Industrial Robot: An International Journal

Real-time velocity scaling and obstacle avoidance for industrial robots using fuzzy dynamic movement primitives and virtual impedances

Iman Kardan, Alireza Akbarzadeh, Ali Mousavi Mohammadi,

Real-time velocity scaling and obstacle avoidance for industrial robots using fuzzy dynamic movement primitives and virtual impedances

Iman Kardan, Alireza Akbarzadeh and Ali Mousavi Mohammadi
Center of Excellence on Soft Computing and Intelligent Information Processing, Mechanical Engineering Department,
Ferdowsi University of Mashhad, Mashhad, Iran

Abstract

Purpose – This paper aims to increase the safety of the robots' operation by developing a novel method for real-time implementation of velocity scaling and obstacle avoidance as the two widely accepted safety increasing concepts.

Design/methodology/approach – A fuzzy version of dynamic movement primitive (DMP) framework is proposed as a real-time trajectory generator with imbedded velocity scaling capability. Time constant of the DMP system is determined by a fuzzy system which makes decisions based on the distance from obstacle to the robot's workspace and its velocity projection toward the workspace. Moreover, a combination of the DMP framework with a human-like steering mechanism and a novel configuration of virtual impedances is proposed for real-time obstacle avoidance.

Findings – The results confirm the effectiveness of the proposed method in real-time implementation of the velocity scaling and obstacle avoidance concepts in different cases of single and multiple stationary obstacles as well as moving obstacles.

Practical implications – As the provided experiments indicate, the proposed method can effectively increase the real-time safety of the robots' operations. This is achieved by developing a simple method with low computational loads.

Originality/value – This paper proposes a novel method for real-time implementation of velocity scaling and obstacle avoidance concepts. This method eliminates the need for modification of original DMP formulation. The velocity scaling concept is implemented by using a fuzzy system to adjust the DMP's time constant. Furthermore, the novel impedance configuration makes it possible to obtain a non-oscillatory convergence to the desired path, in all degrees of freedom.

Keywords Safety, Fuzzy inference system, Collision avoidance, Dynamic movement primitives, Velocity scaling

Paper type Research paper

1. Introduction

Different design-level considerations are proposed for safe human–robot interaction, e.g. limiting the range of the robot's motion, imbedding emergency and safe stopping sensors and switches and defining reduced speed working regions (Gaskill, 1994). Moreover, some operation-level considerations may be incorporated to increase the safety. Camera visions, laser trackers, proximity sensors and even physiological monitoring systems (Kulić and Croft, 2007) are frequently used to provide a perception of operator and environment for the robot. Different actions may be taken when nearby obstacles are detected. Reducing the robot speed, also termed as velocity scaling, is a well-established safety increasing strategy. This strategy is required by some safety standards, such as ISO 10218 and ISO 15066, and widely used in industries and literatures (Kulić and Croft, 2007; Kulić, 2006; Zanchettin *et al.*, 2016). The safety of the robot's operations is usually enhanced further by implementing some obstacle avoidance strategies such as repelling potential fields (Khatib, 1986; Kim and Khosla, 1992; Park *et al.*, 2008; Zavlangas and Tzafestas, 2000), virtual impedances (Cai *et al.*, 2014; Lacevic *et al.*, 2013; Lo *et al.*, 2016; Arai *et al.*, 1989; Khansari-Zadeh and Khatib, 2017), elastic strips (Brock and Khatib, 2002; Sun *et al.*, 2016) and human-like steering (Fajen and Warren, 2003; Huang *et al.*, 2006; Fajen *et al.*, 2003; Ijspeert *et al.*, 2013; Hoffmann *et al.*, 2009; Pastor *et al.*, 2013) techniques.

Practical implementation of velocity scaling and obstacle avoidance techniques requires real-time path generation methods. These methods should have low computation loads and should be capable of synchronized modification of all the robot's degrees-of-freedom (DOF). Moreover, when velocity scaling or obstacle avoidance modifications are imposed, the real-time path generation method should preserve the smoothness and continuity of the end-effector (EE) path.

Dynamic movement primitive (DMP) framework is a biologically inspired trajectory generation method and has recently gained great attentions from the researchers (Ijspeert *et al.*, 2013; Hoffmann *et al.*, 2009; Ijspeert *et al.*, 2002; Ijspeert *et al.*, 2003; Basa and Schneider, 2015; Pastor *et al.*, 2013; Park *et al.*, 2008;

The current issue and full text archive of this journal is available on Emerald Insight at: **www.emeraldinsight.com/0143-991X.htm**

Industrial Robot: An International Journal
45/1 (2018) 110–126
© Emerald Publishing Limited [ISSN 0143-991X]
[DOI 10.1108/IR-02-2017-0035]

Received 11 February 2017
Revised 1 July 2017
Accepted 3 July 2017

Ude *et al.*, 2010). The DMP framework is used in a wide array of applications including learning by demonstration (Tamosiunaite *et al.*, 2011; Pignat and Calinon, 2017; Deniša *et al.*, 2016; Kramberger *et al.*, 2016; Abu-Dakka *et al.*, 2014), mobile robots (Jiang *et al.*, 2016), robot locomotion (Rosado *et al.*, 2016), control applications (Krug and Dimitrov, 2015), human–robot and robot–robot cooperative tasks (Maeda *et al.*, 2016; Kulvicius *et al.*, 2013), exoskeletons (Kamali *et al.*, 2016; Huang *et al.*, 2016), humanoid robots (Mukovskiy *et al.*, 2017; Li *et al.*, 2014) and obstacle avoidance (Park *et al.*, 2008; Ijspeert *et al.*, 2013; Hoffmann *et al.*, 2009; Tan *et al.*, 2011; Hoffmann and Mitchell, 2016). The DMP method has a simple structure and low computation loads. Moreover, as the path is generated based on motion equations of a second-order dynamic system, the smoothness and continuity of the trajectory and its derivatives are guaranteed (Nemec and Ude, 2012). Furthermore, the DMP method makes it possible to easily and smoothly scale the task velocity by changing only one parameter, which is the DMP's time constant (Ijspeert *et al.*, 2002; Nemec and Ude, 2012).

In this paper, a fuzzy inference system (FIS) is used to adjust the DMP's time constant. The FIS increases the DMP's time constant or equivalently reduces the robot's speed, according to the detected risk level. The subject's distance from the workspace and its velocity projection toward the workspace are considered as inputs to the FIS.

Adding some terms to the system dynamics, the DMP framework can also be used for obstacle avoidance purposes. The potential field (Park *et al.*, 2008; Tan *et al.*, 2011; Hoffmann and Mitchell, 2016) and human-like steering (Hoffmann *et al.*, 2009; Ijspeert *et al.*, 2013) terms are successfully added to the DMP equations to obtain automatic obstacle avoidance methods. The human-like steering technique is inspired from human behavior in avoiding the obstacles and therefore provides more natural performances (Fajen and Warren, 2003; Huang *et al.*, 2006; Fajen *et al.*, 2003). Moreover, this method removes some problems with the potential field technique, e.g. leading to local minima in the case of multiple nearby obstacles. However, these methods need to modify the original DMP equation which may cause some problems in the convergence of the deviated path to the desired path when obstacles are passed. Therefore, some stability and convergence considerations are required.

In this paper, we propose a combination of the DMP framework with the virtual impedance method to avoid obstacles in real-time. The virtual impedance method for real-time obstacle avoidance was first proposed by Arai *et al.* (Arai *et al.*, 1989) and has found extensive applications (Cai *et al.*, 2014; Lacevic *et al.*, 2013; Lo *et al.*, 2016; Zavlangas and Tzafestas, 2000; Khansari-Zadeh and Khatib, 2017). Combination of the virtual impedance method with the DMP framework eliminates the need for manipulating the original DMP formulation. The DMP formulation outputs the desired path with scaled velocity, while distraction from the desired path is caused by the virtual repelling forces exerted to the virtual mass. The virtual impedance is consisted of some simple mass-spring-damper components. Therefore, it provides a better physical interpretation and the stability of the path generation system is guaranteed as well as its convergence to the desired path when obstacles are passed.

In the existing applications (Cai *et al.*, 2014; Lacevic *et al.*, 2013; Lo *et al.*, 2016; Arai *et al.*, 1989; Zavlangas and Tzafestas, 2000; Khansari-Zadeh and Khatib, 2017), the virtual impedance is assumed as a single point mass, simply pinned to the desired path through a spring and a damper. Adjusting the impedance parameters, it is possible to obtain a critically damped longitudinal convergence to the desired path when repelling forces are diminished. However, some lateral oscillations will be always present in the motion of the virtual mass. In this paper, we propose a novel configuration of the virtual impedance which makes it possible to obtain critically damped convergence in all directions.

The main contributions of this paper are as listed as follows:

- proposing a fuzzy version of the DMP framework for implementation of the velocity scaling concept;
- proposing a combination of the DMP formulation with human-like steering mechanism and virtual impedance method to avoid collision with obstacles; and
- proposing a novel configuration of the virtual impedance to obtain critically damped convergence in all directions.

The effectiveness of the proposed method is evaluated through some simulations and experiments, carried out on a custom made SCARA robot, called FUM SCARA (Shariatee *et al.*, 2014; Mousavi *et al.*, 2015b; Mousavi *et al.*, 2015a). Figure 1 displays the FUM SCARA robot, designed and manufactured in FUM Robotics Lab at Ferdowsi University of Mashhad.

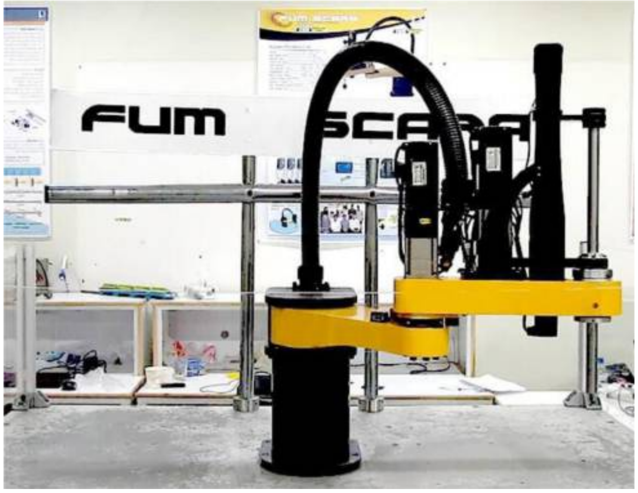
The rest of the paper is organized as follows. Section 2 details the proposed method in two subsections. Subsection 2.1 describes the DMP framework with fuzzy time constant. The new configuration of the virtual impedance is given in Subsection 2.2. The inverse kinematic model (IKM) of the FUM SCARA robot is given in Section 3. Sections 4 and 5 provide the results of evaluating the performance of the proposed method in simulation and practice. Finally, Section 6 concludes the paper.

2. Proposed method

2.1 Dynamic movement primitives with fuzzy time constant

Unlike the traditional online trajectory generation methods, using the DMPs makes it possible to smoothly and synchronically reduce the robot speed in all directions by

Figure 1 FUM SCARA robot



modulating only one parameter. The general formulation of DMP is given as in (Ijspeert *et al.*, 2013):

$$\tau\dot{\boldsymbol{\rho}} = \alpha_\rho\left(\beta_\rho(\boldsymbol{P}_{goal}-\boldsymbol{P})-\boldsymbol{\rho}\right)+\boldsymbol{f}(s) \tag{1}$$

$$\tau\dot{\boldsymbol{P}} = \boldsymbol{\rho} \tag{2}$$

in which $\boldsymbol{\rho}$ is the vector of state variables and $\boldsymbol{P}$ is the vector of trajectory variables which may include joint space or workspace variables. To facilitate the obstacle avoidance procedure, in this paper, $\boldsymbol{P}$ is considered to include EE's workspace variables as $\boldsymbol{P} = \{x, y, z\}^T$. α_ρ and β_ρ are positive constants that affect learning and generalization abilities of the DMP model, usually chosen as $\beta_\rho = \alpha_\rho/4$. $\boldsymbol{P}_{goal}$ is the vector of the goals or the final values that the trajectory variables should attain at the end of the motion. τ is a time constant resembling parameter that governs the overall time for going through the trajectory and is somehow reciprocal to the robot's motion speed. Larger τ indicates slower motions and vice versa. $\boldsymbol{f}(s)$ is the vector of forcing terms calculated as:

$$\boldsymbol{f}(s) = \frac{s}{\sum_{i=1}^{N}\psi_i(s)}\left(\boldsymbol{P}_{goal}-\boldsymbol{P}_{init}\right)\circ\sum_{i=1}^{N}\psi(s)\boldsymbol{w}_i \tag{3}$$

Here, ° denotes entry-wise (element wise) product, P_{init} includes the initial values of the desired trajectory variables and $w_i = [w_{ix}, w_{iy}, w_{iz}]^T$ is the i^{th} column of $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_i, \ldots, \boldsymbol{w}_N]$ which is the weighting matrix to be adjusted to learn the desired trajectory. The N basis functions, $\psi_i(s)$, are obtained as:

$$\psi_i(s) = \exp\left(-\frac{1}{2\sigma_i^2}(s-c_i)^2\right) \tag{4}$$

where, σ_i is the width of the i^{th} basis function and c_i is its center location. The center locations should cover the entire interval of the variable $s \in [0, 1]$ which is calculated as:

$$\tau\dot{s} = -\alpha_s s \tag{5}$$

where, α_s is a constant. The variable s can be considered as the phase variable of the system with $s = 1$ and $s = 0$ indicating the initial and the final state of the motion, respectively. Note that, s is a scalar variable which is common between all the trajectory variables. Therefore, all the motion DOF will have the same phase all over the trajectory and the synchronization is automatically assured. The supervised learning approach (Schaal *et al.*, 2007) is applied to adjust the weights. For a given ideal trajectory, the target values of the forcing vector are obtained as:

$$\boldsymbol{f}_{\boldsymbol{q}}^{*} = \tau^2\ddot{\boldsymbol{P}}_q + \alpha_\rho\dot{\boldsymbol{P}}_q - \alpha_\rho\beta_\rho\left(\boldsymbol{P}_{goal}-\boldsymbol{P}_q\right), \qquad q = 1, \ldots, m \tag{6}$$

where, subscript q indicates the variables calculated for the q^{th} sample of the ideal trajectory and m is the total number of the points in the learning set, sampled from the ideal trajectory. Then, the weighting matrix is obtained as:

$$\boldsymbol{w}_i(k) = \boldsymbol{\Lambda}_k^T\boldsymbol{\Psi}_i\boldsymbol{\Gamma}_k/\boldsymbol{\Lambda}_k^T\boldsymbol{\Psi}_i\boldsymbol{\Lambda}_k, \quad k = 1, 2, 3 \tag{7}$$

where:

$$\boldsymbol{\Psi}_i = \begin{bmatrix} \psi_i(s_1) & 0 & \ldots & 0 \\ 0 & \psi_i(s_2) & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \ldots & 0 & \psi_i(s_m) \end{bmatrix} \tag{8}$$

$$\boldsymbol{\Gamma}_k = \begin{pmatrix} \boldsymbol{f}_1^{\star}(k) \\ \vdots \\ \boldsymbol{f}_m^{\star}(k) \end{pmatrix} \tag{9}$$

$$\boldsymbol{\Lambda}_k = \begin{pmatrix} \xi_1(k) \\ \vdots \\ \xi_m(k) \end{pmatrix} \tag{10}$$

in which, s_q is the phase variable for the q^{th} sample and ξ_q is the scaled phase variable for the q^{th} sample calculated as:

$$\xi_q = s_q\left(\boldsymbol{P}_{goal}-\boldsymbol{P}_{init}\right), \qquad q = 1, \ldots, m \tag{11}$$

Distance from obstacle to the workspace, the obstacle velocity and its motion direction are important safety factors. The distance from obstacle to the robot's base is considered as the distant measure, r_n, non-dimensionalized by the radius of the workspace as:

$$r_n = \frac{\left|\boldsymbol{P}_o^j\right|}{r_w} \tag{12}$$

where, r_w is the radius of the workspace sphere of the robot and $\boldsymbol{P}_o^j$ is the position vector of the j^{th} obstacle, described in a coordinate system whose origin is located at the robot's base. The two latter safety factors can be combined by considering the projection of the obstacle velocity on the line connecting the obstacle to the robot's base. The velocity projection, non-dimensionalized by the maximum speed of the EE, is considered as the velocity measure:

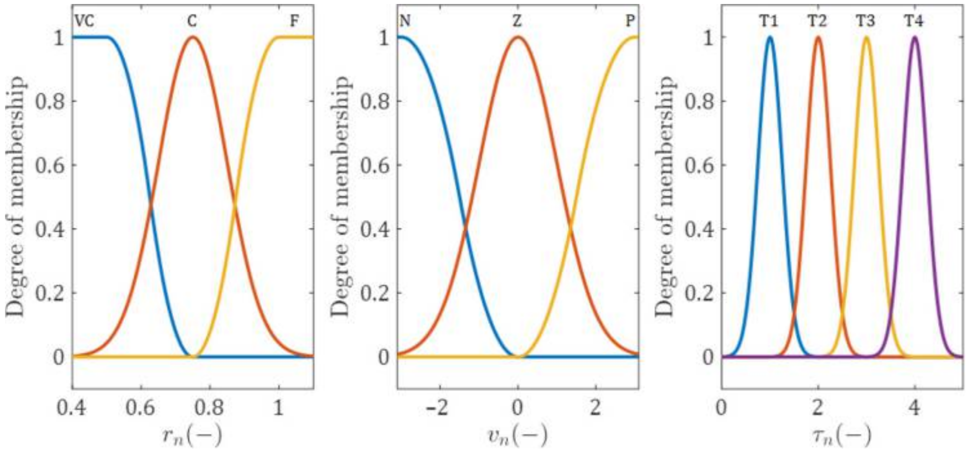
$$v_n = \frac{\dot{\boldsymbol{P}}_o^j \cdot \boldsymbol{P}_o^j}{\left|(v_e)_{max}\right|} \tag{13}$$

where, $|(v_e)_{max}|$ is the absolute maximum velocity of the EE, $\dot{P}_o^j$ is the velocity vector of the j^{th} obstacle and the inner product $\dot{\boldsymbol{P}}_o^j \cdot \boldsymbol{P}_o^j$ gives the projection of the obstacle velocity. Note that a negative value of v_n indicates an approaching obstacle. r_n and v_n are chosen as decisive factors in safe robot operation, and the robot speed should be set as a function of these two factors. The function is implemented as a fuzzy inference engine which takes the two factors as its inputs and decides the non-dimensionalized time constant for the DMPs, τ_n, as its output. The output variable is defined as:

$$\tau_n = \frac{\tau}{\tau_I} \tag{14}$$

where, τ_I is the ideal time constant which is the duration of the ideal trajectory, P_I, in the absence of the nearby obstacles. Figure 2 depicts the Gaussian membership functions for input and output parameters. The rules table of the fuzzy inference

Figure 2 Membership functions for the input and output parameters



engine is given in Figure 3, designed to be intuitive and to provide a continuous and smooth rule surface.

Each cell of the rule table represents a fuzzy rule of the form:

$$Rule\ \sharp l:\ If\ r_n\ is\ A_l\ and\ v_n is\ B_l\ then\ \tau_n\ is\ C_l \tag{15}$$

in which $l \in \{1\ldots9\}$ is the number of the considered rule. Moreover, $A_l \in \{F, C, VC\}$ is the membership function of r_n, $B_l \in \{P, Z, N\}$ is the membership function of v_n and $C_l \in \{T_1, T_2, T_3, T_4\}$ is the membership function of τ_n, all corresponding to the l^{th} rule.

Using Mamdani inference engine, singleton fuzzifier and center average defuzzifier, the FIS calculates the non-dimensionalized output as:

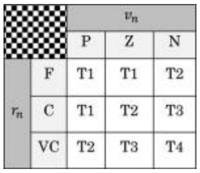
$$\tau_n = \frac{1}{\sum_{l=1}^{9} \mu_{A_l}(r_n)\mu_{B_l}(v_n)} \sum_{l=1}^{9} b_l \mu_{A_l}(r_n)\mu_{B_l}(v_n) \tag{16}$$

where, b_l is the center value of the output membership function C_l. Moreover, $\mu_{A_l}(r_n)$ and $\mu_{B_l}(v_n)$ are the membership values of r_n and v_n in A_l and B_l, respectively.

In the case of multiple obstacles, the time constant is calculated for each obstacle separately, and the largest time constant is applied.

If there is no obstacle nearby, the FIS sets τ as $\tau = \tau_I$. Therefore, the DMP outputs the ideal trajectory, i.e. $\boldsymbol{P} = \boldsymbol{P}_I$ where $\boldsymbol{P}_I$ is the ideal EE trajectory. When there are some obstacles nearby, the FIS sets τ as $\tau > \tau_I$. Consequently, the DMP outputs the EE trajectory with scaled velocity, i.e. $\boldsymbol{P} = \boldsymbol{P}_{sv}$, where $\boldsymbol{P}_{sv}$ is the same as the ideal EE trajectory but with reduced speeds. Note that, in the cases of $\boldsymbol{P}_I$ and $\boldsymbol{P}_{sv}$, the physical paths followed by the EE are identical. In other words, in the case of $\boldsymbol{P}_{sv}$, the EE still moves on the ideal path but with scaled velocity.

Figure 3 Fuzzy rules for time constant adjustment



		v_n		
		P	Z	N
r_n	F	T1	T1	T2
	C	T1	T2	T3
	VC	T2	T3	T4

2.2 Virtual impedance and human-like steering model for obstacle avoidance

To avoid the manipulation of the DMP formula, the concept of virtual impedance is used in this paper. In this approach, the EE is set to follow the trajectory of a point mass with arbitrary inertia, M_v, pinned to the scaled velocity trajectory of the EE (output of the DMP framework, P_{sv}) by means of a spring and a damper of arbitrary stiffness, K_v, and damping, D_v. Nearby obstacles apply repelling forces on the virtual point mass to keep it away from collision. Therefore, when obstacles are present around the ideal path, the point mass is deviated from the ideal path. As the EE is set to follow the trajectory of the point mass, the collision between EE and obstacle is prevented. Considering Figure 4, the equation of motion of the point mass is obtained as:

$$M_v\ddot{\boldsymbol{P}}_d + D_v\big(\dot{\boldsymbol{P}}_d - \dot{\boldsymbol{P}}_{sv}\big) + K_v(\boldsymbol{P}_d - \boldsymbol{P}_{sv}) = \sum_{j=1}^{p} \boldsymbol{F}_o^j \tag{17}$$

where, $\boldsymbol{P}_d$ is the position vector of the point mass which is considered as the desired position to be followed by the EE. $\boldsymbol{P}_{sv}$ locates the scaled velocity position of the EE on the ideal path, $\boldsymbol{F}_o^j$ is the force vector applied to the point mass from j^{th} obstacle and p is the total number of obstacles. The parameters are also illustrated in Figure 4. The position vectors are defined in the inertial frame W located at the robot's base. In Figure 4, it is assumed that the EE perfectly follows the position of the point mass. Therefore, the actual trajectory of the EE, $\boldsymbol{P}_a$, is depicted to exactly match the trajectory of the point mass, $\boldsymbol{P}_d$. However, some tracking errors are always present in the practice, and the matching between $\boldsymbol{P}_d$ and $\boldsymbol{P}_a$ depends on the position control accuracy of the robot.

The inertia, stiffness and damping coefficients of the virtual impedance are usually chosen such that a critically damped transient behavior is obtained with no oscillation and a reasonable speed. Although the coefficients can be chosen to remove the longitudinal oscillations (toward the desired trajectory), some lateral oscillations will be present in the EE motion if the point mass is simply pinned to the desired trajectory (Figure 5).

To avoid the lateral oscillations, a new arrangement of the virtual impedance is considered in this paper. Figure 6 illustrates a 2D version of the new arrangement of the virtual impedance. In this arrangement, the point mass is assumed to be connected to

Figure 4 Conventional virtual impedance method

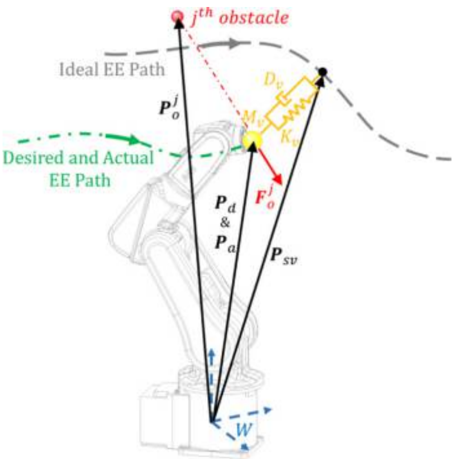


Figure 5 Lateral vibrations in conventional virtual impedance method

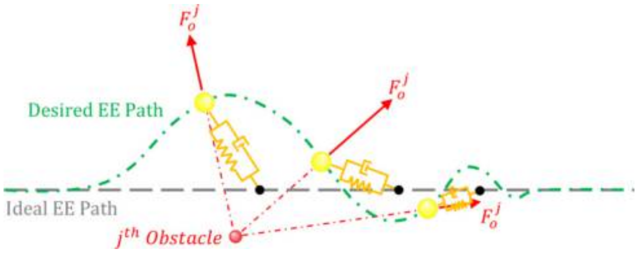
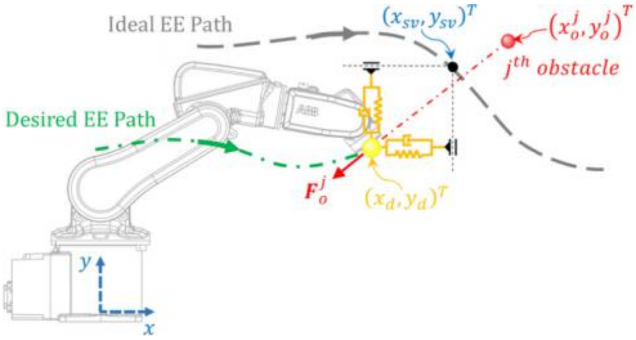


Figure 6 Two-dimensional representation of the proposed configuration for the virtual impedance



the desired trajectory, through two sets of springs and dampers. The springs and dampers are confined in two perpendicular prismatic guides. The prismatic guides are always aligned with x and y axes of the inertial frame and do not take rotations.

Therefore, the equation of motion of the point mass is obtained as:

$$\begin{cases} M_v\ddot{x}_d + D_{vx}(\dot{x}_d - \dot{x}_{sv}) + K_{vx}(x_d - x_{sv}) = \sum_{j=1}^{p} F^j_{ox} \\ M_v\ddot{y}_d + D_{vy}(\dot{y}_d - \dot{y}_{sv}) + K_{vy}(y_d - y_{sv}) = \sum_{j=1}^{p} F^j_{oy} \end{cases} \quad (18)$$

where, $(x_d, y_d)^T$ denotes the location of the virtual point mass and $(x_{sv}, y_{sv})^T$ shows the location on the scaled velocity trajectory to which the point mass is connected. K_{vx} and K_{vy} are the stiffness coefficients of the virtual springs located in x and y directions, D_{vx} and D_{vy} are the damping coefficients of the virtual dampers located in x and y directions and F^j_{ox} and F^j_{oy} are the projections of the j^{th} obstacle force along x and y directions. In this configuration, the impedance coefficients can be chosen to obtain critically damped convergence in all directions. Although it may be a bit challenging to graphically represent the new configuration in higher dimensions, its formulation can be easily extended to more DOF systems. For example, in the case of a system with three translational DOF, the equations of motion of the point mass are simply obtained as:

$$\begin{cases} M_v\ddot{x}_d + D_{vx}(\dot{x}_d - \dot{x}_{sv}) + K_{vx}(x_d - x_{sv}) = \sum_{j=1}^{p} F^j_{ox} \\ M_v\ddot{y}_d + D_{vy}(\dot{y}_d - \dot{y}_{sv}) + K_{vy}(y_d - y_{sv}) = \sum_{j=1}^{p} F^j_{oy} \\ M_v\ddot{z}_d + D_{vz}(\dot{z}_d - \dot{z}_{sv}) + K_{vz}(z_d - z_{sv}) = \sum_{j=1}^{p} F^j_{oz} \end{cases} \quad (19)$$

In this paper, the human-like steering model is used to run the point mass away from the obstacle. According to Fajen and Warren (2003), Huang *et al.* (2006), Fajen *et al.* (2003), Ijspeert *et al.* (2013) and Hoffmann *et al.* (2009), the human-like steering model adds an acceleration, perpendicular to the relative velocity of the EE and the obstacle. The direction of acceleration is obtained by rotating the relative velocity, V_{rel}, by an amount of 90 degrees. As shown in Figure 7, the rotation is performed about an axis perpendicular to the relative velocity and the vector from the virtual mass to the j^{th} obstacle, $\boldsymbol{P}^j_{do}$. The repelling acceleration is given as:

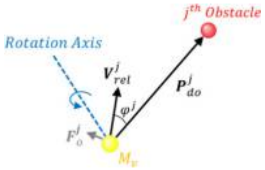
$$\dot{V}^j = \gamma\boldsymbol{R}\boldsymbol{V}^j_{rel}\varphi^j e^{-\alpha_\varphi|\varphi^j|}e^{-\alpha_r|\boldsymbol{P}^j_{do}|} \quad (20)$$

where, γ, α_r and α_φ are three constants, R is the rotation matrix and φ^j is the heading angle of the virtual mass toward the j^{th} obstacle. This acceleration can be simply replaced by an equivalent force as:

$$\boldsymbol{F}^j_o = M_v\dot{\boldsymbol{V}}^j = M_v\gamma\boldsymbol{R}\boldsymbol{V}^j_{rel}\varphi^j e^{-\alpha_\varphi|\varphi^j|}e^{-\alpha_r|\boldsymbol{P}^j_{do}|} \quad (21)$$

Finally, the overall block diagram of the proposed method is depicted in Figure 8, where the FIS takes in the obstacle trajectory, P_o, and determines the time constant, τ, for the DMP. The DMP framework is designed to produce the EE trajectory with scaled velocity, P_{sv}. The desired trajectory of the EE, P_d, is

Figure 7 The repelling force by the human-like steering mechanism



determined as the trajectory of the virtual point mass, moving under the attracting and repelling forces from the obstacles as well as the virtual springs and dampers. However, the position controllers of the joint actuators need the desired and actual joint space trajectories (θ_d and θ_a), to calculate the joint torques, u. Therefore, the IKM of the robot is required. IKM of FUM SCARA is briefly introduced in Appendix 1. Appendix 2 provides a complete list of the symbols and notations used in this paper.

3. Simulations

In this section the performance of proposed method is evaluated in three different sets of simulations by considering:

1 a single stationary obstacle;
2 multiple stationary obstacles; and
3 a single moving obstacle

around the ideal path of the EE. In all the simulation sets, the three-dimensional (3D) path of Figure 9(a) is considered as the

Figure 8 Block diagram of the proposed method

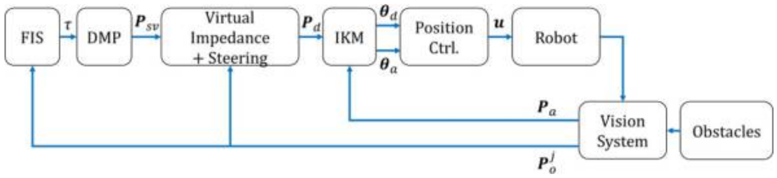
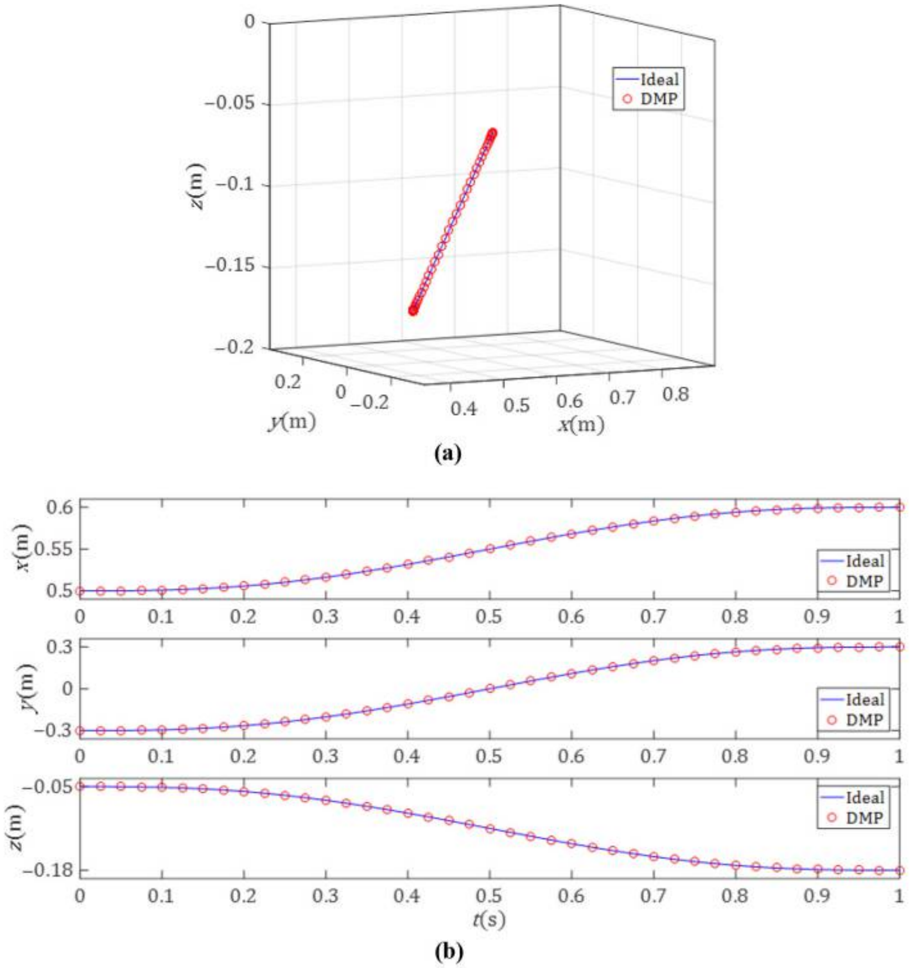


Figure 9 The ideal path used in the simulations



Notes: (a) The 3D ideal path versus the DMP output; (b) the path coordinates versus the DMP outputs ideal EE trajectory. The ideal trajectory is a straight path from the point $(0.5, -0.3, -0.05)^T$ to $(0.6, 0.3, -0.18)^T$ with zero speeds and accelerations in all DOF at the beginning and end points of the path. The ideal trajectory completes in 1 s. Therefore, $\tau_I = 1(s)$.

Figure 9(a) also shows the EE trajectories as learnt by the DMP. Moreover, Figure 9(b) compares the ideal time trajectories of different DOF of the EE with those learnt by the DMP. The good agreement between the ideal trajectories and the ones learnt by the DMP indicates the suitable choice of the DMP parameters. The results are obtained for, $\alpha_\rho = 15$, $\beta_\rho = \alpha_\rho/4$, $N = 300$, $\alpha_s = \alpha_\rho/3$ and $\sigma_i = 1/N$.

For all the simulation sets, the parameters of the virtual impedance and the human-like steering mechanism are set as $M_v = 1(kg)$, $D_{vx} = D_{vy} = D_{vz} = 40(Ns/m)$, $K_{vx} = K_{vy} = K_{vz} = 400$ (N/m), $\gamma = 2 \times 10^6(1/(rad \times s))$, $\alpha_\varphi = 10/\pi(1/rad)$ and $\alpha_r = 30$ $(1/m)$.

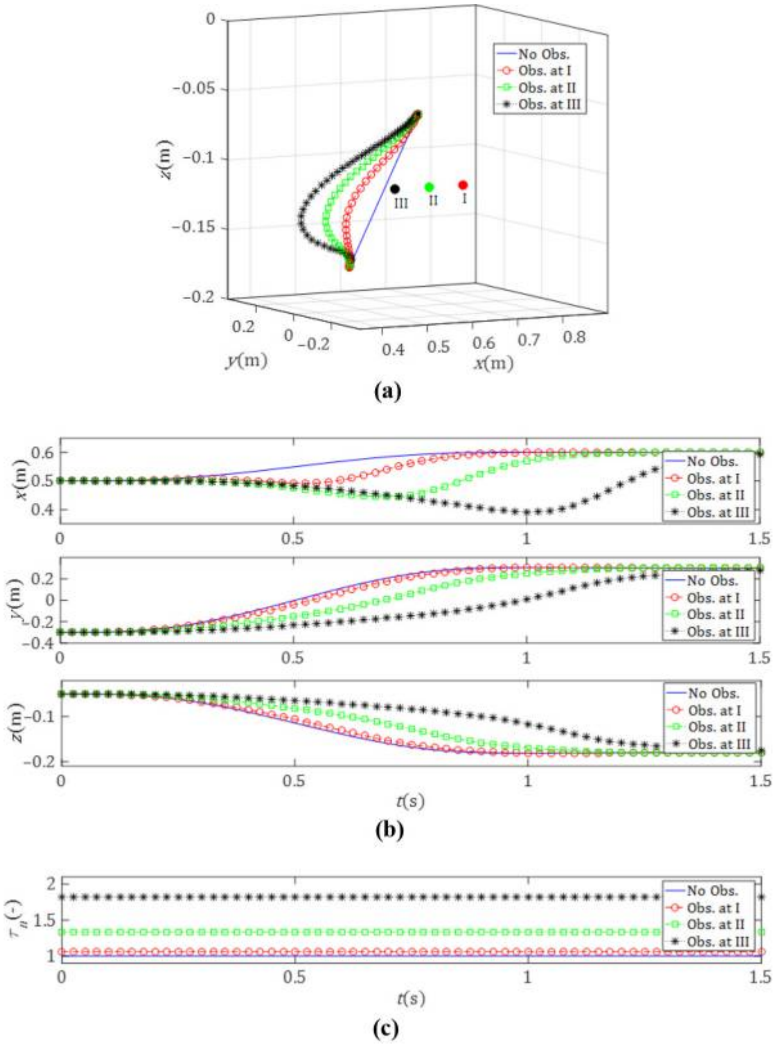
3.1 Single stationary obstacle

In the first set of simulations, a stationary obstacle is put at three different locations around the ideal path. Therefore, the first simulation set includes three separate simulations of evaluating the performance of the proposed method for each location of the obstacle.

The results of the first set of simulations are depicted in Figure 10, where the obstacle locations I, II and III refer to the points with coordinates of $(0.725,0,-0.115)^T$, $(0.65,0,-0.115)^T$ and $(0.575,0,-0.115)^T$, respectively. Figure 10(a) shows the 3D actual paths tracked by the EE, along with its ideal path. It is clear that the proposed algorithm has successfully distracted the EE from its ideal path to prevent it from hitting the obstacles.

Figure 10(b) depicts the time trajectories of the three DOF of the EE in the first simulation set. It is observed that the fuzzy inference engine has successfully altered the time constants according to the obstacles location. As shown in Figure 10(b), when the obstacle is put closer to the workspace, it takes longer for all the DOF to reach their final values. For example, when there is no obstacle, the x DOF of the EE reaches its final value of $0.6(m)$ in about 1 s. However, when the obstacle is put at location III, it takes almost 1.5 s for the x DOF of the EE to get to its final value. Although all the simulations are performed for

Figure 10 Results of the first set of simulations for a single stationary obstacle put at different locations around the ideal path



Notes: (a) The 3D path; (b) the path coordinates; (c) the DMP's time constant

1.5 s, in the case of no obstacle and far obstacles, the robot DOF have clearly reached their final value before the final time of 1.5 s. The delays are appeared because the FIS has down-scaled the EE velocity, as the result of detecting nearby obstacles. Note that no oscillations are observed in the motion of the EE, before and after passing the obstacles.

The successful performance of the fuzzy inference engine is better shown in Figure 10(c), which illustrates the variations of the non-dimensionalized DMP's time constants. As shown in Figure 10(c), the time constant increases when the obstacle is put closer to the workspace. In the first simulation set, the obstacles are stationary. That is why the time constants in Figure 10(c) have fixed vales during the entire motion.

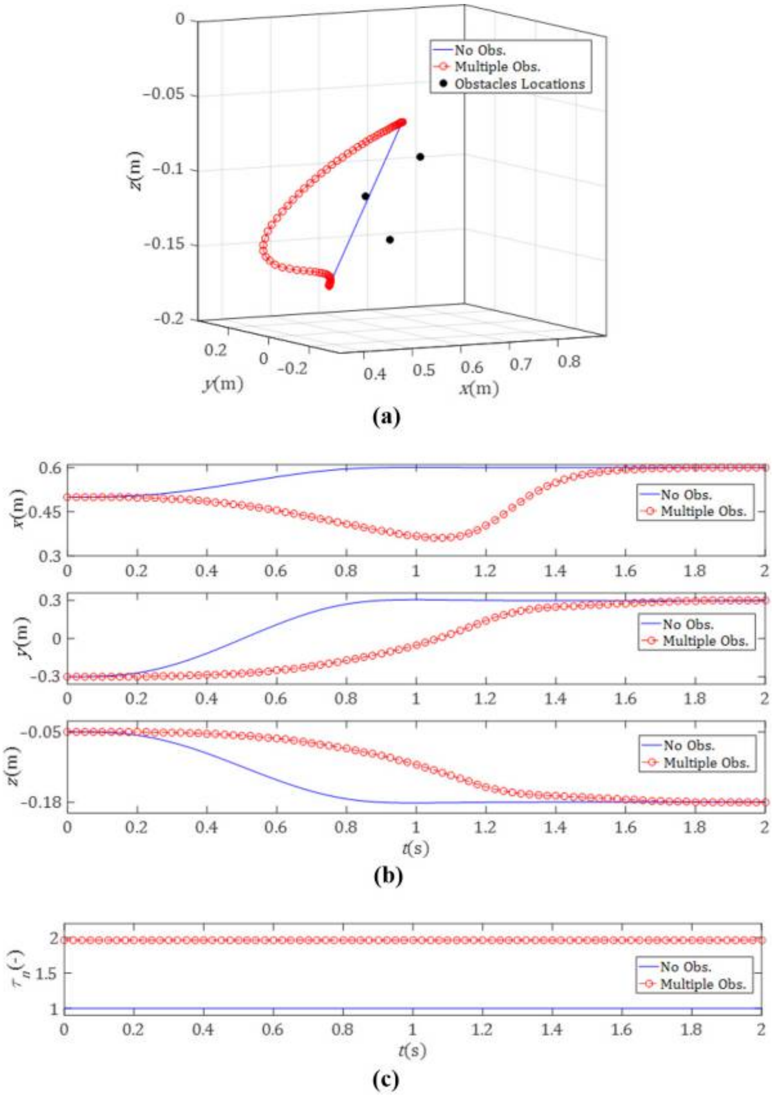
3.2 Multiple stationary obstacles

In the second simulation, multiple stationary obstacles are put around the ideal path. As shown in Figure 11(a), the stationary obstacles are simultaneously put around the ideal path at three points with coordinates of $(0.55, 0, -0.11)^T$, $(0.6, -0.15, -0.08)^T$ and $(0.6, 0, -0.14)^T$. Figure 11(a) also shows the ideal path of the EE as well as the actual path followed by the EE.

Figure 11(b) compares the time trajectories of the robot DOF in the case of multiple obstacles with that of the ideal path. According to Figure 11(b), when the obstacles are put around the ideal path, it clearly takes a longer time for all the robot DOF to get to their final values. This observation along with the increased DMP's time constant, depicted in Figure 11(c), are clear indications of successful performance of the FIS.

Note that the same time constant would have obtained if a single stationary obstacle was put at the coordinate of $(0.55, 0, -0.11)^T$. This is due to the fact that the FIS sets the DMP's time constant according to the most unsafe obstacle. Clearly, in the case of stationary obstacles where all the velocity

Figure 11 Results of the second simulation for multiple stationary obstacles put around the ideal path



Notes: (a) The 3D path; (b) the path coordinates; (c) the DMP's time constant projections are zero, the most unsafe obstacle is the one with the smallest distance from the robot's base.

However, all of the three obstacles contribute in distracting the EE from its ideal path, by applying the repelling forces to the virtual mass. Figure 12(a) illustrates the time history of the components of the repelling forces, applied from each obstacle. The components of the total repelling force are also depicted in Figure 12(a). The magnitude of the total repelling force applied to the virtual mass is shown in Figure 12(b), along with the magnitude of the obstacle forces.

3.3 Single moving obstacle

In the third simulation, a moving obstacle is considered to travel on a circular path around the ideal path. The results of the third simulation are shown in Figure 13.

Figure 13(a) shows the ideal and actual paths of the EE along with the circular path of the moving obstacle. The time trajectories of the robot's DOF are depicted in Figure 13(b) and the time constant of the DMP is depicted in Figure 13(c). Here again, the successful performance of the FIS in increasing the DMP's time constant, or equivalently down scaling the EE velocity, is clearly evident from Figure 13(b) and 13(c).

It is noteworthy that, unlike the stationary obstacles, the moving obstacle moves toward the workspace and away from it. Therefore, the inputs to the FIS (r_n and v_n) change during the simulation and the FIS outputs a varying time constant, as shown in Figure 13(c).

The components and the magnitude of the repelling force applied from the moving obstacle to the virtual mass, are respectively depicted in Figure 14(a) and 14(b).

To gain a better insight of how the robot avoids the moving obstacle, four snapshots of the robot structure are shown if Figure 15. Note that in Figure 15, the desired and the actual paths of the EE and the obstacle trajectory are the same as Figure 13, displayed from another angle of view.

4. Experiments

Practical performance of the proposed method is evaluated in a couple of experiments performed on FUM SCARA robot. This robot is used to run two-dimensional experiments. Therefore, in the following tests, only the two first joints of the robot are actuated and only the two first links of the robot are considered whose lengths are $l_1 = 0.4(m)$ and $l_2 = 0.3(m)$.

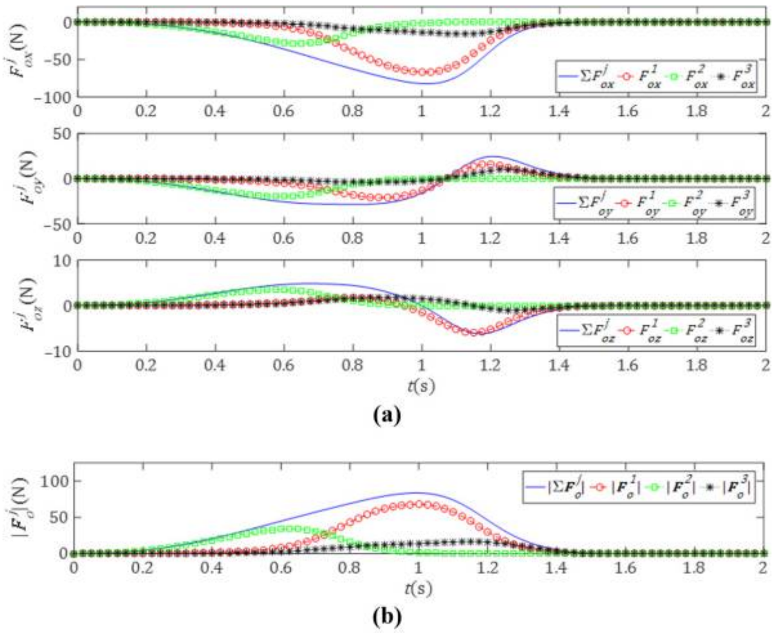
A TSP MDI interface board by tsPishro Company is used to connect the motor drives with a supervising PC on which the proposed method is implemented in a C code running at 50 Hz. The interface board communicates with the PC through an Ethernet connection with User Datagram Protocol (UDP).

Two separate cameras are installed above the workspace. A high-definition (HD) camera of a brand new smart phone is used for monitoring purposes, while a Pixy (CMUcam5) vision system (Lee *et al.*, 2015; Dang and Kwon, 2016) is used to obtain the position of the EE and the obstacle. The Pixy camera outputs the position data at 50 Hz. As shown in Figure 16, a blue ball is used to indicate the location of the EE, and a purple ball is used to indicate the obstacle location.

In both of the experiments, the ideal path of the EE is chosen as a straight path from $\{0.5, -0.3, -0.05\}^T$ to $\{0.6, 0.3, -0.05\}^T$ in 2 s. The z coordinate of the robot is locked at $z = -0.05$ to obtain a 2D motion and the Pixy camera is calibrated to give x and y data at the height of the indicating balls rather than the actual position of the EE.

As shown in Figure 17, the coordinate system W, whose origin is located at the robot base, is defined for the Pixy camera to report the locations of the balls. The Pixy Camera is then calibrated to measure the coordinates of the balls with respect

Figure 12 Repelling forces applied to the virtual mass from multiple stationary obstacles

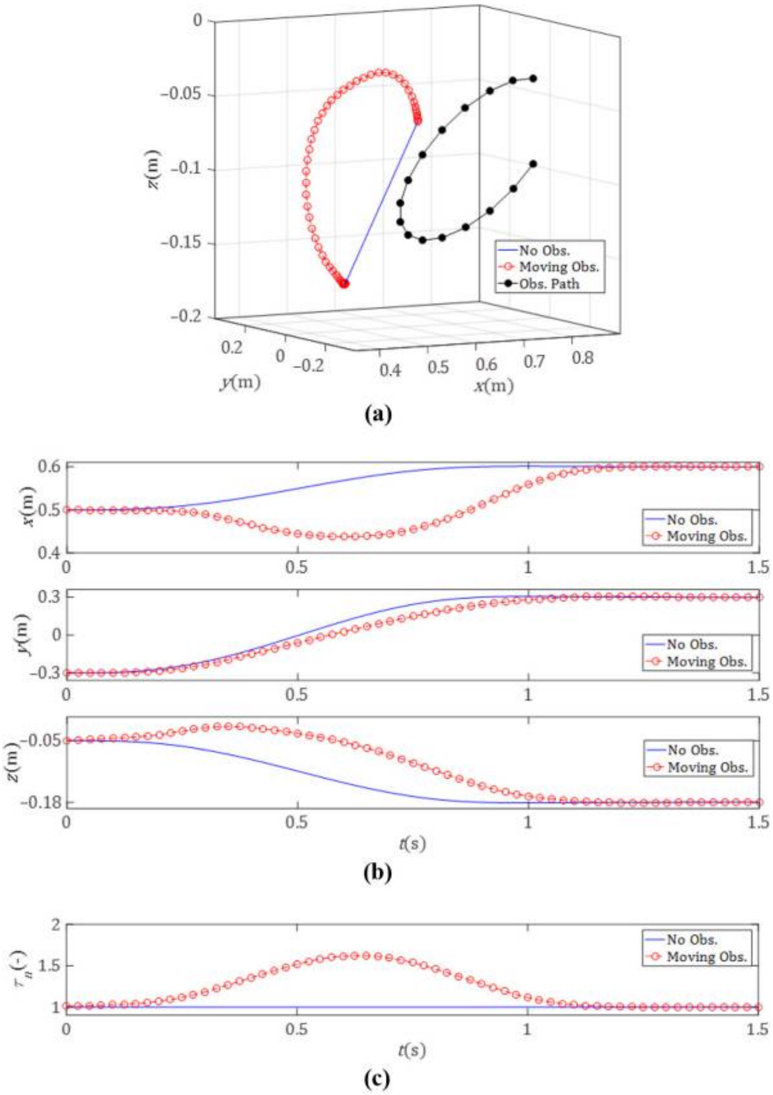


(a)

(b)

Notes: (a) Force components; (b) force magnitudes

Figure 13 Results of the third simulation for a moving obstacle



Notes: (a) The 3D path; (b) the path coordinates; (c) the DMP's time constant

to the origin of the coordinate system. The distance from obstacle to the robot's base is then calculated as $|\boldsymbol{P}_o^j| = \sqrt{\left(x_o^j\right)^2 + \left(y_o^j\right)^2}$ where, $\left(x_o^j,\ y_o^j\right)^T$ is the location of the obstacle, reported by Pixy camera. The position vector of the obstacle is also formed as $\boldsymbol{P}_o^j = \left(x_o^j \quad y_o^j\right)^T$. The obstacle velocity vector is calculated by subtracting two successive position vectors and dividing by the sampling time of the Pixy camera, i.e. 20 ms. Therefore, $\dot{\boldsymbol{P}}_o^j = \left(\boldsymbol{P}_o^j(i+1) - \boldsymbol{P}_o^j(i)\right)/0.02 = \left(\dot{x}_o^j \quad \dot{y}_o^j\right)^T$. Finally, the velocity projection is obtained by calculating the inner product $\boldsymbol{P}_o^j \cdot \dot{\boldsymbol{P}}_o^j = x_o^j \dot{x}_o^j + y_o^j \dot{y}_o^j$.

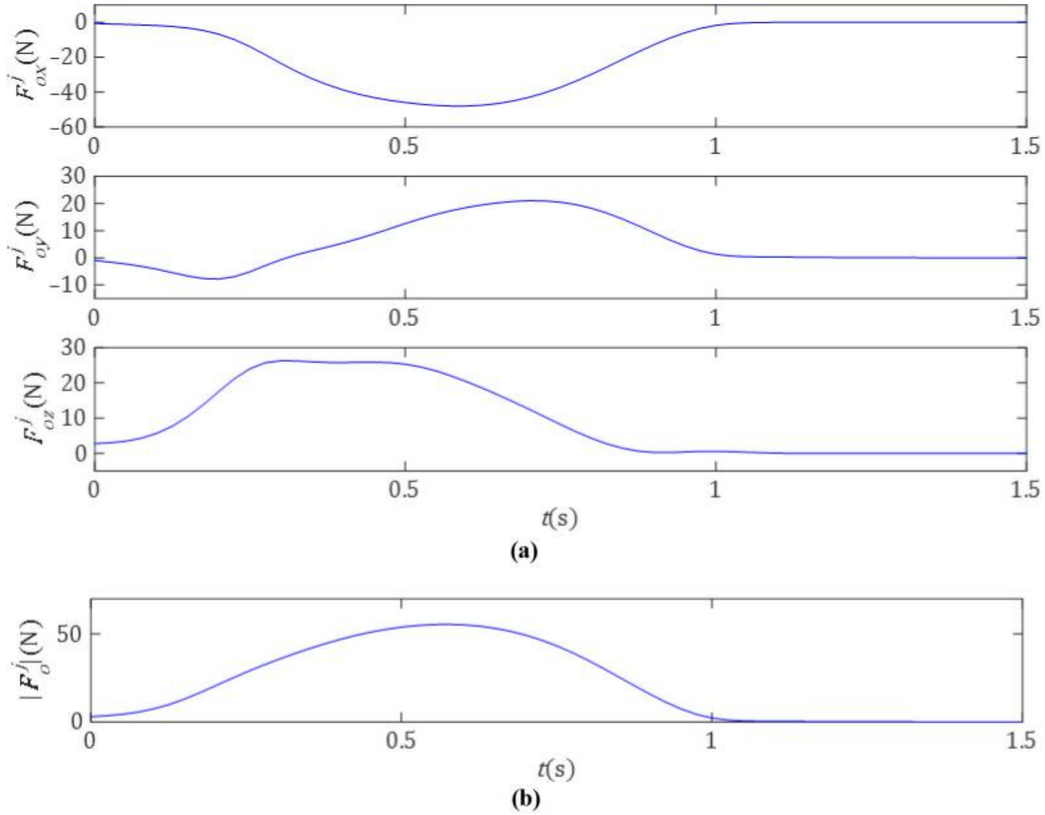
In the first experiment, the obstacle ball is positioned stationary in the workspace at the same height of the EE ball. In the second experiment, an operator holds the obstacle ball at the same height of the EE ball and moves it arbitrarily around the ideal path. Practical performance of the proposed method is illustrated in Figures 18 and 19 by providing some snapshots of the FUM SCARA avoiding the stationary and the moving obstacles, respectively.

The snapshots in Figures 18 and 19 show that the proposed method has successfully distracted the EE from the ideal path to avoid collision with the obstacle.

In the first experiment, the robot has reached to the final position in about 3.1 s, while in the second experiment, it took about 3.5 s. It could be justified by the fact that in the second experiment, the operator has put the obstacle ball closer to the workspace, compared to the first experiment.

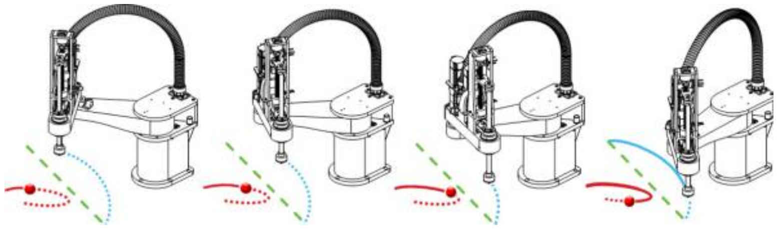
As previously mentioned, the method is implemented in a C code running at 50 Hz. Therefore, it is clear that each cycle of the proposed method requires less than 20 ms in a PC with 4

Figure 14 Repelling forces applied to the virtual mass from the moving obstacle



Notes: (a) Force components; (b) force magnitudes

Figure 15 Snapshots of the virtual model of FUM SCARA, avoiding the simulated moving obstacle



gigabytes of RAM and a 2.4 gigahertz CPU. However, it is not an easy task to calculate the exact required time for each cycle of the proposed method in a C code because of the latencies caused by the Windows operating system. To provide a better estimation of the cycle time, the proposed method is implemented in a Simulink real-time (formerly known as XPc Target) scheme where no operating system is active other than the Simulink code. In this approach, the required time for each cycle of the proposed method, excluding the time needed for obtaining the outputs of the Pixy camera, is calculated about 30 ms. Note that this cycle time is obtained in a relatively powerful PC and will definitely increase in the case of using a weaker PC or using a programmable logic controller (PLC). Therefore, the proposed method has a very low computational load and the overall cycle time of the method is governed by the time required by the vision system to report the locations.

The provided experiments evaluate the practical performance of the proposed method in 2D motions. In the case of 3D motions in real environments, there will be a need for extra cameras and more advanced vision systems to obtain the 3D locations of the EE and the obstacles. If feedbacks from joint angles of the robot (encoder feedbacks from joint motors) are accessible, it may be easier to use forward kinematic model of the robot to calculate the location of the EE. In such cases, the vision system will be only responsible for obtaining the locations of the obstacles.

5. Conclusion

In this paper, a novel method is proposed to implement the concepts of velocity scaling and obstacle avoidance to increase the safety of the robots' operation. A fuzzy DMP framework is

Figure 16 Test-bed for performing the experiments

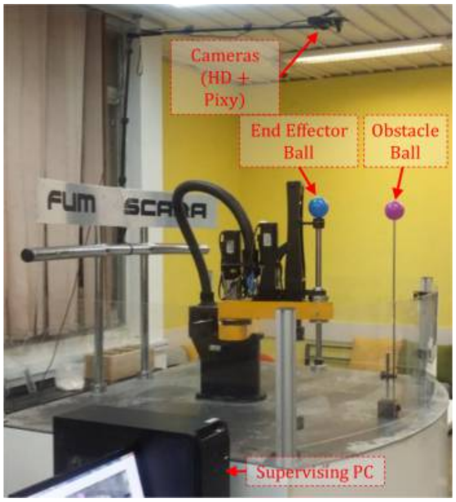


Figure 17 the coordinate system W, defined for the Pixy camera to report the locations

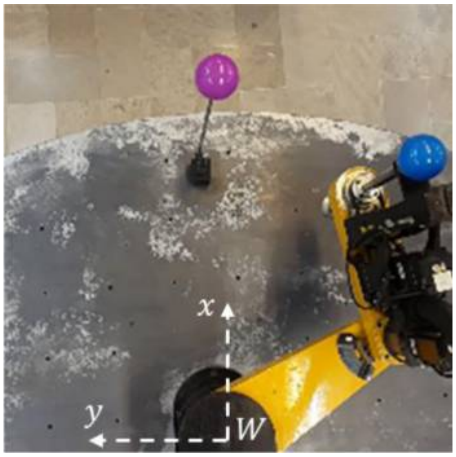


Figure 18 Snapshots of FUM SCARA avoiding the stationary obstacle

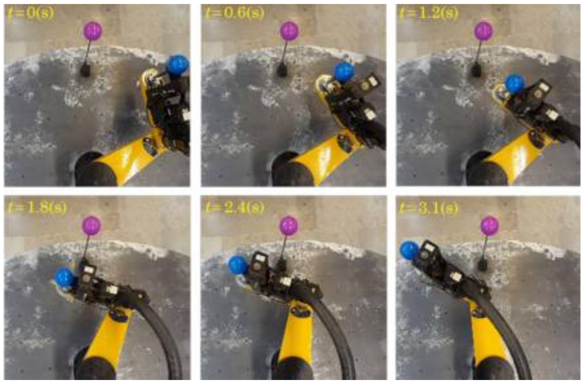
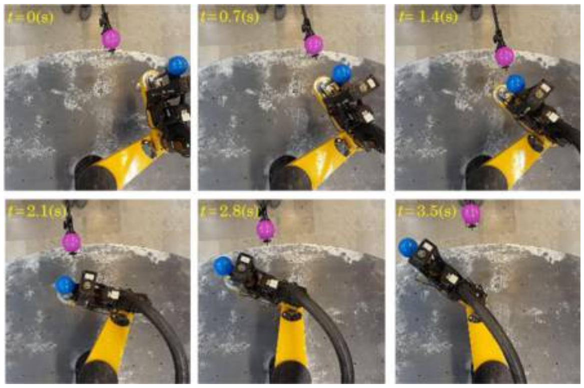


Figure 19 Snapshots of FUM SCARA avoiding the moving obstacle



proposed for downscaling the EE velocity when nearby obstacles are detected. The velocity projection of the obstacle and its distance from the workspace are considered as inputs to a FIS which determines the DMP's time constant. The fuzzy DMP framework is then combined with a new configuration of the virtual impedance, enabling a non-oscillatory convergence of the EE to the desired path in all directions. The human-like steering mechanism is used to apply repelling forces to the virtual impedance. The combination of the fuzzy DMP framework with the virtual impedance and the human-like steering mechanisms provides the velocity scaling and the obstacle avoidance capabilities for the robot.

Some simulation and experiments are performed on a custom made SCARA type robot. The performance of the proposed method is evaluated in single and multiple stationary obstacles as well as moving obstacles. The obtained results clearly verify the practical performance of the proposed method. The EE speed is effectively downscaled when nearby obstacles are detected and the obstacles are successfully avoided.

Velocity scaling and obstacle avoidance are two common safety considerations for the robots working in unstructured environments. These considerations are simultaneously handled by the proposed method. Simple structure and low computational cost of the DMP frame work and virtual impedance technique provide the capability of real-time implementation for the proposed method with medium to low range supervisory controllers. Moreover, the proposed method simply synchronizes the velocity scaling among the entire robot's DOF while maintaining the smoothness and continuity of the trajectory. This removes the need for extra formulations to ensure the DOF synchronization and to guarantee the trajectory smoothness and continuity. This is another aspect of the simplicity of the proposed method which facilitates its low-cost real-time implementation. Consequently, it may be possible to implement the proposed method on industrial robots with already developed supervisory controllers having limited performances.

In this paper, the collision is only considered between the end effector and obstacles. However, future works may focus on avoiding the collision of robot's links with the obstacles as well as the collisions between the robot's links. Practical implementation of the proposed method on a robot with more DOF and practical evaluation of the possibility of implementing the proposed method on a low range PLC may also be focused in the future works.

References

Abu-Dakka, F.J., Nemec, B., Kramberger, A., Buch, A.G., Krüger, N. and Ude, A. (2014), "Solving peg-in-hole tasks by human demonstration and exception strategies", *Industrial Robot: An International Journal*, Vol. 41 No. 6, pp. 575-584.

Arai, T., Ogata, H. and Suzuki, T. (1989), "Collision avoidance among multiple robots using virtual impedance", *IEEE/RSJ International Workshop on Intelligent Robots and Systems (IROS)*, 4-6 September, IEEE, Tsukuba, pp. 479-485.

Basa, D. and Schneider, A. (2015), "Learning point-to-point movements on an elastic limb using dynamic movement primitives", *Robotics and Autonomous Systems*, Vol. 66, pp. 55-63.

Brock, O. and Khatib, O. (2002), "Elastic strips: a framework for motion generation in human environments", *The International Journal of Robotics Research*, Vol. 21 No. 12, pp. 1031-1052.

Cai, C., Somani, N., Nair, S., Mendoza, D. and Knoll, A. (2014), "Uncalibrated stereo visual servoing for manipulators using virtual impedance control", *13th International Conference on Control Automation Robotics & Vision (ICARCV)*, 10-12 December, IEEE, Singapore, pp. 1888-1893.

Dang, L. and Kwon, J. (2016), "Design of a new cost-effective head for a low-cost humanoid robot", *Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 20-22 October, IEEE, New York, NY, pp. 1-7.

Deniša, M., Gams, A., Ude, A. and Petrič, T. (2016), "Learning compliant movement primitives through demonstration and statistical generalization", *IEEE/ASME Transactions on Mechatronics*, Vol. 21 No. 5, pp. 2581-2594.

Fajen, B.R. and Warren, W.H. (2003), "Behavioral dynamics of steering, obstable avoidance, and route selection", *Journal of Experimental Psychology: Human Perception and Performance*, Vol. 29 No. 2, p. 343.

Fajen, B.R., Warren, W.H., Temizer, S. and Kaelbling, L.P. (2003), "A dynamical model of visually-guided steering, obstacle avoidance, and route selection", *International Journal of Computer Vision*, Vol. 54 Nos 1/3, pp. 13-34.

Gaskill, S. (1994), "Safety issues in modern applications of robots", IEE Colloquium on Safety and Reliability of Complex Robotic Systems, 6 April, IET, London, pp. 5/1-5/13.

Hoffmann, H. and Mitchell, D. (2016), "Dynamic obstacle avoidance in a robotic system", GM Global Technology Operations LLC, Patent, US9403275 B2.

Hoffmann, H., Pastor, P., Park, D.-H. and Schaal, S. (2009), "Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance", *IEEE International Conference on Robotics and Automation (ICRA)*, 12-17 May, IEEE, Kobe, pp. 2587-2592.

Huang, R., Cheng, H., Guo, H., Chen, Q. and Lin, X. (2016), "Hierarchical interactive learning for a human-powered augmentation lower exoskeleton", *IEEE International Conference on Robotics and Automation (ICRA)*, 16-21 May, IEEE, Stockholm, pp. 257-263.

Huang, W.H., Fajen, B.R., Fink, J.R. and Warren, W.H. (2006), "Visual navigation and obstacle avoidance using a steering potential function", *Robotics and Autonomous Systems*, Vol. 54 No. 4, pp. 288-299.

Ijspeert, A.J., Nakanishi, J. and Schaal, S. (2002), "Learning rhythmic movements by demonstration using nonlinear oscillators", *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 30 September-4 October, IEEE, Lausanne, pp. 958-963.

Ijspeert, A.J., Nakanishi, J. and Schaal, S. (2003), "Learning attractor landscapes for learning motor primitives", *Advances in neural information processing systems*, Vol. 15, pp. 1547-1554.

Ijspeert, A.J., Nakanishi, J., Hoffmann, H., Pastor, P. and Schaal, S. (2013), "Dynamical movement primitives: learning attractor models for motor behaviors", *Neural Computation*, Vol. 25 No. 2, pp. 328-373.

Jiang, M., Chen, Y., Zheng, W., Wu, H. and Cheng, L. (2016), "Mobile robot path planning based on dynamic movement primitives", *IEEE International Conference on Information and Automation (ICIA)*, 1-3 August, IEEE, Ningbo, pp. 980-985.

Kamali, K., Akbari, A.A. and Akbarzadeh, A. (2016), "Trajectory generation and control of a knee exoskeleton based on dynamic movement primitives for sit-to-stand assistance", *Advanced Robotics*, Vol. 30 No. 13, pp. 846-860.

Khansari-Zadeh, S.M. and Khatib, O. (2017), "Learning potential functions from human demonstrations with encapsulated dynamic and compliant behaviors", *Autonomous Robots*, Vol. 41 No. 1, pp. 45-69.

Khatib, O. (1986), "Real-time obstacle avoidance for manipulators and mobile robots", *The International Journal of Robotics Research*, Vol. 5 No. 1, pp. 90-98.

Kim, J.-O. and Khosla, P.K. (1992), "Real-time obstacle avoidance using harmonic potential functions", *IEEE Transactions on Robotics and Automation*, Vol. 8 No. 3, pp. 338-349.

Kramberger, A., Kramberger, A., Piltaver, R., Piltaver, R., Nemec, B., Nemec, B., Gams, M., Gams, M., Ude, A. and Ude, A. (2016), "Learning of assembly constraints by demonstration and active exploration", *Industrial Robot: An International Journal*, Vol. 43 No. 5, pp. 524-534.

Krug, R. and Dimitrov, D. (2015), "Model predictive motion control based on generalized dynamical movement primitives", *Journal of Intelligent & Robotic Systems*, Vol. 77 No. 1, p. 17.

Kulić, D. (2006), "Safety for human-robot interaction", PHD Thesis, University of British Columbia, Vancouver.

Kulić, D. and Croft, E. (2007), "Pre-collision safety strategies for human-robot interaction", *Autonomous Robots*, Vol. 22 No. 2, pp. 149-164.

Kulvicius, T., Biehl, M., Aein, M.J., Tamosiunaite, M. and Wörgötter, F. (2013), "Interaction learning for dynamic movement primitives used in cooperative robotic tasks", *Robotics and Autonomous Systems*, Vol. 61 No. 12, pp. 1450-1459.

Lacevic, B., Rocco, P. and Zanchettin, A.M. (2013), "Safety assessment and control of robotic manipulators using danger field", *IEEE Transactions on Robotics*, Vol. 29 No. 5, pp. 1257-1270.

Lee, S., Tewolde, G.S., Lim, J. and Kwon, J. (2015), "Vision based localization for multiple mobile robots using low-cost vision sensor", *IEEE International Conference on Electro/Information Technology (EIT)*, 21-23 May, IEEE, Dekalb, IL, pp. 280-285.

Li, C., Lowe, R. and Ziemke, T. (2014), "A novel approach to locomotion learning: actor-critic architecture using central pattern generators and dynamic motor primitives", *Frontiers in Neurorobotics*, Vol. 8.

Lo, S.-Y., Cheng, C.-A. and Huang, H.-P. (2016), "Virtual impedance control for safe human-robot interaction", *Journal of Intelligent & Robotic Systems*, Vol. 82 No. 1, pp. 3-19.

Maeda, G.J., Neumann, G., Ewerton, M., Lioutikov, R., Kroemer, O. and Peters, J. (2016), "Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks", *Autonomous Robots*, Vol. 41 No. 3, pp. 1-20.

Mousavi, A., Akbarzadeh, A., Shariatee, M. and Alimardani, S. (2015a), "Design and construction of a linear-rotary joint for robotics applications", *3rd RSI International Conference on Robotics and Mechatronics (ICROM)*, 7-9 October, IEEE, Tehran, pp. 229-233.

Mousavi, A., Akbarzadeh, A., Shariatee, M. and Alimardani, S. (2015b), "Repeatability analysis of a SCARA robot with planetary gearbox", *3rd RSI International Conference on Robotics and Mechatronics (ICROM)*, 7-9 October, IEEE, Tehran, pp. 640-644.

Mukovskiy, A., Vassallo, C., Naveau, M., Stasse, O., Soueres, P. and Giese, M.A. (2017), "Adaptive synthesis of dynamically feasible full-body movements for the humanoid robot HRP-2 by flexible combination of learned dynamic movement primitives", *Robotics and Autonomous Systems*, Vol. 91, pp. 270-283.

Nemec, B. and Ude, A. (2012), "Action sequencing using dynamic movement primitives", *Robotica*, Vol. 30 No. 5, pp. 837-846.

Park, D.-H., Hoffmann, H., Pastor, P. and Schaal, S. (2008), "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields", *8th IEEE-RAS International Conference on Humanoid Robots*, 1-3 December, IEEE, Daejeon, pp. 91-98.

Pastor, P., Kalakrishnan, M., Meier, F., Stulp, F., Buchli, J., Theodorou, E. and Schaal, S. (2013), "From dynamic movement primitives to associative skill memories", *Robotics and Autonomous Systems*, Vol. 61 No. 4, pp. 351-361.

Pignat, E. and Calinon, S. (2017), "Learning adaptive dressing assistance from human demonstration", *Robotics and Autonomous Systems*, Vol. 93, pp. 61-75.

Rosado, J., Silva, F., Santos, V. and Amaro, A. (2016), "Adaptive robot biped locomotion with dynamic motion primitives and coupled phase oscillators", *Journal of Intelligent & Robotic Systems*, Vol. 83 Nos 3/4, pp. 375-391.

Schaal, S., Mohajerian, P. and Ijspeert, A. (2007), "Dynamics systems vs. optimal control – a unifying view", *Progress in Brain Research*, Vol. 165, pp. 425-445.

Shariatee, M., Akbarzadeh, A., Mousavi, A. and Alimardani, S. (2014), "Design of an economical SCARA robot for industrial applications", *2nd RSI/ISM International Conference on Robotics and Mechatronics (ICROM)*, 15-17 October, Tehran, IEEE, pp. 534-539.

Sun, W., Torres, L.G., Van Den Berg, J. and Alterovitz, R. (2016), "Safe motion planning for imprecise robotic manipulators by minimizing probability of collision", in Inaba, M. and Croke, P. (Eds), *Robotics Research*, Springer, Berlin, pp. 685-701.

Tamosiunaite, M., Nemec, B., Ude, A. and Wörgötter, F. (2011), "Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives", *Robotics and Autonomous Systems*, Vol. 59 No. 11, pp. 910-922.

Tan, H., Erdemir, E., Kawamura, K. and Du, Q. (2011), "A potential field method-based extension of the dynamic movement primitive algorithm for imitation learning with obstacle avoidance", *International Conference on Mechatronics and Automation (ICMA)*, 7-10 August, IEEE, Beijing, pp. 525-530.

Ude, A., Gams, A., Asfour, T. and Morimoto, J. (2010), "Task-specific generalization of discrete and periodic dynamic movement primitives", *IEEE Transactions on Robotics*, Vol. 26 No. 5, pp. 800-815.

Zanchettin, A.M., Ceriani, N.M., Rocco, P., Ding, H. and Matthias, B. (2016), "Safety in human-robot collaborative manufacturing environments: metrics and control", *IEEE Transactions on Automation Science and Engineering*, Vol. 13 No. 2, pp. 882-893.

Zavlangas, P.G. and Tzafestas, S.G. (2000), "Industrial robot navigation and obstacle avoidance employing fuzzy logic", *Journal of Intelligent and robotic Systems*, Vol. 27 Nos 1/2, pp. 85-97.

Appendix 1. Inverse kinematic model of FUM SCARA

In this paper, the performance of the proposed algorithm is evaluated through implementation on FUM SCARA. The required parameters for defining the IKM of FUM SCARA are illustrated in Figure A1. Knowing the joint angles, the position and orientation of the EE are obtained as follows.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} l_1c_1 + l_2c_{12} \\ l_1s_1 + l_2s_{12} \\ -d \end{bmatrix} \tag{A1}$$

where, $c_1 = \cos(\theta_1)$, $s_1 = \sin(\theta_1)$, $c_{12} = \cos(\theta_1+\theta_2)$ and $s_{12} = \sin(\theta_1+\theta_2)$.

Therefore, the IKM of the FUM SCARA is obtained as the following closed-form equations

$$c_2 = \frac{\left(x^2 + y^2 - l_1^2 - l_2^2\right)}{2l_1l_2} \tag{A2}$$

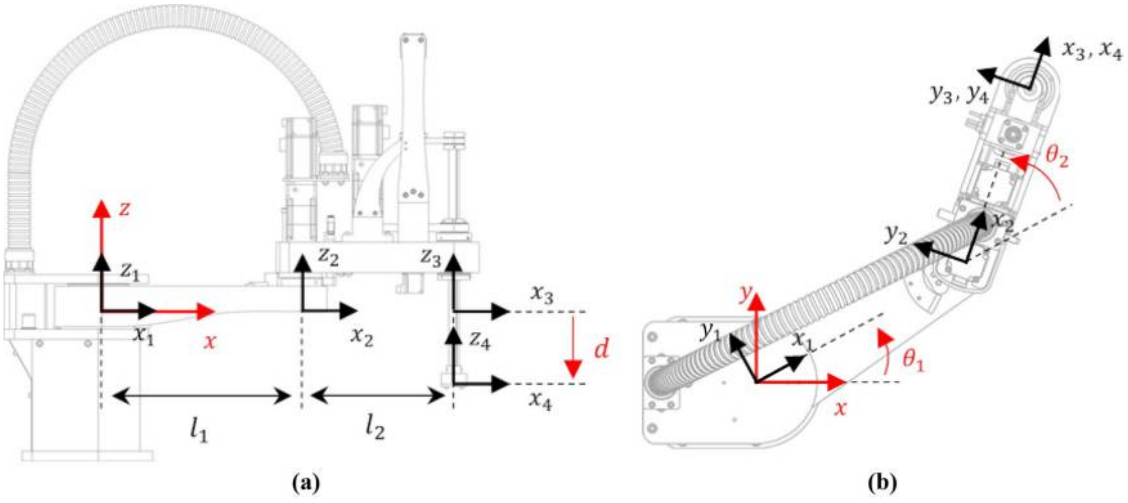
$$\theta_2 = atan2\left(\sqrt{1-c_2^2}, c_2\right) \tag{A3}$$

$$\theta_1 = atan2(y, x) - atan2(l_2s_2, l_1 + l_2c_2) \tag{A4}$$

$$d = -z \tag{A5}$$

in which $c_2 = \cos(\theta_2)$.

Figure A1 Kinematic parameters of FUM SCARA



Notes: (a) Side view; (b) top view

Appendix 2

Table AI Symbols and notations

Symbols	Description
Acronyms	
DMP	Dynamic movement primitive
EE	End-effector
DOF	Degree(s)-of-freedom
IKM	Inverse kinematic model
FIS	Fuzzy inference system
Variables	
τ	DMP's time constant
τ_n	Non-dimensionalized DMP's time constant
τ_I	Ideal DMP's time constant (duration of the ideal trajectory)
α_ρ, β_ρ	Constants affecting learning and generalization of the DMP
$\boldsymbol{P}$	Vector of trajectory variables
$\boldsymbol{P}_{goal}$	Vector of the goals values of the trajectory variables
$\boldsymbol{P}_{init}$	Vector of the initial values of the trajectory variables
$\boldsymbol{P}_q, \ddot{\boldsymbol{P}}_q, \ddot{\boldsymbol{P}}_q$	Position, velocity and acceleration vectors of the q^{th} sample of the ideal trajectory, used for teaching the DMP
$\boldsymbol{P}_I$	Ideal EE trajectory
$\boldsymbol{P}_{sv}$	Ideal EE trajectory with scaled velocity
$(x_{sv}, y_{sv}, z_{sv})^T$	The point on P_{sv} to which the virtual point mass is connected
$\boldsymbol{P}_d$	Desired EE trajectory (trajectory of the virtual point mass)
$(x_d, y_d, z_d)^T$	Location of the virtual point mass
$\boldsymbol{P}_a$	Actual EE trajectory
$\boldsymbol{P}_o^j$	Trajectory of the j^{th} obstacle
$\left(x_o^j, y_o^j, z_o^j\right)^T$	Location of the j^{th} obstacle
$\boldsymbol{P}$	State vector of the DMP system
$\boldsymbol{f}$	Vector of forcing terms in DMP
$\boldsymbol{f}_q^*$	Target value of the forcing term for the q^{th} sample of the ideal trajectory
$\psi_i(s)$	DMP's i^{th} basis function
$\boldsymbol{w}_i$	i^{th} column of the DMP's weighting matrix
$\boldsymbol{W}$	DMP's weighting matrix
σ_i	width of the i^{th} basis function of the DMP
c_i	center location of the i^{th} basis function of the DMP
s	DMP's phase variable
s_q	Phase variable for the q^{th} sample of the ideal trajectory
α_s	DMP's phase coefficient
Ψ_i	Accumulated matrix of i^{th} basis function
Γ_k	Accumulated vector of the target forcing value for k^{th} DOF
Λ_k	Accumulated vector of $\xi_q(k)$ for k^{th} DOF
ξ_q	Scaled phase variable for the q^{th} sample of the ideal trajectory
r_n	Non-dimensionalized distance from obstacle to the robot's base
r_w	Radius of the workspace sphere
v_n	Non-dimensionalized projected obstacle velocity
$\lvert(v_e)_{max}\rvert$	Absolute maximum velocity of the EE
$A_l \in \{F, C, VC\}$	Membership functions of the input r_n, corresponding to the l^{th} rule of the FIS (VC = very close, C = close, F = far)
$B_l \in \{P, Z, N\}$	Membership functions of the input v_n, corresponding to the l^{th} rule of the FIS (P = positive, Z = zero, N = negative)
$C_l \in \{T_1, T_2, T_3, T_4\}$	Membership functions of the output τ_n, corresponding to the l^{th} rule of the FIS (T1 = 1, T2 = 2, T3 = 3, T4 = 4)
b_l	Center value of the output membership functions, corresponding to the l^{th} rule of the FIS
θ_d	Desired joints trajectory
U	Joints torques
M_v, K_v, D_v	Mass, stiffness and damping coefficients of the conventional virtual impedance
K_{vx}, K_{vy}, K_{vz}	Stiffness of the virtual springs located in x, y and z direction
D_{vx}, D_{vy}, D_{vz}	Damping of the virtual dampers located in x, y and z direction
$\boldsymbol{F}_o^j = \left[F_{ox}^j, F_{oy}^j, \ F_{oz}^j\right]^T$	Force vector from j^{th} obstacle to the virtual mass

(***continued***)

Table AI

Symbols	Description
$\boldsymbol{V}^{j}_{rel}$	Relative velocity of the EE with respect to j^{th} obstacle
$\boldsymbol{P}^{j}_{do}$	Position vector of j^{th} obstacle relative to the EE
$\ddot{V}^{j}$	Repelling acceleration from j^{th} obstacle
φ^{j}	Angle formed between vectors $\boldsymbol{V}^{j}_{rel}$ and $\boldsymbol{P}^{j}_{do}$
γ, α_{r}, α_{φ}	Constant coefficients in the human-like steering mechanism
$\boldsymbol{R}$	Rotation matrix of the human-like steering mechanism
l_1, l_2	Lengths of the two first links of FUM SCARA
d	Translational displacement of the third link of FUM SCARA
Subscripts	
$i = 1...N$	i^{th} basis function
$q = 1...m$	q^{th} sample of the ideal trajectory, used for teaching the DMP
$k = 1, 2, 3$	k^{th} DOF of the EE corresponding to x, y and z DOF
$l = 1...9$	l^{th} rule of the FIS
$j = 1...p$	j^{th} obstacle

Corresponding author

Alireza Akbarzadeh can be contacted at: Ali_Akbarzadeh@um.ac.ir