An On-Line Path Planner for Industrial Manipulators

Regular Paper

Fabrizio Padula¹ and Veronique Perdereau²,*

¹ Dipartimento di Ingegneria dell’Informazione, University of Brescia - Brescia, Italy
² ISIR CNRS, Paris, France
* Corresponding author E-mail: veronique.perdereau@upmc.fr

Received 4 Oct 2012; Accepted 14 Nov 2012

DOI: 10.5772/55063

© 2013 Padula and Perdereau; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract In this paper, an on-line path planner for an industrial manipulator is presented. The proposed control architecture is capable of driving the manipulator in its environment while avoiding collisions. Potential fields are used in order to control the joint velocities in such a way that the robot avoids the obstacles. We also propose a new weighted pseudoinverse matrix that improves the manipulator capability of finding feasible paths to move around obstacles and pass through narrow corridors without relying on the manipulator dynamic model. The proposed technique fits to both redundant and non-redundant manipulators. Experimental results show the effectiveness of the proposed solution.

Keywords On-Line Path Planning, Obstacle Avoidance, Weighted Pseudoinverse

1. Introduction

It is well-known that robotic manipulation is an established technology widely diffused in industry. Nevertheless, in recent years, the necessity for lean production makes it desirable to have robots with a certain degree of autonomy. In this context, the manipulator capability of autonomously taking some decisions by avoiding reprogramming off-line robot tasks each time is a key issue in the production process (e.g., replan the trajectory when moving pieces with different sizes).

Hence, developing a path planner capable of making a manipulator arm move in its environment by avoiding collisions is of significant concern. Bearing in mind that one of the goals is to avoid off-line reprogramming, a computationally light planner is desirable so that the trajectory can be modified at run time. Moreover, for an industrial robot, it is quite common to have very efficient and reliable velocity feedback control loops embedded in each motor drive and it is not always possible to open the velocity feedback control loops in order to directly control the motor torques. Thus, a purely kinematic control system whose output is a reference signal for each single motor velocity feedback control loop, instead of a torque signal, is preferable to implement in practical cases. Moreover, in this way, many local problems, such as friction compensation, are resolved in the motor inner control loops. Path planning and obstacle avoidance in a known environment have received much interest in recent years (see, for example, [LZR00, Vis00, DV03]). Many solutions have been proposed in the literature in
order to solve these problems. They can broadly be divided into two main approaches: global strategies and local strategies.

The global solutions are commonly based on an off-line computation of a collision-free path from an initial configuration to the desired one [LP83, Lat91, LaV06]. Despite the desirable completeness property i.e., the capability of finding a feasible path in a finite amount of time (if it exists), these path planners are computationally heavy. Indeed, their computational burden exponentially increases with the number of manipulator degrees-of-freedom (DOF). For this reason, in view of the industrial requirement of on-line planning, they are far from being applicable.

Alternatives to the global approach are local techniques. They are less computationally demanding and thus preferred for on-line application. Basically, they are based on the definition of potential fields in the robot operational space. The obstacles are surrounded by repulsive potential fields and the goal position is represented by an attractive potential field. Hence, the potential fields push the manipulator links away from obstacles, thus avoiding collisions, while driving the manipulator end-effector toward the goal position. Even though local techniques are not complete, they are suitable for industrial applications. Indeed, it has to be considered that a typical industrial environment is not so complex (compared to typical research environments such as labyrinths or multidimensional bug traps) and can be well described via simple geometric shapes. As a matter of fact, in these kinds of environments, local techniques are capable of finding a feasible path while remaining far more suitable for computation.

Two different types of local techniques have been proposed in the literature. The first class is based on the joint torque control [Kha83, Kha86, VK90, ASZK98]. Artificial potential fields are used to exert virtual forces on the manipulator links (i.e., to control the joint torques). The main drawback of these solutions in view of an industrial application (besides the problems of torque controlled joints discussed above), is that they strongly rely on the manipulator dynamic model which is difficult to obtain in many practical cases. Indeed, it would require detailed information about each manipulator’s mechanical component. This information, in many industrial contexts, is not fully available.

The second class of local techniques also uses the potential fields, but to calculate the joint velocity reference signals in order to reach the goal position along a collision-free path. In this context, the manipulator redundancy is exploited to avoid collisions by controlling the joint velocities in the manipulator nullspace [MK85, KK92, AHK93, CK96, ITZ06, LWY09]. In these works, in order to let a certain point on the robot move at a given velocity, the Moore-Penrose pseudoinverse is used. This solution leads to unnatural manipulator behaviour reducing the manipulator capability of turning around obstacles [PP11]. Moreover, using the nullspace to avoid obstacles cannot completely ensure collision avoidance.

A different local approach is the dynamical system-based technique proposed in [KZB12a, KZB12b]. First, the dynamical system describing the manipulator behaviour is derived and then it is modified to avoid collisions. The system is modified accordingly to the given analytical functions describing the obstacles as smooth and non-convex surfaces. Even though these techniques seem promising, from an industrial point of view the derivation of a dynamic model seems difficult.

The solution proposed in this paper uses the potential fields in order to control the joint reference velocities. The collision avoidance control system acts on each link, even in the manipulator nullspace (i.e., to avoid collisions, the end-effector trajectory is also changed), making the proposed control system applicable to both redundant and non-redundant manipulators. The obstacle avoidance control system and the goal position following system are completely separate, as in the solution proposed in [Kha86] (where, on the contrary, joint torques were controlled). Nevertheless, the two systems are not decoupled, since not only the manipulator nullspace is used to avoid collisions, but also those vectors in the joint space that modify the end-effector trajectory.

In order to allow the proposed system work properly, a new pseudoinverse matrix is introduced. The computation of the new pseudoinverse only requires kinematic information (that is commonly known in industrial environments) and increases the manipulator capability of moving around obstacles. Both simulation and experimental results prove the effectiveness of the proposed solution.

The paper is organized as follows: in Section 2, the obstacle avoidance technique is presented, while the goal position following system is described in Section 3. The new weighted pseudoinverse is proposed in Section 4, and Section 5 is devoted to illustrative simulation examples. In Section 6, experimental results are presented, and conclusions are drawn in Section 7.

2. Obstacle avoidance

The proposed solution is aimed at making the manipulator autonomously navigate toward the goal end-effector position along a collision-free path. Being an on-line path planner, the path is not computed in advance, but at run time, while the manipulator is already executing the required task. Each link, when close
to an obstacle, feels the virtual repulsive potential field surrounding the obstacle. This field is used to compute a reference signal for the joint velocity feedback control loops in order to make the link move away from the obstacle surface. Along the same line, the end-effector is pulled toward the goal position by making it move at a certain velocity depending on a given potential field surrounding the goal position. The desired end-effector velocity is then translated into reference signals for the joint velocity feedback control loops.

All the joint velocity terms are finally added. This means that the real end-effector velocity also depends on all the desired link velocities (to make them escape from obstacles) and vice versa.

It is worth stressing that once the desired operational space velocity for a certain point along the manipulator kinematic chain has been computed, this request can be fulfilled by using different joint velocities. Given a general point along the manipulator kinematic chain, we propose here a solution to choose the desired joint velocities in such a way that leads to good results, without involving the knowledge of the manipulator dynamic model.

2.1 Distance and repulsive potential fields

Even though this work is focused on the control system and the solution proposed in the following paragraph is rather general and does not depend on the way the distance and the potential field are defined, a brief discussion about distances and potential fields may help the reader to better understand the following sections. Please note that within the same control strategy, other potential fields and different notions of distance could be used instead of the ones proposed hereafter.

In the literature, several potential field formulations exist, but two main approaches have been proposed: it is possible to define the repulsive potential field using a superquadratic function [VK90] or a harmonic function [KK92]. Actually, using harmonic functions has a great advantage for mobile robots, since local minima are avoided. However, it is not very useful for manipulator arms where there are several points controlled by the potential fields, in addition, it is much more complex and computationally heavier. Indeed, it is worth noting that in the case of industrial manipulators, being in general relatively simple environments, it is quite unusual to get the end-effector trapped in a local minimum depending on the fields acting on it. On the contrary, it is more realistic to get the whole robot trapped in local minima depending on the superposition of the effects of the repulsive potential fields acting all along the manipulator kinematic chain and the effect of the attractive potential field acting on the end-effector (e.g., the manipulator is trying to surround an obstacle whose perimeter is longer than the manipulator kinematic chain). Bearing this idea in mind, we use here the well-known potential field proposed in [Kha 86]:

$$U(x) = \begin{cases} \frac{k_{av}}{2} \left( \frac{1}{\rho} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho \leq \rho_0, \\ 0 & \text{if } \rho > \rho_0 \end{cases}$$

where $k_{av}$, $\rho_0$, and $\rho$ are (respectively) a numeric constant, the limit distance of the potential field influence and the distance between a certain point where the potential fields is calculated and the obstacle. In order to calculate the desired velocity for a certain point $I$, we calculate the opposite of the potential field gradient at $x = I$:

$$\dot{x} = -\nabla U(x)|_{x=I}. \quad (2)$$

Evidently the concept of potential field is intimately related to the concept of distance. Indeed, in order to compute the effect of a given field on a given point along a given link, we must know the distance between the link and the obstacle surrounded by the field. However, calculating the distance between an obstacle and a manipulator link could be quite complex unless the geometry of the obstacle itself is relatively easy. But this is exactly the typical case of an industrial environment. Moreover, in many cases, instead of using the Euclidean distance, we can use a pseudodistance a approximating the obstacle shape with a superquadratic surface [Pas01, PPD02]. Furthermore, using superquadratic surfaces to approximate the obstacle shapes, allows a fast and automatic computation of the distance between a given link and the obstacle.

![Figure 1. The control system block scheme.](image)

Note that the use of superquadratic surfaces may reduce the real feasible space. While this is a major drawback in the case of a mobile robot navigating in hypercluttered environments, from an industrial point of view this is acceptable and it actually increases the safety margins.

2.2 Control system

Now that a potential field and a method to compute the distance between obstacles and links (they are taken from the literature and they are not the only usable approaches) have been defined, we can describe the obstacle avoidance control system. In order to avoid
collisions, the virtual potential fields are used to change the joint reference velocities. The repulsive fields act on some control points called $psp$, distributed along the manipulator kinematic chain. At each instant, the $psp$ are dynamically computed as the nearest points between every link and every obstacle. Note that the calculation of the nearest point by using superquadratic surfaces to envelope the obstacles is an optimization problem whose outcome is an algebraic equation $[Pas01, PPD02]$, so that this procedure can be completely automatized.

As stated before, we use the repulsive field to control the manipulator joint velocities. Hence, each control point is a velocity controlled point whose desired velocity depends on the repulsive potential fields surrounding the obstacles according to (2). In order to calculate the desired velocity for the $ith psp$ for a given obstacle, we apply (2) to (1) obtaining the following equation:

$$x_{psp} = \begin{cases} k_v \left( \frac{1}{\rho_i} - \frac{1}{\rho_0} \right) \frac{1}{\rho_1^2} \frac{\partial \rho_i}{\partial x_{psp}} & \text{if } \rho_i \leq \rho_0 \\ 0 & \text{if } \rho_i > \rho_0, \end{cases} \tag{3}$$

where $k_v$ and $\rho_0$ are (respectively) the intensity of the repulsive field potential and the limit distance of the potential field influence (both parameters to be tuned); $\rho_1$ is the distance between the $ith$ control point and the obstacle.

In the case of multiple obstacles, the same procedure should be applied to every obstacle in the environment and for each control point.

Once the desired velocity for each control point has been computed, the requirement should be translated into desired joint velocities to be used as reference signals for the joint velocity loops. Given a single $psp$, the joint reference velocities can be obtained by means of a Jacobian pseudoinverse. Here, all the control points, at least one for each link, simultaneously generate joint velocity signals.

As it will be shown in the next section, the end-effector is controlled by a different system, called a goal position following system, that is used to compute the desired operational space velocity. This request, again, is translated via Jacobian pseudoinversion into a joint velocity additional term.

All the joint velocity terms are eventually added in order to obtain reference signals for the joint velocity feedback control loops. This solution is summarized by the control scheme represented in Figure 1.

It is worth stressing that because reference signals are finally computed as the summation of all the velocity terms generated from all the control points and from the end-effector, the actual velocity of a given point is different from the desired one. This approach guarantees obstacle avoidance because, if a given $psp$ is really close to an obstacle, it is pushed away even if, in order to do that, the end-effector trajectory is perturbed.

**Remark 1.** This work is based on the hypothesis that a perfect joint velocity tracking is guaranteed. Note that this hypothesis is quite realistic, since in practical cases industrial robots have very fast and effective servomotors. Otherwise, even a simple planned task such as trajectory following would be infeasible.

3. End-effector control

In order to make the end-effector reach the goal position, two proportional controllers are used, one for the position $x_p$ and another for the orientation $x_\alpha$. Let $x_{goal}$ be the goal generalized position (i.e., position and orientation) and $x$ the end-effector generalized position in the operational space, defined as follows:

$$x = \begin{bmatrix} x_p \\ x_\alpha \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \alpha \\ \beta \\ \gamma \end{bmatrix}. \tag{4}$$

The position following controller is described by this simple equation:

$$\dot{x}_p = k_v (x_{goal} - x_p), \tag{5}$$

where $k_v$ is a control parameter used to tune the controller.

For the end-effector orientation, a more complex solution has been developed. It is known that once two different configurations of a rigid body (i.e., the end-effector) are chosen, it is always possible to define a unique screw motion $[MLS94]$ linking the two configurations. We can calculate two parameters: a translation $d$ along the screw axis and a rotation $\theta$ around the screw axis will move the rigid body from the initial generalized position to the final one. We are only interested in the $\theta$ parameters: at each instant, we calculate the screw axis, since we know the actual position and the goal position. Then we calculate the desired angular velocity $\omega$ as a vector having the same direction of the screw axis and norm proportional to the angle $\theta$. Let $M$ be the homogeneous matrix describing the end-effector generalized position with respect to the base frame and $M_{goal}$ the homogeneous matrix describing the generalized position
\( x_{\text{goal}} \): The above described screw motion is represented by the following matrix: \( Q = M_{\text{goal}} M^{-1} \). It can be shown [MLS94] that the screw axis direction is represented by the following unit vector:

\[
\begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \begin{bmatrix} \frac{Q_{32} - Q_{33}}{2 \sin \theta} \\ \frac{Q_{31} - Q_{33}}{2 \sin \theta} \\ \frac{Q_{21} - Q_{32}}{2 \sin \theta} \end{bmatrix}
\]

(6)

where \( \theta \) is calculated as follows:

\[
\begin{align*}
\mathbf{c} &= \frac{Q_{11} + Q_{22} + Q_{33}}{2} \\
s &= \pm \sqrt{\left(\frac{Q_{32} - Q_{33}}{2}\right)^2 + \left(\frac{Q_{31} - Q_{33}}{2}\right)^2 + \left(\frac{Q_{21} - Q_{32}}{2}\right)^2} \\
\theta &= \tan^{-1}(s,c)
\end{align*}
\]

(7)

Note that (6) may become singular (e.g., when the pitch of the skew motion is null). In these cases the screw axis can be computed using different formulae [MLS94, Leg00]. Now we have all the necessary information to build the orientation following controller:

\[
\omega = k_w \theta u_x,
\]

(8)

where \( k_w \) is a control parameter for tuning the orientation controller.

Using this equation and (5), we describe the whole end-effector position controller as follows:

\[
x = \begin{bmatrix} x_p \\ \omega \end{bmatrix} = \begin{bmatrix} kv (x_{\text{goal}} - x_p) \\ k_w \theta u_x \end{bmatrix}.
\]

(9)

The previous equation is simply addressed as controller in Figure 1, and is represented like a unique block scheme. Note that this solution exactly corresponds at having a quadratic attractive potential field (a paraboloid) defined around the goal position and \( \theta \).

However, this goal position following system, when using fixed control parameters, presents some problems. In fact, an aggressive tuning, when the end-effector is far from the goal configuration, leads to excessive joint velocities. On the other hand, a parameter detuning would lead to a sluggish controller and consequently the robot would take a lot of time to reach the goal configuration.

To overcome this problem, it is possible to increase the coefficients \( k_v \) and \( k_w \) when the end-effector is close the goal configuration. In order to do that, we propose a gain scheduling system that tunes the parameters at run time. This solution has been already used in the literature [ZVL 06].

3.1 Gain scheduling

Let \( k_{v0} \) and \( k_{w0} \) be the initial values of the control parameters, \( \rho \), the initial distance between the end-effector and the goal position and \( \rho \) the distance between the end-effector and the goal position. We can tune up the parameters as follows:

\[
\begin{align*}
&\text{if } \rho < \rho_{\text{max}} \\
&k_v = k_{v0} \frac{\rho_{\text{max}}}{\rho} \\
&k_w = k_{w0} \frac{\rho_{\text{max}}}{\rho}
\end{align*}
\]

(10)

The control parameters are anyway bounded by using the \( v_{\text{max}} \) value in order to avoid instability. Using this simple solution allows us to obtain better results, namely, a lower time to execute the task and bounded motor velocities.

In particular, regarding the motor velocities, there is a safety threshold \( q_{\text{max}} \) in order to avoid excessively stressing the joint motors. If a single motor crosses the maximum allowed velocity threshold (in absolute value), the whole vector \( q \) is reduced by multiplying it by this ratio: \( \frac{q_{\text{max}}}{\max (|q|)} \). Using this solution, we do not change the end-effector velocity direction, but only the intensity, thus we do not change the path.

4. Weighted pseudoinverse

In order to make the proposed control architecture work properly, an effective pseudoinverse of the kinematic model is needed. In particular, because the control points are generally distributed along the kinematic chain, and not located at the end of it, we are completely free to decide how to move the joints between a given \( psp \) and the end-effector. Moreover, since the manipulator can be redundant, even when controlling the end-effector, we may have free movements in the joint space. We propose here a solution to address this problem after showing that the classical choice of using the Moore-Penrose pseudoinverse leads to poor results, making the manipulator moving in an unnatural manner.

Let \( n \) be the number of manipulator DOFs, \( P \) a certain point on the \( k \)th link, with \( k \leq n \) and \( \vec{x}_{am} \times 1 \) vector, the desired velocity for the point \( P \) (with \( m \leq n \)).
The most diffused solution in the literature is the classical Moore-Penrose pseudoinverse: let $J$ be the Jacobian matrix of the $P$ point, $J$ is a $m \times n$ matrix defined as follows:

$$
\begin{bmatrix}
\frac{\partial x_1}{\partial q_1} & \cdots & \frac{\partial x_1}{\partial q_k} & 0 & \cdots & 0 \\
\frac{\partial x_i}{\partial q_1} & \cdots & \frac{\partial x_i}{\partial q_k} & \vdots & \ddots & \vdots \\
\frac{\partial x_m}{\partial q_1} & \cdots & \frac{\partial x_m}{\partial q_k} & 0 & \cdots & 0
\end{bmatrix}
$$

(11)

The Moore-Penrose pseudoinverse is then defined as:

$$
J^+ = J^T (J J^T)^{-1}
$$

(12)

This solution minimizes the joint velocity Euclidean norm $\|\dot{q}\|$, thus the joints far from $P$ toward the robot base are moved faster than the nearer ones, while, as shown in Figure 2, the joints from $k$ to $n$ are kept rigid.

The consequence of this behaviour is that bigger joint torques are required because the joints toward the manipulator base see bigger inertias. Evidently, this is not desirable in an industrial perspective considering that it means increased energy consumption and components stress. Moreover, an unnatural manipulator behaviour is obtained because the joints between the considered $P$ and the end-effector are kept blocked. In particular, the second problem is the most important one: it is frequently needed to make the manipulator turn around an obstacle to find the goal position, or pass through a narrow corridor between two obstacles moving around the obstacles to avoid collisions. When using the Moore-Penrose pseudoinverse, the manipulator escapes the $P$ point from the obstacle moving also the end-effector like a rigid body. This behaviour (see Figure 3) strongly reduces the path planner capability of finding a feasible path toward the goal position.

In order to avoid the dynamic model definition, which requires a lot of information on the robot mechanical structure, we have developed a weighted pseudoinverse that leads to suitable results without involving the knowledge of the manipulator dynamic properties, as Figure 2 and Figure 4 show. In order to create the weighted pseudoinverse, we define a $3n \times 3n$ diagonal matrix $W$ ($2n \times 2n$ for a planar manipulator) whose diagonal terms correspond to the elements of a vector $w$.

$$
W = \begin{bmatrix}
w_1 & w_1 & w_1 & \cdots & w_1 \\
w_1 & w_1 & w_1 & \cdots & w_1 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
w_n & w_n & w_n & \cdots & w_n
\end{bmatrix}
$$

(13)
the virtual kinetic energy, and $C = \dot{x} - J(q)\dot{q} = 0$ the equality constraint. We can write the Lagrangian equation as follows:

$$L = O + \lambda^T C,$$

(17)

where $\lambda$ is the Lagrange multiplier vector. Differentiating the previous equation with respect to $q$ and $\lambda$, we obtain the following system

$$\frac{\partial L}{\partial q} = G(q)\dot{q} - J^T \lambda = 0,$$

$$\frac{\partial L}{\partial \lambda} = \dot{x} - J\dot{q} = 0.$$

(18)

(16) follows straightforwardly solving the system.

Using this pseudoinverse, we force the manipulator to move in the same way a kinematic chain having all the masses localized at the joints at the end of the links and at the end-effector would do, pushed at the $P$ point when we assume null joint torques (see Figure 2).

The $w$ vector is a control parameter, the bigger a certain term $w_i$ is, the more the control system moves slowly the $i$-th link in order to get the same Cartesian space velocity. Indeed, it means the $i$th virtual link is heavier and is therefore moved slowly to minimize kinetic energy. When a link at the middle of the manipulator kinematic chain approaches an obstacle, it is pushed away. The proposed pseudoinverse minimizes the virtual kinetic energy so that the obstacle avoidance controller does not actuate the joints between the link close to the obstacle and the end-effector. In this way, this part of the kinematic chain remains completely driven by the end-effector controller that makes the robot bend around the obstacle, improving its capability of turning around obstacles. Now consider the case of one of the terminal links approaching an obstacle. To push away from the obstacle a point along this link, the weighted pseudoinverse strongly moves the last joints and barely actuates the first ones. In this way, the last links turn along the obstacle surface, allowing the manipulator to reach the opposite side of the obstacle. A clear example of both these behaviours can be seen in subsection 5.3. Summarizing, the weighted pseudoinverse avoids the obstacle avoidance system and the end-effector controller to work one against the other.

A simple and effective choice is to set every element of $w$ equal to one, that means that every link has the same virtual mass. This is an effective solution, as it will be shown in the next sections.

It is worth noting again that this solution never involves the knowledge of the robot dynamic parameters and the calculation of the dynamic model.
5. Simulation examples

Several simulations have been performed, in order to test the effectiveness of the proposed technique before implementing it on a real industrial manipulator. In particular, the obstacle avoidance capability has been tested. We tested the capacity of dealing with redundancy with both planar and spatial manipulators, the capability of driving the manipulator through narrow passages and the effect of the weighted pseudoinverse and of the gain scheduling system. During these simulations a perfect joint velocity tracking has been considered. The experimental results proposed in the next section will prove that this assumption is indeed realistic when dealing with industrial servodrives.

5.1 Example 1: obstacle avoidance

The control system has been first tested on a seven DOF spatial manipulator arm, with the same kinematic structure of the Motoman IA20 robot. This manipulator, as shown in Figure 5, can be seen like two spherical robots, the first three joints and the last three joints (spherical wrist) coupled with a joint (the fourth one) located between the two sphere centres.

![Figure 5. The kinematic scheme of a Motoman IA20 manipulator.](image)

Figure 5. The kinematic scheme of a Motoman IA20 manipulator.

Figure 6 shows a simple example of obstacle avoidance: the control system is able to make the manipulator avoid the collision turning around the obstacle.

![Figure 6. An example of obstacle avoidance with a 7R IA20 spatial manipulator.](image)

5.2 Example 2: narrow passages

As a second simulation, the path planner is required to make the same spatial manipulator move between two ellipsoidal obstacles. Figure 7 shows the capability of the control system to avoid collisions and to pass inside the corridor between the two obstacles. The path planner drives the end-effector, avoiding at the same time collisions between all the manipulator links and the obstacles.

![Figure 7. Capability of passing through narrow passages while moving in three dimensional space.](image)

5.3 Example 3: high redundancy

In order to test the capability of dealing with high redundant structures, a seven DOF planar manipulator has been simulated. The robot environment is cluttered with multiple obstacles. The simulation proves the capability of the proposed control system to deal with hyper redundant manipulators, turn around obstacles and deal with several obstacles as shown in Figure 8.

![Figure 8. The path planner is able to drive the manipulator at the goal position using the weighted pseudoinverse.](image)

5.3 Example 4: weighted pseudoinverse

We have also tried to execute the same task of the previous simulation, but using the Moore-Penrose pseudoinverse
instead of the weighted one we have proposed. Without the weighted pseudoinverse, the path planner fails: it cannot drive the end-effector to the goal position.

Figure 9 shows that the manipulator cannot find a feasible path.

5.4 Example 5: gain scheduling

Finally, we have simulated the same task, the first time with fixed control parameters, the second time using the gain scheduling proposed in this paper. In order to test the gain scheduling effect in depth, we have executed the task keeping the maximum joint velocity bounded under 17 rad/s.

Using the gain scheduling, we have strongly detuned the control parameters (more than 50% less) in order to have lower velocities at the beginning of the task execution. We have also reduced the time needed to complete the same task by more than half (see Figure 10 and Figure 11). This happens because, using the gain scheduling, we have higher velocities when we are already close to the goal configuration.

6. Experimental results

The results shown in the previous section have demonstrated the effectiveness of the proposed solution. In order to have a full confirmation of this, we have tested the proposed solution on a real manipulator. The considered robot is the two DOF planar parallel machine shown in Figure 12. The geometrical characteristics of the robot are reported in Table 1 (refer to Figure 12). As Figure 12 shows, the robot has been heavily loaded with a 10 [kg] mass, mounted on the manipulator end-effector (the P point). On the contrary, the links are made of very light aluminium bars, so that their masses can be safely neglected.

![Figure 9. The path planner fails using the Moore-Penrose pseudoinverse.](image)

![Figure 10. Joint velocities without the gain scheduling.](image)

![Figure 11. Joint velocities using the gain scheduling.](image)

![Figure 12. The industrial robot and its kinematic scheme.](image)

| Link | Length |
|------|--------|
| 0    | 14[cm] |
| 1    | 5[cm]  |
| 2    | 14[cm] |
| 3    | 14[cm] |
| 4    | 5[cm]  |

Table 1: Manipulator link lengths.
symmetrical behaviour between direct and reverse motion). The global momentum of inertia of motor rotor and speed reducers is \( I = 0.66 \text{ kg cm}^2 \). Note that it cannot be neglected because, due to the speed reducer effect, its dynamic behaviour is increased \( i^2 = 100 \) times on the manipulator dynamic.

In this context, the aim of loading the end-effector is to test the control system under strongly coupled inertial behaviours (note that they are typical of robotic systems). Indeed, without the load, the manipulator dynamic would be dominated by the motors’ inertia, leading to a simple decoupled behaviour. Figure 12 also shows that the manipulator works in the vertical plane. Hence, the servomotor velocity loops also have to compensate the gravity force. In this framework, a successful result would testify to the effectiveness of the choice of using the inner joint velocity loops in order to drive the robot in its environment.

The velocity control loops are embedded into the servomotor drives - they receive the joint reference velocity signals via the real-time Ethernet bus Powerlink (see [Aut] for details) from the Programmable Logic Controller (PLC). The control algorithm is implemented in the real-time operative system Automation Runtime (see [Aut] for details) that runs on the PLC accordingly to the PLCopen standard [PLC]. The control cycle is of 0.8 [ms], so that the velocity reference signals are refreshed with the same frequency.

![Image](https://example.com/image.png)

**Figure 13.** The obstacle in the robot environment.

It is worth stressing that all the components employed are off-the-shelf products [Aut]. This proves that the proposed technique is indeed suitable for industrial applications. A circular obstacle is placed in the manipulator operational space as Figure 13 shows. The considered obstacle size is a circle whose diameter is of 7 [cm] and its centre is at the position \( x = 0 \text{ cm}, y = 12.8 \text{ cm} \); see Figure 14. It is worth stressing that this measure is bigger than the actual obstacle size because the end-effector size also has to be considered in order to avoid collisions. Note that considering a bigger obstacle (increase of the load size) and an ideal (a point) end-effector, or the real obstacle and the real (with the load size) end-effector, leads to the same results.

The robot is required to reach three points in its operational space, represented by the coloured diamonds in Figure 14, where also the ideal (obstacle free) and the actual trajectory are shown. The first set of tuning parameters has been chosen with \( k_w = 0.08 \) and \( k_v = 0.5 \) (note that here, having a two DOF robot, \( k_w \) is not needed). The gain scheduling parameter has been chosen such that the gain can increase at most ten times its initial value, i.e., \( v_{max} = 10 \). The minimum distance under which the potential field starts acting on the manipulator is \( p_0 = 1 \text{ cm} \).

The proposed path planner successfully drives the end-effector to all the goal positions moving around the obstacles as shown in Figure 14. It is worth noting that, as shown in Figure 15, despite the big load and the gravity, the actual velocities and the reference ones are practically undistinguishable, except for the small noise due to numerical differentiation of the encoder signals.

In order to analyse the proposed solution in depth, a more aggressive tuning has been tested, increasing both \( k_w \) and \( k_v \) ten times. The results are reported in Figure 16. The time required to execute the task is greatly reduced, but again, despite the increased dynamics, the velocity tracking error is small.

It is interesting to note that when strongly increasing the tuning parameter, the manipulator exhibits an underdamped behaviour when it is very close to the obstacle, because of the strong potential field that locally behaves like a stiff spring.

![Image](https://example.com/image.png)

**Figure 14.** The obstacle (black line), the obstacle-free trajectory (blue line) and the actual trajectory (red line).
points along the manipulator kinematic chain. Using the weighted pseudoinverse, the manipulator capability of turning around obstacles and moving in narrow passages is increased.

Once a potential field formulation was chosen, we tested the control system on both planar and spatial redundant manipulators with successful results.

Finally, the proposed solution has been experimentally tested on a real industrial robot by implementing the algorithm on a motion control industrial PLC driving two industrial brushless servomotors. The obtained results confirm its effectiveness. It is remarkable that the performance remains satisfactory also when the control parameters span over a wide range.

8. Acknowledgement

The authors gratefully thanks Mr. Michele Pagani for his valuable help.

The research leading to these results has received funding from the European Community’s Seventh Framework Programme (FP7/2007-2013) under Grant Agreement n222107.

9. References

[1] S. Akishita, T. Hisanobu, and S. Kawamura. Fast path planning available for moving obstacle avoidance by use of laplace potential. In Proceedings of the 1993 IEEE/WRSJ International Conference on Intelligent Robots and Systems, pages 26–30, Yokohama, Japan, July 1993.

[2] K. Althoefer, L.D. Seneviratne, P. Zavlangas, and B. Krekelberg. Fuzzy navigation for robotic manipulators. International Journal of Uncertainty and Knowledgebased Systems, 6:179–188, April 1998.

[3] B&R Automation. www.br-automation.com.

[4] W. Cho and D. Kwon. A sensor based obstacle avoidance for redundant manipulator using a velocity potential function. In IEEE International Workshop on Robot and Human Communication, 1996.

[5] E. Dyllong and A. Visioli. Planning and real-time modifications of a trajectory using spline techniques. Robotica, 21:475–482, 2003.

[6] K. Ikeda, H. Tanaka, T. Zhang, M. Minami, and Y. Mae. On-line optimization of avoidance ability for redundant manipulator. In Proceedings of the 2006 IEEE/RSJInternational Conference on Intelligent Robots and Systems, Beijing, China, 2006.

[7] O. Khatib. Dynamic control of manipulator in operational space. In 6th CISMIFToMM Congress Theory Machines Mechanism, pages 156–161, New Delhi, December 1983.

Figure 15. The actual joint velocities (red and green lines) and the reference joint velocities (black lines) with the first set of tuning parameters.

Figure 16. The joint actual velocities (red and green lines) and the joint reference velocities (black lines) with the second set of tuning parameters.

7. Conclusions

In this paper, a path planner suitable for industrial applications has been proposed. It is capable of driving the manipulator in a cluttered environment toward the desired position while avoiding collisions.

The proposed technique is computationally light and solely based on the robot kinematic model. Hence, it is suitable for on line applications and does not require the manipulator dynamic model. This solution is thus very useful when, as is usual in industrial contexts, it is difficult to model the manipulator dynamics or to control the joint torques.

The proposed solution works properly both with redundant and non-redundant manipulators.

A new pseudoinverse Jacobian has been proposed in order to solve the inverse kinematic problem for several
[8] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. The International Journal of Robotics Research, 5(1):90–98, 1986. 24
[9] J. O. Kim and P. Khosla. Real-time obstacle avoidance using harmonic potential functions. IEEE Transactions on Robotics and Automation, 8:338–349, June 1992.
[10] S. M. Khansari-Zadeh and A. Billard. A dynamical system approach to real time obstacle avoidance. Autonomous Robots, 32(4):433–454, 2012.
[11] S. M. Khansari-Zadeh and A. Billard. Realtime avoidance of fast moving objects: A dynamical system-based approach. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Algarve, Portugal, 2012.
[12] J. C. Latombe. Robot motion planning. Kluwe Academic Publisher, 1991.
[13] S. M. LaValle. Planning Algorithms. Cambridge University Press, 2006.
[14] G. Legnani. Robotica industriale. ed. Ambrosiana, 2000.
[15] T. Lozano-Prez. Spatial planning: a configuration space approach. IEEE Transactions on Computers, C-32(2):108–120, 1983.
[16] D. Li, H. Wang, and B. You. Path planning research and simulation of planar 3r redundant robot. In 4th IEEE Conference on Industrial Electronics and Applications, Xi’an China, 2009.
[17] S. Lahouar, S. Zeghloul, and L. Romdhane. Real-time path planning for multi-dof manipulators in dynamic environment. International Journal of Advanced Robotic Systems, 3(2):125–132, 2000.
[18] A. A. Maciejewski and C. A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. International Journal on Robotics Research, 4:109–117, Fall 1985.
[19] R. M. Murray, Z. Li, and S. S. Sastry. A Mathematical Introduction to Robotic Manipulation. CRC Press, 1994.
[20] C. Passi. Commande en temps reel de robots manipulateurs redondants avec evitamentd’obstacles mobiles. PhD thesis, University Pierre et Marie Curie, Paris, 2001.
[21] PLCopen. www.plcopen.org.
[22] F. Padula and V. Perdereau. A new pseudoinverse for manipulator collision avoidance. In 18th IFAC World Congress, Milano, 2011.
[23] V. Perdereau, C. Passi, and M. Drouin. Real-time control of redundant robotic manipulator for mobile obstacle avoidance. Robotics and Autonomous Systems, 41:41–59, 2002.
[24] A. Visioli. Trajectory planning of robot manipulators by using algebraic and trigonometric splines. Robotica, 18:611–631, 2000.
[25] R. Volpe and P. Khosla. Manipulator control with superquadratic artificial potential functions: Theory and experiments. IEEE Transactions Systems, Man, and Cybernetics, 20:1423–1436, December 1990.
[26] G. Zilliani, A. Visioli, and G. Legnani. Gain scheduling for hybrid force/velocity control in contour tracking task. International Journal of Advanced Robotic Systems, 3(4):367–374, 2006.