Mobile Robot Navigation and Obstacles Avoidance based on Planning and Re-Planning Algorithm
Mehdi Mouad, Lounis Adouane, Djamel Khadraoui, Philippe Martinet

To cite this version:
Mehdi Mouad, Lounis Adouane, Djamel Khadraoui, Philippe Martinet. Mobile Robot Navigation and Obstacles Avoidance based on Planning and Re-Planning Algorithm. 10th International IFAC Symposium on Robot Control (SYROCO’12), Sep 2012, Dubrovnik, Croatia. hal-01714850

HAL Id: hal-01714850
https://hal.archives-ouvertes.fr/hal-01714850
Submitted on 23 Feb 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Abstract: This paper deals with a multi-mode control architecture for robot navigation and obstacle avoidance. It presents an adaptive and flexible algorithm of control which guarantees the stability and the smoothness of mobile robot navigation dealing with unexpected events. Moreover, the proposed Planning and Re-Planning (PRP) algorithm combine the two schools of thought, the one based on the path planning to avoid obstacles and reach the target, described as cognitive, and the second using the reactive algorithms. In fact the mix of these two approaches allows us to develop a very reliable algorithm. It provides us a scalable mobile robot navigation and obstacle avoidance, with less processing. It is accomplished by making an initial path planning, then to resolve the problem of unexpected static or dynamic obstacles while tracking the trajectory. A system of hierarchical action selection allows us to switch to a reactive avoidance, then to re-plan a new and safe trajectory to reach the target. A large number of simulations in different environments are performed to show the efficiency of the proposed PRP algorithm.

Keywords: Control Architectures, Obstacles Avoidance, Planning and Re-Planning, Reactive Avoidance.

1. INTRODUCTION

Controlling mobile robots in uncertain environments is of a central importance to many important fields, and calls for many disciplines such as path planning, path tracking and obstacle avoidance.

Navigation in unfamiliar environment needs to give the robot the ability to generate its action plan and to track it. The literature presents different methods of path planning in robotics, like Artificial Potential field method as described in Khatib et al. (1986), Arkin et al. (1989), where robot actions are guided by the sum of attractive and repulsive fields generated respectively by the target and the obstacles. Cell decomposition method cf. Lingelbach et al. (2004 a, b), which consists on decomposing the robots configuration plan on a set of adjacent connected areas from where a graph is generated, the planning consists on exploring this graph and detecting the adjacent areas. Another planning method called Vector field histogram developed by Hafner et al. (2000) and Nehmzow et al. (2000), based on a grid representing the environment each cell contain a value representing the probability of obstacle presence, to plan a path we choose the nearest direction to the target direction which doesn't contain obstacle.

To have an efficient path planning, we need to provide to the robot a precise obstacle map of the environment at the planning moment. Since it is impossible in most of robotic contexts, it is necessary for the robot to carry a motion planner cf. Belta et al. (2005), Conner et al. (2006). For each environment changing the planner can recalculate a new path; however in certain circumstances this may not be appropriate, because these algorithms generally have a lack of responsiveness. Indeed, the obstacles are often discovered late and plan generation is a time consuming operation. We must therefore stop the robot to start the process of re-planning but this is not a solution in the case of moving obstacles, or when the stop distance is too long.

Control software architectures are usually classified into three main categories as shown in Ridao et al. (2001):

- Reactive vs. Cognitive (deliberative) architectures, many modules connects several inputs sensors/actuators, each module implements a behaviour. These behaviours are called “reactive” because they provide an immediate output of an input value, and cognitive otherwise cf. Brooks et al. (1986), Rosenblatt et al. (1997).

- Hierarchical vs. Non-Hierarchical architectures, the hierarchical architectures are built in several levels, usually three. Decisions are taken in the higher level; the intermediate level is dedicated to control and supervision. The low level deals with all periodical treatment related to the instrumentation, such as actuator control or measuring instrument management (Lumia et al. (1990)).

- Hybrid architectures are a mix of the two previous ones such as Ridao et al. (2001) has described. Usually these are structured in three layers: the deliberative layer, based on planning, the control execution layer and a functional reactive...
layer (Schneider et al. (1998)). It’s in the same time reactive with a cognitive level (planning for example).

Several robot control architecture prefer then to base their navigation on a reactive navigation where the control of the actuators is directly coupled with the perception of the environment (Toibero et al. (2007)). This allows the inclusion of very fast dynamic phenomena of the environment as described these works Minguez (2008), Egerstedt et al. (2002), Belta et al. (2005), Conner et al. (2006), Stuart et al. (1996), Lim et al. (2002). These architectures, does not include internal representation of the environment state; make it difficult to plan a sequence of actions to reach a goal.

We propose to combine the two approaches according to the context. The architecture is constructed with modular and bottom-up approach as developed by Adouane et al. (2006) and contains a planner with a model of the environment that generates a trajectory, and then when an obstacle is encountered, a reactive avoidance is triggered. Meanwhile the environment map is enriched and a new trajectory is calculated and followed after the obstacle avoidance. This combination allows us to benefit of the advantages of the two approaches to safely generate a new solution to deal with unexpected events.

The proposed Planning and Re-Planning (PRP) algorithm uses in a first time an off-line path planning algorithm based on the principle of artificial potential fields, taking into account the known obstacles in the environment, and the limit cycles based approach for obstacle avoidance algorithm for reactive avoidance of unexpected obstacles cf. Stuart et al. (1996), Khalil et al. (2002). The proposed algorithm provides reliable and continuous robot navigation with less processing.

The rest of the paper is organized as follows. Section II gives the specification of the task to achieve. The details of the proposed PRP algorithm are given in section III. In section IV we present in details the transitions criteria and hierarchical action selection of our PRP algorithm, whereas section V describes and analyses the simulation results. We end this paper with conclusion and future works.

2. NAVIGATION IN PRESENCE OF OBSTACLES

Navigation is a fundamental problem in mobile robotics. The local navigation problem deals with navigation on the scale of a few meters, where the main problem is obstacle avoidance.

The ultimate goal is the capability of achieving coordinated movements and of carrying out tasks that usually require human assistance. This need for autonomy requires from the robot a certain capacity of being able at any moment to assess both its state and its environment that are usually combined with different other robots states as well as with its mission requirements in order to make coherent control decisions. If we consider navigation aspects, autonomous mobile robots are usually embedded with sensors/actuators according to the mission to be performed. This complexity induces major challenges both at the development of robotics control architecture system but also at the design of navigation software.

Indeed, an autonomous mobile robot has to carry out a set of sensors/actuators dedicated to its own navigation and another sensor set that can change according to the mission to be performed. Therefore the navigation software developed for these vehicles become complex and requires a design methodology.

The unexpected obstacles detection is not addressed in this paper, we use obstacle detection algorithm developed in a previous work cf. Vilica et al. (2012).

Before describing our PRP algorithm, let’s show the use kinematic robot model (cf. Figure 1):

\[
\xi = \begin{pmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{pmatrix} = \begin{pmatrix}
\cos \theta & -l_2 \cos \theta - l_1 \sin \theta & 0 \\
\sin \theta & -l_2 \sin \theta + l_1 \cos \theta & 0 \\
0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
v \\
\omega
\end{pmatrix}.
\]

With:

- \(x, y, \theta\): Configuration state of the unicycle at the point “\(P_t\)" of absissa and ordinate \((l_1, l_2)\) according to the mobile reference frame \((X_m, Y_m)\).
- \(v\): Linear velocity of the robot at the point “\(P_t\)”,
- \(\omega\): Angular velocity of the robot at the point “\(P_t\)".

![Figure 1. Robot configuration in a Cartesian reference frame.](image)
control the re-planning action. Its last connection is with a switch to choose the controller needed and apply its behaviour.

The trajectory tracking controller is the default activated controller. The Hierarchical Action selection module activates the reactive obstacle avoidance controller as soon as it exists at least one obstacle that can obstruct the future robot movement toward the target, and ask for a re-planning of a new trajectory after the robot has avoided the obstacle (value of $x_o$ positive as shown in figure 9), then switch to trajectory tracking to track the new trajectory. This allows us to decrease the time to reach the target especially in cluttered environments, and lower the processing volume. Thus, we hierarchically give a priority to each behaviour, and choose to apply a specific one, depending on the situation and its priority.

4. PLANNING AND RE-PLANNING ALGORITHM

The objective of proposed PRP algorithm is to obtain safe, smooth and fast robot navigation in order to be able to navigate through the environment from Point A to point B; the robot should be able to choose an action that maximizes its chances of getting to its goal. Our PRP algorithm relies on three main algorithms that are: off-line path planning, trajectory tracking algorithm and the orbital reactive obstacle avoidance, with determined transitions in a context of hierarchical action selection, it consists of three main stages.

4.1 Off-line path planning

When the navigation goal is specified, we compute a first quick off-line path planning, based on the well-known principle of artificial potential fields. It takes into account the known obstacles in the environment cf. Khatib et al. (1986), to generate an initial trajectory to track. We assume that the robot actions are guided by the sum of attractive field generated by the goal, and the repulsive field generated by each obstacle.

The figure (3) shows an example of path planning with artificial potential fields algorithm. The attractive force is defined as follow (cf. Figure 4):

$$U_{att}(q) = \xi \rho_{goal}(q).$$  \hspace{1cm}(2)$$

With $q$ the state of the mobile robot, $\xi$ a positive scalar and $\rho_{goal}(q)$ the Euclidian distance: $\|q-q_{goal}\|$. The function $U_{att}(q)$ is positive or null, and reaches its minimum $q_{goal}$, where $U_{att}(q_{goal})=0$.

The force $F_{att}$ is differentiable and:

\begin{align*}
F_{att}(q) &= -\nabla U_{att}(q). \\
F_{att}(q) &= -\xi \nabla \rho_{goal}(q). \\
&= -\xi (q-q_{goal}) / \| q-q_{goal} \|. \\
\end{align*}

\hspace{1cm}(3) (4) (5)

Once the initial trajectory is computed, we activate the tracking algorithm to permit to the robot to track its trajectory. If an unexpected obstacle is detected while

Figure 3. Trajectory generation by the artificial potential fields algorithm.

Figure 4. Shape of attractive field.

The repulsive force is defined as follow cf. Figure 5:

\begin{align*}
U_{rep}(q) &= \frac{1}{2} \eta \left( \frac{1}{\rho(q)} - \frac{1}{\rho_0} \right)^2 \text{ if } \rho(q) < \rho_0. \\
&= 0 \text{ if } \rho(q) > \rho_0. \\
\rho_0 \text{ is the distance of influence of the obstacle. The function } U_{rep} \text{ is positive or null and tends to infinity when approaching the boundary of the obstacle.} \\
F_{rep}(q) &= -\nabla U_{rep}(q). \\
&= \eta \left( \frac{1}{\rho(q)} - \frac{1}{\rho_0} \right) \frac{1}{\rho^2(q)} \nabla \rho(q) \text{ if } \rho(q) < \rho_0. \\
&= 0 \text{ if } \rho(q) > \rho_0. \\
\end{align*}

\hspace{1cm}(6) (7) (8) (9) (10)

\hspace{1cm}(2)

Figure 5. Shape of repulsive field.

Once the initial trajectory is computed, we activate the tracking algorithm to permit to the robot to track its trajectory. If an unexpected obstacle is detected while...
tracking the trajectory, we send a transition to switch to the reactive avoidance.

4.2 Trajectory tracking

In the used trajectory tracking algorithm we apply command laws described in Maalouf et al. (2006). Knowing that the kinematical model of a differentially wheeled mobile robot in Cartesian coordinates is given by eq. (1) (while using \( l_1 = l_2 = 0 \)), the objective is to track a virtual reference robot which has respectively \( v_r \) and \( \omega_r \) as its linear and angular velocities:

\[
\begin{bmatrix}
\dot{x}_r \\
\dot{y}_r \\
\dot{\theta}_r
\end{bmatrix} = \begin{bmatrix}
\cos \theta_r & 0 & v_r \\
\sin \theta_r & 0 & 0 \\
0 & 1 & \omega_r
\end{bmatrix}.
\]  

(11)

Then the error variables \( e_x, e_y, \) and \( e \theta \) that correspond to the instantaneous errors in posture variables are chosen as:

\[
\begin{bmatrix}
e_x \\
e_y \\
e \theta
\end{bmatrix} = A \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix}, \quad \text{with } A = \begin{bmatrix}
\cos \theta & \sin \theta & 0 \\
-\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix}.
\]  

(12)

Equation (12) corresponds to the errors in posture according to local reference frame of the robot (cf. Figure 6). The transformation matrix \( A \) converts global coordinates to local coordinates. While using the derivatives of the errors using the constraint \( \dot{x}_r \sin \theta_r = \dot{y}_r \cos \theta_r \) and \( e \theta = \dot{\theta}_r - \theta \), we obtain:

\[
\begin{bmatrix}
\dot{e}_x \\
\dot{e}_y \\
\dot{e} \theta
\end{bmatrix} = \begin{bmatrix}
-1 & 0 & 0 \\
0 & -1 & 0 \\
0 & 0 & -1
\end{bmatrix} v + \begin{bmatrix}
e_x \\
e_y \\
\omega_r
\end{bmatrix} + \begin{bmatrix}
v_r \cos e \theta \\
v_r \sin e \theta \\
\omega_r
\end{bmatrix}.
\]  

(13)

From this equation (13), the aim of the control law is to make the errors converge to zero. The used velocity inputs \( v_r \) and \( \omega_r \) of the control law are (Maalouf et al. (2006) and Kanayama et al. (1990)):

\[
\begin{align*}
v_r &= v_r \cos e \theta + K_x e_x. \\
\omega_r &= \omega_r + v_r K_y e_y + K_\theta \sin e \theta.
\end{align*}
\]  

(14)

By substituting \( v_r \) and \( \omega_r \) in the error of eq. (12) we get:

\[
\begin{bmatrix}
\dot{e}_x \\
\dot{e}_y \\
\dot{e} \theta
\end{bmatrix} = \begin{bmatrix}
e_x \left( \omega_r + v_r (K_y e_y + K_\theta \sin e \theta) \right) - K_x e_x \\
-e_x \left( \omega_r + v_r (K_y e_y + K_\theta \sin e \theta) \right) + v_r \sin e \theta \\
-v_r (K_y e_y + K_\theta \sin e \theta)
\end{bmatrix}.
\]  

(15)

The proof is based on the choice of Lyapunov function:

\[
V = \frac{1}{2} (e_x^2 + e_y^2) + (1 - \cos(e \theta))/K_y.
\]  

(16)

Deriving \( V \) with respect of time gives us:

\[
\dot{V} = -K_x e_x^2 - \frac{K_\theta \sin^2 e \theta}{K_y} \leq 0.
\]  

(17)

Given that \( K_x, K_y, \) and \( K_\theta \) are all positive constants, the above inequality would be satisfied and the system with the control law would be stable. The null result corresponds to the equilibrium point where \( e_x = e_y = e \theta = 0 \).

4.3 Reactive Obstacles Avoidance

The reactive avoidance algorithm is activated when an unexpected obstacle is detected, it’s based on the orbital obstacle avoidance algorithm and provides a scalable avoidance with several mechanisms to prevent oscillations, local minima and dead end robot situations. The proof of controllers stabilities are given using Laypunov functions cf. Adouane (2009).

To perform the obstacle avoidance behaviour in our algorithm, the robot needs to follow accurately limit-cycle vector fields, these vector fields are given by two differential equations:

For the clockwise trajectory motion (cf. Figure 7.a):

\[
\begin{align*}
\dot{x}_s &= y_s + x_s (R_c^2 - x_s^2 - y_s^2), \\
\dot{y}_s &= -x_s + y_s (R_c^2 - x_s^2 - y_s^2).
\end{align*}
\]  

(18)

For the counter-clockwise trajectory motion (cf. Figure 7.b):

\[
\begin{align*}
\dot{x}_s &= -y_s + x_s (R_c^2 - x_s^2 - y_s^2), \\
\dot{y}_s &= x_s + y_s (R_c^2 - x_s^2 - y_s^2).
\end{align*}
\]  

(19)

Where \((x_s, y_s)\) corresponds to the position of the robot according to the center of the convergence circle (characterized by \( R_c \) radius).
To implement such kind of obstacle avoidance it is important to:

- Detect the obstacle to avoid.
- Give the direction of the avoidance (clockwise or counter-clockwise).
- Define an escape criterion which defines if the obstacle is completely avoided or not yet.

All these different steps must be followed and applied while guaranteeing that: the robot trajectory is safe, smooth and avoids undesirable situations as deadlocks or local minima; and that the stability of the applied control law is guaranteed.

The necessary steps to carry out the obstacle avoidance algorithm are given below:

1) Among the set of disturbing obstacles (which can constrain the robot to reach the target), determine the closer to the robot. This specific obstacle has the following features: radius \( R \) and \((x_{\text{obst}}, y_{\text{obst}})\) position.

2) After the determination of the closest constrained obstacle, we need to obtain four specific areas (cf. Figure 8) which give the robot behaviour: clockwise or counter-clockwise obstacle avoidance; repulsive or attractive phase. To distinguish between these four areas we need to:

- define a specific reference frame which has the following features (cf. Figure 8):
  i. The \( x_0 \) axis connects the center of the obstacle \((x_{\text{obst}}, y_{\text{obst}})\) to the center of the target. This axis is oriented towards the target.
  ii. The \( y_0 \) axis is perpendicular to the \( x_0 \) axis and it is oriented while following trigonometric convention.

- Apply the reference frame change of the position robot coordinate \((x, y)_A\) (given in absolute reference frame) towards the reference frame linked to the obstacle \((x, y)_O\). The transformation is achieved while using the following homogeneous transformation:

\[
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}_O =
\begin{bmatrix}
  \cos \alpha & -\sin \alpha & 0 & x_{\text{obst}} \\
  \sin \alpha & \cos \alpha & 0 & y_{\text{obst}} \\
  0 & 0 & 1 & 0
\end{bmatrix}^{-1}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}_A
\]  
(20)

Once all necessary perceptions are obtained, one can apply our obstacle avoidance strategy. To obtain the set-points, it is necessary to obtain the radius \( R_{\text{e}} \) and the direction “clockwise or counter-clockwise” of the limit-cycle to follow. The position \((x_0, y_0)\) gives the configuration \((x, y)\) of the robot according to obstacle reference frame. The definition of this specific reference frame gives an accurate means to the robot to know what it must to do. In fact, the sign of \( x_0 \) gives the kind of behaviour which must be taken by the robot. In repulsive phase, the limit-cycle takes different radii to guarantee the trajectory smoothness. The sign of \( y_0 \) gives the right direction to avoid the obstacle. Indeed, if \( y_0 \geq 0 \) then apply clockwise limit-cycle direction else apply counter-clockwise direction. This choice permits to optimize the length of robot trajectory to avoid obstacles.

The stability proofs are described in Adouane (2009).

4.4 Re-Planning

While avoiding the obstacle, the sign of robot’s abscissa in the obstacle frame vary, it gives us the switch moment, then we consider that the robot has avoided the obstacle when its abscissa in obstacle frame has a positive value.

Once the robot has accomplished the reactive avoidance, we send a transition to make the Re-Planning from the first positive value of the robot’s abscissa in the obstacle frame; it consists to another call to the planning algorithm which provides a new safe trajectory. After that, we can track our new path to the target (cf. Section 5).

5. SIMULATION RESULTS

In what follow we present some graphical illustrations describing the simulation results of our proposed Planning and Re-Planning algorithm, showing its reliability and effectiveness. These simulations were implemented in Matlab environment, and by a core 2 duo computer (2.00 GHz, 1.99 GHz, 1.96 GB of RAM).

We first tried to simulate a simple planning and re-planning scenario to compare its results with what we have obtained with our PRP algorithm. The problem we encounter with this test simulation is that we couldn’t have a continuous navigation when detecting an unexpected obstacle. Indeed,
we were forced to stop the robot to get the time to re-plan a new trajectory; because the robot’s stop time was too long (cf. Figure 9).

This new result is as good as the results given by the dynamic window in Brock et al. (1999), obstacle avoidance approach.

The following two figures (11) and (12) describes respectively a first scenario of reactive avoidance preventing local minima, and the second shows the reaction of our algorithm in the case of when the obstacles are disposed as U-shape in Wang et al. (2008). This obstacle configuration leads generally to dead end, which is not the case with the used algorithm.

We made simulations in batch mode of twenty configuration cases, comparing the proposed algorithm results with the normal planning and re-planning method. We found that the proposed new approach (PRP algorithm) reduces significantly the time delay to reach the target by about 5%, with smoothness all in preventing oscillation, local minima and dead end robot situations.

6. CONCLUSIONS AND FUTURE WORKS

In this paper we describe an algorithm of planning and re-planning dedicated for the mobile robots, which aims to improve robot navigation and the obstacle avoidance system. It is based on two main algorithms, the first one is a planning algorithm based on the artificial potential field method, generating a safe path taking into account the existing static obstacles in the environment, and the second is a reactive avoidance algorithm based on the principle of limit-cycles which is used in the case of unexpected obstacles. The proposed algorithm takes the advantages of both approaches, and has obtained efficient and flexible results as shown in simulation results.

The proposed algorithm allows also reducing the time needed to reach the target. In fact, according to this algorithm, robot generates an initial safe trajectory and anticipate the collision with unexpected obstacles according to local smooth trajectory modification.

Future work will first experiment the proposed algorithm on the KHEPERA robots, and address other experimental...
aspects such as errors and uncertainty in the localisation of the robot, and sensor reading. The second step is to integrate it in MAS2CAR architecture cf. Mouad et al. (2011) and test its efficiency in a multi-robot context based on multi-agent cooperation.

REFERENCES


