Cost-Aware Path Planning under Co-Safe Temporal Logic Specifications: Supplementary Material

Kyunghoon Cho, Junghun Suh, Claire J. Tomlin, and Songhwai Oh

In this supplementary material, we provide full proofs for probabilistic completeness and asymptotic optimality in Section I. In Section II, additional simulation results are provided on the convergence of the proposed method using more complex examples, the effects of different parameter configurations, and the use of a traveled-distance based cost function.

I. Analysis

In this section, we prove the probabilistic completeness and asymptotic optimality of the proposed algorithm (CARL). For simplicity, proofs are made under the following conditions.

(c1) Each weight of high-level state $w(q,r)$ is larger than $\tilde{w}$, where $0 < \tilde{w} \leq 1$. The maximum value of $w(q,r)$ is set as 1.
(c2) Sampling parameter in low-level layer is 0, i.e., $p_L = 0$.
(c3) *SelectTargetState* in Algorithm 1 uniformly samples a random discrete region.

The first condition (c1) assumes that each high-level state has a minimal probability to be selected. The second condition (c2) means that the vertex to extend is selected as the closest vertex to the sampled point; which makes *ExtendTree* (Algorithm 2) similar to traditional sampling based methods [1], [2]. In the proposed method, *SelectTargetState* samples a region of interest (with probability p_H) or a feasible random state (with probability $1 - p_H$). The last condition (c3) still reflects this property.

In addition, since the use of long extensions does not impair probabilistic completeness and asymptotic optimality [3], we do not mention it separately.

A. Notation

These are notations that we are using in this material.

ϕ	: syntactically co-safe LTL formula
$\mathcal{A}_\phi$	: deterministic finite automaton of ϕ, $\mathcal{A}_\phi = (Q, \Sigma, \delta, q_{init}, Q_{acc})$
$\mathcal{D}$	: graph returned by discrete abstraction, $\mathcal{D} = (R_d, E_d)$
$\mathcal{P}$	: high-level states $\mathcal{A}_\phi.Q \times \mathcal{D}$
N_d	: number of discrete regions $\lvert\mathcal{D}.R_d\rvert$
N_p	: number of high-level states
$\mathcal{X}$	: state space of robot ($\mathcal{X} \subset \mathbb{R}^d$)
$\mathcal{X}_{free}$	: feasible state space of robot ($\mathcal{X}_{free} \subset \mathcal{X}$)
$\mathcal{U}$	: control input space
x_{init}	: initial state
$\mathcal{T}$	: search tree generated by CARL
Υ_d	: mapping function, $\mathcal{X} \to \mathcal{D}.R_d$
$\mu(\mathcal{X})$	: the Lebesgue measure of set $\mathcal{X}$ (i.e. volume)
$\mathcal{B}_\epsilon(\mathrm{x})$	: the closed ball centered at x with the radius ϵ, i.e., $\mathcal{B}_\epsilon(\mathrm{x}) = \{\mathrm{x}' \in \mathcal{X} \mid \lVert\mathrm{x}' - \mathrm{x}\rVert \leq \epsilon\}$
$\mathcal{R}_\epsilon(z)$	: the set of states in $\mathcal{X}$ that are reachable from z without leaving $\mathcal{B}_\epsilon(z)$
f	: dynamic equation of robot (continuously differentiable)
c	: cost function (bounded and continuous), $\mathcal{X} \to \mathbb{R}^+$

B. Probabilistic Completeness

Let $\bar{R} = r_1, ..., r_M$ be a sequence of interest regions ($r_i \in \mathcal{D}.R_d$). A sequence of states $\mathbf{x} = \mathrm{x}_1, ..., \mathrm{x}_M$,where $\mathrm{x}_i \in r_i$, satisfies the temporal logic formula ϕ, which can be represented as $trace(\mathbf{x}) \vDash \phi$.

We define a sequence of subsets of $\mathcal{X}$ denoted as $\mathcal{A}_{\bar{R}} = \{A_0, A_1, A_2, ..., A_k\}$, which passes $\bar{R}$. $\bar{i} = i_1, ..., i_M$ is a sequence of indices of $\mathcal{A}_{\bar{R}}$ with $M \leq k+1$. Following properties are hold for $\mathcal{A}_{\bar{R}}$ and $\bar{i}$.

- $i_j < i_{j+1},\ \forall j = 1, ..., M-1$.
- $i_M = k$.
- $A_0 = \{\mathrm{x}_{init}\}$.
- $A_{i_j} \subseteq r_j, \forall j = 1, ..., M$.

Notice that each adjacent subsets in $\mathcal{A}_{\bar{R}}$ do not have to be overlapped, and the sequence $\mathcal{A}$ ends with the specific interest region $A_k \subseteq r_M$. This is natural since atomic propositions of an LTL formula correspond to regions of interest, and trajectory should pass a specific region of interest in order to progress toward an accepting automaton state.

Let $D(A_i)$ be a set of decomposed regions which cover A_i:

$$D(A_i) = \{\Upsilon_d(\mathrm{x}) \mid \forall \mathrm{x} \in A_i\}.$$

For a search tree $\mathcal{T}$, let $Q_{\mathcal{T}}(A_i)$ be high-level states of vertices in A_i, whose path from the root x_{init} follows the sequence of subsets $\{A_0, A_1, ..., A_i\}$.

Let $\tilde{p}$ be defined as

$$\tilde{p} = \min_{\substack{i \in \{0,1,...,k\}, \\ r \in D(A_i)}} \frac{\mu(A_i \cap {\Upsilon_d}^{-1}(r))}{\mu(\mathcal{X})},$$

This work was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2017R1A2B2006136) and by NSF CPS:Large:ActionWebs, award number 0931843.

K. Cho, J. Suh, and S. Oh are with the Department of Electrical and Computer Engineering and ASRI, Seoul National University, Seoul, Korea (e-mail: {kyunghoon.cho, junghun.suh, songhwai.oh}@cpslab.snu.ac.kr). C.J. Tomlin is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, USA (e-mail: tomlin@berkeley.edu).

which corresponds to a lower bound on the probability that a random state will lie in the intersection of particular A_i and one of decomposed region $D(A_i)$.

The following lemma explains the expected number of iterations in the proposed algorithm.

Lemma 1: If a sequence $\mathcal{A}_{\bar{R}}$ with length k exists, then the expected number of iterations required to pass specific regions of interest $\bar{R}$ (or satisfy ϕ) is no more than k/p, where

$$p = \frac{\tilde{w}}{N_d \cdot N_p^2} \cdot \tilde{p}.$$

Proof: If A_{i-1} contains a vertex v, then we can compute the probability that A_i will contain a vertex v' connected from v, which is denoted as $P(A_i)$. Let $\langle q_s, r_s \rangle$, σ_H, and r_t are the results of functions (*SelectInitialState*, *DiscretePlanner*, *SelectTargetRegion*) in Algorithm 1. $P(A_i)$ is now computed as follows:

$$P(A_i) = P\Big(\langle q_s, r_s \rangle \in Q_{\mathcal{T}}(A_{i-1})\Big) \cdot P\Big(Q_{\mathcal{T}}(A_i) \cap \sigma_H \neq \emptyset\Big) \cdot P\Big(r_t \in D(A_i)\Big) \cdot \frac{\mu(A_i \cap \Upsilon_d^{-1}(r_t))}{\mu(\Upsilon_d^{-1}(r_t))}.$$

For each term in $P(A_i)$, we can find the lower bound.

(i) Since the high-level state to be extended is selected by weight $w(q, r)$ and $w(q, r)$ satisfying the condition (c1), the following inequality holds:

$$P\Big(\langle q_s, r_s \rangle \in Q_{\mathcal{T}}(A_{i-1})\Big) = \frac{\sum_{\langle q', r' \rangle \in Q_{\mathcal{T}}(A_{i-1})} w(q', r')}{\sum_{\langle q, r \rangle \in \mathcal{P}} w(q, r)} \geq \frac{\tilde{w}}{N_p}.$$

(ii) A certain way for high-level plan σ_H to include one of $Q_{\mathcal{T}}(A_i)$ is to set it as the target state in the discrete planning procedure. By the condition (c2), we get the following inequality:

$$P\Big(Q_{\mathcal{T}}(A_i) \cap \sigma_H \neq \emptyset\Big) \geq \frac{1}{N_d}.$$

(iii) Since each edge cost in $\mathcal{D}$ is positive, σ_H has no duplicated elements $|\sigma_H| \leq N_p$:

$$P\Big(r_t \in D(A_i)\Big) = \frac{1}{|\sigma_H|} \geq \frac{1}{N_p}.$$

From the above inequalities, the lower bound of $P(A_i)$ can be found as below:

$$\begin{aligned} P(A_i) &\geq \frac{\tilde{w}}{N_d \cdot N_p^2} \cdot \frac{\mu(A_i \cap \Upsilon_d^{-1}(r_t))}{\mu(\Upsilon_d^{-1}(r_t))} \\ &\geq \frac{\tilde{w}}{N_d \cdot N_p^2} \cdot \frac{\mu(A_i \cap \Upsilon_d^{-1}(r_t))}{\mu(\mathcal{X})} \\ &\geq \frac{\tilde{w}}{N_d \cdot N_p^2} \cdot \tilde{p} \\ &\geq p. \end{aligned}$$

In the worst case, the iterations can be considered as Bernoulli trials with probability of success p. A feasible solution is obtained after k successful extension progresses from A_{i-1} to A_i. Let $C_1, C_2, .., C_n$ be i.i.d. random variables with the Bernoulli distribution with parameter p. The random variable $C = C_1 + C_2 + ... + C_n$ is defined as the number of successes after n iterations and C has the binomial distribution

$$\binom{n}{k} p^k (1-p)^{n-k},$$

where k is the number of successes. The expected number of iterations for finding path satisfying ϕ is k/p ($E[C] = np$). Since we consider the worst case, this value represents an upper bound. ■

The following establishes that the probability of failure decreases exponentially with the number of iterations.

Lemma 2: If a sequence $\mathcal{A}_{\bar{R}}$ with length k exists, the probability that search tree fails to find a path satisfying ϕ after n iterations is at most $\exp\big(-\frac{1}{2}(np - 2k)\big)$.

Proof: The random variable C in the proof of Lemma 1 has a binomial distribution. A Chernoff-type bound on tail probabilities can be applied to C. If $\delta \in (0, 1]$ and $\mu_C = E[C]$, then

$$P\Big(C \leq (1-\delta) \cdot \mu_C\Big) < \exp\Big(\frac{\mu_C \cdot \delta^2}{2}\Big).$$

By setting $\delta = 1 - k/(np)$ and $\mu_C = np$,

$$\begin{aligned} \exp\Big(\frac{\mu_C \cdot \delta^2}{2}\Big) &= \exp\Big(-\frac{np}{2} + k - \frac{k^2}{2np}\Big) \\ &= \exp\Big(-\frac{np - 2k}{2}\Big) \cdot \exp\Big(-\frac{k^2}{2np}\Big). \end{aligned}$$

Since $\exp(-\frac{k^2}{2np}) \leq 1$, we get the following result:

$$P(C \leq k) < \exp\Big(-\frac{1}{2}(np - 2k)\Big).$$

■

The state at time $t + \Delta t$ is determined as

$$\mathrm{x}(t + \Delta t) = \mathrm{x}(t) + \int_t^{t+\Delta t} f(\mathrm{x}(t), \mathrm{u}(t)) dt$$

for $\mathrm{u}(t)$. Let $\bar{u}$ be the correct input for state x. Since f is a smooth (continuously differentiable) function and

$$\dot{\mathrm{x}} = f(\mathrm{x}, \mathrm{u}) = \frac{d\mathrm{x}}{dt} \simeq \frac{\mathrm{x}(t + \Delta t) - \mathrm{x}(t)}{\Delta t},$$

for small amount of time Δt, the state after input $\bar{u}$ is

$$\bar{\mathrm{x}} = \mathrm{x}(t + \Delta t) = \mathrm{x} + \Delta t f(\mathrm{x}, \bar{u}) + H_1,$$

where H_1 is high order terms. If a perturbed input $\bar{u} + \delta$ is applied to the system for small δ, then the state x' at time $t + \Delta t$ is as follows with high-order terms H_2:

$$\mathrm{x}' = \mathrm{x} + \Delta t f(\mathrm{x}, \bar{u} + \delta) + H_2.$$

Since $|H_1 - H_2|$ is negligible for small Δt, we can find δ such that $\|\mathrm{x}' - \bar{\mathrm{x}}\| < \epsilon$. Define a set for δ as $D(\delta) = \{\delta' \in \mathcal{U} \mid \|\delta'\| \leq \|\delta\|\}$. If $\mathcal{U}$ is bounded, there exists a positive probability at least $\mu(D(\delta))/\mu(\mathcal{U}) > 0$ such that a randomly selected input can steer state x into $\mathcal{B}_\epsilon(\bar{\mathrm{x}})$.

We now prove the probabilistic completeness of proposed method.

Theorem 1: Algorithm 1 is probabilistically complete.

Proof: If there exists a trajectory satisfying ϕ, an appropriate $\mathcal{A}_{\bar{R}}$ with length k can be defined according to the trajectory. Let x_i be a vertex in A_i, for $i = 0, ..., k-1$. Assume that the tree $\mathcal{T}$ contains x_i as a vertex after some finite number of iterations. There exists a positive probability p_1 for extending x_i toward x_{i+1}. Also, by positive probability p_2 depending on p_1, correct input u_i is selected (even for random input when $\mathcal{U}$ is bounded). Both x_i and u_i have a probability at least p_2 to be selected. Therefore, from Lemma 2, the probability of finding a solution trajectory tends to 1 as the number of iterations increases. ■

C. Asymptotic Optimality

We state two assumptions from [4], which are modified to our problem. First, we assume local controllability of the system defined as follows.

Assumption 1: There exist constants α, $\bar{\epsilon} \in \mathbb{R}^+$, $p \in \mathbb{N}$, such that for any $\epsilon \in (0, \bar{\epsilon})$, and any state $z \in \mathcal{X}$, the set $\mathcal{R}_\epsilon(z)$ contains a ball of radius $\alpha\epsilon^p$.

Second, we present the hypothesis about the environment, such as obstacles and regions of interest, to ensure that there exits an optimal trajectory satisfying the given temporal logic specification with enough free space around it.

Assumption 2: There exits an optimal feasible trajectory $\Xi^* : [0, T^*] \to \mathcal{X}_{free}$, constants $\bar{\epsilon}$, $\alpha \in \mathbb{R}^+$, $p \in \mathbb{N}$ and a continuous function $q : \mathbb{R}^+ \to \mathcal{X}$ with $\lim_{\epsilon \downarrow 0} q(\epsilon) = \Xi^*$ such that for all $\epsilon \in (0, \bar{\epsilon})$ following two properties hold for the path $\Xi_\epsilon = q(\epsilon)$.

- Ξ_ϵ has strong ϵ-clearance [2] and satisfies the LTL formula ϕ.
- For any $t_1 < t_2$, let $z_1 = \Xi_\epsilon(t_1)$ and $z_2 = \Xi_\epsilon(t_2)$, then $\mathcal{B}_{\alpha \|z_1 - z_2\|^p}(z_2) \subset \mathcal{R}_\epsilon(z_1)$ for some $p \geq 1$.

The function q ensures the existence of a class of paths near the optimal trajectory Ξ^*, starting from the initial state and satisfying the given LTL specification. The constants α and p in the second property of Assumption 2 are an assumption on the dynamics, so that such trajectories can be generated from the given dynamic f.

We define the last assumption on each decomposed region $\mathcal{D}.R_d$.

Assumption 3: For each decomposed region $r \in \mathcal{D}.R_d$ and a state $x \in \Upsilon_d^{-1}(r)$, there exists a constant $\kappa \in (0, 1]$ such that the following inequality holds:

$$\mu(\mathcal{B}_\epsilon(x) \cap \Upsilon_d^{-1}(r)) \geq \kappa \cdot \mu(\mathcal{B}_\epsilon(x)),$$

where $\epsilon \in \mathbb{R}^+$.

Let σ_H^* be the corresponding high-level plan of Ξ^*. Assuming that the optimal low-cost trajectory between any pair of points is unique, σ_H^* is unique since Ξ^* is the minimum cost solution. Consider two adjacent high-level states $\sigma_{H,1}^*, \sigma_{H,2}^* \in \sigma_H^*$ and corresponding two vertices v_1^*, v_2^* in the search tree $\mathcal{T}$. There exists a trajectory segment $\Xi_{1,2}^* \subset \Xi^*$, connecting v_1^* and v_2^* optimally. Let $\Xi_{1,2}^n$ be the trajectory from $\mathcal{T}$, which connects $\mathcal{B}_{r_n}(v_1^*)$ and $\mathcal{B}_{r_n}(v_2^*)$ with $r_n = \gamma\big(\frac{\log(n)}{n}\big)^{1/d}$, where n is the number of vertices in $\mathcal{T}$.

The following lemma shows that the proposed method can search $\Xi_{1,2}^*$ as iteration proceeds.

Lemma 3: For sufficiently large γ, the difference in cost between $\Xi_{1,2}^n$ and $\Xi_{1,2}^*$ goes to 0, as $n \to \infty$.

Proof: By conditions (c1) and (c3), there exists a positive probability that high-level states $\sigma_{H,1}^*$ and $\sigma_{H,2}^*$ include vertices of the tree $\mathcal{T}$, which also means that the existence of vertices near v_1^* and v_2^*, respectively. During the proof, vertices in high-level states $\sigma_{H,1}^*$ and $\sigma_{H,2}^*$ are considered, and we assume that the number of vertices and iteration number is equal for simplicity.

We can define a function $q_{1,2}(\epsilon)$ for $\Xi_{1,2}$ which satisfies properties in Assumption 2 beside satisfying the LTL formula ($\lim_{\epsilon \downarrow 0} q_{1,2}(\epsilon) = \Xi_{1,2}^*$). For each $n \in \mathbb{N}$ and the selected $\epsilon_n \in (0, \bar{\epsilon})$, a sequence of overlapping balls that cover $q_{1,2}(\epsilon_n)$ is constructed as

$$B_n = \{B_{n,1}, B_{n,2}, ..., B_{n,M_n}\}.$$

Each ball in B_n has a radius of $r_n = \gamma\big(\frac{\log(n)}{n}\big)^{1/d}$, and centers are separated by the distance $l^{1/p}$, where $l = \beta\epsilon_n$ for constant β. $\bar{\epsilon}$ and p are defined in Assumption 2. Since the balls are overlapped each other, the distance between centers are bounded $l^{1/p} \leq 2r_n$.

For two adjacent balls in B_n whose centers are x_1 and x_2, Assumption 1 guarantees that $\mathcal{R}_\epsilon(x_1)$ has a positive volume, and there exists a constant α such that $\mathcal{B}_{\alpha l}(x_2)$ is reachable form x_1 by Assumption 2 ($\|x_1 - x_2\|^p = l$). It means that there exists a suitable control that can connect two adjacent balls.

If the proposed method sample vertices inside all balls in B_n, it will return a path which has a cost close to that of $q_{1,2}(\epsilon_n)$ since the cost function c is continuous and bounded.

Let us assume enough large n so that $\beta' r_n \leq l$ satisfies with some constant β'. An event F_n denotes that no vertices are included in B_n at iteration n, and $P(F_n)$ for its probability. As $n \to \infty$, $P(F_n)$ can be upper-bounded by the product of the number of balls that cover the trajectory $q_{1,2}(\epsilon_n)$ and the probability that a ball in B_n does not contain a vertex. If L_i is the length of trajectory of $q_{1,2}(\epsilon_i)$, then $L_i/l^{1/p}$ is an approximate number of balls in B_n. From Lemma 1 and Assumption 3, the minimum probability that a ball $B_{n,i} \in B_n$ contains a vertex for the single iteration is proportional to $\mu(B_{n,i})/\mu(\mathcal{X})$, where $\mu(B_{n,i}) = \pi_d \gamma^d (\frac{\log n}{n})$ with the volume of unit sphere π_d in d dimensions. Now $P(F_n)$ satisfies the following inequality:

$$P(F_n) \leq \gamma_1 \cdot \frac{L_i}{l^{1/p}} \cdot \Big(1 - \frac{\gamma^d}{\gamma_2} \frac{\log n}{n}\Big)^n, \tag{1}$$

where γ_1 and γ_2 are constants ($\gamma_1, \gamma_2 \in \mathbb{R}^+$). From $\beta' \cdot r_n = \beta' \cdot \gamma\big(\frac{\log(n)}{n}\big)^{1/d} \leq l$, the inequality (1) can be written as

$$P(F_n) \leq \frac{\gamma_1}{\beta'^{1/p} \gamma^{1/p}} \cdot \frac{1}{(\frac{\log n}{n})^{d/p}} \cdot \Big(1 - \frac{\gamma^d}{\gamma_2} \frac{\log n}{n}\Big)^n. \tag{2}$$

From $x \leq \exp(x - 1)$,

$$\begin{aligned} \Big(1 - \frac{\gamma^d}{\gamma_2} \frac{\log n}{n}\Big)^n &\leq \Big(\exp\Big(-\frac{\gamma^d}{\gamma_2} \frac{\log n}{n}\Big)\Big)^n \\ &\leq \exp\Big(-\frac{\gamma^d}{\gamma_2} \log n\Big) \\ &\leq n^{-\gamma^d/\gamma_2}. \end{aligned}$$

Apply the above result to the inequality (2), we have

$$P(F_n) \leq \frac{\gamma_1}{{\beta'}^{1/p}\gamma^{1/p}} \cdot \frac{1}{(\frac{\log n}{n})^{d/p}} \cdot n^{-\gamma^d/\gamma_2}$$
$$\leq G \cdot n^{-\gamma^d/\gamma_2 + d/p},$$

where G is a constant. Since $\sum_{n=1}^{\infty} P(F_n) < \infty$ for sufficiently large γ, the event F_n can occur only finitely often due to the Borel-Cantelli lemma [5]. Therefore, as iteration proceeds, the generated trajectory $\Xi_{1,2}$ becomes close to $q_{1,2}(\epsilon_n)$, which means approaching to $\Xi^*_{1,2}$. ■

From Lemma 3, we state the asymptotic optimality of the proposed algorithm.

Theorem 2: Algorithm 1 is asymptotically optimal.

Proof: CARL returns a solution if there exists a vertex whose automaton state belongs to an accepting state in $\mathcal{A}_\phi.Q_{acc}$. Hence, the solution trajectory satisfies the LTL specification. The high-level planner selects the initial high-level states randomly and considers all possible next high-level states with positive probability. Since the number of high-level states is finite, all feasible transitions are selected infinitely often asymptotically. The trajectory segment from the search tree, which connects two vertices of two adjacent high-level states in σ^*_H, becomes the optimal low-cost trajectory as iteration proceeds. Therefore, as iteration goes, the cost between the solution trajectory Ξ and the optimal trajectory Ξ^* becomes 0, while the corresponding high-level plan is σ^*_H. ■

II. Experimental Results

In this section, we provide additional simulation results which are omitted in the paper due to the page limitation.

A. Convergence

In the paper, we only shows the convergence of the proposed method for an LTL formula for a sequential mission. We consider two additional missions:

$$\phi_1 = \Diamond(a) \wedge \Diamond(b) \wedge \Diamond(c) \wedge \Diamond(d)$$
$$\phi_2 = \Diamond(a \wedge ((w \vee a)\mathrm{U}(b \wedge ((w \vee b)\mathrm{U}(c))))),$$

where ϕ_1 is a coverage mission of four discrete regions and ϕ_2 is a strict sequential mission with w denoting the workspace beside regions of interest. ϕ_1 represents for "cover regions a, b, c and d", and ϕ_2 for "visit regions with the strict order: a, b and c".

The results are shown in Figure 1, where brighter regions are higher cost areas. The cost of solution trajectory converges to the optimal cost as the iteration number increases.

B. Strategic Planning for Surveillance

In the paper (section V-B), we compared proposed algorithm with other sampling-based path planning algorithms with LTL constraints [6], [7]. Additional simulation results are stated in below.

1) Computation time: In Figure 2, we show the cost and computation time for generating the first trajectory, i.e., a single run of each algorithm. The algorithm [7] is fast to find a solution and the proposed method requires a longer computation time to find the first solution trajectory due to its long-extension and rewiring steps. However, the cost of the trajectory found by the proposed method is lower than other approaches.

2) Effect of parameters: The proposed method is based on the following parameters: α, β, p_H, and p_L. α and β are parameters used to determine the weight of a high-level state, where α controls the effect of the coverage of vertices and β on the frequency of high-level state visitations. p_H and p_L are planning parameters for the high-level layer and the low-level layer, respectively. We have tested how each parameter affects the performance of the algorithm. Figure 3 shows simulation results. Default values of parameters are $\alpha = 1,\ \beta = 1,\ p_H = 0.6$, and $p_L = 0.2$. Ten independent runs are performed for each configuration. Notice that the parameter α has almost no effect on the average performance of the proposed method. In general, we find that there is no single parameter configuration which works the best and the performance of the proposed method is not on average dominated by the parameter configuration.

3) Different cost measure: To demonstrate the effectiveness of the proposed method, we have tested the case when the cost of a trajectory is the length of the trajectory. This cost function is generally used by most existing methods. Simulations are tested with planning parameters $p_H = 0.8, p_L = 0.2$, and results are shown in Figure 4. The proposed method and ML-RRT*, a variation of the proposed method without long extension, still outperform [6], [7]. This is due to the rewiring procedure which is required to find a trajectory with the minimum length. However, the performance difference between the proposed method and other algorithms are reduced. In Scenario 3, ML-RRT* shows the best performance since the effect of long extensions is not significant when the cost is the length of a trajectory.

References

[1] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, May 2001.

[2] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, Jun. 2011.

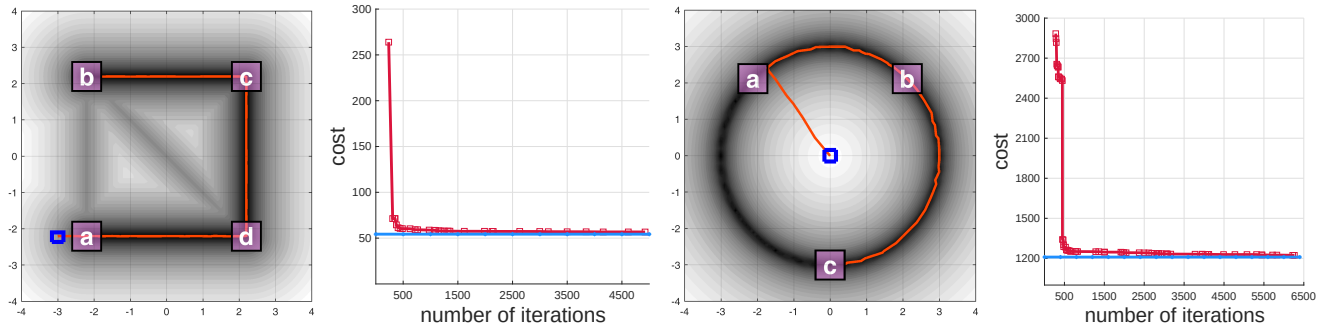
[3] J. Suh, "A robust localization and efficient path planning for mobile sensor networks," Ph.D. dissertation, Seoul National University, 2016.

[4] S. Karaman and E. Frazzoli, "Optimal kinodynamic motion planning using incremental sampling-based methods," in *Proc. of the IEEE Conference on Decision and Control*, Dec. 2010.

[5] G. Grimmett and D. Stirzaker, *Probability and random processes*. Oxford university press, 2001.

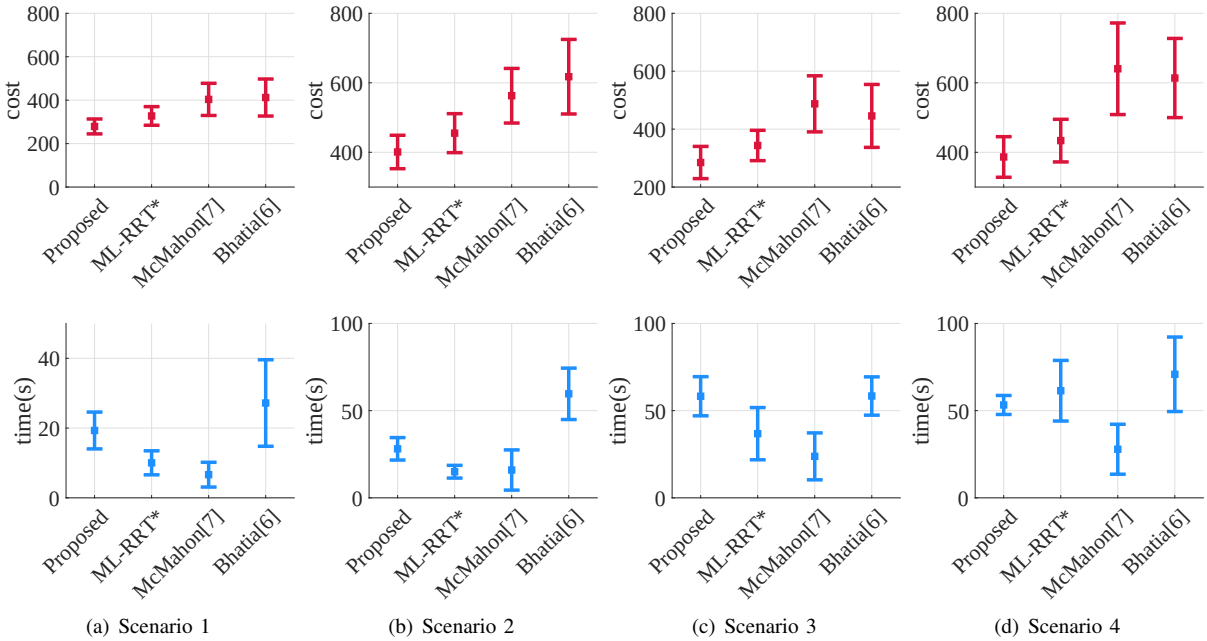
[6] A. Bhatia, L. E. Kavraki, and M. Y. Vardi, "Sampling-based motion planning with temporal goals," in *Proc. of the IEEE International Conference on Robotics and Automation*, May 2010.

[7] J. McMahon and E. Plaku, "Sampling-based tree search with discrete abstractions for motion planning with dynamics and temporal logic," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2014.



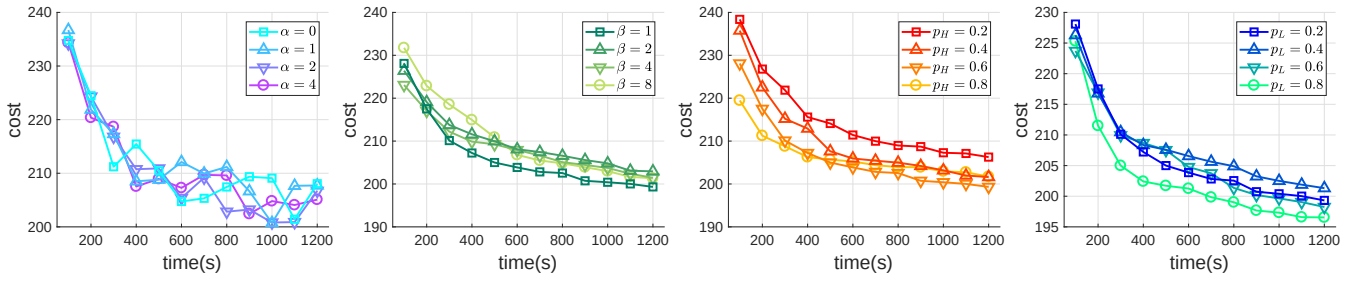
(a) (b) (c) (d)

Fig. 1. (a) A solution found by the proposed method for ϕ_1. (c) A solution found by the proposed method for ϕ_2. The initial states are marked by blue squares. (b,d) The cost of a trajectory found by the proposed method (in red) as a function of the iteration number. The optimal cost is shown in blue.

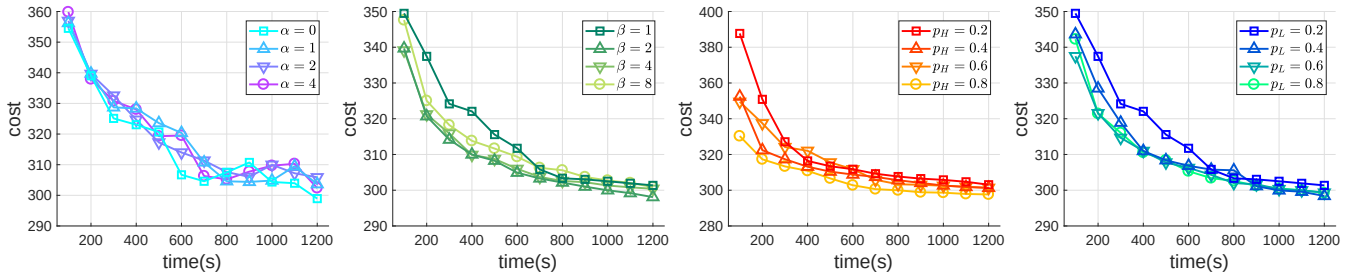


(a) Scenario 1 (b) Scenario 2 (c) Scenario 3 (d) Scenario 4

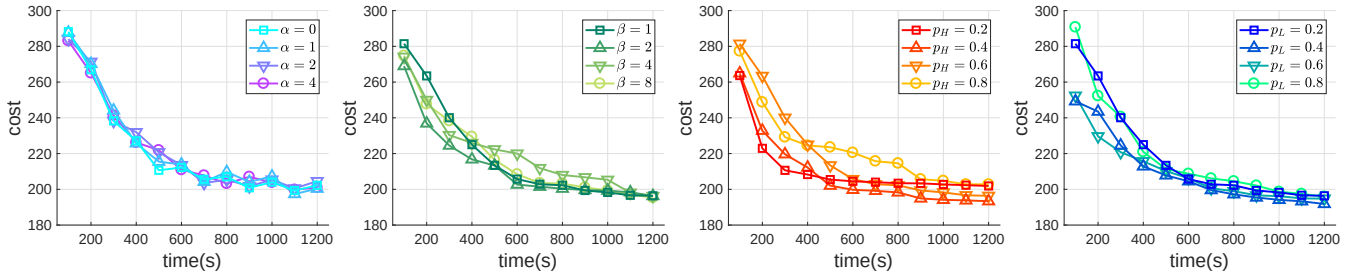
Fig. 2. Simulation results showing the cost and computation time (in seconds) of the first trajectory found by each algorithm. The average value of each algorithm is computed from 15 independent trials and one standard deviation is shown as an error bar.



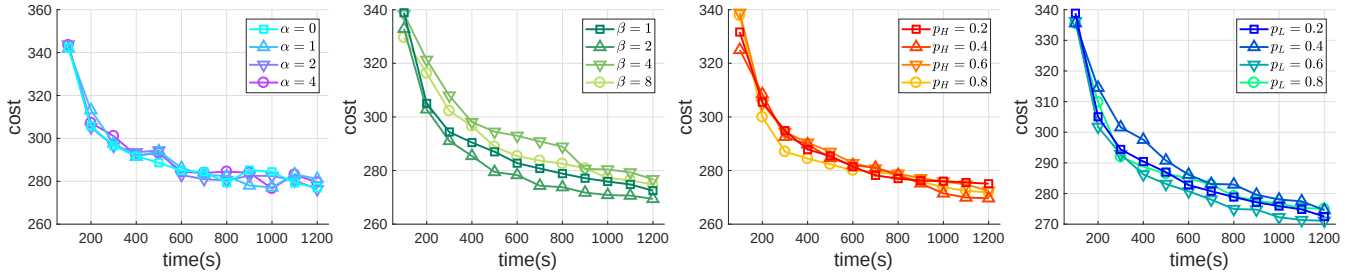
(a) Scenario 1



(b) Scenario 2

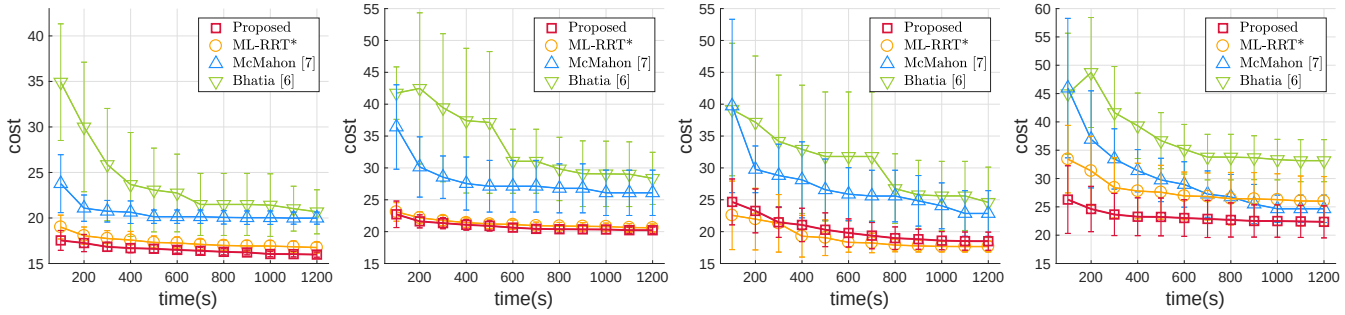


(c) Scenario 3



(d) Scenario 4

Fig. 3. Simulation results showing average trajectory costs as a function of running time (in second) with different parameter settings.



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3 (d) Scenario 4

Fig. 4. Simulation results showing average trajectory cost for four scenarios. The average cost of each algorithm is computed from 15 independent runs and one standard deviation is shown as an error bar.