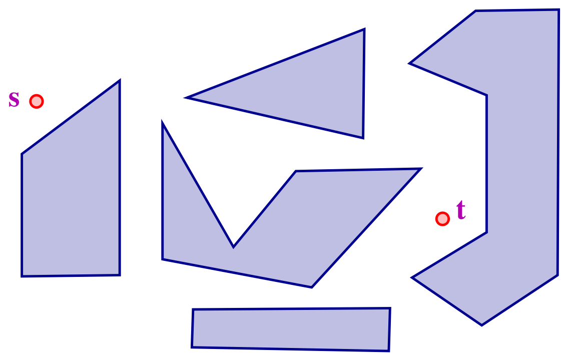
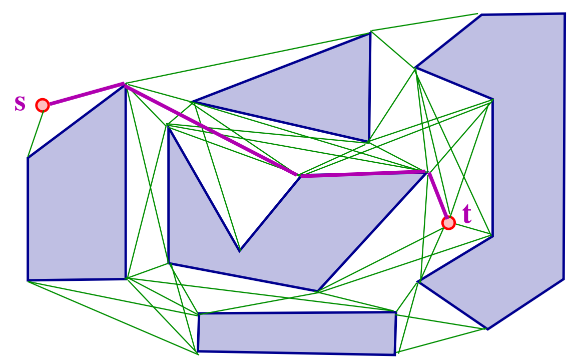
Shortest Paths



- A workspace with polygonal obstacles.
- Find shortest obstacle-avoiding path from s to t.
- Properties of Shortest Path:
  - Uses straight line segments.
  - No self-intersection.
  - Turns at convex vertices only.

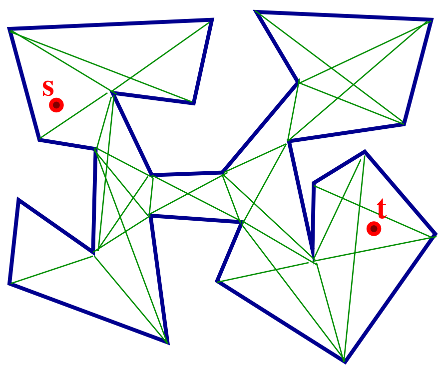
Visibility Graph

- Construct a visibility graph $G = (V, E)$, where V is set of polygon vertices (and s, t), E is pairs of nodes that are mutually "visible".
- Give each edge (u, v) the weight equal to the Euclidean distance between u and v.
- The shortest path from s to t in this graph is the obstacle avoiding shortest path.
- G can have between $c_1 n$ and $c_2 n^2$ edges. Run Dijkstra's algorithm.



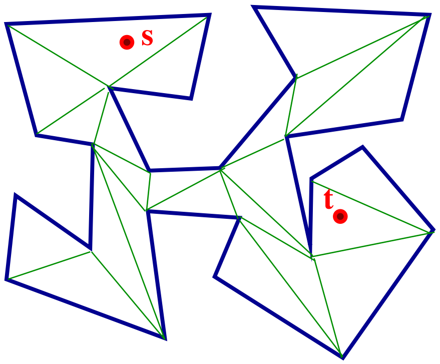
Paths in a Polygon

- Workspace interior of a simple polygon.
- Can we compute a shortest path faster?
- The visibility graph can still have $\Theta(n^2)$ edges.



- Using polygon triangulation, we show an $O(n \log n)$ time algorithm.

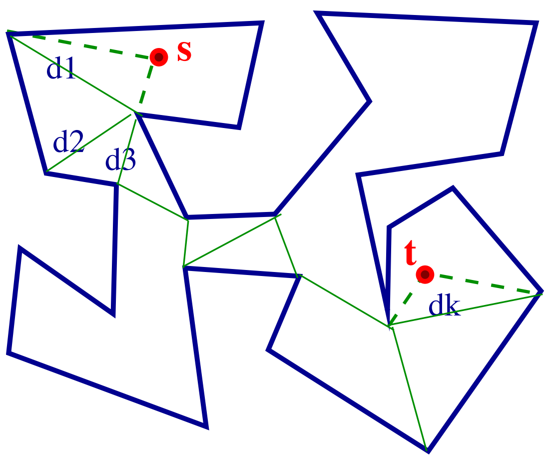
Fast Algorithm



- Let P be a simple polygon and s, t be source and target points.
- Let T be a triangulation of P.
- Call a diagonal d of T essential if s, t lie on opposite sides of d.
- Let $d_1, d_2, \ldots, d_k$ be ordered list of essential diagonal.

Algorithm

- Essential diagonals $d_1, d_2, \ldots, d_k$.

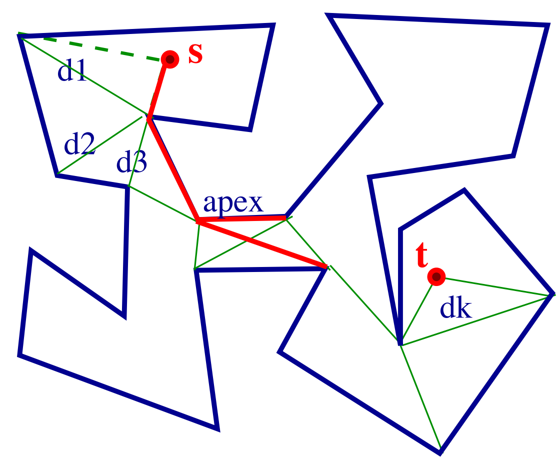


- The algorithm works as follows:

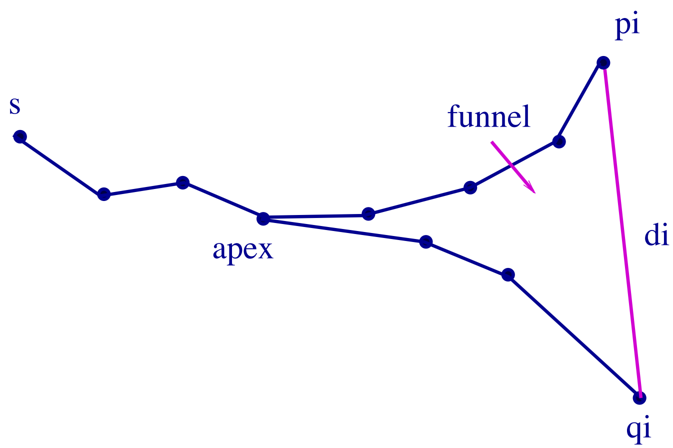
1. Start with $d_0 = s$.
2. for $i = 1$ to $k + 1$ do
3. Extend path from s to both endpoints of d_i

Path Extending: Funnel

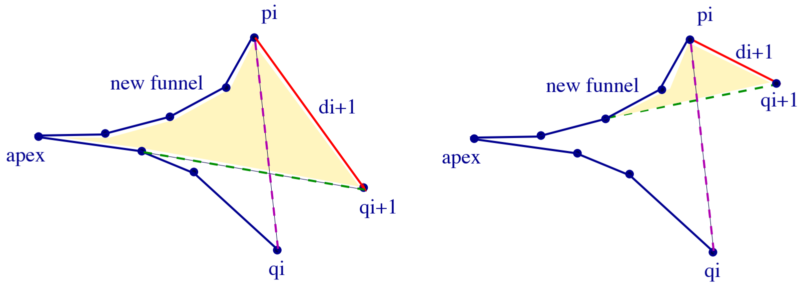
- Union of $path(s, p_i)$ and $path(s, q_i)$ forms a **funnel.**
- The vertex where paths diverge is called **apex.**



Funnel

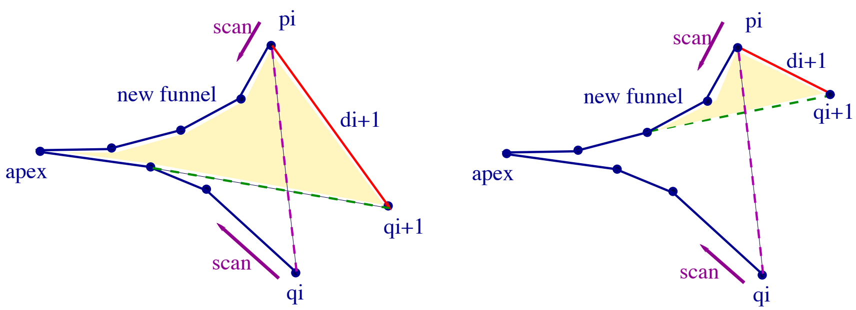


Path Extending



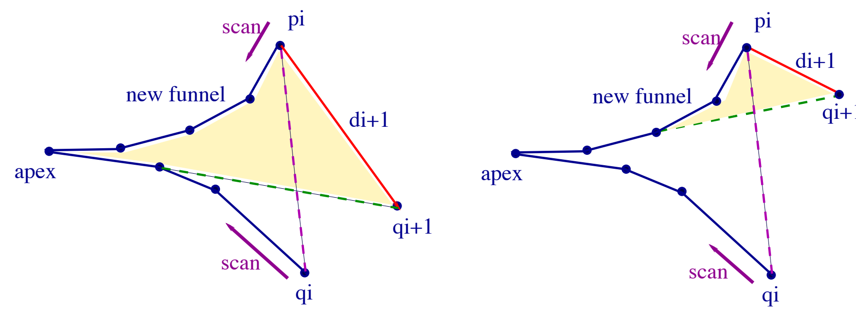
- **Two cases of how to extend the path.**
- **In case I, funnel contracts.**
- **In case II, apex shifts, tail extends, funnel contracts.**
- **In each case, funnel property maintained.**

Data Structure & Update



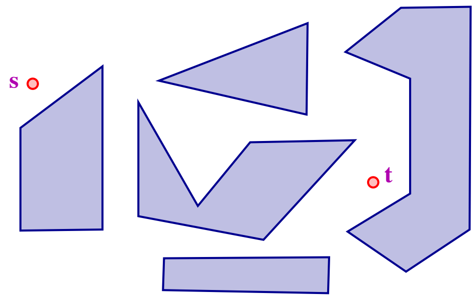
- **How to determine tangent to funnel?**
- **Can't afford to spend $O(n)$ time for each tangent.**
- **Idea: If x edges of funnel are removed by the new tangent, spend $O(x)$ time for finding the tangent.**
- **How to tell a tangent?**

Data Structure & Update



- **Start scanning the funnel from both ends, until tangent determined.**
- **At most $2x + 2$ vertices scanned.**
- **Since each vertex inserted once, and deleted once, total cost for all the tangents is $O(n)$.**
- **Data structure for the funnel: Double-ended queue. Insert/delete in $O(1)$ time.**

Paths Among Obstacles



Approach	Complexity	Reference
Vis. Graph	$O(n^3)$	**L. Perez, Wesley ’79**
	$O(n^2 \log n)$	**Sharir-Schorr ’84**
	$O(n^2)$	**Welzl, AAGHI ’86**
	$O(E + n \log n)$	**Ghosh-Mount ’91**
SP Map	$O(k^2 + n \log n)$	**Kapoor-Maheshwar**
	$O(nk \log n)$	**Reif-Storer ’91**
	$O(n^{5/3+\varepsilon})$	**Mitchell ’93**
	$O(n \log n)$	**Hershberger-S ’93**