# Efficient Traignaulation-Based Pathfinding

Douglas Demyen and Michael Buro


{demyen|mburo}@cs.ualberta.ca

Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8

UNIVERSITY OF ALBERTA

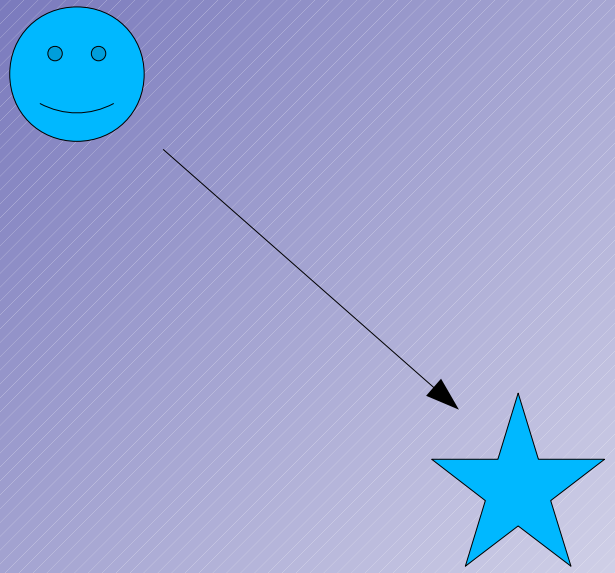
# Overview

- Pathfinding Introduction
- Grid-Based Approaches
- Triangulations
- Funnel Algorithm
- Triangle “Width”
- “Naïve” Search
- Optimality Concerns
- TA* (Base-Level) Search
- Triangulation Abstraction
- TRA* (Abstraction) Search
- Experiments
- Results
- Conclusion
- Future Work


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Pathfinding Introduction

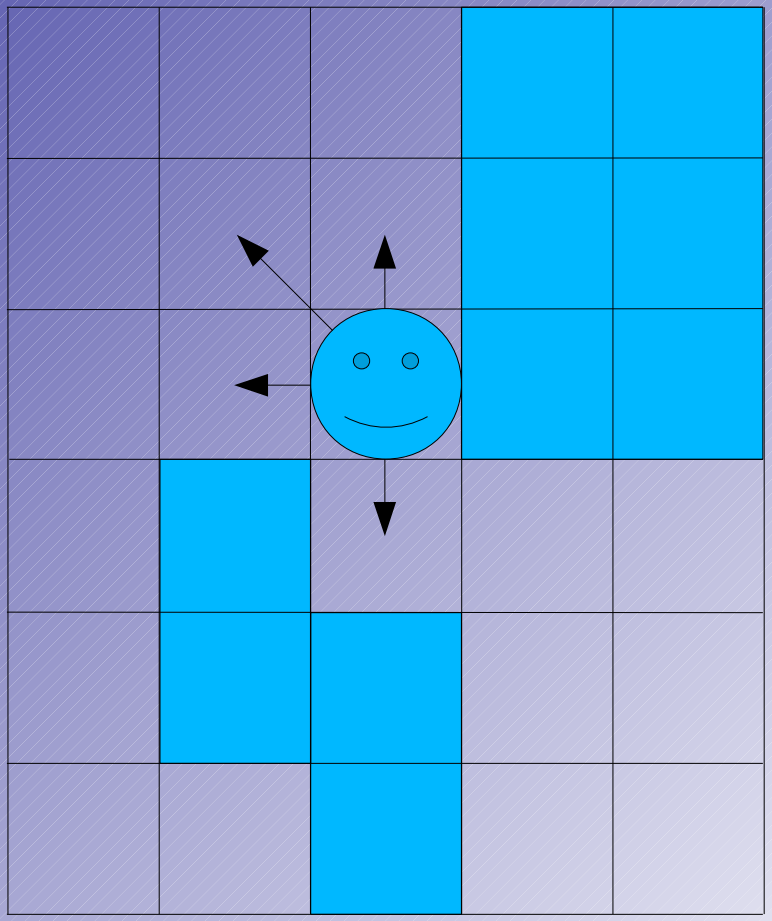
- Want to get some object from one point to another
- Robotics: needs to avoid obstacles (by some margin)
- Games: needs to be very fast and use little resources




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Grid-Based Methods

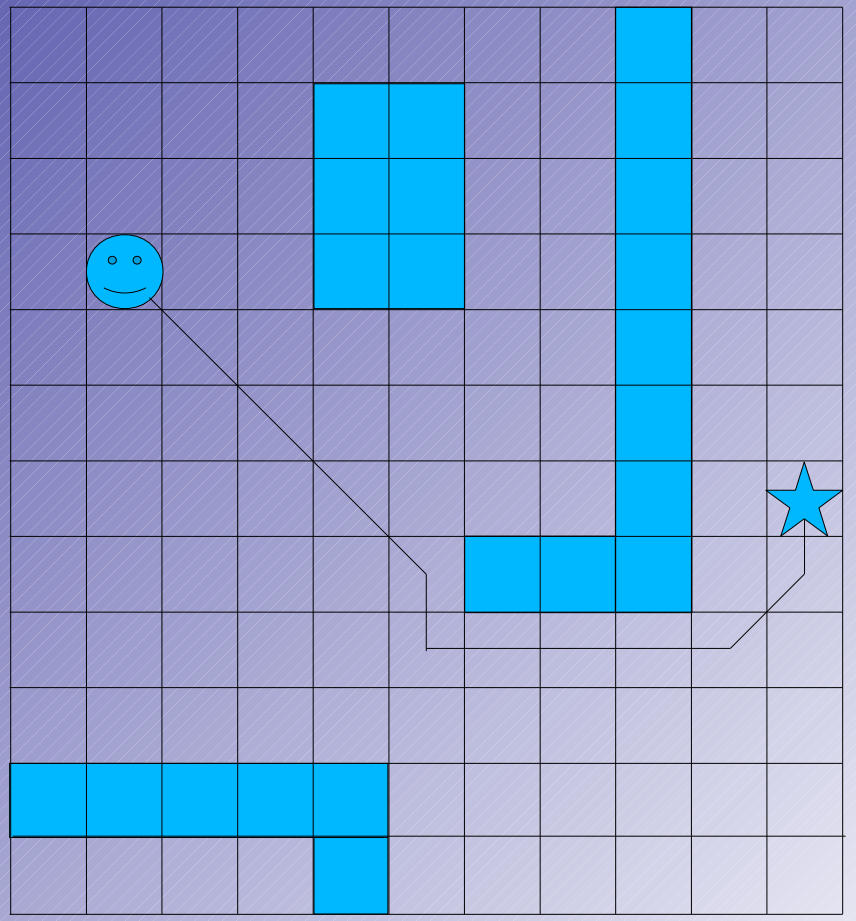
- Represent the environment by a grid of (usually square) cells
- Each cell is either traversible or obstructed
- Object (on a traversible cell) can move to any adjacent traversible cell each move




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Grid-Based Methods: Advantages

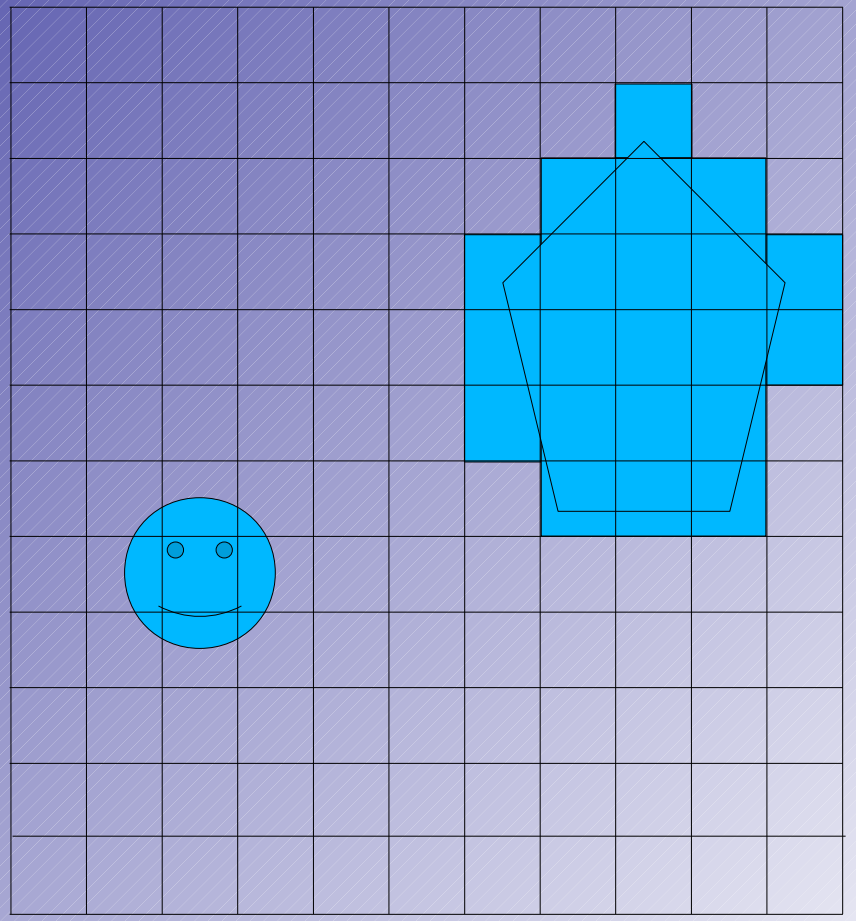
- Conceptually simple representation
- Many existing algorithms
- Perfectly represents tile-based environments
- Exact paths easy to determine for cell-sized objects




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Grid-Based Methods: Disadvantages

- Imprecise representation of arbitrary barriers
- Increased precision in one area increases complexity everywhere
- Awkward for objects that are not tile-sized and shaped
- Does not represent the structure of the environment




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

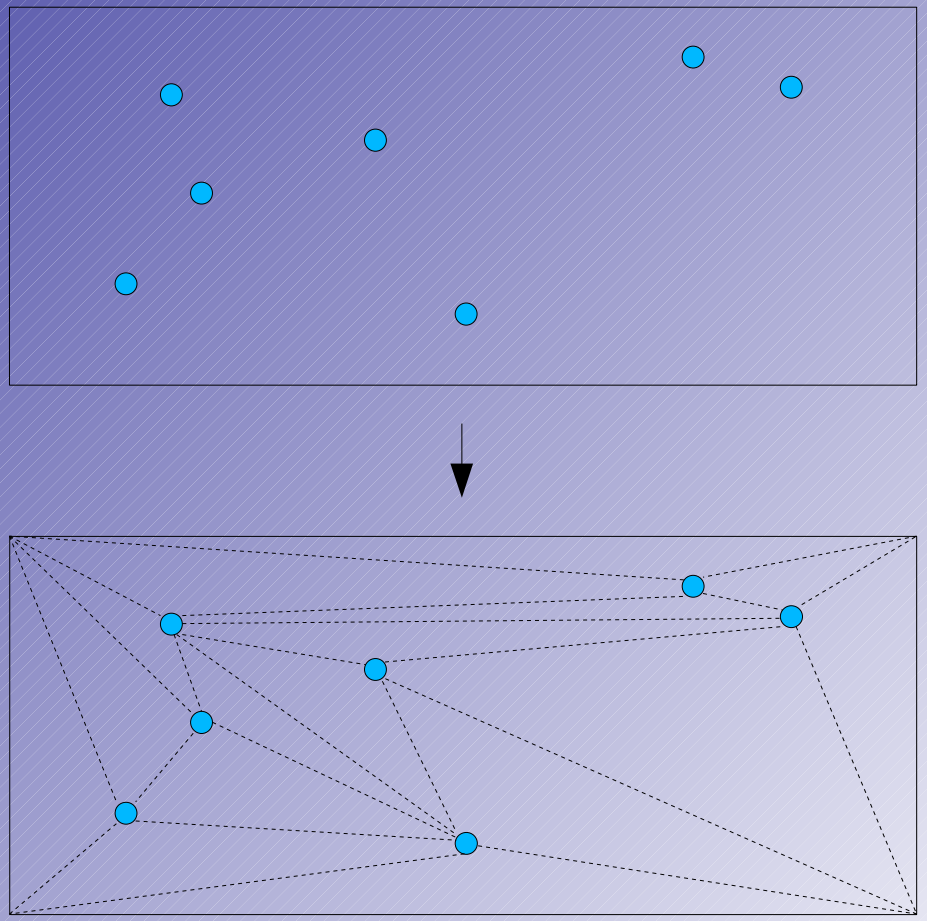
# Motivation: ORTS


- Open Real-Time Strategy engine needed a fast pathfinder to deal with its unique requirements:
  - Straight segment obstacle barriers both axis-aligned and on diagonals
  - Objects with circular footprints of various size
  - Considerations for multiple objects


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Introduction to Triangulations

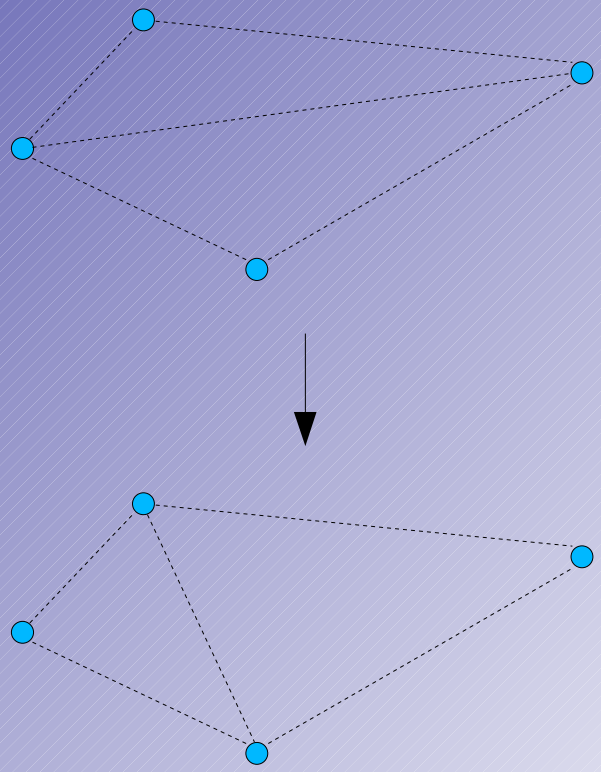
- Another way to represent the environment is with a triangulation
- Starting with an area (like a rectangle) and a collection of points
- Add edges between the points without such edges crossing
- Continue until no more such edges can be added




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Delaunay Triangulations

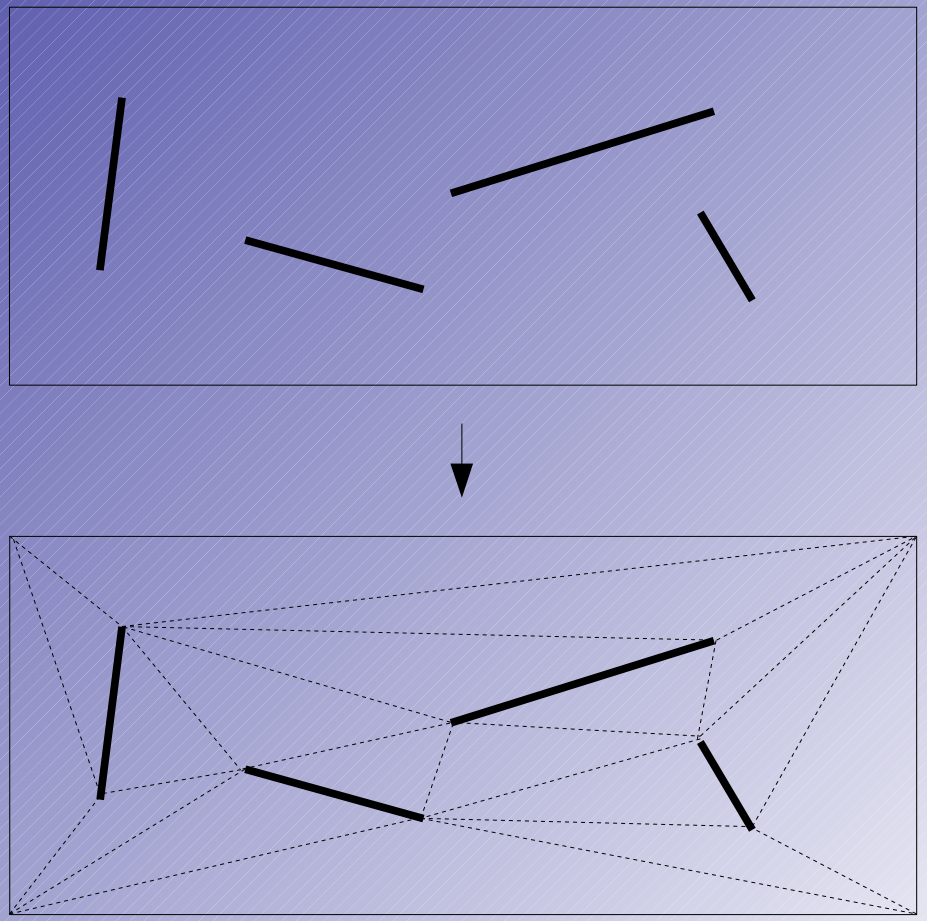
- Triangulations where the minimum interior angle of all triangles is maximized
- Makes “nice” triangulation, tends to avoid thin, sliver-like triangles
- Can be done locally by “edge flipping” diagonals across quadrilaterals




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Constrained Triangulations

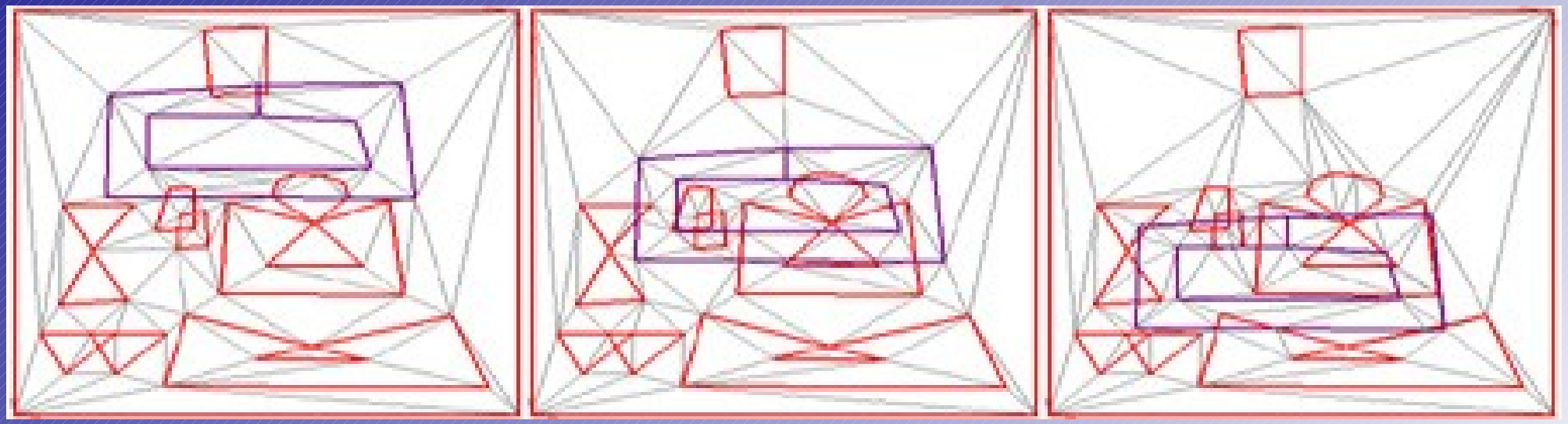
- Triangulations where certain (constrained) edges are required to be in the triangulation
- Then other (unconstrained) edges are added as before
- Constrained Delaunay Triangulations maximize the minimum angle as much as possible while keeping constrained edges




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Dynamic Constrained Delaunay Triangulations (DCDT)

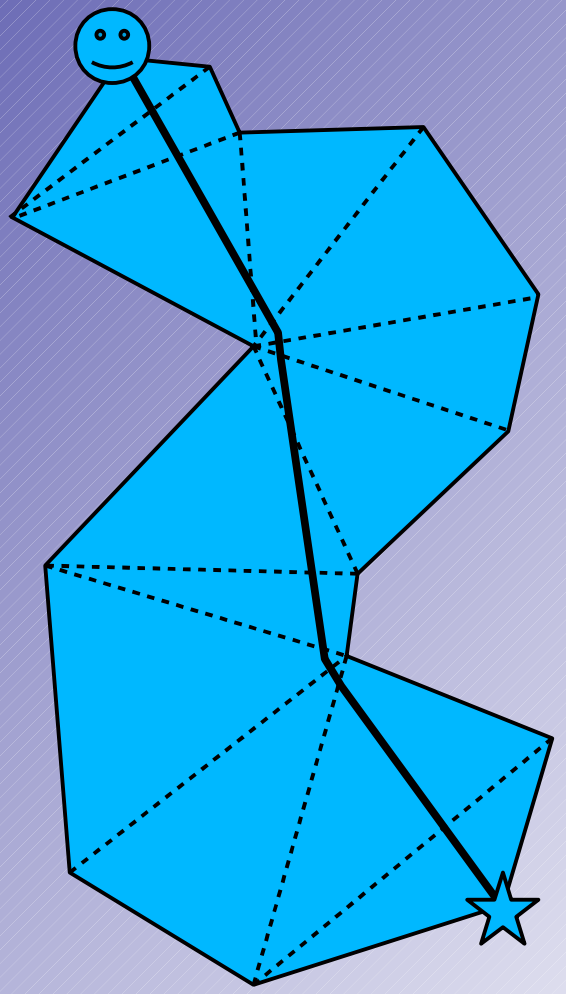
- Marcelo Kallmann's DCDT software can repair a triangulation dynamically when constraints change
- Repairs can be made using local information allowing it to work in a real-time setting




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangulation-Based Pathfinding

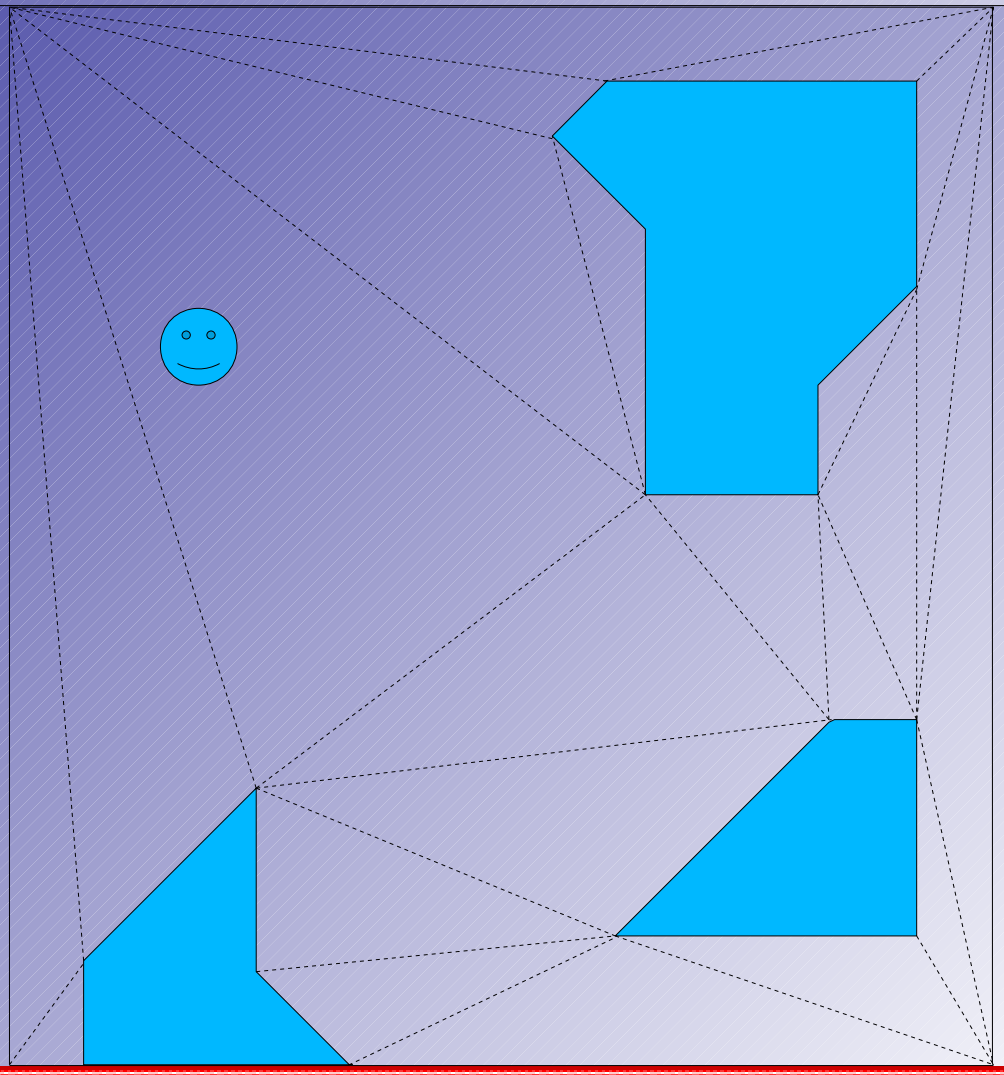
- Using a constrained triangulation with barriers represented as constraints
- Find which triangle the start (and goal) point is in
- Search adjacent triangles across unconstrained edges
- Finds a *channel* of triangles inside which we can easily determine the shortest path




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangulation-Based Pathfinding: Advantages

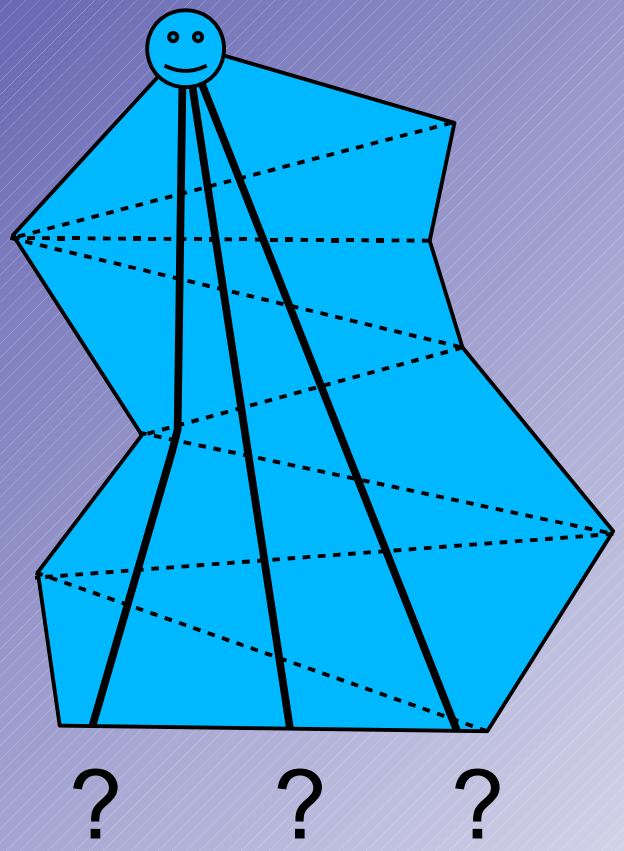
- Remedies grid-based methods' deficiency with off-axis barriers
- Representing detailed areas better doesn't complicate “open” areas
- Triangulations have much fewer cells and are more accurate than grids
- Advantages for multiple object pathfinding




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangulation-Based Pathfinding: Disadvantages

- Curved obstacle barriers must be approximated by straight segments
- We do not know what path we will take through the triangles until after we have found the goal
  - Can lead to either suboptimal paths or multiple paths to nodes

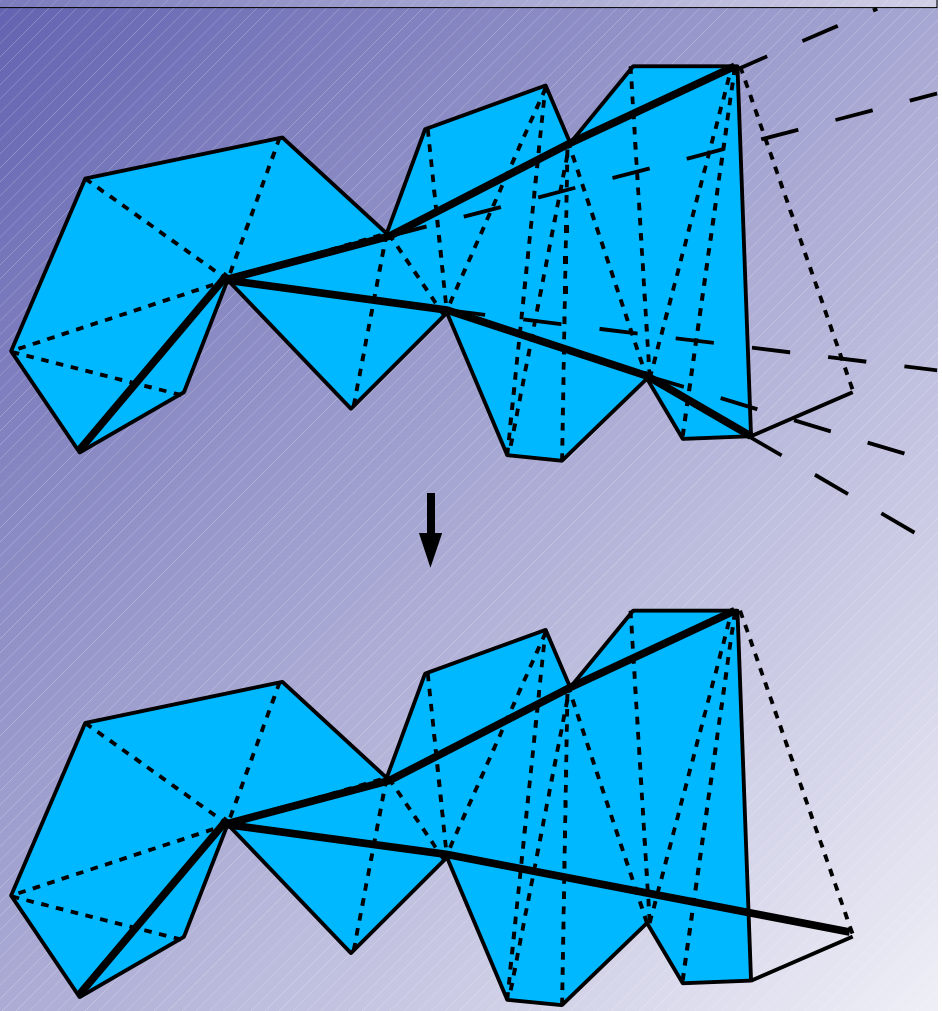




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Funnel Algorithm

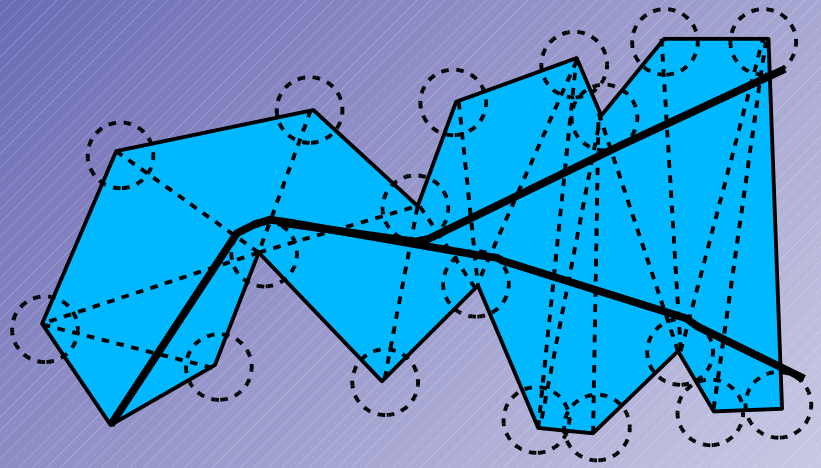
- To find the exact path through a channel of triangles, we use the *funnel algorithm*
- Finds the shortest path in this simple polygon in time linear in the number of triangles in it
- Maintains a *funnel* which contains the shortest path to the end of the channel so far
- Funnel is updated for each new vertex in the channel




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Modified Funnel Algorithm

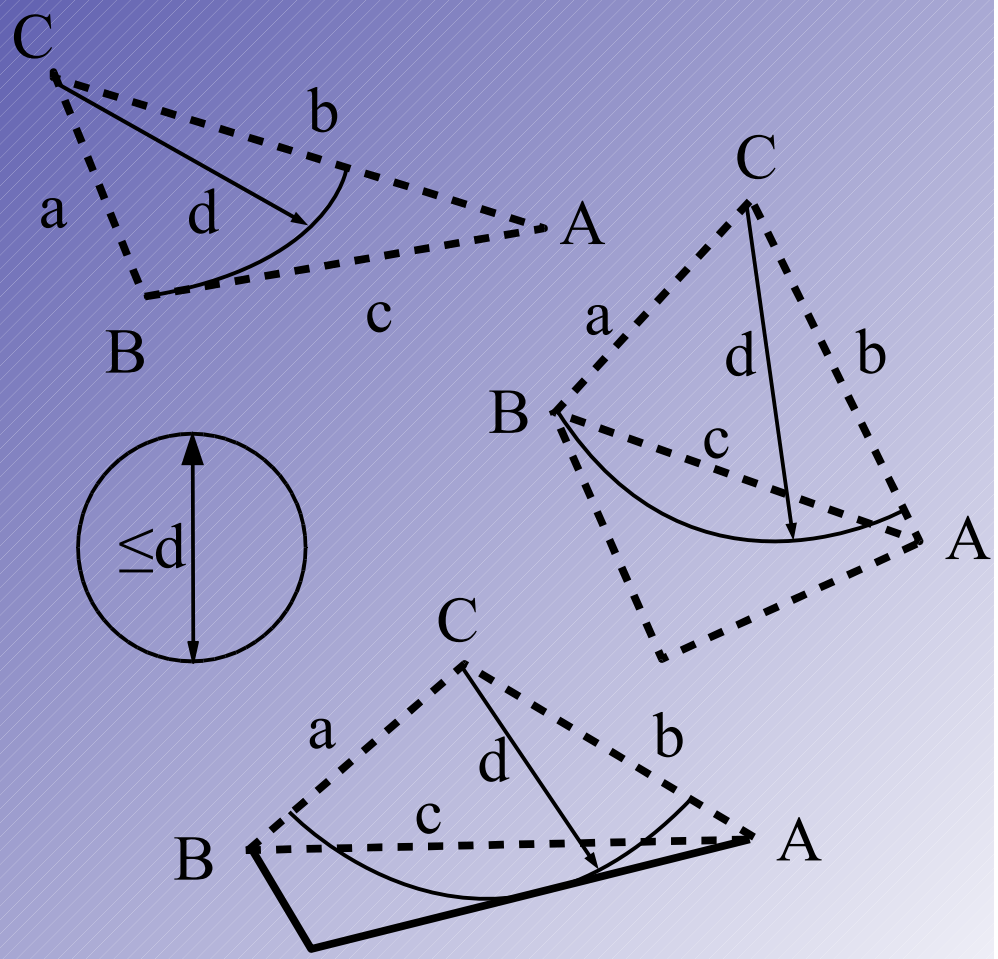
- For circular units of nonzero radius
- Conceptually attach circles of equal radius around each vertex of the channel
- Consider segments tangent to these circles and arcs along them




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangle "Width"

- How do we know when an object of some radius can go through a channel?
- Measure the triangle width
- The distance *d* to the closest obstacle to vertex *C* between edges *a* and *b* is the diameter of the largest object that can fit through between those two edges

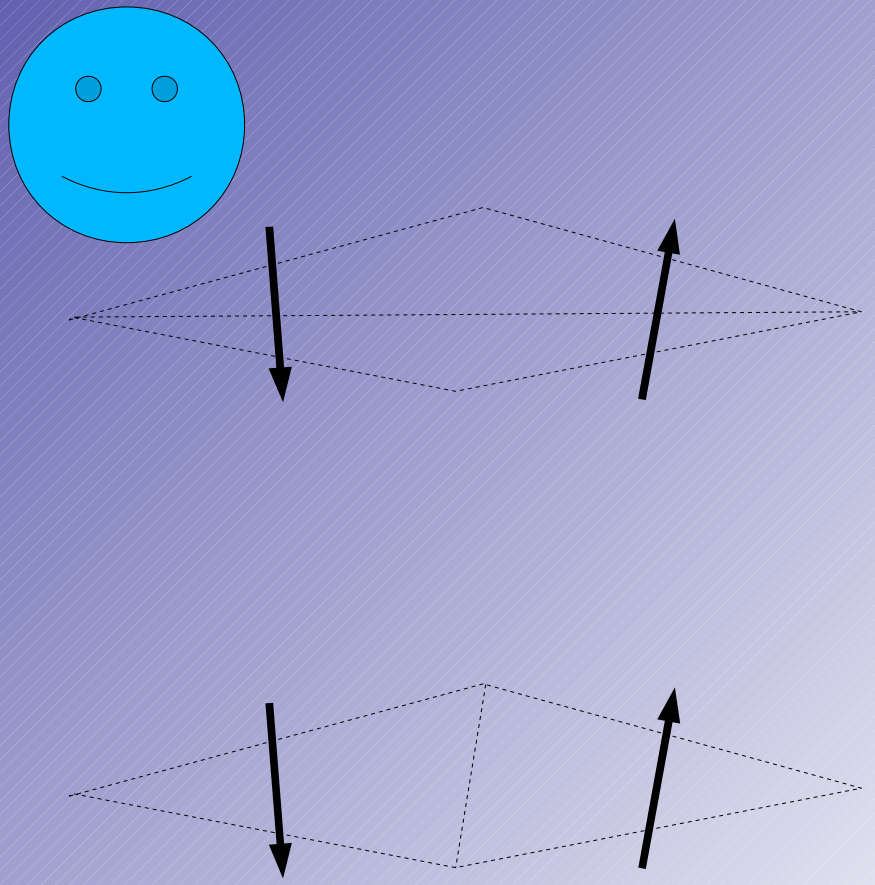




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Delaunay Property

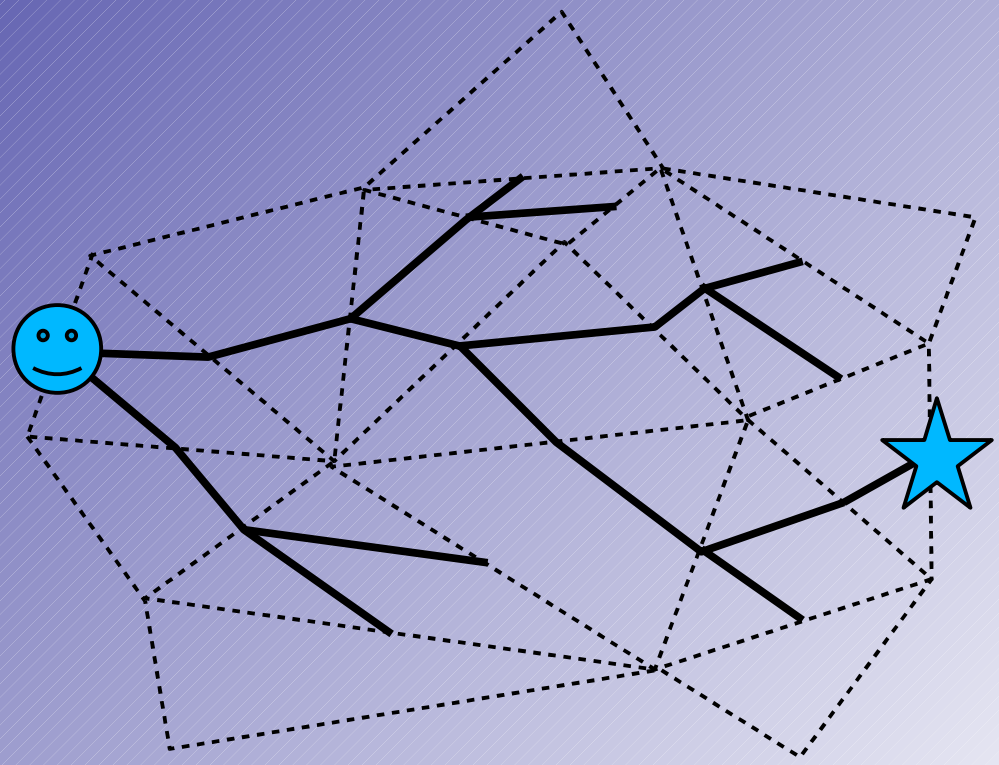
- The Delaunay property allows us to eliminate paths going through a triangle multiple times
- An optimal path will never traverse a triangle more than once




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# "Naïve" Search

- Assume, while searching, that we know the exact path through the triangles
- Use this to prune nodes
- For example, assume straight-segment paths between edge midpoints
- May result in suboptimal paths but finds the solution quickly




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Optimality Concerns

- What do we have to do to find an optimal path?
- (Under)estimate the distance travelled so far
- Allow multiple paths to any node
- When a channel is found to the goal, calculate the length of the shortest path in this channel
- If it is the shortest path found so far, keep it, otherwise, reject it (anytime algorithm)
- When the distance travelled so far for the paths yet to be searched exceeds the length of the shortest path, the algorithm ends and we have an optimal path


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangulation A* (TA*)

- A* running on the base triangulation
  - Using a triangle for a node and generating the adjacent triangles across unconstrained edges as its children
- Using anytime algorithm and considering multiple paths to a triangle as described earlier
- For a heuristic (h-value), take the Euclidean distance between the goal and any point on the triangle's entry edge
- To make sure at least one path is found quickly, don't expand a node from a triangle that's already been expanded, until the first solution is found


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

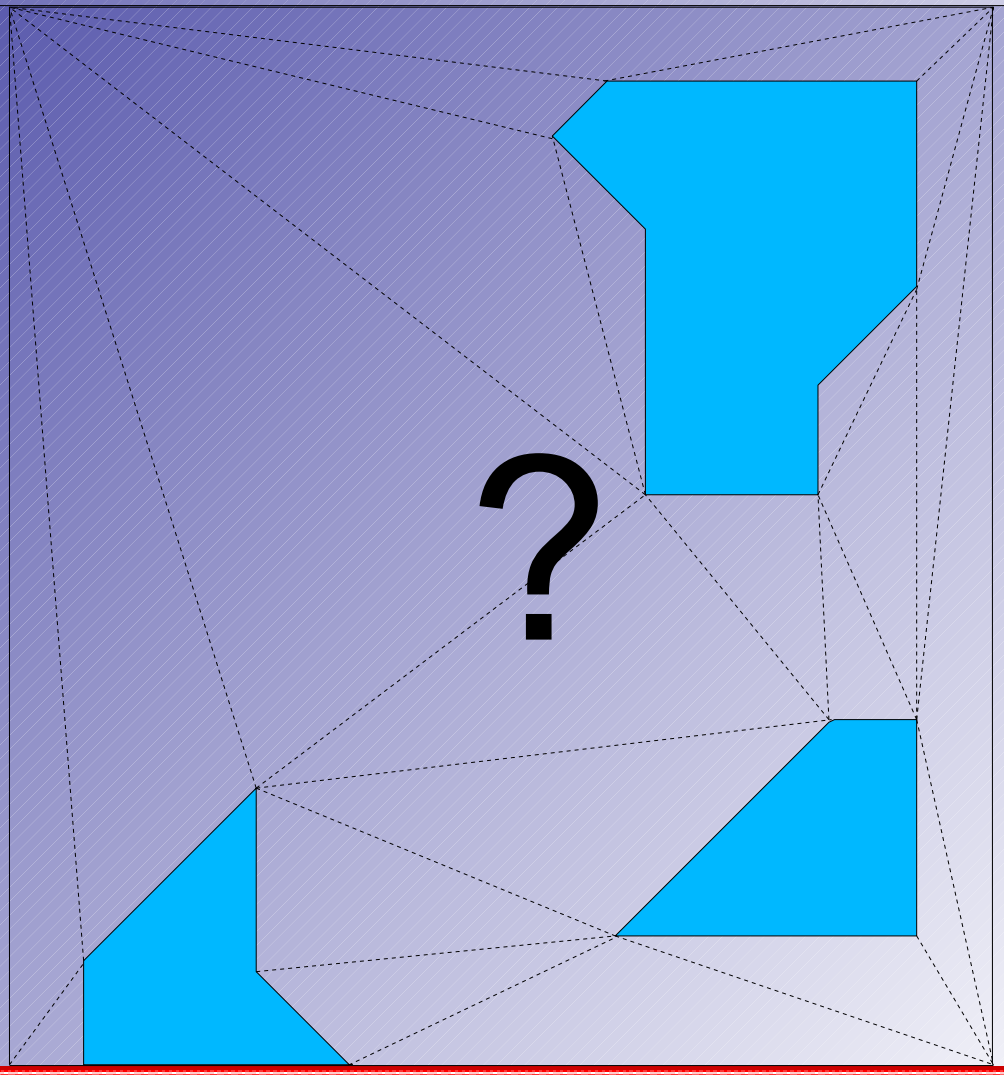
# Triangulation A* (TA*) Cont'd

- Calculate an underestimate for the distance-travelled-so-far (g-value) as the maximum of the underestimates:
  - The euclidean distance from the start to a point on the entry edge (the edge by which this node's triangle was entered)
  - The euclidean distance between the start and goal minus the node's heuristic (h-value)
  - The parent node's g-value plus the difference between its h-value and that of this node
  - The parent node's g-value plus the least distance between its entry edge and that of the current triangle


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangulation Abstraction

- Want to abstract the triangulation without losing its topographic structure
- Determine triangles as being decision points, on corridors, or in dead ends
- Define a degree-*n* triangle as one with exactly 3-*n* triangles adjacent across unconstrained edges that are not degree-1

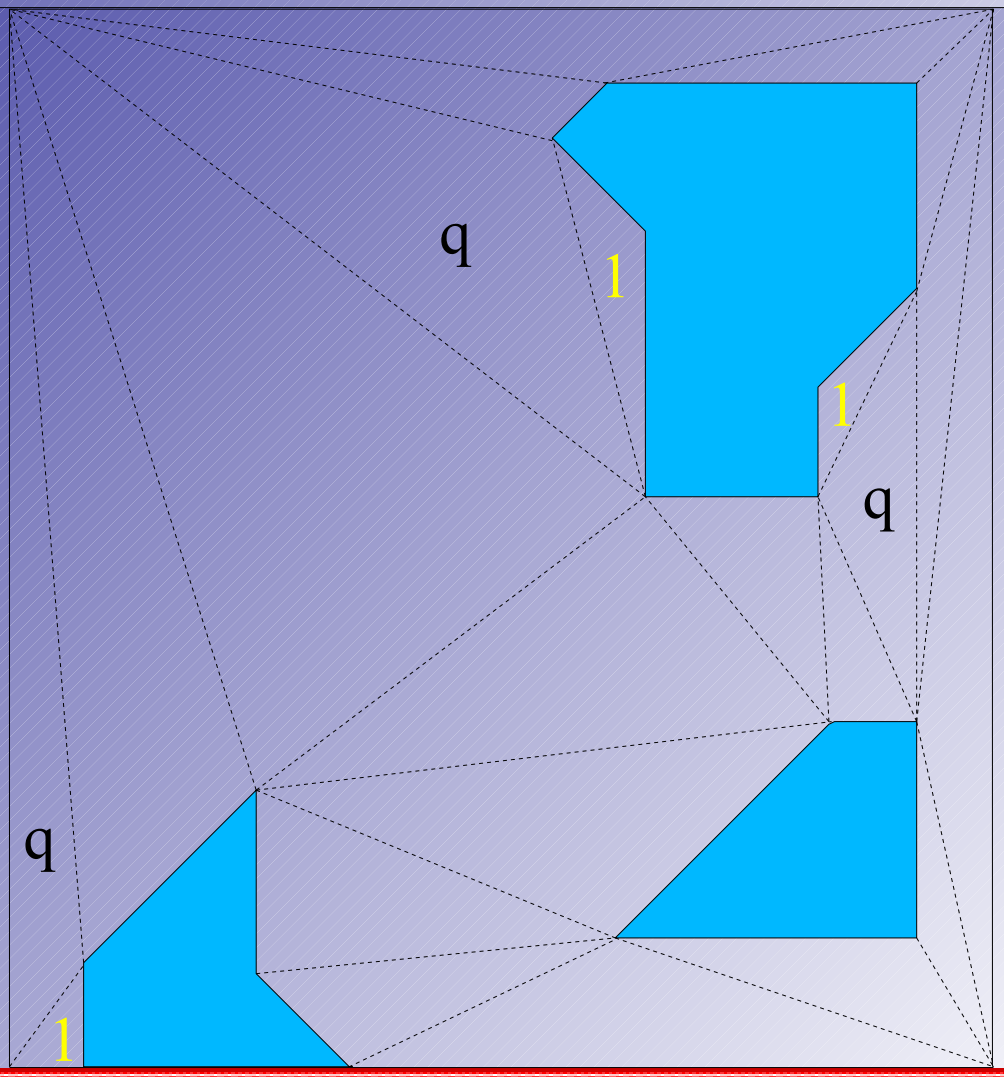




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Abstraction Algorithm

- Abstract triangles with 3 constrained edges as degree-0
- Abstract triangles with 2 constrained edges as degree-1
- Put the triangle adjacent the unconstrained edge on a queue

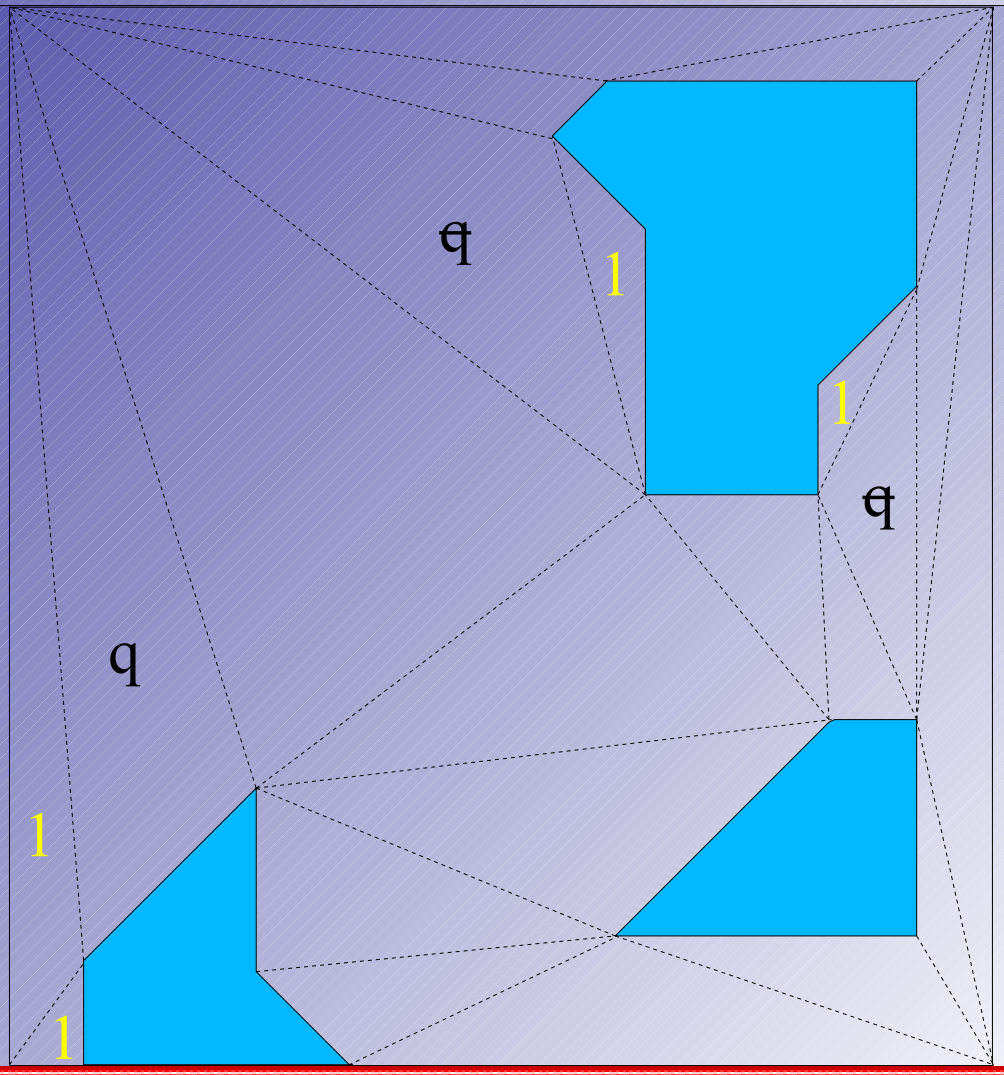




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Abstraction Algorithm, Cont'd

- Go through the queue
  - If the triangle is now degree-1, abstract it as one
  - And put the unabstracted face across the unconstrained edge onto the end of the queue
  - Otherwise, just remove it
- Sometimes a connected component is “collapsed” into all degree-1 triangles

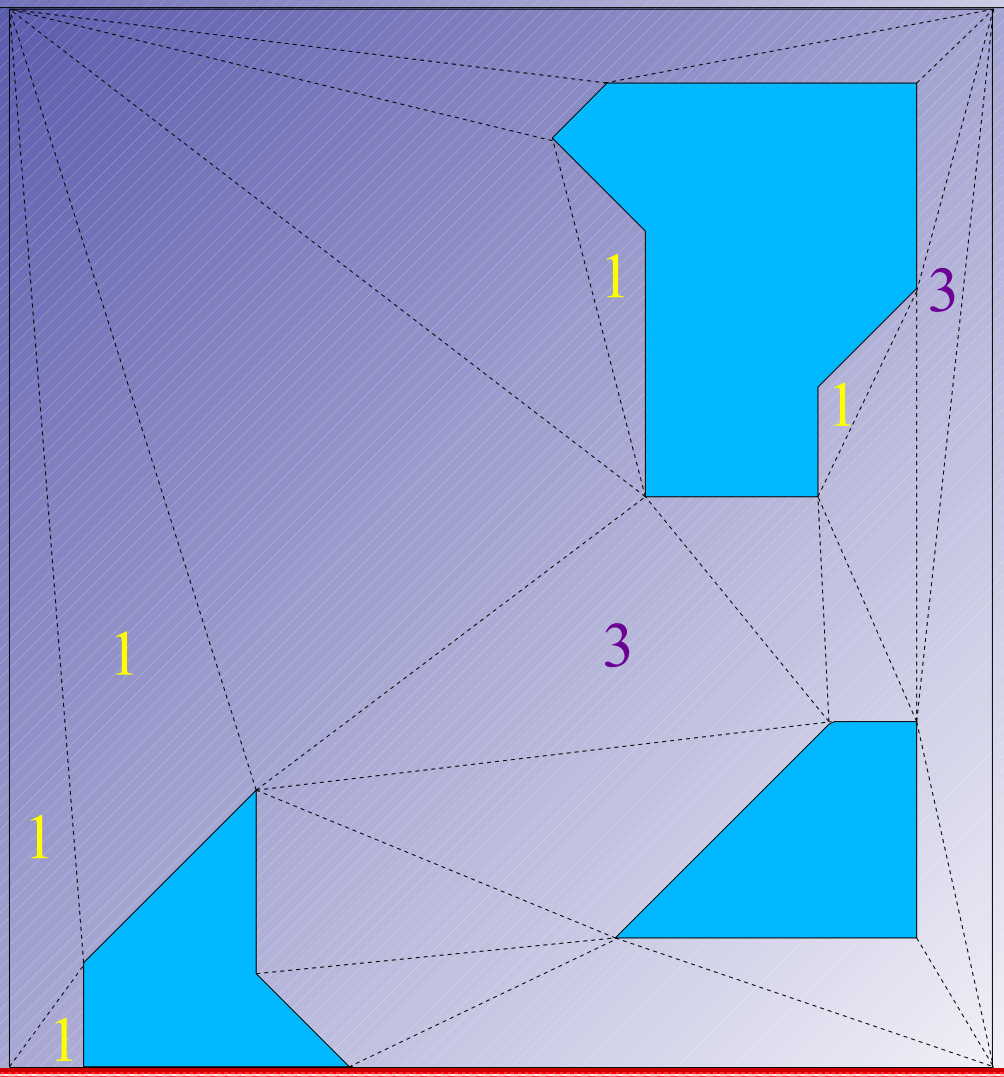




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Abstraction Algorithm, Cont'd

- Go through the other triangles
- Determine which ones have neither constrained edges nor adjacent degree-1 triangles
- Abstract these as degree-3
- There are $2n - 2$ for a component with $n$ obstacles

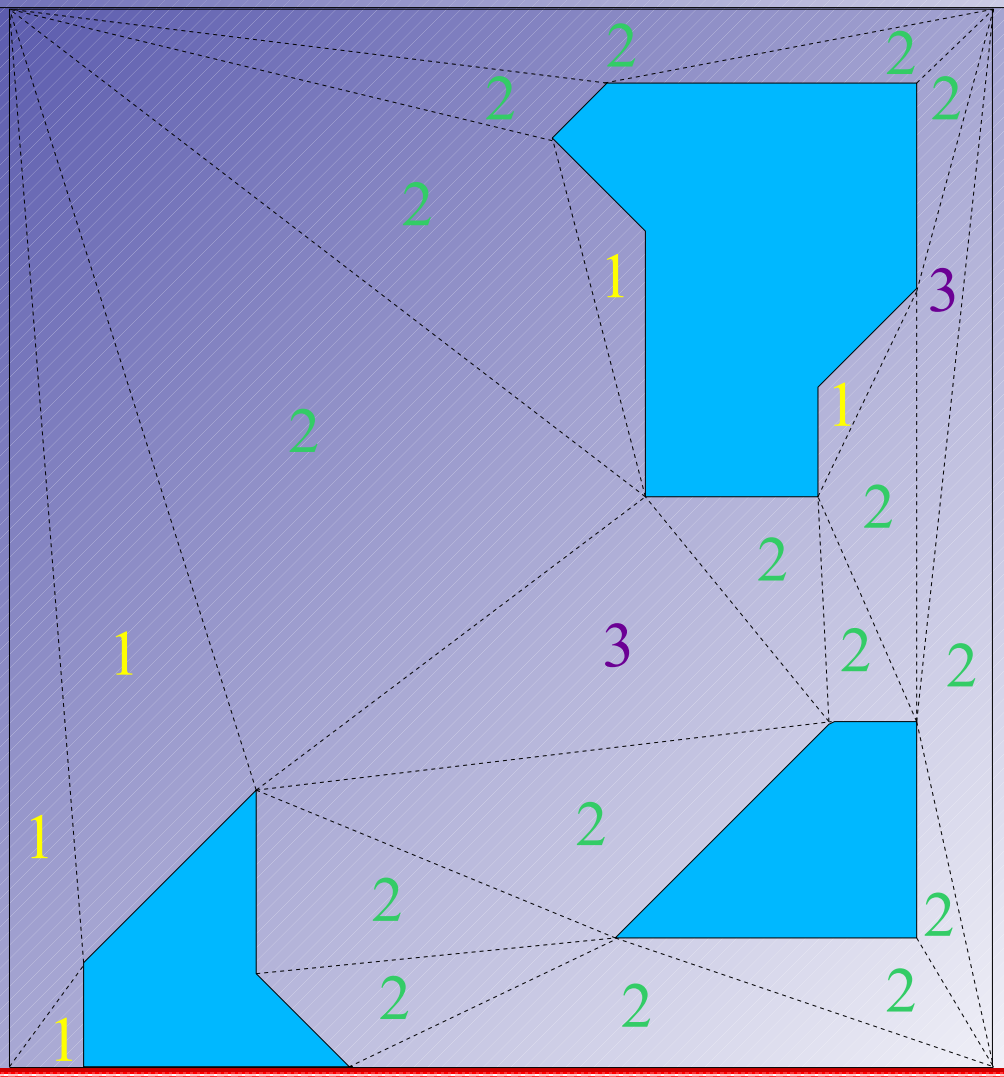




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Abstraction Algorithm, Cont'd

- From degree-3 triangles, move through the corridors of unabstracted triangles to the next degree-3 triangles
- Abstract these triangles as degree-2
- If there are still any unabstracted nodes, abstract them into one or more “rings” of degree-2 triangles

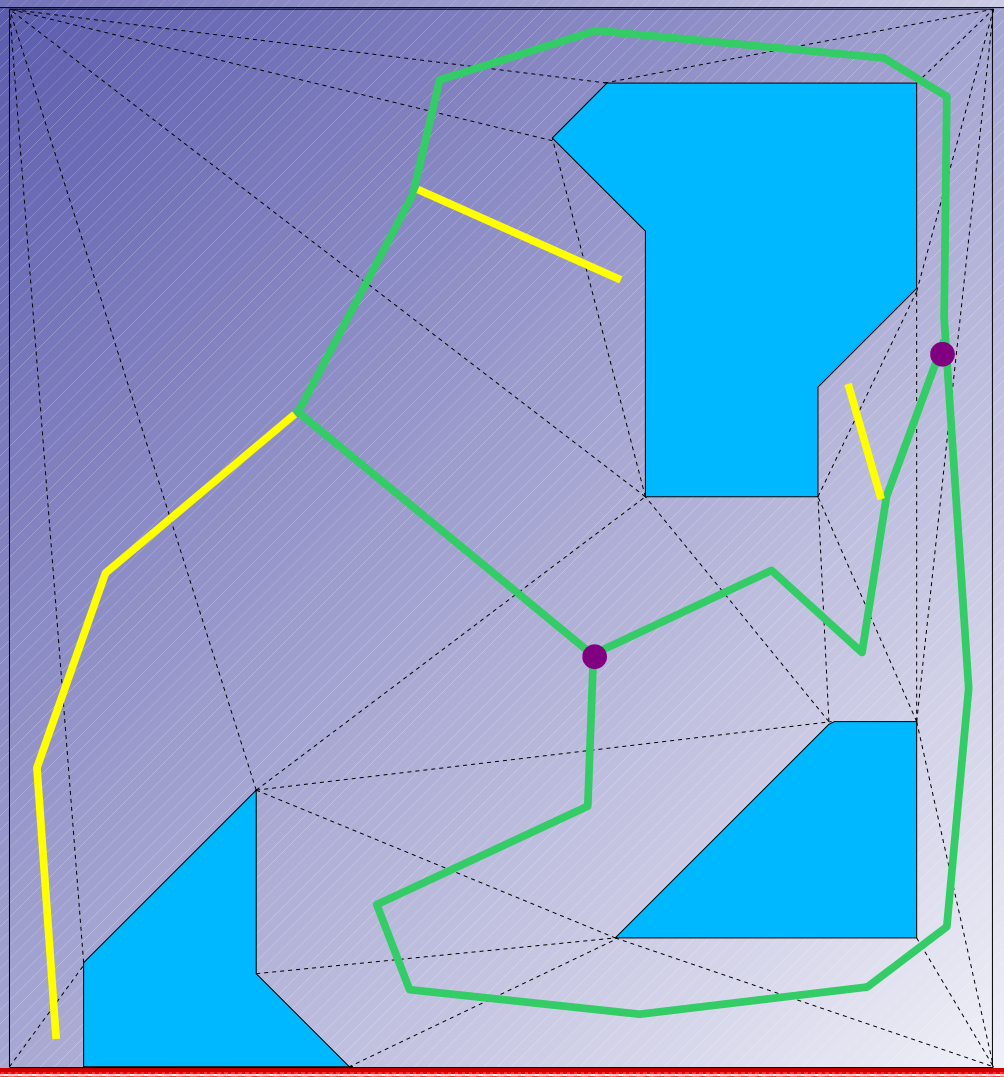




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Abstraction Information

- A triangle's abstraction contains information about:
  - Adjacent structures
  - Choke points (the narrowest point between this triangle and the adjacent structure)
  - A lower bound on the distance to each adjacent structure
  - The triangle “widths”




{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangulation Reduction A* (TRA*)

- A* running on the abstraction described before
  - Using degree-3 triangles as nodesand generating their children as the degree-3 nodes adjacent across corridors
  - If the goal is on a degree-3 triangle, we stop at the goal's triangle, if not, we use the degree-3 triangles at the ends of the corridor the goal (or the root of the tree containing it) is on
  - Similarly the search queue is initialized with the starting point's triangle if it is degree-3 or the adjacent degree-3 triangles otherwise
- As with TA*, use the same anytime algorithm, allow multiple paths to a node, and use the same g- and h-values


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Triangulation Reduction A* (TRA*) Cont'd

- There are a number of cases for which no search occurs:
  - If the start or goal is the root of a tree containing the other, we can “walk” to the root for the only path
  - If they are on the same degree-2 ring or “loop”, we walk both ways around and pick the shortest path
  - If they are in the same degree-1 tree, we can do a simple search (eliminating suplicates) in that tree to find the only path
  - If they are on the same degree-2 corridor, we take one path by walking through the corridor and start searching from the endpoints to see if we can find a shorter one
- If none of these hold we perform actual degree-3 search


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Conclusions

- Triangulations can accurately and efficiently represent polygonal environments
- Triangulations offer unique possibilities for pathfinding for a non-point (especially circular) object
- Triangulation-based pathfinding finds paths very quickly and can also find optimal paths given a bit more time
- Our abstraction technique identifies useful structures in the environment: dead-ends, corridors, and decision points
- This abstraction can be used to find paths even more quickly, only depending on the number of obstacles


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Future Work

- Further abstraction is possible by collapsing doubly-connected components of the abstract graph into single nodes of an even more abstract graph, forming trees:
  - Would identify “rooms” in the environment
  - Pathfinding in the tree is trivial, and paths between entry points of the components could even be cached
- Channels resulting from triangulation pathfinding are useful in pathfinding involving multiple objects
- Terrain analysis is possible with the abstraction information
- Some enhancements for grids could work on triangulations


{demyen|mburo}@cs.ualberta.ca
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
UNIVERSITY OF ALBERTA

# Questions?

Thank you for your attention


{demyen|mburo}@cs.ualberta.ca

Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8

UNIVERSITY OF ALBERTA