Robotica (2013) volume 31, pp. 861–874. © Cambridge University Press 2013
doi:10.1017/S0263574713000143

Safety-oriented path planning for articulated robots

Bakir Lacevic†,* and Paolo Rocco‡

†*University of Sarajevo, Faculty of Electrical Engineering, Sarajevo, Bosnia and Herzegovina*
‡*Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Milano, Italy*

(Accepted January 23, 2013. First published online: February 21, 2013)

SUMMARY

This work presents an approach to motion planning for robotic manipulators that aims at improving path quality in terms of safety. Safety is explicitly assessed using the quantity called *danger field*. The measure of safety can easily be embedded into a heuristic function that guides the exploration of the free configuration space. As a result, the resulting path is likely to have substantially higher safety margin when compared to one obtained by regular planning algorithms. To this end, four planning algorithms have been proposed. The first planner is based on volume trees comprised of bubbles of free configuration space, while the remaining ones represent modifications of classical sampling-based algorithms. Several numerical case studies are carried out to validate and compare the performance of the presented algorithms with respect to classical planners. The results indicate significantly lower danger metric for paths obtained by safety-oriented planners even with some decrease in running time.

KEYWORDS: Motion Planning; Path Planning; Redundant Manipulators; Safety.

1. Introduction

Ensuring safety for humans is certainly the highest priority in all applications where humans and robots may interact, particularly in those where robots can cause harm to humans. In industrial environment, robotic manipulators can have significant amounts of mechanical energy that can, in case of undesired contact, inflict severe injuries to human operators. Traditional safety measures, defined for instance by ANSI/RIA R15.06-1999 (American National Standard for Industrial Robots and Robot Systems—Safety Requirements)[1] prescribe that safety is achieved by physically separating personnel from the robots. This standard is written specifically for industrial robots, and is not applicable to autonomous or service robots.[25] A more elaborated standard ISO 10218-1:2006[2] was introduced as an attempt to define new collaborative operation requirements for industrial robots. However, the results reported in ref. [15] demonstrated that these requirements are unnecessarily conservative, and therefore strongly limit the performance of the robot.

A large attention is devoted to the problem of safety in the literature. In the work of Ikuta *et al.*,[18,19,39] the safety strategies were classified as pre-contact or post-contact strategies. Moreover, the minimization of the risk in interaction by means of mechanical design and by means of control is discussed. Although the main issue was safety in human-care robotics, they introduced the first systematic quantitative method called *danger index* in safety evaluation, concerning human–robot interaction in general.

In ref. [15], Haddadin *et al.* give an overview of their systematic evaluation of safety in human–robot interaction, covering various aspects of the most significant injury mechanisms. Zinn *et al.*[52] used empirical formulas developed by the automotive industry to correlate head acceleration to injury severity (head injury criteria) in order to evaluate the potential for serious injury due to impact. The work was mainly oriented towards new actuation concepts in the human-friendly robot design. They stressed out the importance of joint torque control approach and series elastic actuation.

Heinzmann and Zelinsky[16] proposed a control scheme for robotic manipulators that restricts the torque commands of a position control algorithm to values that comply to predefined quantitative safety restrictions. For that purpose, they defined a quantity called *impact potential* as a maximum impact force that a moving mechanical system can create in a collision with a static obstacle.

In the successive publications of Kulic and Croft,[26–28] several specific safety strategies were presented as components of an extensive methodology for safe planning and control in human–robot interaction. They addressed the important issue of estimating the human intent and affective state during the interaction. The information about the intent or the state of the human is used within a planning and control strategy to improve safety and intuitiveness of the interaction. Further, several danger indices have been formulated and used as an input to a real-time trajectory generation. A motion strategy consists of minimizing the danger index during a stable robot operation. The information about the human state, intent, and the environment is acquired using a computer vision-based system and the measurement of some physiological signals.

In ref. [9], Brock and Khatib propose a general framework for motion planning and execution in human environment. It is based on the notion of elastic strips—structures deformable according to external influences. Elastic strips are built upon the initial path that is considered *a priori* known. Safety is not addressed explicitly.

As far as the path planning problem is concerned, the exact algorithms that are based on the *a priori*

* Corresponding author. E-mail: bakir.lacevic@etf.unsa.ba

knowledge of the complete configuration space (C-space) are proven to be exponential in the dimensionality of the C-space.[11] Due to the questionable applicability of such an approach for the problems with high dimensionality, a sampling-based paradigm has gained more popularity. The two most prominent methods are: rapidly exploring random tree (RRT) algorithm[24,35] and probabilistic roadmap (PRM) planner.[23] Although incomplete, these methods have been proven capable of efficiently solving many challenging, high-dimensional motion planning problems.[13] Both methods rely on randomly sampling the C-space, constructing a graph upon these samples via local collision-free paths, and outputting the solution as the path from the initial to goal configuration across the graph edges. Since their invention, both PRM and RRT have subsequently undergone many modifications to become extremely efficient tools for solving specific motion planning problems (see e.g., refs. [8,17,42,45–48,51]). For a large overview of motion planning methods, the reader is referred to refs. [12,34,36].

Most of the planning algorithms provide feasible paths when considering solely the "hard constraints," taking care only that no intersections between the robot and the obstacles occur.[43] Some efforts to impose additional requirements upon the path, like staying away from certain areas as much as possible, can be found in refs. [38,43]. In ref. [14], authors propose an algorithm for improving a clearance along an already given path in a post-processing phase. The method is rather elaborate, but computationally expensive and not easily reproducible. Moreover, improving clearance does not necessarily imply the safety enhancement.[29] In ref. [49], authors perform an incremental exploration of the configuration space to generate high-quality paths that are comparable with probabilistic methods and can even improve some aspects, such as the completeness. Jaillet *et al*.[20] provide a framework for RRT-based path planning that considers a generic cost function defined over the configuration space. The proposed algorithm uses transition tests to accept or reject new potential states. Unfortunately, such transition tests appear to be a limiting factor in terms of performance when considering bidirectional RRT.[20] Berenson *et al*.[4] combined gradient descent search with the approach from ref. [20] to navigate cost space chasms, which are loosely defined as narrow, low-cost regions surrounded by increasing cost. In ref. [37], Mainprince *et al*. presented a method to increase the quality of the human–robot interaction at motion planning level. The method accounts for three constraints: distance, visibility, and comfort that capture relevant properties like position, kinematics, and field of view of the human. In ref. [41], visibility-graph-based heuristic algorithm for safe path planning in 2D and 3D space has been proposed. The method relies on the concept of formidable zones to prevent unsafe joint angle configurations. Karaman and Frazzoli[22] provided a strong theoretical result by rigorously analyzing the asymptotic behavior of the cost of the solution returned by stochastic sampling-based algorithms as the number of samples increases. They proposed PRM* and RRT*—the asymptotically optimal extensions of the classical algorithms—that have been proved to lack the feature of optimality.

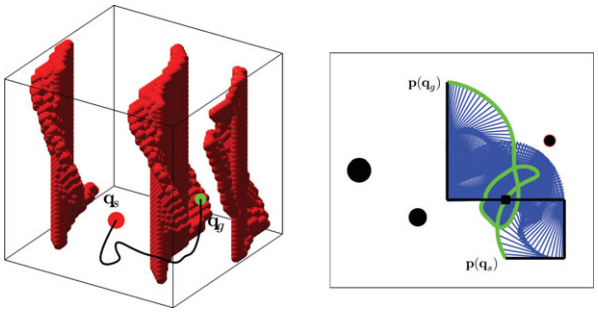

Fig. 1. (Colour online) Path in C-space and the corresponding manipulator motion.

This work integrates the authors' contributions on design of safety-oriented path planning algorithms.[1] Four algorithms are presented and tested within a unique case study that is used to evaluate and compare their performance. For illustrating the performance enhancement, corresponding classical planning algorithms are also considered. The main idea behind the presented approach is to use a meaningful measure of safety/danger, the danger field, in order to bias the exploration of the configuration space within the planning phase. Consequently, the planning algorithm outputs safer path.

The remainder of the paper is organized as follows. In Section 2, the problem of safety-oriented planning is defined. Safety/danger assessment is described in Section 3. Section 4 brings the description of the algorithms: the one based on bubbles of free configuration space, PRM-based planner, and two algorithms based on RRT. Case studies and discussion of results are given in Section 5 while conclusions and future work directions are given in Section 6.

2. Problem Description

The proposed planning algorithms are designed to find a path in C-space from the start configuration $\mathbf{q}_s$ to the goal configuration $\mathbf{q}_g$ (or to a goal region if the task is defined in the workspace). It is desirable that the path is collision-free while at the same time minimizing the danger field (or keeping it below a certain value) induced by the robot's configuration at the obstacles' locations. An example of a collision-free path in C-space and a consequent motion of a 3-degree of freedom (DOF) planar manipulator is shown in Fig. 1. The C-space is represented as a cube $[-\pi, \pi]^3$ whose axes correspond to joint angles. It includes the C-space obstacles—the set of all configurations that cause the intersection between the robot and the obstacles.[7] The initial and final positions ($\mathbf{p}(\mathbf{q}_s)$ and $\mathbf{p}(\mathbf{q}_g)$ respectively), as well as the path described by the end-effector are indicated.

For redundant manipulators, the goal configuration $\mathbf{q}_g$, can be extended to the *goal region*, i.e., the set of all configurations that enable the robot to complete the task (e.g., grasp the object).[5–7] For instance, if the goal is equivalent to the end-effector reaching the point $\mathbf{p}$ in the workspace, the goal region in C-space is the collision-free portion of the self-motion manifold corresponding to $\mathbf{p}$.[10]

[1] Partial results of this paper can be found in refs. [31–33].

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

3. Danger Assessment

In ref. [30], the concept of the danger field as a safety assessment tool is introduced. For the alternative ways to assess the danger in the environment of the robotic manipulator, the reader is referred to refs. [16,18,27,28]. In this work, the danger field is used as an ingredient for the heuristic function of the path planner. For the sake of completeness, we give a brief definition of the static version of the danger field.[2]

Let T be a point mass whose position is given with $\mathbf{r}_t = (x_t\ y_t\ z_t)^T$. For convenience, we put $\rho_t = \|\mathbf{r} - \mathbf{r}_t\|$, where $\mathbf{r} = (x\ y\ z)^T$ is a generic point in the world frame.

Definition 3.1. A differentiable scalar function $DF = DF(\mathbf{r}, \mathbf{r}_t)$ is called a static danger field (SDF) if it satisfies the conditions:

(i) $\exists f : \mathbb{R}^3 \to \mathbb{R}^+$, such that $DF(\mathbf{r}, \mathbf{r}_t) \equiv f(\rho_t)$,
(ii) $\frac{\partial f(\rho_t)}{\partial \rho_t} < 0,\ \forall \rho_t > 0$,

We now extend the principle of SDF in the sense that the danger source is no longer a point, but a part of curve in $\mathbb{R}^3$. Let $\mathbf{r}_t : [0, S] \to \mathbb{R}^3$ be the mapping that represents the piecewise smooth curve $\mathbf{r}_t(t) = (x(t)\ y(t)\ z(t))^T$, where t is the natural parameter and S is the length of the curve.

Definition 3.2. If $DF(\mathbf{r}, \mathbf{r}_t)$ is SDF, then the cumulative static danger field *(*CSDF*)* is defined as:

$$CDF(\mathbf{r}) = \int_0^S DF(\mathbf{r}, \mathbf{r}_t)dt. \tag{1}$$

CSDF captures the contribution of the curve's position in $\mathbb{R}^3$. It is now possible to define the CSDF of the rigid robot manipulator using Eq. (1), where the curve over which the integration is performed is the line approximation of the kinematic chain. Knowing the position of the link endpoints, one could evaluate the position of any point on the chain just by using forward kinematics. Further, it is natural to compute the contribution of each link separately and then obtain the CSDF as the superposition of these contributions. Let $\mathbf{r}_i$ and $\mathbf{r}_{i+1}$ be the positions of the endpoints of link i. Any point $\mathbf{r}_t$ on the link i could be represented as:

$$\mathbf{r}_t = \mathbf{r}_i + t\,(\mathbf{r}_{i+1} - \mathbf{r}_i)\,, \quad t \in [0, 1]. \tag{2}$$

Now, we can express some characteristic quantities that play role in the expressions for CSDF. First of all:

$$\mathbf{r} - \mathbf{r}_t = \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} r_{ix} + t(r_{i+1\,x} - r_{ix}) \\ r_{iy} + t(r_{i+1\,y} - r_{iy}) \\ r_{iz} + t(r_{i+1\,z} - r_{iz}) \end{bmatrix} \equiv \begin{bmatrix} \alpha_1 + \alpha_2 t \\ \beta_1 + \beta_2 t \\ \gamma_1 + \gamma_2 t \end{bmatrix}. \tag{3}$$

The module of the above vector is $\rho_t^2 = \|\mathbf{r} - \mathbf{r}_t\|^2 = at^2 + bt + c$, where $a = \alpha_2^2 + \beta_2^2 + \gamma_2^2$, $b = 2(\alpha_1\alpha_2 + \beta_1\beta_2 + \gamma_1\gamma_2)$, and $c = \alpha_1^2 + \beta_1^2 + \gamma_1^2$.

[2] The general version of the danger field takes into account both the robot's position and velocity, while in this paper we consider only the position.

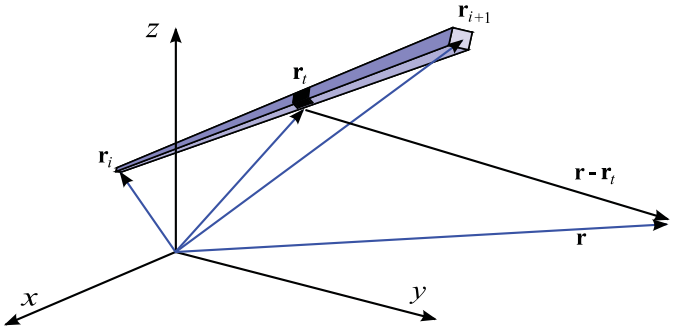

Fig. 2. (Colour online) Elements that play role in the computation of elementary danger field.

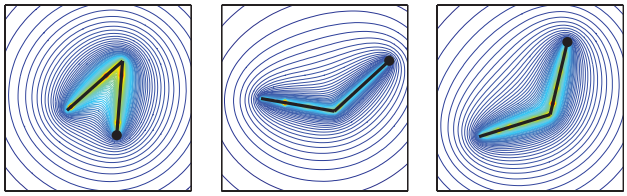

Fig. 3. (Colour online) Snapshots of the danger field's contour plot: a 2-DOF example (the robot base is marked).

The CSDF of the link i is given with:

$$CDF_i(\mathbf{r}) = \int_0^1 DF(\mathbf{r}, \mathbf{r}_t)dt = \int_0^1 f(\rho_t)dt. \tag{4}$$

Consequently, the CSDF of the n-DOF robot arm is:

$$CDF(\mathbf{r}) = \sum_{i=1}^{n} CDF_i(\mathbf{r}). \tag{5}$$

We propose the elementary CSDF, induced by the motion of infinitesimal portion of the link as:

$$DF(\mathbf{r}, \mathbf{r}_t) = \frac{k_1}{\|\mathbf{r} - \mathbf{r}_t\|}, \tag{6}$$

where k_1 is a positive constant, $\mathbf{r}$ is a point in space at which the field is being computed, and $\mathbf{r}_t$ is the position of the link's element (see Fig. 2). The CDF induced by the motion of the complete link can be expressed as:

$$CDF(\mathbf{r}, \mathbf{r}_i, \mathbf{r}_{i+1}) = k_1 \int_0^1 \frac{dt}{\sqrt{at^2 + bt + c}}. \tag{7}$$

The integral (7) is solvable analytically and hence, the value of the danger field at any point of the space can be evaluated via an algebraic expression. Inputs to the expression are $\mathbf{r}_i$ and $\mathbf{r}_{i+1}$ (obtainable from forward kinematics) and a generic position $\mathbf{r}$. As previously stated, by the simple superposition of the influences of many arbitrarily articulated links, one could obtain the danger field induced by the complete kinematic chain. Figure 3 shows the contour plot of the field induced by the motion of a 2-DOF planar manipulator. To make it presentable as a function of two variables, the danger field is restricted to the plane in which the robot moves. The field $CDF(\mathbf{r})$ is by definition a scalar field. Nevertheless, a vector field can easily be constructed upon it. The most

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

natural way to do so is by using its gradient:

$$\overrightarrow{CDF}(\mathbf{r}) = CDF(\mathbf{r})\frac{\nabla CDF(\mathbf{r})}{\|\nabla CDF(\mathbf{r})\|}. \quad (8)$$

Thus, $\overrightarrow{CDF}(\mathbf{r})$ is a vector, anchored in $\mathbf{r}$, with the magnitude $CDF(\mathbf{r})$, pointing in the direction defined by $\nabla CDF(\mathbf{r})$.

4. Planning Algorithms

In this section, an approach to path planning that seeks for safe paths using a suitably tailored heuristic function that embeds the safety assessment is described. In particular, four versions of the planner are considered. The first version is based on the tree expansion via so-called *bubbles of free configuration space* that are first introduced in ref. [40]. The algorithm simultaneously chains the bubbles both from initial and goal configurations until these chains intersect. The chain growth is guided by the danger field information. The second version of the planner is designed within a PRM context. A bidirectional, safety-oriented heuristic search is applied to obtain a path between initial and goal configurations. The third planner is based on RRT paradigm. Two modifications of classical RRT planners are proposed. The first is based on *Jacobian Transpose-directed* RRT algorithm[48] that grows a single tree from the start configuration towards the goal defined in the workspace. The second is a modification of the standard bidirectional RRT-connect planner[24] where the inputs to the algorithm are the start and the goal configuration that serve as seeds for the tree's growth. Parts of the work from this section can be found in refs. [31–33].

4.1. Safe path planning based on bubbles of free C-space

The algorithm is based on the concept of so-called bubbles of free configuration space, introduced in ref. [40]. The bubble $\mathcal{B}(\mathbf{q})$ at the current configuration $\mathbf{q}$ is a compact region computed using a distance d_c of the robot in configuration $\mathbf{q}$ from the closest obstacle in the workspace. For the robots with n revolute joints, it takes a diamond shape:[40]

$$\mathcal{B}(\mathbf{q}) = \left\{ \mathbf{x} \ : \ \sum_{i=1}^{n} r_i |x_i - q_i| < d_c \right\}. \quad (9)$$

The quantity r_i is the radius of the cylinder whose axis is collocated with the axis of the ith joint and that encloses all the links starting from ith joint to the end-effector. Changing the configuration from $\mathbf{q}$ to an arbitrary configuration within a bubble implies that no point on the kinematic chain will move more than d_c and thus no collision will occur. An elaborate assertion of the simplicity in computing the bubbles can be found in ref. [40]. Figure 4 shows a couple of configurations for the 2-DOF planar manipulator and the corresponding bubbles of free configuration space. The proposed algorithm tries to connect initial configuration $\mathbf{q}_s$ and the goal configuration $\mathbf{q}_g$ by simultaneously chaining bubbles both from initial and goal configurations until these chains intersect. Its principle is given with the pseudocode for the procedure BUBBLE_PLANNER.

The code is based on the bidirectional A^* search algorithm.[21,36] Lists named "Closed()" stand for the lists of visited nodes (configurations), while the waiting lists of nodes yet to be considered are labeled "Open()." Index 1 stands for the list (tree) that is currently being processed. At each iteration, function INTERSECT checks whether there are $\mathbf{q}_1 \in$ Closed(1) and $\mathbf{q}_2 \in$ Closed(2) such that $\mathcal{B}(\mathbf{q}_1) \cap \mathcal{B}(\mathbf{q}_2) \neq \emptyset$. If that is the case, there is a collision-free path between the trees expanded from $\mathbf{q}_s$ and $\mathbf{q}_g$ and hence a path between $\mathbf{q}_s$ and $\mathbf{q}_g$. Otherwise, the most promising node $\mathbf{q}_{new}$ is selected from the list Open(1) that minimizes the heuristic function f (described later). The configuration $\mathbf{q}_{new}$ is deleted from the list Open(1) and is added to the list of visited nodes Closed(1). Then the bubble $\mathcal{B}(\mathbf{q}_{new})$ is computed and its vertices are added to the list Open(1). The search direction is then reversed by swapping the corresponding lists. The variable dir preserves the information about the search direction. If we expand the tree originating from $\mathbf{q}_s$ then $dir = 1$, otherwise $dir = 2$. If the collision-free path is not obtained within a predefined number of iterations k_{max}, the algorithm returns failure.

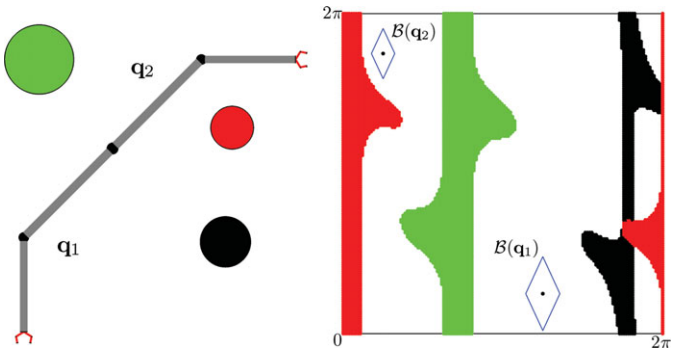

Fig. 4. (Colour online) Two configurations $\mathbf{q}_1$ and $\mathbf{q}_2$ of the 2-DOF planar manipulator and the corresponding bubbles $\mathcal{B}(\mathbf{q}_1)$ and $\mathcal{B}(\mathbf{q}_2)$ shown in C-space.

```
procedure BUBBLE_PLANNER(q_s, q_g)
    Closed(1) ← [q_s];
    Closed(2) ← [q_g];
    Open(1) ← [q_s];
    Open(2) ← [q_g];
    dir ← 1; q_current_goal ← q_g;
    for k = 1 to k_max do
        if INTERSECT (Closed(1), Closed(2)) then
            return PATH (Closed(1), Closed(2));
        end if
        q_new ← argmin_{x ∈ Open(1)} f(x);
        REMOVE (Open(1), q_new);
        ADD (Closed(1), q_new);
        ADD (Open(1), BUBBLE_ENDPOINTS (q_new));
        SWAP (Closed(1), Closed(2));
        SWAP (Open(1), Open(2));
        dir ← 3 − dir;
        q_current_goal ← q_new;
    end for
    return Failure
end procedure
```

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

The heuristic function f is defined as:

$$f(\mathbf{x}) = g(\mathbf{x}) + h(\mathbf{x}) + \alpha C\hat{D}F(\mathbf{x}) - \beta \min_{z \in C} \|\mathbf{x} - \mathbf{z}\|, \quad (10)$$

where $g(\mathbf{x})$ is a cost function of a traversed path from root to $\mathbf{x}$, and $h(\mathbf{x}) = \|\mathbf{x} - \mathbf{q}_{current\_goal}\|_1$ is the underestimate of the distance between $\mathbf{x}$ and the current goal $\mathbf{q}_{current\_goal}$. The term $C\hat{D}F(\mathbf{x})$ serves as the estimate of the maximum value of the danger field CDF induced by the robot in configuration $\mathbf{x}$ over all of the relevant subjects/obstacles locations. The last term in Eq. (10) represents the contribution of the point $\mathbf{x}$ to the spatial diversity of the samples (in the corresponding tree with a list of visited nodes $C = \text{Closed}(dir)$) already processed during the search. The point is considered better if it has a larger minimum distance from the set of already visited nodes. The idea is to increase the tendency of the algorithm to explore less-visited parts of the configuration space. Variables α and β are positive tunable parameters.

Unlike the planners that are based on RRT paradigm, the proposed algorithm updates the search trees in a deterministic manner. As for the computational effort needed to extend the tree towards the new configuration, the proposed algorithm does not require the collision-checking routine (unlike the RRT algorithm) because the entire edge that is being added to the tree lies within the bubble of free configuration space. On the other hand, the computation of the bubble around a given configuration requires only marginally more effort than determining solely whether a configuration is collision-free.[40] The only additional requirement is the memory space needed for maintaining the Open() lists. Partial resolution to this problem may be the substitution of the A^* search with the *iterative deepening* A^* algorithm.[21,36] If the original diamond-shaped bubble is replaced by its smaller subset—a largest inscribed axis-aligned hypercube—the completeness of the algorithm can be proved.[32]

4.2. PRM-based safe path planning

As in the bubble-based planner, the algorithm is based on the bidirectional A^*-search algorithm.[21,36] It builds two graphs simultaneously from the initial and the goal configurations using random samples as milestones. When the two graphs meet, the algorithm outputs the collision-free path. The expansion of the graphs is dictated by the modified heuristic function f, similar to that used within the BUBBLE_PLANNER, but without the diversity term. In a similar fashion as in refs. [8,42], the algorithm is lazy in collision checking, i.e., the collision-checking routine is invoked only when the algorithm tries to establish an edge that connects the current milestone to its candidate neighbors. The principle of the algorithm is given in the procedure SAFE_PRM_PLANNER. Note that some of the details typical for A^*-search are omitted for brevity.

At each iteration, function CONNECTED checks whether there are $\mathbf{q}_1 \in$ Closed(1) and $\mathbf{q}_2 \in$ Closed(2) such that $\mathbf{q}_1 = \mathbf{q}_2$. If that is the case, there is a collision-free path between the graphs expanded from $\mathbf{q}_s$ and $\mathbf{q}_g$ and hence a collision-free path between $\mathbf{q}_s$ and $\mathbf{q}_g$. Otherwise, the most promising node $\mathbf{q}_{new}$ is selected from the list Open(1) that minimizes the heuristic function. The function NEIGHBORS picks the nearest neighbors of the node $\mathbf{q}_{new}$ and puts them to the list Open(1). It is done as follows. The nearest K nodes ($K \in \mathbb{N}$) to $\mathbf{q}_{new}$ from the set Samples\Closed(1) are chosen as the candidate neighbors of $\mathbf{q}_{new}$. The choice is made with respect to the metric:

$$D(\mathbf{q}_1, \mathbf{q}_2) = \sum_{i=1}^{M} \|\mathbf{p}_i(\mathbf{q}_1) - \mathbf{p}_i(\mathbf{q}_2)\|^2, \quad (11)$$

where $\mathbf{p}_i$ represents the world coordinate for the distal point of the ith link of the M-DOF manipulator. This metric function captures the area swept by the robot while moving from the configuration $\mathbf{q}_1$ to the configuration $\mathbf{q}_2$. The smaller this metric is, the more likely it is that the corresponding local path defined by the straight line from $\mathbf{q}_1$ to $\mathbf{q}_2$ will be collision-free. Now, for each of the candidate neighbors, an attempt is made by a local planner to connect it to its parent $\mathbf{q}_{new}$. The local planner simply connects the two given configurations by a straight line (in C-space) and subsequently checks this line for collisions.[23] If the line segment is collision-free, the corresponding candidate neighbor is added to the list Open(1) via the function NEIGHBORS. The search direction is then reversed by swapping the corresponding lists.

```
procedure SAFE_PRM_PLANNER(q_s, q_g)
    Samples←RANDOM_SET();
    Samples←Samples ∪ {q_s, q_g}
    Closed(1) ← [q_s];
    Closed(2) ← [q_g];
    Open(1) ← [q_s];
    Open(2) ← [q_g];
    q_current_goal ← q_goal;
    while ¬EMPTY(Open(1))∧¬EMPTY(Open(2)) do
        if CONNECTED (Closed(1), Closed(2)) then
            return PATH (Closed(1), Closed(2));
        end if
        q_new ← argmin_{x ∈ Open(1)} f(x);
        REMOVE (Open(1), q_new);
        ADD (Closed(1), q_new);
        ADD (Open(1), NEIGHBORS (q_new));
        SWAP (Closed(1), Closed(2));
        SWAP (Open(1), Open(2));
        q_current_goal ← q_new;
    end while
    return Failure
end procedure
```

4.3. Safe path planning based on JT-RRT algorithm

SAFE_JT-RRT algorithm is a modification of *Jacobian Transpose-directed* RRT (JT-RRT) algorithm[48] that grows a single tree from the start configuration and uses the transpose of the Jacobian to guide the sampling towards the goal defined in the workspace. The modified algorithm accounts for safety in the tree expansion. Note that the goal is not a single point in C-space, but any configuration that is mapped by

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

procedure SAFE_JT-RRT($\mathbf{q}_s$, $\mathbf{x}_g$)
  $Q \leftarrow [\mathbf{q}_s]$;
  while (WS_DISTANCE(Q, $\mathbf{x}_g$) $\geq D_{Threshold}$) **do**
    $p \leftarrow$ RAND(0, 1);
    if $p < p_g$ **then**
      $Q \leftarrow$EXTEND_TOWARDS_GOAL(Q,$\mathbf{x}_g$);
    else
      $Q \leftarrow$EXTEND_RANDOMLY(Q);
    end if
  end while
end procedure

forward kinematics into a desired workspace position $\mathbf{x}_g$. The pseudocode of the algorithm is given by the procedure SAFE_JT-RRT.

The core of the algorithm is the unidirectional RRT where the tree comprised of already processed nodes (configurations) is represented by Q. As in ref. [48], the tree expansion switches between two modes until the robot becomes close enough to its workspace-defined goal (defined by the function WS_DISTANCE). The first mode (procedure EXTEND_RANDOMLY) is similar to a common random tree extension. When a random configuration $\mathbf{q}_{rand}$ is generated, the tree is extended from the node $\mathbf{q}$ that has the smallest value of the function $\|\mathbf{q} - \mathbf{q}_{rand}\| + \alpha\|\overrightarrow{CDF}(\mathbf{q})\|$, where α is a positive parameter. The term $\|\overrightarrow{CDF}(\mathbf{q})\|$ stands for the maximum value of the danger field induced by the robot in configuration $\mathbf{q}$ over all of the relevant obstacle locations. The choice of node $\mathbf{q}$, from which the tree is extended, represents the compromise between the distance to $\mathbf{q}_{rand}$ and the measure of danger.

The second mode is the extension from the current configuration towards the one that reduces the distance measured in the workspace (procedure EXTEND_TOWARDS_GOAL). Thus, the algorithm has both exploitation features in terms of the biased search towards the goal and the capability to efficiently explore the configuration space.[48]

Within the EXTEND_TOWARDS_GOAL mode, a pseudoinverse $\mathbf{J}^\dagger$ of the Jacobian matrix $\mathbf{J}$ is used to compute the desired displacement $\Delta\mathbf{q}$ in the C-space that should ensure the decrease of the corresponding position error in the workspace. It draws motivation from the closed-loop inverse kinematics algorithm (CLIK).[44,50] Unlike the original JT-RRT algorithm, the modified one uses the information about the danger field to shape the null-space motion in order to exploit the robot's kinematic redundancy in a way that the robot takes safer postures without compromising the prescribed motion of the end-effector. Clearly, this approach will take effect only if the manipulator is functionally redundant with respect to a given task. Note that the original JT-RRT does not require a pseudoinverse $\mathbf{J}^\dagger$ but uses $\mathbf{J}^T$ instead to emulate the inverse kinematics solution from ref. [50].

At first, the configuration $\mathbf{q}_{old}$ with the smallest value of the function

$$\text{WS\_DISTANCE}\left(f_k(\mathbf{q}), \mathbf{x}_g\right) + \alpha\|\overrightarrow{CDF}(\mathbf{q})\|$$

procedure EXTEND_TOWARDS_GOAL(Q, $\mathbf{x}_g$)
  $\mathbf{q}_{old} \leftarrow \underset{\mathbf{q}\in Q}{\text{argmin}}\left\{\text{WS\_DISTANCE}\left(f_k(\mathbf{q}), \mathbf{x}_g\right) + \alpha\|\overrightarrow{CDF}(\mathbf{q})\|\right\}$;
  while 1 **do**
    $\Delta\mathbf{x} \leftarrow \mathbf{x}_g - f(\mathbf{q}_{old})$;
    $\Delta\mathbf{q}_0 \leftarrow k_1 \sum_{j=1}^{N}\sum_{i=1}^{n} \mathbf{J}_{i,j}^T \overrightarrow{CDF}(\mathbf{r}_j)$;
    $\Delta\mathbf{q} \leftarrow k_2\left[\mathbf{J}^\dagger\Delta\mathbf{x} + \left(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J}\right)\Delta\mathbf{q}_0\right]$;
    $\mathbf{q}_{new} \leftarrow \mathbf{q}_{old} + \Delta\mathbf{q}$;
    if COLLISION_FREE($\mathbf{q}_{old}$, $\mathbf{q}_{new}$)) **then**
      if ($\|\overrightarrow{CDF}(\mathbf{q}_{new})\| < CDF_{Threshold}$) **then**
        $Q \leftarrow Q \cup \mathbf{q}_{new}$;
      else return Q;
      end if
    else return Q;
    end if
    if ($\|\mathbf{x}_g - f(\mathbf{q}_{new})\| < D_{Threshold}$) **then**
      return Q;
    end if
    $\mathbf{q}_{old} \leftarrow \mathbf{q}_{new}$;
  end while
end procedure

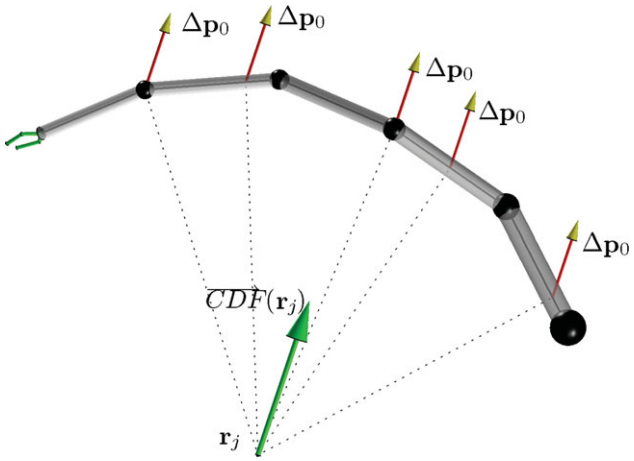

Fig. 5. (Colour online) Mapping the vector of the danger field into desired displacements $\Delta\mathbf{p}_0$ of several points of interest on the manipulator.

is chosen, where the function f_k stands for the forward kinematics mapping. For efficient implementation, a sorted list of configurations with respect to cost function may be maintained. Picking larger α in general yields safer paths, in a similar manner to the safety-oriented PRM approach (see ref. [31]). This choice of $\mathbf{q}_{old}$ represents the compromise between the vicinity to a workspace-defined goal and the measure of danger. If $\mathbf{q}_{old}$ has been chosen before, the next best configuration is taken. Then, the position error $\Delta\mathbf{x}$ in the workspace is computed. Moreover, a configuration displacement $\Delta\mathbf{q}_0$ whose role is to decrease the danger is obtained via expression given in the pseudocode. The idea is to map the danger field vector $\overrightarrow{CDF}$ into the desired Cartesian displacement $\Delta\mathbf{p}_0$ for each link (see Fig. 5).

These displacements are then easily mapped into configuration displacements via corresponding Jacobian matrices. The matrix $\mathbf{J}_{i,j}$ represents the first three rows of the Jacobian associated to a relevant point on the link i, $i = 1, 2, \ldots, n$, where n is the number of links. An intuitive

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

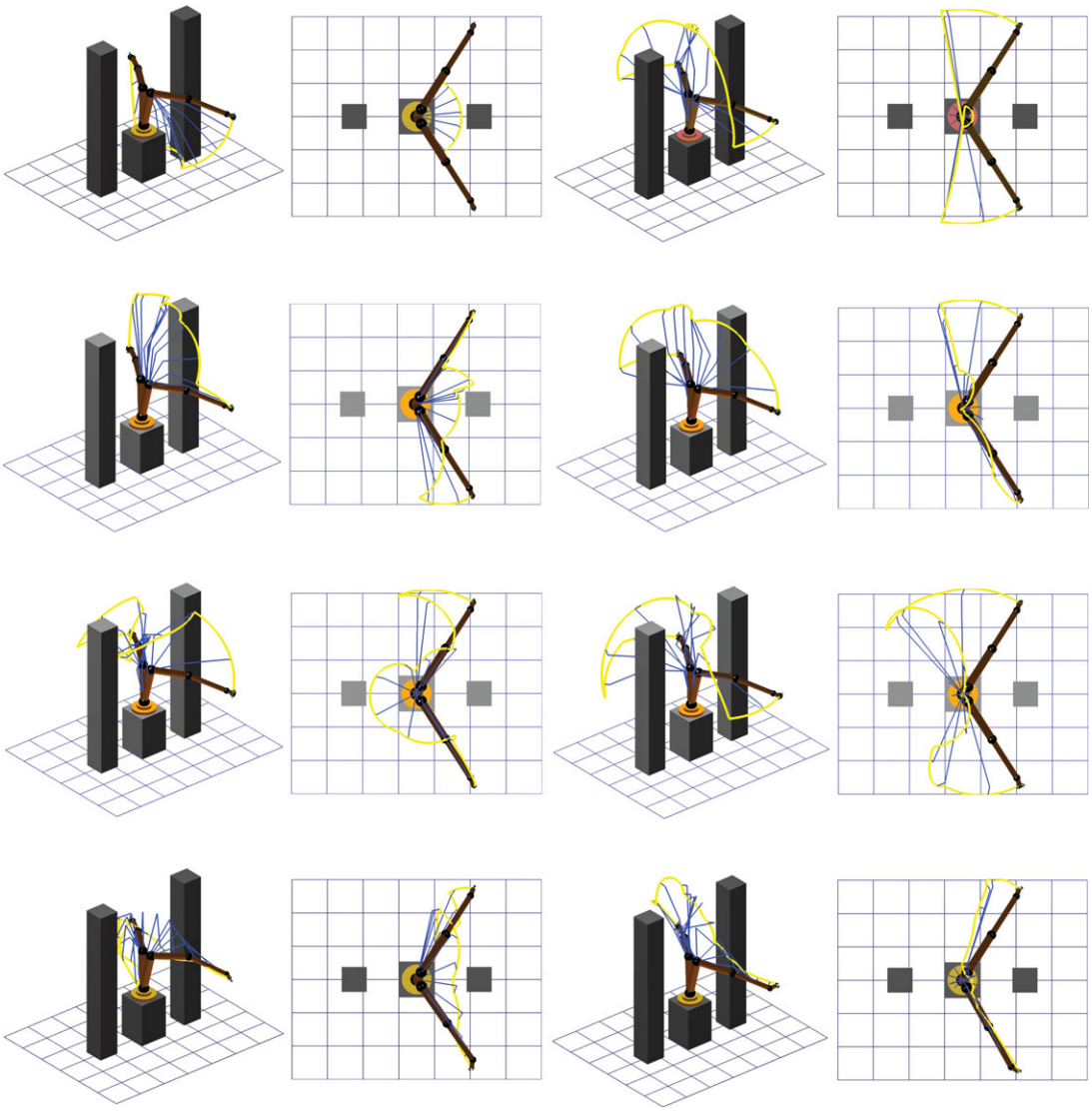

Fig. 6. (Colour online) First scenario: typical solution paths for the considered planners. First row: BUBBLE_PLANNER without safety heuristic (left) and BUBBLE_PLANNER with safety heuristic (right). Second row: Classical PRM planner (left) and SAFE_PRM_PLANNER (right). Third row: RRT-Connect planner (left) and SAFE_RRT-Connect planner (right). Fourth row: JT-RRT planner (left) and SAFE_JT-RRT planner (right).

choice for a relevant point is the point on the link that is the closest to the obstacle[3] $\mathbf{r}_j$ (as in Fig. 5). A computationally more convenient option, which is indeed used for the validation of the algorithm, is to choose these points as the endpoints of the links. Extensive simulations show that such a choice does not compromise the performance of the algorithm.

An update of the configuration $\Delta\mathbf{q}$ comprises two parts. The first part $\mathbf{J}^{\dagger}\Delta\mathbf{x}$ leads the robot towards the workspace goal. The second part is responsible for guiding the robot to safer posture. Note that the vector $\Delta\mathbf{q}_0$ does not affect the position/orientation of the end-effector because a suitable null-space projection is performed via matrix $\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}$. If the danger in the new configuration $\mathbf{q}_{new}$ does not exceed a certain threshold and the local path from $\mathbf{q}_{old}$ to $\mathbf{q}_{new}$ is collision-free, the node $\mathbf{q}_{new}$ is added to the tree. Otherwise, the loop is terminated. Lowering the threshold for the danger field usually increases the safety of the resulting path but may also increase the running time of the algorithm.

As pointed out in ref. [48], the step taken along the direction defined by $\Delta\mathbf{q}$ should not be too large before the Jacobian is re-evaluated. The norm of such a step is directly controlled by positive tunable parameters k_1 and k_2. Picking k_1 and k_2 too large may cause significant oscillations in the path. The approach used within the case study is to set k_1 and k_2 as large as possible provided that the oscillations do not occur.

4.4. Safe path planning based on RRT-Connect algorithm

An alternative version of RRT-based safe planning algorithm relies on the classical RRT-connect planner.[24] The original RRT-connect algorithm grows two RRTs, rooted at start and goal configurations, and tries to merge them in each iteration

[3] Note that in the expression for $\Delta\mathbf{q}_0$, the summation is performed over all of $j = 1, \ldots, N$ relevant obstacles. Moreover, the notation of the configuration $\mathbf{q}_{old}$, at which the Jacobian matrix J is computed, is omitted for brevity.

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

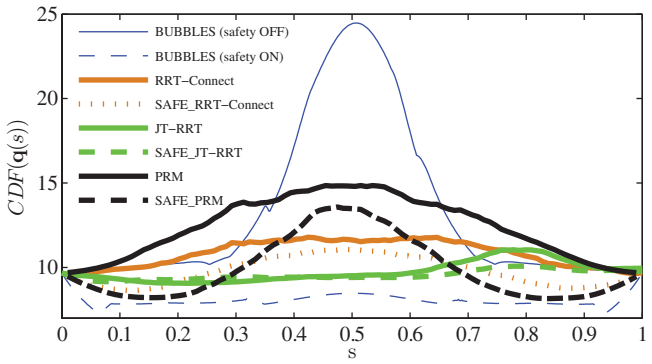

Fig. 7. (Colour online) Scenario 1: averaged danger field profiles vs. the path's natural parameter.

using greedy Connect heuristic (see ref. [24] for details). The modified algorithm (SAFE_RRT-CONNECT) differs in the way the trees are expanded whereas the Connect heuristic remains the same.

When a random configuration $\mathbf{q}_{rand}$ is generated, the best node $\mathbf{q}_{old}$ from the tree Q_1 is chosen and an attempt is made to expand the tree Q_1 from $\mathbf{q}_{old}$ to $\mathbf{q}_{rand}$. If such an expansion is possible, the new node $\mathbf{q}_{new}$ is added to the tree Q_1. In the original algorithm, the node $\mathbf{q}_{old}$ represents the configuration from Q_1 that is the closest to $\mathbf{q}_{rand}$ with respect to some metric function ρ, defined in C-space. In the SAFE_RRT-CONNECT algorithm, the configuration $\mathbf{q}_{old}$ minimizes the function $\rho(\mathbf{q}, \mathbf{q}_{rand}) + \alpha\|\overrightarrow{CDF}(\mathbf{q})\|$ that captures both the distance from $\mathbf{q}_{rand}$ and the danger assessment of the configuration $\mathbf{q}$. Thus, the trees are expanded from safer regions. Like within the SAFE_JT-RRT algorithm, larger α in general yields safer but longer paths. Each time the configuration $\mathbf{q}_{new}$ is obtained, an attempt is made to merge the trees Q_1 and Q_2 via nodes $\mathbf{q}_{new}$ and $\bar{\mathbf{q}}_{old}$, where $\bar{\mathbf{q}}_{old}$ is the configuration from the tree Q_2 that minimizes the function $\rho(\mathbf{q}, \mathbf{q}_{new}) + \alpha\|\overrightarrow{CDF}(\mathbf{q})\|$. If the connection is possible, the collision-free path can be constructed from the trees Q_1 and Q_2. Otherwise, the roles of trees Q_1 and Q_2 are reversed by swapping them and the expansion continues. If the collision-free path is not obtained after

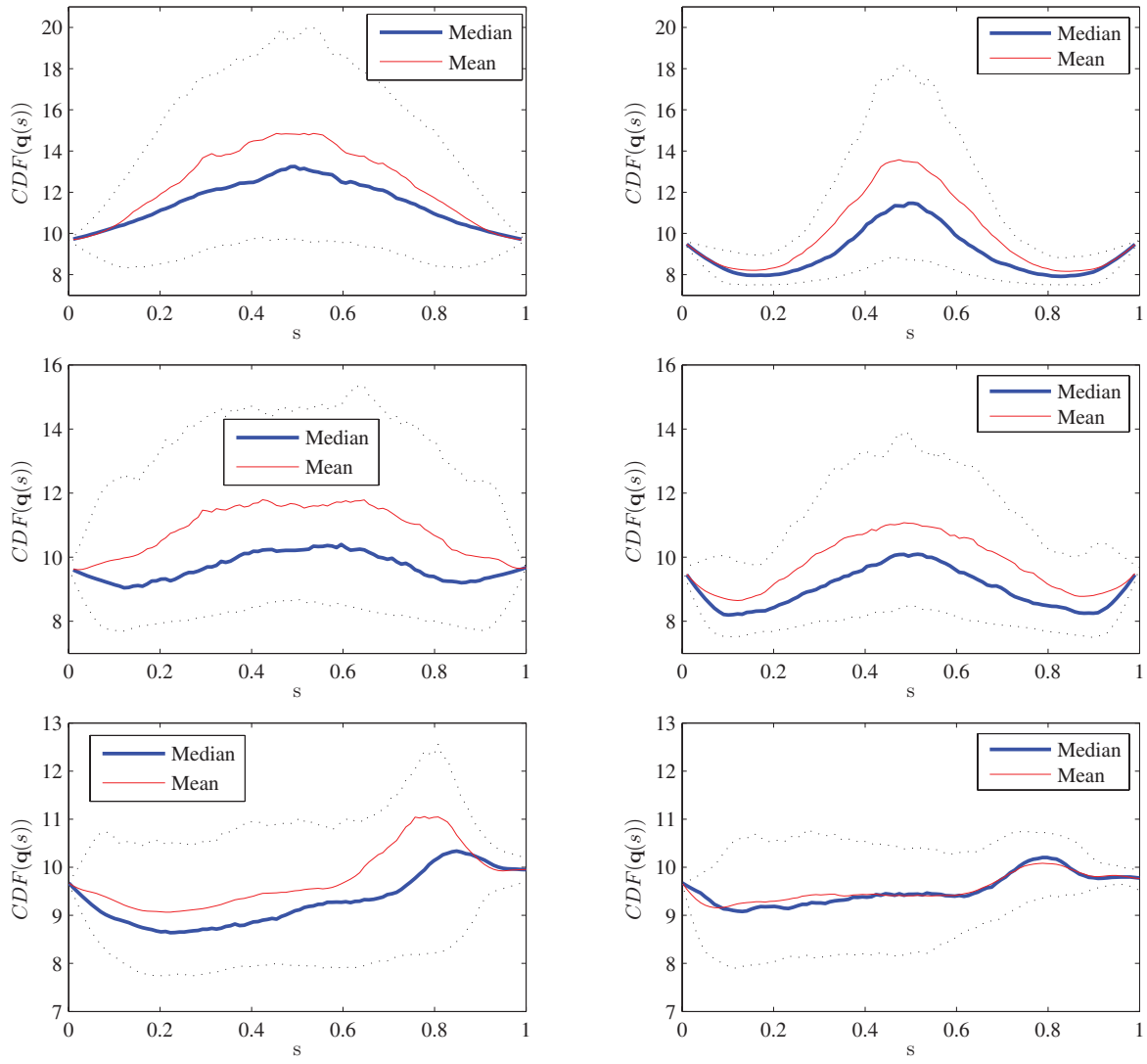

Fig. 8. (Colour online) Scenario 1: mean, median, 15th, and 85th percentiles of the danger field along the path's natural parameter. Top row: PRM (left) and SAFE_PRM (right). Middle row: RRT-Connect (left) and Safe_RRT-Connect (right). Bottom row: JT-RRT (left) and Safe_JT-RRT (right).

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

procedure SAFE_RRT-CONNECT($\mathbf{q}_s$, $\mathbf{x}_g$)
  $Q_1 \leftarrow [\mathbf{q}_s]$;
  $Q_2 \leftarrow [\mathbf{q}_g]$;
  for k=1 to K **do**
    $\mathbf{q}_{rand} \leftarrow$ RANDOM_CONFIG();
    $\mathbf{q}_{old} \leftarrow \underset{\mathbf{q} \in Q_1}{\text{argmin}} \left\{ \rho(\mathbf{q}, \mathbf{q}_{rand}) + \alpha \| \overrightarrow{CDF}(\mathbf{q}) \| \right\}$;
    $\mathbf{q}_{new} \leftarrow$ EXTEND($\mathbf{q}_{old}$, $\mathbf{q}_{rand}$);
    $\bar{\mathbf{q}}_{old} \leftarrow \underset{\mathbf{q} \in Q_2}{\text{argmin}} \left\{ \rho(\mathbf{q}, \mathbf{q}_{new}) + \alpha \| \overrightarrow{CDF}(\mathbf{q}) \| \right\}$;
    if CONNECT($\bar{\mathbf{q}}_{old}$, $\mathbf{q}_{new}$) = *Successful* **then**
      RETURN_PATH(Q_1, Q_2)
    end if
    SWAP(Q_1, Q_2)
  end for
  return *Failure*
end procedure

the predefined number of iterations K, the algorithm returns failure.

5. Numerical Experiments

In this section, all the planning algorithms are tested within two different scenarios. For validation purposes, a model of a 6-DOF robotic arm (anthropomorphic arm with spherical wrist[44]) is considered. It is worth pointing out that the efficiency of tested algorithms has also been shown using other types of robotic manipulators, e.g., a 3-DOF planar robot,[32] a 6-DOF robot with alternative kinematic structure,[32,33] and a 7-DOF robotic manipulator.[31] The algorithms are implemented within MATLAB, including the geometrical models of the robot and the environment. All the simulations are performed on Intel(R) Core(TM) i3-2100 CPU @ 3.10 GHz PC with 4 GB RAM. For a single scenario, each algorithm is executed 1000 times because of the innate randomness within the algorithms. This does not hold for the BUBBLE_PLANNER, which is inherently deterministic. The numerical study also considers classical algorithms: PRM, RRT-Connect, and JT-RRT. Figure 6 shows a simple scenario with the typical solution paths obtained by the tested algorithms. The manipulator has to reach the goal from a given initial configuration $\mathbf{q}_s$. For JT-RRT and SAFE_JT-RRT planners, the goal is defined by the desired Cartesian position of the end-effector without the specified orientation. For the remaining algorithms, the goal is defined by the desired configuration $\mathbf{q}_g$. According to Fig. 6, safety-oriented versions of the planning algorithms provide considerably safer postures of the manipulator along yielded paths. Figure 7 shows the averaged profiles of the danger field (over 1000 runs, except for the algorithms based on bubbles that have been run once) as the function of the path's scaled natural parameter $s \in [0, 1]$. Not surprisingly, the safety-oriented versions of the considered planners provide substantially safer paths. Another interesting indicator of the algorithm performance is the deviation of the danger field profiles from the averaged one. A smaller deviation implies the higher robustness of the planner with respect to inherent randomness within the algorithm in the sense that paths obtained under the same conditions are more likely to exhibit the same or similar safety features. Figure 8 shows mean, median, 15th, and 85th percentiles of the danger field along the path's natural parameter. Clearly, safety-based algorithms appear to output solution paths whose danger

Table I. Scenario 1: some numerical results.

Algorithm	Avg. time (s)	Nodes	Coll. checks/ Dist. computations	DF computations
BUBBLES (safety OFF)	3.1668	545	5268	–
BUBBLES (safety ON)	0.2808	181	550	206
PRM	0.8076	106	1545	–
SAFE_PRM	0.8427	90	1597	268
RRT-Connect	0.1387	28	359	–
SAFE_RRT-Connect	0.1636	19	379	46
JT-RRT	0.1604	103	328	–
SAFE_JT-RRT	0.1583	40	309	63

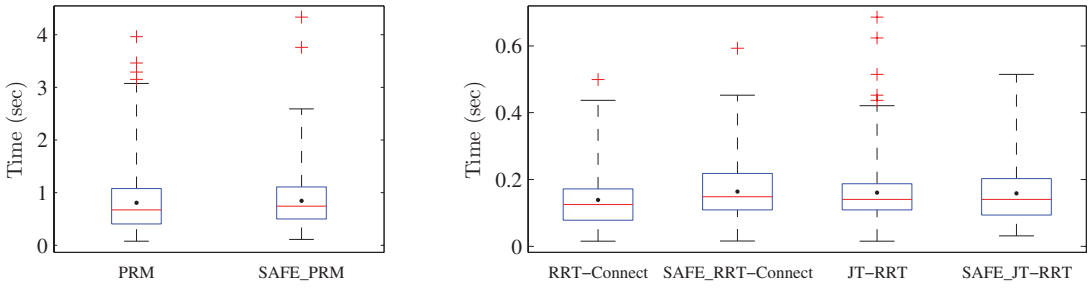

Fig. 9. (Colour online) Scenario 1: box plots for running times of considered algorithms. Mean value (·) and the outliers (+) are also indicated. Since the running times for PRM and SAFE_PRM are considerably higher in comparison with the remaining algorithms, they are shown separately for scaling purposes.

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

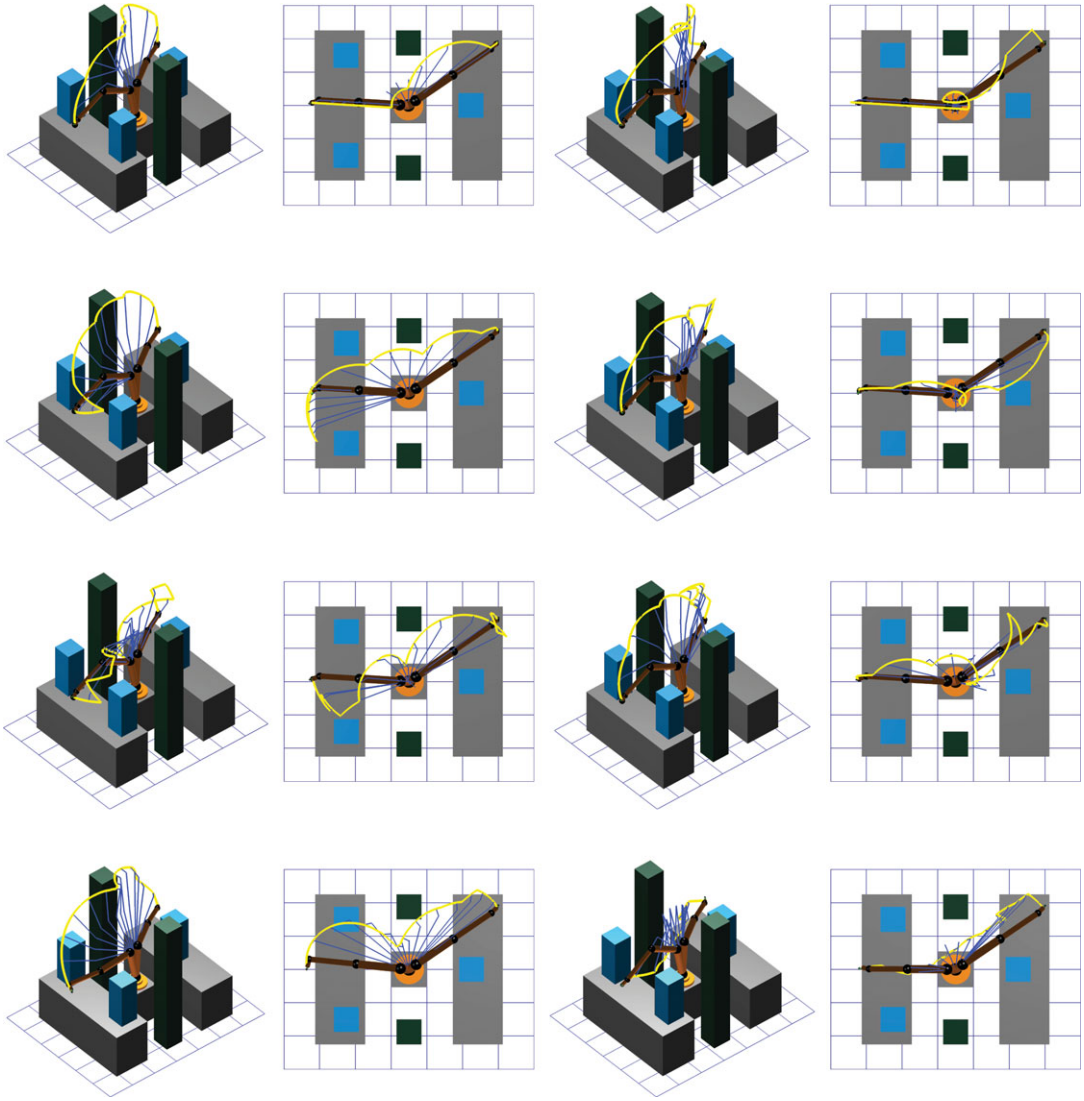

Fig. 10. (Colour online) Second scenario: typical solution paths for the considered planners. First row: BUBBLE_PLANNER without safety heuristic (left) and BUBBLE_PLANNER with safety heuristic (right). Second row: Classical PRM planner (left) and SAFE_PRM_PLANNER (right). Third row: RRT-Connect planner (left) and SAFE_RRT-Connect planner (right). Fourth row: JT-RRT planner (left) and SAFE_JT-RRT planner (right).

profiles deviate less from the expected one. Table I shows some relevant parameters for all of the algorithms for the first scenario. Figure 9 shows how running times of the tested algorithms (except for those based on bubbles) are distributed within the set of 1000 runs. For a convenient representation of distributions, the box plots are used to report the lower quartile, the median, and the upper quartile (solid lines), and whiskers that quantify the dispersion of the data.[3] Clearly, there is no substantial difference between the safety-oriented planners and the original algorithms.

Figure 10 shows another scenario with the typical solution paths obtained by the tested algorithms.

As in the previous example, safety-oriented versions of the planning algorithms provide much safer paths. This is clearly supported by the averaged profiles of the danger field shown in Fig. 11.

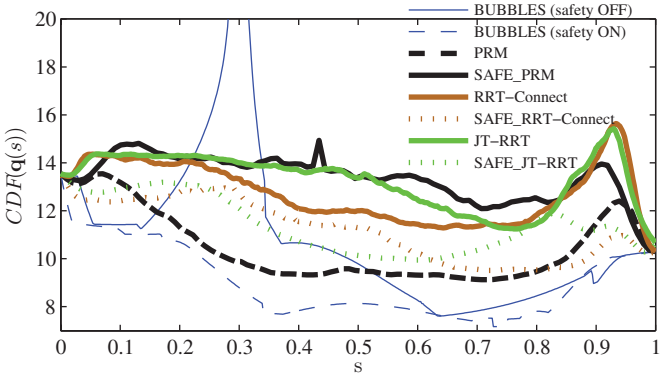

Fig. 11. (Colour online) Scenario 2: averaged danger field profiles vs. the path's natural parameter. The peak of the danger field profile for BUBBLE_PLANNER without safety heuristic function has a value of cca 35. It is not included for better depiction of the remaining profiles.

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

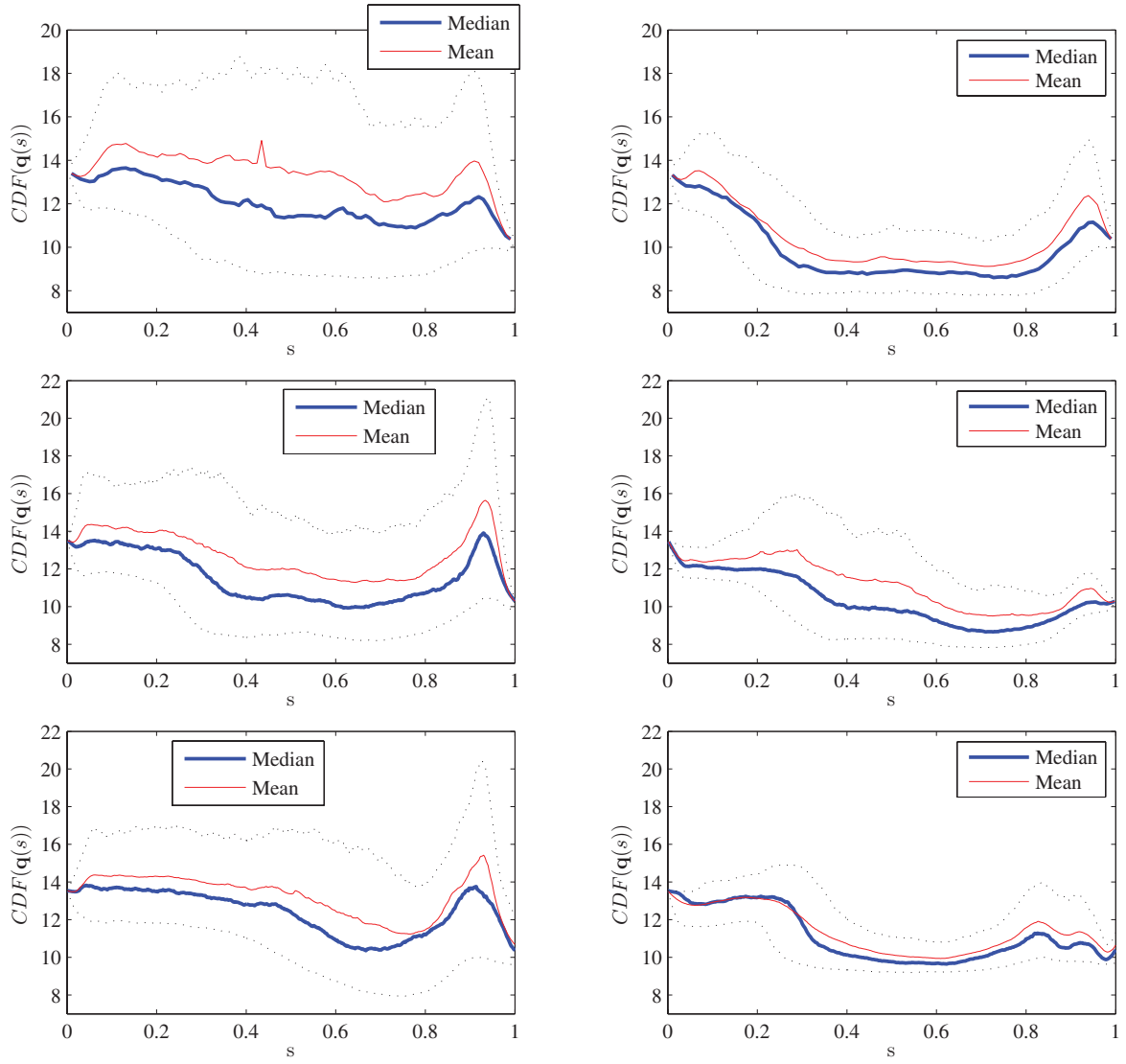

Fig. 12. (Colour online) Scenario 2: mean, median, 15th, and 85th percentiles of the danger field along the path's natural parameter. Top row: PRM (left) and SAFE_PRM (right). Middle row: RRT-Connect (left) and Safe_RRT-Connect (right). Bottom row: JT-RRT (left) and Safe_JT-RRT (right).

Figure 12 shows how danger field profiles deviate from the average one, while Table II shows some relevant numerical indicators for the second example.

Box plots in Fig. 13 give some insight about the distribution of running times for second scenario. It appears that, in average, safety-based planners need considerably less time to output a path. Moreover, the corresponding deviation is fairly smaller as well.

5.1. Discussion of results and future work directions

Apart from inferring that safety-oriented planners substantially enhance path quality in terms of safety, it is worth pointing out that this improvement is not dearly paid by noticeable deterioration of some other important aspects of the presented approach, such as running time. On the contrary, in most cases, safety-oriented algorithms need even less expected time to find the solution path, though they comprise additional computational burden—danger/safety evaluation. This can be partly explained by tendency of safety-oriented algorithms to explore less cluttered regions of C-space, which results in less nodes and less collision checks or distance computations needed to eventually complete the path (see Tables I and II). Some empirical evidence that supports this assertion can be also found in ref. [33].

It is also interesting to compare the performance of the safety-oriented algorithms. Based on Figs. 7 and 11, clearly the safe BUBBLE_PLANNER yielded the safest path in both scenarios. However, in terms of running time, it is outperformed by both SAFE_RRT-Connect and SAFE_JT-RRT algorithm in both example scenarios, particularly in the second one. Besides this, one important drawback of safe BUBBLE_PLANNER, that has been noticed within a broader set of simulation setups than actually presented in this paper, is its inconsistent performance (mostly in terms of running time) with respect to various test scenarios with comparable complexity. We presume that this is due to algorithm's sensitivity to parameters in the heuristic function. The equalization of performance toward the improvement could be achieved indeed, yet this would require particular tuning of the parameters for individual scenarios. Such

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

Table II. Scenario 2: some numerical results.

Algorithm	Avg. time (s)	Nodes	Coll. checks/ Dist. computations	DF computations
BUBBLES (safety OFF)	1.6640	474	2093	–
BUBBLES (safety ON)	1.4040	436	1778	454
PRM	2.3915	86	6599	–
SAFE_PRM	1.8768	90	6259	265
RRT-Connect	0.7296	54	2526	–
SAFE_RRT-Connect	0.6280	39	2427	84
JT-RRT	0.7221	127	2071	–
SAFE_JT-RRT	0.4267	37	1124	47

performance inconsistency and sensitivity to parameters has not been observed during the validation of remaining safety-oriented algorithms.

As for sampling-based safety-oriented planners covered in this paper, a slight advantage can be attributed to RRT-based algorithms over PRM-based one for several reasons. Considering current implementations, the running times are considerably lower. Furthermore, when dealing with the high-dimensional configuration space, SAFE_PRM_PLANNER needs a large number of samples to "populate" it. On the other hand, safety-oriented RRT-based planners inherit all the advantages from classical RRT algorithms, such as efficient and fast exploration of high-dimensional spaces. Some drawbacks attributed to classical RRTs, e.g., unsuccessfully coping with narrow passages problem, do not represent an issue here, since pathological scenarios with narrow passages are not expected in human–robot interaction applications. Among two proposed safety-oriented RRT planners, it is straightforward to choose the proper one by the way how the goal of the planner is defined. If the goal is defined in the workspace, than SAFE_JT-RRT algorithm should be used. However, if the goal is represented as a desired configuration, SAFE_RRT-CONNECT planner is a logical choice.

Further research will address the problem of robustness of bubble-based planner by improving the heuristic function in order to decrease the sensitivity to design parameters. It would be also interesting to use the slightly more elaborated algorithms that comprise a generic cost function, like those in refs. [4,20,47], to explicitly address the problem of safety. A problem where the goal is defined in the workspace will be also tackled by the safety-oriented modifications of the algorithms described in refs. [5,6,45]. Finally, an idea to further enhance the path safety in smoothing phase seems attractive as well.

6. Concluding Remarks

This paper presented several different approaches to obtaining safe collision-free paths for robotic manipulators. A deterministic approach to path planning for robotic manipulators is described first. The planner searches for a collision-free path via bubbles of free C-space while trading off the path length with a defined degree of danger. The algorithm is based on bidirectional A^*-search technique with a heuristic function that makes account of the danger assessment. The danger degree is estimated using the danger field. In addition, a sampling-based method for safe path planning is presented. It relies on the classical PRM context. The planning algorithm is based on the bidirectional search with a heuristic function that takes account of the safety measure. Hence, the planner strives for collision-free paths that are safe at the same time. Moreover, two safety-oriented modifications of RRT-based planners are proposed. The first is unidirectional RRT algorithm that uses Jacobian transpose (or pseudoinverse) to guide the tree growth towards the goal defined in the workspace. The second is an extension of the classical RRT-connect planner where the inputs to the algorithm are the start and the goal configurations that serve as the seeds for the tree's growth.

All the planners have been tested within a simulation study, where they have been compared with respect to their tendency toward safety, running time, and the deviation of path safety. Safety-oriented planners have consistently outperformed the

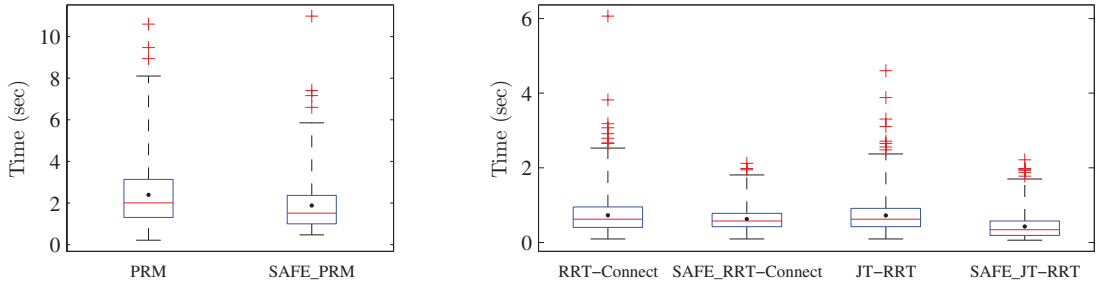

Fig. 13. (Colour online) Scenario 2: box plots for running times of considered algorithms. Mean value (·) and the outliers (+) are also indicated. Since the running times for PRM and SAFE_PRM are considerably higher in comparison with the remaining algorithms, they are shown separately for scaling purposes.

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press

original algorithms, indicating how a simple, straightforward extension have substantially improved the quality of solution paths.

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013—Challenge 2—Cognitive Systems, Interaction, Robotics, under grant agreement No. 230902 - ROSETTA. The authors thank Morten Strandberg for many useful discussions related to this work.

References

1. ANSI, *ANSI/RIA R15.06-1999: Industrial Robots and Robot Systems—Safety Requirements* (American National Standards Institute, New York, 1999).
2. ISO, *Robots for Industrial Environments—Safety Requirements—Part 1: Robot. ISO10218* (International Organization for Standardization, Geneva, 2006).
3. Y. Benjamini, "Opening the box of a boxplot," *Am. Stat.* **42**(4), 257–262 (1988).
4. D. Berenson, T. Siméon and S. S. Srinivasa, "Addressing Cost-Space Chasms in Manipulation Planning," *Proceedings of the IEEE International Conference on Robotics and Automation*, Shanghai, P.R. China (May 9–13, 2011) pp. 4561–4568.
5. D. Berenson, S. Srinivasa and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. Robot. Res.* **30**(1), 1435–1460 (2011).
6. D. Berenson, S. S. Srinivasa, D. Ferguson, A. Collet and J. J. Kuffner, "Manipulation Planning with Workspace Goal Regions," *Proceedings of the IEEE International Conference on Robotics and Automation*, Piscataway, NJ, USA (May 17–12, 2009) pp. 1397–1403.
7. D. Bertram, J. Kuffner, R. Dillmann and T. Asfour, "An Integrated Approach to Inverse Kinematics and Path Planning for Redundant Manipulators," *Proceedings of the IEEE International Conference on Robotics and Automation*, Orlando, FL, USA (May 15–19, 2006) pp. 1874–1879.
8. R. Bohlin and L. E. Kavraki, "Path Planning Using Lazy PRM," *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA, USA (Apr. 24–18, 2000) pp. 521–528.
9. O. Brock and O. Khatib, "Elastic strips: A framework for motion generation in human environments," *Int. J. Robot. Res.* **21**(12), 1031–1052 (2002).
10. J. W. Burdick, "On the Inverse Kinematics of Redundant Manipulators: Characterization of the Self-Motion Manifolds," *Proceedings of the IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, USA (May 14–19, 1989) pp. 264–270.
11. J. F. Canny, *The Complexity of Robot Motion Planning* (MIT Press, Cambridge, MA, 1988).
12. H. Choset, W. Burgard, S. Hutchinson, G. Kantor, L. E. Kavraki, K. Lynch and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation* (MIT Press, Cambridge, MA, 2005).
13. R. Geraerts and M. H. Overmars, "A Comparative Study of Probabilistic Roadmap Planners," *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*, Nice, France (Dec. 15–17, 2002) pp. 43–57.
14. R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *Int. J. Robot. Res.* **26**(8), 845–863 (2007).
15. S. Haddadin, A. Albu-Schäffer and G. Hirzinger, "Requirements for safe robots: Measurements, analysis and new insights," *Int. J. Robot. Res.* **28**(11–12), 1507–1527 (2009).
16. J. Heinzmann and A. Zelinsky, "Quantitative safety guarantees for physical human–robot interaction," *Int. J. Robot. Res.* **22**(7–8), 479–504 (2003).
17. D. Hsu, L. E. Kavraki, J.-C. Latombe, R. Motwani and S. Sorkin, "On Finding Narrow Passages with Probabilistic Roadmap Planners," *Proceedings of the Third International Workshop on the Algorithmic Foundations of Robotics*, Houston, TX, USA (Mar. 5–7, 1998) pages 141–153.
18. K. Ikuta, H. Ishii and M. Nokata, "Safety evaluation method of design and control for human-care robots," *Int. J. Robot. Res.* **22**(5), 281–298 (2003).
19. K. Ikuta, M. Nokata and H. Ishii, "General danger evaluation method for control strategy of human-care robot," *J. Robot. Soc. Japan* **19**(1), 81–90 (2001).
20. L. Jaillet, J. Cortes and T. Simeon, "Sampling-based path planning on configuration-space costmaps," *IEEE Trans. Robot.* **26**(4), 635–646 (2010).
21. H. Kaindl and G. Kainz, "Bidirectional heuristic search reconsidered," *J. Artif. Int. Res.* **7**(1), 283–317 (1997).
22. S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.* **30**(7), 846–894 (2011).
23. L. E. Kavraki, P. Svestka, J.-C. Latombe and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.* **12**(4), 566–580 (1996).
24. J. J. Kuffner Jr. and S. M. LaValle, "RRT-Connect: An Efficient Approach to Single-Query Path Planning," *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA, USA (Apr. 24–28, 2000) pp. 995–1001.
25. D. Kulic, Safety for Human–Robot Interaction *Ph.D Thesis* (Vancouver, Canada: The University of British Columbia, 2005).
26. D. Kulic and E. A. Croft, "Affective state estimation for human–robot interaction," *IEEE Trans. Robot.* **23**(5), 991–1000 (2007).
27. D. Kulić and E. A. Croft, "Safe planning for human–robot interaction," *J. Robot. Syst.* **22**(7), 383–396 (2005).
28. D. Kulic and E. A. Croft, "Real-time safety for human–robot interaction," *Robot. Auton. Syst.* **54**(1), 1–12 (2006).
29. B. Lacevic, Safe Motion Planning and Control for Robotic Manipulators *Ph.D Thesis* (Milan: Politecnico di Milano, 2011).
30. B. Lacevic and P. Rocco, "Kinetostatic Danger Field: Novel Safety Assessment for Human–Robot Interaction," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei (Oct. 18–22, 2010) pp. 2169–2174.
31. B. Lacevic and P. Rocco, "Sampling-Based Safe Path Planning for Robotic Manipulators," *IEEE International Conference on Emerging Technologies and Factory Automation*, Bilbao, Spain (Sep. 13–16, 2010) pp. 1–7.
32. B. Lacevic and P. Rocco, "Towards a Complete Safe Path Planning for Robotic Manipulators," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei (Oct. 18–22, 2010) pp. 5366–5371.
33. B. Lacevic, P. Rocco and M. Strandberg, "Safe Motion Planning for Articulated Robots Using RRTs," *XXIII International Symposium on Information, Communication and Automation Technologies - ICAT*, Sarajevo, Bosnia and Herzegovina (Oct. 27–29, 2011) pp. 1–7.
34. J.-C. Latombe, *Robot Motion Planning* (Kluwer Academic, Norwell, MA, 1991).
35. S. M. LaValle and J. J. Kuffner, "Rapidly-exploring Random Trees: Progress and Prospects," *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*, Hanover, NH, USA (Mar. 16–18, 2000).
36. S. M. LaValle, *Planning Algorithms* (Cambridge University Press, New York, 2006).
37. J. Mainprice, E. A. Sisbot, T. Simeon and R. Alami, "Planning Safe and Legible Hand-Over Motions for Human–Robot Interaction," *Proceedings of the IARP Workshop on*

Technical Challenges for Dependable Robots in Human Environments, Toulouse, France (Jun. 16–17, 2010) pp. 153–158.
38. P. Melchior, B. Orsoni, O. Lavialle, A. Poty and A. Oustaloup, "Consideration of obstacle danger level in path planning using A* and fast-marching optimisation: Comparative study," *Signal Process.* **83**(11), 2387–2396 (2003).
39. M. Nokata, K. Ikuta and H. Ishii, "Safety-optimizing method of human-care robot design and control," *IEEE International Conference on Robotics and Automation*, Washington, DC, USA (May 11–15, 2002), pp. 1991–1996.
40. S. Quinlan, Real-Time Modification of Collision-Free Paths *Ph.D Thesis* (Stanford, CA: Stanford University, 1995).
41. D. Roy, "Algorithmic path planning of static robots in three dimensions using configuration space metrics," *Robotica* **29**(2), 295–315 (2011).
42. G. Sánchez-Ante and J.-C. Latombe, "A Single-query Bi-directional Probabilistic Roadmap Planner with Lazy Collision Checking," *Proceedings of the 10th International Symposium on Robotics Research*, Lorne, Australia (Nov. 9–12, 2001) pp. 403–417.
43. D. Sent and M. H. Overmars, "Motion Planning in Environments with Dangerzones," *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea (May 21–26, 2001) pp. 1488–1493.
44. B. Siciliano, L. Sciavicco, L. Villani and G. Oriolo, *Robotics Modelling, Planning and Control* (Springer, London, 2009).
45. M. Stilman, "Global manipulation planning in robot joint space with task constraints," *IEEE Trans. Robot.* **26**(3), 576–584 (2010).
46. M. Strandberg, "Augmenting RRT-Planners with Local Trees," *Proceedings of the IEEE International Conference on Robotics and Automation*, New Orleans, LA, USA (Apr. 26–May 1, 2004) pp. 3258–3262.
47. C. Urmson and R. Simmons, "Approaches for Heuristically Biasing RRT Growth," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, NV, USA (Oct. 27–31, 2003).
48. M. Vande Weghe, D. Ferguson and S. S. Srinivasa, "Randomized Path Planning for Redundant Manipulators without Inverse Kinematics," *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*,Pittsburgh, PA, USA (Nov. 29–Dec. 1, 2007), pp. 477–482.
49. A. S. Vázquez and A. Adán, "Nonprobabilistic anytime algorithm for high-quality trajectories in high-dimensional spaces," *Robotica* **30**(2), 289–303 (2012).
50. W. A. Wolovich and H. Elliott, "A Computational Technique for Inverse Kinematics," *Proceedings of the IEEE International Conference on Decision and Control*, Las Vegas, NV, USA (Dec. 1984) pp. 1359–1363.
51. Y. Yang and O. Brock, "Elastic roadmaps–motion generation for autonomous mobile manipulation," *Auton. Robots* **28**(1), 113–130 (2010).
52. M. Zinn, O. Khatib, B. Roth and J. K. Salisbury, "Playing it safe [human-friendly robots]," *IEEE Robot. Autom. Mag.* **11**(2), 12–21 (2004).

https://doi.org/10.1017/S0263574713000143 Published online by Cambridge University Press