# Shortest Paths among Obstacles in 2D(problem 21)

Shuvra Kanti Nath

*Parasol Lab, Texas A&M University*

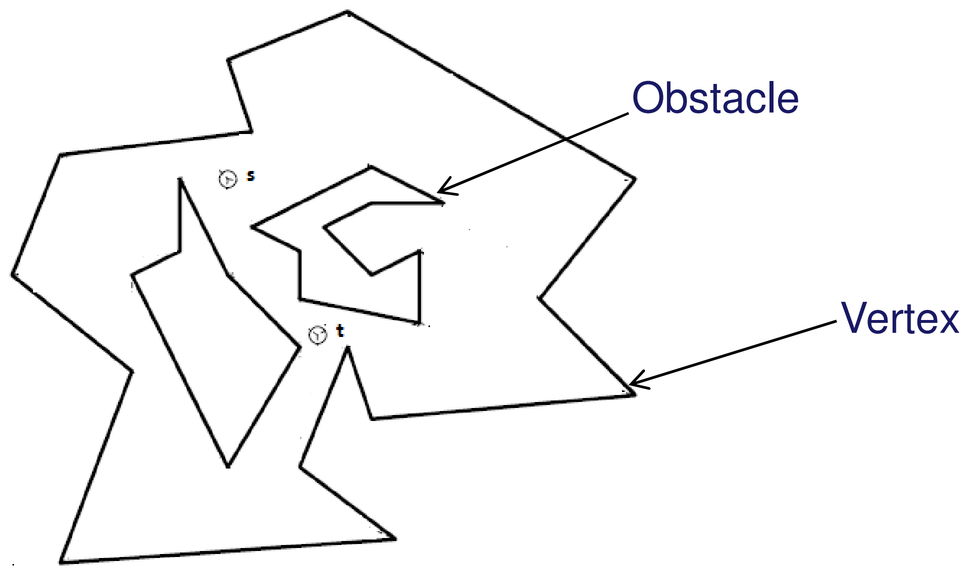
http://maven.smith.edu/~orourke/TOPP/P21.html#Problem.21

Parasol
Smarter computing.
Texas A&M University

http://parasol.tamu.edu

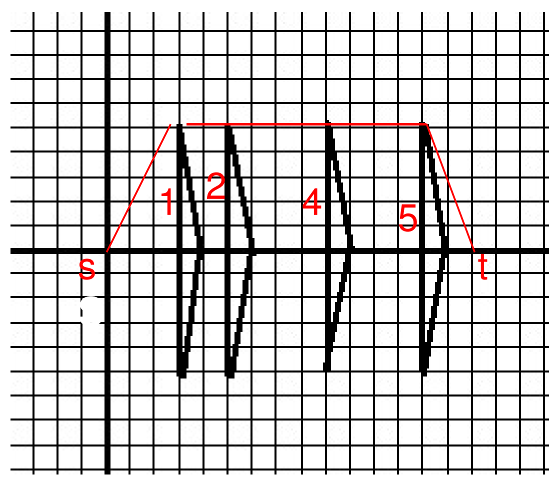
# Problem description

Can shortest paths among *h* obstacles in the plane, with a total of *n* vertices, be found in optimal $O(n + h \log h)$ time using $O(n)$ space?



# Proof of lower bound Ω(n+hlogh) (Reduction from sorting)

- Sorting to shortest path reduction example:
- Problem: Sort the numbers 5, 2, 1, 4.




Point x ->triangle(
(2x+1,c)

(2x+1,-c)

(2x+2,0)

Here c=5 )

Source s(0,0)

Target t($2x_{max}$+3,0)

c is arbitrary +ve number

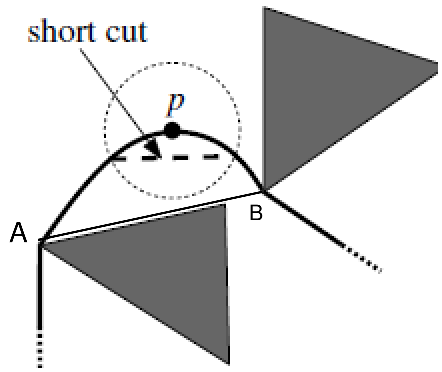
- Mapping from sorting to shortest path problem takes O(h) time for h points
- Shortest path calculation from s to t O(?) time
- Transform back from shortest path to sorted points O(h) time.
- Lower bound of sorting is O(hlogh)
- Reading the polygon co-ordinates take O(n) . So, lower bound is Ω(n + hlogh)

# Application

- Robotics
- Geographic Information Systems
- Shipping/distribution problem
- VLSI design/wire routing
- Military mission planning
- Regional planning
- Game development

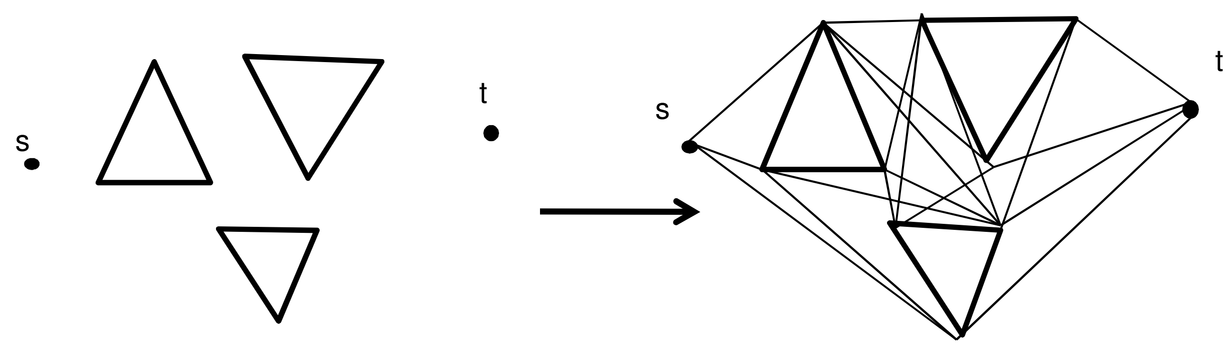
# Characteristics of a shortest path

- **Theorem:** Any shortest path between *s* and *t* among a set *S* of disjoint polygonal obstacles is a polygonal path whose inner vertices are vertices of *S*.



# Visibility Graph

- Visibility: Two vertices *v* and *w* are mutually visible if *vw* does not intersect the interior of any obstacle; segment *vw* is a visibility edge.
- Visibility Graph V(G): It contains all the nodes of the graph G and all the edges which are visible
- Edges: $O(n^2)$ edges




- Naïve computation: $O(n^3)$
- Rotational plane sweep: $O(n^2 \log n)$
- Output sensitive algorithm :$O(E + n\log n)$ (E is number of edges in graph)

# Shortest path from visibility graph

- 1. Construct visibility graph
- 2 . Shortest path is found by running Dijkstra's shortest path algorithm on the resulting graph
- Dijkstra's algorithm: O(E+nlogn)
- Total running time: $O(n^2 \log n)$ and $O(n^2)$ space ->large

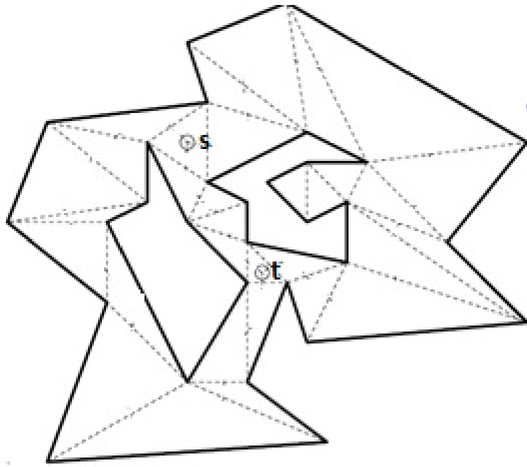
- Improved General Approach:

  1. Contraction of the region/graph.
  2. Visibility graph is computed from the contracted region/graph.

# Related work

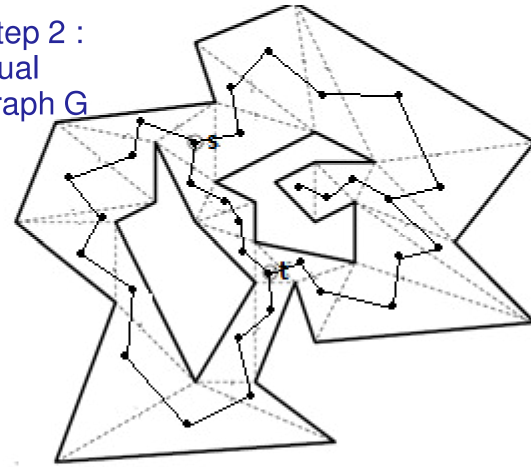
- S. Kapoor, S. N. Maheshwari, and Joseph S. B. Mitchell,1997
- Approach: Contraction of region by building hourglasses and computing visibility graph from that.
- Running time: $O(n+h^2 \log n)$
- Space: $O(n)$

Step 1 : Triangulation

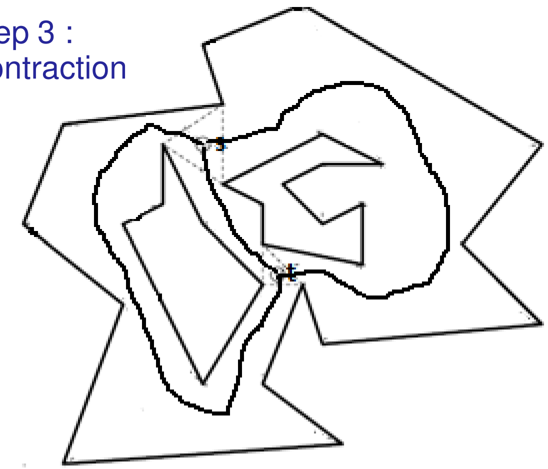



Parasol

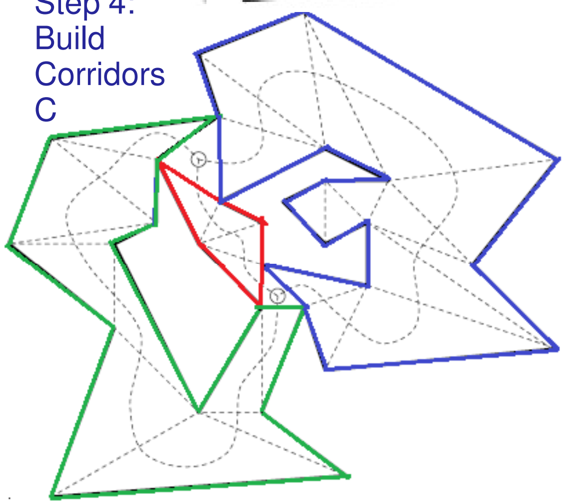
Step 2 : Dual graph G



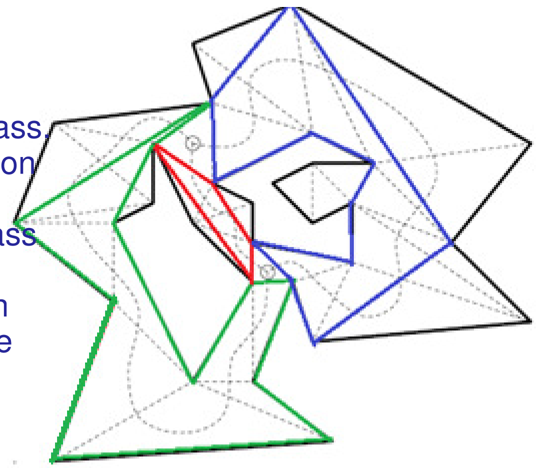

Step 3 : Contraction



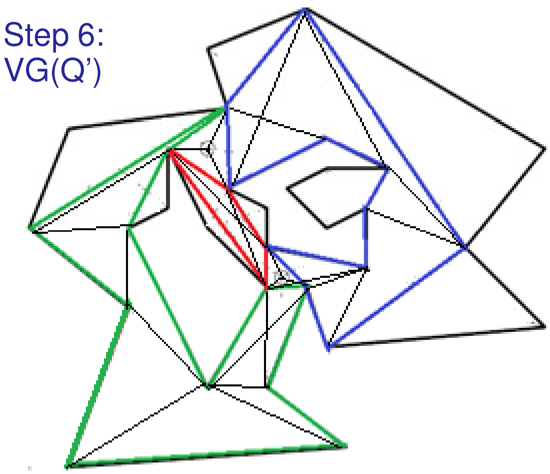

Step 4: Build Corridors C



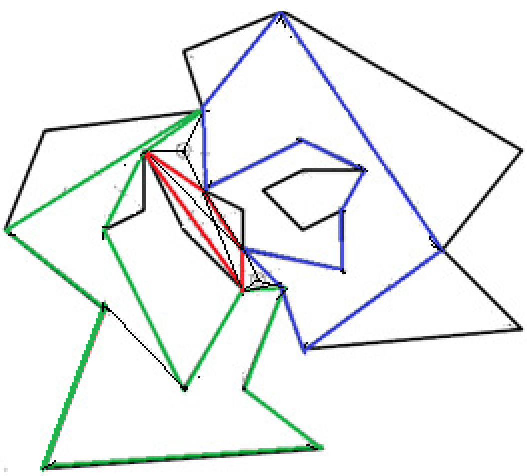
Step 5: Build Hourglass, Q'= union of hourglass and junction Triangle



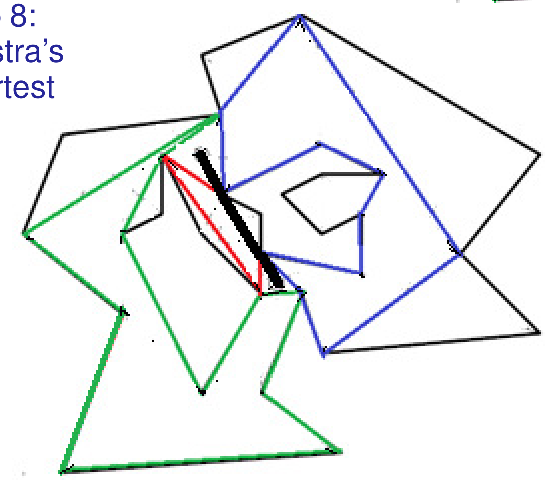
Step 6: VG(Q')



Step 7: Relevant Visibility graph VG'(Q')



Step 8: Dijkstra's Shortest path



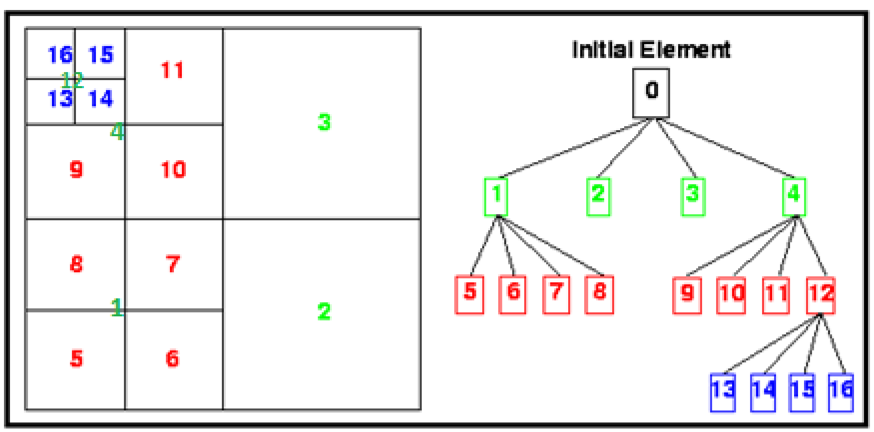
- Running time: Computation of VG'(Q') takes $O(n+h^2 \log n)$ time.
- Triangulation; $O(n+h\log^{1+\varepsilon}h)$ time
- Dijsktra's algorithm: $O(h\log h)$

- Note: One convex polygon has $O(h)$ tangent segment. Any tangent segment can be computed in $O(\log n)$ time. There are $O(h)$ polygons. So, running time is $O(n+h^2 \log n)$.

Parasol

- John Hershberger and Subhash Suri, 1999
- Approach: Contraction of region by quad-tree style subdivision of the region and wavefront propagation on the contracted region.
- Running time: O(nlogn) . Building subdivision considering vertices of obstacles. Inserting obstacle edges into cells. Subdivision computation takes O(nlogn) time.
- Space: O(nlogn)

## Quadtree/Octree Data Structures



- α Conforming subdivision:
- 1) Each point of P is in separate cell
- 2) O(1) cells within distance of α|e| of every subdivision edge e. So, there may be non-obstacle edges(transparent edge) and obstacle edge(opaque edge).
- Well-covering property of internal edges(transparent):

(W1) There exists a set of cells $\mathcal{C}(e) \subseteq \mathcal{S}$ such that $e$ lies in the interior of their union. The union is denoted $\mathcal{U}(e) = \{c \mid c \in \mathcal{C}(e)\}$.
(W2) The total complexity of all the cells in $\mathcal{C}(e)$ is $O(\alpha)$.
(W3) If $f$ is an edge on the boundary of the union $\mathcal{U}(e)$, then the Euclidean distance between $e$ and $f$ is at least $\alpha \cdot \max(|e|, |f|)$.

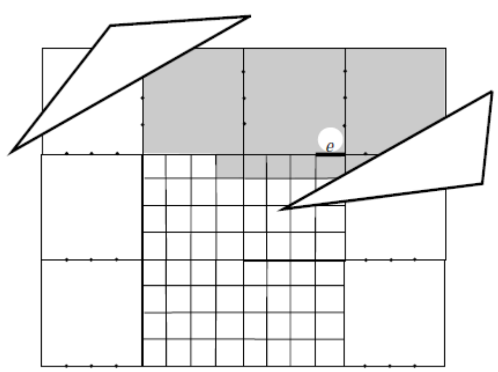



FIG. . *Part of a 1-conforming subdivision of free space. The shaded region is the well-covering region $\mathcal{U}(e)$.*

- Construction of subdivision:
- 1. Consider only the vertices of obstacles
- 2. Insert the obstacle edges
- Subdivision algorithm takes O(nlogn) time.
- Shortest path from conforming subdivision:
- Propagate wavefronts through the cells of conforming subdivision and it can only go through transparent edges.

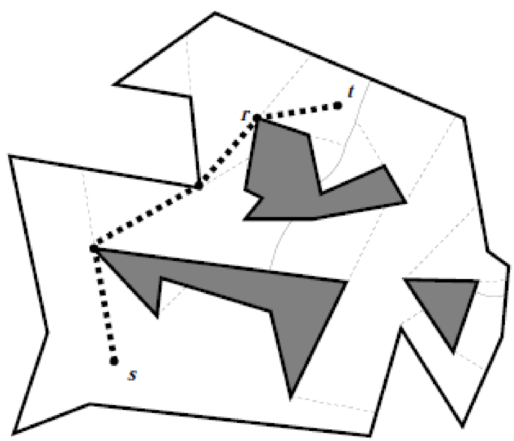



Figure : A shortest path map with respect to source point $s$ within a polygonal domain with $h = 3$. The heavy dashed path indicates the shortest $s$-$t$ path, which reaches $t$ via the root $r$ of its cell.

- Danny Z. Chen, Haitao Wang, March 2011(curved obstacle)
- Approach: Contraction of graph into hourglasses and use “Good pseudo triangulation”
- Running time:O(n+k+hlogh) (k is number of free common tangents). Relevant visibility graph computation O(n + k+hlogh)
- Space: O(n)

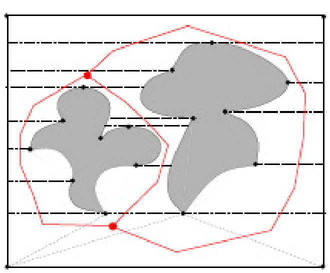


Figure : Illustrating a bounded degree decomposition of $\mathcal{F}$ (with dashed segments) and the corridors (with red solid arcs). There are two junction regions indicated by large (red) points inside them, connected by three solid (red) arcs. Removal of these two junction regions results in three corridors.

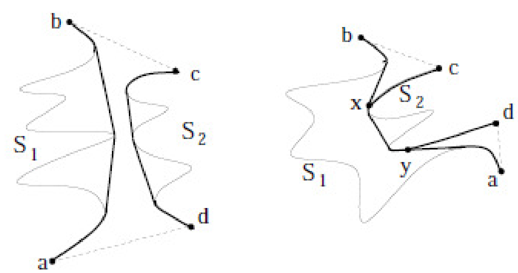



Figure : Illustrating an open hourglass (left) and a closed hourglass (right) with a corridor path linking the apices $x$ and $y$ of the two funnels. The dashed segments are diagonals. The paths $\pi(a, b)$ and $\pi(c, d)$ are shown with thick solid curves.

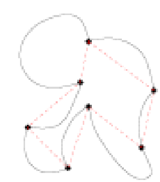


Figure : A splinegon (solid curves) defined on a polygon (red or dashed segments).

- Step 2: Computation of relevant visibility graph by flipping edge of good psuedo triangulation. O(n+k+hlogh)

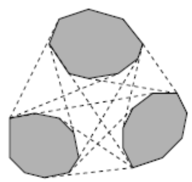


Figure : The relevant visibility graph of three convex objects.

- Compute good psuedo triangulation

Psuedo triangulation: Sub-division of free space by maximum number of bi-tangents(common tangent of two obstacles)

Good psuedo triangulation(T): Triangulation for which there is a way to assign every bi-tangent a direction such that a partial order on the bi-tangents has properties : 1) Partial order corresponds to the point-slope order on B(σ(T) ) with respect to $b_T$ 2) Direction of bi-tangents is compatible with both of its end-points, 3) For any bi-tangent bεB -B(T), all bi-tangents in B(T) intersecting b crosses the directed b from left to right.

B(T) is set of all free bi-tangents of T. σ(T) is boundary of T. $b_T$ has min slope.

- Starts with 1 good psuedo triangulation( construction takes O(n+hlogh) time
- Now flip minimum bi-tangent(smallest slope) at each iteration( k times)

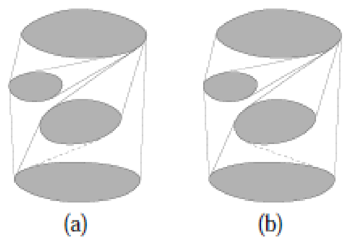



Figure : A flip operation on a free bi-tangent $b$: (a) The dashed bitangent is $b$ before the flip; (b) the dashed bitangent is $\varphi(b)$ after the flip.

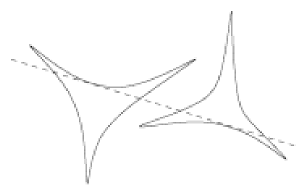


Figure : The common tangent line of two pseudo-triangles.

- Flipping takes O(n+k) time
- So total running time of the algorithm is O( n + k +hlogh)

# Summary

- S. Kapoor, S. N. Maheshwari, and Joseph S. B. Mitchell,1997
  Running time: O(n+$h^2$logn)
  Space: O(n)
- John Hershberger and Subhash Suri, 1999
  Running time: O(nogn)
  Space: O(nogn)
- Danny Z. Chen, Haitao Wang, March 2011(curved obstacle)
  Running time: O(n+k+hlogh)
  Space: O(n)

Achievement of optimal $O(n + h \log h)$ time using $O(n)$ space still remains an open problem.