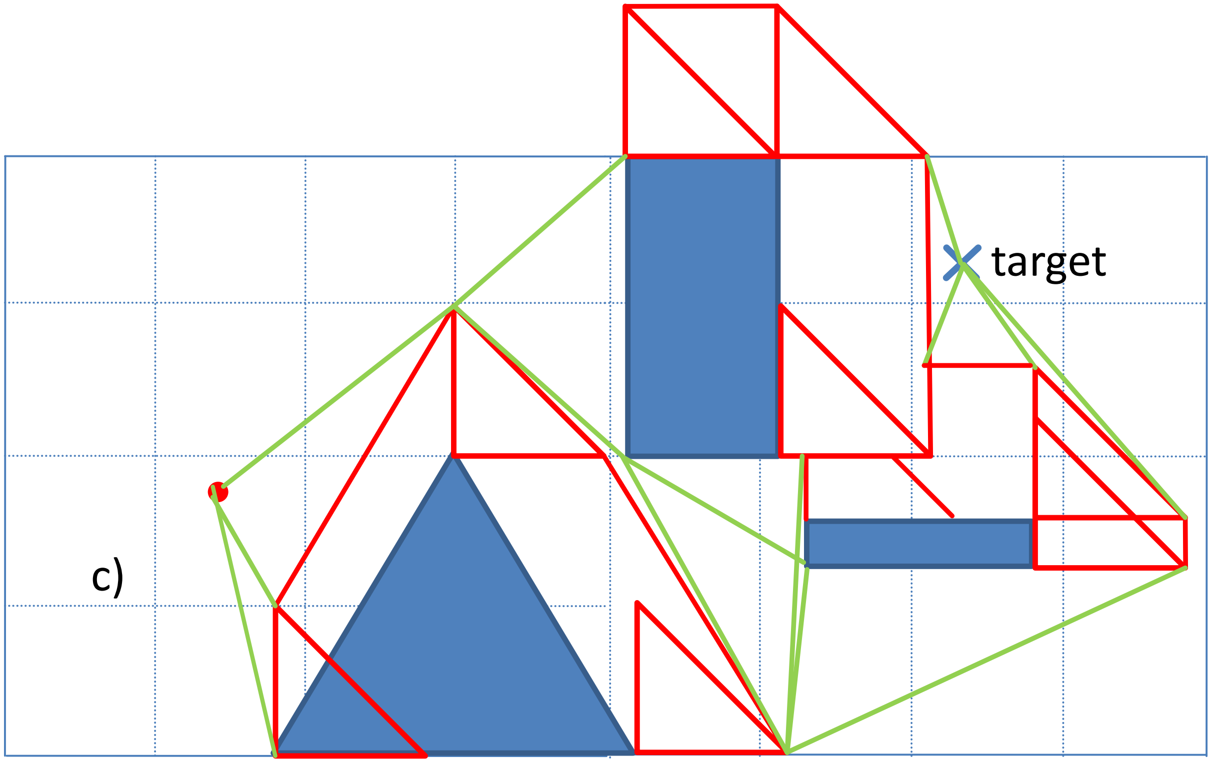
- Choose reference point in the polygon, e.g. a corner.
- Calculate Minkowski difference of the polygon (relating to the reference point) to every polygon (=Minkowski sum of the object point reflected by the reference point)
- Like that the polygon itself „shrinks“ to a point, obstacles „grow“ (see also robotic work space-> configuration space (C-Space, e.g. all positions of a roboter in a room), C-obstacles, free space)
- Move point to the target through free space.



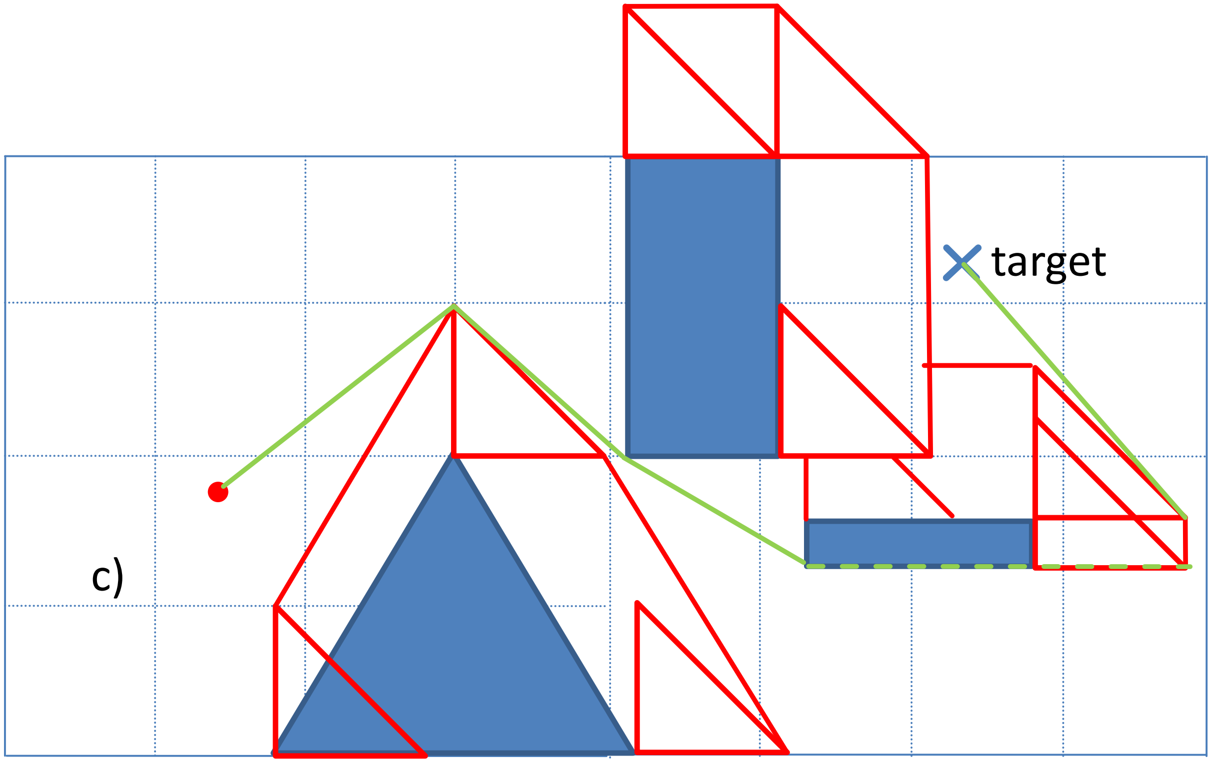
Now the shortest path to the target is again the shortest path in the visibility graph.

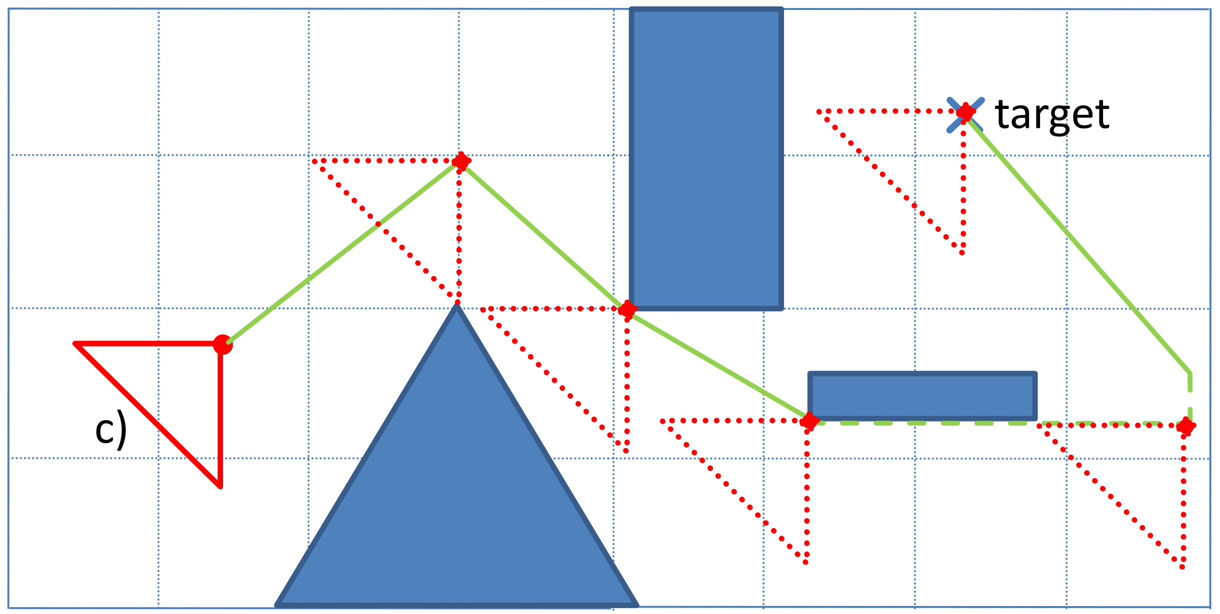


Now the shortest path to the target is again the shortest path in the visibility graph.



Now the shortest path to the target is again the shortest path in the visibility graph.



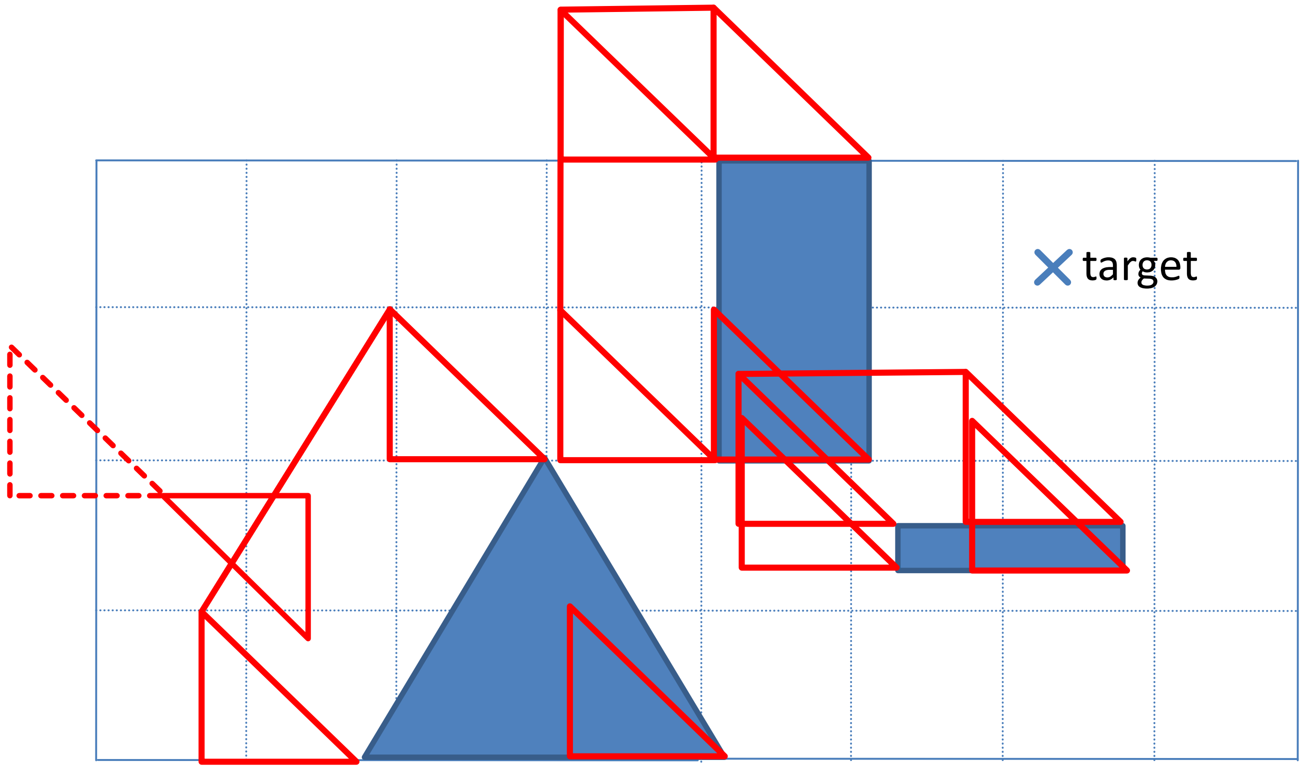


Path finding of such a triangle is thus not solvable with the rather coarse approximation of a circle.

But it is possible using the Minkowski difference, which though depends on the shape and can be computationally intensive for a high variety of shapes in a game

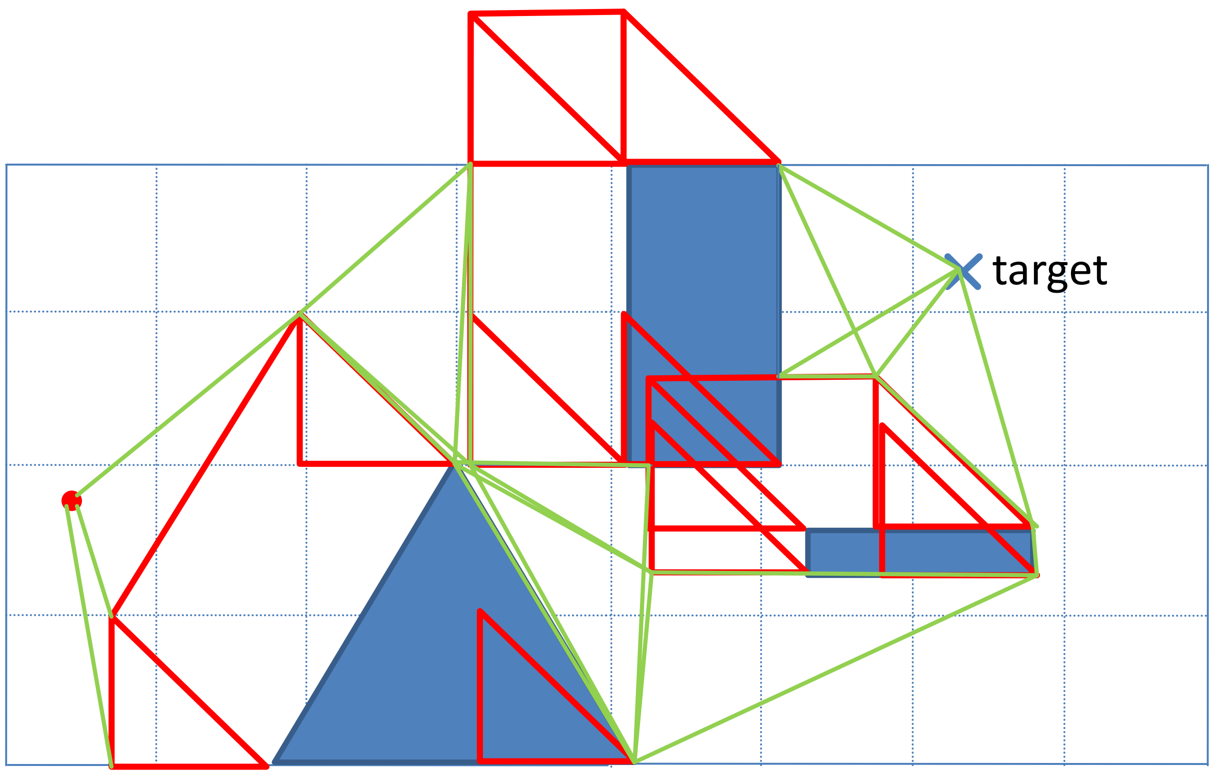
Addition: if another corner of the triangle is chosen as reference point, it looks like this:

c)



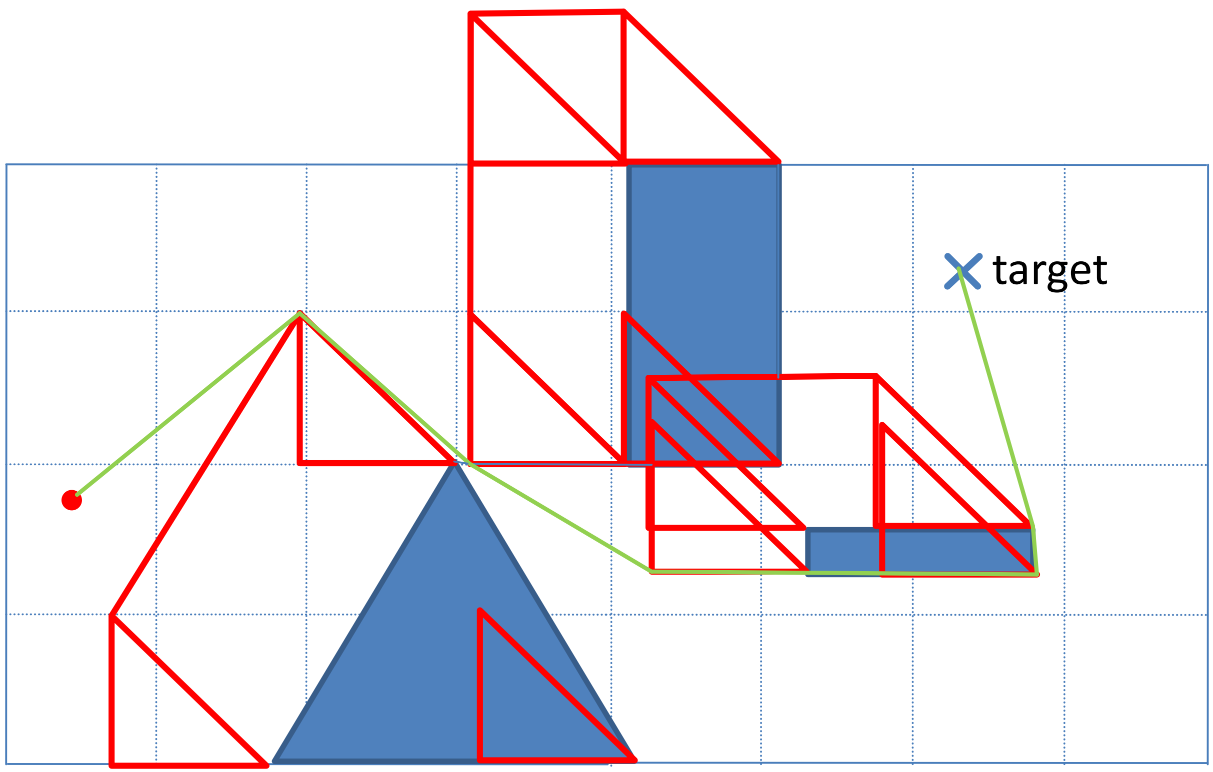
Addition: if another corner of the triangle is chosen as reference point, it looks like this:

c)



Addition: if another corner of the triangle is chosen as reference point, it looks like this:

c)



Addition:

Comparison: The orange path was the shortest path found previously. As we can see, it is the same, so the shortest path does not depend on the reference point chosen.

c)