Acta Polytechnica **00**(0):1–5, 0000

PREPRINT 2015-07-14

MULTI ROBOT OPTIMAL TRAJECTORY GENERATION

José Magno Mendes Filho[a,b], Eric Lucet[a,*]

[a] *CEA, LIST, Interactive Robotics Laboratory, Gif-sur-Yvette, F-91191, France*
[b] *ENSTA Paristech, Unité d'Informatique et d'Ingénierie des Systèmes, 828 bd des Marechaux, 91762, France*
[*] corresponding author: eric.lucet@cea.fr

Abstract. Abstract.

Keywords: multi robot path planning, mobile robot.

1. Introduction

The control of mobile robots is a long-standing subject of research in the iterative robotics domain. As many modern companies started to adopt mobile robot systems such as teams of autonomous forklift trucks [1] to improve their commercial performance, this subject becomes increasingly important and complex. Especially when those robots are executing theirs tasks in human-occupied environment.

One basic requirement for such robotic systems is the capacity of generating a trajectory that connects two arbitrary configurations. For that, different constraints must be taken into account to ideally find an appropriated trajectory, in particular:

- kinematic and dynamic constraints;
- geometric constraints;
- constraints associated with uncertainties about the current state of world and the outcome of actions.

The first constraints derive directly from the mobile robot architecture implying in nonholonomic constraints for many types of mobile robots. Geometric constraints result from the impossibility of the robots to assume some specific configurations due to the presence of obstacles or environment bounds. In turn, uncertainty constraints come from the impossibility of the robots to have an absolute and complete perception of the world (including theirs own states) as well as the outcome of non-stochastic events.

We are particularly interest in solving the problem of dynamic planning a trajectory for a team of nonholonomic mobile robots in an partially known environment occupied by static obstacles being near optimal with respect to the execution time (time spent from going from initial to final configurations).

In recent years, a great amount of work towards collision-free trajectory planning has been proposed.

Some work has been done towards analytic methods for solving the problem for certain classes of systems ([]). However [?] shows that analytic methods are inapplicable for nonholonomic systems in presence of obstacles.

Cell decomposition methods as presented in [?] have the downside of requiring a structured configuration space and an *a priori* model of its connectivity. Besides, the cell decomposition reduces the space of admissible solutions. TODO verify/understand.

Initially proposed in [?], the vector field obstacle avoidance method was improve along time for treating problems such as oscillation of the solution for narrow passages. This method does not present a near optimal generated trajectory.

Elastic band approach initially proposed by [?] and extend to mobile manipulators in [? ?] uses approximations of the trajectory shape (combinations of arcs and straight lines) limiting the space of solutions and make this method inappropriate for really constrained environments.

The dynamic window approach [?] can handle trajectory planning for robots at elevated speeds and in presence of obstacles but is not flexible enough to be extended to a multi robot system.

In this paper, we focus on the development of a motion planning algorithm. This algorithm finds collision-free trajectories for a multi robot system in presence of static obstacles which are perceived by the robots as they evolve in the environment. The dynamic trajectories found are near optimal with respect to the total time spend going from the initial configuration to the final one. Besides, this algorithm uses a decentralized approach making the system more robust to communication outages and individual robot failure than compared to a centralized approach. Identified drawbacks are the dependence on several parameters for achieving real-time performance and good solution optimality, and not being able to handle dynamic obstacles as it is.

This algorithm is based mainly on the work done in [2] but we made changes in particular to respect a precise final configuration for the multi robot system and about how to set parameters for solving the nonlinear programming problem (NLP).

This paper is structured as follows: The second section presents a trajectory planning algorithm that solves the problem of one robot going from an initial configuration to a final one in the presence of static obstacles. The third section extends the method presented in the second section so a collision-free trajectory that maintains the communication link between the robots in the team can be computed. The forth section is dedicated to the results found by this method and the analysis of the computation time and solution quality and how they are impacted by the algorithm parameters. The fifth section presents the comparison of this approach to another one presented in []. Finally, in section six we present our conclusions and perspectives.

2. Problem Formulation

2.1. Assumptions

In the development of this approach the following assumptions were made:

(1.) The team of robots consists of a set $\mathcal{R}$ of B nonholonomic mobile robots;

(2.) A robot (denoted $R_b,\ R_b \in \mathcal{R},\ b \in \{0,\ldots,B-1\}$) is geometrically represented by a circle of radius ρ_b;

(3.) All obstacles in the environment are considered static. They can be represented by a set $\mathcal{O}$ of M static obstacles;

(4.) An obstacle (denoted O_m, $O_m \in \mathcal{O}$, $m \in \{0,\ \ldots, M-1\}$) is geometrically represented either as circle or as a convex polygon;

(5.) For a given instant $t_k \in [t_{init},\ t_{final}]$, any obstacle O_m closer to the geometric center of the robot R_b to the having its geometric center within the detection radius $d_{b,sen}$ from the geometric center of the robot R_b is considered detected by the robot R_b. Thus, this obstacle is part of the set $\mathcal{D}$ ($\mathcal{D} \subset \mathcal{O}$) of detected obstacles;

(6.) A robot has precise knowledge of the position and geometric representation of a detected obstacle;

(7.) A robot can access information about any robot in the team using a wireless communication link.

(8.) Latency, communication outages and other problems associated to the communication between robots in the team are neglected;

(9.) Dynamics was neglected.

(10.) The input of a mobile robot R_b is limited.

2.2. Constraints and cost functions

After loosely defining what is the motion planning problem in Section **??** and presenting the assumptions in the previous Subsection we can identify and define what are the constraints and the cost function for the multi robot navigation.

(1.) The system kinetic model must hold for a computed solution to the motion planning problem problem:

$$\dot{q}_b = f(q_b, u_b) \tag{1}$$

with $q_b : \mathbb{R}^+ \to \mathbb{R}^n$ the robot's configuration vector, $u_b : \mathbb{R}^+ \to \mathbb{R}^p$ the robot's input vector and $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ the vector field.

(2.) The practical limitations of the input can be taken into account by the following constraint:

$$|u_{b,i}| \leq u_{b,i,max} \quad \forall i \in [1, p] \tag{2}$$

(3.) The cost for the multi robot system is defined as:

$$L(q, u) = \sum_{b=0}^{B-1} L_b(q_b, u_b, q_{b,final}) \tag{3}$$

where $L_b(q_b, u_b, q_{b,final})$ is the integrated cost for one robot stabilisation (see [**?**]);

(4.) To ensure collision avoidance with obstacles the euclidean distance between a robot and an obstacle (denoted $\mathrm{d}(R_b, O_m) \mid O_m \in \mathcal{O}_b, R_b \in \mathcal{B}$) has to satisfy:

$$\mathrm{d}(R_b, O_m) > 0 \tag{4}$$

For the circle representation of an obstacle the distance $\mathrm{d}(R_b, O_m)$ is defined as:

$$\sqrt{(x_b - x_{O_m})^2 + (y_b - y_{O_m})^2} - \rho_b - r_{O_m}$$

with r_{O_m} being the obstacle radius.

For the polygon representation, the distance was calculated using three different definitions according to the Voronoi region [**?**] within which R_b is located. Figure 1 shows an example of the thee kind of regions for a quadrilateral $ABCD$ representation. The regions are defined by the lines containing the sides (s lines) and by the lines passing through the vertices that are orthogonal to the sides (r lines).

The region in which the robot R_b is located can be computed by evaluating the line equations s_{AB}, s_{BC}, s_{CD}, s_{DA}, r_{AB}, r_{AD}, r_{BA}, r_{BA}, r_{BC}, r_{BA}, r_{BC}, r_{CB}, r_{CD}, r_{DC} and r_{DA} for the position associated with the configuration q_b.

For the example in the Figure 1 the distance robot-to-quadrilateral could be found as follows:

(a) If robot in region 1:

$$\sqrt{(x_b - x_A)^2 + (y_b - y_A)^2} - \rho_b$$

which is simply the distance of the robot to the vertex A.

(b) If robot in region 2:

$$\mathrm{d}(s_{DA}, (x_b, y_b)) - \rho_b$$

where

$$\mathrm{d}(s_{DA}, (x_b, y_b)) = \frac{|a_{s_{DA}} x_b + b_{s_{DA}} y_b + c_{s_{DA}}|}{\sqrt{a_{s_{DA}}^2 + b_{s_{DA}}^2}}$$

The distance $\mathrm{d}(s_{DA}, (x_b, y_b))$ represents from the robot to the side DA.

(c) If robot in region 3:

$$-\min\left(\mathrm{d}(s_{AB}, (x_b, y_b)), \cdots, \mathrm{d}(s_{DA}, (x_b, y_b))\right) - \rho_b$$

which represents the amount of penetration of the robot in the obstacle.

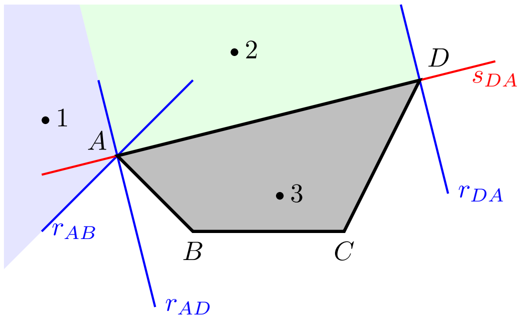

FIGURE 1. Voronoi regions used for case differentiation.

(5.) In order to prevent inter-robot collision the following constraint must be respected: $\forall\ (R_b, R_c) \in \mathcal{R} \times \mathcal{R}, b \neq c,$

$$\mathrm{d}(R_b, R_c) - \rho_b - \rho_c > 0 \tag{5}$$

where $\mathrm{d}(R_b, R_c) = \sqrt{(x_b - x_c)^2 + (y_b - y_c)^2}$. This constraint justifies the need for a communication link between the robots.

(6.) In order to keep the communication between robots (R_b, R_c) the following equation must be respected:

$$\mathrm{d}(R_b, R_c) - \min(d_{b,com}, d_{c,com}) < 0 \tag{6}$$

with $d_{b,com}, d_{c,com}$ being the communication link reach of the robots.

3. Distributed motion planning

3.1. Receding horizon approach

As said before trying to find the solution from the initial configuration until the goal is not feasible. Thus, the planning has to be computed on-line as the multi robot system evolves in the environment. One way to do so is to use a receding horizon approach **??**.

All robots in the team use the same constant planning horizon T_p and update horizon T_c. T_p is the time horizon for which a solution will be computed, T_c is the time horizon during which the plan will be followed (usually this is the time needed to solve the planning problem for T_p).

For each receding horizon planning problem the following is done:

Step 1. Compute intended trajectory by ignoring coupling constraints

Step 2. Robots involved in a conflict update their trajectories by solving another optimal problem that take into account coupling constraints using the other robots' intended trajectories computed in the first step.

However, two aspects of the implementation of this algorithm are neglected in [2]: the initial values for the control points and the procedure for reaching the goal state.

3.2. Motion planning termination

As the robots evolve theirs states approximate to the goal states. But simply stopping the motion planner as the robots are in the neighbourhood of their final configuration is not a satisfying approach.

We propose At some point the constraints associated to the final state shall be integrated into the optimal problem and the timespan for performing this last step shall not be fixed and must be one of the values calculated. final state.

The criterion used to pass from the NLP used during for the initial and intermediates steps to the last step NLP is define below in the equation **??**:

$$d_{rem} \geq d_{min} + T_c \cdot v_{max} \tag{7}$$

This way we insure that the last planning section will be done for at least a d_{min} distance from the robot's final position. This minimal distance is assumed to be sufficient for the robot to reach the final state.

After stopping the sliding window algorithm we calculate new parameters for the solution representation and computation taking into account the estimate remaining distance.

The following pseudo code summarizes the algorithm:

Algorithm 1 Sliding window planning algorithm

```
procedure PLAN
    knots ← GENKNOTS(t_p, d_spl, n_knots)
    time ← LINESPACING(0, t_p, n_s)
    q_latest ← q_initial
    d_rem ← |POS(q_final) − POS(q_latest)|
    while d_rem ≥ d_min + T_c · v_max do
        q_latest ← PLANSEC
        d_rem ← |POS(q_final) − POS(q_latest)|
    end while
    Δt ← PLANLASTSEC
end procedure
```

4. Simulation results

A straight forward extension of the previous algorithm can be done in order to support a multi robot system. The sliding window algorithm presented before remains virtually the same. The changes are done within the PLANSEC PLANLASTSEC routine.

After solving the NLP stated before each robot will have generated an intended trajectory that would be valid if we were dealing with a mono robot system. For the multi robot system some exchange of information among the robots and possibly some replanning has to be done.

Right after solving the standalone NLP a given robot represented by the index i computes a conflict list that is based on all robots' positions as of when they started planning their intended trajectories (solving the latest standalone NLP). This conflict list contains the indexes of the robots that can possibly cause some conflict. The word conflict here is understood as a collision or a loss of communication between robots in the team.

Notice that the i robot can compute its conflict list as soon as it finishes its planning even though other robots may still be doing so.

For the next step of replanning all robots involved in a conflict have to be done computing the first standalone planning. This is needed simply because all intended trajectories will be taken into account on the replanning part.

Using the intended trajectory as the initialization of the optimization parameters a new NLP is solved where collision avoidance between robots and keeping communication are translated into constraints.

After solving this second NLP, the trajectories are updated and the planning goes on to the next section.

In Figures 2 and 3 the results of the decentralized multi robot algorithm can be seen. In Figure 2 no conflict handling is done and two collisions zones can be identified. For trajectory showed in the Figure 3 the robots optimize their trajectories using the multi robot adaptation of the algorithm. No conflict occurs and we can observe a change in the robots velocities and total execution time.

the multi robot system TODO

4.1. Conflict detection

Conflict detection is computed TODO

4.2. Aditional constraints

The additional constraints associated to the multi robot system TODO

5. Parameters' impact analyses

Four criteria considered important for the validation of this method were studied. We tested different parameters configuration and scenario in order to understand how they influence those criteria. The four criteria were:

- *Maximum computation time* over the computation horizon (MCT/T_c ratio);
- Obstacle penetration area (P).
- The total execution time (T_{tot});
- Additional time for conflict handling???.

5.1. Maximum computation time over computation horizon MCT/T_c

The significance of this criterion lays in the need of quarantining the real-time property of this algorithm. In a real implementation of this approach the computation horizon would have always to be superior than the maximum time took for computing

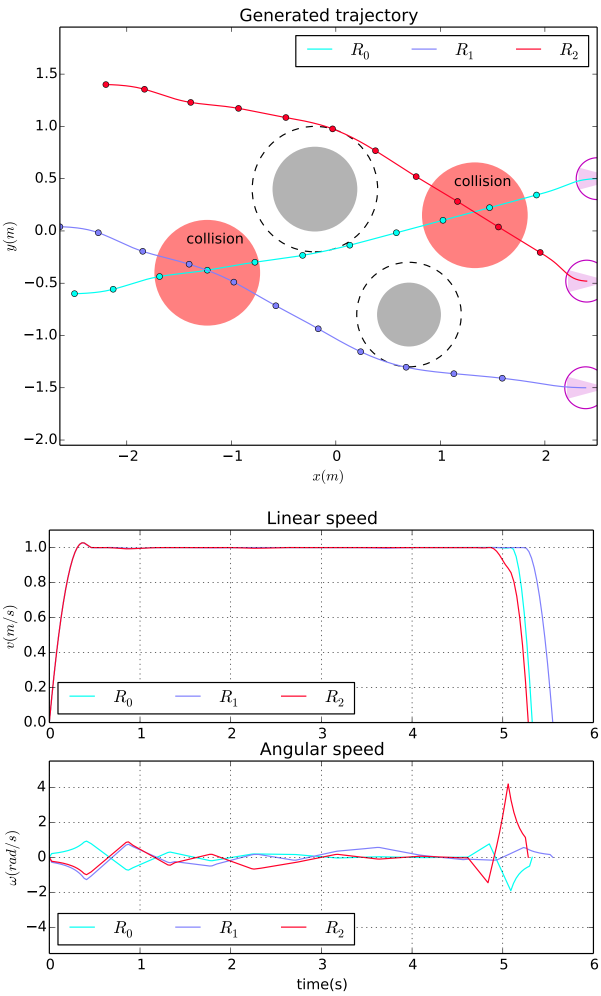

Figure 2. Our results: black box (top) and black box (bottom).

a planning section (robot-to-robot conflict taken into account).

Based on several simulations with different scenarios we were able to TODO

- SLSPQ method request $O(n^3)$ time, n being the number of knots;

5.2. Obstacle penetration P

5 TODO rescale images

5.3. Total execution time T_{tot}

5.4. Additional time for conflict handling$P$

TODO Comparison with the other method;
TODO Before concluding do comparison with other approach and make sure to have multi-robot stuff

6. Conclusions

TODO perspectives
Analise influence of dynamics of system, sensors, communication latency;

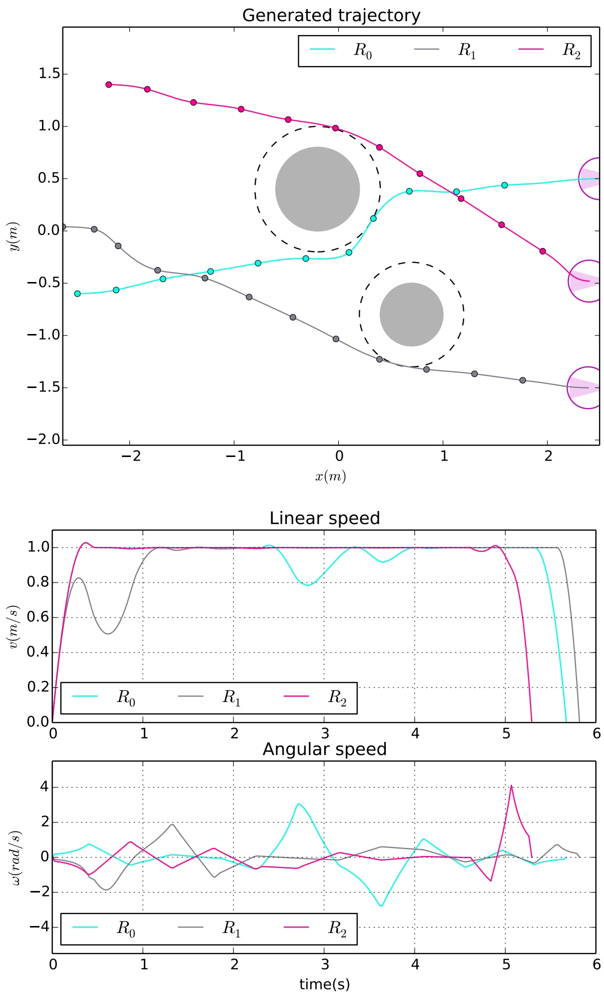

FIGURE 3. Our results: black box (top) and black box (bottom).

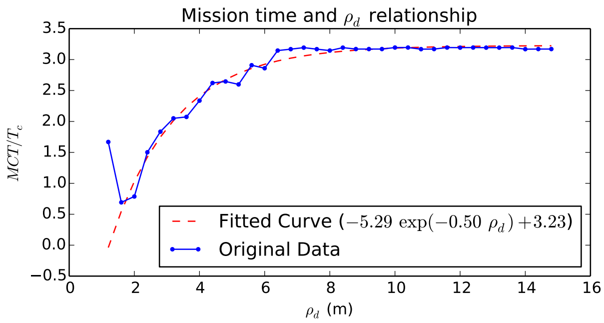

FIGURE 4. Increasing of detection radius and impact on a MTC/T_c ratio

FIGURE 5. Obstacle penetration decreasing as sampling increases

REFERENCES

[1] Autonomous robots are helping to pack your amazon orders. `http://www.gizmag.com/amazon-kiva-fulfillment-system/34999/`. Accessed: 2015-07-07.

[2] M. Defoort. Contributions à la planification et à la commande pour les robots mobiles coopératifs. *Ecole Centrale de Lille* 2007.