Soft Subdivision Motion Planning for Complex Planar Robots

Bo Zhou*  Yi-Jen Chiang*  Chee Yap**

* CSE Department, Tandon, New York University, USA
** CS Department, Courant, New York University, USA

^ Conference version appeared in ESA ‘18, August 2018
Motion Planning

• A central problem in robotics
  – There is a fixed rigid robot: $R_0 \subseteq R^k \ (k = 2,3)$
  – Configuration: pos. & orientation of a point $p$ in $R_0$

INPUT : $(\alpha, \beta, \Omega)$
• Start and Goal configurations $\alpha, \beta$
• Polyhedral obstacle set $\Omega \subseteq R^k \ (k = 2,3)$

OUTPUT:
• A path from $\alpha$ to $\beta$ avoiding all obstacles in $\Omega$, if it exists.
• Else report “NO PATH”. 
State of the Art

(A) Exact Methods
   + Strong theoretical guarantees
   - High complexity
     e.g., roadmap is single exponential time [Canny 93]
       basic path planning is semi-algebraic (book of [Basu-Pollack-Roy])
   - Complex to implement & expensive to compute
     (rarely implemented and not practical)

(B) Subdivision Methods
   Fairly popular but "does not scale"
   Often degenerate into “grid method”
State of the Art (cont.)

(C) Sampling Methods
* Dominate the field in the last 2 decades.
* Probabilistic Road Map (PRM) [Kravraki 96]; many variants: EST, RRT, SRT, etc.

Major Issue: Halting Problem (“Narrow Passage” problem) --- Don’t know how to halt when there is no path (except for artificial cut-off)

• Some subdivision work (e.g., [Zhang et al 08]) can detect non-existence of paths, but cannot guarantee to always detect that (sol. is partial).
State of the Art (cont.)

Resolution-Exact Algorithms

• We initiated in [Wang-Chiang-Yap SoCG13], [Yap 13]
• Use subdivision and soft predicates --- Soft Subdivision Search (SSS)
• Avoid exact computation, easy to implement correctly, run fast, always halt, with theoretical guarantees (see paper for details).

• Further extended for 2-link planar robot with 4 degrees of freedom (4 DOFs) [Luo-Chiang-Yap 14], [Chee-Luo-Hsu16], 5-DOF 3D robots [Hsu-Chiang-Yap 18].

• In this paper, we work on 2D complex robot under this framework.
New Results: SSS for Complex Robots

- 2D rigid complex robots with arbitrary complexity \((m\text{-sided polygon, } m\geq 5)\).
- Use triangulation.
- Our previous [SoCG 13] method for triangle robot does not work since the triangles in a complex robot must share a common origin (rotation center).
Review: Resolution Exactness

- An resolution-exact planner takes an extra input parameter $\varepsilon > 0$. It always halts and outputs either a path or NO-PATH. The output satisfies:

  There is an accuracy constant $K > 1$, s.t.
  - If exists a path of clearance $K\varepsilon$, it must output a path;
  - If there is no path of clearance $\varepsilon/K$, it must output NO-PATH.

  Indeterminacy allowed (small price for avoiding exact computation)
Review: Search Framework

• Maintain a subdivision tree $T$ rooted at box $B_0$
• Each internal node is a box $B$, which is split into $2^i$ $(1 \leq i \leq d)$ congruent subboxes
• Each box $B$ is classified as
  free (each $t \in B$ is a free configuration),
  stuck (each $t \in B$ is in the exterior of the free space),
  mixed (otherwise).

• We maintain connected components on free boxes and use a Union-Find data structure for that
• Priority Queue $Q$ of mixed boxes to be expanded later
Star-Shaped Robots

A star-shaped region $R$: there exists a point $A \in R$ s.t. A can “see” every point in $R$. We call $A$ a center of $R$.

When a robot $R_0$ is star-shaped, we decompose $R_0$ into a set of triangles that share a common vertex at a center $A$.

We need a predicate that can easily classify boxes $B$ as free/stuck/mixed.
Star-Shaped Robots

- Triangular Set: $T = H_1 \cap H_2 \cap H_3$ (intersection of three half-spaces)
  - $T$ can be bounded (triangle) or unbounded (Figure (a) below)
- Apex: distinguished vertex (red)
- Truncated Triangular Set (TTS): $TTS = T \cap D = H_1 \cap H_2 \cap H_3 \cap D$
  - $T$ intersects with a disc $D$ centered at $A$ (Figure (b) below)
Star-Shaped Robots

- Angular Range: $\Theta = [\alpha, \beta]$
- Swept area $T_0[\Theta] = T_0[\alpha, \beta]$ for triangle $T_0 (A,B,C$ where $A$ is the apex)
- Nice swept area (Figure (c) below) and not nice (Figure (d) below)
- Nice triangle: $b \geq \pi / 2 = 90^\circ$
  (where $a,b,c$ are the angles of vertices $A,B,C$ resp.)
Star-Shaped Robots

- Angular Range: $\Theta = [\alpha, \beta]$
- Swept area $T_0[\Theta] = T_0[\alpha, \beta]$ for triangle $T_0 (A,B,C$ where $A$ is the apex)
- Nice swept area (Figure (c) below) and not nice (Figure (d) below)
- Nice triangle: $b \geq \pi / 2 = 90^\circ$
  (where $a,b,c$ are the angles of vertices $A,B,C$ resp.)
- Lemma 1: $T$ is nice $\iff$ Footprints $T[0, \alpha]$ and $T[-\alpha, 0]$ are TTS for all $\alpha$
  where $0 < \alpha < \pi - a$
Star-Shaped Robots

- Lemma 2: A star-shaped robot $R_0$ (an $n$-gon) can be decomposed into an essentially disjoint union of at most $2n$ nice triangles sharing apex $A$. 

![Diagram of star-shaped robots with decompositions into triangles]
Star-Shaped Robots

• Complex Predicates: \( R_0 = \bigcup_{j=1}^{m} T_j \)

\[
\tilde{C}(B) = \begin{cases} 
\text{FREE} & \text{if each } \tilde{C}_j(B) \text{ is FREE} \\
\text{STUCK} & \text{if some } \tilde{C}_j(B) \text{ is STUCK} \\
\text{MIXED} & \text{otherwise}
\end{cases}
\]

• T/R Splitting

• Do *translational* (T) splitting only and keep the rotational component full until the box width is \( \leq \varepsilon \) (\( \varepsilon \)-small) --- top: quad tree

• Do *rotational* (R) splitting if the box is \( \varepsilon \)-small --- bottom: binary tree
Feature Set of a Box

- Each box $B$: we use its feature set $\tilde{\phi}(B)$ to classify $B$ as free/stuck/mixed.
- Obstacles $\Omega$: polygonal set $\Omega \subseteq \mathbb{R}^2$
- Look at boundary of $\Omega$. Feature $f$: corner or edge of boundary of $\Omega$
- Feature set $\tilde{\phi}(B)$: contains all $f$ that are potentially in conflict with robot $R_0$ when its configuration is in $B$.
- As we split a box into subboxes, the feature sets become smaller.
- **Classification:**
  - $\tilde{\phi}(B)$ is non-empty: $B$ is **mixed**.
  - $\tilde{\phi}(B)$ is empty: $B$ is in no conflict with obstacle boundary $\Rightarrow$ $B$ is **free** or **stuck** (use parent feature set to decide)
Feature Sets for Star-Shaped Robots (I)

Soft Predicates for classification

- When B is a T-split box (we only split its translational box; quad-tree part) its features set \( \tilde{\phi}(B) \) comprises those features \( f \) such that

\[
\text{Sep}(m_B, f) \leq r_B + r_0
\]

where \( m_B \) and \( r_B \) are the midpoint and radius of translational box of B, \( r_0 \) is the radius of robot \( R_0 \).
Feature Sets for Star-Shaped Robots (II)

When B is an **R-split box** (we only split its **rotational box**; binary-tree part)

- $\phi(B)$: a collection of $\phi_j(B)$ for each nice triangle $T_j$
- $TTS_j$: *apex is at the box center $m_B$*
- Let $Q$ be a shape and $s$ be a real number, **s-expansion** of $Q$ is defined as the Minkowski sum of $Q$ with the $Disc(s)$ of radius $s$ centered at the origin

- $\phi_j(B)$ comprises those features $f$ satisfying
  1. $\text{Sep}(m_B, f) \leq r_B + r_j$
  2. $f$ also intersects the $r_B$-expansion of $TTS_j$ (yellow: super set)

Condition (2) can be easily checked
General Complex Robots

• $R_0$ is a general polygon, we can still decompose $R_0$ into a set of triangles $T_j \ (j = 1, \ldots, m)$

• The rotation of these triangles are relative to a fixed point $O$

• We will define $T_j$ to be “nice relative to a point $O$”
General Complex Robots

- Let $T = [A, B, C]$, $O$ be the origin (outside of $T$)
- Let $0 \leq ||A|| \leq ||B|| \leq ||C||$ where $||A||$ is the Euclidean norm of a vector $A$

- We say $T$ is **nice** if
  - $\langle A, B - A \rangle \geq 0$
  - and $\langle A, C - A \rangle \geq 0$
  - and $\langle B, C - B \rangle \geq 0$. 
General Complex Robots

- If $T$ is a nice triangle, $T[\alpha, \beta]$ is called a nicely swept set (NSS).

We want an easy way to detect the intersection between an s-expansion of NSS and any feature (point or edge)
- We define a subset of $\mathbb{R}^2$ as a:
  - 0-basic shape: half-space, a disc or complement of a disc
  - 1-basic shape: finite intersection of 0-basic shapes
  - 2-basic shape: finite union of 1-basic shapes
General Complex Robots

- E.g. 1-basic shapes:
  - Triangles (ABC)
  - Sectors (A’C’C”)
  - Truncated strips (ACC”A’ --- shown in yellow)
- The s-expansion of a sector / truncated strip / triangle is 2-basic.

Theorem: Let $T[\alpha, \beta]$ be a nicely swept set where $[\alpha, \beta]$ has width $\leq \pi / 2$. It can be decomposed into a triangle, a sector and a truncated strip. The s-expansion of $T[\alpha, \beta]$ has a basic decomposition into 2-basic shapes.

Testing intersection of 2-basic shapes with any feature is $O(1)$. 
General Complex Robots

- Partitioning an \( n \)-gon into Nice Triangles
  - First triangulate into \( n-2 \) triangles
  - For the one contains the origin \( O \), split into 6 nice triangles using the star-shaped technique
  - **Lemma**: If \( T \) is an arbitrary triangle and \( O \) is exterior to \( T \), then we can partition \( T \) into at most 4 nice triangles.

**Theorem**: Given any triangulation of \( P \) into \( n \)-2 triangles, we can refine it into \( \leq 4n - 6 \) nice triangles.
General Complex Robots

• Soft Predicates: similar to the technique for star-shaped robots

• \( \tilde{\phi}_j(B) \) comprises those features \( f \) satisfying
  (1) \( \text{Sep}(m_B, f) \leq r_B + r_j \)
  (2) \( f \) also intersects the \( r_B \)-expansion of \( TTS_j NSS_j \)
Experimental Results

• Created challenging environments with several complex robots.
Summary of Experimental Results

Comparing with several sampling methods (PRM, RRT, EST, KPIECE) in open-source library OMPL.

- OMPL planners often have unsuccessful runs and have to time out even when there is a path.

- Our algorithms perform in real time, often much faster than OMPL planners, in addition to being able to report NO-PATH.
Video Demo

• Video is available at (link given in the paper)
  https://cs.nyu.edu/exact/gallery/complex/complex-robot-demo.mp4

• Code is available (link given in the paper): Core Library
  https://cs.nyu.edu/exact/core_pages/downloads.html
Conclusions

• We extended our SSS resolution-exact approach to challenging planning problems where no exact algorithms exist.

• Experiments show that our methods typically outperform OMPL sampling methods.

• Open Problems:
  (1) Optimal decomposition of m-gons into nice triangles?
  (2) Complex rigid robots in 3D?