Dynamic Validation of Multi-robot Motion Planning Using a Distributed Receding Horizon Approach

José M. Mendes Filho [a, b, *], Eric Lucet [a] and David Filliat [b]

Abstract— **This paper analyzes the real-time implementation of an algorithm for collision-free motion planning of a team of wheeled mobile robots in the presence of obstacles in a realistic enviromnent. Planning and navigation are simulated with three robots. Deviations from the planned motion caused by the system dynamics can be overcome by doing few changes in the optimization problem uderlying the planning algorithm and using a feedback controller.**

I. INTRODUCTION

The capability of defining a collistion-free motion plan for passing from one configuration to another is a crutial aspect of robotics that can be specially hard to solve for mobile multi-robot systems. A trending application that require this capability is the use of robotic systems in industrial supply chains for processing orders and optimizing the storage and distribution of products. For example, Amazon employs the Kiva mobile-robot system, and IDEA Groupe employs the Scallog system for autonomously processing client orders [**?**], [**?**]. Such logistics tasks became increasingly complex as sources of uncertainty, such as human presence, are admitted in the work environment.

For efficiently solving the motion planning problem, different constraints must be taken into account, in particular, geometric, kinematic and dynamic constraints. The first constraints result from the need of preventing the robot to assume specific configurations in order to avoid collisions, communication loss, etc. Kinematic constraints derive directly from the mobile robot architecture implying, in particular, in nonholonomic constraints. Dynamics constraints come mainly from inertial effects and interaction between different bodies in contact.

In [1], a Distributed Receding Horizon Motion Planning is presented. It is intended for planning the motion of a team of nonholonomic mobile robots, in a partially known environment occupied by static obstacles, being efficient with respect to the travel time (amount of time to go from initial to goal configuration). However, only kinematic validation of that approach was done and its applicability in a more realistic scenario remained to be tested.

This work builds directly on that approach and aims to analyze that motion planning method in a realistic scenario. By means of a physics engine that can simulate rigid body dynamics (including collision detection), the aproach is tested, evaluated and improved. The improvements are meant to take effects such as inertia and sliding motion into account and overcome possible deviations between the executed and planned motion.

Related work (...)

This paper outline is as follows. The second section gives an overview of the Distributed Receding Horizon Motion Planning. It shows how this approach manages to find motion plans that respect geometric and kinematics cosntraints while minimizing the travel time of each robot in the team. The third section proposes a simple change on the optimization problem uderlying the motion planning for providing better plans with respect to the system dynamics. It also proposes a feedback control that asymptotically stabilizes the tracking error. The forth section is dedicated to the dynamic simulation aspects and the results of using the proposed motion planning in that realistic scenario. (...) Finally, in last section we present our conclusions and perspectives.

II. DITRIBUTED RECIDING HORIZON MOTION PLANNING

As a team of robots evolves in their work enviroment they progressively perceive new obstacles in their way to their goal configuration. Thus, try to plan for the whole motion from initial to goal configurations is not a satisfying approach. Planning locally and replanning is more suitable for taking new information into account as it comes.

In the Distributed Reciding Horizon approach for motion planning, each robot in the team computes its own local plan. Two fundamental concepts of this approach are the planning horizon T_p and update/computation horizon T_c. T_p is the timespan for which a solution will be computed and T_c is the time horizon during which a plan is executed while the next plan, valid for the next timespan T_p, is being computed. The problem of producing a motion plan for a T_p horizon during the T_c time interval is called here a receding horizon planning problem.

For each receding horizon planning problem, the following steps are performed:

Step 1: Each robot in the team computes it own intended solution trajectory (denoted $(\hat{q}_b(t), \hat{u}_b(t))$ with q_b the configuration vector a robot b and u_b its input vector) by solving a constrained optimization problem that takes geometric and kinematics constraints into account. Coupling constraints, that is, constraints that involve solving

[a] CEA, LIST, Interactive Robotics Laboratory, Gif-sur-Yvette, F-91191, France

[b] Unité d'Informatique et d'Ingénierie des Systèmes, ENSTA Paristech, 828 bd des Marechaux, 91762, France

[*] corresponding author: `jose.mendesfilho@cea.fr`

a conflict between two robots such as collision or loss of communication, are ignored in this step.

Step 2: Robots involved in a potential conflict (risk of collision, lost of communication) update their trajectories computed during Step 1 by solving a second constrained optimization problem that additionally takes into account constraints for avoiding the conflict. This is done by using the intended trajectories of other robots computed in the previous step as an estimate of those robots' final trajectories. If a robot is not involved in any conflict, Step 2 is not executed and its final solution trajectory is identical to the one estimated in Step 1.

All robots in the team use the same T_p and T_c for assuring that their intended trajectories begin at the same time and are valide for the same time horizon. However, this is not necessary condition.

For each of these steps and for each robot in the team, one constrained optimization problem is resolved. The cost function to be minimized in those optimization problems is the geodesic distance of a robot's current configuration to its goal configuration. This assures that the robots are driven towards their goal.

This two step scheme is explained in details in [**?**] where constrained optimization problems associated to the receding horizon optimization problem are formulated.

However, when a robot arrives closer to its goal the receding horizon planning scheme does not produce the desired effect. For instance, near the goal, a plan for reaching it can possibly take less time than the T_p planning horizon.

In [1] a termination procedure for reaching the goal is proposed. It takes the goal configuration as a hard constraint in the optimization problem and uses the time for reaching the goal as the cost function to be minimized.

Figure **??** illustrates how plans would be generated through time by the receding horizon scheme with termination plan.

Although, the plans generated with this approach are not necessarily appropriated for

III. Predictive control law synthesis

A. Extended Model

For finding a predictive control law an extended model which integrates the dynamics of the unicycle is needed. The model bellow in equation 1 can be used for that purpose:

$$\underbrace{\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\omega} \end{bmatrix}}_{\dot{q}} = \underbrace{\begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \\ \frac{p_3}{p_1}\omega^2 - \frac{p_4}{p_1}v \\ -\frac{p_5}{p_2}v\omega - \frac{p_6}{p_2}\omega \end{bmatrix}}_{f(q)} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{p_1} & 0 \\ 0 & \frac{1}{p_2} \end{bmatrix}}_{g(q)=[g_1(q)\ g_2(q)]} \underbrace{\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}}_{u} \quad (1)$$

$q \in Q \subset \mathbb{R}^n$ is the state vector and $u \in U \subset \mathbb{R}^p$ the input.

The parameters vector $p \in \mathbb{R}^6$ can be determined by system identification or based on properties of the unicycle such as mass, moment of inertia with respect to different axes, impedance of motors etc.

B. Optimal predictive control

The objective is to synthesize a control law that minimizes the quadratic error in position and orientation (i.e. pose) over a time horizon T ahead of the current instant t.

Since only error in pose is to be minimized, the system output can be written as follows:

$$h = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

And the error as:

$$e(t) = h(t) - h_{\text{ref}}(t)$$

where $h_{\text{ref}}(t)$ is derived from the trajectory planner output.

The criterion to be minimized represented by J can be written as:

$$J = \sum_{i=1}^{m} J_i$$

with

$$J_i = \int_0^T (\hat{e}_i(t+\tau))^2 d\tau$$

where $\hat{e}_i(t+\tau)$ represents the prediction error at $t+\tau$ with $0 < \tau \leq T$.

To find the control law that minimizes J is to find u satisfying the equation:

$$\frac{\partial J}{\partial u} = 0_{p\times 1}$$

For solving the above equation a expression for the prediction error must be defined and the criterion rewritten in a matrix form.

C. Predictive error definition

Using Taylor series, each of the coefficients of $h(t+\tau)$ can be written as bellow:

$$h_i(t+\tau) = \sum_{k=0}^{\rho_i} h_i^{(k)}(t)\frac{\tau}{k!} + R(\tau^{\rho_i})$$

Rewriting in a matrix form and excluding the remainder term an approximation for the output h_i is given bellow:

$$h_i(t+\tau) \simeq \underbrace{\begin{bmatrix} 1 & \tau & \frac{\tau^2}{2} & \dots & \frac{\tau^{\rho_i}}{\rho_i!} \end{bmatrix}}_{\Lambda_i} \begin{bmatrix} h_i(t) \\ \dot{h}_i(t) \\ \ddot{h}_i(t) \\ \vdots \\ h_i^{(\rho_i)}(t) \end{bmatrix} \quad (2)$$

Replacing the first matrix by the more compact notation Λ_i and using the standard geometric notation for Lie derivatives the previous can be written as:

$$h_i(t+\tau) \simeq \Lambda_i L_{h_i} \tag{3}$$

where

$$L_{h_i} = \begin{bmatrix} L_f^{(0)} h_i(t) \\ L_f^{(1)} h_i(t) \\ \vdots \\ L_f^{(\rho_i - 1)} h_i(t) \\ L_f^{(\rho_i)} h_i(t) + L_g L_{h_i}^{(\rho_i - 1)} h_i(t) u(t) \end{bmatrix}$$

$$\begin{cases} L_f^{(k)} h_i & = & L_f L_f^{(k-1)} h_i = \frac{\partial L_f^{\rho - 1} h_i}{\partial q}(q) f(q) \\ L_f^{(0)} h_i & = & h_i \end{cases}$$

The second term in the prediction error expression, $h_{\text{ref}}(t+\tau)$, can be analogously written as:

$$h_{\text{ref},i}(t+\tau) \simeq \Lambda_i \underbrace{\begin{bmatrix} h_{\text{ref},i}(t) \\ \dot{h}_{\text{ref},i}(t) \\ \ddot{h}_{\text{ref},i}(t) \\ \vdots \\ h_{\text{ref},i}^{(\rho_i)}(t) \end{bmatrix}}_{L_{h_{\text{ref},\,i}}} \tag{4}$$

$L_{h_{\text{ref},\,i}}$ is supposed to be known from the trajectory planner output for any ρ_i.

D. Control law equation

After some (a lot of) algebraic manipulation we can show that:

$$\frac{\partial J}{\partial u} = 0 \Rightarrow u = -(D^T D)^{-1} D^T K E$$

with D the decoupling matrix, K the the gain matrix and E the prediction error matrix defined as bellow:

$$D = \begin{bmatrix} L_{g_1} L_f^{\rho_1 - 1} h_1 & \dots & L_{g_p} L_f^{\rho_1 - 1} h_1 \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{\rho_m - 1} h_m & \dots & L_{g_p} L_f^{\rho_m - 1} h_m \end{bmatrix} \tag{5}$$

$$E = \begin{bmatrix} h_1 - h_{\text{ref},\,1} \\ \vdots \\ L_f^{(\rho_1)} h_1 - h_{\text{ref},\,1}^{(\rho_1)} \\ \vdots \\ h_m - h_{\text{ref},\,m} \\ \vdots \\ L_f^{(\rho_m)} h_m - h_{\text{ref},\,m}^{(\rho_m)} \end{bmatrix} \tag{6}$$

$$K = \begin{bmatrix} \Pi_1^{ss} & & 0 \\ & \ddots & \\ 0 & & \Pi_m^{ss} \end{bmatrix}^{-1} \begin{bmatrix} \Pi_1^{s} & & 0 \\ & \ddots & \\ 0 & & \Pi_m^{s} \end{bmatrix} \tag{7}$$

with Π_i^s the last line of the matrix Π_i and Π_i^{ss} the last element of the vector Π_i^s.

Π_i being defined as:

$$\Pi_i = \int_0^{T_i} \Lambda_i^T \Lambda_i d\tau \tag{8}$$

$$= \begin{bmatrix} T_i & \frac{T_i^2}{2} & \dots & \frac{T_i^{(\rho_i - 1)+1}}{((\rho_i - 1)+1)(\rho_i - 1)!} & \frac{T_i^{\rho_i + 1}}{(\rho_i + 1)\rho_i!} \\ \frac{T_i^2}{2} & \frac{T_i^3}{3} & \dots & \frac{T_i^{((\rho_i - 1)+1)+1}}{(((\rho_i - 1)+1)+1)(\rho_i - 1)!} & \frac{T_i^{(\rho_i + 1)+1}}{((\rho_i + 1)+1)\rho_i!} \\ & & \ddots & & \\ \frac{T_i^{\rho_i + 1}}{(\rho_i + 1)\rho_i!} & & \dots & \frac{T_i^{(\rho_i + (\rho_i - 1))+1}}{((\rho_i + (\rho_i - 1))+1)\rho_i!(\rho_i - 1)!} & \frac{T_i^{(\rho_i + \rho_i)+1}}{((\rho_i + \rho_i)+1)\rho_i!\rho_i!} \end{bmatrix} \tag{9}$$

The MIMO (Multiple Input Multiple Output) system has relative degree vector $\rho = [\rho_1(t) \dots \rho_m(t)]$ for all q in the neighborhood of q^0 if:

- $L_{gj} L_f^k h_i(x) = 0$ for all $1 \leq j \leq p$, for all $k < \rho_i - 1$, for all $1 \leq i \leq m$ and for all q in the neighborhood of q^0
- the product $D^T D$ is non-singular, D being the decoupling matrix of dimension $m \times p$, given by:

$$D = \begin{bmatrix} L_{g_1} L_f^{\rho_1 - 1} h_1 & \dots & L_{g_p} L_f^{\rho_1 - 1} h_1 \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{\rho_m - 1} h_m & \dots & L_{g_p} L_f^{\rho_m - 1} h_m \end{bmatrix} \tag{10}$$

A way of trying to find a vector ρ for our particular system is by computing $L_{g_i} L_f^k h_j$ for k beginning at 0 and incrementing it until the conditions above are satisfied.

For $\rho = [1\ 1\ 1]$, $L_{g_i} L_f^0 h_j = L_{g_i} h_j = 0$ for all $1 \leq i \leq p$, for all $1 \leq j \leq m$.

For computing $L_{g_i} L_f^k h_j$ with $\rho = [2\ 2\ 2]$ we need first $L_f h_j$:

$$\begin{cases} L_f h_1 & = & [1\ 0\ 0\ 0\ 0\ 0] f(q) = v \cos\theta \\ L_f h_2 & = & [0\ 1\ 0\ 0\ 0\ 0] f(q) = v \sin\theta \\ L_f h_3 & = & [0\ 0\ 1\ 0\ 0\ 0] f(q) = \omega \end{cases} \tag{11}$$

Computing now $L_{g_j} L_f h_j$ we obtain:

$$\begin{cases} L_{g_1} L_f h_1 & = & [0\ 0\ -v\sin\theta\ \cos\theta\ 0\ 0] g_1(q) = \cos\theta / p_1 \\ L_{g_1} L_f h_2 & = & [0\ 0\ v\cos\theta\ \sin\theta\ 0\ 0] g_1(q) = \sin\theta / p_1 \\ L_{g_1} L_f h_3 & = & [0\ 0\ 0\ 0\ 0\ 1] g_1(q) = 0 \end{cases} \tag{12}$$

$$\begin{cases} L_{g_2} L_f h_1 & = & [0\ 0\ -v\sin\theta\ \cos\theta\ 0\ 0] g_2(q) = 0 \\ L_{g_2} L_f h_2 & = & [0\ 0\ v\cos\theta\ \sin\theta\ 0\ 0] g_2(q) = 0 \\ L_{g_2} L_f h_3 & = & [0\ 0\ 0\ 0\ 0\ 1] g_2(q) = 1/p_2 \end{cases} \tag{13}$$

Which gives the following decoupling matrix:

$$D = \begin{bmatrix} \frac{\cos\theta}{p_1} & 0 \\ \frac{\sin\theta}{p_1} & 0 \\ 0 & \frac{1}{p_2} \end{bmatrix} \tag{14}$$

and consequently:

$$D^T D = \begin{bmatrix} \frac{1}{p_1^2} & 0 \\ 0 & \frac{1}{p_2^2} \end{bmatrix} \tag{15}$$

which is non-singular for all $p_1, p_2 \neq 0$.

$$K = \begin{bmatrix} \frac{10}{3T_1^2} & \frac{5}{2T_1} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{10}{3T_2^2} & \frac{5}{2T_2} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{10}{3T_3^2} & \frac{5}{2T_3} & 1 \end{bmatrix} \tag{16}$$

$$E = \begin{bmatrix} x - x_{\text{ref}} \\ v\cos\theta - v_{\text{ref}}\cos\theta_{\text{ref}} \\ \left(\frac{p_3}{p_1}\omega^2 - \frac{p_4}{p_1}v\right)\cos\theta - v\omega\sin\theta - a_{\text{ref}}\cos\theta_{\text{ref}} \\ y - y_{\text{ref}} \\ v\sin\theta - v_{\text{ref}}\sin\theta_{\text{ref}} \\ \left(\frac{p_3}{p_1}\omega^2 - \frac{p_4}{p_1}v\right)\sin\theta + v\omega\cos\theta - a_{\text{ref}}\sin\theta_{\text{ref}} \\ \theta - \theta_{\text{ref}} \\ \omega - \omega_{\text{ref}} \\ -\frac{p_5}{p_2}v\omega - \frac{p_6}{p_2}\omega - \alpha_{\text{ref}} \end{bmatrix} \tag{17}$$

$$u(q, q_{\text{ref}}, a_{\text{ref}}, \alpha_{\text{ref}}, T, p) = -(D^T D)^{-1} D^T K E$$

$$\dot{q}(t) = f(q(t), u(t)) \Rightarrow \begin{cases} \dot{x} = u_1\cos\theta \\ \dot{y} = u_1\sin\theta \\ \dot{\theta} = u_2\cos\theta \end{cases} \tag{18}$$

For the feedback control, a specific method applicable to unicycle-type robots for tracking a reference vehicle with the same kinematics was used [**?**]. The configuration trajectory $t \longmapsto q_b^*(t)$ can be considered as the solution to the robot's kinematic model for the specific control input trajectory $t \longmapsto u_b^*(t)$. Therefore, those trajectories can be used as references ((19) and (20)) in the control law in (21) that renders the system globally asymptotically stable under certain conditions[1].

$$q_b^*(t) = [x_r\ y_r\ \theta_r]^T \tag{19}$$

$$u_b^*(t) = [u_{1,r}\ u_{2,r}]^T \tag{20}$$

$$\begin{cases} u_1 = (-k_1|u_{1,r}|(w_1 + w_2 w_3) + u_{1,r})\cos^{-1}(\theta - \theta_r) \\ u_2 = (-k_2 u_{1,r} w_2 - k_3|u_{1,r}|w_3)\cos^2(\theta - \theta_r) + u_{2,r} \end{cases} \tag{21}$$

with $w_1 = x - x_r$, $w_2 = y - y_r$, $w_3 = \tan(\theta - \theta_r)$.

[1]The orientation error between the physical robot and the reference robot is smaller than $\pi/2$, $u_{1,r}$ is a bounded differentiable function whose derivative is bounded and which does not tend to zero as t tends to infinity

The gains k_1, k_2 and k_3 were tunned using pole placement based on the linearization of the system and control. 32 For simplicity, the observation of the configurations of each robot was considered exact. This is not an issue if we consider that the robot can detect landmarks in its enviromnent whose absolute positions are known in advance and that data fusion and filtering fsrom different sensors can bound the observation error to small values.

TABLE I: An Example of a Table

One	Two
Three	Four

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an document, this method is somewhat more stable than directly inserting a picture.

Fig. 1: Inductance of oscillation winding on amorphous magnetic core versus DC bias magnetic field

IV. CONCLUSIONS

REFERENCES

[1] J. M. Mendes Filho and E. Lucet, "Multi-robot motion planning: a modified receding horizon approach for reaching goal states," *IROS 2015 Workshop on On-line decision-making in multi-robot coordination*, pp. 1–8, 2015. [Online]. Available: http://robotics.fel.cvut.cz/demur15/wp-content/uploads/2015/09/demur15-filho.pdf