Sampling-based Path Planning on Configuration-Space Costmaps

Léonard Jaillet, Juan Cortés and Thierry Siméon

***Abstract*—This paper addresses path planning considering a cost function defined over the configuration space. The proposed Transition-based RRT planner computes low-cost paths that follow valleys and saddle points of the configuration-space costmap. It combines the exploratory strength of RRTs with transition tests used in stochastic optimization methods to accept or to reject new potential states. The planner is analyzed and shown to compute low-cost solutions with respect to a path quality criterion based on the notion of mechanical work. A large set of experimental results is provided to demonstrate the effectiveness of the method. Current limitations and possible extensions are also discussed.**

I. INTRODUCTION

Sampling-based path planning has proven to be an effective framework suitable for a large class of problems in domains such as robotics, manufacturing, computer animation and computational biology (see [1], [2] for a survey). These techniques handle complex problems in high-dimensional spaces but usually operate in a binary world aiming to find out collision-free solutions rather than the optimal path.

Specific path planning methods have been developed in field robotics for outdoor navigation, where the goal is to find optimal paths according to a cost function, usually computed from a model of the terrain. Classical grid-based methods such as A* or D* [3] can be used to compute resolution-optimal paths over a costmap. However, compared to sampling-based algorithms, these methods are limited to problems involving low-dimensional spaces that can be discretized and searched using grid search techniques.

Some recent works [4]–[8] have tried to bridge the gap between sampling-based planners and grid-based costmap planners. They mainly rely on the RRT algorithm [9], and are generally focused on specific applications (e.g. real time problems [7], [10] or statistical learning of feasible paths [8]) in the context of 2D robot navigation problems.

This paper presents a general algorithm, called Transition-based RRT (T-RRT),[1] for path planning on configuration-space costmaps. The algorithm considers a user-given cost function defined over the configuration space as an additional input to the standard path planning problem, and it produces solution paths that are not only feasible (e.g. collision-free), but also have a good quality with respect to the input costmap. For instance, the costmap may correspond in outdoor navigation problems to the elevation map of the terrain, aiming at computing motions that minimize climbing of high-slope regions. Also, in robotic manipulation problems, the cost function may be defined from distances to be maximized between the robot and some objects, in order to find high-clearance solution paths. Finally, in computational biology applications, the costmap can be viewed as the energy landscape of the conformational space to be considered for the simulation of low-energy molecular motions.

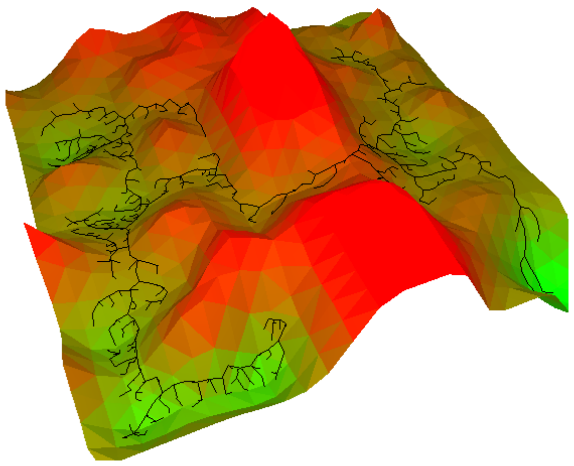


Fig. 1. Transition-based RRT on a 2D costmap (the elevation corresponds to the costs). The exploration favors the expansion in valleys and saddle points connecting low-cost regions.

The proposed algorithm combines the exploratory strength of RRTs with the efficiency of stochastic optimization methods (e.g. Monte Carlo optimization, simulated annealing) that use transition tests to accept or reject new potential states. The filtering of the transition test relies on the gradient of cost function along the local motion to connect a given state to the RRT tree, resulting in an expansion biased to follow the valleys and the saddle points of the configuration-space costmap (Figure 1). Solution paths computed by T-RRT fulfill a quality property based on the notion of mechanical work, also introduced in the paper as an effective criterion to evaluate path quality for costmap planning.

The paper is organized as follows. After a brief presentation of related work (Section II), we introduce and discuss the notion of Minimal Work paths (Section III). These paths are optimal according to a new criterion called mechanical work that is used to evaluate path quality. Comparison with other existing criteria shows the advantage of this criterion that may be more suitable in many situations, since it yields better paths following the low-cost valleys of the costmap. Additional properties of Minimal Work are also presented for a deeper understanding of this notion. Section IV describes the T-RRT

Léonard Jaillet is with the Institut de Robòtica i Informàtica Industrial CSIC-UPC; c/ Llorens i Artigas 4-6, 08028 Barcelona, Spain (e-mail: ljaillet@iri.upc.edu). Juan Cortés and Thierry Siméon are with CNRS; LAAS; 7 avenue du colonel Roche, F-31077 Toulouse, France, and with Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077 Toulouse, France (e-mail: jcortes@laas.fr; nic@laas.fr).

[1]The T-RRT planner was introduced in a shorter version published in [11].

algorithm, including the methods for self-tuning of parameters and for the expansion rate control. Section V shows how T-RRT implicitly computes Minimal Work paths and discusses its probabilistic completeness. An experimental validation of the planner is conducted in Section VI. The overall efficacy of T-RRT is shown on different problems and positively compared with other existing techniques [4], [6]. This section also analyzes the influence of the intrinsic parameters of the algorithm on the overall performance, and results indicate that no specific tuning is actually needed. Section VII presents some extensions of T-RRT, and conclusions are outlined in Section VIII.

II. Related Work

Early potential field methods [12], as well as their combination with strategies for escaping local minima, e.g. the randomized planner of [13], rely on some numerical field defined over the configuration space that may be viewed as a specific kind of costmap. Note however that the artificial potential field of these methods is only defined as a way for planning collision-free paths, without considering path optimality. Thus, these methods do not address the problem considered here of computing low-cost, feasible paths from an arbitrarily complex costmap given as input to the planner.

Recent sampling-based planners have proven to be very effective at finding feasible solutions that can be locally optimized in a post-processing stage. Local path optimization methods such as the shortcut algorithm [14] are generally used to improve path quality with respect to simple criteria, like path length, clearance, or a combination of both [15]. These smoothing methods only aim to locally improve a solution path, as opposed to the global exploration algorithm proposed in the paper. Moreover, their extension to arbitrary cost functions has not yet been addressed, and the resulting efficacy of such extension remains to be further evaluated.

Only few papers consider sampling-based path planning on arbitrary cost spaces. In [4], an adaptation of the RRT-Connect planner is used to find low-cost paths for rough terrain navigation. The idea is to keep new configurations only if their cost is under a given threshold, first initialized to a low value, and then iteratively increased during the search. One limitation of this technique comes from the non-decreasing threshold, which limits the efficiency of low-cost search to the vicinity of the root nodes. To overcome this issue, the extension proposed in [5] considers multiple RRTs trees grown from randomly sampled root configurations. However, this solution still expects an appropriate number of initial samples in order to get enough low-cost seeds among the space. Moreover, it requires a manual tuning of the parameter that controls the cost threshold growth rate. This tuning is highly problem-dependent.

In [6], the heuristically-guided RRT (hRRT) biases the search using a quality measure based on the integral of the cost along the path from the root node and an estimation of the optimal cost to the goal. Such an approach, inspired from graph search techniques, can also be found in the context of real-time applications [7], [10] and statistical learning of feasible paths [8]. However, with these techniques, the estimated *cost to goal* is heuristic and tends to bias the search straight toward the goal at the expense of lower-quality solution paths. Moreover, the aforementioned methods have only been demonstrated on simple low-dimensional examples with discrete cost states (invalid, low cost, high cost). Their scalability and performance for problems involving complex cost spaces in higher dimensions are yet to be established.

The T-RRT algorithm introduced below is inspired by Monte Carlo optimization techniques. Developed in order to find global optima in very complex spaces [16], they introduce randomness as a means to avoid local minima traps. Many variants have been developed (e.g. random walk, simulated annealing [17]). The basic exploration process typically relies on successive transition tests, performed using the Metropolis criterion (see Section IV-B). Note also that the Probabilistic Conformational Roadmap [18] developed for exploring molecular energy landscapes in computational biology applications integrates a similar transition test in the PRM framework [19].

III. Minimal Work Paths

This section introduces the mechanical work criterion to measure path quality in a space that is mapped by a given cost function. Paths that are optimal according to this criterion are called Minimal Work (MW) paths. The T-RRT algorithm presented in the next section tends to produce such MW paths, as shown by the theoretical analysis and the experimental results in Sections V and VI respectively. First, we introduce the notion of MW paths and illustrate how this criterion generally yields more natural solution paths (i.e. paths following well the low-cost valleys of the costmap) compared to other existing path quality measures.

A. Notation

Let us consider a system with a configuration space $\mathcal{C}$, possibly constrained by "binary" obstacle regions. Let us also consider a cost function $c : \mathcal{C} \rightarrow \mathbb{R}_+^*$ mapping this space, i.e. a cost $c(q) > 0$ can be computed for each $q \in \mathcal{C}$. This cost function c is assumed to be continuous. A path $\mathcal{P}$ of length l is represented by a unit-speed parametric function[2] $\tau : [0, l] \rightarrow \mathcal{C}$ with $\tau(s) = q_s \in \mathcal{P}$. Then, we define the parametric cost function $v : [0, l] \rightarrow \mathbb{R}_+^*$ of a path as $v(s) = c \circ \tau(s) = c(q_s)$.

B. Classical Path Quality Measures

Several criteria have been proposed to evaluate the quality of a path from its parametric cost function, e.g. maximal cost [5], average cost [5]–[7], or costs sum over discrete path configurations [5], [8] (as a way to approximate the integral of the cost along the path). The maximal cost criterion is the most limited one since it only relies on a point value of the parameterized cost function. The average cost can also be misleading since it does not account for path length (a path involving many detours inside a low-cost region will have an average cost smaller than a path going straight through this

[2]This representation assumes that the parameterized curve representing the path is regular, which simplifies the notation.

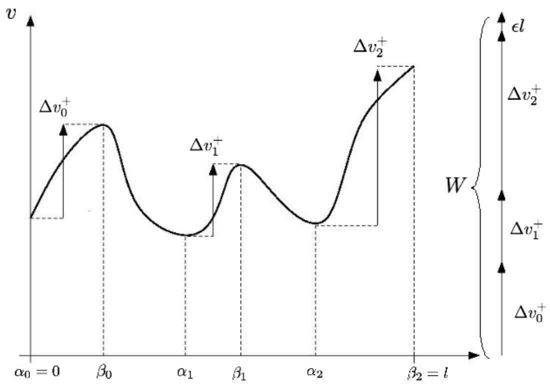


Fig. 2. Decomposition of a path into portions of monotonic cost variation. α_i and β_i correspond respectively to local minima and maxima. The mechanical work (right) is the sum of positive cost variations between consecutive extrema plus a small value ϵl proportional to the path length.

region). Thus, the integral of the cost along a path appears to be a more reliable criterion. It is mathematically defined as:

$$S(\mathcal{P}) = \int_0^l v(s)ds.$$

A discrete approximation of the integral leads to:

$$S(\mathcal{P}) \sim \frac{l}{n} \sum_{k=0}^{n-1} v(s_k), \text{ with } s_k = \left(\frac{k}{n-1}\right) l.$$

In what follows, optimal paths according to the Integral of the Cost criterion are called IC Paths. The next section introduces an alternative way for measuring path quality based on the notion of mechanical work.This alternative technique will then be compared to IC paths in Subsection III-D.

C. Mechanical Work of a Path

The key idea is that positive variations of the parametric cost function can be seen as forces acting against motion and thus producing mechanical work. We propose to use this loss of "energy" induced by the mechanical work for measuring the quality of a path. In the case of negative variation of costs, the system does not lose any energy. Then, a small penalty proportional to the distance is added in order to favor shortest paths of equal mechanical energy. Based on this principle, the mechanical work of a path is defined as:

$$W(\mathcal{P}) = \int_{\mathcal{P}^+} \frac{\partial v}{\partial s} ds + \epsilon \int_{\mathcal{P}} ds, \tag{1}$$

where $\mathcal{P}^+$ represents the portions of path with positive slopes (i.e. where the parametric cost function is strictly increasing), and ϵ is assumed to be very low compared to cost values.

The continuous expression of W in Equation (1) can be transformed into a discrete formulation expressed from the local extrema values along the path:

$$\begin{aligned} W(\mathcal{P}) &= \sum_i (v(\beta_i) - v(\alpha_i)) + \epsilon l \\ &= \sum_i \Delta v_i^+ + \epsilon l, \end{aligned} \tag{2}$$

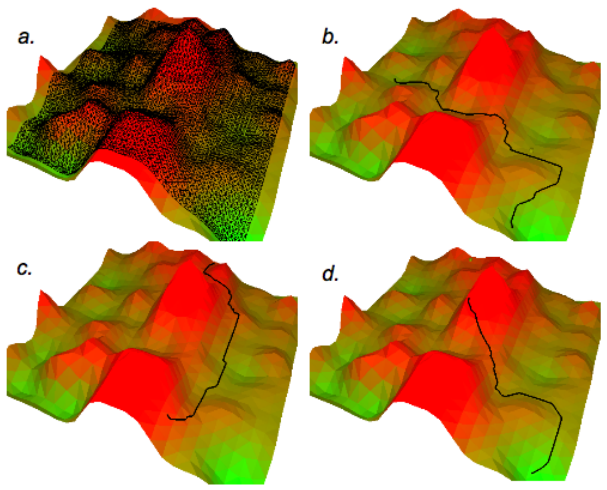


Fig. 3. Minimal Work solution paths. The paths are computed using the A^* algorithm within a 2D grid discretizing the space (a). The examples illustrate respectively down-to-down (b), top-to-top (c), and top-to-down (d) queries.

where α_i and β_i are consecutive minima and maxima of the costs along the paths and $\Delta v_i^+ = v(\beta_i) - v(\alpha_i)$ are the positive variations between two consecutive extrema (Figure 2). The mechanical work of a path is simply obtained by summing up the positive differences between extrema of its parameterized cost function and adding ϵl in order to favor shortest paths among the ones having equally positive cost variations. Paths that minimize the mechanical work for a given query are called Minimal Work (MW) paths.

Figure 3 shows examples of MW paths for several queries on a 2D hilly costmap. The paths were computed using a standard A^* search performed on a grid discretizing the two-dimensional landscape. As one can see, the shapes of the MW paths appear to be suitable in the sense that they follow as much as possible the low-cost regions of the space. In order to better state the pertinence of the mechanical work criterion, we first compare it with the IC criterion in the next subsection. Then, we state some interesting properties of the MW criterion in Subsection III-E.

D. Minimal Work vs. Integral of the Cost

This section compares the optimal solutions for the integral of the cost S (IC paths) and for the mechanical work W (MW paths) on representative cost spaces.

1) Constant Slope: Let us first consider the example of a planar landscape with a constant slope. In this simple case, IC solutions can be numerically characterized from calculus of variations. As shown in Figure 4, the solutions obtained for two different slopes show that IC paths (in black) are not intuitive and moreover depend on the plane inclination. In contrast, the MW path is in both cases the trivial straight-line path (in blue). Indeed, the cost of MW paths is always lower-bounded by the cost variation between the initial and final configurations. In situations for which the query configurations can be connected through a set of paths having a monotonic cost variation (as for the specific case of a constant slope landscape), the MW path will be the shortest

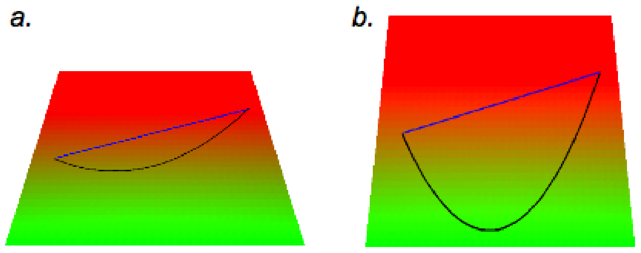


Fig. 4. The straight-line MW path (blue) and two different IC paths for two different inclinations of the plane representing the cost function.

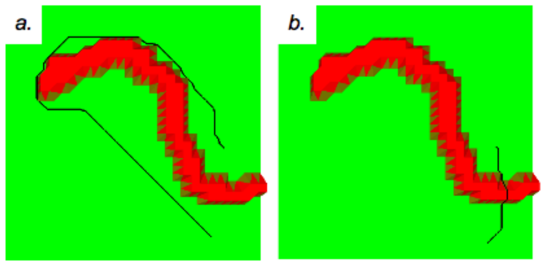


Fig. 5. High-cost barrier problem: (a) MW path, (b) IC path.

one among the set of minimal cost variation paths, yielding a straight-line solution for the planar slope example.

2) High-Cost Barrier: This example corresponds to a flat cost surface with a high-cost barrier that should be preferably avoided (Figure 5). In this case, the MW path is the shortest path getting around the barrier (Figure 5a) while the IC solution is a direct path that crosses the barrier (Figure 5b). This example highlights another possibly negative feature of the integral of the cost criterion that may favor undesirable paths with short high-cost portions.

3) Hilly Costmap: In this more complex example, solution paths have to go through a saddle point to link the query configurations located at two opposite corners of the hilly landscape (Figure 6). The MW path makes necessary detours to follow low-cost valleys of the space. In contrast, the IC solution prefers shortest paths at the expense of local high costs (circled in blue in Figure 6b). Looking at the parameterized cost functions of the two kinds of optimal paths (Figure 7), one can see that the cost profile of the IC path (red) is globally much higher than the one of the MW path (green). This observation is particularly true when the IC path goes through the high-cost region avoided by the MW path.

Finally, Table I compares the costs of the two solutions with respect to various path quality measures. It shows that both the average and maximum costs are better for the MW path than for the IC path. Indeed the IC path characteristics are intermediate between the ones of the MW path and of a simple straight-line path, not biased to avoid high-cost regions.

The results highlight some interesting features of the MW criterion. Compared to IC paths, MW paths avoid steep variations of the cost function. This may be particularly important in applications such as outdoor navigation (for avoiding high-slope motions), or computational biology (for minimizing the

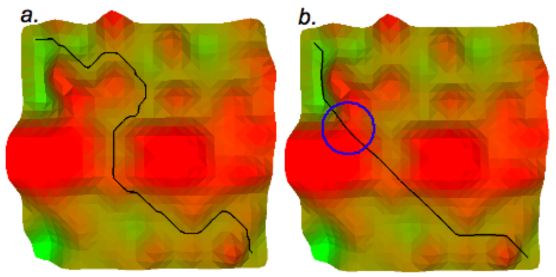


Fig. 6. Hilly costmap problem: (a) MW path, (b) IC path.

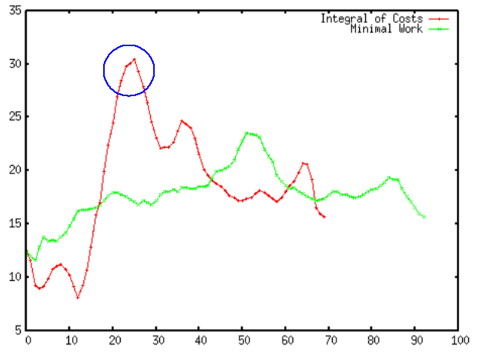


Fig. 7. Parameterized cost functions of the MW path (green) and the IC path (red) shown in Figures 6a and 6b.

crossing of high-energy barriers). Besides, in the presented cases, MW paths look more natural. In the next subsection, we present some additional properties of MW paths for a deeper understanding of the mechanical work criterion.

E. Minimal Work Path Properties

1) Negative Slopes Minimization: Section III-C states that minimizing the mechanical work means minimizing the amount of positive cost variations. A first property is that, between two given configurations, it also leads to minimizing negative cost variations. Indeed, the total amount of cost variations along the path can be expressed as:

$$v(l) - v(0) = \sum_i \Delta v_i^+ + \sum_j \Delta v_j^-,$$

where Δv_i^+ and Δv_j^- are the intervals of positive and negative cost variation respectively. Using Equation (2), we obtain:

$$W(\mathcal{P}) = v(l) - v(0) + \epsilon l + \sum_j |\Delta v_j^-|. \tag{3}$$

Because $v(l)$ and $v(0)$ are constants and ϵl is small relative to cost values, Equation (3) states that minimizing W is equivalent to minimizing the last term in the right part of the equation, that is, the total amount of negative cost variations.

2) Cost Variations Minimization: Since the Minimal Work path $\mathcal{P}$ minimizes both positive and negative cost variations, $\mathcal{P}$ is indeed the path that minimizes any cost variation between two given configurations. Let $V(\mathcal{P})$ be a function that sums positive and negative variations:

$$V(\mathcal{P}) = \sum_i \Delta v_i^+ + \sum_j |\Delta v_j^-|.$$

	$Length$	C_{ave}	C_{max}	S	W
MW path	186	**17.9**	**23.6**	3324	**15.9**
IC path	139	18.6	30.4	2592	32.5
Str. Line	127	22.6	39.6	2877	41.0

TABLE I
MW AND IC OPTIMAL PATHS OF FIGURE 6 COMPARED TO A REFERENCE STRAIGHT-LINE SOLUTION.

Using Equations (2) and (3), we get:

$$V(\mathcal{P}) = 2.W(\mathcal{P}) - (v(l) - v(0) + 2\epsilon l). \tag{4}$$

Thus, the ordering of the paths remains the same regardless of the criterion (V or W), which, indeed, means that they are equivalent. However we will keep the formulation of Minimal Work path since this notion facilitates the analysis of the T-RRT algorithm.

3) Reversibility of Minimal Work Paths: Let ${}^{-1}\mathcal{P}$ be the reverse path of $\mathcal{P}$. Since the parametric cost functions v and ${}^{-1}v$ have opposed variations, i.e. $\Delta{}^{-1}v^{+} = |\Delta v^{-}|$, we have:

$$W({}^{-1}\mathcal{P}) = \sum_j |\Delta v_j^{-}| + \epsilon l,$$

and using Equation (3), we get:

$$W({}^{-1}\mathcal{P}) = W(\mathcal{P}) + v(0) - v(l). \tag{5}$$

Consequently, the mechanical work of a path is equal to the mechanical work of its inverse, except for a constant. This property allows us to speak about the MW path between two configurations without the need for orienting the path.

IV. TRANSITION-BASED RRT

A. Main Algorithm

The T-RRT algorithm combines the advantages of two methods. First, it benefits from the exploratory strength of RRT-like algorithms resulting from their expansion bias toward large Voronoi regions of the space. Additionally, it integrates features of stochastic optimization methods developed for computing global minima in complex spaces: it uses transition tests to accept or reject potential states.

Algorithm 1 shows the pseudo-code of the T-RRT planner. Similarly to the *Extend* version of the basic RRT algorithm [20], a randomly sampled configuration q_{rand} is used to determine both the nearest tree node q_{near} to be extended and the extension direction. The extension from q_{near} is performed toward q_{rand} with an increment step δ. In the case of T-RRT, δ has to be small enough to avoid cost picks to be missed by the linear interpolation between q_{near} and q_{new}. This stage also integrates collision detections in the presence of "binary obstacles." Thus, if the new portion of path leads to a collision, a null configuration is returned and the extension fails, independently of the associated costs. This extension process ensures the bias toward unexplored free regions of the space. The goal of the second stage is to filter irrelevant configurations regarding the search of low-cost paths before inserting q_{new} in the tree. Such filtering is performed by the `TransitionTest` function. It relies

Algorithm 1: Transition-based RRT

```
input  : the configuration space $\mathcal{C}$;
         the cost function $c : \mathcal{C} \rightarrow \mathbb{R}_{+}^{*}$;
         the root $q_{\text{init}}$ and the goal $q_{\text{goal}}$;
output : the tree $\mathcal{T}$;
begin
    $\mathcal{T} \leftarrow$ InitTree($q_{\text{init}}$);
    while not StopCondition($\mathcal{T}$, $q_{goal}$) do
        $q_{\text{rand}} \leftarrow$ SampleConf($\mathcal{C}$) ;
        $q_{\text{near}} \leftarrow$ NearestNeighbor($q_{\text{rand}}, \mathcal{T}$);
        $q_{\text{new}} \leftarrow$ Extend($\mathcal{T}, q_{\text{rand}}, q_{\text{near}}$);
        if $q_{new} \neq NULL$
        and TransitionTest($c(q_{near}), c(q_{new}), d_{near-new}$)
        and MinExpandControl($\mathcal{T}$, $q_{near}$, $q_{rand}$) then
            AddNewNode($\mathcal{T}$, $q_{\text{new}}$);
            AddNewEdge($\mathcal{T}$, $q_{\text{near}}$, $q_{\text{new}}$);
end
```

on the Metropolis criterion commonly used in stochastic optimization methods. This test integrates a self-tuning technique in order to automatically control its filtering strength, and thus to ensure continuous growth of the tree. Finally, the `MinExpandControl` function forces the planner to maintain a minimal rate of expansion toward unexplored regions of the space, and avoids possible blocking situations during the search. The following subsections detail the `TransitionTest` and `MinExpandControl` functions.

B. Transition Test

The `TransitionTest` function is presented in Algorithm 2. First, configurations with a higher cost than the maximum cost threshold c_{max} are filtered. The probability of acceptance of a new configuration is defined by comparing its cost c_j relatively to the cost c_i of its parent in the tree. This test is based on the Metropolis criterion initially introduced in statistical physics and molecular modeling. The transition probability p_{ij} is defined as:

$$p_{ij} = \begin{cases} exp(-\frac{\Delta c_{ij}}{K \cdot T}) & \text{if } \Delta c_{ij} > 0, \\ 1 & \text{otherwise,} \end{cases} \tag{6}$$

where:

- $\Delta c_{ij} = \frac{c_j - c_i}{d_{ij}}$, is the slope of the cost, i.e. the cost variation divided by the distances between the configurations.[3]
- K is a constant value used to normalize the expression. It is based on the order of magnitude of the considered costs. K is taken as the average cost of the query configurations since they are the only cost values known at the beginning of the search process.
- T is a parameter called *temperature* that is used to control the difficulty level of transition tests, as further explained below. Note that the term temperature is employed in analogy with methods in statistical physics, but in our case it does not have any physical meaning.

[3]Contrarily to classical Monte Carlo methods, the cost variation is normalized by the distance to the previous state since this distance is not necessarily constant.

Algorithm 2: TransitionTest(c_i, c_j,d_{ij})

```
begin
    $nFail$ = GetCurrentNFail();
    if $c_j > c_{max}$ then return False;
    if $c_j < c_i$ then return True;
    $p = exp(\frac{-(c_j - c_i)/d_{ij}}{K \cdot T})$;
    if Rand(0, 1) < p then
        $T = T/\alpha$;
        $nFail = 0$;
        return True;
    else
        if $nFail > nFail_{max}$ then
            $T = T \cdot \alpha$;
            $nFail = 0$;
        else
            $nFail = nFail + 1$;
        return False;
end
```

Algorithm 3: MinExpandControl($\mathcal{T}$, q_{near},q_{rand})

```
begin
    if Distance ($q_{near}, q_{rand}$) > $\delta$ then
        UpdateNbNodeTree($\mathcal{T}$);
        return True;
    else
        if $\frac{NbRefineNodeTree(\mathcal{T})+1}{NbNodeTree(\mathcal{T})+1} > \rho$ then return False;
        else
            UpdateNbRefineNodeTree($\mathcal{T}$);
            UpdateNbNodeTree($\mathcal{T}$);
            return True;
end
```

Using this transition probability, downhill transitions are automatically accepted whereas for uphill transitions, the chance of acceptance decreases exponentially with the cost increment.

1) Temperature Parameter: T is a key parameter of the algorithm since it defines the level of difficulty of a transition for a given cost increment. Low temperatures limit the expansion to slightly positive slopes. In contrast, higher temperatures enable the climbing of steeper slopes. Within methods involving the Metropolis criterion, the temperature is usually kept constant (e.g. Monte Carlo search) or decreases gradually as the search progresses (e.g. simulated annealing). In our algorithm, this parameter is dynamically tuned according to the information acquired during the exploration.

2) Adaptive Tuning: The TransitionTest function performs an adaptive tuning of the temperature during the search process (second stage of Algorithm 2). At the initialization, T is set to a very low value (e.g. 10^{-6}) in order to only authorize very easy positive slopes (and negative ones). Then, during the exploration, the number *nFail* of consecutive times the Metropolis criterion discards a configuration is recorded and used for temperature tuning. When the T-RRT search reaches a maximal number of rejections $nFail_{max}$, the temperature is multiplied by a given factor α. Each time an uphill transition test succeeds, the temperature is divided by the same factor α. Thus, the temperature automatically adapts itself such that an extension corresponding to a positive cost variation is performed in average every $nFail_{max}$ times. The influence of parameters α and $nFail_{max}$ is analyzed in the Results section.

C. Minimal Expansion Control

The adaptive temperature tuning introduced above ensures a given success rate of positive slope transitions. A possible side effect may appear when the tree expansion toward unexplored regions remains slow, and the new nodes contribute only to refine already explored regions. We discuss below this issue and explain how the MinExpandControl function overcomes this problem.

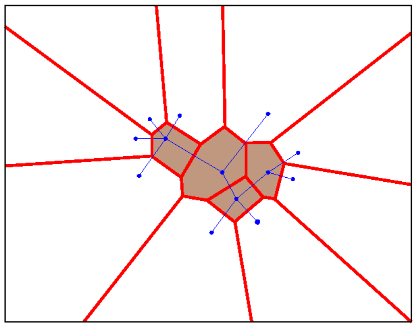


Fig. 8. *Frontier* nodes (in white regions) have a Voronoi region bounded by the space limits. On the contrarily, the Voronoi region of *non frontier* nodes is bounded by the Voronoi region of other nodes (in brown/gray regions).

1) Exploration versus Refinement: The behavior of the RRT expansion can be explained by distinguishing two types of nodes [21]. *Frontier* nodes are the external nodes of the tree with a Voronoi region bounded by the space limits whereas *non frontier* nodes are the internal ones, whose Voronoi region is entirely bounded by the Voronoi region of the other nodes (see Figure 8). Thus, the extension of a frontier node tends to explore new regions of the space and the extension of a non frontier node only refines the existing tree. The problem of unbalanced refinement and exploration modes was addressed in [21], [22] for standard RRTs. However, for T-RRT, the interaction between these two kinds of extensions is more subtle than for the basic RRT. Indeed, situations occur where the temperature is stabilized by new non frontier nodes refining the tree in easier regions of the space; however, the expansion toward new regions requires the development of frontier nodes. Figure 9a illustrates this issue with an example of a tree whose expansion has been slowed down by the too frequent insertion of non frontier nodes. Figure 9b shows the tree obtained using the minimal expansion control detailed below.

2) Minimal Exploration Rate: The proposed solution is to force the planner to explore new regions by controlling the ratio between exploration and refinement steps. Note that as long as the tree coverage remains limited compared to the size of the space, non frontier nodes have a Voronoi region much smaller than the one of frontier nodes. Hence, extension

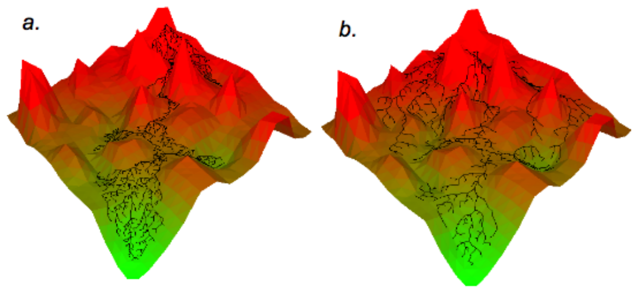


Fig. 9. Impact of the minimal expansion control on the T-RRT algorithm. Without control (a), the insertion of non frontier nodes tends to slow down the exploration by decreasing the temperature. With control (b), the planner is forced to keep exploring new regions of the space.

steps can be estimated as refinements or expansions depending on the distance between q_{near} and q_{rand}. For a large distance value, q_{near} has greater chances to be a frontier node, whereas a small distance value corresponds more probably to the case of a non frontier node extension. The control of minimal exploration rate is performed by the `MinExpandControl` function presented in Algorithm 3. If the distance q_{near}–q_{rand} is greater than the expansion step δ, q_{new} is considered to participate in the tree expansion, and it is inserted in the data structure. Otherwise q_{new} is considered to be participating in the tree refinement. The configuration is not inserted in the tree if it makes the ratio of non frontier nodes be greater than a given maximal value ρ. The influence of this parameter is further discussed in the Results section.

V. Theoretical analysis of T-RRT

A. *T-RRT and Minimal Work Path*

This section analyzes the relationship between T-RRT and the notion of Minimal Work path introduced in Section III. An important property is obtained first for the simplified case of a discrete search process. Then, we discuss the extension of this result to the general case of the T-RRT search.

1) Simplified Case: Consider a path search within a discrete set of n equal length possible paths, each one defined by a sequence of m edges and $m+1$ nodes (Figure 10). Using a T-RRT scheme, each expansion of a given path requires the path to be selected and the associated transition test to succeed. Thus, the probability P_k of a given path $\mathcal{P}_k$ to be entirely developed in m iterations is equal to:

$$P_k = \prod_{i\in[1,m]} {}^e p_i^k = \prod_{i\in[1,m]} ({}^s p_i^k)\cdot({}^t p_i^k)$$

where ${}^e p$ denotes the probability for a given node to be *extended*, ${}^s p$ is the probability to be *selected* and ${}^t p$ the probability of having an accepted *transition*. Also, we assume that the paths have equal chances of being extended at each step (i.e. the node selection process is not biased by Voronoi regions), that is:

$$P_k = \frac{1}{n^m}\prod_{i\in[1,m]} {}^t p_i^k.$$

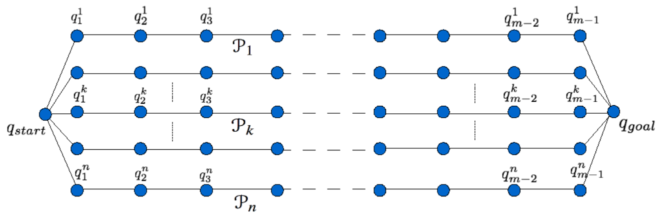


Fig. 10. Case of n equal-length paths. With T-RRT, the branches with the lowest mechanical work have the highest chances to reach the goal first.

If the transition probability depends only on the transition tests (i.e the `MinExpandControl` is omitted), we get:

$$P_k = \frac{1}{n^m}\prod_j e^{\frac{-\Delta v_j^{k+}}{K\cdot T_j}}.$$

Moreover, assuming that the temperature remains constant during the expansion, we have:

$$P_k = \frac{1}{n^m}\cdot e^{\frac{1}{K\cdot T}}\cdot e^{-\sum_j \Delta v_j^{k+}},$$

where the Δv_j^{k+} are summed over the positive variations of cost along the path k. Finally, since ϵl is negligible in the Mechanical Work expression, we get:

$$P_k = \frac{1}{n^m}\cdot e^{\frac{1}{K\cdot T}}\cdot e^{-W(\mathcal{P}_k)}. \quad (7)$$

Since $\frac{1}{n^m}\cdot e^{\frac{1}{K\cdot T}}$ is the same for all the paths, we obtain an important property for this simplified version: the paths with the lowest mechanical work have the highest probability of reaching the goal first.

2) General Case: One first assumption made in the analysis above is that each branch has an equal chance of being chosen for the expansion. In practice, the various paths developed by the T-RRT algorithm (from the root node to each leaf) are not spatially independent. Each branch expansion tends to increase its global Voronoi region, and thus, increases the chance for its nodes to be selected at the next iteration. This process reinforces the extension of the paths with the most favorable mechanical work, and increases the convergence of the planner toward lower cost solutions.

The simplified version assumed also that the temperature is constant. This parameter affects each path in the same way. Thus, we can argue that the property remains valid even when T varies during the search.

Finally, whereas the above property is established for the discrete case of equal-length solution paths, T-RRT search is performed among an infinite number of variable-length paths. Since shortest paths require less expansion steps to connect the queries, it is not possible to guarantee that paths of lower cost have always better chances to reach the goal first. However, as one can see from Equation (7), the mechanical work of a path affects exponentially its chances of success. This reveals how strongly the T-RRT exploration is implicitly biased toward solution paths of low mechanical work.

B. Probabilistic Completeness

The T-RRT algorithm is a probabilistically complete planner [19]. This property is directly inherited from the probabilistic completeness of the RRT planner (see Section 4 of [9]). The only difference is that in the present case, the extension steps can be rejected because of the transition tests, even in the case of a convex, open, n-dimensional subset of an n-dimensional configuration space. However, we argue that the success probability of the transitions is always strictly positive since the cost function takes finite values in this subset, and thus, the cost variations are bounded. As a result, the planner converges eventually toward an entire coverage of the considered subset, and the transition tests affect only the convergence rate of the algorithm.

VI. Experimental Results

A large set of experiments has been conducted to evaluate the performance of the planner. First, the general behavior of the method is presented on various problems. Second, its performance is compared to the one of existing methods, highlighting the good quality of the T-RRT solutions. Finally, we investigate the influence of some intrinsic parameters on the overall efficacy of the method. All the algorithms have been implemented within the path planning software *Move3D* [23]. The performance results summarized in the tables are values averaged over 10 runs.

A. General Performance

A variety of problems are proposed to illustrate the generality of the method. The examples vary not only in the geometrical complexity and the configuration space dimensionality, but also in the nature of the cost function. Two settings of T-RRT are considered: a *greedy* version of the planner referred to as T-RRT$_g$ that takes $nFail_{max} = 10$, and a *tempered* version, referred to as T-RRT$_t$, with $nFail_{max} = 100$. The latter leads to higher quality solution paths, but is more computationally expensive. We used $\alpha = 2$ in all the examples. The results obtained with the Extend version of the basic RRT planner are given as references. The tables also present comparative results with two existing cost-based methods that will be discussed in the next subsection.

The first set of experiments is performed on the two-dimensional cost space of Figure 1. In this example, the solution paths have to go through a saddle point to link the query configurations located at two opposite corners of the landscape. Figure 11 shows snapshots of the exploration tree and the solution path found (Figure 11c), which is close to the optimal one (Figure 11d). Table II presents the characteristics of the paths obtained with each planner.[4] It provides also values for the MW and IC optimal paths (computed with an A^* search within a 128×128 grid discretizing the landscape). The mechanical work of solutions obtained by the different methods is reported in the W column. The numbers in

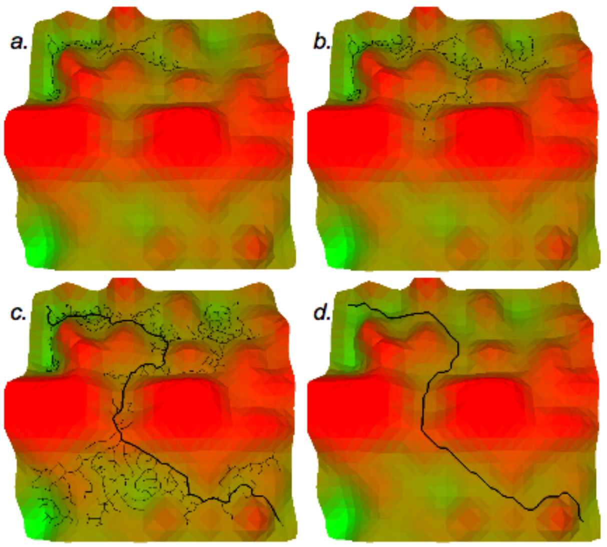


Fig. 11. Construction process of the Transition-based RRT planner (a-b). The solution path (c), is close to the optimal one (d) computed from a space discretization.

	Length	C_{ave}	C_{max}	$S \times 10^{-3}$	W	*Time*
RRT	148	22.6	41.2	3.3	45.1 (36.9)	0.1
T-RRT$_g$	182	18.2	25	3.9	28.0 (19.7)	0.9
T-RRT$_t$	214	17.1	23.2	4.0	23.1 (16.9)	11.0
MW path	186	17.9	23.6	3.3	15.9	-
IC path	139	18.6	39.6	2.6	41.0	-
Thresh.	157-192	17.8-20.9	23.7-32.7	3.3-3.4	27-30 (21-24)	0.3-329
hRRT	155	20.9	33.3	3.2	41.4 (32.9)	0.6

TABLE II
Comparative results for the costmap problem.

parentheses integrate the effect of some local smoothing of the solution path with a simple procedure based on the shortcut algorithm [14]. As one can see from the table, the mechanical work of the reference RRT path is almost 3 times higher than for the optimal MW solution, and smoothing does not successfully get close to the optimal value (36.9 vs. 15.9). In comparison, the mechanical work of the path obtained with the tempered version of T-RRT is only 45% higher than for the MW path, and it becomes only 6% higher than the optimal value after smoothing. Most importantly, the overall shape of the T-RRT solution is very close to the optimal-MW path and follows the same low-cost regions. Also, note that the relatively slight loss of path quality of the greedy version is compensated by a much smaller computing time ($0.9s$ vs. $11.0s$). Comparative results obtained with other existing costmap planners (Thresh. and hRRT rows in the table) are discussed in Subsection VI-B.

In the next experiment, a 6-dof manipulator arm is carrying a stick in a 3D workspace with obstacles (Figure 12). Here, the goal is to extract the stick from a hole, while keeping the stick as far as possible from the obstacles. Thus, the cost function considered here is the inverse of the distance between the stick and the obstacles. Results are presented in Table III.

The costs of the T-RRT solution paths are considerably lower than the ones of RRT. This shows the effectiveness of the planner for finding low-MW paths in higher-dimensional

[4]In the case of RRT, since there is no obstacle in the scene, connection attempts to the goal are only performed when $d(q_{new}, q_{goal}) < 15\delta$ to avoid getting a trivial straight-line solution.

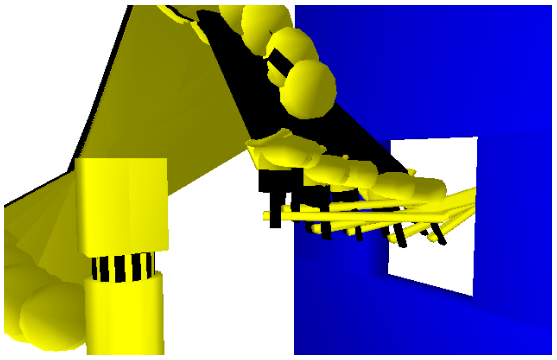


Fig. 12. Stick extraction problem. A 6-dof manipulator arm has to extract a stick from a hole. The T-RRT solution path keeps the stick horizontal to maximize its distance to the obstacles.

	Length	C_{ave}	C_{max}	S	W	*Time*
RRT	55.3	6.6	377	363	1196	1.5
T-RRT$_g$	53.9	0.4	1.0	21.8	1.9	7.4
T-RRT$_t$	51.5	0.3	0.9	17.4	1.1	32.8
Thresh.	53.2-54.9	0.6-3.2	1.6-40.1	32.0-172	4.2-196	1.6-2.7
hRRT	53.5	5.0	341	265	786	3.0

TABLE III
COMPARATIVE RESULTS FOR THE STICK EXTRACTION PROBLEM.

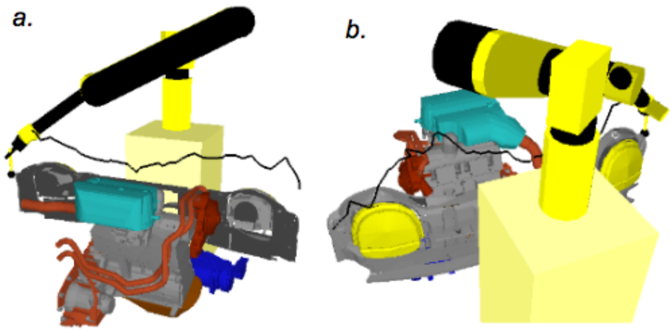


Fig. 13. Car part inspection problem. The figure shows the path for a 6-dof arm manipulating a sensor (black sphere) that needs to remain close to the surface during the inspection task.

	Length	C_{ave}	C_{max}	$S \times 10^{-3}$	W	*Time*
RRT	3785	457	875	1730	1434	0.98
T-RRT$_g$	3515	24.4	87.5	85.8	400	16.4
T-RRT$_t$	3396	4.3	28.1	14.6	187	206
Thresh.	2848-3942	36-186	79-436	116-529	447-635	4.3-19
hRRT	3127	253	452	792	943	18.0

TABLE IV
COMPARATIVE RESULTS FOR THE CAR PART INSPECTION PROBLEM.

spaces. Whereas the basic RRT planner produces erratic paths, T-RRT solutions tend to keep the stick horizontal during its extraction from the hole in order to remain as far as possible from the obstacles. Once again, the slight loss of path quality of the greedy version of the T-RRT (1.9 vs. 1.1) is compensated by a significant speed-up ($7.4s$ vs. $32.8s$).

The third scenario involves the same manipulator arm carrying a sensor with a spherical extremity for the inspection of the surface of a car part. The goal here is to keep the sensor close to the surface of the car part during the motion, in order to satisfy the requirements for the surface following task (Figure 13, Table IV). Note that for such a scenario where the robot is subject to task space constraints, specific path planning schemes also exist (e.g. [24]).

As to be expected, the T-RRT computing time is higher than the one of RRT because computing a collision-free path with RRT and without any cost consideration is a much easier problem than obtaining a solution that minimizes the distance to the inspected surface. However, regarding paths quality, the mechanical work of T-RRT$_g$ and T-RRT$_t$ are 3.6 times and 7.7 times lower than the one of RRT respectively. The average and maximal costs reported in Table IV are interesting indicators to get a better idea of the quality of the results, since they correspond directly to the average and maximal distances between the sensor and the part. For a distance reference, the diameter of the black sphere at the extremity of the sensor is 40mm. For T-RRT$_g$, the maximal cost corresponds approximately to twice this value, whereas the average distance is close to the sensor radius. In the case of T-RRT, solution paths follow the surface of the part so well that the maximal distance never exceeds the size of the sphere and the average one is about one tenth of this diameter.

Finally, the last scenario corresponds to a molecular model shown in Figure 14. The task is to compute the pathway extracting the ligand (small molecule in red/dark) from the active site located inside a protein. This problem can be seen as a mechanical disassembly path planning problem for the free-flying ligand [25]. Energetic constraints are translated into geometric ones by considering a steric model of the molecule, and a collision detection algorithm [26] is applied as a geometric filter that rejects conformations with prohibitively high van der Waals (VdW) energy. The cost function considered for this problem is the inverse of the distance between the ligand and the protein. The interest of this molecular model is to provide a simple way to quantify the quality of the computed solution path. The ligand free space can be simply dilated by shrinking the atoms radii. The results reported in Table V are obtained by applying both RRT and T-RRT algorithms on the shrunk model shown in Figure 14b (25% of VdW radii).

The T-RRT solution paths have a much lower cost compared to the one computed by RRT. The higher clearance of the T-RRTs solutions are also quantified by the maximal VdW ratio indicated in the last row of the table. This maximal ratio was obtained by testing solution paths by increasing the VdW radii until a collision was detected between the ligand and the protein. While no growing is possible for the RRT solution, the T-RRT paths (computed with a 25% ratio) remain valid up to 65% and 69% growing, depending of the variant. These values are close to the maximal radius that allows the ligand to exit (80%). The high clearance of T-RRT paths reflects their good quality with respect to the considered distance-based cost.

B. *T-RRT vs. Existing Methods*

T-RRT has been compared with two existing cost-based planners: the maximal `Threshold` technique proposed by Ettlin and Bleuler [4] and the heuristically-guided RRT (*hRRT*) of Urmson and Simmons [6]. Results obtained for the set of experiments with these planners are reported in the two last

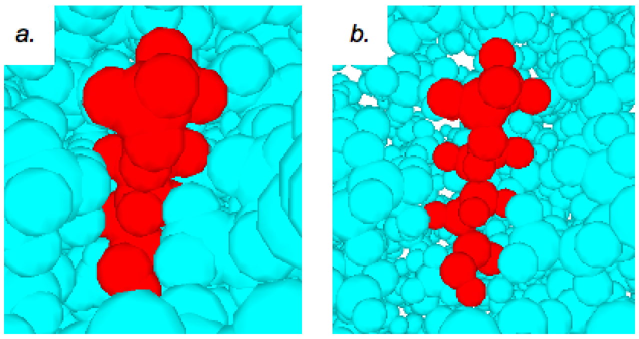


Fig. 14. Two representations of the same ligand-protein “disassembly” problem, with different van der Waals radii: (a) maximal radius and (b) shrunk radius. The goal is to compute paths that maximize the clearance and thus remain valid for large van der Waals radii.

	$Length$	C_{ave}	C_{max}	S	W	*Time*	VdW_m
RRT	59	14	2236	826	471	0.7	25
T-RRT$_g$	62	0.4	1.0	22	1.25	8.39	65
T-RRT$_t$	64	0.3	0.9	22	1.0	426	69
Thresh.	58-62	.4-.7	1.1-3.7	25-42	1.4-4.3	4.8-73.5	34-62
hRRT	59	8.3	733	490	353	15.2	26

TABLE V
COMPARATIVE RESULTS FOR THE LIGAND-PROTEIN PROBLEM.

Hilly costmap

	Time			*W*		
	nFail$_{max}$			*nFail*$_{max}$		
α	10	100	1000	10	100	1000
2	**0.9**	**11.0**	121	**28.0**	**23.1**	23.4
10	0.7	7.1	93.4	32.1	26.6	24.4
50	0.7	7.0	85.5	32.0	28.7	24.8

Stick extraction

	Time			*W*		
	nFail$_{max}$			*nFail*$_{max}$		
α	10	100	1000	10	100	1000
2	**7.4**	**32.8**	226	**1.9**	**1.1**	1.1
10	8.2	31.0	218	5.5	2.9	2.4
50	7.0	29.8	226	3.5	3.3	1.2

Car part inspection

	Time			*W*		
	nFail$_{max}$			*nFail*$_{max}$		
α	10	100	1000	10	100	1000
2	**16.4**	**206**	2012	**400**	**187**	166
10	14.3	171	1697	439	188	165
50	15.9	167	1583	507	212	203

TABLE VI
INFLUENCE OF THE α AND *nFail*$_{max}$ PARAMETERS.

Bold values are the default settings used in previous tests.

rows of Tables II to V. In the case of the *threshold* method, results are highly sensitive to the threshold growing speed, and thus, reported data correspond to the extremal values obtained when varying this parameter in the range (1-100).

Regarding the mechanical work criterion, results show that T-RRT$_t$ provides significantly better solutions than existing methods in all tests. Remarkably, T-RRT solutions are also better with respect to the IC criterion in the three more difficult problems involving a six-dimensional cost space.

The bad overall performance of the *hRRT* method is due to the strong bias introduced by the heuristic that steers the exploration toward the goal, resulting in a poor exploratory ability, making it unable to circumvent high-cost regions and find higher quality paths. Comparatively, the threshold technique can provide solution paths whose quality is close to the one of T-RRT$_g$, but its performance is highly sensitive to the parameter that regulates the variation speed of the threshold. Depending on the value of this parameter, the running time increases up to 1000 times for the costmap problem, the mechanical work increases up to 47-fold for the stick extraction problem, and both the running time and the mechanical work are notably affected by the threshold speed value for the car part inspection problem. Furthermore, this sensitive parameter is problem-dependent and has to be tuned for each application, whereas T-RRT parameters remain robust to problem changes, as shown in the next subsection.

C. *Influence of Intrinsic Parameters*

We analyze now the influence of the main parameters of the T-RRT algorithm. Experiments are performed on three problems that correspond to three different types of cost functions: the hilly costmap (Figure 11), the stick extraction problem (Figure 12), and the car part inspection problem (Figure 13). The results are presented in Tables VI and VII.

1) Temperature Variation Control: nFail$_{max}$ and α are the two parameters that control the derivative of the temperature, and hence the selectivity of the transition test (as explained in Section IV-B).

Table VI shows that *nFail*$_{max}$ is an important parameter that determines the appropriate balance between time performance and solution path quality. In the costmap problem, when *nFail*$_{max}$ is increased by a factor of 10, the running time also increases 9 to 13-fold. Its influence on the runtime performance is less direct on the two manipulator problems (due to the additional cost of collision checking), even though the tendency is the same. Finally, note that higher values of *nFail*$_{max}$ improve path quality, but only up to a point: the quality increases when *nFail*$_{max}$ varies from 10 to 100, but remains approximately constant from 100 to 1000.

Regarding the α parameter, results show that it affects only slightly the behavior of the algorithm even if higher values tend to increase the time performance while decreasing the path quality. Overall, values *nFail*$_{max}$ $= 100$ and $\alpha = 2$ provide the best results for the three examples and are used as default setting for all tests.

2) Expansion vs. Refinement Control: Table VII presents results for various values of the ρ parameter used in the `MinExpandControl` function to set the maximal ratio of refinement nodes.

In the first line of the table, $\rho = 1$ means that the `MinExpandControl` function is inactive. The results for the 2D hilly costmap highlight the importance of this function, the computing time being much higher when $\rho = 1$. This example illustrates the case where the refinement process slows down the exploration by decreasing the temperature. This effect is less visible in the two other examples where refinement steps are less likely to happen because of the large size of the space. Results for the other settings (i.e. $\rho \neq 1$) are quite similar, meaning that ρ does not require to be tuned precisely. In all

	Hilly costmap		Stick extraction		Car part inspection	
ρ	*Time*	*W*	*Time*	*W*	*Time*	*W*
1	420	19.6	30.3	1.1	201	192
1/2	16.7	23.7	33.4	1.2	198	202
1/10	**11.0**	**23.1**	**32.8**	**1.1**	**206**	**187**
1/100	9.7	23.8	32.0	1.2	269	263

TABLE VII
INFLUENCE OF THE ρ PARAMETER.

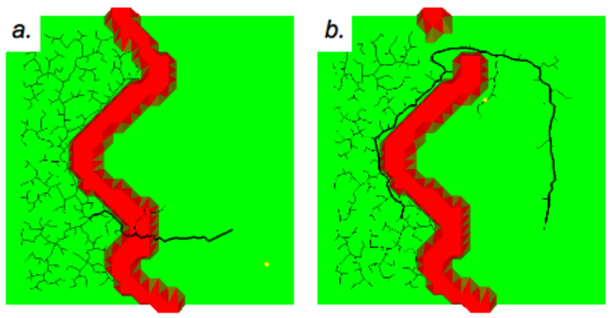


Fig. 15. A tricky problem for T-RRT. A large low-cost region has to be explored before deciding to cross the high-cost barrier: useless in (a) or leading to a better solution (b).

experiments, the default setting $\rho = 1/10$ appears to be a good compromise between computing time and path quality.

VII. EXTENSIONS

A. *Bi-directional T-RRT*

Similarly to the bi-directional version of the RRT planner [9], a bi-directional T-RRT can be envisaged. However, a naive approach using the same transition test for both trees would lead to poor quality solutions. It would tend to create paths with consecutive downhill and uphill cost variations, corresponding to branches expanded from the *init*-tree and *goal*-tree respectively, and may fail to find a more flat solution path of lower-MW cost. A better alternative, using the property of Subsection III-E2, which states that the MW paths minimize any cost variations, is to modify transition tests in order to filter both positive and negative cost variations when expanding the two trees. This can be achieved easily by replacing the transition probability p_{ij} of Equation (6) by the expression $p_{ij} = exp(-\frac{|\Delta c^*_{ij}|}{K.T})$. Preliminary results show that this approach performs well in problems where positive and negative cost variations for the best cost paths are globally of the same amplitude. However, in problems where the profile of the cost between query nodes is asymmetric, it turns out to reject too many configurations during the transition test, which degrades the performance. In that case, a method based on a more sophisticated transition test should be designed.

B. *Toward a Greedy Anytime T-RRT*

In this section, we discuss a possible extension of T-RRT for performance improvement in tricky situations such as the one illustrated in Figure 15. In this example, the large low-cost region has to be fully explored before determining the need to cross the higher cost barrier (a) or discovering the low-cost passage that yields a better solution (b). In both cases, the

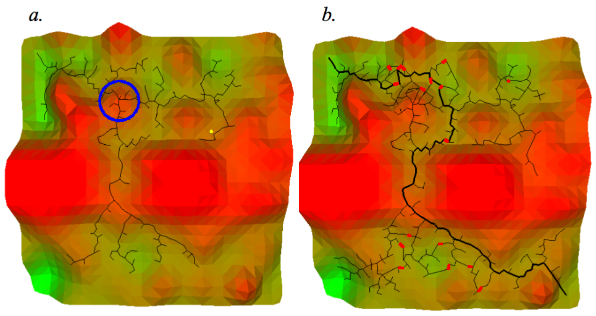


Fig. 16. (a) Initial tree built using a greedy T-RRT version. (b) The addition of cycles (in red) leads to higher quality paths.

greedy T-RRT$_g$ version may rapidly cross the barrier, and thus speed-up the computation compared to the tempered T-RRT$_t$. However, it may miss the preferred detour path in problem (b), for which a longer exploration is needed to find the passage. To keep the performance of an aggressive exploration while avoiding this issue, we propose to combine the greedy version of the planner with a cycle addition mechanism. The idea is to create cycles in the tree when good paths initially missed during the search are discovered afterwards. The idea has been tested using the technique of [27] for cycle addition, leading to early results. Figure 16 shows an initial tree built using a greedy version of T-RRT that goes through a medium cost region (circled in blue on Figure 16a) that could have been avoided. The addition of cycles provides alternative paths and yields higher quality solutions (Figure 16b).

VIII. CONCLUSION AND FUTURE WORK

We have presented a sampling-based algorithm to compute paths in problems involving high-dimensional cost spaces. The proposed method combines the exploratory strength of RRTs, with the efficiency of stochastic optimization methods. It integrates an adaptive mechanism that helps to ensure a good performance for a large set of problems.

The notion of Minimal Work path has been proposed to quantify the quality of solution paths. By design, the proposed T-RRT algorithm computes paths that tend to satisfy such a quality property. A large set of experiments were performed to show the efficacy of the T-RRT planner.

Experimental results have shown that the planner is general enough to be applied, at least, to 6-dimensional spaces constrained by obstacles. Future work concerns the application of T-RRT to new classes of problems such as the integration of human-robot interaction constraints within path planning or the exploration of energy landscapes in computational biology problems. Extensions discussed in the previous section also need to be further explored for performance improvement. Furthermore, another direction is to incorporate in the planner other methods inspired by Monte Carlo optimization techniques, such as stochastic tunneling [28] or parallel tempering [29]. Finally, it would be interesting to test our approach on benchmark problems of the stochastic optimization community, since T-RRT could be used as a generic optimization tool and, in principle, applied to any metric cost space.

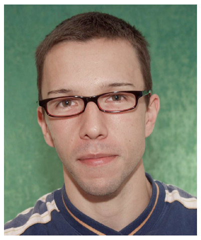
IX. Acknowledgments

This work has been partially supported by the the French National Research Agency under project "GlucoDesign", by the European Community under FP7-ICT project 216239 "DEXMART", and by the Spanish Ministry of Science and Innovation under project DPI2007-60858. Léonard Jaillet was supported by CSIC under JAE-Doc fellowship.

References

[1] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge: MIT Press, 2005.
[2] S. LaValle, *Planning Algorithms*. New York: Cambridge University Press, 2006.
[3] A. Stentz, "Optimal and efficient path planning for partially-known environments," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 3310–3317, 1994.
[4] A. Ettlin and H. Bleuler, "Rough-terrain robot motion planning based on obstacleness," *Proc. Int. Conf. on Control, Automation, Robotics and Vision*, pp. 1–6, 2006.
[5] ——, "Randomised rough-terrain robot motion planning," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 5798–5803, 2006.
[6] C. Urmson and R. Simmons, "Approaches for heuristically biasing RRT growth," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1178–1183, 2003.
[7] D. Ferguson and A. Stentz, "Anytime RRTs," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 5369 – 5375, 2006.
[8] R. Diankov and J. Kuffner, "Randomized statistical path planning," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1–6, 2007.
[9] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 995–1001, 2000.
[10] J. Lee, C. Pippin, and T. Balch, "Cost based planning with RRT in outdoor environments," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 684–689, 2008.
[11] L. Jaillet, J. Cortés, and T. Siméon, "Transition-based RRT for path planning in continuous cost spaces," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 2145–2150, 2008.
[12] J.-C. Latombe, *Robot Motion Planning*. Boston: Kluwer Academic Publishers, 1991.
[13] J. Barraquand and J.-C. Latombe, "A Monte-Carlo algorithm for path planning with many degrees of freedom," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 1712–1717, 1990.
[14] S. Sekhavat, P. Svestka, and M. Overmars, "Multi-level path planning for nonholonomic robots using semi-holonomic subsystems," *International Journal of Robotics Research*, vol. 17(8), pp. 840–857, 1998.
[15] R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *International Journal of Robotics Research*, vol. 26, pp. 845–863, 2007.
[16] J. Spall, *Introduction to Stochastic Search and Optimization*. New York: Wiley, 2003.
[17] S. Kirkpatrick, C. Gelatt, and M. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.
[18] M. Apaydin, A. Singh, D. Brutlag, and J.-C. Latombe, "Capturing molecular energy landscapes with probabilistic conformational roadmaps," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 932–939, 2001.
[19] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12(4), pp. 566–580, 1996.
[20] S. LaValle, "*Rapidly-Exploring Random Trees: A New Tool for Path Planning*," 1998, TR 98-11, CS Dept., Iowa State University.
[21] A. Yershova, L. Jaillet, T. Siméon, and S. LaValle, "Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 3867–3872, 2005.
[22] L. Jaillet, A. Yershova, S. LaValle, and T. Siméon, "Adaptive tuning of the sampling domain for dynamic-domain RRTs," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 4086–4091, 2005.
[23] T. Siméon, J.-P. Laumond, and F. Lamiraux, "Move3D: A generic platform for path planning," *Proc. IEEE Int. Symp. on Assembly & Task Planning*, pp. 25–30, 2001.
[24] M. Stilman, "Task constrained motion planning in robot joint space," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 3074–3081, 2007.
[25] J. Cortés, L. Jaillet, and T. Siméon, "Disassembly path planning for complex articulated objects," *IEEE Transactions on Robotics and Automation*, pp. 475–481, 2008.
[26] V. Ruiz de Angulo, J. Cortés, and T. Siméon, "BioCD: An efficient algorithm for self-collision and distance computation between highly articulated molecular models," in *Robotics: Science and Systems*, S. T. snd G. Sukhatme, S. Schaal, and O. Brock, Eds. Cambridge: MIT Press, 2005, pp. 6–11.
[27] D. Nieuwenhuisen and M. Overmars, "Useful cycles in probabilistic roadmap graphs," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 446– 452, 2004.
[28] K. Hamacher and W. Wenzel, "Scaling behavior of stochastic minimization algorithms in a perfect funnel landscape," *Phys. Rev. E*, vol. 59, no. 1, pp. 938–941, Jan 1999.
[29] D. J. Earl and M. W. Deem, "Parallel tempering: Theory, applications, and new perspectives," *Physical Chemistry Chemical Physics*, vol. 7, pp. 3910–3916, 2005.

Léonard Jaillet received the Eng. degree in mechanical engineering from the Institut Supérieur de Mécanique de Paris and the Ph.D. degree in robotics from the University of Toulouse, France, in 2001 and 2005, respectively. Since 2008, he is Postdoctoral Fellow at the Institut de Robótica i Informàtica Industrial, Spanish National Research Concil, Barcelona. His research interest includes motion planning for complex robotic systems and molecular simulations for structural biology.



Juan Cortés received the engineering degree in control and robotics from the Universidad de Zaragoza (Spain) in 2000. In 2003, he received the Ph.D. degree in robotics from the Institut National Polytechnique de Toulouse (France). From 2004, he is CNRS researcher at LAAS (Toulouse, France). His research interest is focused around the development of algorithms for computing and analyzing the motion of complex systems in robotics and structural biology. He co-chairs the IEEE-RAS TC on Algorithms for Planning and Control of Robot Motion.

Thierry Siméon received the engineering degree in computer science from INSA, Toulouse, France and the Ph.D degree in robotics from the University of Toulouse, in 1985 and 1989, respectively. Since 1990, he has been with LAAS-CNRS, Toulouse, France. His primary research interest includes robot motion planning and applications to structural bioinformatics. He is co-author of more than 100 papers and was a co-founder of the LAAS spin-off Kineo CAM. He is currently an Associate Editor of the IEEE Transactions on Robotics.