HG-RRT*: Human-Guided Optimal Random Trees for Motion Planning

Néstor García, Raúl Suárez and Jan Rosell

***Abstract*—The paper deals with the problem of designing an RRT*-based planning algorithm that allows the user to guide the tree growth in a simple and transparent way. The key idea of the proposal is to create a planning algorithm, called HG-RRT*, that minimizes an optimization function over the configuration space where a state cost function is established. This state cost is defined as the combination of several potential fields. Each of these potential fields will attract the solution path or move it away from certain areas. The planning algorithm will try to minimize the path length, the motion effort and the variations of the cost along the path. The paper presents a description of the proposed approach as well as simulation results from a conceptual and an application example, including a thorough comparison with the TRRT planning algorithm.**

I. Introduction

A. Previous Works

Some of the best approaches to motion planning for robotic systems with high number of degrees of freedom are those based on optimization techniques and those based on sampling. Optimization approaches, like CHOMP [1] or TrajOpt [2], incorporate collision avoidance into trajectory optimization, i.e. an optimization method starts from a trajectory (or several [3]) that contains collisions and perhaps violates constraints and tries to converge to a high-quality trajectory satisfying constraints. The differences between these optimization approaches lie in the numerical optimization method used and in the method of checking for collisions and penalizing them. Sampling-based approaches, like RRT [4] or KPIECE [5], on the other hand, put the emphasis in the exploration of the free configuration space in order to capture its connectivity relevant to the query to be solved. To improve efficiency, different ways to bias the sampling have been proposed, like the one based on features present in the underlying workspace [6]. Also, in order to seek for optimal solutions some variants have been proposed, like RRT* [7] or TRRT [8].

The basic idea of RRTs is to build a tree of feasible motions, rooted at the initial configuration, by iteratively sampling the configuration space to get a random configuration ($\boldsymbol{q}_{\text{rand}}$), searching the node of the tree nearest to it ($\boldsymbol{q}_{\text{near}}$), and moving a small amount from $\boldsymbol{q}_{\text{near}}$, towards $\boldsymbol{q}_{\text{rand}}$. If the generated motion is collision-free then the reached configuration ($\boldsymbol{q}_{\text{new}}$) is added as a node of the tree and the path connecting $\boldsymbol{q}_{\text{near}}$ and $\boldsymbol{q}_{\text{new}}$ as an edge. RRT*, on the other hand, proceeds as follows. Once $\boldsymbol{q}_{\text{new}}$ has been computed as in the RRT case, it is not directly connected to $\boldsymbol{q}_{\text{near}}$ but to the node (among a given set of neighbors) that minimizes the cost to reach $\boldsymbol{q}_{\text{new}}$ according to a given cost function. Afterwards, RRT* checks whether each neighbor node can be reached, through $\boldsymbol{q}_{\text{new}}$, with a cost smaller than its current one and, if so, rewires the edges of the tree. Thanks to this rewiring process the solution keeps improving while the number of samples increases. RRT* has usually been used to optimize the path length or the path clearance, although other alternative costs have been proposed [9], [10]. In order to improve efficiency, the combination of RRT* and potential fields has been recently proposed [11]. In this approach, the position of $\boldsymbol{q}_{\text{rand}}$ is modified by moving it a small amount in the direction defined by the gradient of a potential function, directing it away from obstacles and towards the goal. Similar combinations of sampling-based approaches and potential fields were explored in the past, e.g. [12] and [13]. A potential field codifying a given cost-map is also used with an optimization perspective in the TRRT approach. This method computes low-cost paths over a configuration-space cost-map by combining an RRT with an stochastic optimization method that modulates the tree growth by accepting or rejecting new potential nodes (mechanical work is used as optimization criterion, i.e. only positive variations of the cost function along the path are considered to compute the cost of the path). Following this approach, the Environment-Guided RRT (EG-RRT [14]) exploits the cost-guided exploration of TRRTs to minimize the probability of collisions when uncertainty in control and sensing is present.

B. Problem Statement and Solution Overview

The aim of this work is to develop a sampling-based planning algorithm that allows the user to easily guide the path towards some preferred regions while avoiding some others. The proposal uses an RRT* with an optimization cost function based on a combination of several potential fields that modulate the tree growth. Repulsive potential fields are set to define regions to be avoided, whilst attractive potential fields are set to define the preferred ones. The potential fields are generated by points or segments specified by the user in the workspace. The sampling-based nature of the RRT* planning algorithm allows to easily find collision-free path in cluttered

The authors are with the Institute of Industrial and Control Engineering (IOC), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, raul.suarez@upc.edu. This work was partially supported by the Spanish Government through the projects DPI2011-22471, DPI2013-40882-P and DPI2014-57757-R.

978-1-4673-7929-8/15/$31.00

environments for robots with many degrees of freedom, while its optimization nature is exploited to guide the path by using the user-defined cost function. The minimization of this cost function steers the robot to safe areas while it is moved away from the obstacles and unsafe areas. The cost function is defined so that the planning algorithm takes into account the path length, the average value and the variations of the cost along the path.

II. The HG-RRT* Planning Algorithm

A. Cost Function

RRT*-based planners use an optimization cost function to evaluate and compare different paths. In RRTs, the initial and the goal configurations are connected by piece-wise paths composed of a set of edges, called motions. When there are only geometric constraints, the paths can be piece-wise linear and the motions straight-line segments. Each configuration of the robotic system is considered as a different state for the planning algorithm.

The state cost used in this work is defined as the combination of potentials surfaces. Each potential can be repulsive or attractive, and it is created by a point or a segment in the workspace (although it can easily be extended to a free-form surface). The cost of a state $\boldsymbol{s}$ is computed as:

$$c(\boldsymbol{s}) = \sum_i \max(-\lambda_i, 0) + \lambda_i e^{-\alpha_i d_i(\boldsymbol{s})^2} \in (0, \textstyle\sum_i |\lambda_i|] \quad (1)$$

with $\lambda_i \in \mathbb{R}$, $\alpha_i \geqslant 0$ and $d_i(\boldsymbol{s})$ being the distance in the workspace between the reference point of the robot at the state $\boldsymbol{s}$ and the i-th point or segment that generates the potential field. λ_i is the magnitude parameter that defines how repulsive or attractive the i-th element is. Note that for $\lambda_i > 0$ the i-th element is repulsive and for $\lambda_i < 0$ it is attractive. Nevertheless, $c(\boldsymbol{s})$ is strictly positive. α_i is the diffusion parameter and determines how concentrated the potential of the i-th element is.

Let $\boldsymbol{s}_s$ and $\boldsymbol{s}_g$ be, respectively, the start and the goal states, $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ be two arbitrary states and $d(\boldsymbol{s}_1, \boldsymbol{s}_2)$ the Euclidean distance in the state space between them. Then, the cost of the rectilinear motion connecting the two states $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ is defined as the linear combination of three costs:

$$c(\boldsymbol{s}_1, \boldsymbol{s}_2) = \frac{k_\mathrm{P}\, c_\mathrm{P}(\boldsymbol{s}_1, \boldsymbol{s}_2) + k_\mathrm{I}\, c_\mathrm{I}(\boldsymbol{s}_1, \boldsymbol{s}_2) + k_\mathrm{D}\, c_\mathrm{D}(\boldsymbol{s}_1, \boldsymbol{s}_2)}{k_\mathrm{P} + k_\mathrm{I} + k_\mathrm{D}} \quad (2)$$

where

$$c_\mathrm{P}(\boldsymbol{s}_1, \boldsymbol{s}_2) = d(\boldsymbol{s}_\mathrm{s}, \boldsymbol{s}_\mathrm{g})^{-1} \int_{\boldsymbol{s}_1}^{\boldsymbol{s}_2} d\boldsymbol{s} = \frac{d(\boldsymbol{s}_1, \boldsymbol{s}_2)}{d(\boldsymbol{s}_\mathrm{s}, \boldsymbol{s}_\mathrm{g})} \quad (3)$$

$$c_\mathrm{I}(\boldsymbol{s}_1, \boldsymbol{s}_2) = \left(\frac{c(\boldsymbol{s}_\mathrm{s}) + c(\boldsymbol{s}_\mathrm{g})}{2}\, d(\boldsymbol{s}_s, \boldsymbol{s}_g)\right)^{-1} \int_{\boldsymbol{s}_1}^{\boldsymbol{s}_2} c(\boldsymbol{s})\, d\boldsymbol{s} \quad (4)$$

$$c_\mathrm{D}(\boldsymbol{s}_1, \boldsymbol{s}_2) = |c(\boldsymbol{s}_\mathrm{g}) - c(\boldsymbol{s}_\mathrm{s})|^{-1} \int_{\boldsymbol{s}_1}^{\boldsymbol{s}_2} \left|\frac{\mathrm{d}\, c(\boldsymbol{s})}{\mathrm{d}\boldsymbol{s}}\right| \mathrm{d}\boldsymbol{s} \quad (5)$$

with k_P, k_I, $k_\mathrm{D} \geq 0$ and $k_\mathrm{P} + k_\mathrm{I} + k_\mathrm{D} > 0$.

The motion cost presented in this work represents the path integral of a state cost function defined over the state space. $c_\mathrm{P}(\boldsymbol{s}_1, \boldsymbol{s}_2)$ measures the length of the motion, $c_\mathrm{I}(\boldsymbol{s}_1, \boldsymbol{s}_2)$ measures the motion effort, computed as the product of the state cost average and the motion length, and $c_\mathrm{D}(\boldsymbol{s}_1, \boldsymbol{s}_2)$ measures the variations of the state cost function along the path. Note that $c(\boldsymbol{s}_\mathrm{s}) + c(\boldsymbol{s}_\mathrm{g}) \neq 0$ and $|c(\boldsymbol{s}_\mathrm{g}) - c(\boldsymbol{s}_\mathrm{s})| \neq 0$ is required to have bounded values of the costs. The former is satisfied by definition and the latter is easily accomplished by always setting $\boldsymbol{s}_g$ as an attractive point.

The planning objective is the minimization of the total cost $c(\mathcal{P})$ of a path $\mathcal{P}$ defined in the HG-RRT* by a sequence of n consecutive states $\boldsymbol{s}_i$ connected by rectilinear motions. $c(\mathcal{P})$ is computed as:

$$c(\mathcal{P}) = \sum_{i=1}^{n-1} c(\boldsymbol{s}_i, \boldsymbol{s}_{i+1}) \quad (6)$$

Another optimization objective, given a state cost, is proposed by the TRRT algorithm [8] that we will use later for comparison purposes. It uses the mechanical work as optimization criterion, wich is computed as follows,

$$MW(\mathcal{P}) = \int_{\mathcal{P}} \left[\max\left(\frac{\mathrm{d}\, c(\boldsymbol{s})}{\mathrm{d}\, \boldsymbol{s}}, 0\right) + \epsilon\right] \mathrm{d}\boldsymbol{s} \quad (7)$$

where ϵ is assumed to be a very small value compared to the state costs and it favors shortest paths of equal mechanical work. This optimization objective takes only into account the positive variations of the state cost function along the path. However, the mechanical work is asymmetric (unlike the optimization objective presented here). The use of the mechanical work as an optimization criterion in the RRT* leads to motion cost recomputation when some rewiring of the tree is needed and so the performance may be degraded.

B. The algorithm

The proposed planning procedure is described in Algorithm 1, where the following functions are used:

- Sample($\mathcal{C}$, $\boldsymbol{q}_{goal}$, P_{goal}): Returns a random configuration of the configuration space $\mathcal{C}$ with probability $1 - P_{goal}$, and returns $\boldsymbol{q}_{goal}$ with probability P_{goal}.
- k-Near($V, \boldsymbol{q}$): Returns the k nearest neighbors of $\boldsymbol{q}$ from the set V of tree nodes (the value of k depends on the dimension of the configuration space and on the number of tree vertices [7]).
- Path($\boldsymbol{q}$): Returns a piece-wise rectilinear path, composed of tree edges, that connects the root node $\boldsymbol{q}_{\text{init}}$ to node $\boldsymbol{q}$. The path is computed by backtracking from $\boldsymbol{q}$ following the parent relationship in the tree.
- CollisionFree($\boldsymbol{q}_1, \boldsymbol{q}_2$): Returns true if the rectilinear edge in $\mathcal{C}$ connecting $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ is collision-free, and false otherwise.
- Nearest($\boldsymbol{q}$): Returns the closest node to configuration $\boldsymbol{q}$ from the tree nodes.
- Cost($\boldsymbol{q}$): Returns the state cost of the configuration $\boldsymbol{q}$ using Eq. (2).
- EdgeCost($\boldsymbol{q}_1, \boldsymbol{q}_2$): Returns the cost of the rectilinear motion from $\boldsymbol{q}_1$ to $\boldsymbol{q}_2$ using Eq. (6).

Algorithm 1 HG-RRT*

Input: Configurations $\boldsymbol{q}_{\text{init}}, \boldsymbol{q}_{\text{goal}} \in \mathcal{C}$
Probability P_{goal}
Advance step ϵ
Output: A path from $\boldsymbol{q}_{\text{init}}$ to $\boldsymbol{q}_{\text{goal}}$

$V \leftarrow \{\boldsymbol{q}_{\text{init}}\}; E \leftarrow \emptyset$;
for $i \leftarrow 0$ **to** n **do**
  $\boldsymbol{q}_{\text{rand}} \leftarrow$ Sample($\mathcal{C}$, $\boldsymbol{q}_{goal}$, P_{goal})
  $\boldsymbol{q}_{\text{nearest}} \leftarrow$ Nearest($\boldsymbol{q}_{\text{rand}}$)
  $\boldsymbol{q}_{\text{new}} \leftarrow$ Steer($\boldsymbol{q}_{\text{nearest}}, \boldsymbol{q}_{\text{rand}}, \epsilon$)
  if CollisionFree($\boldsymbol{q}_{\text{nearest}}, \boldsymbol{q}_{\text{new}}$) **then**
    $Q_{\text{near}} \leftarrow k$-Near($V, \boldsymbol{q}_{\text{new}}$)
    $V \leftarrow V \cup \{\boldsymbol{q}_{\text{new}}\}$
    $\boldsymbol{q}_{\text{min}} \leftarrow \boldsymbol{q}_{\text{nearest}}$
    $c_{\text{min}} \leftarrow$ Cost($\boldsymbol{q}_{\text{nearest}}$) + EdgeCost($\boldsymbol{q}_{\text{nearest}}, \boldsymbol{q}_{\text{new}}, \mathbf{w}, \boldsymbol{U}$)
    for all $\boldsymbol{q} \in Q_{\text{near}}$ **do**
      if CollisionFree($\boldsymbol{q}, \boldsymbol{q}_{\text{new}}$) $\wedge$ Cost($\boldsymbol{q}$) + EdgeCost($\boldsymbol{q}, \boldsymbol{q}_{\text{new}}, \mathbf{w}, \boldsymbol{U}$) $< c_{\text{min}}$ **then**
        $\boldsymbol{q}_{\text{min}} \leftarrow \boldsymbol{q}$
        $c_{\text{min}} \leftarrow$ Cost($\boldsymbol{q}$) + EdgeCost($\boldsymbol{q}, \boldsymbol{q}_{\text{new}}, \mathbf{w}, \boldsymbol{U}$)
      end if
    end for
    $E \leftarrow E \cup \{\boldsymbol{q}_{\text{min}}, \boldsymbol{q}_{\text{new}}\}$ //Connect along a minimum-cost path
    for all $\boldsymbol{q} \in Q_{\text{near}}$ **do**
      if CollisionFree($\boldsymbol{q}_{\text{new}}, \boldsymbol{q}$) $\wedge$ Cost($\boldsymbol{q}_{new}$) + EdgeCost($\boldsymbol{q}_{\text{new}}, \boldsymbol{q}, \mathbf{w}, \boldsymbol{U}$) < Cost($\boldsymbol{q}$) **then**
        $E \leftarrow (E \backslash \{\text{Parent}(\boldsymbol{q}), \boldsymbol{q}\}) \cup \{\boldsymbol{q}_{\text{new}}, \boldsymbol{q}\}$ //Rewire
      end if
    end for
    if $\boldsymbol{q}_{\text{new}} = \boldsymbol{q}_{\text{goal}}$ **then**
      return Path($\boldsymbol{q}_{\text{goal}}$)
    end if
  end if
end for
return $\emptyset$

It should be remarked that the HG-RRT* algorithm is based on the RRT* algorithm but the optimization cost function has been modified (using Eq. (2) and Eq. (6)). Instead of minimizing only the path length, the new optimization cost function allows to minimize the path length, the motion effort and the variations of the state cost function along the path.

III. Approach Validation

A. Implementation issues

The proposal has been implemented within The Kautham Project [15], a motion planning and simulation environment developed at the Institute of Industrial and Control Engineering (IOC-UPC) for teaching and research. The core of the planners provided belong to the Open Motion Planning Library (OMPL) [16], which codes a wide set of the state-of-the-art sampling-based motion planning algorithms at an abstract level, i.e. without including issues related to robot modelling, collision-check or visualization.

The cost function proposed has been coded as a class derived from the OMPL OptimizationObjective class and the planning algorithm as a derived class from the OMPL RRT* class.

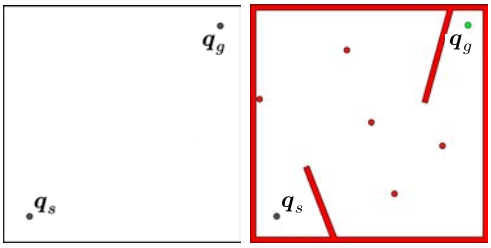

Fig. 1. Conceptual example. A 2-dof mobile robot must go from the start configuration $\boldsymbol{q}_s$ to the goal configuration $\boldsymbol{q}_g$ in a collision-free scenario (left). $\boldsymbol{q}_g$ has an attractive potential field and the points and segments red painted have a repulsive one (right).

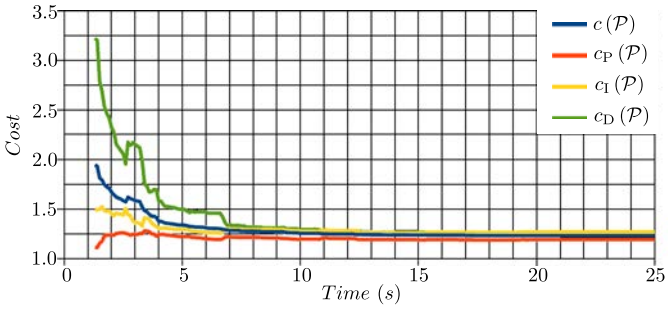

Fig. 2. Path cost $c(\mathcal{P})$ and its components ($c_{\text{P}}(\mathcal{P})$, $c_{\text{I}}(\mathcal{P})$ and $c_{\text{D}}(\mathcal{P})$) as a function of the planning time (in seconds) resulting from the average of 10 executions when solving the conceptual example with the HG-RRT* algorithm and $(k_{\text{P}}, k_{\text{I}}, k_{\text{D}}) = (1, 1, 1)$.

B. Conceptual Example

The proposed planning procedure was used to plan the motions on a 2D scenario (see Fig. 1). It consists of an obstacle-free workspace with the start configuration $\boldsymbol{q}_s$ at the bottom-left corner and the goal configuration $\boldsymbol{q}_g$ at the top-right corner. A potential field is established by setting several repulsive points and segments. Moreover, the limits of the workspace and the goal configuration act, respectively, as repulsive walls and as an attractive point.

Fig. 2 shows how the path cost $c(\mathcal{P})$ decreases as the HG-RRT* runs for more time until reaching a stable value. It can be seen that after 20 seconds the cost has already converged to its final value. In Fig. 3 (top) several solution paths obtained with a different planning time are shown. In Fig. 3 (bottom) the evolution of the state cost along the path for different planning time limits is shown. Note that cost peaks are decreased as the planning time increases. Nevertheless, the length of the solution path is not extremely increased. The query is solved for 20 seconds and the solution found is shown in Fig. 4. The path obtained minimizes the combination of the three costs $c_{\text{P}}(\mathcal{P})$, $c_{\text{I}}(\mathcal{P})$ and $c_{\text{D}}(\mathcal{P})$, avoiding large variations of the cost along the path by trying to go directly to the goal configuration without climbing any mountain.

To illustrate the proposed approach, the resulting solution path when only one component of the motion cost is minimized is also shown:

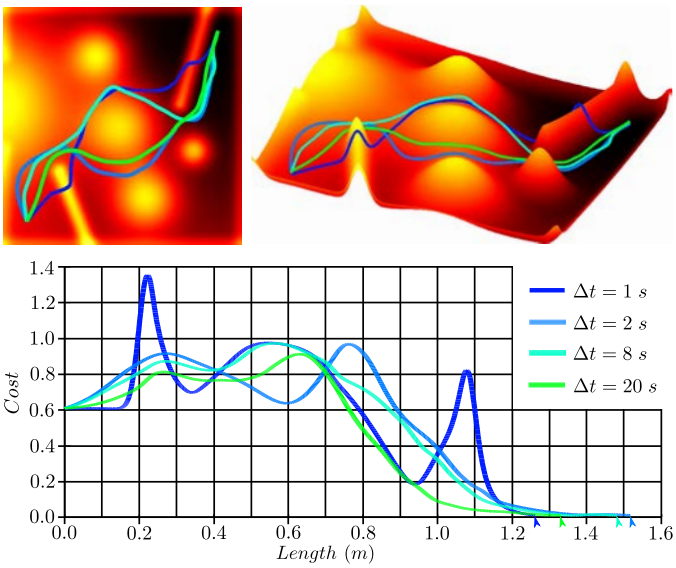

Fig. 3. Paths obtained after solving the conceptual example with the HG-RRT* algorithm for $(k_P,\ k_I,\ k_D) = (1,\ 1,\ 1)$ and with different limits of planning time (top) and state cost evolution along the path for the different cases (bottom).

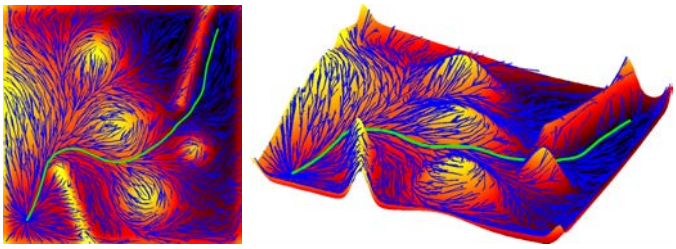

Fig. 4. Tree and solution path obtained after solving the conceptual example with the HG-RRT* algorithm for 20 seconds with $(k_P,\ k_I,\ k_D) = (1,\ 1,\ 1)$.

- Fig. 5 shows the path that minimizes $c_P(\mathcal{P})$. Since the distance is being minimized the solution is the rectilinear motion connecting $\boldsymbol{q}_s$ and $\boldsymbol{q}_g$ no matter the mountains that are climbed along the path. It can be seen that the tree edges are radial motions starting at $\boldsymbol{q}_s$.
- Fig. 6 shows the path that minimizes $c_I(\mathcal{P})$. In this case only the narrow mountains are climbed instead of being surrounded because it would imply more effort. Only the wide mountains are surrounded.
- Fig. 7 shows the path that minimizes $c_D(\mathcal{P})$. Note that the number of ascents and descents is reduced and the tree edges try to follow the contour lines. In this case no mountain is climbed even if this means a longer path.

Fig. 8 shows the evolution of the state cost along the path obtained with different values of k_P, k_I, and k_D. Note that the path minimizing $c_P(\mathcal{P})$ has the shortest length but the higher cost peaks, the path minimizing $c_I(\mathcal{P})$ has lower peaks but it is a little bit longer, the path minimizing $c_D(\mathcal{P})$ has the lowest peaks but it is also the longest, and the path minimizing $c(\mathcal{P})$ presents a balanced solution as a combination of the other three.

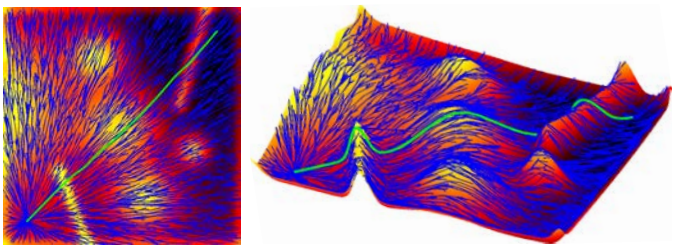

Fig. 5. Tree and solution path obtained after solving the conceptual example with the HG-RRT* algorithm for 20 seconds with $(k_P,\ k_I,\ k_D) = (1,\ 0,\ 0)$.

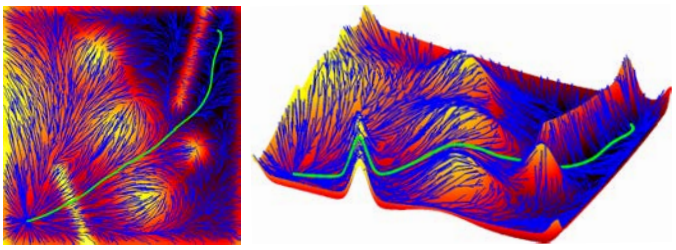

Fig. 6. Tree and solution path obtained after solving the conceptual example with the HG-RRT* algorithm for 20 seconds with $(k_P,\ k_I,\ k_D) = (0,\ 1,\ 0)$.

Table I shows the average results obtained after 100 executions for this example using the HG-RRT* algorithm and the standard TRRT when running in a 2.13-GHz Intel Core 2, 4-GB RAM PC. The conceptual example has been solved with the HG-RRT* (for $(k_P,\ k_I,\ k_D) = (1,\ 1,\ 1)$) and the TRRT algorithms. As stated above, while the HG-RRT* algorithm uses the optimization function presented in this work, the TRRT algorithm minimizes the mechanical work of the path. A limit of 20 seconds was imposed to the executions. If the planner cannot find a solution with this constraint the planner run is considered as a failure. The table includes: the success rate, the time needed to find a solution, the solution path length, the number of tree nodes, the cost of the solution path calculated with the mechanical work criterion (cost MW), the cost of the solution path calculated with the method proposed in this work (cost PID), the average state cost value along the path and the maximum state cost value along the path.

From the experimental results it can be stated that the

TABLE I
AVERAGE RESULTS OF THE MOTION PLANNING FOR 100 EXECUTIONS.

planner	HG-RRT*	TRRT
success rate (%)	100	100
used time (s)	20	0.774
solution length (m)	133.074	188.785
tree nodes	8866.089	658.358
path cost MW	0.348	0.594
path cost PID	1.749	2.411
$\bar{c}$	0.594	0.801
c_{max}	0.911	0.918

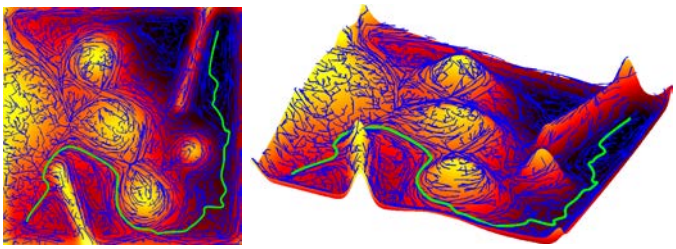

Fig. 7. Tree and solution path obtained after solving the conceptual example with the HG-RRT* algorithm for 20 seconds with $(k_{\mathrm{P}},\ k_{\mathrm{I}},\ k_{\mathrm{D}}) = (0,\ 0,\ 1)$.

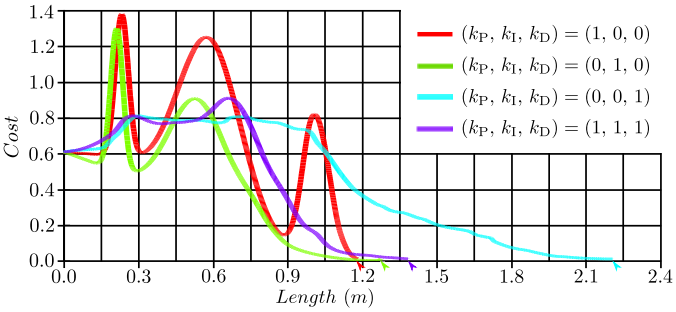

Fig. 8. State cost as a function of the resulting path length with different values of k_{P}, k_{I}, and k_{D} when solving the conceptual example with the HG-RRT* algorithm.

HG-RRT* algorithm produces shorter paths and better results despite whether the path cost is computed with the mechanical work method or with the one presented in this paper. It can also be seen that the average state cost along the path is lower in the case of the HG-RRT*. The case using TRRT planner has a lower planning time since the HG-RRT* planner never terminates and it keeps trying to find a more optimal solution within the allowed time.

Fig. 9 shows a path obtained when solving the example with the TRRT and using the mechanical work optimization function (Eq. 7). The shown path has a cost similar to the obtained average value of 100 executions. It can be seen that the whole configuration space has not been explored due to the TRRT tree growth modulation. No mountain is climbed but as shown in Table I with the HG-RRT* paths with lower mechanical work can be obtained.

The proposed approach and TRRT executions have a huge difference in terms of timing and so the direct comparison of the HG-RRT* planner against TRRT is not fair. Unlike the RRT*, the TRRT cannot be forced to run for a fixed amount of time. Nevertheless, the TRRT transition test can be tuned to be more strict and then obtain better solution paths, at a cost of increasing the planning time. Hence, by tunning the transition test, the TRRT planning time can be increased. In this way, the same conceptual example was solved again but with different values of the transition test parameters. Fig. 10 shows the cost of the solution path obtained for the two planners with different planning times. Although TRRT is able to find a solution with less than one second, HG-RRT* quickly obtains

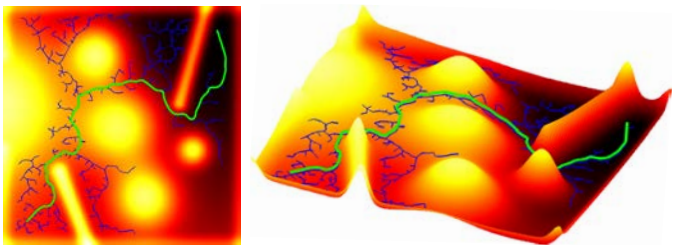

Fig. 9. Tree and solution path obtained after solving the conceptual example with the TRRT algorithm.

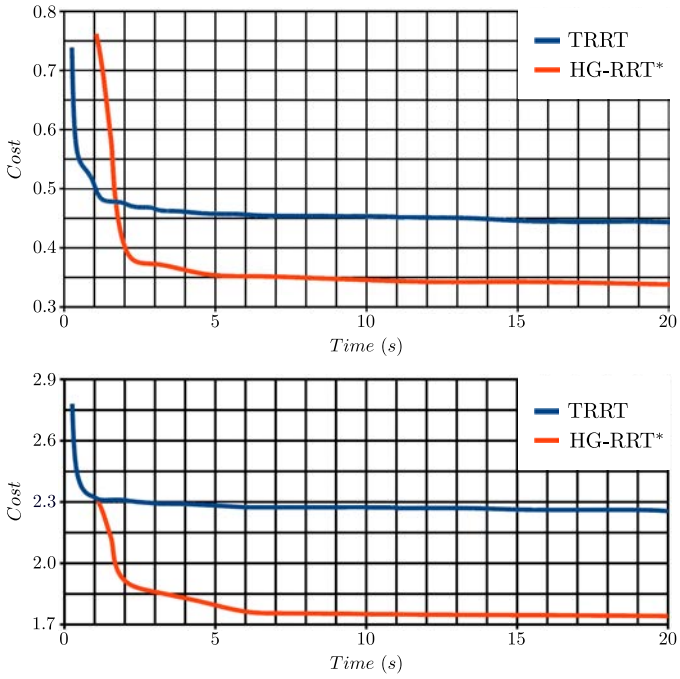

Fig. 10. Path cost $c(\mathcal{P})$, computed with the mechanical work (top) and the PID (bottom) methods, as a function of the planning time resulting from the average of 100 executions when solving the conceptual example with the TRRT and the HG-RRT* algorithms, the latter with $(k_{\mathrm{P}},\ k_{\mathrm{I}},\ k_{\mathrm{D}}) = (1,\ 1,\ 1)$.

better results for the same used time no matter whether the cost was measured with the mechanical work criterion or with that proposed in this paper.

C. Application Example

The proposed planning procedure was used to plan the motions of a quadrotor while it performs aerial inspection of the chimneys of a power plant (see Fig. 11). The quadrotor has been modelled as a sphere for collision checking and its dynamics has not been taken into account. Then, if necessary, dynamics of the quadrotor, supposed differentially flat, can be considered by selecting waypoints from the optimal path and jointly optimizing a set of polynomials through them to obtain a minimum-snap path, like in [17].

The state cost function has been constructed making each chimney repulsive with a different diffusive coefficient α_i depending on the chimney radius. A set of horizontal attractive

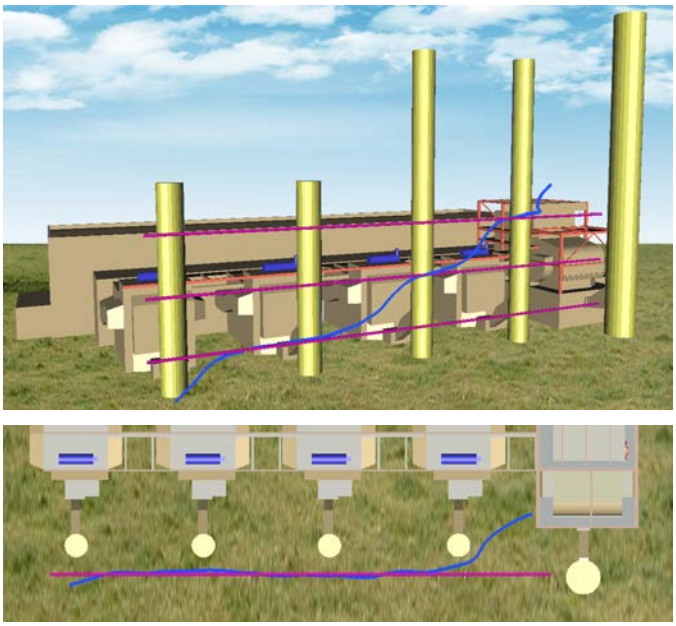

Fig. 11. Application example. A quadrotor has to do an aerial inspection of the chimneys of a power plant. The start configuration $\boldsymbol{q}_s$ has been set at ground level and the goal configuration $\boldsymbol{q}_g$ has an attractive potential. The chimneys act as repulsive segments while the lines magenta painted reprent attractive lines that steer the path. The figure shows the HG-RRT* solution path after 100 seconds of motion planning with $(k_{\mathrm{P}},\, k_{\mathrm{I}},\, k_{\mathrm{D}}) = (1,\, 1,\, 1)$.

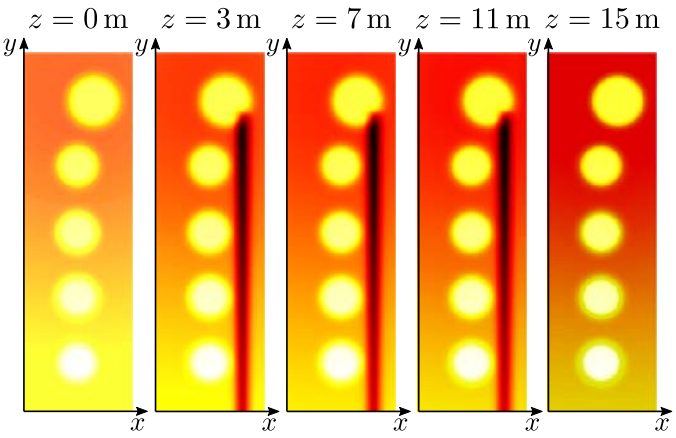

Fig. 12. State cost function of the application example computed at different heights (darker colors represent lower state cost values).

lines has been added to the scenario to steer the solution path. As done with the conceptual example, the goal configuration acts as an attractive point. Fig. 12 shows the potential surfaces for different heights where darker colors represent lower state cost values. The start configuration is at ground level ($z = 0\,\mathrm{m}$) while the goal configuration is at a height of $z = 15\,\mathrm{m}$.

Fig. 13 shows how the path cost $c(\mathcal{P})$ keeps decreasing as the query is solved for more time. It is considered that only after 100 seconds of motion planning the cost has converged to its final value. Then, the solution path obtained after this time is shown in Fig. 11. It can be seen that the path tries

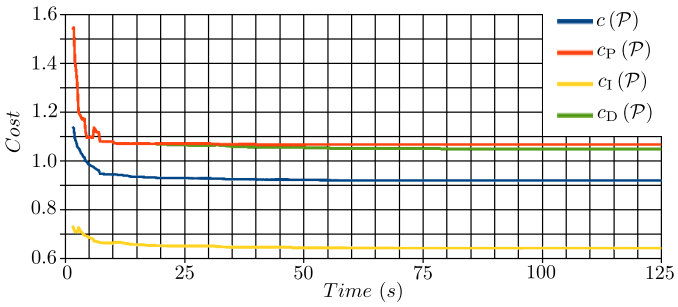

Fig. 13. Path cost $c(\mathcal{P})$ and its components ($c_{\mathrm{P}}(\mathcal{P})$, $c_{\mathrm{I}}(\mathcal{P})$ and $c_{\mathrm{D}}(\mathcal{P})$) as a function of the planning time (in seconds) resulting from the average of 10 executions when solving the application example with the HG-RRT* algorithm and $(k_{\mathrm{P}},\, k_{\mathrm{I}},\, k_{\mathrm{D}}) = (1,\, 1,\, 1)$.

TABLE II
AVERAGE RESULTS OF THE MOTION PLANNING FOR 100 EXECUTIONS.

planner	HG-RRT*	TRRT
success rate (%)	100	100
used time (s)	100	6.562
solution length (m)	37.472	59.037
tree nodes	8759.201	6597.298
path cost MW	0.000397	0.00283
path cost PID	0.923	4.061
$\bar{c}$	0.333	0.607
$c_{\max}$	0.696	1.024

to follow the guiding lines as long as it does not increase too much the path length. In addition, the solution obtained avoids getting close to the chimneys.

Table II shows the average results obtained after 100 executions for the quadrotor example using the HG-RRT* algorithm and the standard TRRT. In this case the time limit has been set to 100 seconds.

It can be noted that using the HG-RRT* algorithm we get shorter solutions and with a lower average state cost value than with the TRRT planner. HG-RRT* executions have the lowest path cost values (with the mechanical work criterion and also with our method). They have more tree nodes since the HG-RRT* planner consumes all the available planning time.

As it was done previously, the same example has been run with different parameter values of the TRRT transition test to get executions that require different planning times. Fig. 14 shows the obtained results. It can be seen that, again, the HG-RRT* obtains solution paths with lower cost. Note that the TRRT path cost has not converged to a constant value for the maximum planning time considered.

IV. Conclusions

This paper has proposed original work dealing with the design of a planning algorithm that, in a simple way, lets the user to guide the motion planning towards some desirable areas while the tree growth is moved away from some preferably avoidable zones of the configuration space. A state cost

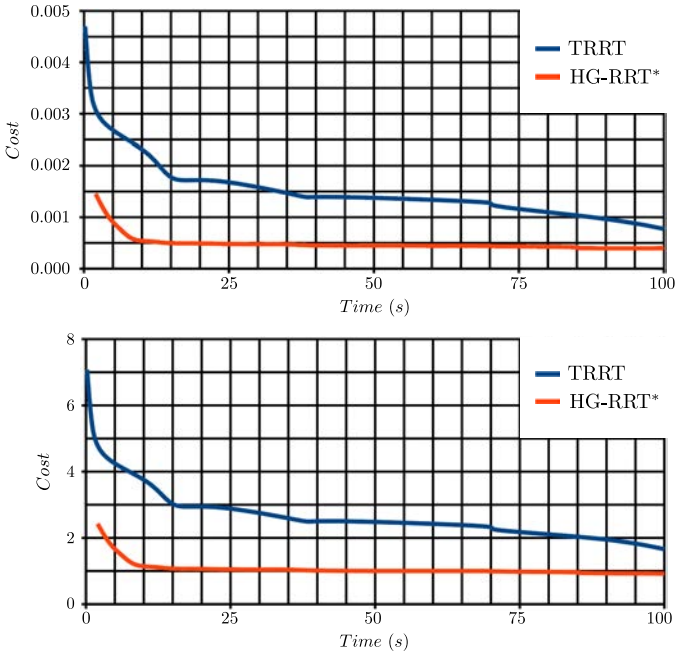

Fig. 14. Path cost $c(\mathcal{P})$, computed with the mechanical work (top) and the PID (bottom) methods, as a function of the planning time (in seconds) resulting from the average of 100 executions when solving the application example with the TRRT and the HG-RRT* algorithms and $(k_{\mathrm{P}}, k_{\mathrm{I}}, k_{\mathrm{D}}) = (1, 1, 1)$.

function has been defined over the configuration space as the combination of several potential fields. The user can create as many potential fields, repulsive or attractive, as he wants. A RRT*-based planner, called HG-RRT*, has been developed. This planner algorithm minimizes the path length, the average state cost along the path and the variations of the state cost function along the path. The proposed approach has been implemented and a conceptual and an application example were presented to illustrate the proposed ideas. It has been also compared with the TRRT planning algorithm, showing better results.

REFERENCES

[1] N. Ratliff, M. Zucker, J. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2009, pp. 489–494.

[2] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, August 2014.

[3] J. Pan, Z. Chen, and P. Abbeel, "Predicting initialization effectiveness for trajectory optimization," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 5183–5190.

[4] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An efficient approach to single-query path planning," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 995–1001.

[5] I. Sucan and L. Kavraki, "A sampling-based tree planner for systems with complex dynamics," *IEEE Trans. on Robotics*, vol. 28, no. 1, pp. 116–131, 2012.

[6] M. Zucker, J. Kuffner, and J. Bagnell, "Adaptive workspace biasing for sampling-based planners," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2008, pp. 3757–3762.

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

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

[9] J. Rosell and R. Suárez, "cRRT*: Planning loosely-coupled motions for multiple mobile robots," in *19th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2014.

[10] J. Rosell and R. Suarez, "Using hand synergies as an optimality criterion for planning human-like motions for mechanical hands," in *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, Nov 2014, pp. 232–237.

[11] A. Qureshi, K. Iqbal, S. Qamar, F. Islam, Y. Ayaz, and N. Muhammad, "Potential guided directional-RRT* for accelerated motion planning in cluttered environments," in *Mechatronics and Automation (ICMA), 2013 IEEE International Conference on*, Aug 2013, pp. 519–524.

[12] D. Aarno, D. Kragic, and H. Christensen, "Artificial potential biased probabilistic roadmap method," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 1, April 2004, pp. 461–466 Vol.1.

[13] J. Rosell and P. Iñiguez, "Path planning using harmonic functions and probabilistic cell decomposition," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, April 2005, pp. 1803–1808.

[14] L. Jaillet, J. Hoffman, J. van den Berg, P. Abbeel, J. Porta, and K. Goldberg, "EG-RRT: Environment-guided random trees for kinodynamic motion planning with uncertainty and obstacles," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, Sept 2011, pp. 2646–2652.

[15] J. Rosell, A. Pérez, A. Aliakbar, Muhayyuddin, L. Palomo, and N. García, "The Kautham Project: A teaching and research tool for robot motion planning," in *IEEE Int. Conf. on Emerging Technologies and Factory Automation, ETFA'14*, 2014.

[16] I. A. Suçan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012.

[17] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," *Proceedings of the International Symposium on Robotics Research (ISRR)*, 2013.