A Cost-Aware Path Planning Algorithm for Mobile Robots

Junghun Suh and Songhwai Oh

Abstract— **In this paper, we propose a cost-aware path planning algorithm for mobile robots. As a robot moves from one location to another, the robot is penalized by the cost at its current location. The overall cost of the robot is determined by the trajectory of the robot over the cost map. The goal of the proposed cost-aware path planning algorithm is to find the trajectory with the minimal cost. The cost map of a field can represent environmental parameters, such as temperature, humidity, chemical concentration, wireless signal strength, and stealthiness. For example, if the cost map represents packet drop rates at different locations, the minimum cost path between two locations is the path with the best possible communication, which is desirable when a robot operates under the environment with weak wireless signals. The proposed cost-aware path planning algorithm extends the rapidly-exploring random tree (RRT) algorithm by applying the cross entropy (CE) method for extending motion segments. We show that the proposed algorithm finds a path which is close to the near-optimal cost path and gives an outstanding performance compared to RRT and CE-based path planning methods through extensive simulation.**

I. Introduction

Path planning has attracted much attention in the field of robotics due to its importance. The goal of a path planning algorithm is to find a continuous trajectory of a robot from an initial state to a goal state without colliding with obstacles while maintaining robot-specific constraints. A popular path planning algorithm is the rapidly-exploring random tree (RRT) [1] which is a sampling based method. It is applied and extended by many researchers under static environments [2], [3] and dynamic environments [4], [5].

In contrast to RRT, which solves a single query problem, the probabilistic roadmap (PRM) [6] is another sampling based path planning algorithm which is applied to solve multiple query problems. However, since the performance is determined by the number of samples, importance sampling based PRM approaches have been proposed in order to select more samples near the region of interest (see [7] and references therein). Recently, cross entropy (CE) [8], a combination of importance sampling and optimization, has been applied to path planning problems [9].

The aforementioned methods do not account for cost which may accumulate as a robot moves. For instance, if a robot has to travel through a dangerous environment and a map of the danger level is available, then we want to move the robot along the path with the minimum danger. To handle such case a number of improvements have been made and applied to more complex cost maps in recent years [10]–[15]. In order to increase the quality of a path, the nearest node of the tree to a random point is selected by computing the cost along the path when RRT extends a tree in [10], [11], [13]. Ettlin et al. [12] applied RRT to find the paths having low cost in rough terrain. They computed the cost of trajectories by biasing RRT towards low-cost areas. In [14], mobile robots are used to estimate environmental parameters of the field and coordinated towards the location with the most information. However, they did not consider the information gain along the trajectories of robots. In [15], Jaillet et al. presented the transition-based RRT (T-RRT) which finds low-cost paths with respect to a user given configuration space cost map based on a stochastic optimization method. However, since [15] extends the tree using a finite set of possible controls, the resulting paths can be suboptimal.

Motivated by these developments, we propose a cost-aware navigation strategy. We assume an availability of a cost map of the field of interest, which represents environmental parameters, such as temperature, humidity, chemical concentration, wireless signal strength, and stealthiness. Our objective is to design a path planning algorithm which guides a robot to follow a trajectory from the initial location to the destination with the minimal accumulated cost, along with the terminal cost and travel time. In this paper, the cost map of the field is represented using a Gaussian process to model complex physical phenomena. A Gaussian process (or Kriging in geostatistics) is a nonparametric regression or classification method which has been successfully applied to estimate and predict complex physical phenomena [16].

In this paper, we present a new path planning algorithm which extends the RRT algorithm by applying the cross entropy method when extending motion segments as a local planner extends the RRT tree. In contrast to the cross entropy based path planning algorithm, which can get stuck in local minima, the proposed algorithm takes advantage of RRT, which is probabilistically complete. On the other hand, the proposed method improves the quality of a trajectory by applying cross entropy. For nonholonomic dynamics, the RRT algorithm operates by extending the tree by applying a control from a finite set of possible controls. Hence, the resulting path from RRT is suboptimal in general. The application of cross entropy allows a search over a continuous space of possible controls and the quality of the resulting path is significantly better than RRT as demonstrated in this paper.

The contribution of this paper is as follows. We first show that environmental parameters can be well modeled by Gaussian processes. We then propose a cost-aware path planning algorithm for finding the minimum cost path of a robot from one location to another based on a cost map. We demonstrate the performance of the proposed method in simulation against RRT and cross entropy based path

This work has been supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2010-0013354).

Junghun Suh and Songhwai Oh are with the School of Electrical Engineering and Computer Science and ASRI, Seoul National University, Seoul 151-744, Korea (e-mail: {junghun.suh, songhwai.oh}@cpslab.snu.ac.kr).

planning algorithms and the near-optimal solution computed by the A^* algorithm.

The remainder of this paper is structured as follows. In Section II, we briefly review Gaussian process regression, rapidly-exploring random trees and the cross entropy method. Section III defines the cost-aware path planning problem and explains the proposed algorithm. The results from simulation are presented in Section IV.

II. Preliminaries

A. Gaussian Process Regression

A Gaussian process (GP) is a generalization of any finite number of random variable sets having a Gaussian distribution over a space of function. If $z(X)$ is a GP, where $X \in D \subseteq \mathbb{R}^n$, it can be fully described by its mean function $\mu(X) = \mathbb{E}(z(X))$ and covariance function $K(X, X') = \mathbb{E}[(z(X) - \mu(X))(z(X') - \mu(X'))]$, respectively, as follows

$$z(X) \sim \mathcal{GP}(\mu(X), K(X, X')). \tag{1}$$

We can represent the data $Y = [y_1, y_2, \ldots, y_n]^T \in \mathbb{R}^n$ measured at location $X = [X_1, X_2, \ldots, X_n] \in D$ as $y_i = f(X_i) + w_i$, where w_i is a white Gaussian noise with variance σ_w^2. For a new location X_*, based on the measured data Y, the predicted value of $z(X_*)$ has the Gaussian distribution with mean and variance as shown below:

$$\hat{z}(X_*) = K_*^T (K + \sigma_w^2 I)^{-1} Y \tag{2}$$

$$cov(\hat{z}(X_*)) = k(X_*, X_*) - K_*^T (K + \sigma_w^2 I)^{-1} K_* \tag{3}$$

where, $K = [K_{ij}]$ is the kernel matrix with (i, j) entries $K_{ij} = k(X_i, X_j)$, $K_* = [k(x_1, x_*), \ldots, k(x_n, x_*)]^T$, and $cov(\hat{z}(X_*))$ is the covariance of the estimated value $\hat{z}(X_*)$ [16].

In this paper, we use the squared exponential as a kernel function,

$$k(X, X') = \sigma_f^2 \exp\left(-\frac{1}{2\sigma_l^2} \sum_{m=1}^{n} (X_m - X'_m)\right) \tag{4}$$

where, σ_f and σ_l are hyperparameters of the kernel.

B. Rapidly-Exploring Random Tree

The rapidly-exploring random tree (RRT) [1] is a commonly used path planning algorithm, which explores a high dimensional configuration space using random sampling. Once a starting point and a goal point are defined, the algorithm iteratively samples a random point x_{rand} over the configuration space and makes an attempt to expand the search tree $\mathcal{T}$, which is initialized with the starting point. The x_{rand} is sampled from a uniform distribution over the space but we can make the distribution biased towards the goal point for faster search of a path to the goal. The tree $\mathcal{T}$ is extended by connecting from x_{near}, which is the nearest point of the tree from x_{rand}, to a new point x_{new} in the direction of x_{rand}, provided that x_{near} can be found. x_{new} is computed by applying a control input $u \in U$, where U is a set of possible controls, to the vehicle dynamics for a fixed duration $\triangle t$. Once the tree $\mathcal{T}$ reaches the goal point, the path can be built by searching the tree $\mathcal{T}$ recursively from the goal point to the starting point.

C. Cross Entropy Based Path Planning

Cross entropy (CE) is a sampling method designed to compute the probability of rare events [17]. It has been also extended to solve combinatorial optimization problems, such as the traveling salesman problem [8]. In [9], the cross entropy method is applied to path planning, where the optimal control law with minimum time is sought for. Algorithm 1 shows the cross entropy based path planning algorithm which finds a trajectory for a robot from the start position x_{start} to the end position x_{end}. We sample a trajectory X from distribution $p(\cdot; v)$, where v is a set of controls. The algorithm first generates N random trajectories $X_1, \ldots, X_N$ from $p(X; v_i)$ considering constraints such as obstacles and vehicle dynamics over the configuration space. If $\varrho > 0$ is a small number, the algorithm selects ϱN elite trajectory samples among all trajectory samples that have less cost based on the cost function H, which is defined as:

$$H(X) = \int_0^T C(u(t), x(t))dt$$

where $C = 1 + \beta \|x - x_{end}\|^2$ with a positive constant β and T is the termination time of the trajectory [9]. Then, it updates the parameter v_i (i.e., a set of pairs of control input and its duration) using the elite set. The algorithm iterates until the sampling distribution $p(X; v_i)$ converges to a delta distribution.

Algorithm 1 Cross Entropy Based Path Planning

Input: 1) Start position x_{start} and end position x_{end}
2) Number of trajectory samples N
3) Coefficients ϱ and β

Output: the shortest time path from x_{start} to x_{end}

1: $i = 0$.
2: Draw N samples $X_1, \ldots, X_N$ from $p(X, v_i)$, where $p(X, v_i)$ is a uniform distribution when $i = 0$.
3: Select ϱN trajectory samples among all trajectory samples that have less cost depending on cost function H.
4: Update the parameter v_i using the elite set.
5: $i = i + 1$.
6: Repeat steps 2–5 until $p(X; v_i)$ converges to a delta function.

III. Cost-Aware Path Planning

RRT and cross entropy based path planning algorithms find a path to the goal point considering only the length of the path, collision-free path, or both. However, the obtained trajectory may not be the optimal solution in a situation where environmental parameters such as temperature, humidity, chemical concentrations, wireless signal strength and stealthiness are represented as a cost map. Therefore, in this section, we propose a new path planning algorithm considering the cost of a trajectory based on the given cost map.

A. Problem Statement

Let $\mathcal{X} \subset \mathbb{R}^n$ be the state space of a robot, where $\mathcal{X} = \mathcal{X}_{free} \cup \mathcal{X}_{obs}$. $\mathcal{X}_{obs}$ represents the space where obstacles are placed or the robot may be in collision due to actuator bounds

and $\mathcal{X}_{free} = \mathcal{X} \setminus \mathcal{X}_{obs}$ is a free space. Let $\mathcal{U} \in \mathbb{R}^m$ denote the control input space. We assume that the state of the robot $x \in \mathcal{X}$ is determined by

$$\dot{x} = f(x, u). \tag{5}$$

Let $x_0 \in \mathcal{X}$ be the initial state of the robot and $x_{goal} \subset \mathcal{X}$ be the goal region. Let π_u be the solution to the differential equation (5) for given u from $t = 0$ to $t = T$. For input u, the cost of a trajectory based on the cost map, which is defined as $C : \mathcal{X} \to \mathbb{R}^+$, is defined as follows:

$$J(u) = \left(\frac{1}{T} \int_0^T C(\pi_u(t))dt + C_T(\pi_u(T)) \right), \tag{6}$$

where T is the terminating time of the trajectory and $C_T : \mathcal{X} \to \mathbb{R}^+$ is the terminal cost. The line integral of C along the path of the robot gives the quality of the robot trajectory. Now, the minimum cost path planning problem can be stated as:

$$\begin{aligned} &\min_{u(t):0\le t\le T} J(u) \\ &\qquad \text{subject to } \pi_u(t) \in \mathcal{X}_{free} \text{ for all } t \in [0, T]. \end{aligned} \tag{7}$$

The terminal cost function C_T can measure, for example, how close the final state of the robot is to the goal state. The travel time of the robot can be entered into the cost function by defining $C(x) = \alpha + C_{\text{costmap}}(x)$, where $\alpha > 0$ is a coefficient used to give a different weight to the travel time cost. There can be complications if we use (6) as a cost function since there can be a trajectory with a lower cost than trajectories which arrive at the goal state. To avoid this problem, we reformulate the minimum cost path planning problem as follows:

$$\begin{aligned} &\min_{u(t):0\le t\le T} \left(\frac{1}{T} \int_0^T C(\pi_u(t))dt \right) \\ &\qquad \text{subject to } \pi_u(t) \in \mathcal{X}_{free} \text{ for all } t \in [0, T] \\ &\qquad \text{and } C_T(\pi_u(T)) < \tau_T, \end{aligned} \tag{8}$$

where τ_T is a small number. Hence, we find a minimum cost trajectory from a set of trajectories which ends at a location very near the goal state, avoiding complications which might arise in (7).

B. Cost Map

Consider the field $\mathcal{X}$ on which robots perform their assigned tasks. We assume that a cost map is defined over $\mathcal{X}$, representing environmental parameters. Since we cannot measure the environmental parameter at every location, we use a nonparametric regression method, namely Gaussian process regression, to predict the environmental parameter at a site where no measurement is made.

We assume that the environmental parameter of interest, which defines the cost function C, follows a Gaussian process. Suppose we have made N samples from the field $\mathcal{X}$. Let $x_1, x_2, \ldots, x_N$ be the locations at which samples are taken and $y_1, y_2, \ldots, y_N$ be the measurements. Let $Y = [y_1, y_2, \ldots, y_N]^T$. Then, using (2), for any $x_* \in \mathcal{X}$, the expected value of the cost function at x_* is

$$C(x_*) = K_*^T (K + \sigma_w^2 I)^{-1} Y.$$

The resulting cost map is defined over the continuous space $\mathcal{X}$ and this is different from previous approaches where cost function is defined over a discretized space. If cost values are known at discrete sites, the same method can be applied to smooth the cost map. Hence, by applying Gaussian process regression, we obtain a cost map of infinite resolution. Note that the appropriate parameters for the kernel function and the variance of the observation model have to be learned from the data samples before the cost function is constructed.

C. Cost-Aware Path Planning Algorithm

We now describe a cost-aware path planning algorithm based on RRT which uses cross entropy for extending motion segments. The algorithm shares the overall structure as RRT as shown in Algorithm 2. However, x_{rand} is now sampled based on the cost map C using rejection sampling. Using rejection sampling, we can draw samples $\{x_{rand}\}$ such that we obtain more samples from low cost regions than high cost regions.

The tree $\mathcal{T}$ initially contains only one point x_0. After finding the nearest point x_{near} of the x_{rand} in the tree $\mathcal{T}$ according to the distance metric $\mathcal{D}$, which is explained in Section IV-C, we apply a modified version of the cross entropy based path planning algorithm (Algorithm 1) to extend the tree. In order to solve the modified path planning problem (8), we define η_a and η_b. In our algorithm, we first select η_a trajectories with the smallest $C_T(\pi_u(T))$ and then select η_b trajectories based on the discrete-time version of the running cost (8) when we select an elite set for cross entropy. With this two layers of filtering, we can find a solution to (8) which always arrives at the goal region. Using the modified cross entropy based path planning algorithm, we find a minimum cost path from x_{near} to x_{rand}. The maximum number of iterations in the cross entropy based path planning algorithm is fixed to N_{iter} in our algorithm. Only a truncated trajectory with time interval τ from the x_{near} is added to $\mathcal{T}$. The algorithm terminates when the tree $\mathcal{T}$ reaches the goal region x_{goal}.

The proposed algorithm can reach the goal region because it shares the overall structure as RRT, unlike the CE based method. On the other hand, while a solution from RRT is not finely tuned to minimize the cost, the proposed algorithm enjoys fine tuning of control over a continuous control input space using the cross entropy based optimization.

IV. Simulation results

Before describing simulation results, we first describe the state space, vehicle dynamics, and distance metric.

A. State Space

We denote the workspace (i.e., environment) as $\mathcal{X}$, where $\mathcal{X} \subset \mathbb{R}^2$. The workspace used in simulation is $[-45, 45]^2$. The workspace consists of two subspaces $\mathcal{X}_{free}$ and $\mathcal{X}_{obs}$ as mentioned in III-A. $\mathcal{X}_{obs}$ contains a number of obstacles, i.e., $\mathcal{O}_1, \cdots, \mathcal{O}_n \subset \mathcal{X}_{obs}$ and the region where the vehicle may be in collision with obstacles because of actuator bounds. We call the region the soft-obstacle zone which is similar with the inevitable collision state in [18] and the soft-obstacle zone has radius r from each obstacle. We define $\Phi : \mathcal{X} \to$

Algorithm 2 A cost aware navigation strategy

Input: 1) Start position x_0 and goal position x_{goal}
2) Number of trajectories N_t
3) Cost map $C : \mathcal{X} \to \mathbb{R}^+$
4) Number of elite samples η_a, η_b
5) Number of iteration N_{iter}
6) Step size τ

Output: Minimum cost path from x_0 to x_{goal}

1: Choose a sample point x_{rand} over state space based on rejection sampling.
2: Find a point in the tree $\mathcal{T}$ which is nearest to x_{rand} according to the distance metric $\mathcal{D}$.
3: Apply the modified version of Algorithm 1 in order to extend the tree $\mathcal{T}$ based on the cost map C. The maximum number of iterations in Algorithm 1 is set to N_{iter}.
4: Add the truncated trajectory for the first duration of τ.
5: Repeat steps $1-4$ until tree $\mathcal{T}$ reaches x_{goal}.

$\mathbb{R}^+$ such that $\Phi(x) = 0$ if $d > r$, $\Phi(x) = 1$ if $d = 0$, and $\Phi(x) = \frac{1}{1+exp(-\nu(d-\frac{r}{2}))}$ if $0 < d \leq r$, where d is the distance between x and the nearest obstacle and ν is a coefficient. In other words, $\Phi(x) = 0$ for $x \in \mathcal{X}_{free}$, $\Phi(x) = 1$ for $x \in \mathcal{X}_{obs}$, and the soft-obstacle zone has the value based on $\Phi(x)$.

B. Vehicle Dynamics

We used a dynamical model of the Dubins' car as a two-wheeled robot in this paper. Let (p_x, p_y) be the position of a robot and θ be its heading. Suppose $v(t)$ is the translational velocity and $u(t)$ is the angular velocity, then the whole dynamics is as follows:

$$\dot{p}_x(t) = v(t)\cos(\theta(t)), \quad \dot{p}_y(t) = v(t)\sin(\theta(t)), \quad \dot{\theta}(t) = u(t),$$

where we fixed v as a constant and used $2m/s$ in simulation. A discrete-time version of the above dynamical model is used in simulation where the sampling period t_s is set to 0.1.

C. Distance Metric

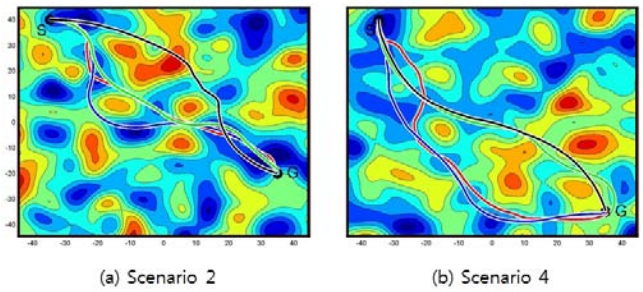
In order to measure the closeness between two states of a robot $x^1 = (p_x^1, p_y^1, \theta^1)^T$ and $x^2 = (p_x^2, p_y^2, \theta^2)^T$ we heuristically define a simple metric function $\mathcal{D} : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ as follows:

$$\mathcal{D}(x^1, x^2) = \omega_p \|(p_x^1, p_y^1)^T - (p_x^2, p_y^2)^T\|^2 + \omega_a |\theta^1 - \theta^2|,$$

where ω_p and ω_a are positive weights for position and orientation, respectively. When $\|(p_x^1, p_y^1)^T - (p_x^2, p_y^2)^T\|^2$ is smaller than threshold ϵ_b, we increased the weight ω_a and fixed ω_p at 1.

D. Results

In order to study the performance of the proposed algorithm, we compared algorithm with the basic RRT (B-RRT), the cost-aware RRT (CA-RRT), the cross entropy based path planning (CEPP) algorithm, where CA-RRT is a modified RRT which finds a trajectory with the minimal cost over the cost map by running the RRT algorithm multiple times and choosing a solution with the minimum cost. We also compared our algorithm against the near-optimal cost trajectories computed by the A^* algorithm on a grid by discretizing the workspace $\mathcal{X}$.



(a) Scenario 2 (b) Scenario 4

Fig. 1. Examples of scenarios used in simulation and the minimum cost trajectory found by each algorithm. The color over the field represents the cost of the field. The blue color represents low cost and the red color represents high cost. The black, red, green, and blue lines represent of trajectories found from B-RRT, CA-RRT, CEPP, and the proposed scheme, respectively.

We generated five scenarios with different cost maps from a Gaussian process (1) with kernel function (4), where $\sigma_f^2 = 1.0$ and $\sigma_l^2 = 5.0$. Some scenarios used in simulation are shown in Figure 1. We used the Gaussian process MATLAB toolbox [16] for modeling the cost map.

A fixed set of controls $\mathcal{U}$ are used for B-RRT and CA-RRT. A total of 99 controls were used from $(-\frac{\pi}{2} + 0.01)$ to $(\frac{\pi}{2} - 0.01)$ at an interval of 0.01. We used an adaptive number of N_t trajectory samples and N_{iter} iterations in order to reduce the computation time of extending the tree $\mathcal{T}$. We set N_t to 50 and N_{iter} to 20, but when the x_{goal} point is sampled as x_{rand}, we change N_t to 100 and N_{iter} to 50. We have found that the cross entropy based path planning tends to fail when x_{near} and x_{rand} are too close. So if two points are too close, we use the basic RRT extension method to extend the tree $\mathcal{T}$ in the proposed algorithm.

The terminal cost is defined as $C_T(x_T) = \beta \|x_T - x_{goal}\|^2$ where β is a positive constant and we do not include the heading of the x_T for computation of $C_T(x_T)$. The number of elite samples η_a and η_b in Section III-C are set to 25 and 10, respectively, for simulation. For each scenario, ten independent trials were performed and each trial was simulated for 600 seconds. This is the maximum deadline for all algorithms. We limited the number of times the tree $\mathcal{T}$ could be extended to 500 to reduce computation time.

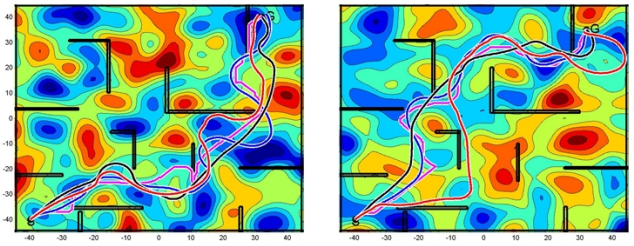
We first compared our algorithm against B-RRT, CA-RRT and CEPP for each scenario without obstacles. Figure 1 shows the cost map of each scenario and the minimum cost trajectory found by each algorithm. The cost map is shown as contours with different colors. The high cost region is represented by red and the low cost region is represented by blue. The dark circles represent the starting position and the goal location as written in Figure 1(a) and (b). The black, red, green, and blue trajectories represent results from B-RRT, CA-RRT, CEPP, and the proposed algorithm, respectively.

Since each algorithm requires a different run time, we assigned a deadline and repeated the algorithm under each deadline. A solution from each method is the lowest cost trajectory from multiple runs of the algorithm until the

Deadline	200 s	300 s	400 s	500 s	600 s
B-RRT	0.0779	0.0757	0.0749	0.0745	0.0739
CA-RRT	0.0404	0.0392	0.0387	0.0383	0.0378
CEPP	0.0477	0.0467	0.0456	0.0451	0.0451
Proposed	**0.0380**	**0.0369**	**0.0365**	**0.0356**	**0.0348**

TABLE I

THE AVERAGE COSTS OF SCENARIOS WITHOUT OBSTACLES AT DIFFERENT DEADLINES.



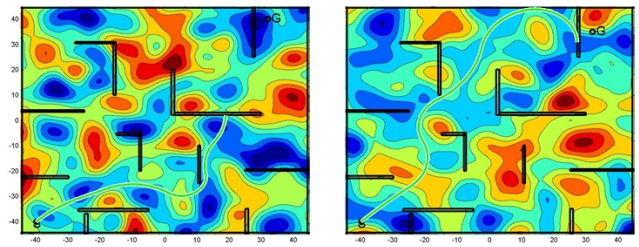
(a) Scenario 2 (b) Scenario 4

Fig. 3. Scenarios with obstacles. Only the trajectories of B-RRT (black), CA-RRT (red), the our scheme (blue) and the near-optimal trajectories found by the A^* algorithm (magenta) are shown since CEPP failed to arrive at the goal region in most cases.

deadline. We tried 5 different deadlines from 200 seconds to 600 seconds with an interval of 100 seconds. The average cost shown in Figure 2 is the average cost from 10 trials for each algorithm, along with one standard deviation error bar. The cost of B-RRT was too high, so it is not included in Figure 2. Results from CEPP are shown in green, results from CA-RRT are shown in red, and results from the proposed algorithm are shown in blue. The average cost and its variance tend to decrease as the deadline gets longer. But the mean and variance increased for CEPP between deadlines of 200 and 300. This is due to the fact that the running times are not fixed. Some runs of CEPP did not complete before 200 but terminated before 300 with higher costs. The proposed algorithm shows the lowest cost for all deadlines as shown in the figure. The average cost of each algorithm is given in Table I. The proposed algorithm shows the minimum average costs at all deadlines.

We also conducted simulations with obstacles based on the scenarios used in the previous simulations. The minimum cost trajectory for each scenario is shown in Figure 3. The black thick lines represents the soft-obstacle zone including the obstacles. In this paper, a random sample is not chosen in this zone and any trajectory can not traverse this zone. For scenarios with obstacles, CEPP failed in most cases as the algorithm gets stuck at local minima. Examples of failures from CEPP for the second and fourth scenarios are shown in Figure 4. The final state of the trajectories got stuck by an obstacle near the goal.

The average trajectory costs from CA-RRT and the proposed algorithm are shown in Figure 5. For all cases, either the proposed algorithm found a lower cost trajectory or a trajectory with cost which is comparable to that of a trajectory found by CA-RRT. The average cost of each algorithm is given in Table II. When compared to the near-



(a) Scenario 2 (b) Scenario 4

Fig. 4. Examples of failure from CEPP. The green trajectories did not reach the goal and got stuck by an obstacle.

Deadline	200 s	300 s	400 s	500 s	600 s
B-RRT	0.0951	0.0861	0.0853	0.0842	0.0826
CA-RRT	0.0600	0.0575	0.0553	0.0531	0.0509
Proposed	**0.0471**	**0.0458**	**0.0448**	**0.0441**	**0.0440**

TABLE II

THE AVERAGE COSTS OF SCENARIOS WITH OBSTACLES AT DIFFERENT DEADLINES.

Algorithm	B-RRT	CA-RRT	Proposed	A^*
Cost	**0.0580**	**0.0443**	**0.0391**	**0.0354**
Trajectory Length	661	753	703	622

TABLE III

THE AVERAGE COSTS AND TRAJECTORY LENGTHS FOR ALL SCENARIOS WITH OBSTACLES

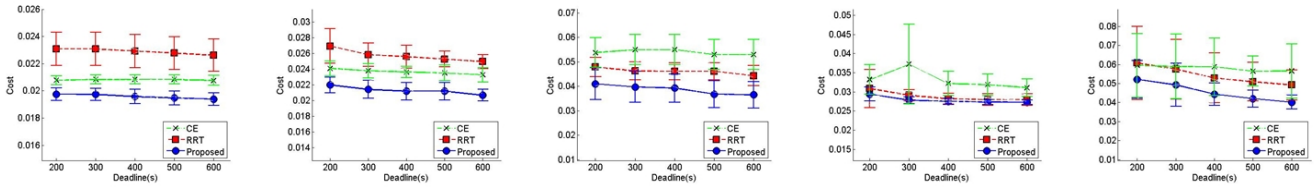
Algorithm	Scenario	1	2	3	4	5
CA-RRT	**No. Runs**	92	63	79	182	76
	No. Successes	57	26	47	154	37
CEPP	**No. Runs**	40	68	70	90	90
	No. Successes	0	1	1	1	1
Proposed	**No. Runs**	32	22	28	23	19
	No. Successes	31	18	26	23	19

TABLE IV

THE NUMBER OF RUNS AND SUCCESSFUL RUNS FROM EACH ALGORITHM BEFORE THE MAXIMUM DEADLINE (600S) FOR SCENARIOS WITH OBSTACLES.

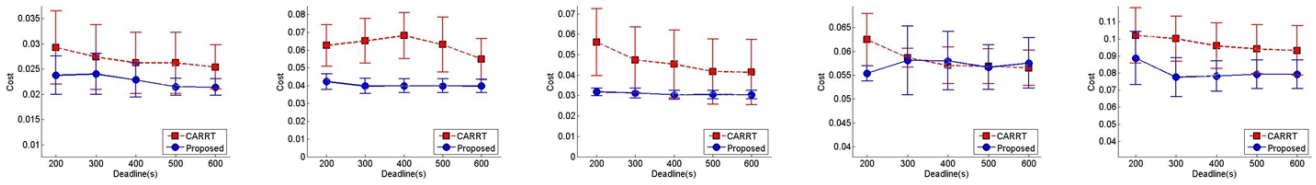
optimal cost paths by the A^* algorithm, the cost of ours is only 10% higher than the cost of the near-optimal cost paths as shown in Table III. Also, the trajectories from the proposed algorithm are very similar to the near-optimal paths found by the A^* algorithm (see Figure 3 for an example). Notice that the proposed algorithm achieves the preformance which is comparable to the near-optimal solution found by the A^* algorithm at a fraction of computation time compared to the A^* algorithm, which requires over 12 hours to compute a near-optimal cost path. Furthermore, the success rate of the proposed method is much higher than CA-RRT, showing the robustness of the proposed method against local minima. The number of runs and successful runs, i.e., when a trajectory arrives at the goal region, from each algorithm are shown in Table IV.

As can be seen from Figure 3, the trajectories found by



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3 (d) Scenario 4 (e) Scenario 5

Fig. 2. Average trajectory cost as a function of deadlines in the scenarios without obstacles. The average cost of each algorithm is computed from 10 independent trials and one standard deviation is shown as an error bar. Legend: CA-RRT (red), CEPP (green), and proposed algorithm (blue).



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3 (d) Scenario 4 (e) Scenario 5

Fig. 5. Average trajectory cost as a function of deadlines in the scenarios with obstacles. The average cost of each algorithm is computed from 10 independent trials and one standard deviation is shown as an error bar. Legend: CA-RRT (red) and proposed algorithm (blue).

the proposed algorithm have a shorter length while visiting lower cost regions more frequently. This is due to the fact the control at each instance is searched over a continuous space of control inputs, which is not possible for the general RRT methods.

V. Conclusions

In this paper, we have presented a cost-aware path planning algorithm. When a cost map over a configuration space is available, the proposed algorithm finds a suitable trajectory of a robot with the minimum cost. The proposed algorithm extends the well-known RRT algorithm by applying cross entropy (CE) for extending motion segments. In simulation, we compared the proposed algorithm against RRT and the cross entropy based path planning algorithm and showed its superior performance against the other two algorithm. While the cross entropy based path planning algorithm fails to find a path as it can fall into the local minima, the proposed algorithm shows robustness against local minima by taking advantage of the completeness of RRT. While a solution from RRT is not finely tuned to minimize the cost, the proposed algorithm enjoys fine tuning of control using the cross entropy based optimization. We expect the proposed algorithm can be applied to a number of path and motion planning problems.

References

[1] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 3, pp. 378–400, May 2001.

[2] N. A. Melchior and R. Simmons, "Particle RRT for path planning with uncertainty," in *Proc. of the IEEE International Conference on Robotics and Automation*, Roma, April 2007.

[3] J. V. der Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information," in *Proc. of the Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[4] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier, "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and Gaussian processes," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, Sep. 2008.

[5] Y. Kuwata, G. A. Fiore, J. Teo, E. Frazzoli, and J. P. How, "Motion planning for urban driving using RRT," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robotics and Systems*, Nice, France, Sep. 2008.

[6] L. E. Kavraki and J.-C. Latombe, "Randomized preprocessing of configuration for fast path planning," in *Proc. of the IEEE International Conference on Robotics and Automation*, San Diego, CA, June 1994.

[7] A. Rodríguez, A. Pérez, J. Rosell, and L. B. nez, "Sampling-based path planning for geometrically-constrained objects," in *Proc. of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009.

[8] D. P. Kroese, S. Porotsky, and R. Y. Rubinstein, "The cross-entropy method for continuous multi-extremal optimization," *Methodology and Computing in Applied Probability*, vol. 8, no. 3, pp. 383–407, Nov. 2006.

[9] M. Kobilarov, "Cross-entropy randomized motion planning," in *Proc. of the Robotics: Science and Systems*, Los Angeles, USA, June 2011.

[10] C. Urmson and R. Simmons, "Approaches for heuristically biasing RRT growth," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robotics and Systems*, Las Vegas, USA, Oct. 2003.

[11] D. Ferguson and A. Stentz, "Cost based planning with RRT in outdoor environment," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robotics and Systems*, Beijing, China, Oct. 2006.

[12] A. Ettlin and H. Bleuler, "Rough-terrain robot motion planning based on obstacleness," in *Proc. of the International Conference on Control, Automation, Robotics and Vision*, Singapore, Dec. 2006.

[13] J. Lee, C. Pippin, and T. Balch, "Cost based planning with RRT in outdoor environment," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robotics and Systems*, Nice, France, Sep. 2008.

[14] Y. Xu, J. Choi, and S. Oh, "Mobile sensor network navigation using Gaussian processes with truncated observations," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 1118–1131, Dec. 2011.

[15] L. Jaillét, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 635–646, Aug. 2010.

[16] C. E. Rasmussen and C. K. Williams, Eds., *Gaussian Processes for Machine Learning*. Cambridge: The MIT Press, 2006.

[17] P. T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[18] S. Petti and T. Fraichard, "Safe motion planning in dynamic environments," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, USA, Aug. 2005.