Robotica (2013) volume 31, pp. 1143–1153. © Cambridge University Press 2013
doi:10.1017/S0263574713000404

Walk-through programming for robotic manipulators based on admittance control

Luca Bascetta*, Gianni Ferretti, Gianantonio Magnani and Paolo Rocco

Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Piazza Leonardo da Vinci 32, 20133 Milano, Italy

(Accepted April 11, 2013. First published online: May 14, 2013)

SUMMARY

The present paper addresses the issues that should be covered in order to develop walk-through programming techniques (i.e. a manual guidance of the robot) in an industrial scenario. First, an exact formulation of the dynamics of the tool the human should feel when interacting with the robot is presented. Then, the paper discusses a way to implement such dynamics on an industrial robot equipped with an open robot control system and a wrist force/torque sensor, as well as the safety issues related to the walk-through programming. In particular, two strategies that make use of admittance control to constrain the robot motion are presented. One slows down the robot when the velocity of the tool centre point exceeds a specified safety limit, the other one limits the robot workspace by way of virtual safety surfaces. Experimental results on a COMAU Smart Six robot are presented, showing the performance of the walk-through programming system endowed with the two proposed safety strategies.

KEYWORDS: Force control; Control of robotic systems; Haptic interfaces; Man–Machine systems; Automation.

1. Introduction

Industrial robots are complex and powerful machines, able to execute a variety of different tasks with high speed and accuracy. Nevertheless, they still have a low degree of autonomy and adaptability, always needing huge efforts of a human operator to learn new tasks or tune existing ones

A manual generation of a finishing path, for example, is a very complex and tedious task, taking up to ten weeks to create the part program to deburr a single aluminium wheel.[21] This is a rather common and crucial aspect in a variety of different robotic tasks, preventing robots from spreading in small and medium enterprises, where mid/low lot size production does not allow for such a costly and time-consuming set-up. Generally, the fact that the programming phase is complex and time demanding is considered one of the major weaknesses of today's industrial robotic systems.

The introduction of a programming paradigm, based on a physical interaction between the human operator and the robot, represents a step forward in making the programming phase simple, intuitive and faster, and even in promoting an increased autonomy and cognitive ability of the robotic system.

In this paradigm, the so-called walk-through programming,[19] the human operator plays the role of a teacher that physically moves ("walks") the robot through the desired positions within the robot's working envelope. During this phase, the robot's controller may scan and store coordinate values.[15] At the end of this operation, the robot controller has recorded all the significant points in the trajectory demonstrated by the human, and is thus able to interpolate it and play it back.

The walk-through programming obviously entails implementation and safety issues.[1] Concerning the implementation, the walk-through programming usually requires significant changes in the robot control software that can be accomplished only with open robot control platforms. On the other side, safety is an important issue as the walk-through method requires the operator to be within the robot's working envelope with the robot's controller energised: the robot programmer is thus in a potentially hazardous situation.[15]

For the sake of completeness, it must be noticed that the walk-through programming is of particular interest in robotic surgery as well.[17, 18, 20] In fact, it allows for a synergy between the surgeon and the robot: the robot provides geometric accuracy and increases safety, preventing the execution of operations outside a predefined safe region; the surgeon guides the robot using his or her superior human senses and understanding of the overall situation to perform the surgery.

A walk-through programming technique, based on admittance control, is presented here. The paper develops the ideas already introduced in refs. [6–8, 24] for arm-manipulator coordination in load sharing, and in refs. [11, 12] for welding and spray painting.

Other examples of the walk-through programming can be found in the scientific literature and in a few industrial products. In ref. [13], a complete architecture that allows to program a robot using a physical human–robot interaction is presented. This work is very comprehensive, but does not take into account the concept of virtual tool, i.e. how to give

* Corresponding author. E-mail: luca.bascetta@polimi.it

[1] In this paper, we use the term "safety", with a slight abuse of notation, to refer to control functionalities that can improve the safety in the human–robot interaction, but that cannot replace certified safety devices.

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press

to the operator the impression to move the robot as moving the tool used to accomplish the task. On the other hand, in refs. [10, 11], the authors propose a walk-through teaching method for a welding process that is very similar to the one discussed here. The present paper, in addition, proposes a nonlinear model of the dynamics of the virtual tool, giving to the reader the opportunity to choose between this model and a set of linear admittance filters. Furthermore, in refs. [9, 14], an advanced technique for programming by touch is presented. This methodology, however, is conceived for a redundant manipulator with torque sensors at the joints, and cannot be easily applied to an industrial manipulator with a force/torque sensor at the end-effector.

Concerning the industrial state-of-the-art, two products exist. One is the so-called manual guidance device,[23] a low cost programming by demonstration device based on an optical mouse with 6 d.o.f. that can be used to program COMAU robots. Though this device is very intuitive and can be placed on the robot, or attached to the teach pendant, or even kept in the teacher's hands, it does not allow for a physical human–robot interaction, thus being conceptually different from the solution proposed in this paper.

The other one is the ABB programming handle,[22] a device based on a force/torque sensor and two handles that can be used to move the robot end-effector. Though this approach seems to be very similar to the one proposed here, the ABB product does not consider the concept of virtual tool, and it seems that it does not allow the execution of complex paths, combining translations and rotations at the same time.

Experimental evidence of the good performance and versatility of the approach is given, with reference to a set-up composed of a COMAU Smart Six industrial robot equipped with an ATI force/torque sensor.

The paper is organised as follows. In Section 2, the motion equations of the virtual tool are introduced. Section 3 covers all the aspects related to the development of the walk-through programming technique in an industrial scenario. Section 4 describes the robotic cell set-up and shows the results of an experimental validation of the proposed control approach.

2. Virtual Tool Dynamics

In the walk-through programming, the human operator plays the role of a teacher that physically walks the robot through the desired path. Furthermore, in an industrial application, like metal finishing or painting operations, the physical interaction between the human operator and the robot should be conceived in such a way that the teacher has the impression to grab a real tool, e.g. a deburring tool or a spray gun, instead of the robot end-effector.

The role of the control system is thus to accommodate for the motion commanded by the teacher, mimicking the same dynamic behaviour of the real tool, i.e. behaving like a virtual tool that exhibits the same mechanical properties of the real tool. Considering again an industrial application, the robot is equipped with a force/torque sensor mounted at the end-effector, and a handle connected to the sensor. The human operator grabs the handle in order to transmit the desired motion to the manipulator. The robot, instead, controls the handle trajectory in such a way that the human has the impression to move an object, having the same weight and inertial properties of the real tool, instead of the manipulator. For this reason, the name virtual tool is used to refer to the imaginary object, whose mechanical properties are the same as the real tool, that is emulated by the robot.

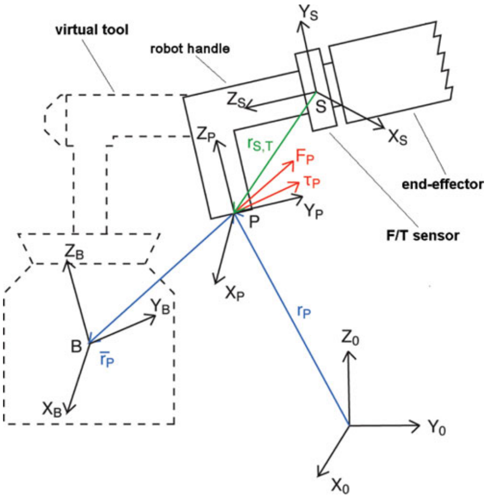


Fig. 1. (Colour online) Frames used to derive the dynamics of the virtual tool.

The first step towards the development of the walk-through programming technique is thus the derivation of the exact Newton–Euler dynamic equations of the virtual tool. Since there are several bodies involved in the system (robot end-effector, force/torque sensor, real tool and virtual tool), it is particularly important to provide a rigorous and unambiguous definition of the frames and vectors needed to define the said dynamics.

Consider Fig. 1, where the following frames are reported: (x_0, y_0, z_0), the absolute frame; (x_S, y_S, z_S), a frame attached to the force/torque sensor; (x_P, y_P, z_P), a frame attached to the actual tool, with the origin located where the operator grabs the tool; (x_B, y_B, z_B), a frame attached to the virtual tool, with the origin located in its centre of mass.

The dynamic model is written using the following coordinates:[2]

$$\mathbf{r}_P = \begin{bmatrix} x_P \\ y_P \\ z_P \end{bmatrix} \quad \text{and} \quad \mathbf{p}_P = \begin{bmatrix} e_{P,0} \\ e_{P,1} \\ e_{P,2} \\ e_{P,3} \end{bmatrix}, \tag{1}$$

which represent the coordinates of the origin of frame P with respect to the absolute frame and the relative orientation of the same frames, respectively.

The equations of motion of the virtual tool with respect to frame P can thus be written as

$$\mathbf{f}_P = m[\mathbf{a}_P - \mathbf{g} + \boldsymbol{\varepsilon}_P \times \mathbf{r}_{B,P} + \boldsymbol{\omega}_P \times (\boldsymbol{\omega}_P \times \mathbf{r}_{B,P})], \tag{2}$$

$$\boldsymbol{\tau}_P = \mathbf{I}_B \boldsymbol{\varepsilon}_P + \boldsymbol{\omega}_P \times (\mathbf{I}_B \boldsymbol{\omega}_P) + \mathbf{r}_{B,P} \times \mathbf{f}_P, \tag{3}$$

[2] Notice that the orientation in (1) is expressed in terms of the four Euler parameters (unit quaternion): while being redundant, this representation avoids the so-called *gimbal lock*, i.e. the situation when two axes are aligned and the system loses one rotational d.o.f.

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press

where the following symbols have been introduced:

- m, mass of the virtual tool;
- $\mathbf{I}_B$, inertia tensor of the virtual tool with respect to a frame B, with origin in the centre of mass of the virtual tool, expressed in the absolute frame;
- $\mathbf{g}$, gravity acceleration vector, expressed in the absolute frame;
- $\mathbf{f}_P$, $\boldsymbol{\tau}_P$, vectors of the forces and moments, respectively, applied by the human operator to the actual tool, expressed in the absolute frame;
- $\mathbf{a}_P$, absolute acceleration of the origin of frame P, expressed in the absolute frame;
- $\boldsymbol{\omega}_P$ and $\boldsymbol{\varepsilon}_P$, angular velocity and angular acceleration of frame P, expressed in the absolute frame;
- $\mathbf{r}_{B,P}$, position of the origin of frame B with respect to the origin of frame P, expressed in the absolute frame.

Note that in the equations of motion of the virtual tool no translation or rotational viscous friction has been introduced. In fact, considering that the tool moves through the air and the typical velocity of the human hands is quite low, the contribution of viscous friction is negligible.

In Eqs. (2)–(3), the following kinematic terms have been introduced as well:

- the absolute velocity of the origin of frame P, expressed in the absolute frame

$$\mathbf{v}_P = \frac{d\mathbf{r}_P}{dt} = \begin{bmatrix} v_{P,x} \\ v_{P,y} \\ v_{P,z} \end{bmatrix};$$

- the absolute acceleration of the origin of frame P, expressed in the absolute frame

$$\mathbf{a}_P = \frac{d\mathbf{v}_P}{dt} = \begin{bmatrix} a_{P,x} \\ a_{P,y} \\ a_{P,z} \end{bmatrix};$$

- the angular velocity of frame P, expressed in the absolute frame

$$\boldsymbol{\omega}_P = 2\begin{bmatrix} -e_{P,1} & e_{P,0} & -e_{P,3} & e_{P,2} \\ -e_{P,2} & e_{P,3} & e_{P,0} & -e_{P,1} \\ -e_{P,3} & -e_{P,2} & e_{P,1} & e_{P,0} \end{bmatrix}\frac{d\mathbf{p}_P}{dt} = 2\mathbf{E}\frac{d\mathbf{p}_P}{dt};$$

- the angular acceleration of frame P, expressed in the absolute frame

$$\boldsymbol{\varepsilon}_P = 2\frac{d\mathbf{E}}{dt}\frac{d\mathbf{p}_P}{dt} + 2\mathbf{E}\frac{d^2\mathbf{p}_P}{dt^2};$$

- the rotation matrix of frame P with respect to the absolute frame

$$\mathbf{R}_P = 2[\mathbf{n}_{x,P}\mathbf{n}_{y,P}\mathbf{n}_{z,P}],$$

where

$$\mathbf{n}_{x,P} = \begin{bmatrix} e_{P,0}^2 + e_{P,1}^2 - \frac{1}{2} \\ e_{P,1}e_{P,2} + e_{P,0}e_{P,3} \\ e_{P,1}e_{P,3} - e_{P,0}e_{P,2} \end{bmatrix},$$

$$\mathbf{n}_{y,P} = \begin{bmatrix} e_{P,1}e_{P,2} - e_{P,0}e_{P,3} \\ e_{P,0}^2 + e_{P,2}^2 - \frac{1}{2} \\ e_{P,2}e_{P,3} + e_{P,0}e_{P,1} \end{bmatrix},$$

$$\mathbf{n}_{z,P} = \begin{bmatrix} e_{P,1}e_{P,3} + e_{P,0}e_{P,2} \\ e_{P,2}e_{P,3} - e_{P,0}e_{P,1} \\ e_{P,0}^2 + e_{P,3}^2 - \frac{1}{2} \end{bmatrix}$$

and

$$e_{P,0}^2 + e_{P,1}^2 + e_{P,2}^2 + e_{P,3}^2 = 1.$$

Note that in Eqs. (2)–(3) the inertia tensor of the virtual tool, $\mathbf{I}_B$, is time varying, as it is expressed with reference to the absolute frame. Alternatively, the equations can be rewritten by projecting all the quantities onto frame P attached to the real, and thus also to the virtual, tool as follows:

$$^P\mathbf{f}_P = m\left[^P\mathbf{a}_P - {^P\mathbf{g}} + {^P\boldsymbol{\varepsilon}_P} \times {^P\mathbf{r}_{B,P}} + {^P\boldsymbol{\omega}_P} \times \left({^P\boldsymbol{\omega}_P} \times {^P\mathbf{r}_{B,P}}\right)\right], \tag{4}$$

$$^P\boldsymbol{\tau}_P = {^P\mathbf{I}_B}\,{^P\boldsymbol{\varepsilon}_P} + {^P\boldsymbol{\omega}_P} \times \left({^P\mathbf{I}_B}\,{^P\boldsymbol{\omega}_P}\right) + {^P\mathbf{r}_{B,P}} \times {^P\mathbf{f}_P}. \tag{5}$$

In this formulation the inertia tensor $^P\mathbf{I}_B$ is used, which is now time invariant.

Finally, notice that the frame used to command the robot motion is the sensor frame S. Specifically, the motion will be assigned by commanding the motion of the origin of frame S with respect to the absolute frame, and the rotation matrix of the said frame. The following parameters of the real tool are introduced:

- $^S\mathbf{r}_{S,P}$, position of the origin of frame P with respect to the origin of frame S, expressed in frame S;
- $^S\mathbf{R}_P$, rotation matrix of frame P with respect to sensor frame S.

The position and orientation of frame S are thus readily given by

$$\mathbf{r}_S = \mathbf{r}_P - \mathbf{R}_P\left(^S\mathbf{R}_P\right)^T {^S\mathbf{r}_{S,P}} \qquad \mathbf{R}_S = \mathbf{R}_P\left(^S\mathbf{R}_P\right)^T.$$

3. Walk-Through Programming Using Admittance Control

In order to ensure that the robot behaves as the virtual tool, in response to the teacher's forces, a controller that enforces the dynamics described by Eqs. (4)–(5) must be devised. It must be noticed, however, that Eqs. (4)–(5) are a set of six nonlinear and coupled differential algebraic equations whose integration in the real-time robot controller is a challenging task. Furthermore, considering that the teaching phase is characterised by low linear, and even lower, angular velocities and accelerations, the couplings between the Newton and Euler dynamics can be neglected.

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press

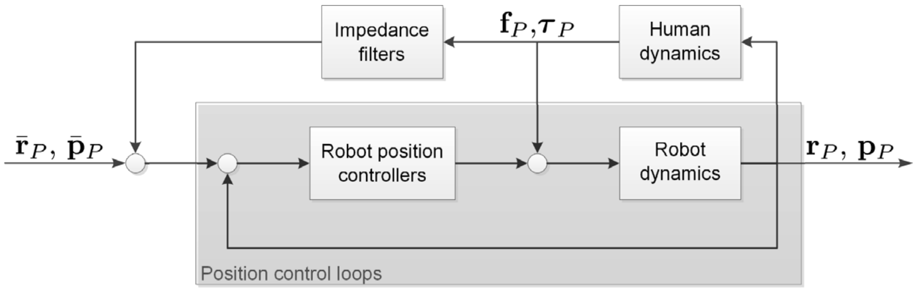

Impedance filters
$\mathbf{f}_P, \boldsymbol{\tau}_P$
Human dynamics
$\bar{\mathbf{r}}_P,\ \bar{\mathbf{p}}_P$
Robot position controllers
Robot dynamics
$\mathbf{r}_P,\ \mathbf{p}_P$
Position control loops

Fig. 2. The position controlled system with the admittance controller.

The dynamic equations in (4)–(5) can thus be simplified as follows:

$$ {}^P\mathbf{f}_P + m\,{}^P\mathbf{g} = m\,{}^P\mathbf{a}_P, \tag{6}$$

$$ {}^P\boldsymbol{\tau}_P + {}^P\mathbf{f}_P \times {}^P\mathbf{r}_{B,P} = {}^P\bar{\mathbf{I}}_B\,{}^P\boldsymbol{\varepsilon}_P, \tag{7}$$

where the inertia tensor ${}^P\mathbf{I}_B$ has been approximated with a diagonal (constant) matrix

$$ {}^P\bar{\mathbf{I}}_B = \operatorname{diag}\left({}^P\bar{I}_{B_i},\ i = 1\ldots3\right). $$

We can now state that the relations between the virtual tool position and orientation, and the forces and moments exerted by the teacher, are a set of six independent double integrators. The real-time integration of such unstable system, however, could lead to numerical problems that can be easily avoided introducing a dissipative term, i.e. a linear viscous friction, in the linear and rotational motion. As explained later, this makes the relations between forces/moments and velocities passive, contributing to the system stability and, possibly, improving the precision achievable with the walk-through programming technique.

Equations (6)–(7) can be then rewritten as

$$ {}^P\mathbf{f}_P + m\,{}^P\mathbf{g} = m\,{}^P\mathbf{a}_P + D_t\,{}^P\mathbf{v}_P, $$

$$ {}^P\boldsymbol{\tau}_P + {}^P\mathbf{f}_P \times {}^P\mathbf{r}_{B,P} = {}^P\bar{\mathbf{I}}_B\,{}^P\boldsymbol{\varepsilon}_P + D_r\,{}^P\boldsymbol{\omega}_P, $$

where D_t and D_r are the viscous friction matrix for the translational and rotational motion, respectively. Applying the Laplace transform to the previous equations yields

$$ {}^P\mathbf{r}_P = \mathcal{L}^{-1}\left\{\mathbf{Z}_t^{-1}(s)\,\mathbf{F}_P(s)\right\} $$

$$ \mathbf{p}_P = \mathcal{L}^{-1}\left\{\mathbf{Z}_r^{-1}(s)\,\mathbf{T}_P(s)\right\}, $$

where

$$ \mathbf{Z}_t^{-1}(s) = \operatorname{diag}\left(\frac{1}{ms^2 + D_{t_i}s},\ i = 1\ldots3\right), $$

$$ \mathbf{Z}_r^{-1}(s) = \operatorname{diag}\left(\frac{1}{{}^P\bar{I}_{B_i}s^2 + D_{r_i}s},\ i = 1\ldots3\right) $$

are two diagonal matrices of translational and rotational admittance filters, and

$$ \mathbf{F}_P(s) = \mathcal{L}\left\{{}^P\mathbf{f}_P + m\,{}^P\mathbf{g}\right\}, $$

$$ \mathbf{T}_P(s) = \mathcal{L}\left\{{}^P\boldsymbol{\tau}_P + {}^P\mathbf{f}_P \times {}^P\mathbf{r}_{B,P}\right\} $$

are vectors of Laplace transforms of the corresponding variables.

3.1. Admittance control: implementation and safety issues

The easiest way to implement an admittance control strategy on an industrial robot controller is by closing an external loop outside the position control loops: the admittance controller, fed by the measurements of force and moments acquired by a force/torque sensor mounted at the end-effector, yields modifications to the Cartesian (position and orientation) set-points in order to guarantee the prescribed compliant behaviour. If the industrial controller does not allow to directly feed the Cartesian position/orientation set-points, a kinematic inversion has to be added in order to convert the outputs of the admittance filters into the joint position set-points.

Consider thus the block diagram in Fig. 2: the part enclosed in the grey box represents the industrial position control, whose set-points are updated based on the force measured at the end-effector. The dynamics of the human operator have been also represented[7,8] as he or she closes a further loop, outside the industrial control, that, in particular situations, can lead to instability. In order to simplify the analysis, it is sometimes assumed, see e.g. refs. [7, 8], that the dynamics of the human operator are relatively slow with respect to the dynamics of the position controlled robot.

Assuming anyway that the dynamics of the human are passive, a sensible way to design the admittance filter is to guarantee a passive mapping between sensed forces and resulting velocities on the robot side.

Then, denoting with $\bar{\mathbf{r}}_P$, $\bar{\mathbf{p}}_P$ the reference trajectory, and assuming the dynamics of the closed-loop position controlled robot to be sufficiently fast, it follows

$$ \mathbf{r}_P \approx \bar{\mathbf{r}}_P - \mathbf{Z}_t^{-1}\mathbf{f}_P \qquad \mathbf{p}_P \approx \bar{\mathbf{p}}_P - \mathbf{Z}_r^{-1}\boldsymbol{\tau}_P $$

or

$$ \mathbf{f}_P \approx \mathbf{Z}_t\left(\bar{\mathbf{r}}_P - \mathbf{r}_P\right) \qquad \boldsymbol{\tau}_P \approx \mathbf{Z}_r\left(\bar{\mathbf{p}}_P - \mathbf{p}_P\right), $$

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press

which shows that an impedance relation between applied forces (moments) and position (orientation) errors is, at least approximately, obtained.

As far as safety aspects are concerned, industrial robots are designed to work in a highly structured environment, performing fast and accurate movements in an area that is rigidly separated, by way of safety fences, from the areas occupied by humans. Robots safety requirements for industrial environments are regulated by the international standard ISO 10218-1.[5]

As already pointed out, the walk-through programming entails important safety issues, as the operator is within the robot working envelope with the robot controller energised. In particular, the safety aspects that have to be considered concern the ergonomic design and the safe motion of the robot during the human–robot interaction.

Considering a single human operator interacting with a single robot, the following safety procedure should be enforced:

- *Activation of drive power.* Drive power has to be turned on from outside the robot area, after checking that the area is not occupied by anyone (as described by general safety rules).
  Once the drive power has been activated, the operator can start the walk-through programming.
- *Walk-through programming.* The operator should grab and activate a three-position enabling device, compliant with the requirements of Section 5.8.3 in ref. [5], interlocked with the safety devices of the cell/robot control: as long as he or she holds the device in a centre-enabled position, robot motion is allowed.
  The robot motion is commanded by the teacher through the Teach Pendant or the device adopted for the walk-through programming.

According to ref. [5], the speed of the end-effector mounting flange and of the tool centre point shall not exceed 250 mm/s, and the device adopted for the walk-through programming shall be located close to the end-effector.

In the next sections, two further functionalities, which can be developed exploiting the admittance control framework, are introduced. The former, presented in Section 3.2, uses a variable viscous friction to enhance the motion precision and to introduce a safety limit on the maximum tool centre point velocity. The latter, described in Section 3.3, introduces virtual barriers to separate the human from the robot end-effector.

Notice, however, that these functionalities, though useful to improve the safety in the human–robot interaction, cannot in any case replace the safety procedure detailed previously. In fact, they can only prevent a possible collision occurring at the robot end-effector, and they cannot guarantee any protection if a failure at the force/torque sensor occurs.

3.2. *Introducing a nonlinear viscous friction*

As explained at the beginning of this section, the introduction of a dissipative term in the equations of motion of the virtual tool makes the relations between forces/moments and velocities passive, contributing to the system stability.

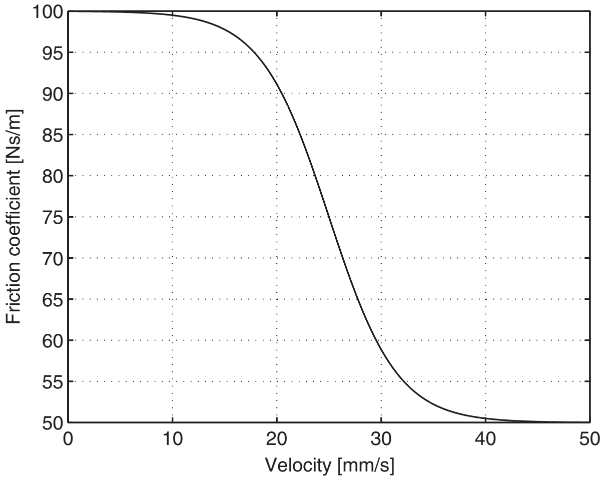


Fig. 3. An example of the variable friction coefficient characterised by a maximum viscous friction, $D_{\max}$, of 100 Ns/m, a minimum viscous friction, $D_{\min}$, of 50 Ns/m, a velocity $v_{\max}$ of 40 mm/s and a velocity $v_{\min}$ of 10 mm/s.

Furthermore, as shown in ref. [16], the adoption of a variable friction coefficient can even enhance the motion precision. In fact, for fine positioning, i.e. for a motion characterised by a low velocity, a high damping is beneficial, but for gross motion, characterised by a high velocity, a low damping makes motion easier.

In principle, this argument holds for the translational and rotational motion as well. In practice, however, a gross rotational motion is quite rare. For this reason, though the methodology presented here is general (i.e. it can be used for the translational and rotational viscous friction coefficients), we consider only the case of a variable translational viscous friction coefficient.

To guarantee continuity and differentiability, the relation between the absolute value of the velocity and the friction coefficient is represented by a logistic function as follows:

$$D\left(|v|\right) = \frac{\alpha_1}{1 + e^{\alpha_2(|v|-\alpha_3)}} + \alpha_4. \tag{8}$$

Using the absolute value of the velocity ensures that velocity vectors having the same magnitude give rise to the same friction coefficient, independently of the direction of the vector. In other words, the curve representing the relation between the friction coefficient D and the velocity v is symmetric with respect to the y axis. For this reason, in the following we consider only the positive half of this curve.

The shape of the positive half ($v \geq 0$) of the logistic function can be characterised through the following (Fig. 3) coefficients:

- $D_{\max}$ and $D_{\min}$, the maximum and minimum values of the friction coefficient, respectively;
- $v_{\max}$ and $v_{\min}$, the velocities corresponding to a value of the friction coefficient of $101\% D_{\min}$ and $99\% D_{\max}$, respectively.

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press

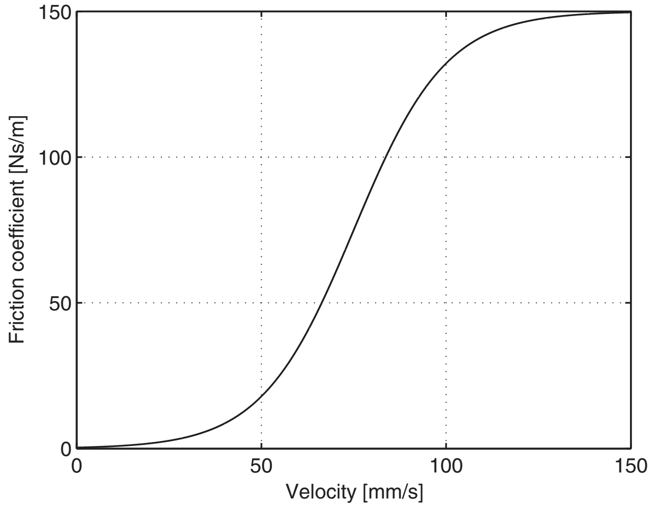


Fig. 4. An example of the variable friction coefficient used to actively enforce safety. The curve is characterised by a safety friction, D_{safe}, of 150 Ns/m, a maximum acceptable velocity, v_{safe}, of 75 mm/s and a range of velocities Δv_{safe} of 25 mm/s.

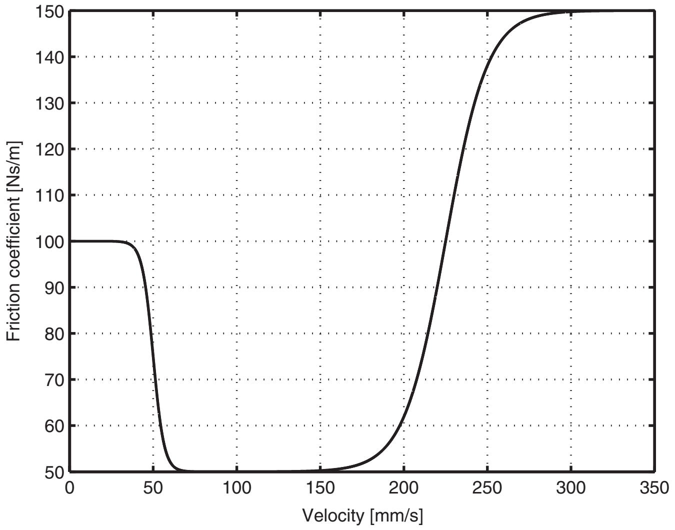


Fig. 5. An example of the variable friction coefficient that aims at improving the accuracy at low velocity and enforcing safety at high velocity.

Once these four coefficients have been selected, the values of α_i can be determined as follows:

$$\alpha_1 = D_{\max} - D_{\min}, \qquad \alpha_3 = \frac{v_{\min} + v_{\max}}{2},$$
$$\alpha_4 = D_{\min},$$

and

$$\alpha_2 = \frac{1}{v_{\max} - \alpha_3} \ln\left(\frac{100\alpha_1 - \alpha_4}{\alpha_4}\right) \quad \text{or}$$
$$\alpha_2 = \frac{1}{v_{\min} - \alpha_3} \ln\left(\frac{\alpha_1 + \alpha_4}{99\alpha_1 - \alpha_4}\right),$$

where the first relation is obtained imposing $D(v_{\min}) = 99\% D_{\max}$, the second one imposing $D(v_{\max}) = 101\% D_{\min}$.

Since D_t is a diagonal matrix, relation (8) is used to independently determine the value of each friction coefficient with respect to the corresponding component of the translational velocity.

The same principle can be used to actively enforce safety, exerting a rapidly increasing friction force as the tool centre point velocity approaches a specified velocity safety limit. In this case, the logistic function has the following expression (Fig. 4):

$$D(|v|) = \frac{\beta_1}{1 + e^{-\beta_2(|v| - \beta_3)}} \tag{9}$$

and is characterised by

- v_{safe}, the maximum acceptable tool centre point velocity;
- D_{safe}, the value of the friction coefficient to be enforced when the tool centre point velocity exceeds v_{safe};
- Δv_{safe}, the range of velocities around v_{safe} (from $v_{\text{safe}} - \Delta v_{\text{safe}}$ to $v_{\text{safe}} + \Delta v_{\text{safe}}$), where the friction coefficient is monotonously increasing from 0 to D_{safe}.

The coefficients β_i are given by the following relations:

$$\beta_1 = D_{\text{safe}}, \qquad \beta_2 = \frac{2}{\Delta v_{\text{safe}}}, \qquad \beta_3 = v_{\text{safe}}.$$

Finally, the two effects previously described can be combined, giving rise to the friction/velocity relation (Fig. 5)

$$D(|v|) = \frac{\alpha_1}{1 + e^{\alpha_2(|v| - \alpha_3)}} + \alpha_4 + \frac{\beta_1 - \alpha_4}{1 + e^{-\beta_2(|v| - \beta_3)}}$$

that allows to improve the motion accuracy at low velocity and to enforce a safety velocity limit at high velocity.

3.3. *Introducing virtual safety surfaces*

A variable friction coefficient allows to enforce a limitation on the maximum tool centre point Cartesian velocity, slowing down the robot motion when the operator exceeds the safety limit.

Besides relating the safety to the robot velocity, however, in many tasks the introduction of a safety relation depending on the position of the tool centre point is extremely important. Considering, for example, the walk-through programming in the context of metal finishing operations, a virtual safety fence surrounding the robot workstation establishes a barrier between the human body and the robot workspace, so that only the human arms are within the allowed robot's working envelope.

Following this idea, another way to enforce safety, in the framework of the admittance control, is to introduce a virtual separation between the robot end-effector and the human operator through a virtual safety surface. The simplest example of such surface is a virtual wall, a plane that reacts to penetration generating a viscoelastic virtual force.

Consider the case of a virtual wall, characterised by a unit vector $\mathbf{n}_w$, being the outer-pointing normal to the plane at point $\mathbf{p}_w$. Assuming that the tool centre point position in the absolute reference frame is given by vector $\mathbf{r}_{TCP}$, the penetration μ of the tool centre point into the virtual wall can be determined as follows:

$$\mu = \begin{cases} (\mathbf{p}_w - \mathbf{r}_{TCP}) \cdot \mathbf{n}_w & \text{if } (\mathbf{p}_w - \mathbf{r}_{TCP}) \cdot \mathbf{n}_w > 0. \\ 0 & \text{otherwise} \end{cases}$$

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press

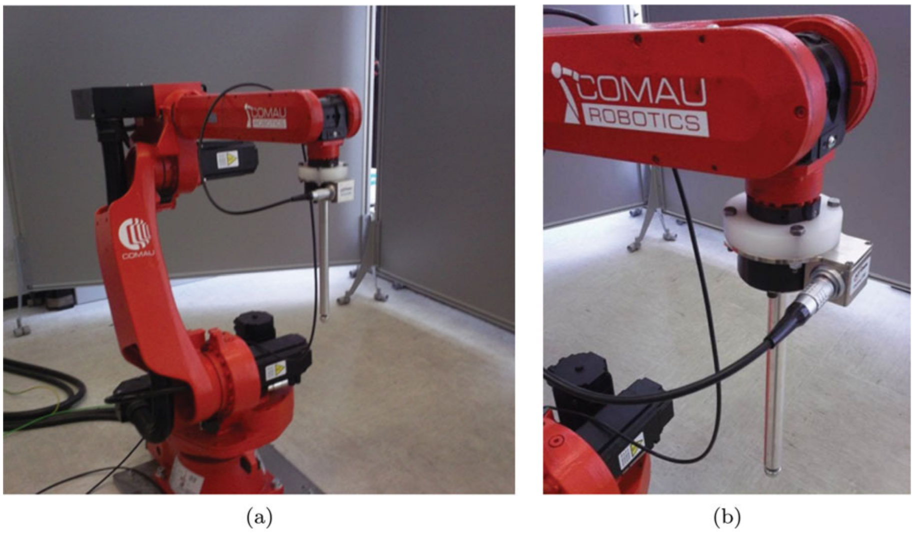


Fig. 6. (Colour online) The Smart Six robot and a detail of the end-effector with the ATI force/torque sensor.

Table I. Main specifications of the ATI force/torque sensor.

ATI SI-130-10	
Range F_x, F_y	±130 N
Range F_z	±400 N
Range T_x, T_y	±10 Nm
Range T_z	±10 Nm
Resolution F_x, F_y	1/20 N
Resolution F_z	1/10 N
Resolution T_x, T_y	1/400 Nm
Resolution T_z	1/400 Nm

The viscoelastic virtual force $\mathbf{f}_{vw}$ generated by the wall is then given by

$$\mathbf{f}_{vw} = (K_w\mu + D_w\dot{\mu})\mathbf{n}_w.$$

Adding this virtual force to the real forces measured by the sensor allows to prevent, as much as possible, the robot tool centre point from entering the space behind the virtual wall.

The virtual and real forces are then fed together into the admittance filter to generate the robot motion.

Then, with the same approach, more complex virtual surfaces can be introduced. Note that, however, the approach proposed here is limited to a point contact between the virtual wall and the tool centre point, and any extension to more complex constraints requires further investigation.

4. Experimental Results

The experimental set-up consists of a robot Smart Six (Fig. 6), a 6 d.o.f., 6 kg payload industrial manipulator manufactured by COMAU, equipped with the open version of the COMAU C4G controller.[4] In the open version, the C4G, acting as a network client, is linked to a real-time external PC through a real-time ethernet connection based on the RTnet protocol.[3] The real-time external PC, acting as a network server, is based on the RTAI Linux real-time extension.[2] The two devices communicate at a frequency of 500 Hz: every 2 ms, the industrial controller sends to the external PC the actual joint

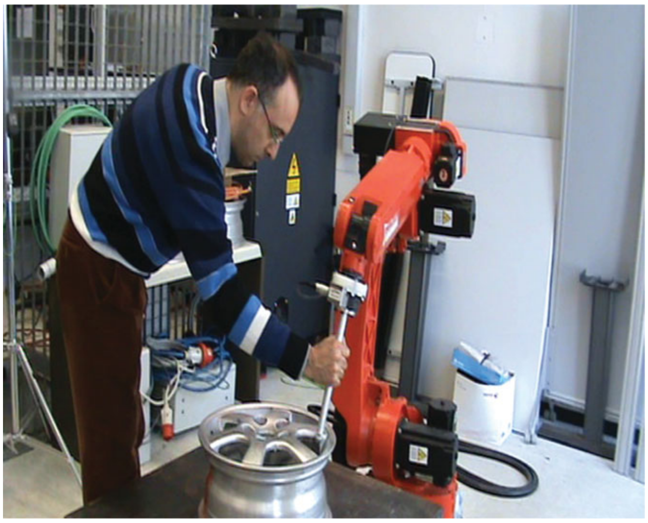
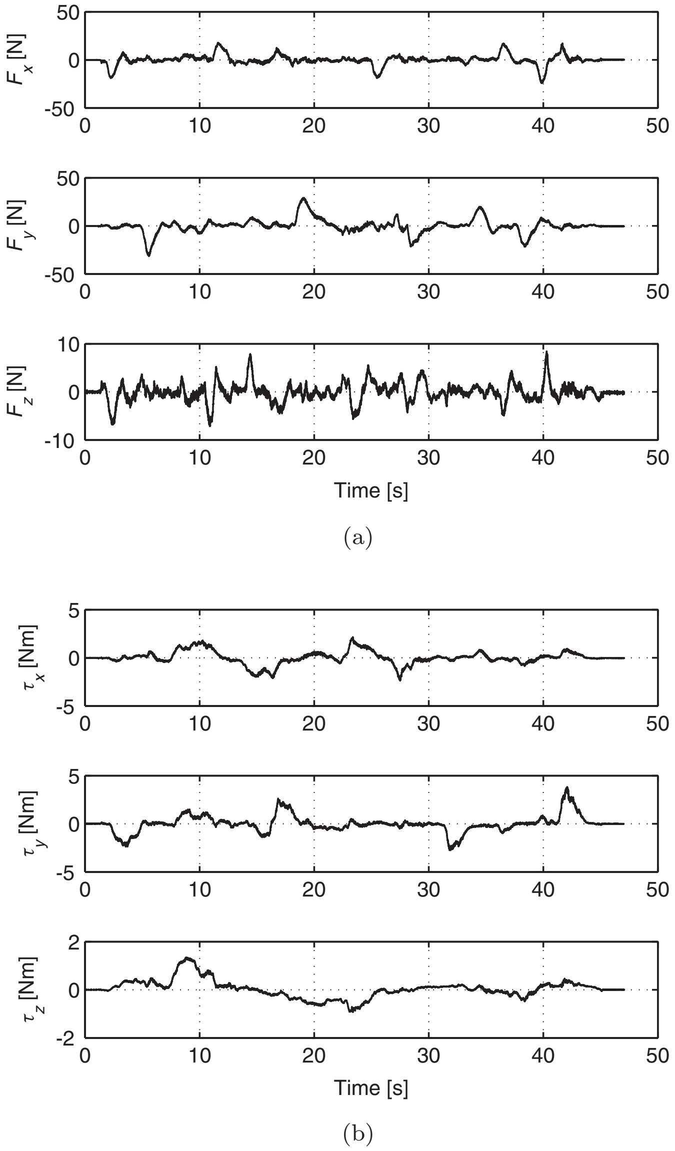


Fig. 7. (Colour online) The operator interacting with the robot during a walk-through programming operation.

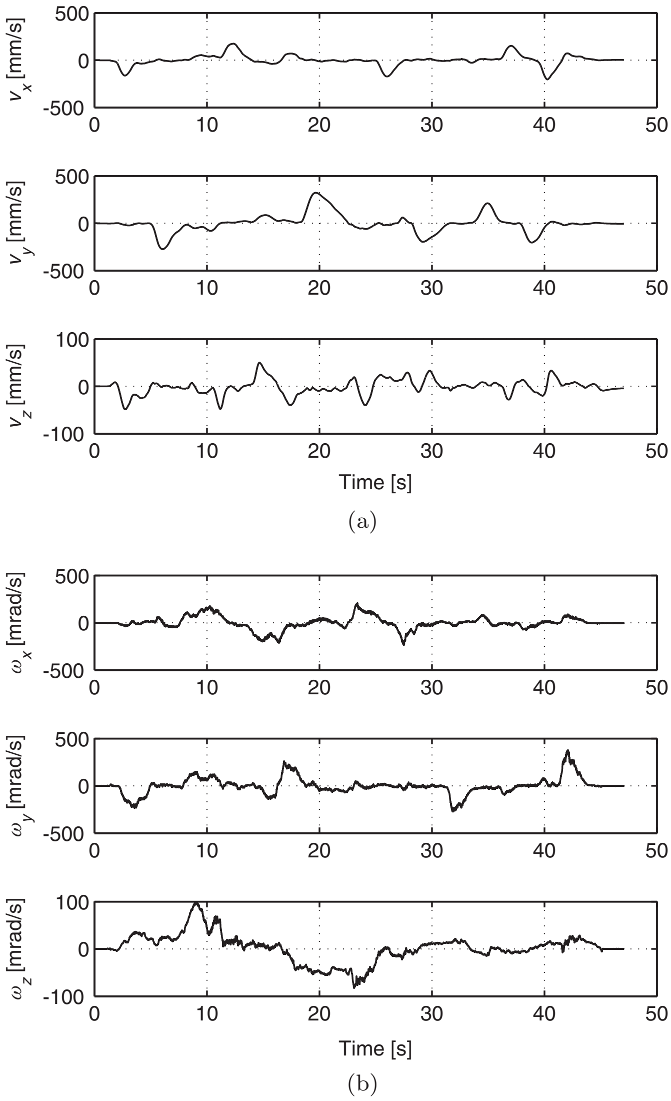
https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press



(a)

(b)

Fig. 8. Forces (a) and torques (b), expressed in the absolute frame, during a walk-through programming experiment.



(a)

(b)

Fig. 9. The virtual tool linear (a) and angular (b) velocities, expressed in the absolute frame, during a walk-through programming experiment.

positions and velocities, and the reference motor currents, and the external PC replies sending the reference joint positions and velocities.

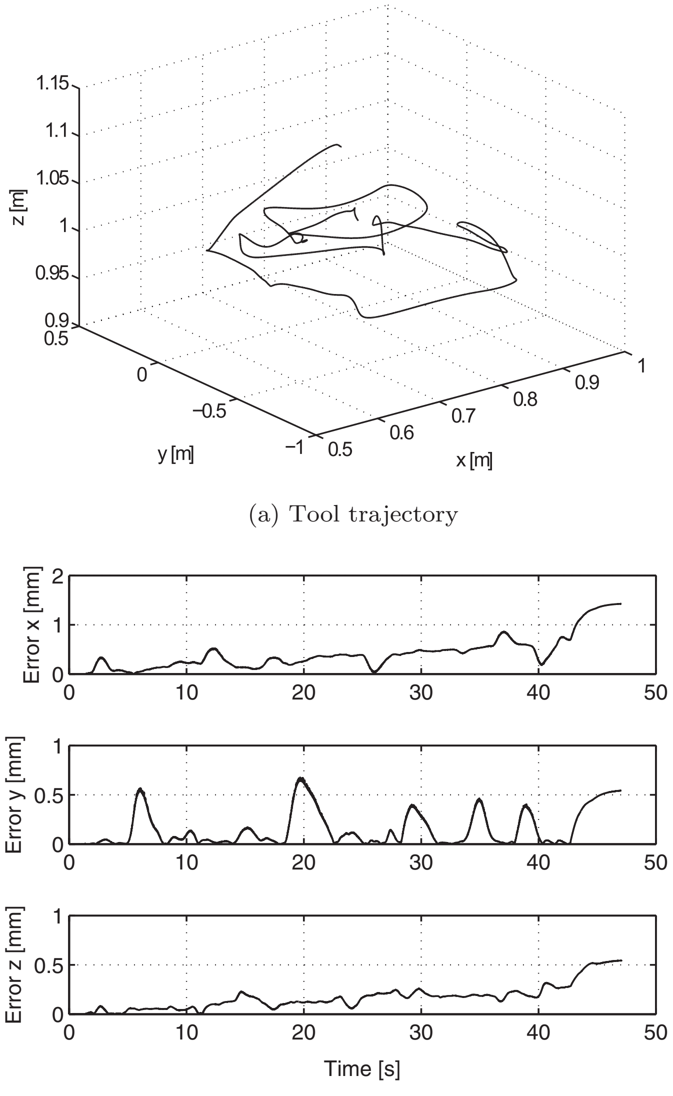
An ATI 6-axis wrist force/torque sensor (see Fig. 6 and Table 3.3 for further details) is fitted to the arm end-effector and linked to the PC through a DAQ board that is managed by the RTAI system, thanks to a real-time extension of the Comedi drivers.[1] The analogue voltages generated by the device are acquired by a dedicated real-time thread that runs periodically at a frequency of 10 kHz. This thread is also in charge of reducing the effect of measurement noise, filtering each channel with a 10th-order moving average filter.

The easiest way to implement the walk-through programming technique on an industrial robot controller is by enforcing the dynamics of the virtual tool through an admittance controller, as described in Section 3.1. An external loop is then closed outside the position control loops in the industrial controller (Fig. 2): the admittance controller, fed by the measurements of forces and moments acquired by a force/torque sensor mounted on the end-effector, yields modifications to the joint position set-points in order to guarantee the prescribed compliant behaviour.

Furthermore, in order to fulfil all the safety requirements discussed in Section 3.1, a set of safety constraints is introduced:

- a virtual force proportional to the tool centre point Cartesian velocity, by way of a nonlinear viscous friction coefficient that rapidly increases as the velocity approaches the value of 250 mm/s, is superimposed to the forces measured by the force/torque sensor (this force prevents the human operator from exceeding the maximum velocity prescribed by the safety requirements for the teaching phase);
- three virtual walls, surrounding the workstation at the front, left and right side, generate viscoelastic virtual forces proportional to the penetration, that are superimposed to the forces measured by the force/torque sensor (these forces prevent the robot from exiting the predefined

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press



(a) Tool trajectory

(b) Position errors

Fig. 10. A comparison between the virtual tool experimental positions and the virtual tool trajectory obtained integrating the nonlinear motion equations in (4) and (5).

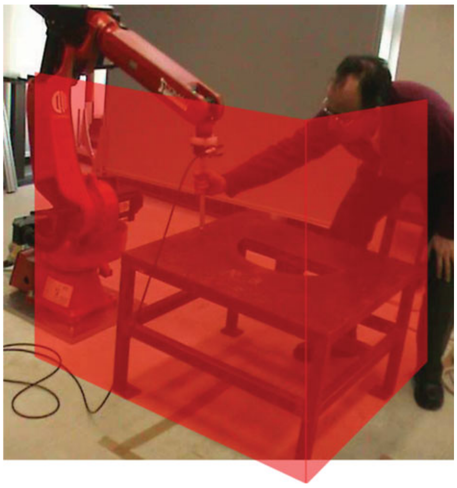


Fig. 11. (Colour online) The experimental set-up with two examples of the virtual walls.

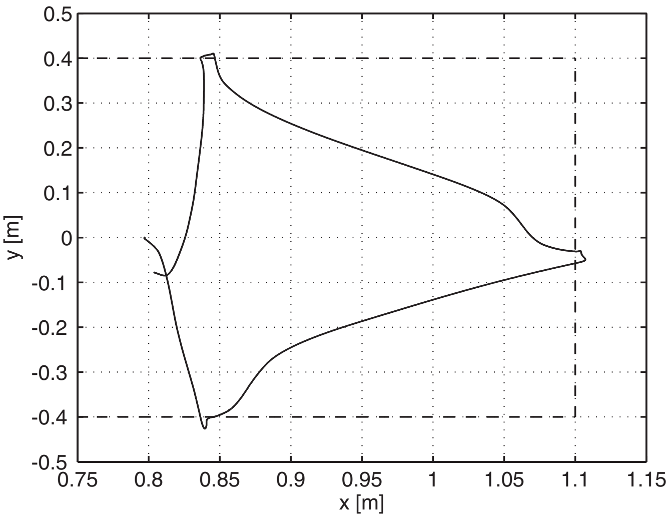


Fig. 12. The tool centre point trajectory in the xy plane (solid line) and the projection of the virtual walls on the xy plane (dash–dotted line).

working area, avoiding, as much as possible, a collision between the robot and the human body).

A preliminary experiment shows the behaviour of the system in the absence of any safety constraint (Fig. 7). It aims at comparing the trajectory executed by the robot, emulating a spherical mass as the virtual tool, and the one generated by a simulator that integrates the virtual tool motion equations (4)–(5).

In this experiment, the teacher walks the end-effector around a workstation, a table of 40 × 80 cm length and 40 cm height, placed in front of the robot at 50 cm from the robot base. The virtual tool was given the mechanical properties of a sphere of 50 kg of mass and 5 cm of radius, with a constant viscous friction coefficient of 50 Ns/m along each translational direction, and 10 Nms/rad along each rotational direction.

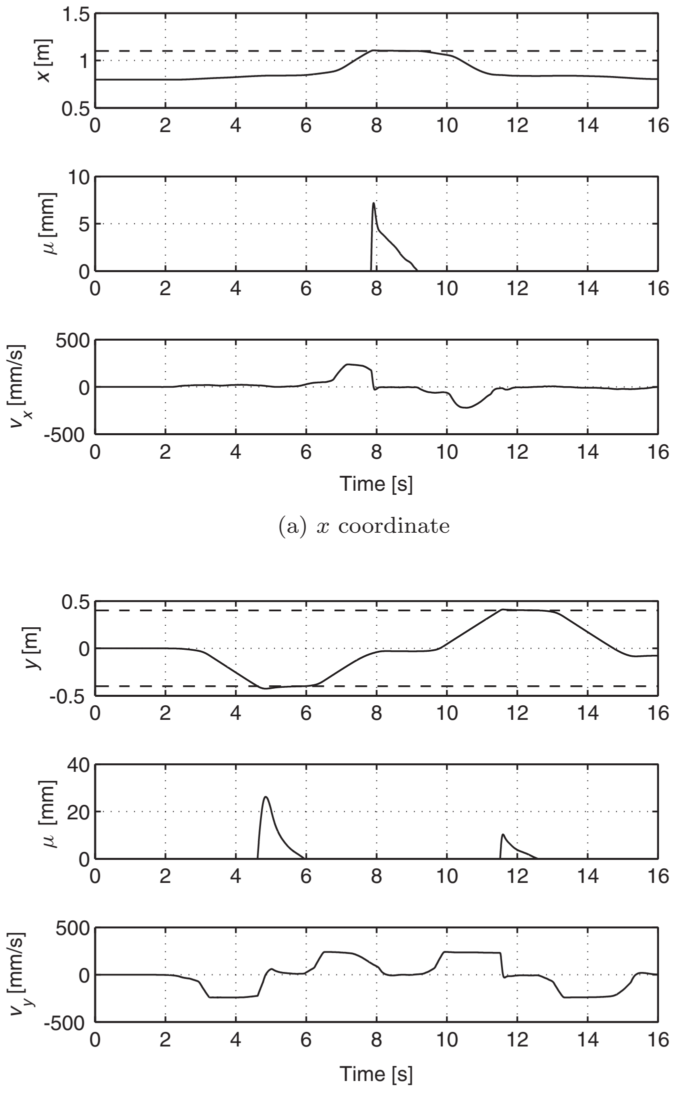
Figures 8 and 9 show the forces and moments (expressed in the absolute frame) and the corresponding virtual tool linear and angular velocities (again expressed in the absolute frame). These plots are useful to better understand the action applied by the operator on the robot end-effector and the corresponding motion of the virtual tool, which has then been compared with the actual motion of the end-effector.

Figure 10 shows such comparison between the trajectory walked by the robot and the trajectory obtained integrating the motion equations (4)–(5) fed by the forces and torques measured during the experiment.

Notice that the walk-through programming experiment depicted here is characterised by several movements, involving tool position and orientation (Fig. 10a). Though the accelerations exerted by the teacher are sometimes even larger than the ones applied by an actual human operator during a teaching operation, the robot always smoothly accommodates for the teacher's commands.

In any case, Fig. 10(b) clearly shows that the admittance filters, though simple and linear, are adequate to describe the motion of the virtual tool. In fact, considering that the aim of the walk-through programming technique is to give the

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press



(a) x coordinate

(b) y coordinate

Fig. 13. Time history of the x/y coordinates, limit imposed by the virtual wall on the x/y coordinates (dotted line), wall penetration μ and tool centre point velocity along the x/y directions.

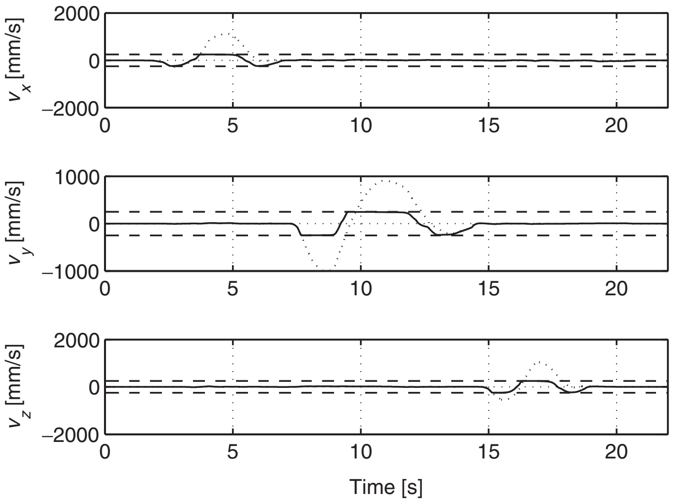


Fig. 14. An example of the effect of the variable friction coefficient (the dotted lines represent the velocity obtained simulating a set of impedance filters with constant friction) in keeping the tool centre point velocity below the safety limit of 250 mm/s (shown in the figure by the dashed lines).

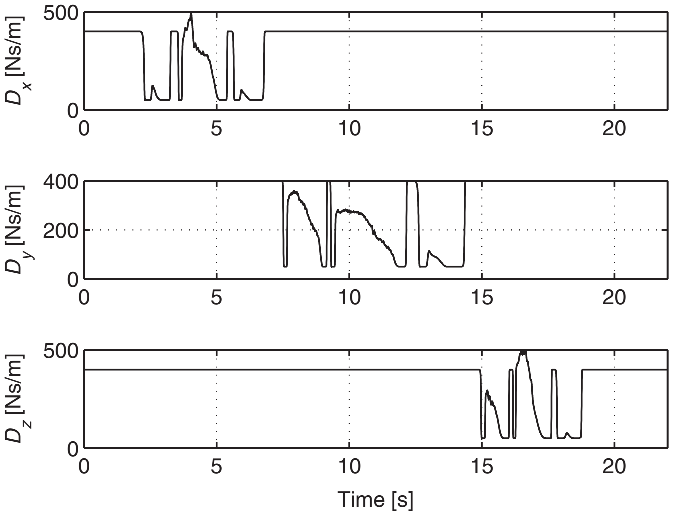


Fig. 15. Time history of the experimental friction coefficient applied to limit the tool centre point velocity to 250 mm/s.

teacher the impression of moving the real tool, instead of a robot, an error in the range of few millimetres is acceptable.

Another set of experiments aims at showing the effectiveness of the safety constraints introduced in Sections 3.2 and 3.3.

To prevent, as much as possible, dangerous situations, the motion of the robot has been confined to a box that includes the workstation (Fig. 11). Three virtual walls have been defined: one in front of the robot, 1.1 m far from the robot base, and two at the left and right sides, respectively, each at a distance of 0.4 m from the robot base. All these virtual walls react to a penetration of the robot end-effector generating a viscoelastic virtual force characterised by a stiffness, K, of 5000 N/m and a damping, D, of 1000 Ns/m. For safety reasons, these virtual forces are limited to a maximum value of 50 N.

Figure 12 shows the projection of the virtual walls on the ground plane (xy plane) together with the end-effector trajectory: the teacher drives the robot through a square that touches all the three walls from right to left. Figure 13 shows in detail the time history of the x and y coordinates and the time instants at which a penetration in the virtual wall occurs. Note that when this happens, the tool centre point velocity is almost instantaneously reduced so as to decrease as much as possible the energy associated with the robot in case of an impact.

Another way to enforce safety, as described in Section 3.2, relies on the introduction of a nonlinear viscous friction coefficient that forces the teacher to keep the tool centre point velocity below the safety limit of 250 mm/s. Following this requirement, relation (9) has been parameterised considering a maximum velocity, v_{safe}, of 250 mm/s, a maximum value of the friction coefficient, D_{safe}, of 800 Ns/m, and a range Δv_{safe} of 0.01 m/s.

An experiment has been performed to show the effectiveness of this approach. Three high-speed linear movements, along the x, y and z axis, have been imposed by the teacher. Figure 14 shows a comparison between the tool centre point experimental velocities (solid lines) and the velocities computed in simulation with a set of impedance filters characterised by a constant friction coefficient (dotted lines). The dashed lines point out the limits at ±250 mm/s.

The time history of the velocity demonstrates that the introduction of a nonlinear friction coefficient, which increases as the tool centre point velocity approaches the safety limit (Fig. 15), can effectively impose a smooth constraint on these velocities.

5. Conclusions

An implementation of the walk-through programming mode using the admittance control approach has been described in this paper.

All the issues involved, from the implementation of the admittance control on an industrial robot controller to the safety aspects due to the presence of the operator in the robot's working envelope, have been carefully considered. In particular, two safety strategies, which exploit the admittance controller in order to keep the robot Cartesian velocity below a safety threshold and to virtually separate the human body from the robot workspace, have been investigated.

A set of experiments completes the paper, demonstrating the behaviour of the walk-through programming mode and the effects of the safety functionalities.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013—Challenge 2: Cognitive Systems, Interaction, Robotics—under grant agreement No. 231143, ECHORD (Experiment FIDELIO).

References

1. COMEDI, "Linux Control and Measurement Device Interface," available at: http://www.comedi.org.
2. "RTAI – Real Time Application Interface," available at: http://www.rtai.org.
3. "RTnet – Hard Real-Time Networking for Real-Time Linux," available at: http://www.rtnet.org.
4. *Comau Robotics instruction handbook,* C4G OPEN System Software Rel. 3.33, 2011.
5. ISO 10218-1:2011 *Robots for industrial environments: Safety requirements*–Part 1: Robot (European Committee for Standardization, 2011).
6. O. M. Al-Jarrah and Y. F. Zheng, "Arm-Manipulator Coordination for Load Sharing using Compliant Control," *Proceedings of the IEEE International Conference on Robotics and Automation*, Minneapolis, MN (Apr. 1996) pp. 1000–1005.
7. O. M. Al-Jarrah and Y. F. Zheng, "Arm-Manipulator Coordination for Load Sharing Using Reflexive Motion Control," *Proceedings of the IEEE International Conference on Robotics and Automation*, Albuquerque, NM (Apr. 1997) pp. 2326–2331.
8. O. M. Al-Jarrah and Y. F. Zheng, "Arm-Manipulator Coordination for Load Sharing Using Variable Compliance Control," *Proceedings of the IEEE International Conference on Robotics and Automation*, Albuquerque, NM (Apr. 1997) pp. 895–900.
9. A. Albu-Schaeffer and G. Hirzinger, "Cartesian Impedance Control Techniques for Torque Controlled Light-Weight Robots," *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, DC, USA (2002) pp. 657–663.
10. H. Ang, W. Lin and S.-Y. Lim, "A walk-through programmed robot for welding in shipyards," *Ind. Robots* **26**(5), 377–388 (1999).
11. M. H. Ang Jr and L. S. Yong, "An Industrial Application of Control of Dynamic Behavior of Robots – A Walk-Through Programmed Welding Robot," *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA, USA (Apr. 2000) pp. 2352–2357.
12. G. Ferretti, G. Magnani and P. Rocco, "Assigning Virtual Tool Dynamics to an Industrial Robot Through an Admittance Controller," *Proceedings of the IEEE International Conference on Advanced Robotics*, Munich, Bavaria (Jun. 2009) pp. 1–6.
13. M. Frigola, J. Poyatos, A. Casals and J. Amat, "Improving Robot Programming Flexibility Through Physical Human-Robot Interaction," *IROS Workshop on Robot Programming by Demonstration*, Las Vegas, USA (Oct. 2003) pp. 1–8.
14. G. Grunwald, G. Schreiber, A. Albu-Schaffer and G. Hirzinger, "Programming by touch: The different way of human-robot interaction," *IEEE Trans. Ind. Electron.* **50**(4), 659–666 (2003).
15. A. K. Gupta and S. K. Arora, *Industrial Automation and Robotics* (Laxmi Publications, Daryaganj, New Delhi, 2007).
16. R. Ikeura and H. Inooka, "Variable Impedance Control of a Robot for Cooperation with a Human," *Proceedings of the IEEE International Conference on Robotics and Automation,* Nagoya, Japan (1995) pp. 3097–3102.
17. M. Jakopec, S. J. Harris, F. Rodriguez y Baena, P. Gomes, J. Cobb and B. L. Davies, "The first clinical application of a "hands-on" robotic knee surgery system," *Comput. Aided Surg.* **6**(6), 329–339 (2001).
18. R. Kumar, P. Berkelman, P. Gupta, A. Barnes, P. S. Jensen, L. L. Whitcomb and R. H. Taylor, "Preliminary Experiments in Cooperative Human/Robot Force Control for Robot Assisted Microsurgical Manipulation," *Proceedings of the IEEE International Conference on Robotics and Automation,* San Francisco, CA (Apr. 2000) pp. 610–617.
19. Occupational Safety and Health Administration (OSHA), *OSHA Technical Manual, Industrial robots and robot system safety (Section IV, Chapter 4).* U.S. Department of Labor Occupational Safety & Health Administration, NW, Washington, DC.
20. T. Ortmaier, H. Weiss, U. Hagn, M. Grebenstein, M. Nickl, A. Albu-Schäffer, C. Ott, S. Jörg, R. Konietschke, L. Le-Tien and G. Hirzinger, "A Hands-on-Robot for Accurate Placement of Pedicle Screws," *Proceedings of the IEEE International Conference on Robotics and Automation*, Orlando, FL (May 2006) pp. 4179–4186.
21. C. Powell, "Case study: Kuntz Electroplating automated wheel polishing system," *Robotics* (available at: http://www.robotics.org) (2002).
22. ABB Robotics, *FC Programming Handle* (2010).
23. A. Tellaeche, R. Arana, M. A. Pérez and I. Maurtua, "Accurate Manual Guided Robot Programming and Trajectory Correction using 3D Vision by Laser Triangulation," *Workshop Industry–Academia Collaboration in the ECHORD Project: A Bridge for European Robotics Innovation, IEEE International Conference on Robotics and Automation*, St. Paul, Minnesota, USA (2012) pp. 385–394.
24. T. Tsumugiwa, R. Yokogawa and K. Yoshida, "Stability Analysis for Impedance Control of Robot for Human-Robot Cooperative Task System," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan (Oct. 2004) pp. 3883–3888.

https://doi.org/10.1017/S0263574713000404 Published online by Cambridge University Press