An Efficient Algorithm for Computing High-Quality Paths amid Polygonal Obstacles

Pankaj K. Agarwal
Duke University

Kyle Fox
Duke University

Oren Salzman
Tel-Aviv University
Motion Planning

- Robot needs to go from $s$ to $t$
Motion Planning

• Robot needs to go from $s$ to $t$

• What’s the best route?
Many Options

- Shortest path
Many Options

- Shortest path
- Minimum number of links [Mitchell et al. ’92]
Many Options

• Shortest path

• Minimum number of links [Mitchell et al. ’92]

• Maximize the minimum clearance [Dúnlaing and Yap ’85]
• Shortest or fewest links?
• Shortest or fewest links?
• Seems dangerous…
• Maximize the minimum clearance?
• Maximize the minimum clearance?

• Seems exhausting…
• Can we optimize for clearance and length?
• Can we optimize for clearance and length?

• Nothing known theoretically
• Some notation…
Some notation…

\( \mathcal{O} \): set of polygonal obstacles in \( \mathbb{R}^2 \) with \( n \) vertices total
• Some notation…

\( \mathcal{O} \): set of polygonal obstacles in \( \mathbb{R}^2 \) with \( n \) vertices total

\( \gamma \): a path through \( \mathbb{R}^2 \setminus \mathcal{O} \)
• Some notation…

\(\mathcal{O}\): set of polygonal obstacles in \(\mathbb{R}^2\) with \(n\) vertices total

\(\gamma\): a path through \(\mathbb{R}^2 \setminus \mathcal{O}\)

\(\gamma[p,q]\): subpath of \(\gamma\) from \(p\) to \(q\)
• Some notation…

\( \mathcal{O} \): set of polygonal obstacles in \( \mathbb{R}^2 \) with \( n \) vertices total

\( \gamma \): a path through \( \mathbb{R}^2 \setminus \mathcal{O} \)

\( \gamma[p,q] \): subpath of \( \gamma \) from \( p \) to \( q \)

\( ||p,q|| \): the Euclidean distance between \( p,q \in \mathbb{R}^2 \)
Some notation…

\( \mathcal{O} \): set of polygonal obstacles in \( \mathbb{R}^2 \) with \( n \) vertices total

\( \gamma \): a path through \( \mathbb{R}^2 \setminus \mathcal{O} \)

\( \gamma[p, q] \): subpath of \( \gamma \) from \( p \) to \( q \)

\( \|p, q\| \): the Euclidean distance between \( p, q \in \mathbb{R}^2 \)

\( \text{cl}(p) \): the clearance of a point; \( \text{cl}(p) = \min_{o \in \mathcal{O}} \|p, o\| \)
Our Goal

• Minimize the cost $\mu(\gamma)$ of an $s,t$-path $\gamma$ where

$$
\mu(\gamma) = \int_{\gamma} \frac{1}{\text{cl}(\gamma(\tau))} d\tau
$$

[Wein, van den Berg, Halperin ’08]
\( \mu(p, q) \): cost of cheapest path from \( p \) to \( q \)
\( \mu(p,q) \): cost of cheapest path from \( p \) to \( q \)

- Want an \textit{approximately} cheapest \( s,t \)-path \( y \) such that \( \mu(y) \leq (1+\varepsilon) \mu(s,t) \) for any \( \varepsilon > 0 \) (a \( (1+\varepsilon) \)-approximation or \textit{approximation scheme})
\( \mu(p,q) \): cost of cheapest path from \( p \) to \( q \)

- Want an *approximately* cheapest \( s,t \)-path \( \gamma \) such that \( \mu(\gamma) \leq (1+\varepsilon) \mu(s,t) \) for any \( \varepsilon > 0 \) (a \( (1+\varepsilon) \)-approximation or *approximation scheme*)

- Most interested in *combinatorial complexity*
• Wein et al. ['08] gave a $\varepsilon$-additive approximation algorithm with running time polynomial in $n$, $1/\varepsilon$, and $\Lambda$. 
Wein et al. ['08] gave a $\varepsilon$-additive approximation algorithm with running time polynomial in $n, 1/\varepsilon,$ and $\Lambda$

$\Lambda$: essentially the total cost of edges in $\mathcal{O}$’s Voronoi diagram (not a polynomial in input size)
Wein et al. [’08] gave a \(\varepsilon\)-additive approximation algorithm with running time polynomial in \(n, 1/\varepsilon\), and \(\Lambda\).

\(\Lambda\): essentially the total cost of edges in \(\mathcal{O}\)’s Voronoi diagram (not a polynomial in input size)

Can we find a provably efficient multiplicative approximation?
Our Contribution

- First polynomial time approximation scheme
Our Contribution

• First polynomial time approximation scheme

• Returns a path of cost at most \((1+\varepsilon) \mu(s,t)\) in time

\[ O\left(\frac{n^2}{\varepsilon^2 \log \frac{n}{\varepsilon}}\right) \]
Short Sited

- Clearance defined entirely on the *closest* obstacle
Short Sited

• Clearance defined entirely on the closest obstacle

\( \mathcal{V} \): Voronoi diagram of \( \mathcal{O} \); subdivides \( \mathbb{R}^2 \setminus \mathcal{O} \) into cells of points sharing a closest obstacle feature
Short Sited

- Clearance defined entirely on the closest obstacle

\[ \mathcal{V} : \text{Voronoi diagram of } \mathcal{O}; \text{ subdivides } \mathbb{R}^2 \setminus \mathcal{O} \text{ into cells of points sharing a closest obstacle feature} \]

- \( O(n) \) complexity, computable in \( O(n \log n) \) time
• Wein et al. sample several points along edges of $\mathcal{V}$
- Wein et al. sample several points along edges of $\mathcal{V}$
- Some near-optimal path enters cells at sample points
• Wein et al. sample several points along edges of $\mathcal{V}$

• Some near-optimal path enters cells at sample points

• Build a graph whose edges are cheapest paths between sample points sharing a cell
Main Difficulty

• Need to reduce number of sample points and understand where they should go
Main Difficulty

• Need to reduce number of sample points and understand where they should go

• Start by simplifying Voronoi cells
$	ilde{\mathcal{V}}$: Voronoi diagram *refined* by adding several new edges *perpendicular* to obstacle features
\( \tilde{\mathcal{V}} \): Voronoi diagram \textit{refined} by adding several new edges \textit{perpendicular} to obstacle features
\(\tilde{\mathcal{V}}\): Voronoi diagram *refined* by adding several new edges *perpendicular* to obstacle features

- \(\tilde{\mathcal{V}}\) still has complexity \(O(n)\)
\( \tilde{\mathcal{V}} \): Voronoi diagram refined by adding several new edges perpendicular to obstacle features

- \( \tilde{\mathcal{V}} \) still has complexity \( O(n) \)

- And individual cells are less complex
• Each cell $T \in \mathcal{Y}$ is incident to one obstacle feature and has three additional edges:
Each cell $T \in \tilde{\mathcal{V}}$ is incident to one obstacle feature and has three additional edges:

- One *external edge* $\kappa_T$ which is a monotone clearance parabolic arc
• Each cell $T \in \mathcal{Y}$ is incident to one obstacle feature and has three additional edges:
  
  ◦ One *external edge* $\kappa_T$ which is a monotone clearance parabolic arc
  
  ◦ Two *internal edges* $\alpha_T$ (shorter) and $\beta_T$ (longer) which are perpendicular to obstacle feature
• Now the cells have less complexity
• Now the cells have less complexity
• But we still need to pick our sample vertices
• Now the cells have less complexity

• But we still need to pick our sample vertices

• Conceptually, enough to understand where cheapest path crosses $\beta_T$, and how it passes through cell interior
• We take advantage of two main ideas:
• We take advantage of two main ideas:

  ○ Progressively better approximations of $\mu(s, t)$ to help narrow down where cheapest paths can enter cells
• We take advantage of two main ideas:

  ◦ Progressively better approximations of $\mu(s,t)$ to help narrow down where cheapest paths can enter cells

  ◦ Our new technical tools, *well behaved paths* and *anchor points* to understand paths through cell interior
Well-behaved Paths

• A $p,q$-path $\gamma$ through cell $T$ is well-behaved if
Well-behaved Paths

- A $p,q$-path $\gamma$ through cell $T$ is well-behaved if
  1. $\lambda = \gamma \cap \text{int}(T)$ is a connected subpath
Well-behaved Paths

• A \( p, q \)-path \( \gamma \) through cell \( T \) is well-behaved if
  
  i. \( \lambda = \gamma \cap \text{int}(T) \) is a connected subpath
  
  ii. if it exists, then \( \lambda \) has constant clearance
**Lemma:** Let $p$ and $q$ be two points on the edges of cell $T$. There exists a well-behaved path $p,q$-path $\gamma$ where $\mu(\gamma) \leq 7\mu(p,q)$.
• Proof picks lambda based on maximum clearance of cheapest $p, q$-path $\gamma^*$
• Proof picks lambda based on maximum clearance of cheapest $p,q$-path $\gamma^*$

• Meaning we need to know the clearance along $\gamma^*$...
**Anchor Points**

**Lemma**: Let $T$ be a cell of $\tilde{V}$. There exist constant-time computable anchor points $w_{\alpha}^*$ and $w_{\kappa}^*$ on long internal edge $\beta_T$ such that...
For any $p \in \beta_T$ and $q \notin \beta_T$, there exists a well-behaved path $p,q$-path $\gamma$ where $\mu(\gamma) \leq 7\mu(p,q)$ and $\lambda \cap \beta_T \in \{w_\alpha^*, w_\kappa^*, p\}$.
Algorithms

- Compute progressively better approximations
Algorithms

• Compute progressively better approximations
  1. $O(n \log n)$ time $O(n)$-approximation $=>$
Algorithms

• Compute progressively better approximations
  1. $O(n \log n)$ time $O(n)$-approximation =>
  2. $O(n^2 \log n)$ time $O(1)$-approximation =>
Algorithms

• Compute progressively better approximations
  1. $O(n \log n)$ time $O(n)$-approximation =>
  2. $O(n^2 \log n)$ time $O(1)$-approximation =>
  3. $O(n^2 / \epsilon^2 \log (n / \epsilon))$ time $(1+\epsilon)$-approximation
Will focus on the $O(n)$-approximation for this talk
• Will focus on the $O(n)$-approximation for this talk
• Let $y^*$ be the cheapest $s,t$-path
• Will focus on the $O(n)$-approximation for this talk
• Let $y^*$ be the cheapest $s,t$-path
• Let $d^* = \mu(y^*) = \mu(s,t)$
O(n)-approximation

- Build a geometric graph $G_1$ by adding $O(n)$ edges to $\tilde{\mathcal{V}}$.
O(n)-approximation

• Build a geometric graph $G_1$ by adding $O(n)$ edges to $\tilde{V}$

• Edges have a cost equal to the cost of their paths through the plane
O(n)-approximation

- Build a geometric graph \( G_1 \) by adding \( O(n) \) edges to \( \tilde{V} \)

- Edges have a cost equal to the cost of their paths through the plane

- Compute cheapest path between \( s \) and \( t \) in \( G_1 \) in \( O(n \log n) \) time
• For each cell $T$ of $\tilde{\mathcal{V}}$: 

![Diagram with annotations]
• For each cell $T$ of $\tilde{\mathcal{V}}$:
  ◦ Add vertex at point $w_s \in \beta_T$ where $\text{cl}(w_s) = \text{cl}(s)$
• For each cell $T$ of $\tilde{\mathcal{V}}$:
  - Add vertex at point $w_s \in \beta_T$ where $\text{cl}(w_s) = \text{cl}(s)$
  - Add vertices for anchor points $w_{\alpha^*}$ and $w_{\kappa^*}$
• For each cell $T$ of $\tilde{\mathcal{Y}}$:
  ◦ Add vertex at point $w_s \in \beta_T$ where $\text{cl}(w_s) = \text{cl}(s)$
  ◦ Add vertices for anchor points $w_\alpha^*$ and $w_\kappa^*$
  ◦ Add constant clearance edges from $w_s$, $w_\alpha^*$, and $w_\kappa^*$
Lemma: $G_1$ contains an $s,t$-path of cost $O(n) \cdot d^*$
**Lemma**: $G_1$ contains an $s, t$-path of cost $O(n) \cdot d^*$

- Proof replaces subpaths of $y^*$ through each cell $T$ with well-behaved paths through $G_1$
**Lemma:** $G_1$ contains an $s, t$-path of cost $O(n) \cdot d^*$

Proof replaces subpaths of $y^*$ through each cell $T$ with well-behaved paths through $G_1$
**Lemma**: $G_1$ contains an $s, t$-path of cost $O(n) \cdot d^*$

- Proof replaces subpaths of $y^*$ through each cell $T$ with well-behaved paths through $G_1$
**Lemma:** $G_1$ contains an $s,t$-path of cost $O(n) \cdot d^*$

- Proof replaces subpaths of $y^*$ through each cell $T$ with well-behaved paths through $G_1$
**Lemma:** $G_1$ contains an $s,t$-path of cost $O(n) \cdot d^*$

- Proof replaces subpaths of $\gamma^*$ through each cell $T$ with well-behaved paths through $G_1$
Lemma: $G_1$ contains an $s, t$-path of cost $O(n) \cdot d^*$

- Proof replaces subpaths of $\gamma^*$ through each cell $T$ with well-behaved paths through $G_1$
- Each well-behaved path has cost $O(d^*)$
**Lemma**: $G_1$ contains an $s,t$-path of cost $O(n) \cdot d^*$

- Proof replaces subpaths of $y^*$ through each cell $T$ with well-behaved paths through $G_1$
- Each well-behaved path has cost $O(d^*)$
- Total cost of replacement paths is $O(n) \cdot O(d^*)$
O(1)-approximation

• Guess an estimate $d$ of $d^*$
O(1)-approximation

- Guess an estimate $d$ of $d^*$
- Build *planar* graph $G_2$ by adding $O(n^2)$ edges to $\tilde{\mathcal{V}}$
O(1)-approximation

- Guess an estimate $d$ of $d^*$
- Build *planar* graph $G_2$ by adding $O(n^2)$ edges to $\tilde{\mathcal{V}}$
- Compute cheapest path between $s$ and $t$ in $G_2$
O(1)-approximation

- Guess an estimate $d$ of $d^*$
- Build *planar* graph $G_2$ by adding $O(n^2)$ edges to $\tilde{\mathcal{V}}$
- Compute cheapest path between $s$ and $t$ in $G_2$
- Algorithm tries $O(\log n)$ different choice for $d$ based on result of $O(n)$-approximation
• For each cell $T$ of $\tilde{\mathcal{V}}$: 
• For each cell $T$ of $\tilde{\mathcal{V}}$:
  
  - Place $O(n)$ new vertices in a region $\hat{\beta}_T \subseteq \beta_T$ based on $d$
• For each cell $T$ of $\tilde{\mathcal{V}}$:
  
  ✷ Place $O(n)$ new vertices in a region $\hat{\beta}_T \subseteq \beta_T$ based on $d$
  
  ✷ The *cost* between consecutive vertices is $d / n$
• For each cell $T$ of $\tilde{\mathcal{V}}$:
  - Place $O(n)$ new vertices in a region $\hat{\beta}_T \subseteq \beta_T$ based on $d$
  - The cost between consecutive vertices is $d/n$
  - Add constant clearance edges from each new vertex
Approximation Scheme

• Compute \( \hat{d} \) such that \( d^* \leq \hat{d} \leq O(1) \cdot d^* \)
Approximation Scheme

• Compute $\hat{d}$ such that $d^* \leq \hat{d} \leq O(1) \cdot d^*$

• Build graph $G_3$ by adding $O(n^2 / \epsilon)$ vertices and $O(n^2 / \epsilon^2 \log(n / \epsilon))$ edges to $\tilde{V}$
Approximation Scheme

• Compute \( \hat{d} \) such that \( d^* \leq \hat{d} \leq O(1) \cdot d^* \)

• Build graph \( G_3 \) by adding \( O(n^2 / \epsilon) \) vertices and \( O(n^2 / \epsilon^2 \log(n / \epsilon)) \) edges to \( \tilde{\mathcal{V}} \)

• Compute cheapest path between \( s \) and \( t \) in \( G_3 \)
• Sample densely on all edges of each cell
• Sample densely on all edges of each cell

• New edges no longer have constant clearance
• Sample densely on all edges of each cell
• New edges no longer have constant clearance
• Use anchor points to guide a sparse selection of edges
• Sample densely on all edges of each cell
• New edges no longer have constant clearance
• Use anchor points to guide a sparse selection of edges
• See the proceedings for details!
Summary and Open Problems

• Found $O(n^2 / \epsilon^2 \log (n / \epsilon))$ time $(1+\epsilon)$-approximation for high quality paths
Summary and Open Problems

• Found $O(n^2 / \epsilon^2 \log (n / \epsilon))$ time $(1+\epsilon)$-approximation for high quality paths

• Near-linear time?
Summary and Open Problems

• Found $O(n^2 / \epsilon^2 \log (n / \epsilon))$ time $(1+\epsilon)$-approximation for high quality paths

• Near-linear time?

• Problem complexity?
Summary and Open Problems

- Found $O(n^2 / \varepsilon^2 \log (n / \varepsilon))$ time $(1+\varepsilon)$-approximation for high quality paths
- Near-linear time?
- Problem complexity?
  - Algebraic complexity
Summary and Open Problems

• Found $O(n^2 / \epsilon^2 \log (n / \epsilon))$ time $(1+\epsilon)$-approximation for high quality paths

• Near-linear time?

• Problem complexity?
  • Algebraic complexity
  • Combinatorial complexity (NP-hard?)
Summary and Open Problems

- Found $O(n^2 / \varepsilon^2 \log (n / \varepsilon))$ time $(1+\varepsilon)$-approximation for high quality paths

- Near-linear time?

- Problem complexity?
  - Algebraic complexity
  - Combinatorial complexity (NP-hard?)

- Higher dimensions?
Thank you!
Thank you!