cRRT*: Planning loosely-coupled motions for multiple mobile robots

Jan Rosell and Raúl Suárez
Institute of Industrial and Control Engineering (IOC)
Universitat Politècnica de Catalunya (UPC) – Barcelona Tech
Barcelona, Spain, {jan.rosell, raul.suarez}@upc.edu

***Abstract*—The planning of collision-free paths of a team of mobile robots involves many degrees of freedom and therefore the use of sampling-based methods is a good alternative. Among them, the RRT* planner has been proposed to cope with optimization problems, and has been proven to be asymptotically optimal. Any optimization function can be defined, although optimization has been usually focused on the traveled distance or on safety, i.e. to find paths of minimum length or maximum clearance. Other constraints to be considered are related to the coordinate movements of the robots, including aspects like keeping a desired formation o having some similar behavior. In this paper we propose the use of an RRT* to optimize the traveled distance but subject to a coupled behavior between robots, i.e. it is desired that the robots behave as a group with similar or coordinated movements. To achieve so, a cost function has been defined that evaluates the alignment of the edges of the RRT* with the vectors that define the coupling between the motion directions of the robots. The method establishes a compromise between the independence required to avoid obstacles in a flexible way and the desired coupling to behave as a team. The method is illustrated with several examples.**

I. Introduction

Robot motion planning is a field that requires significant efforts to allow a robot autonomously planning its movements, and the scope of the problem ranges from static robot manipulators to mobile robots. The complexity of the problem increases significantly when a set of robots is considered, both when they just work independently in a common workspace, and collisions have to be avoided, and when they have to do a collaborative work to perform a task, and therefore other additional constraints must be satisfied. We deal in this work with a typical case within this problem, the motion planning of a set of mobile robots which have to perform independent tasks but with a common behavior.

A. Previous Works

Multi-robot path planning can be formulated in a centralized or decentralized way. Centralized approaches apply planning algorithms to the composite configuration space, e.g. [1], while decentralized approaches independently plan the path of each robot, either one at a time in a priority order, e.g. [2], or separately and then planning their velocities in order to avoid collisions, e.g. [3]. Other approaches lie in between by constraining the robots to travel on independent networks of paths (roadmaps) and then considering interactions between them and pursuing multiple-objective optimality [4]. Optimization has also been considered at the team level, like in [5] that after planning in a decoupled way uses D* in a coordination diagram to choose the velocity profile that minimizes a global performance index.

Some multi-robot approaches focus on motion coordination with the aim of moving the robots according to some constraints on the team as a whole, like formation keeping, flocking or target tracking, e.g. [6], [7]. The particular case of flocking can be viewed as a loose subcase of the formation control problem, requiring the team of robots to move together along some path, but without setting requirements for the specific robots, i.e. loosely coupling their degrees of freedom (DOF). The coupling of motions was also exploited in the planning of mechanical hand motions [8], with the aim of reducing the complexity (by reducing the number of DOF and therefore the dimension of the planning space) as well as of obtaining human-like motions (by defining couplings for the mechanical hands equivalent to those of the human hands).

Centralized approaches to the multi-robot path planning problem make sense nowadays due to sampling-based techniques, that have demonstrated to be efficient for motion planning problems involving a high number of DOF. Among them, the Rapidly-exploring Random Tree (RRT) [9] and its variants are focused on single query problems. The basic idea of an RRT is to build a tree of feasible motions, rooted at the initial configuration, by iteratively sampling a random configuration ($\boldsymbol{q}_{rand}$), searching the node of the tree nearest to it ($\boldsymbol{q}_{near}$), and moving an small amount from $\boldsymbol{q}_{near}$ towards a new configuration ($\boldsymbol{q}_{new}$) in the direction of $\boldsymbol{q}_{rand}$. If the path connecting $\boldsymbol{q}_{near}$ and $\boldsymbol{q}_{new}$ is collision-free then it is added as an edge of the tree.

Recently, an asymptotically optimal variant called RRT* has been proposed [10]. In the RRT* algorithm, once a configuration $\boldsymbol{q}_{new}$ has been computed as in the RRT case, it is not directly connected to $\boldsymbol{q}_{near}$ but to the node (among a given set of neighbors) that minimizes the cost to reach $\boldsymbol{q}_{new}$. Then, RRT* checks whether each neighbor node can be reached, through $\boldsymbol{q}_{new}$, with a cost smaller than its current one and, if so, rewires the edges of the tree.

This work was partially supported by the Spanish Government through the projects DPI2010-15446, DPI2011-22471 and DPI2013-40882-P.

B. Problem Statement and Solution Overview

Consider the motion planing problem of a set of m mobile robots R_i, $i = 1, ..., m$, each one with n degrees of freedom, that have to be moved from initial configurations $\boldsymbol{q}_o^i$ to goal configurations $\boldsymbol{q}_g^i$. This means going from an initial configuration $(\boldsymbol{q}_o^1, \ldots, \boldsymbol{q}_o^i, \ldots, \boldsymbol{q}_o^m)^T$ to a goal configuration $(\boldsymbol{q}_g^1, \ldots, \boldsymbol{q}_g^i, \ldots, \boldsymbol{q}_g^m)^T$ in a composite configuration space $\mathcal{C}$ of dimension mn.

In this work we seek for a solution path in $\mathcal{C}$ to move the robots in a coordinated way by making their motions as coupled as possible according to a predefined desired behaviour, like, for instance, establishing a correlation between the corresponding DOF of each robot. This is not a hard requirement to be met but a desired behavior, and how much the robot motions are uncoupled can be considered as a cost to be minimized if an optimization-based planning approach is used.

The main contributions of this work is the use of a new optimization cost function to be applied in an RRT* planner working in the composite configuration space $\mathcal{C}$. Besides the usual traveled distance, this cost function includes also the robot couplings such that they all tend to move in a similar fashion while they look for their respective goal configurations.

II. Proposed Optimization Cost Function

In optimization-based planners, like RRT*, an optimization cost function is necessary to evaluate the cost of a motion. This section presents the optimization function proposed for the cost evaluation of a motion along an edge of the RRT*.

Let $\boldsymbol{e}$ be an edge of the RRT* representing a motion (displacement) along the rectilinear segment connecting two configurations $\boldsymbol{q}_a, \boldsymbol{q}_b \in \mathcal{C}$; $\boldsymbol{e}$ can be expressed as a mn-dimensional vector

$$\boldsymbol{e} = \boldsymbol{q}_b - \boldsymbol{q}_a = (e_{11}, \ldots, e_{1n}, \ldots\ldots, e_{m1}, \ldots, e_{mn})^T \quad (1)$$

The proposed cost of a motion along an edge $\boldsymbol{e}$ is defined as a weighted sum of three components, i.e.

$$C = w_d C_d + w_c C_c + w_s C_s, \quad (2)$$

where $w_d, w_c, w_s \in \mathbb{R}^+$ are weighing coefficients and C_d, C_c, and C_s are the three components respectively related with:

- *Traveled Distance*: C_d measures the length of the edge $\boldsymbol{e}$, with the aim of obtaining paths as short as possible.
- *Robot Coupling*: C_c measures how much the edge $\boldsymbol{e}$ is aligned with the directions in $\mathcal{C}$ defining the desired coupling, with the aim of obtaining paths with coupled robot movements.
- *Deviation*: C_s measures how much a motion in $\mathcal{C}$ is aligned with the previous one, with the aim of minimizing the path deviations.

The objective is the minimization of the total cost C of a path defined in the RRT* by a sequence of rectilinear motions along a sequence of edges $\boldsymbol{e}$. The following subsections describe how the components C_d, C_c, and C_s are computed. The weighing coefficients w_d, w_c and w_s are empirically fixed according to the desired behavior.

A. Traveled Distance Cost C_d

The traveled distance cost C_d is directly computed as the Euclidean distance in $\mathcal{C}$ between two consecutive nodes in the RRT*, i.e. the length of an edge $\boldsymbol{e}$ given by:

$$C_d = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} e_{ij}^2}. \quad (3)$$

B. Robot Coupling Cost C_c

It is desired that the robot motion directions have a positive correlation all along the robot paths. This requirement can be modeled in $\mathcal{C}$ by defining directions that represent the desired correlations among the degrees of freedom of the robots and, then, the cost function must indicate how close to these directions is the movement direction in $\mathcal{C}$.

A particular global behaviour is obtained when the DOF of the robots are correlated one-to-one, i.e. establishing that all the robots in the team tend to move in the positive or all in the negative sense of their corresponding degrees of freedom. The correlation directions are indicated by vectors in $\mathcal{C}$, which define a subspace of $\mathcal{C}$. The proposed cost function to measure how much the movement along an edge $\boldsymbol{e}$ of the RRT* fits the desired behaviour is the ratio between the module of $\boldsymbol{e}$ and the module of the projection of $\boldsymbol{e}$ onto the subspace defined by the correlation vectors.

In order to compute the robot coupling cost C_c, let:

- $\boldsymbol{u}_{i,j} \in \mathcal{C}_i$, with $i = 1, \ldots, m;\ j = 1, \ldots, n$, be unitary vectors of dimension n with the j component set to 1 and the other $n-1$ components set to 0. For instance, for $n = 2$, $\boldsymbol{u}_{i,1} = (1, 0)^T$ and $\boldsymbol{u}_{i,2} = (0, 1)^T$.
- $\boldsymbol{u}_j \in \mathcal{C}$, with $j = 1, \ldots, n$, be unitary vectors of dimension mn defined as:

$$\boldsymbol{u}_j = \frac{1}{\sqrt{m}}(\boldsymbol{u}_{1,j}^T, \cdots, \boldsymbol{u}_{m,j}^T)^T \quad (4)$$

  For instance, for $m = 3$ robots with $n = 2$ DOF, $\boldsymbol{u}_1 = \frac{1}{\sqrt{3}}(1, 0, 1, 0, 1, 0)^T$ and $\boldsymbol{u}_2 = \frac{1}{\sqrt{3}}(0, 1, 0, 1, 0, 1)^T$, which means that the movements of the three robots along the axis x and y are respectively coupled. Note that $\boldsymbol{u}_{1,j}$ are unitary vectors, thus the square root in the denominator of eq. (4) makes $\boldsymbol{u}_j$ to be also unitary.
- $\boldsymbol{U}$ be a matrix of dimension $mn \times n$, defined as:

$$\boldsymbol{U} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n] \quad (5)$$

  Matrix $\boldsymbol{U}$ represents the couplings between the DOF of the robots determined by vectors $\boldsymbol{u}_j$. For instance, following the example above for $m = 3$ and $n = 2$:

$$\boldsymbol{U} = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (6)$$

- $\boldsymbol{e_U}$ be the vector of dimension mn:

$$\boldsymbol{e_U} = \sum_{j=1}^{n} (\boldsymbol{u}_j^T \cdot \boldsymbol{e}^T)\boldsymbol{u}_j, \tag{7}$$

Note that the relation $|\boldsymbol{e_U}| \leq |\boldsymbol{e}|$ is always true.

Now, the cost C_c is defined as the alignment of $\boldsymbol{e}$ with respect to the directions determined by the columns of matrix $\boldsymbol{U}$ as:

$$C_c = \min\left(\frac{|\boldsymbol{e}|}{|\boldsymbol{e_U}|}, C_c^{max}\right) \tag{8}$$

where C_c^{max} is a predefined saturation value that avoids useless large values of C_c when $\boldsymbol{e_U} \to 0$.

The cost C_c satisfies $C_c \in [1, C_c^{max}]$, and the more aligned with the coupling directions an edge $\boldsymbol{e}$ is, the smaller its cost. If $\boldsymbol{e}$ belongs to the subspace spanned by the columns of $\boldsymbol{U}$ then the cost equals 1, otherwise it is inversely proportional to projection of $\boldsymbol{e}$ onto this subspace, being set to the saturation maximum value C_c^{max} when the module of this projection is below a given threshold and therefore the inverse is too large and useless.

C. Deviation Cost C_s

The cost C_s measures the deviation that an edge $\boldsymbol{e}$ introduces with respect to the previous edge in a path, called the parent edge $\boldsymbol{e}_p$, and evaluates whether this alignment (or misalignment) is maintained for long or not, i.e. the cost C_s is proportional to both the angle between two consecutive edges $\boldsymbol{e}_p$ and $\boldsymbol{e}$, represented as $\widehat{\boldsymbol{e}\boldsymbol{e}_p}$, and to the length of $\boldsymbol{e}$:

$$C_s = |\boldsymbol{e}| \;\; \widehat{\boldsymbol{e}\boldsymbol{e}_p} \tag{9}$$

For motions along edges starting at the initial configuration, C_s is set to zero since in this case there is no parent edge.

D. Computation of the cost function C

Let Parent($\boldsymbol{q}$) be the function that returns the parent node of $\boldsymbol{q}$ in the tree. The procedure to compute the cost C of an edge $\boldsymbol{e}$ in the RRT* is described in Algorithm 1.

Algorithm 1 Cost function

Input: Configurations $\boldsymbol{q}_1, \boldsymbol{q}_2 \in \mathcal{C}$
  Weights $\mathbf{w} = [w_d, w_c, w_s]$
  Coupling matrix $\boldsymbol{U}$
Output: The cost of the edge from $\boldsymbol{q}_1$ to $\boldsymbol{q}_2$
  $\boldsymbol{e} = \boldsymbol{q}_2 - \boldsymbol{q}_1$
  $C_d = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} e_{ij}^2}$
  $\boldsymbol{e}_p = \boldsymbol{q}_1 -$ Parent($\boldsymbol{q}_1$)
  $\boldsymbol{e_U} = \sum_{j=1}^{n} (\boldsymbol{u}_j^T \cdot \boldsymbol{e}^T)\boldsymbol{u}_j,$
  $C_c = \min(\frac{|\boldsymbol{e}|}{|\boldsymbol{e_U}|}, C_c^{max})$
  $C_s = |\boldsymbol{e}| \;\; \widehat{\boldsymbol{e}\boldsymbol{e}_p}$
  return $C = w_d C_d + w_c C_c + w_s C_s$

III. Planning Algorithm cRRT*

The proposed motion planning algorithm, called cRRT* (from coupled-RRT*), is based on the standard RRT* [10] but particularized to include the cost function C described above. The proposed procedure for the cRRT* is described in Algorithm 2, where the following functions are used:

- Sample($\mathcal{C}, \boldsymbol{q}_{goal}, P_{goal}$): Returns with probability $(1 - P_{goal})$ a random sample from $\mathcal{C}$ using a uniform distribution or, with probability P_{goal}, returns $\boldsymbol{q}_{goal}$.
- Nearest($V, \boldsymbol{q}$): Returns the closest node to configuration $\boldsymbol{q}$ from the set V of nodes of the tree.
- Steer($\boldsymbol{q}_1, \boldsymbol{q}_2, \epsilon$): Returns a new configuration $\boldsymbol{q}$ obtained by moving from $\boldsymbol{q}_1$ an small amount ϵ towards $\boldsymbol{q}_2$, or the final configuration $\boldsymbol{q}_2$ if the distance between $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ is smaller than ϵ.
- k-Near($V, \boldsymbol{q}$): Returns the k nearest neighbors of $\boldsymbol{q}$ from the set V of nodes of the tree (the value of k depends on the dimension of the configuration space and on the number of vertices of the tree, as detailed in [10]).
- Path($\boldsymbol{q}$): Returns a piece-wise rectilinear path, composed of edges of the tree, that connects the root node $\boldsymbol{q}_{init}$ 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.
- EdgeCost($\boldsymbol{q}_1, \boldsymbol{q}_2, \mathrm{w}, \boldsymbol{U}$): Evaluates the cost C of the edge connecting $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ as described in Section II-D.
- Cost($\boldsymbol{q}$): Returns the cost C from $\boldsymbol{q}_{\text{init}}$ to $\boldsymbol{q}$ by computing the cost of the edges of a Path($\boldsymbol{q}$).

IV. Approach Validation

The proposed approach was implemented and applied with good results. Next subsections describe details of the implementation and present some examples to illustrate the performance.

A. Implementation Issues

The proposal has been implemented within the Kautham planning and simulation environment [11], whose main planning core is based on the Open Motion Planning Library (OMPL) [12]. The proposed approach has been coded as a class derived from the OMPL OptimizationObjective class, which is used with the OMPL RRT* planner. The available graphical output includes projections of the configuration space on the subspace defined by the translational DOF of each robot.

B. Conceptual Example

This is a simple artificial example set to illustrate the concepts in a 2D configuration space $\mathcal{C}$, and therefore $\mathcal{C}$ can be completely represented in a graphical way. Consider two robots with a single DOF each one, and it is desired a coupling of the two resulting DOF such that they both move simultaneously with the same sense (either positive

Algorithm 2 cRRT*: RRT* algorithm with the cost function described in Section II.

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 ← {q_init}; E ← ∅;
for i = 0 to n do
  q_rand ← Sample(C, q_goal, P_goal)
  q_nearest ← Nearest(q_rand)
  q_new ← Steer(q_nearest, q_rand, ε)
  if CollisionFree(q_nearest, q_new) then
    Q_near ← k-Near(V, q_new)
    V ← V ∪ {q_new}
    q_min = q_nearest
    c_min = Cost(q_nearest) + EdgeCost(q_nearest, q_new, w, U)
    for all q ∈ Q_near do
      if CollisionFree(q, q_new) ∧
      Cost(q) + EdgeCost(q, q_new, w, U) < c_min then
        q_min ← q
        c_min ← Cost(q) + EdgeCost(q, q_new, w, U)
      end if
    end for
    E ← E ∪ {q_min, q_new} //Connect along a minimum-cost path
    for all q ∈ Q_near do
      if collisionFree(q_new, q) ∧
      Cost(q_new) + EdgeCost(q_new, q, w, U) < Cost(q) then
        E ← (E\{Parent(q), q}) ∪ {q_new, q} //Rewire
      end if
    end for
    if q_new = q_goal then
      return Path(q_goal)
    end if
  end if
end for
return ∅
```

or negative). In this case $\boldsymbol{u}_{i,j}$ are one-dimensional vectors: $\boldsymbol{u}_{1,1} = \boldsymbol{u}_{2,1} = (1)$, and therefore $\boldsymbol{u}_1 = \frac{1}{\sqrt{2}}(1,1)^T$, and $\boldsymbol{U} = [\boldsymbol{u}_1] = \frac{1}{\sqrt{2}}(1,1)^T$.

Both robots are initially located at the middle of their ranges, and one of them has to move to the positive extreme while the other has to move to the negative one, in an obstacle-free environment. This implies going from the center to the top left corner in $\mathcal{C}$ (see Fig. 1). Note that going in a straight direction from the initial to the desired configuration implies a movement in $\mathcal{C}$ completely orthogonal to the desired coupling direction given by $\boldsymbol{U}$, i.e. the most undesired moving direction. Fig. 1 shows the solution using an RRT (top row), an RRT* optimizing the traveled distance (middle row), and the proposed cRRT* optimizing the alignment with the direction defined by $\boldsymbol{U}$ (bottom row).

Note that, as expected, the first two rows present solutions with shorter paths, but completely against the desired coupled movements since they are almost orthogonal to the coupling direction in $\mathcal{C}$ (the RRT* solution becomes even shorter as the number of samples increases, as reported in [10]). On the other hand, the solution obtained with the cRRT* is a path with moving directions closer to the coupling direction, being the worst deviation of about only $\pi/4$ (bottom right).

It is worth remarking that this example was presented just for illustrative purpose and that it does not intend to represent any practical application.

C. Application Examples

Consider first the case of two mobile robots with two translational degrees of freedom moved in a unitary square workspace with one rectangular obstacle as illustrated in Fig. 2. It is desired a coupling between the corresponding x and y DOF of each robot. The robots are initially located as shown in the snapshots on the left, and their goal positions are those shown in the snapshots on the right. The top row shows intermediate positions of the resulting paths of the robots when using the RRT* planner optimizing only distances, and the bottom row shows the solution found using the proposed cRRT* planner with $P_{goal} = 0.05$, $\epsilon = 0.1$, weights $\mathbf{w} = [0.1, 1.0, 1.0]$ and the coupling matrix:

$$\boldsymbol{U} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{10}$$

It can be appreciated that in the first case each robot moves towards its goal without caring about the movements of the other robot, while in the second case, as a result of the coupled movements, the robots move as a team choosing the same side of the obstacle to advance towards their goals. The robot paths are also illustrated in Fig. 3 using 2D projections of the 4-dimensional configuration space where each subfigure shows the projection of the tree and of the final paths.

Consider now the case of two mobile robots with three translational degrees of freedom in a 3D workspace with one obstacle, as illustrated in Fig. 4. It is desired a coupling between the corresponding x, y and z DOF of each robot. The robots must go from the configuration shown on the snapshots on the left to the one on the right. The top row shows intermediate positions of the resulting paths of the robots when using the RRT* planner optimizing only distances, and the bottom row shows the solution found using the proposed cRRT* planner with the same parameter values of the previous example, and the coupling matrix:

$$\boldsymbol{U} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{11}$$

It can be appreciated that in the first case each robot moves independently, the robot in the bottom follows an almost straight line below the obstacle and the one in the top moves above the obstacle. On the contrary, in the second case the robots go around the obstacle near one of its corners following similar motions: up and to the left in the first part and then, once the obstacle has been avoided, down and to the right.

Finally, a motion planning example for four mobile robots with two DOF is illustrated in Fig. 5 and Fig. 6. Again, a

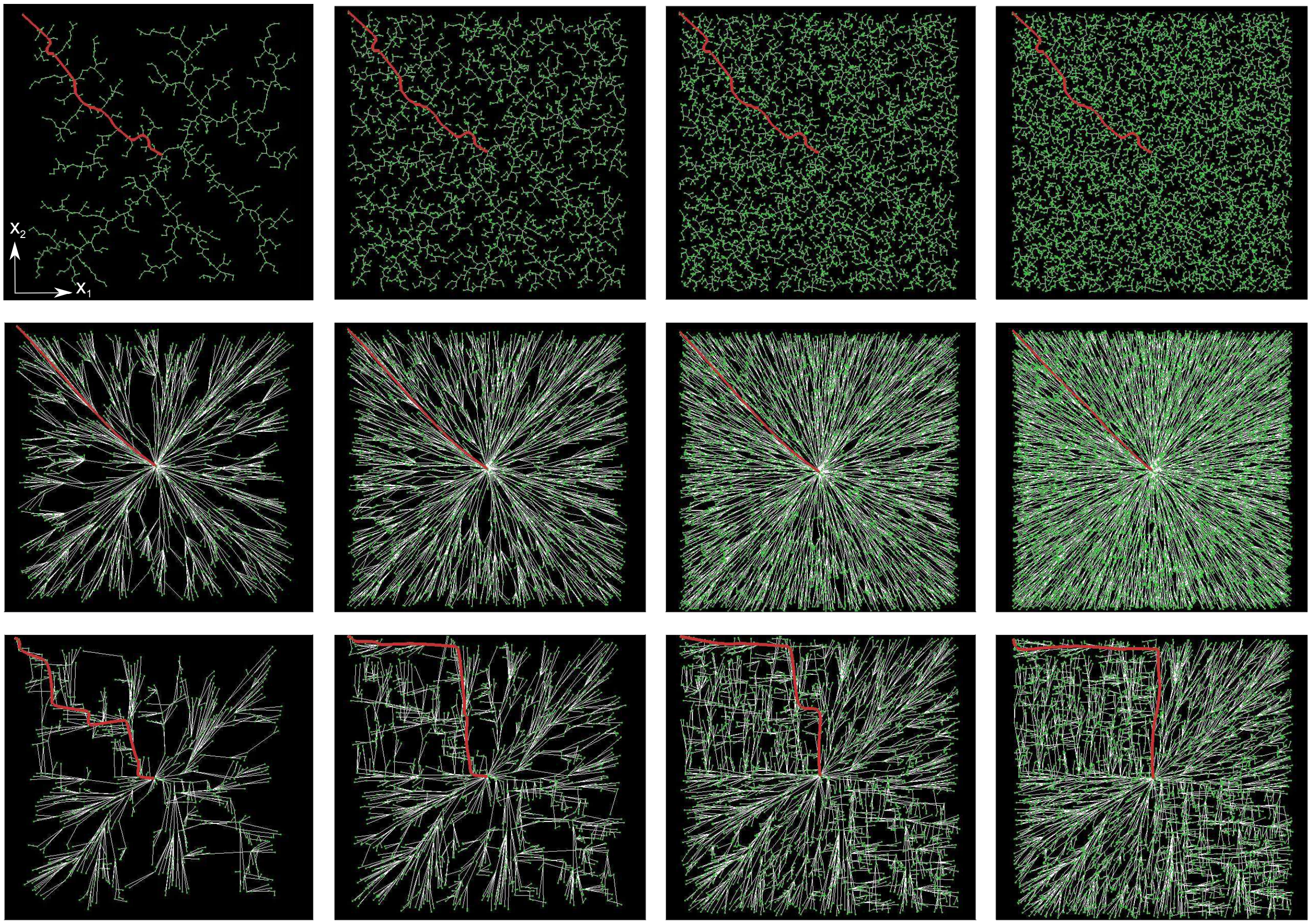


Fig. 1: Solution path (in red) to a query problem in a 2D configuration space without obstacles using an increasing number of samples and the RRT algorithm (top row), the RRT* (middle row) and the cRRT* (bottom row).

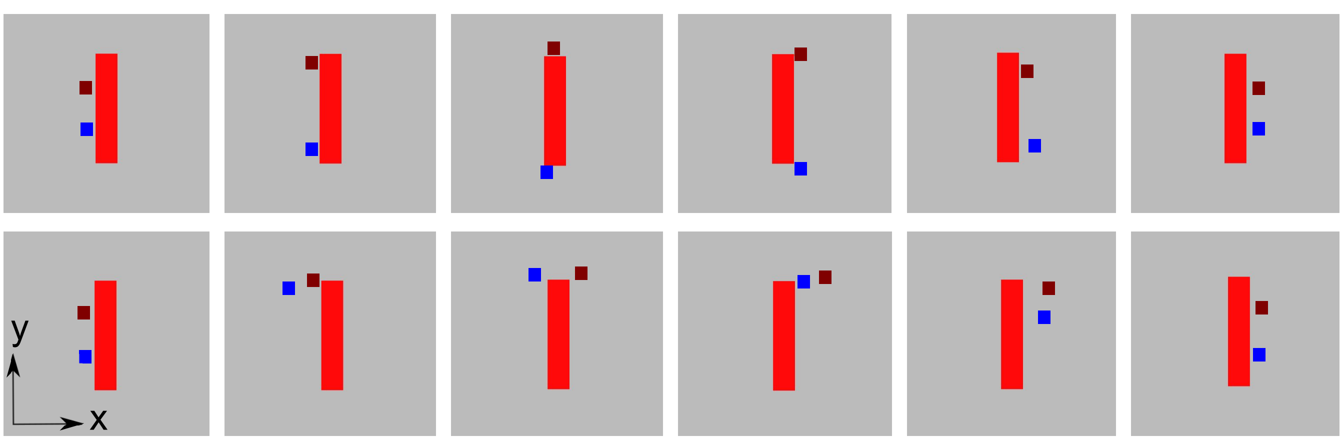
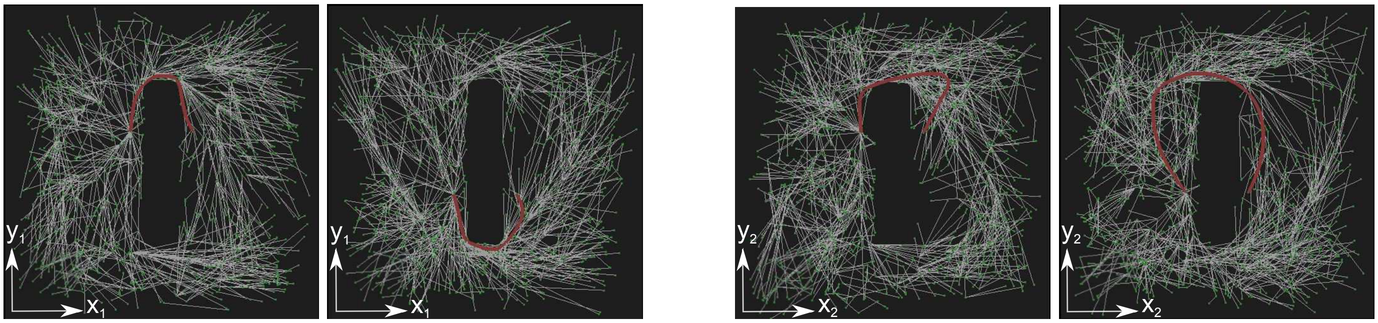


Fig. 2: Example of two robots with two DOF: Solution paths obtained with a RRT* optimizing only the traveled distance (top row) and with the cRRT* (bottom row).



R_1 R_2

Using RRT*

R_1 R_2

Using cRRT*

Fig. 3: Example of two robots with two DOF: 2D projections of the 4-dimensional configuration space showing in red the trajectories of each robot R_i found with an RRT* optimizing the traveled distance (left group) and with the cRRT* (right group).

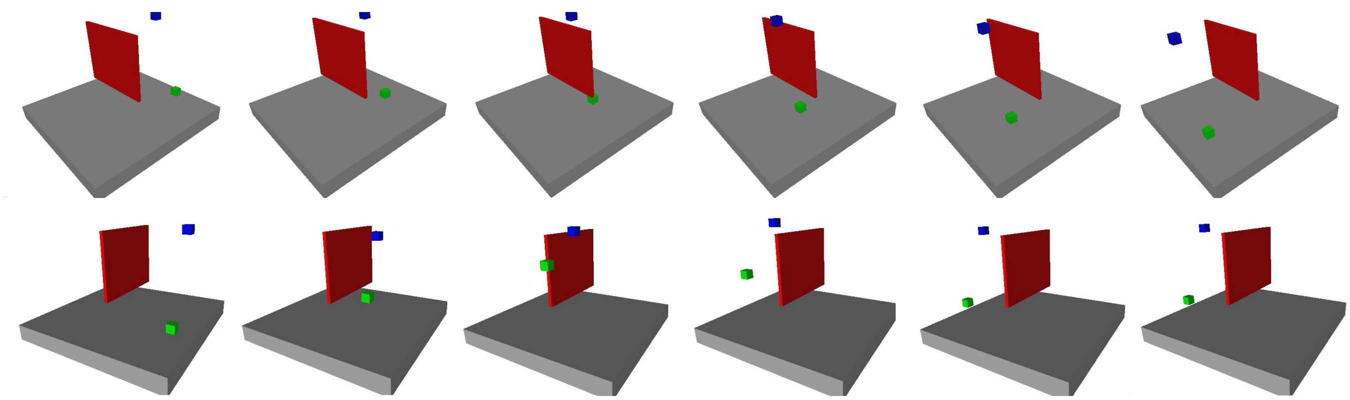


Fig. 4: Example of two robots with three DOF: Solution paths obtained with a RRT* optimizing only the traveled distance (top) and with the cRRT* (bottom).

coupling between the corresponding x and y DOF of each robot is desired. The top row shows intermediate positions of the resulting paths of the robots when using the RRT* planner optimizing only distances, and the bottom row shows the solution found using the proposed cRRT* planner with the same parameter values of the previous examples, and the coupling matrix:

$$\boldsymbol{U} = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{12}$$

It can be appreciated that using the RRT* shorter paths are obtained but using the cRRT* the robots move more as a team by making the turn to cross the obstacles in unison.

V. CONCLUSIONS

The paper has presented a centralized motion planning method for a team of robots based on the RRT*, an efficient asymptotically optimal sampling-based algorithm. The RRT* has been tuned with an optimization function to minimize the traveled distance, maximize the desired coupling of motions between robots and minimize the deviations of the path. The approach, illustrated in a simulation environment with several examples, generates paths that make the robots move coordinately as a flock of birds, with good compromise between the independence required to obtain collision-free paths and the coupling desired to behave as a team.

Two main topics are considered as future work that could improve the approach. On the one side, the determination of criteria to automatically fix the weighing coefficients w_d, w_c and w_s depending on the desired behavior and the task to be executed. On the other hand, in the presented approach the couplings among certain DOF are fixed in a binary way (0 or 1 in the components of vectors $\boldsymbol{u}_{i,j}$), analyzing the effect and possible consideration of partial correlations in the cost functions also deserves some future work.

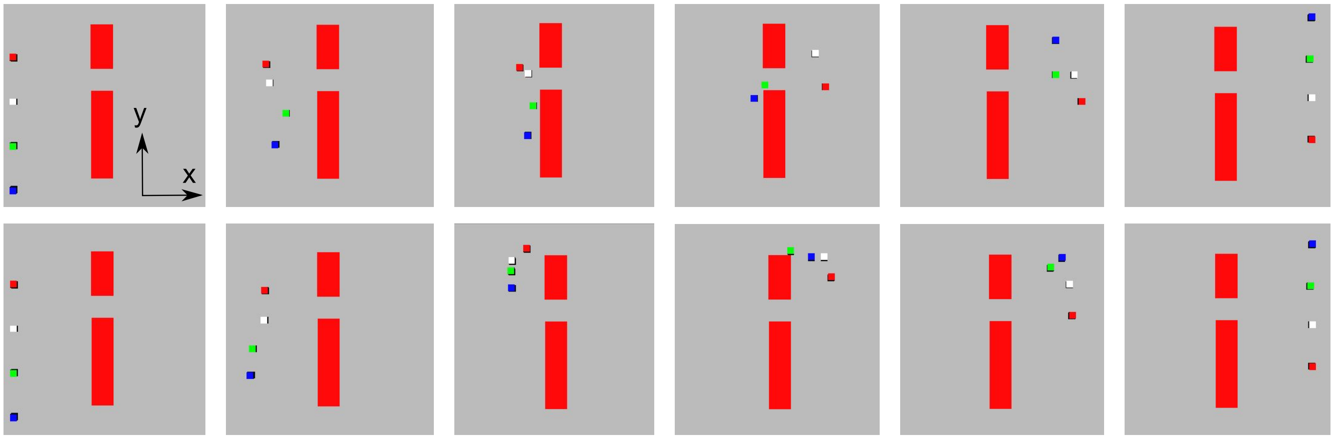


Fig. 5: Solutions of the RRT* (top) and cRRT* (bottom) for the four robots example.

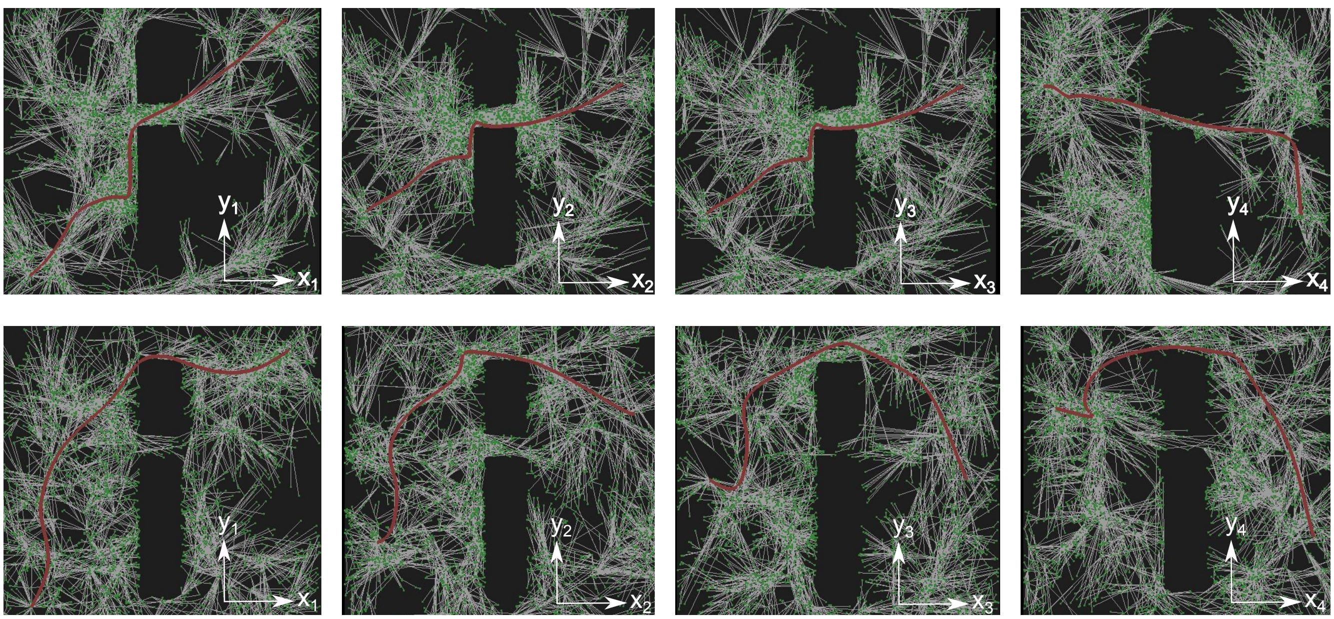


Fig. 6: 2D projections of the eight-dimensional configuration space showing in red the trajectories of each robot found with an RRT* optimizing the traveled distance (top row) and with the cRRT* algorithm (bottom row).

REFERENCES

[1] G. Sánchez and J.-C. Latombe, "On delaying collision checking in prm planning: Application to multi-robot coordination," *The International Journal of Robotics Research*, vol. 21, pp. 5–26, 2002.

[2] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," in *Proc. of the IEEE Int. Conf on Robotics and Automation*, vol. 3, Apr 1986, pp. 1419–1424.

[3] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition." *The International Journal of Robotics Research*, vol. 5, pp. 72–89, 1986.

[4] S. LaValle and S. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 6, pp. 912–925, Dec 1998.

[5] Y. Guo and L. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *Proc. of the IEEE Int. Conf on Robotics and Automation*, vol. 3, 2002, pp. 2612–2619.

[6] O. B. Bayazit, J. ming Lien, and N. M. Amato, "Swarming behavior using probabilistic roadmap techniques," in *Lecture Notes in Computer Science*, 2005, pp. 2005–3342.

[7] C.-C. Lin, P.-Y. Hsiao, and K.-C. Chen, "A motion planning of swarm robots using genetic algorithm," in *Broadband, Wireless Computing, Communication and Applications (BWCCA), 2010 International Conference on*, Nov 2010, pp. 538–543.

[8] J. Rosell, R. Suárez, C. Rosales, and A. Pérez, "Autonomous motion planning of a hand-arm robotic system based on captured human-like hand postures," *Autonomous Robots*, vol. 31, no. 1, pp. 87–102, 2011.

[9] 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.

[10] 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.

[11] 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.

[12] 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.