Robot Path Planning Algorithm

Yonggang Li, Rencai Jin, Xiangrong Xu, et al.

Khang Luong, Ong Xuan Son

December 12, 2025

Outline

Problem Formulation

Goal: Plan a safe, smooth, and efficient trajectory for a mobile robot from start S to goal G in an environment containing *static* and *dynamic* obstacles.

Global Planning Problem

- Find a collision-free global path $\pi = \{p_0, p_1, \ldots, p_T\}$.
- Optimize for: path length, number of turning points, smoothness.

Local Planning Problem

- Robot state: $q = [x, y, \theta, \phi]^T$ with control input $u = [v, \omega]^T$.
- Predict feasible trajectories under kinematic + velocity constraints.
- Avoid static & dynamic obstacles in real time.

Outline

Improved A* Algorithm

The A* algorithm is a heuristic search method for global path planning.

Cost Function: The cost of a node n is evaluated by the function:

$$f(n) = g(n) + h(n)$$

$f(n)$: The total estimated cost of the path through node n.
$g(n)$: The **actual cost** from the start node to the current node n.
$h(n)$: The **heuristic estimated cost** from node n to the target node.

Improvement 1: Adaptive Step Size

To increase flexibility, the algorithm adapts its step size based on the surrounding environment.

Threat Function: The obstacle density is quantified by a threat function $f(x_1, x_2)$:

$$f(x_1, x_2) = \begin{cases} \frac{1}{k_1 x_1 + k_2 x_2 + c} & \text{if } d = 0 \\ 1 & \text{if } d \neq 0 \end{cases}$$

- x_1, x_2: The number of static obstacles in immediate and nearby areas, respectively.
- d: The number of dynamic obstacles in the direction of motion.
- k_1, k_2, c: Weighting coefficients.

Adaptive Step Size Formula: The step length l is then calculated as:

$$l = \begin{cases} f(x_1, x_2) \cdot l_{\max} & \text{if } d = 0 \\ f(x_1, x_2) \cdot l_{\min} & \text{if } d \neq 0 \end{cases}$$

This allows the robot to take larger steps in open spaces and smaller, safer steps near obstacles.

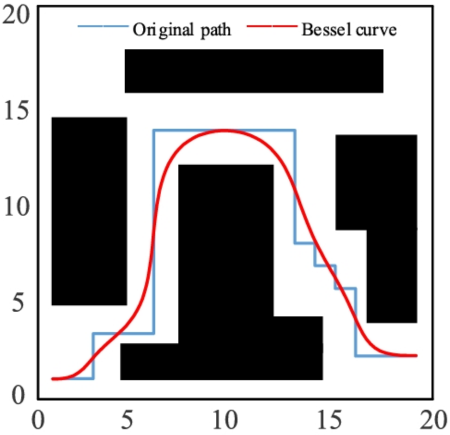
Improvement 2: Path Smoothing with Bezier Curves

Cubic Bezier Curve Formula:

- A curve segment is defined by a start point P_0, an end point P_3, and two control points P_1, P_2.
- The coordinates $B(t)$ on the curve at time $t \in [0, 1]$ are given by:

$$B(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$$

This smoothing process ensures the robot can travel smoothly, reducing motor strain and satisfying motion constraints.



Dynamic Window Approach (DWA)

1. **Velocity Sampling:** Sample multiple pairs of linear (v) and angular (ω) velocities within a "dynamic window" constrained by motor performance, acceleration limits, and a safe braking distance.
2. **Trajectory Simulation:** Predict the robot's trajectory for each sampled velocity pair over a short time interval.
3. **Evaluation and Selection:** Use an evaluation function to score valid (non-colliding) trajectories and select the one with the highest score.

DWA Evaluation Function:

$$G(v,\omega) = \sigma[\alpha \cdot head(v,\omega) + \beta \cdot stob(v,\omega) + \delta \cdot dyob(v,\omega) + \gamma \cdot velo(v,\omega)]$$

$head(v,\omega)$: **Angular deviation** from the global path.
$stob(v,\omega)$: **Distance** to the nearest static obstacle.
$dyob(v,\omega)$: **Distance** to the nearest dynamic obstacle.
$velo(v,\omega)$: The robot's forward **velocity**.

Hybrid Algorithm Performance

Quantitative Comparison:

Algorithm	Turning	Smoothness	Dynamic Avoid	Path Length
Traditional A*	8	No	No	14.07
Improved A*	6	Yes	No	11.92
DWA	-	Yes	Yes	Not reached
Hybrid Algorithm	**4**	**Yes**	**Yes**	**13.56**

Table: Performance comparison of the different algorithms.

Outline

SOO for Continuous Action Selection

Setting

$$a = (v, \omega) \in \mathcal{A} = [v_{\min}, v_{\max}] \times [\omega_{\min}, \omega_{\max}], \qquad f(a) = G(v, \omega)$$

Goal

$$a^* = \arg\max_{a \in \mathcal{A}} f(a)$$

SOO Search Tree

- Each node (h, i) represents a cell $C_{h,i} \subset \mathcal{A}$ and a center point $x_{h,i}$.
- $\mathcal{T}_t$: tree after t function evaluations.
- $\mathcal{L}_t$: leaves of $\mathcal{T}_t$.

SOO for Continuous Action Selection

Initialization: $\mathcal{T}_1 = \{(0,0)\}$ (root node), $t = 1.$

while True **do**

1. Set $f_{\max} \leftarrow -\infty.$
2. **for** $h = 0$ to $\min(\mathrm{depth}(\mathcal{T}_t),\, h_{\max}(t))$ **do**
   1. Among all leaves $(h,j) \in \mathcal{L}_t$ at depth h, select $(h,i) \in \arg\max_{(h,j)\in\mathcal{L}_t} f(x_{h,j}).$
   2. **if** $f(x_{h,i}) \geq f_{\max}$ **then**
      - Expand node (h,i): add to $\mathcal{T}_t$ the K children $\{(h{+}1, i_1), \ldots, (h{+}1, i_K)\}.$
      - Evaluate f at each new center $\{x_{h+1,i_1}, \ldots, x_{h+1,i_K}\}.$
      - Set $f_{\max} = f(x_{h,i}),$ $t = t + 1.$
      - **if** $t = n$ **then return** $x(n) = \arg\max_{(h,i)\in\mathcal{T}_n} f(x_{h,i}).$

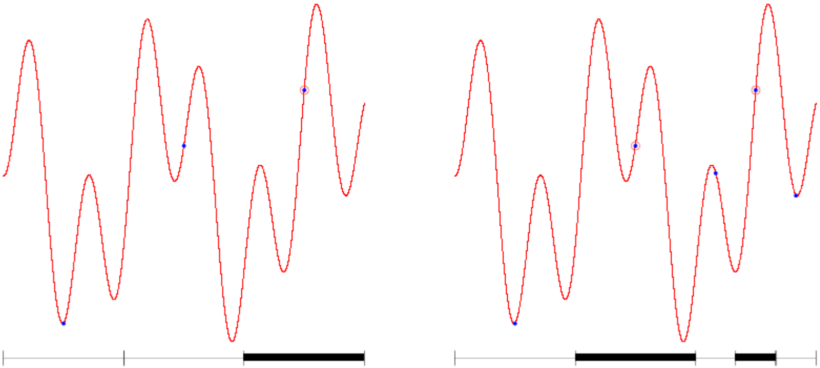
      end if

   end for

end while

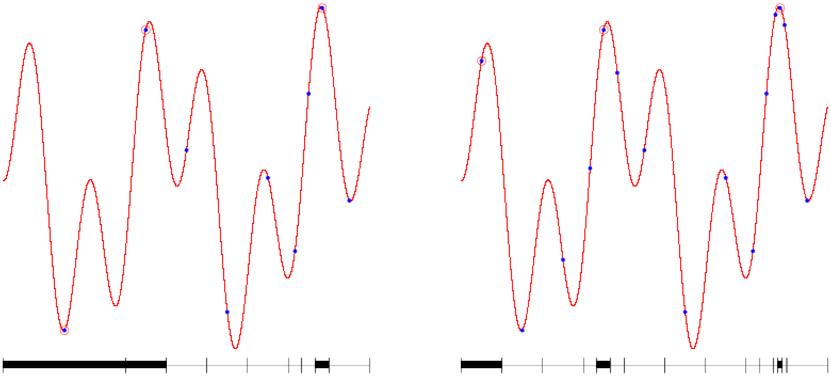
Example of SOO

$$f(x) = \frac{\sin(13x)\,\sin(27x) + 1}{2}$$



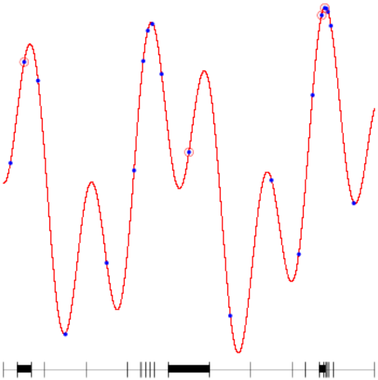
Example of SOO

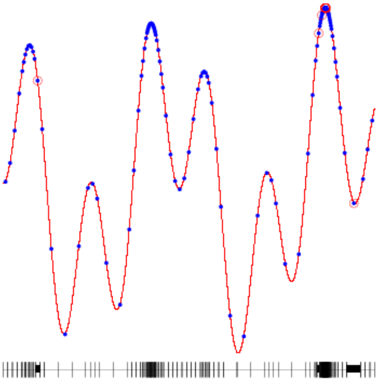
$$f(x) = \frac{\sin(13x)\,\sin(27x) + 1}{2}$$



Example of SOO

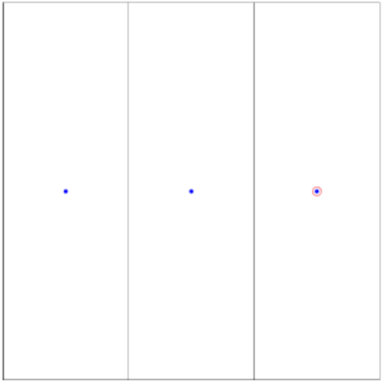
$$f(x) = \frac{\sin(13x)\ \sin(27x) + 1}{2}$$

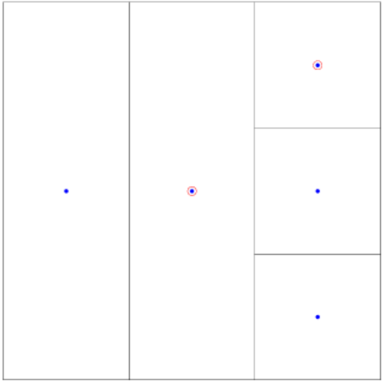




Example of SOO

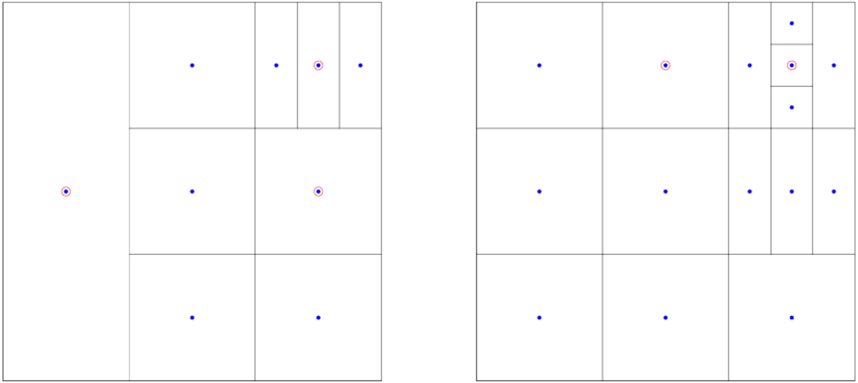
$$f(x_1, x_2) = f(x_1)f(x_2)$$





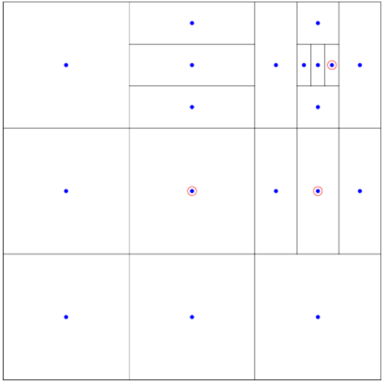
Example of SOO

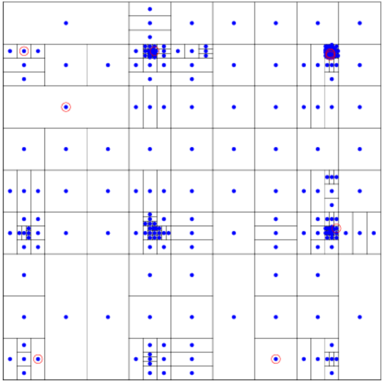
$$f(x_1, x_2) = f(x_1)f(x_2)$$



Example of SOO

$$f(x_1, x_2) = f(x_1)f(x_2)$$





Theoretical Guarantees of SOO

Setting

$$f : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}, \qquad x^* = \arg\max_{x \in \mathcal{X}} f(x)$$

Local Smoothness Assumption There exists a semi-metric ℓ and function ϕ such that

$$f(x^*) - f(x) \le \phi(\ell(x, x^*)),$$

with $\phi(\varepsilon) \to 0$ as $\varepsilon \to 0$. (No Lipschitz constant required.)

Near-Optimality Dimension d^*

$$d^* = \inf\{d \ge 0 : N(\varepsilon) \le C\varepsilon^{-d}\},$$

where $N(\varepsilon)$ is the number of ε-optimal regions.

Main SOO Guarantee (Munos, 2011)

$$f(x^*) - f(x(n)) \le \mathcal{O}\left(n^{-\frac{1}{d^*+2}}\right).$$

References I

- Y. Li, R. Jin, X. Xu, et al., "Improved A* and Hybrid DWA for Robot Path Planning".
- Rémi Munos, "From Bandits to Monte-Carlo Tree Search: The Optimistic Principle Applied to Optimization and Planning".