Comparative study of various machine learning algorithms and Denavit–Hartenberg approach for the inverse kinematic solutions in a 3-PPSS parallel manipulator

Mervin Joe Thomas, Mithun M. Sanjeev, A.P. Sudheer and Joy M.L.
Department of Mechanical Engineering, National Institute of Technology Calicut, India

Abstract

Purpose – This paper aims to use different machine learning (ML) algorithms for the prediction of inverse kinematic solutions in parallel manipulators (PMs) to overcome the computational difficulties and approximations involved with the analytical methods. The results obtained from the ML algorithms and the Denavit–Hartenberg (DH) approach are compared with the experimental results to evaluate their performances. The study is performed on a novel 6-degree of freedom (DoF) PM that offers precise motions with a large workspace for the end effector.

Design/methodology/approach – The kinematic model for the proposed 3-PPSS PM is obtained using the modified DH approach and its inverse kinematic solutions are determined using the Levenberg–Marquardt algorithm. Various prediction algorithms such as the multiple linear regression, multi-variate polynomial regression, support vector, decision tree, random forest regression and multi-layer perceptron networks are applied to predict the inverse kinematic solutions for the manipulator. The data set required to train the network is generated experimentally by recording the poses of the end effector for different instantaneous positions of the slider using the concept of ArUco markers.

Findings – This paper fully demonstrates the possibility to use artificial intelligence for the prediction of inverse kinematic solutions especially for complex geometries.

Originality/value – As the analytical models derived from the geometrical method, Screw theory or numerical techniques involve approximations and needs more computational power, it is not advisable for real-time control of the manipulator. In addition, the data set obtained from the derived inverse kinematic equations to train the network may lead to inaccuracies in the predicted results. This error may generate significant deviations in the end-effector position from the desired position. The present work attempts to resolve this issue by proposing a camera-based approach that uses ArUco library and ML algorithms to create the data set experimentally and predict the inverse kinematic solutions accurately.

Keywords Robot design, Neural networks, Parallel manipulators, Denavit–Hartenberg modelling, Machine learning, ArUco library, Multi-layer perceptron

Paper type Technical paper

1. Introduction

Parallel manipulators (PMs) offer better characteristics over serial manipulators in terms of overall rigidity, dynamic performance, payloads and accuracy (Xu *et al.*, 2017). The closed configurations of most 6-DoF PMs have complicated forward kinematics, coupled motions for the end effector and limited workspace. The Gough mechanism was the first 6-DoF PM introduced in 1956, followed by the Stewart mechanism proposed in 1965 used for flight simulation applications (Han *et al.*, 2019). The Stewart mechanism, which is a 6-UPS (U-Universal; P-Prismatic; S-Spherical joints) manipulator suffer from high inertia effects, as it has six active prismatic joints located on the moving part of its kinematic chain needing more advanced and costly actuation mechanisms. The 6-PUS PM (Nabavi *et al.*, 2018) has the active prismatic joints kept fixed to the ground. This modification to the conventional Stewart mechanism improves its stiffness, static balance and reduces the overall cost of the manipulator (Furqan *et al.*, 2017). By replacing the passive universal joints in the Stewart mechanism with active joints, the number of legs was reduced from 6 to 3 (Abedinnasab *et al.*, 2017). This modification lightens the mechanism and lowers inertia effects. Various approaches are seen in the literature for the computation of direct kinematics of PMs. Most of these approaches can be categorised into either analytical or numerical methods. Many literatures use Screw theory to obtain the kinematic equations for various configurations (Chen *et al.*, 2015)

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

Industrial Robot: the international journal of robotics research and application

[DOI 10.1108/IR-11-2019-0233]

Received 18 November 2019
Revised 9 March 2020
17 April 2020
14 May 2020
Accepted 15 May 2020

(Gallardo-Alvarado *et al.*, 2016). Although this methodology is applicable to complex geometries and has reduced computation to solve for the inverse kinematics, it is most suitable when the manipulator consists of active joints only (Thomas *et al.*, 2020). This is because the kinematic model obtained by this approach does not include passive joint variables. Sylvester dialytic elimination method (Huang *et al.*, 2010) is another approach to get the kinematic model in which the equations are reduced to a univariate polynomial to reduce the computational time required to solve the inverse kinematics. However, this methodology does not follow a general procedure and is challenging to apply to complex geometries.

Even though there are a lot of research studies happening in the development of novel PMs, quite a few spatial mechanisms exist that can overcome the benefits of serial manipulators. In many PMs, the 6-DoF motions of the end effector are usually coupled together because of its geometric design (Sukumar *et al.*, 2019). In this paper, a novel 6-DoF 3-PPSS PM is proposed that can produce decoupled motions for the end effector by following a particular order of actuation for each slider. The kinematic model for the manipulator is obtained using the Denavit–Hartenberg (DH) modelling technique. It has been showcased in this paper that getting the inverse kinematic solutions for PMs using the analytical approach is a cumbersome task (Van Toan and Khoi, 2018). In addition, the analytical approach may also lead to multiple solutions while solving for the inverse kinematics in robotic manipulators (Hernandez, 2008).

Moreover, the analytical model derived from the geometrical method, Screw theory or numerical techniques involves approximations and needs more computational power (Köker, 2013). Artificial neural network (ANN) based approaches have been implemented in well-established robots like the PUMA or SCARA whose inverse kinematic equations are well known and can be derived easily (Alp *et al.*, 2007). However, the data set obtained from the derived inverse kinematic equations may contain mapping error because of non-linear mapping and approximations, leading to inaccuracies in the predicted inverse kinematic solutions. This aspect emphasises the need for an alternative approach that is independent of making the data set from the inverse kinematic equations to generate error free data set. A monocular camera that is initially calibrated can be used to determine the instantaneous six-dimensional pose of a moving frame by using fiducial markers from the ArUco library (Chiddarwar and Babu, 2010). Therefore, a machine learning (ML) based method is proposed in this paper to resolve the inverse kinematics problem most effectively by taking the data set to train the network experimentally. Various regression tools are demonstrated to predict the joint variables most effectively. A multi-layer perceptron (MLP) is also proposed to determine the inverse kinematic solutions for the 3-PPSS PM.

This paper is organised as follows. The architecture of the 3-PPSS PM is explained in Section 2, followed by the explanation of its kinematic modelling using the DH approach in Section 3. The ML method for determining the inverse kinematics is described in Section 4. The results obtained by the proposed methods, the conclusions derived along with the future scope are explained in Sections 5 and 6, respectively.

2. Architecture of 3-PPSS PM

The 3-PPSS PM comprises of three legs, with each having two active prismatic joints and two passive spherical joints mounted in series. The three legs are connected to the equilateral triangle shaped mobile platform by the spherical joints, as shown in the computer aided design (CAD) model depicted in Figure 1. The spherical joints on the horizontal slider and the spherical joint on the triangular platform are connected via a rigid link called as the coupler. The global frame is fixed at (O_2) with its y-axis aligned along the direction of the horizontal slider, z-axis aligned along the axis of the vertical slider and the x-axis normal to both horizontal and vertical slider.

The PPSS arrangement provided for each leg has the advantage of taking more payloads compared to the Delta robot (Dehghani *et al.*, 2014) or 6-RUS (Bonev and Ryu, 2001) PM having revolute joints in them. The six active prismatic joints are actuated individually using lead screws coupled to stepper motors to enable precise motions for each slider. Let X_k and X_{k+1} be the two consecutive prismatic joint variables for one leg. According to the Modified Grübler–Kutzbach criterion,

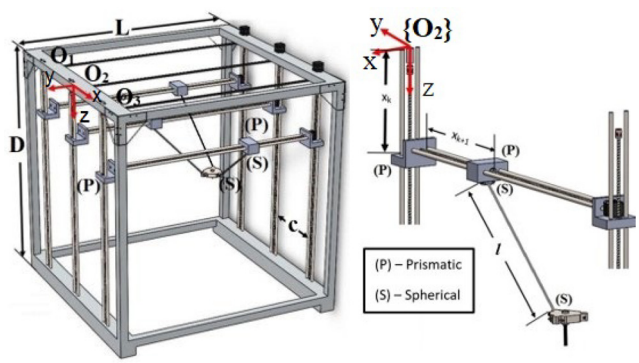
$$DoF = 6(n - g - 1) + s + \sum_{j=1}^{3}\sum_{i=1}^{3}(f_i - f_{id})_j \quad (1)$$

where "n" is the number of elements, "g" is the number of joints, "f_i" is the DoF of the *i-th* joint, "f_{id}" is the identical DoF of the *i-th* joint, "s" is the number of passive joints and "j" refers to the number of legs. For the 3-PPSS manipulator, the total number of elements and joints are 11 and 12, respectively and the number of passive joints is six. The DoF for the individual prismatic and spherical joints are one and three each. The values of f_i and f_{id} for one leg of the manipulator is listed in Table 1. As a result from equation (1), the end effector platform of the 3-PPSS PM will have a DoF of six.

3. Kinematic modelling of the 3-PPSS parallel manipulators using the Denavit–Hartenberg approach

Robot kinematics describes the mathematical relationship between the end effector and joints with respect to a frame regardless of the forces and torques acting on it. Forward kinematics involves mapping from a known set of joint variables

Figure 1 CAD model for the 3-PPSS PM with PPSS arrangement for one leg



to a pose of the moving platform. As the number of loop closure equations in the parallel mechanism increases, the difficulty of solving the forward kinematics increases (Wang *et al.*, 2009). In this paper, the DH method is exploited to obtain the kinematic relations for the 3-PPSS manipulator. The DH convention is applied individually to each leg and the legs are coupled at the mobile platform by considering suitable holonomic constraints. The coupling of the three legs accounts for the closed configuration of the PM. The frames assigned according to the DH convention for all three legs are the same because of its similar configurations, which is shown in Figure 2. Let (O_1), (O_2) and (O_3) be the zeroth frame for the three legs, respectively. In the frame assignment, according to the DH convention, the spherical joints are modelled as three revolute joints moving about the same origin.

Let *"l"* be the length of the coupler link, *"c"* be the perpendicular distance between two consecutive sliders, *"b"* be the distance from the centroid of the equilateral triangle-shaped mobile platform to its corner, *"n"* be the distance from the end effector tip to the centre of platform measured along the *Z*-axis and *"K"* be the length of the mobile platform. The pose of eighth frame (8) with respect to the zeroth frame (O_i) for each leg are determined using the DH parameters listed in Table 2. The set of link coordinates assigned is transformed from the *i-th* coordinate frame to the *i-1-th* frame using the standard homogenous DH transformation matrix (Parhi *et al.*, 2013). The final transformation matrix ${}_{(Oi)}T^8$ is determined from the individual transformation matrices by substituting the DH parameters into equation (2):

$$ {}_{O_i}T^8 = {}_{O_i}T^1 * {}_1T^2 * {}_2T^3 * {}_3T^4 * {}_4T^5 * {}_5T^6 * {}_6T^7 * {}_7T^8, i = 1, 2, 3 \tag{2} $$

Another frame (9) is assigned at the tip of the end effector to obtain its pose from the global frame. The pose of frame(9) along the middle leg is derived from the following equation,

$$ \left({}_{O_2}T^9\right)_{middle} = \left({}_{O_2}T^8\right)_{middle} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & n \\ 0 & 0 & 1 & b \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3} $$

Similarly, the pose of the end effector tip from the global frame (O_2) along the other two legs are given in equations (4) and (5):

$$ \left({}_{O_2}T^9\right)_{left} = {}_{O_2}T^{O_1} * \left({}_{O_1}T^8\right)_{left} * \left({}_8T^9\right)_{\text{left}} = \begin{bmatrix} 1 & 0 & 0 & -c \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \left({}_{O_1}T^8\right)_{left} * \begin{bmatrix} 1 & 0 & 0 & 0.866b \\ 0 & 1 & 0 & n \\ 0 & 0 & 1 & -0.5b \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4} $$

$$ \left({}_{O_2}T^9\right)_{right} = {}_{O_2}T^{O_3} * \left({}_{O_3}T^8\right)_{right} * \left({}_8T^9\right)_{right} = \begin{bmatrix} 1 & 0 & 0 & c \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \left({}_{O_1}T^8\right)_{right} * \begin{bmatrix} 1 & 0 & 0 & -0.866b \\ 0 & 1 & 0 & n \\ 0 & 0 & 1 & -0.5b \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5} $$

As equations (3)-(5) represent the same point from the global frame (O_2), the three equations are equated to the desired pose matrix to obtain the following relation:

$$ \left({}_{O_2}T^9\right)_{left} = \left({}_{O_2}T^9\right)_{middle} = \left({}_{O_2}T^9\right)_{right} = \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6} $$

Equating the orientation and position vectors in equation (6) yields 36 equations in total. However, the linear dependency property of the orientation matrix reduces the number of independent equations to 18. The DH model of the 3-PPSS manipulator has 24 joint variables in total. Therefore, to get the exact solution for the inverse kinematics, the number of equations should be made equal to the number of unknowns. The additional six equations are obtained from the holonomic constraints present within the geometry expressed by the following relations,

$$ \left[(S_{lowX})_i - (S_{lowX})_j\right]^2 + \left[(S_{lowY})_i - (S_{lowY})_j\right]^2 + \left[(S_{lowZ})_i - (S_{lowZ})_j\right]^2 = K^2 \tag{7} $$

Table 1 DoF for each leg in the 3-PPSS manipulator

Type of joint	No. of joint	DoF for *i-th* joint (f_i)	Identical DoF for *i-th* joint (f_{id})	f_i-f_{id}
Vertical slider	1	1	0	1
Horizontal slider	1	1	0	1
Spherical joint	2	3	1	2
				$\sum_{i=1}^{3}(f_i - f_{id})_j = 1 + 1 + 2 = 4$
				$\sum_{j=1}^{3}\sum_{i=1}^{3}(f_i - f_{id})_j = 4 + 4 + 4 = 12$

Figure 2 DH frame assignment for one leg common to all three legs

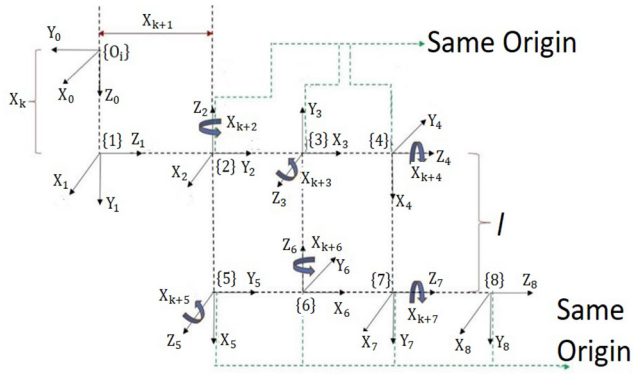


Table 2 DH Parameters for the manipulator where "$k = 0$" for the first leg, "$k = 9$" for middle leg and "$k = 18$" for the third leg

Link	θ (degree)	d	a	α
1	0	X_k	0	90
2	0	X_{k+1}	0	90
3	$X_{k+2} + 90$	0	0	90
4	$X_{k+3} - 90$	0	0	−90
5	X_{k+4}	0	l	90
6	$X_{k+5} + 90$	0	0	−90
7	$X_{k+6} - 90$	0	0	−90
8	X_{k+7}	0	0	0

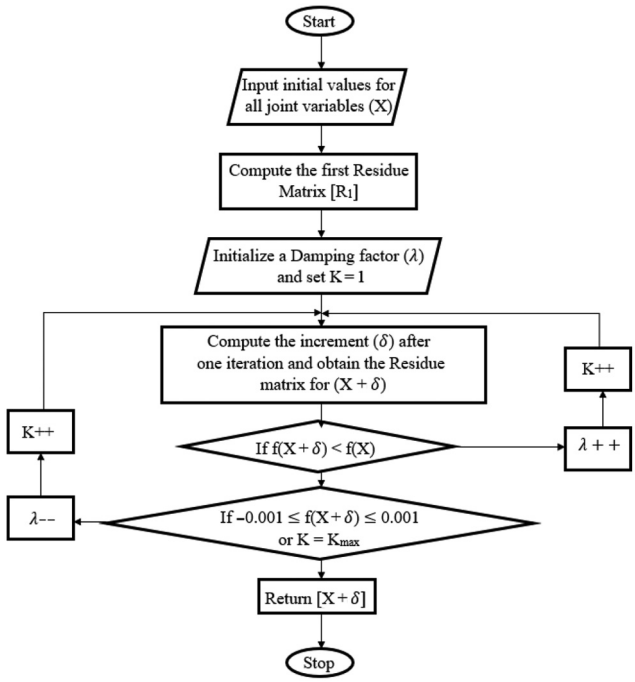
$$\left[(S_{lowX})_i - (S_{upX})_j\right]^2 + \left[(S_{lowY})_i - (S_{upY})_j\right]^2 + \left[(S_{lowZ})_i - (S_{upZ})_j\right]^2 = l^2, \quad i, j = 1, 2, 3; i \neq j \tag{8}$$

where S_{low} and S_{up} represents the position coordinates for the lower and upper spherical joints, respectively. An iterative Levenberg–Marquardt algorithm (LMA) (Gavin, 2019) is used to solve the set of non-linear equations. LMA is a curve fitting method that adaptively varies the parameter updates between Gradient descent method and the Gauss-Newton method. The flowchart for the LMA algorithm is given in Figure 3. The set of equations are expressed of the form "f(X)=0" where "X" represents all the joint variables, $X = (X_1.X_2\ X_3\ \ldots\ X_{24})^T$. LMA uses a damping factor (λ) to control the incremental step size after every iteration. As the value of λ is increased, the step size reduces accordingly. The algorithm begins by initiating the set of values for X. Residues are the values of each function obtained on substituting the joint variables in each iteration. The objective of this algorithm is to minimise this residue and finally approach to zero. The algorithm returns the values of "X" during the last iteration when the residue reaches within a tolerance limit set between ± 0.001 in this work.

4. Inverse kinematic solution using machine learning approach

ML problems can be categorised into three types, namely, regression, classification and clustering. As the problem here is to predict the slider positions corresponding to the non-linear input poses, this becomes a regression problem. As mentioned in the introduction section, the determination of the inverse kinematic solutions by solving equations for complex geometries is a very cumbersome task and is computationally very expensive. Hence, an intelligent method is adopted to get solutions for the inverse kinematics using ML algorithms. In this section, the active slider positions are predicted for the 3-PPSS PM using multiple linear regression (MLR), multivariate polynomial regression (MVPR), support vector regression (SVR), decision tree (DT) regression and random forest (RF) regression.

Figure 3 Flowchart for the LMA

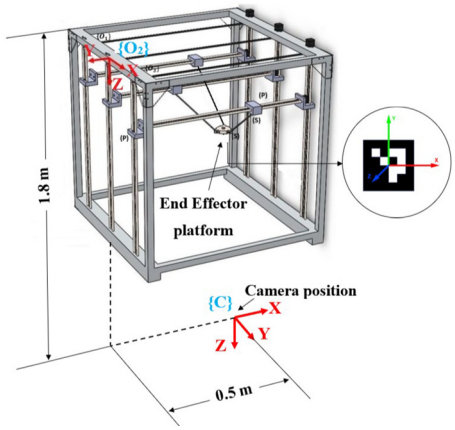


A MLP model is also proposed to predict the instantaneous slider positions of the manipulator. The ML algorithms are written, trained and tested in the PYTHON environment. The performance of the various prediction methods is quantified using root mean square error (RMSE) and R^2 value (R^2). The poses predicted using the ML algorithms are compared with the experimental results obtained from 10 random poses of the platform.

4.1 End effector platform pose determination using ArUco markers

In this paper, the learning task is carried out without the mathematical model by using a camera-based system to track the pose of the mobile platform. The ArUco library (Garrido-Jurado *et al.*, 2014) is used to generate a set of fiducial markers that are used to simplify the estimation of the 6-DoF marker poses. The ArUco markers based on open source computer vision library (OpenCV) are square markers comprising of a black border with an inner black and white pattern. The pose of the ArUco marker pasted on the bottom surface of the mobile platform is recorded with respect to the camera frame (C), as shown in Figure 4. The ArUco marker used in this study is also

Figure 4 Frame representation for the Global (O2) and camera (C) frames



depicted in the figure. The algorithm is provided with the calibration parameters of the camera to describe its optical properties. The camera calibration and generation of the ArUco marker (Neumayr *et al.*, 2011) is a one-time process and no physical alteration should be made to the camera position or the relative position of the markers after fixing. The orientation of the platform (ArUco marker) with respect to the camera frame (C) is expressed in terms of quaternions (Perumal, 2011). The positional coordinates and the quaternion values of the ArUco marker frame are continuously tracked by the calibrated camera, as shown in the screenshot in Figure 5. A sample of the positional and quaternion readings of the platform is shown in Table 3.

4.2 Experimental data set generation to train the network

The data set is generated by recording the end effector platform pose using the ArUco library and its corresponding slider positions measured using the ultrasonic sensors. The platform poses determined in the last section is in terms of the quaternion components measured with respect to the (C) frame. The poses are transformed from the (C) frame to the (O_2) frame to compare the results with that of the analytical approach explained in Section 3. A sample of the experimental data set is given in Table 3 displaying the ArUco readings and the ultrasonic sensor readings. The readings show that the data set is non-linear and accurate up to three decimal places. The pose of the marker from the camera (C) frame is determined from the following matrix (Gopinath and Prince, 2019):

$$ {}_{C}T^{9} = \begin{bmatrix} e_0^2+e_1^2-e_2^2-e_3^2 & 2(e_1e_2+e_0e_3) & 2(e_1e_3-e_0e_2) & t_x \\ 2(e_1e_2-e_0e_3) & e_0^2-e_1^2+e_2^2-e_3^2 & 2(e_2e_3+e_0e_1) & t_y \\ 2(e_1e_3+e_0e_2) & 2(e_2e_3+e_0e_1) & e_0^2-e_1^2-e_2^2+e_3^2 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9) $$

with e_0, e_1, e_2, e_3 representing the quaternion components and t_x, t_y, t_z representing the translational components of the marker. The pose of the end effector expressed in equation (9) is transformed from the camera frame (C) to the global frame (O_2) using equation (10):

$$ {}_{O_2}T^{9} = {}_{O_2}T^{c} \times {}_{C}T^{9} \quad (10) $$

where:

$$ {}_{O_2}T^{c} = R_{(Z,90^o)}\, T_{(0,0,-1.8)}\, T_{(-0.5,0,0)} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -0.5 \\ 0 & 0 & 1 & -1.8 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

The Euler angles and position coordinates of the end effector platform with respect to the (O_2) frame is computed by equating the ${}_{O2}T^{O_2 9}$ matrix in equation (10) with the standard Euler angle transformation matrix (Merlet, 2000). By doing this, the final data set consisting of positional coordinates and Euler angles of the end effector platform with the corresponding slider position is established and further used to train the ML networks. The data set generated is split into training and test data sets, respectively. The ML algorithms are trained using the training data set and evaluated against the test set. The flow diagram in Figure 6 describes the important steps involved in the ML-based computation for inverse kinematics.

4.3 Machine learning algorithms

A comparative study of various ML algorithms for the determination of the inverse kinematics is explained in this section. The algorithms such as MLR, MVPR, SVR, DT regression, RF regression and multi-layer perceptron models are analysed. Correlations between the input pose variables and the output slider positions are defined using the above stated algorithms. All the prediction models are developed in PYTHON and their respective performances are analysed by changing their distinct parameters to choose the most accurate prediction algorithm.

4.3.1 Multiple linear regression and multi-variate polynomial regression.

MLR is the simplest regression tool that uses several input variables to predict the response variable. MLR establishes a linear relationship between every slider position and end effector pose variables. The general regression equation for one dependent and multiple independent variables are defined in equation (11):

$$ y_i = b_{0i} + b_{1i} * x_{1i} + \ldots + b_{6i} * x_{6i} \quad (11) $$

where "y_i" is the estimated slider position of the *i*-th slider, x_1 to x_6 are the six independent pose variables of the mobile platform and b_0 to b_6 are the regression coefficients. The regression coefficients for each input-output relation are determined by the algorithm after training the network. This regression tool is most suitable for data sets having linear characteristics.

In MVPR, the relationship between the input and output variable is modelled as an *n*-th order polynomial equation. This model is an advancement of the MLR algorithm and is used when the response variable shows a non-linear behaviour. The general expression for a second-order MVPR with one dependent variable and two independent variables is:

Figure 5 Screenshot showing the experimental measurement of positional and quaternion values of the end effector platform using ArUco

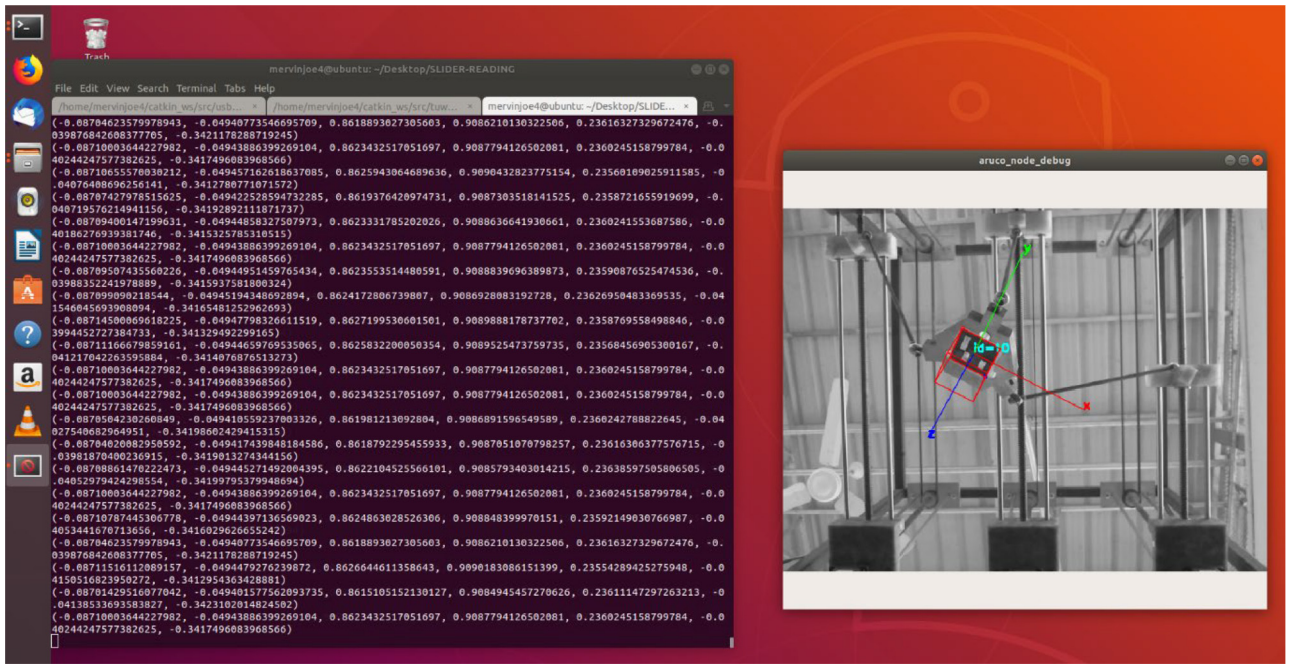
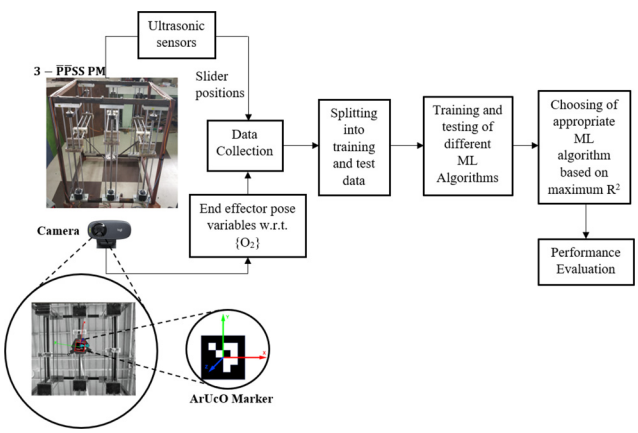


Table 3 Sample of the experimental data set displaying the ArUco readings and ultrasonic sensor readings

Inputs							Outputs (Slider positions) (m)					
t_x	t_y	t_z	e_0	e_1	e_2	e_3	1	2	3	4	5	6
0.013	−0.27	0.940	0.696	0.705	−0.089	−0.094	0.03	0.05	0.03	0.84	0.84	0.84
0.013	−0.25	0.944	0.688	0.719	−0.067	−0.068	0.06	0.05	0.03	0.84	0.84	0.84
0.008	−0.24	0.940	0.632	0.751	−0.133	−0.133	0.15	0.34	0.03	0.84	0.84	0.84
0.004	−0.24	0.975	0.707	0.691	−0.096	−0.110	0.05	0.32	0.06	0.81	0.78	0.81
0.020	0.052	1.076	0.647	0.730	0.215	0.016	0.20	0.65	0.20	0.81	0.63	0.73
0.010	0.173	1.171	0.540	0.837	−0.076	0.008	0.6	0.59	0.33	0.66	0.57	0.64
−0.01	−0.06	1.217	0.701	0.7081	0.0452	0.0670	0.23	0.35	0.22	0.6	0.54	0.61
−0.02	−0.22	1.4078	0.637	0.6896	0.2412	0.2437	0.07	0.43	0.13	0.37	0.42	0.43
−0.04	−0.03	1.308	0.742	0.666	−0.04	0.040	0.13	0.49	0.31	0.47	0.42	0.56
0.02	0.11	1.185	0.67	0.731	−0.096	0.002	0.41	0.64	0.39	0.62	0.54	0.61

Figure 6 Process flow diagram for inverse kinematics using ML



$$y_i = b_{0i} + b_{1i} * x_{1i} + b_{2i} * x_{2i} + b_{3i} * {x_{1i}}^2 + b_{4i} * {x_{2i}}^2 + b_{5i} * x_{1i} * x_{2i} \quad (12)$$

where y_i is the estimated slider position of the i-th slider; x_1, x_2 are the independent variables and b_0, b_1,b_2, etc. are the regression coefficients determined by the algorithm after the training.

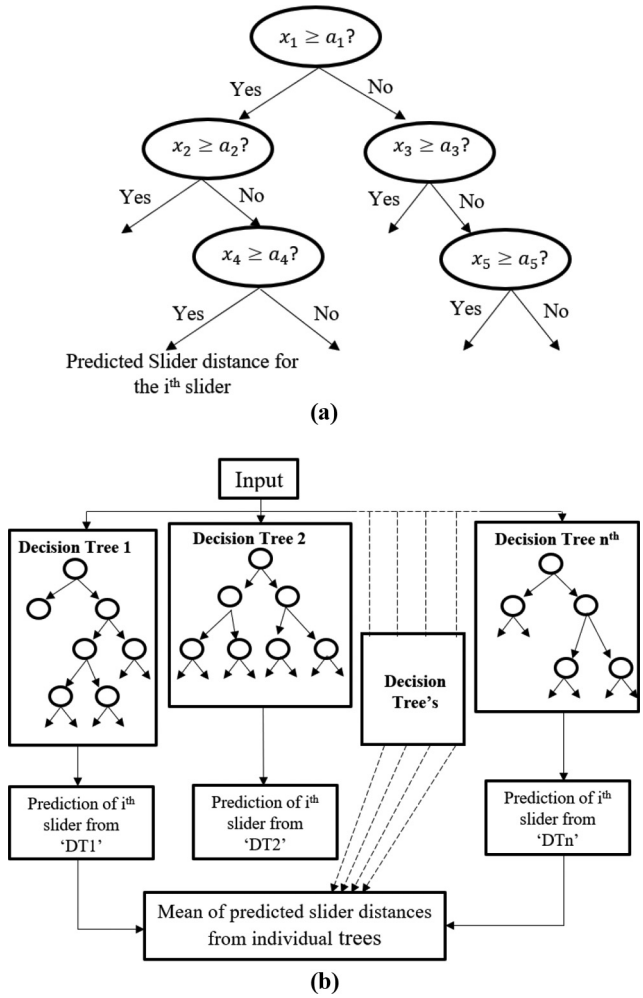
4.3.2 Support vector regression.
SVR is a generalisation of support vector machine that supports both linear and non-linear data giving numerical values as output instead of a classification problem. SVR basically performs linear regression at a higher-dimensional space. This implies that SVR is superior over MLR, as it considers data within a tolerance limit of ε. As the data set generated containing the pose variables and corresponding slider

distances are non-linear, a kernel function is used to map a lower-dimensional data into a higher dimensional data. This function converts the non-linear problem into a linear problem. Appropriate kernel functions are chosen according to the type of data being analysed. The most widely used kernel functions are linear, poly and radial basis function (RBF). Poly and RBF are useful for a non-linear set of data. Several parameters need to be tuned for using the SVR algorithm. One of them is the cost parameter C, taken as unity. Higher values of C tend to give predicted values having higher amplitudes than the actual series. The value of ε in the loss function is set as 0.01, which gives the best performance (Gopinath and Prince, 2019). In this analysis, the RBF kernel function has been used, as the literature claim it to be suitable for predicting a non-linear set of data and is relatively easier to tune. After applying the kernel function, a hyperplane can be fitted to predict the slider distances with much higher accuracy compared to the MLR and MVPR algorithms.

4.3.3 Decision tree and random Forest regression.
DT regression is a non-parametric approach and the algorithm uses a recursive partitioning algorithm consisting of nodes, branches and leaves. This algorithm trains the model similar to the structure of a tree to predict meaningful continuous outputs. Each node in a DT has only one parent node and a binary split. A schematic representation of the DT algorithm is shown in Figure 7(a). This regression algorithm is easy to understand, implement and does not require expensive modelling. The two main steps involved in the DT algorithm are: to divide the training data set space into several distinct and non-overlapping regions; the prediction is the same for an observation that falls within the same region, which will be the mean of the training observations in that region. For the 3-PPSS PM, there are six slider positions to be predicted from the six end effector inputs. This implies that six DTs will be generated simultaneously to predict each slider position corresponding to an end-effector pose, and therefore, cannot be illustrated on a graph. According to this algorithm, when an end effector pose is input into the network, the conditions for every individual region is checked. Each subgroup will have a set of slider length readings that correspond to the i-th slider output. The average of all values within that subgroup will give the slider distance for the i-th slider. A similar procedure is repeated simultaneously for all six sliders while executing the programme for the end-effector pose.

RF regression uses an ensemble learning algorithm having several DTs to improve its predictive accuracy and minimise the chances of overfitting (Mangalathu and Jeon, 2019). According to this algorithm, the mean value of the slider distances from each tree is the predicted output. The DTs are sampled randomly as subsets of the training data set and involve training each DT on different data samples. The main objective behind this method is to combine multiple DTs in determining the final slider distance for the i-th slider rather than relying on individual DTs. RF regression can be regarded as an improvised model of the DT regression requiring an additional parameter, which is the total number of trees. A schematic representation of the RF regression algorithm is depicted in Figure 7(b). The output of the RF prediction is obtained by the following relation,

Figure 7 Schematic illustration of (a) DT algorithm; (b) RF algorithm



$$y_i = \frac{1}{n_t}\sum_{1}^{n_t} f(x_i) \tag{13}$$

where y_i denotes the predicted slider length for the i-th slider using the RF algorithm, x_i represents the individual prediction for the i-th slider using the DT algorithm and n_t refers to the number of trees used in the study.

4.3.4 Multi-layer perceptron.
A MLP is a feed-forward type of ANN comprising of an input layer, at least one hidden layer and an output layer. The back propagation algorithm categorised as a supervised learning technique is used in this model to train the network. Training of a neural network can be expressed as a non-linear mapping between any given input and output data set. Literature indicates that for non-linear regression problems with non-linear data sets, the Multi-Layer Perceptron models can be used. Therefore, a feed-forward multi-layer neural network architecture with six inputs and six outputs is designed to find the inverse kinematic solutions for the 3-PPSS manipulator, as shown in Figure 8. During training, the pose variables are given as input to the network and the corresponding outputs, i.e. the six slider distances from the global frame, are predicted. As

there is no standard procedure to decide the number of hidden layers, the number of neurons and the type of activation function, the MLP model is tested for different combinations to determine the optimum condition. Aiming at finding the best MLP model, several feed-forward networks having one, two and three hidden layers with varying number of neurons and activation functions is considered in this study.

Parameters such as the number of hidden layers, number of neurons in each hidden layer, type of activation functions used at hidden layers and output layers are varied during training of the network. The linear, sigmoid, tanh and rectified linear unit (ReLU) are used as activation functions during the training. The learning optimisers such as adaptive moment estimation (Adam) and stochastic gradient descent (SGD) methods (Haykin, 2014) are used for optimising the weights during the training. By carefully selecting the parameters, the prediction accuracy is significantly improved. For this reason, the prediction accuracy of MLPs with different number of hidden layers, neurons and activation functions is explored in this paper. The maximum number of epochs is fixed as 1,000, as

Figure 8 The neural network topology used in this study

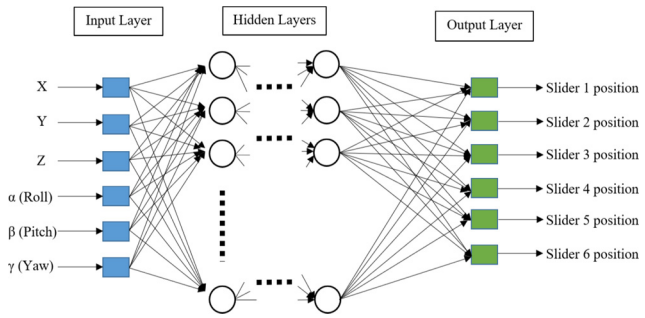
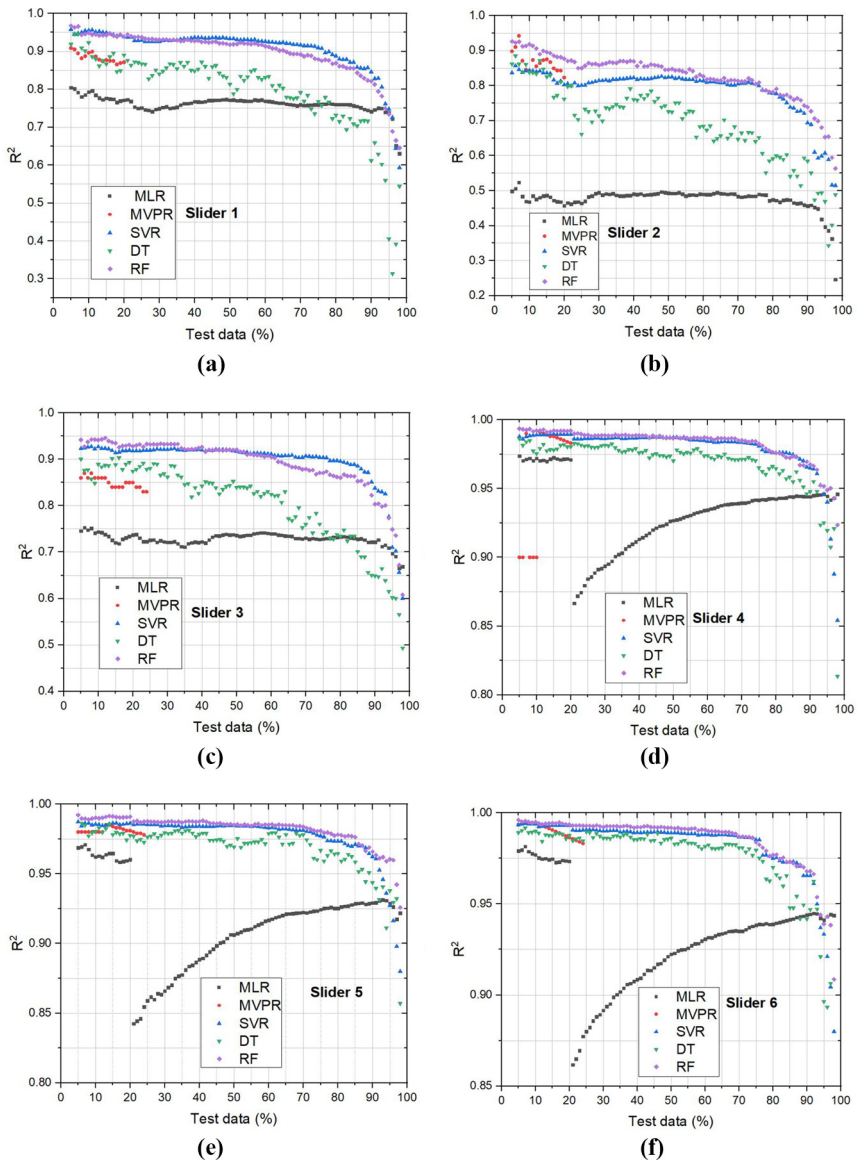


Figure 9 Comparison of training performance vs test data percentage for (a) Slider 1; (b) Slider 2; (c) Slider 3; (d) Slider 4; (e) Slider 5; (f) Slider 6



the accuracy (R^2 and RMSE) remains unchanged for epochs beyond this value.

5. Results and discussion

The performance of the ML algorithms in solving the inverse kinematics problem is validated in this section by comparing simulation and experimental results. The experiments were conducted for 10 random end-effector poses of the 3-PPSS manipulator to choose the best ML model based on its accuracy of results. The simulations were made on a personal system with an Intel CORE i7 processor, 16 GB of random access memory (RAM), with Linux kernel version, MATLAB R2019b and algorithms coded using PYTHON3.7. According to the ML approach explained in this paper, the experimental data set generated is used to train the models to predict the inverse kinematic solutions. A total of 4,500 experimental data sets were taken prior to training. The data set is then divided into the training set and test set in which the test set is used to evaluate the performance of the training set. However, depending upon the random samples in the training set, the model assessed only on a specific training set can either underestimate or overestimate the model. To avoid this condition, the authors' studied the variability in the R^2 value for different ML models with multiple samples of data and test sets. The variations in the R^2 value for all six sliders at different test data percentages and ML algorithms are shown in Figures 9(a)-9(f). The results indicate that the ML algorithms, follow a similar trend whose R^2 value is maximum when the training data is 95% (test set = 5%) and its value gradually decreases as the test data percentage increases, except for the MLR algorithm. The exceptional behaviour of the MLR algorithm happens, as the non-linear mapping between the input and output is impossible by this regression technique; MVPR model does not give significant values for test data percentage beyond 20. This is because of insufficient training data to model the regression equation that fits correctly. Another major issue with MVPR is its multicollinearity property (Sinha, 2019) for higher degree terms in the polynomial equation. This condition restricts the model from the proper estimation of the regression curve; the RF model has the best predictive capability with highest R^2 and lowest RMSE in comparison to MLR, MVPR, SVR and DT models.

The variations in the R^2 value for one, two, three and four-degree polynomials are shown in Figure 10(a). It is observed that for degrees above four, the model gives unrealistic values as output because of multicollinearity property. Therefore, for this reason, the degree of a polynomial function is fixed at four during this simulation study. Similarly, the variations in the R^2 value for different number of trees in the RF model were studied to determine the optimum number of trees needed within the model. Figure 10(b) indicates that the optimum number of trees for the RF model should be 40 corresponding to the maximum R^2 value, beyond which this value remains nearly constant.

An MLP model is also proposed to predict the inverse kinematic solutions for the 3-PPSS manipulator. A summary of the network performances for the MLP models with different network structures after reaching convergence are listed in Table 4. The MLP models were tested for two different learning optimisation algorithms, namely, Adam and SGD, respectively. The parameters for the Adam and SGD algorithms were set to their respective default values during this study. After analysing the results, it was certain that Adam optimiser requires lesser training time compared to the SGD optimiser for comparable R^2 values. Hence, an inference can be made for the MLP inverse kinematics model that Adam optimiser is superior over the SGD optimising algorithm. After this, three different activation functions were examined in the MLP neural network for different number of hidden layers and a variable number of neurons in each hidden layer. As is evident from Table 4, the ReLU activation function having two hidden layers with 45 neurons in each hidden layer gave the best result with an R^2 value of 0.882 (min. of all six sliders). To obtain a better idea of performance for the various MLP networks, the RMSE associated with all the slider positions, are also listed in Table 4.

It is seen from the results that the RF model performs superior over the other ML algorithms. The comparative bar graphs in Figures 11 and 12 displays the R^2 value and the corresponding RMSE for the six ML algorithms considered in this study. This R^2 and RMSE value correspond to the least of all six sliders to indicate the accuracy of prediction for the inverse kinematic problem dealt with in this paper. For example, in Figure 9(b), it is evident that the second slider has the least R^2 value (0.94611) compared to that of the other five sliders taking the case of the RF algorithm for 95% training data. Hence, this value is shown in the bar graph in Figure 11, indicating its overall accuracy of

Figure 10 Variations in R^2 value (a) with degree of polynomial in MVPR; (b) with number of trees in RF algorithm

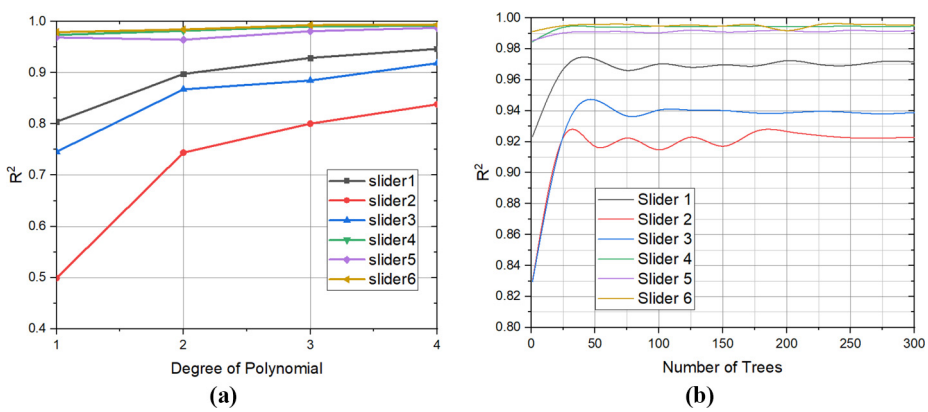


Table 4 Statistical analysis of the MLP model to find the inverse kinematic solutions for the 3- $\overline{PP}$SS manipulator

				Slider 1		Slider 2		Slider 3		Slider 4		Slider 5		Slider 6		
Optimiser	No. of hidden layers	Activation function	No. of neurons	R^2	RMSE (m)	R^2	RMSE (m)	R^2	RMSE (m)	R^2	RMSE (m)	R^2	RMSE (m)	R^2	*RMSE (m)*	*Training time (s)*
Adam	1	Sigmoid	175	0.9210	0.0486	0.796	0.0731	0.873	0.0538	0.983	0.0190	0.971	0.0257	0.983	0.0166	127.57
Adam	1	Tanh	185	0.9196	0.0490	0.824	0.068	0.908	0.0457	0.984	0.0188	0.982	0.0205	0.986	0.0172	119.38
Adam	1	ReLU	75	0.9093	0.052	0.8429	0.0646	0.927	0.0408	0.982	0.0197	0.980	0.0213	0.986	0.0170	84.178
Adam	2	Sigmoid	45, 45	0.937	0.0431	0.8397	0.0649	0.901	0.0476	0.984	0.0189	0.98	0.0212	0.987	0.0163	283.03
Adam	2	Tanh	35, 35	0.917	0.0496	0.843	0.0641	0.914	0.0442	0.985	0.0183	0.977	0.0230	0.986	0.0170	142.64
Adam	2	ReLU	45, 45	0.936	0.0429	0.882	0.0584	0.946	0.0415	0.980	0.0185	0.980	0.0212	0.980	*0.0165*	*98.08*
Adam	3	Sigmoid	25, 25, 25	0.938	0.0429	0.844	0.0640	0.9316	0.0396	0.986	0.0176	0.984	0.0192	0.987	0.0162	255.876
Adam	3	Tanh	25, 25, 25	0.917	0.0498	0.853	0.623	0.916	0.0411	0.980	0.0181	0.973	0.0209	0.980	0.0163	378.109
Adam	3	ReLU	35, 35, 35	0.9104	0.0517	0.8541	0.0619	0.9296	0.0422	0.989	0.0152	0.991	0.0143	0.985	0.0156	195.766
SGD	1	ReLU	15	0.902	0.510	0.781	0.759	0.872	0.541	0.973	0.245	0.969	0.269	0.981	0.198	1644.76
SGD	1	Sigmoid	25	0.879	0.599	0.724	0.852	0.821	0.640	0.977	0.228	0.966	0.280	0.980	0.208	2223.0
SGD	2	Tanh	15, 15	0.867	0.628	0.758	0.797	0.829	0.626	0.973	0.247	0.957	0.315	0.969	0.259	3276.1

prediction. A similar procedure was applied to the remaining algorithms also. The total computational time needed for the different models is shown in Figure 13. Experiments were conducted to compare the results of the inverse kinematic solutions obtained from the RF ML model and the mathematical model (DH model). The experimental procedure is described as follows. First, a set of 10 distinct slider positions for the 3-PPSS manipulator was randomly selected and the sliders in the prototype were brought to their respective positions in order. The pose of the end effector was then recorded using the ArUco marker for the 10 distinct slider positions. These 10 poses are fed into the RF network and the mathematical model to compute the corresponding slider positions by the two methods. Table 6 shows the comparison of the inverse kinematic solutions for the 3-PPSS manipulator computed by ML and DH with the experimental slider positions for 10 different poses given in Table 5. The 10 poses were chosen at random from various points within the workspace to verify the results by the two approaches

Figure 11 R^2 values for different ML algorithms

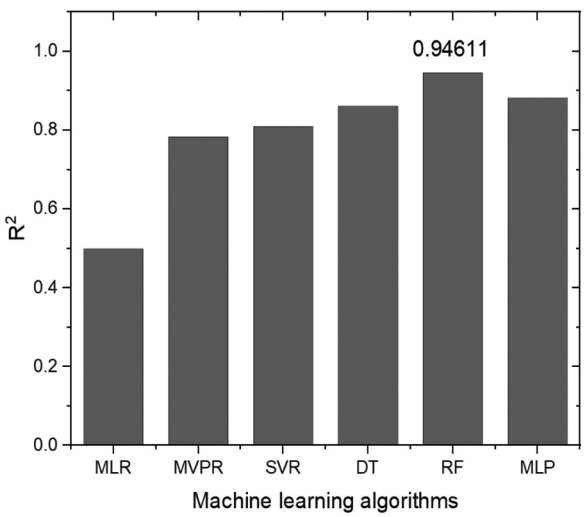


Figure 12 RMSE values for different ML algorithms

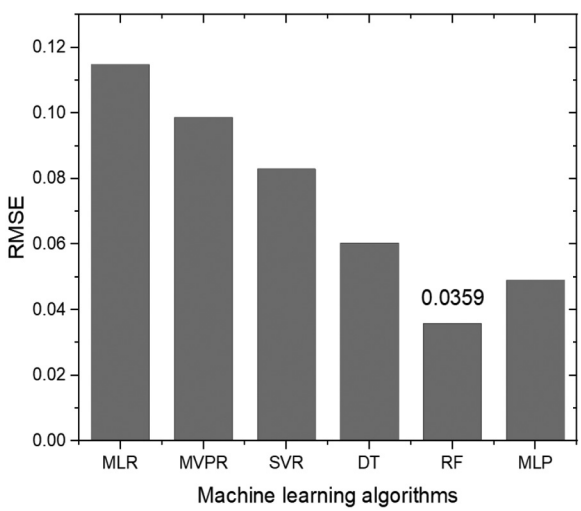
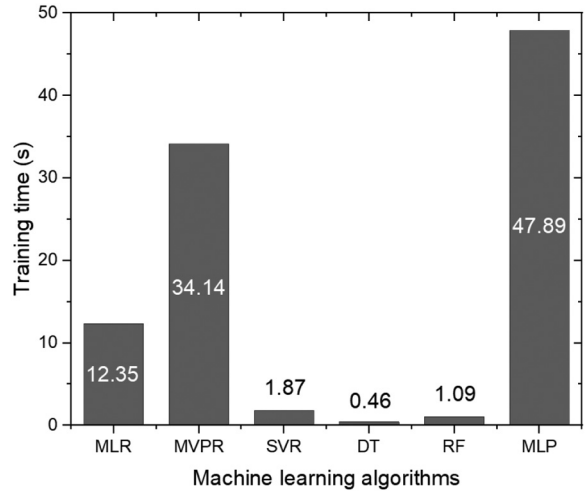


Figure 13 Training time for different ML algorithms



with the experimental results. Figures 14(a) and 14(b) shows the percentage of positional error variations for all the six sliders obtained by the two approaches compared to the experimental readings. It is evident that the RF model gives much higher accuracy in results compared to that from the mathematical model. This is because the data set generated to train the network was taken directly from the prototype, which includes all physical and geometrical characteristics of the manipulator. However, in the mathematical model, the joints and links are modelled assuming its ideal conditions. These approximations in the mathematical model can account for slight deviations from the actual result. Moreover, the computation time needed to predict the inverse kinematic solutions by the ML approach was lower than that from the mathematical method, as is clear from Table 7. Therefore, the ML approach for the inverse kinematics makes it more suitable for real-time control applications.

6. Conclusion

A novel 3-PPSS PM with six active prismatic joints and six passive spherical joints, which offers large workspace and easy mobility to the end effector platform is proposed in this work. The inverse kinematic solutions for the mechanism were determined by considering suitable holonomic constraints within the manipulator using the analytical method. The proposed manipulator is ideal for

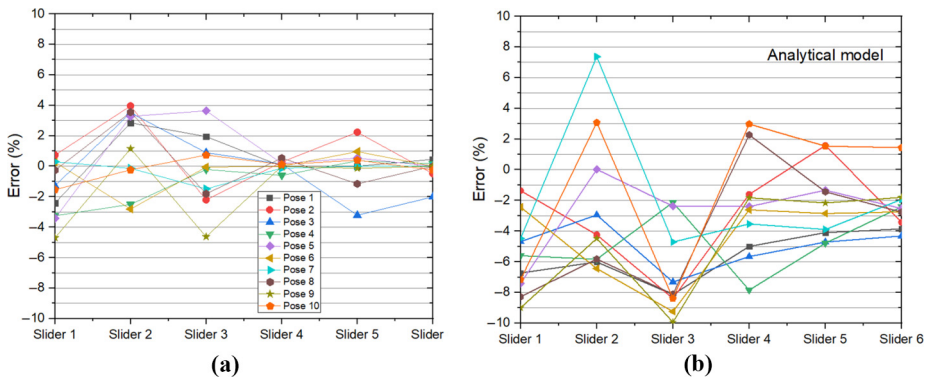
Table 5 Details of 10 random poses taken for the experimental validation

Pose	(X m, Y m, Z m, $\alpha°$, $\beta°$, $\gamma°$)
1	(0.38, −0.46, 0.78, −6.5, −10.6, −8.9)
2	(0.17, −0.48, 0.82, −17.6, −18.6, −2.3)
3	(0.08, −0.14, 0.62, 23.2, 10.5, −10.8)
4	(−0.03, −0.72, 0.70, 31.1, −15.2, 2.5)
5	(−0.07, −0.36, 0.98, 0.1, 22.2, −8.2)
6	(−0.24, −0.55, 0.44, 12.1, 4.2, 0.1)
7	(0.02, −0.14, 0.65, 41.2, −11.2, 1.2)
8	(−0.21, −0.25, 1.09, −33.2, 12.2, 5.2)
9	(−0.08, −0.63, 0.75, 10.2, −38.2, 6.3)
10	(0.12, −0.55, 0.78, 15.2, −31.3, −4.5)

Table 6 Comparison of inverse kinematic solutions computed from RF and DH with experimental results

	Slider 1 (m)			Slider 2 (m)			Slider 3 (m)			Slider 4 (m)			Slider 5 (m)			Slider 6 (m)		
Pose	Exp.	RF	DH	Exp.	RF	DH	Exp.	RF	DH	Exp.	RF	DH	Exp.	RF	DH	Exp.	RF	DH
1	0.14	0.1433	0.1634	0.65	0.6315	0.6891	0.045	0.0441	0.0576	0.6	0.6	0.6299	0.51	0.51	0.5309	0.55	0.5475	0.5712
2	0.17	0.1687	0.1893	0.65	0.6242	0.6775	0.085	0.0868	0.0921	0.63	0.6282	0.6401	0.57	0.5572	0.5609	0.61	0.613	0.6309
3	0.2	0.2025	0.2293	0.37	0.357	0.3809	0.07	0.0693	0.0891	0.53	0.5292	0.5599	0.42	0.4335	0.4398	0.46	0.4692	0.4798
4	0.02	0.0206	0.0312	0.04	0.041	0.0423	0.24	0.2405	0.2691	0.5	0.503	0.5392	0.42	0.4185	0.4401	0.46	0.4592	0.4711
5	0.15	0.1551	0.1611	0.65	0.6287	0.6498	0.275	0.265	0.2815	0.84	0.8385	0.8601	0.84	0.8355	0.8512	0.84	0.84	0.8612
6	0.47	0.4687	0.4814	0.64	0.658	0.6812	0.25	0.2501	0.2781	0.42	0.42	0.4309	0.39	0.3862	0.4011	0.4	0.4	0.4109
7	0.59	0.5882	0.6164	0.56	0.5607	0.5187	0.47	0.477	0.4921	0.6	0.6007	0.6212	0.51	0.51	0.5298	0.58	0.58	0.5912
8	0.115	0.115	0.1291	0.5	0.4822	0.5291	0.09	0.0916	0.1099	0.84	0.8355	0.8209	0.78	0.789	0.7912	0.84	0.84	0.8632
9	0.05	0.0523	0.0610	0.65	0.6425	0.6791	0.2	0.2092	0.2198	0.6	0.6	0.6109	0.57	0.5707	0.5823	0.61	0.61	0.6209
10	0.26	0.264	0.2786	0.65	0.6515	0.6300	0.085	0.0843	0.0921	0.63	0.6292	0.6112	0.57	0.5677	0.5612	0.61	0.6115	0.6012

Figure 14 Percentage error in slider positions for (a) ML approach; (b) analytical approach



challenging applications such as assembly, manufacturing, biomedical, space technologies, rehabilitation systems, entertainment systems, high precision positioning devices, etc. An alternative methodology based on ML is also explained in this paper to determine the inverse kinematic solutions for the manipulator to overcome the computational difficulties involved with the analytical approach. An experimental data set consisting of end effector pose variables and slider positions were generated to train the various ML networks. An overview of the various ML algorithms, namely, MLR, MVPR, SVR, DT, RF and MLP were described in this paper to predict the inverse kinematic solutions. It was evident that the RF algorithm performed with higher accuracy compared to other examined training algorithms. The analytical and the experimental results also indicate that the RF algorithm needs significantly low training time without compromising the accuracy of the inverse kinematic solutions predicted.

The proposed approach can be further used for future work to perform intelligent trajectory planning for different applications. Various torque based controllers can be designed and implemented in the proposed robot to achieve precise motions for the end effector. The manipulator can be upgraded to an autonomous working machine for manufacturing operations and medical applications such as surgery/scanning owing to its high singularity free workspace and real-time control capabilities by designing an ML-based controller.

Table 7 Prediction time for RF algorithm and mathematical approach

	Prediction time (s)	
Pose	ML approach (RF)	Analytical approach
1	0.0156	62.12
2	0.0182	55.09
3	0.0147	57.08
4	0.00997	49.34
5	0.0158	59.11
6	0.0154	61.07
7	0.0197	55.23
8	0.0203	57.19
9	0.0122	55.23
10	0.0129	49.23

References

Abedinnasab, M.H., Farahmand, F., Tarvirdizadeh, B., Zohoor, H. and Gallardo-Alvarado, J. (2017), "Kinematic effects of number of legs in 6-DOF UPS parallel mechanisms", *Robotica*, Vol. 35 No. 12, pp. 2257-2277.

Alp, H., Anli, E. and Özkol, I. (2007), "Neural network algorithm for workspace analysis of a parallel mechanism", *Aircraft Engineering and Aerospace Technology*, Vol. 79 No. 1, pp. 35-44.

Bonev, I.A. and Ryu, J. (2001), "Geometrical method for computing the constant-orientation workspace of 6-PRRS parallel manipulators", *Mechanism and Machine Theory*, Vol. 36 No. 1, pp. 1-13.

Chen, Q., Zhu, S. and Zhang, X. (2015), "Improved inverse kinematics algorithm using screw theory for a Six-DOF robot manipulator", *International Journal of Advanced Robotic Systems*, Vol. 12 No. 10, pp. 1-9.

Chiddarwar, S.S. and Babu, N.R. (2010), "Engineering applications of artificial intelligence comparison of RBF and MLP neural networks to solve inverse kinematic problem for 6R serial robot by a fusion approach", *Engineering Applications of Artificial Intelligence*, Vol. 23 No. 7, pp. 1083-1092.

Dehghani, M., Ahmadi, M., Khayatian, A., Eghtesad, M. and Yazdi, M. (2014), "Vision-based calibration of a hexa parallel robot", *Industrial Robot: An International Journal*, Vol. 41 No. 3, pp. 296-310.

Furqan, M., Suhaib, M. and Ahmad, N. (2017), "Studies on stewart platform manipulator: a review", *Journal of Mechanical Science and Technology*, Vol. 31 No. 9, pp. 4459-4470.

Gallardo-Alvarado, J., García-Murillo, M.A., Islam, M.N. and Abedinnasab, M.H. (2016), "A simple approach to solving the kinematics of the 4-UPS/PS (3R1T) parallel manipulator", *Journal of Mechanical Science and Technology*, Vol. 30 No. 5, pp. 2303-2309.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J. and Marín-Jiménez, M.J. (2014), "Automatic generation and detection of highly reliable fiducial markers under occlusion", *Pattern Recognition*, Vol. 47 No. 6, pp. 2280-2292. Elsevier,

Gavin, H.P. (2019), "The Levenburg-Marqurdt algorithm for nonlinear least squares Curve-Fitting problems", *Duke University*, pp. 1-19.

Gopinath, S. and Prince, P.R. (2019), "A comparison of machine-learning techniques for the prediction of the auroral electrojet index", *Journal of Earth System Science*, Vol. 128 No. 7, pp. 1-15.

Han, H.S.A.Q.E., Han, C.Y., Xu, Z.B., Zhu, M.C., Yu, Y. and Wu, Q.W. (2019), "Kinematics analysis and testing of novel 6-P-RR-R-RR parallel platform with offset RR-joints", *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, Vol. 233 No. 10, pp. 3512-3530.

Hernandez, E.M.A.P. (2008), "Advances in robot kinematics: analysis and design, advances in robot kinematics: analysis and design", Springer, available at: https://doi.org/10.1007/978-1-4020-8600-7_19

Huang, X., Liao, Q. and Wei, S. (2010), "Closed-form forward kinematics for a symmetrical 6-6 stewart platform using algebraic elimination", *Mechanism and Machine Theory*, Vol. 45 No. 2, pp. 327-334.

Köker, R. (2013), "A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization", *Information Sciences*, Vol. 222, pp. 528-543.

Mangalathu, S. and Jeon, J.S. (2019), "Stripe-based fragility analysis of multispan concrete bridge classes using machine learning techniques", *Earthquake Engineering and Structural Dynamics*, Vol. 48 No. 11, pp. 1238-1255.

Merlet (2000), "Jacobian and inverse jacobian matrix", *Parallel Robots. Solid Mechanics and Its Applications*, Springer, Dordrecht, pp. 1-10.

Nabavi, S.N., Akbarzadeh, A. and Enferadi, J. (2018), "A study on kinematics and workspace determination of a general 6-P US robot", *Journal of Intelligent & Robotic Systems*, Vol. 91 Nos 3/4, pp. 351-362.

Neumayr, R., Zsombor-Murray, P. and O'Leary, P. (2011), "Precise pose measurement with single camera calibration for planar parallel manipulators", *Transactions of the Canadian Society for Mechanical Engineering*, Vol. 35 No. 2, pp. 201-213.

Parhi, D.R. Deepak, B.B.V.L. and Amrit, A. (2013), "Forward and inverse kinematic models for an articulated robotic manipulator", I J a I C R, No. July 2015.

Perumal, L. (2011), "Quaternion and its application in rotation using sets of regions", *International Journal of Engineering and Technology Innovation*, Vol. 1 No. 1, pp. 35-52.

Haykin, S. (2014), "Neural networks and learning machines", ArXiv Preprint, available at: https://doi.org/978-0131471399

Sinha, P. (2019), "An effiient AODV routing protocol for vehicular ad hoc network", *International Journal of Scientific & Engineering Research*, Vol. 8 No. 4, pp. 962-965.

Sukumar, S.K., Thomas, M.J., Sudheer, A.P. and Joy, M.L. (2019), "Kinematics, structural analysis and control of 3-ppss parallel manipulator", *International Journal of Innovative Technology and Exploring Engineering*, Vol. 8 No. 6, pp. 911-916.

Thomas, M.J., Joy, M.L. and Sudheer, A.P. (2020), "Kinematic and dynamic analysis of a 3 – PRUS spatial parallel manipulator", *Chinese Journal of Mechanical Engineering*, Vol. 33 No. 1, pp. 1-17.

Van Toan, N. and Khoi, P.B. (2018), "A svd-least-square algorithm for manipulator kinematic calibration based on the product of exponentials formula", *Journal of Mechanical Science and Technology*, Vol. 32 No. 11, pp. 5401-5409.

Wang, Z., He, J., Shang, H. and Gu, H. (2009), "Forward kinematics analysis of a six-DOF stewart platform using PCA and NM algorithm", *Industrial Robot: An International Journal*, Vol. 36 No. 5, pp. 448-460.

Xu, P., Li, B., Cheung, C.F. and Zhang, J.F. (2017), "Stiffness modeling and optimization of a 3-DOF parallel robot in a serial-parallel polishing machine", *International Journal of Precision Engineering and Manufacturing*, Vol. 18 No. 4, pp. 497-507.

Corresponding author

Mervin Joe Thomas can be contacted at: mervin_p170047me@nitc.ac.in