COBEM 2021 26th International Congress of Mechanical Engineering
November 22-26, 2021, Virtual Congress, Brazil
ABCM Associação Brasileira de Engenharia e Ciências Mecânicas

COB-2021-0299
Instrumentation of a Parallel Manipulator with Flexible Links: a Neural Network application

Fábio Lúcio Félix
Guilherme Serpa Sestito
Maíra Martins da Silva
School of Engineering of São Carlos, University of São Paulo, São Carlos-SP, Brazil
felixlfabio@gmail.com, guilhermesestito@gmail.com, mairams@sc.usp.br

Abstract. *Parallel kinematic manipulators (PKMs) present higher speed/acceleration ratios, load capacity, rigidity and compaction when compared with serial manipulators. Some industrial applications explore these advantages, such as flight and automobile simulators, pick-and-place machines, and surgical robots. The reduction of the inertia of their components can improve their dynamic performance and energy efficiency. However, this design alternative might yield vibrations requiring the implementation of model-based joint space or task space control strategies. While the former involves the derivation of precise models, the latter requires adequate computation vision schemes. These requirements impose critical challenges such as the use of model updating and image processing techniques. For this reason, studies regarding the use of redundant sensors have been carried out to obtain relevant data for PKM with flexible links. Experimental data from a planar 3RRR is extracted using encoders, strain gauges and a camera during the execution of pre-determined tasks. An Artificial Neural Network is used for estimating the pose of the end-effector's manipulator. In this work, the implementation of this technique is done using a Multi-Layer Perceptrons (MLP) topology for estimating the manipulator's pose. This estimation can be used for improving dynamic models or for implementing task space control strategies.*

Keywords: *Parallel Kinematic Manipulators, Flexible Links, Artificial Neural Networks, Position Estimation, Instrumentation.*

1. INTRODUCTION

The use of parallel kinematic manipulators (PKMs) has shown to be a promising alternative compared to standard models of robotic architectures due to its lightness, rigidity, load capacity, high speed/acceleration and compaction ratios (Merlet, 2006). For this reason, this kinematic configuration can be an alternative for designing energy-efficient robotic manipulators (Yan Li and Bone, 2001; Ruiz *et al.*, 2018). However, their viability requires the overcome of some practical difficulties. Among these issues, we can indicate the presence of singularities within their workspace (Merlet, 2006) and the complexity of their control strategy (Paccot *et al.*, 2009). This complexity is presented for both joint and task space control strategies. On the one hand, joint space control strategies require calculating the inverse kinematic model (IKM), which is cumbersome and time-consuming for PKMs. On the other hand, task space control strategies require measuring the pose of the manipulator's end-effector via vision-based techniques and image processing algorithms (Bellakehal *et al.*, 2011).

Some authors propose the use of redundancy to overcome these technical issues. For example, the presence of singularities can be reduced using kinematic redundancy (Wu *et al.*, 2010; de Carvalho Fontes *et al.*, 2018, 2021), and the use of measurement redundancy can decrease the complexity of the control strategy. Measurement redundancy, which corresponds to obtaining more measurements than degrees of freedom for the manipulator (Muller, 2008), is used by Zubizarreta *et al.* (2012) for more precise acquisition of the end-effector's pose of a 3RRR planar parallel manipulator. The letter R stands for revolute joint, while the underlined letters represent the active joints. The 3RRR manipulator is composed of three parallel kinematic chains. A redundant feedback strategy at the joint and task space level are proposed by Mohan *et al.* (2017) using appropriate sensors. Due to the measurement redundancy of this approach, the actual error of the joint displacements is computed, improving the control performance. A similar strategy has been proposed by Colombo *et al.* (2019) using a rate-limited camera.

The dynamic performance and energy efficiency of PKMs can also be improved by reducing the inertia of their moving components (Nabat *et al.*, 2005; Zhang *et al.*, 2015). However, the reduction of inertia of its components increases the flexibility of the entire structure, making the control system's design difficult due to the complexity of the kinematic and dynamic models as treated in (Zhang *et al.*, 2015; Wang *et al.*, 2009), and the tight requirements of the vision-based

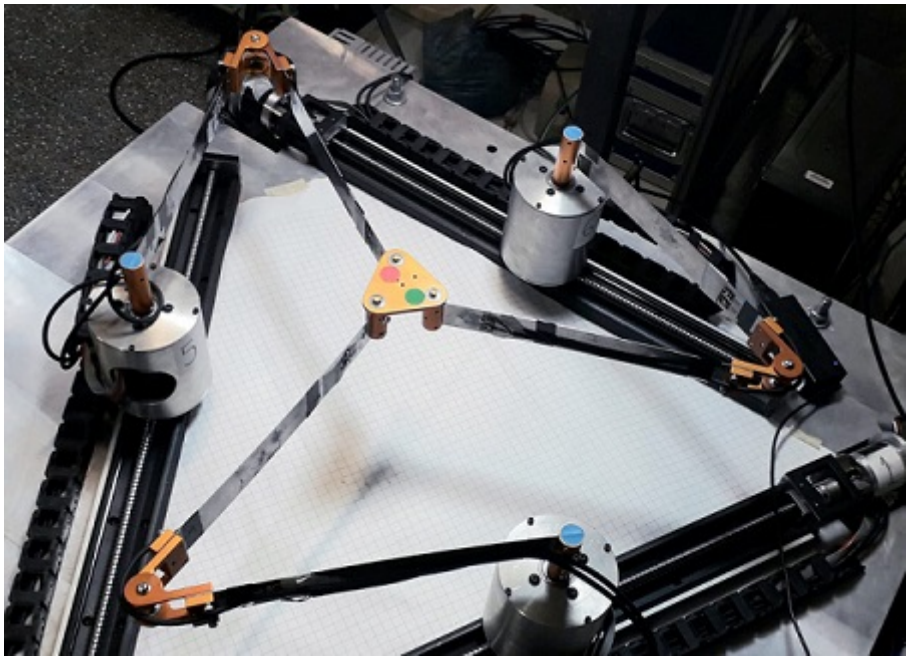

Figure 1. Prototype 3RRR with Flexible Links

techniques. Therefore, measurement redundancy and metamodeling strategies can also be an alternative for implementing controllers for this kind of system.

The basic approach for metamodeling techniques is to build approximations that will provide an efficient response and a functional relationship between the input and output variables under analysis (Simpson *et al.*, 2001). This implementation has three fundamental steps: the experimental data acquisition, the selection of the model that appropriately represents the output-input relationship, and finally, the model parameters adjustment to the measured data (training step). There are several alternatives to perform each of these steps. A comprehensive review of these techniques can be found in Simpson *et al.* (2001).

In this work, the end-effector pose of a 3RRR planar parallel manipulator with flexible links, depicted in Figure 1, is estimated using data acquired by encoders and strain gauges. This estimation is carried out by a metamodel trained to mimic the pose measurement acquired by a camera. The first step was acquiring the training and testing datasets during the execution of several predefined tasks. The second step was selecting an appropriate metamodel to represent the acquired data. In this work, an Artificial Neural Network (ANN) is investigated since this methodology is well-known and shows good results for the approximation of functions (Haykin, 2009). An ANN can have one or multiple layers of neurons, also called perceptron, the simplest form of setting up an artificial neural network. For perceptrons with more than one layer, called Multi-Layers Perceptrons (MLP), the training process is based on the Backpropagation algorithm (da Silva *et al.*, 2017). The third step is the training and testing metamodeling procedure. Figure 2 illustrates the data acquisition with the proposed instrumentation and the proposed method for estimating the posture of the end effector of the 3RRR manipulator.

This work is organized as follows. Section 2 presents a theoretical review of artificial neural networks. Section 3 briefly describes the kinematic model of the flexible 3RRR and details the experimental prototype, including its instrumentation.

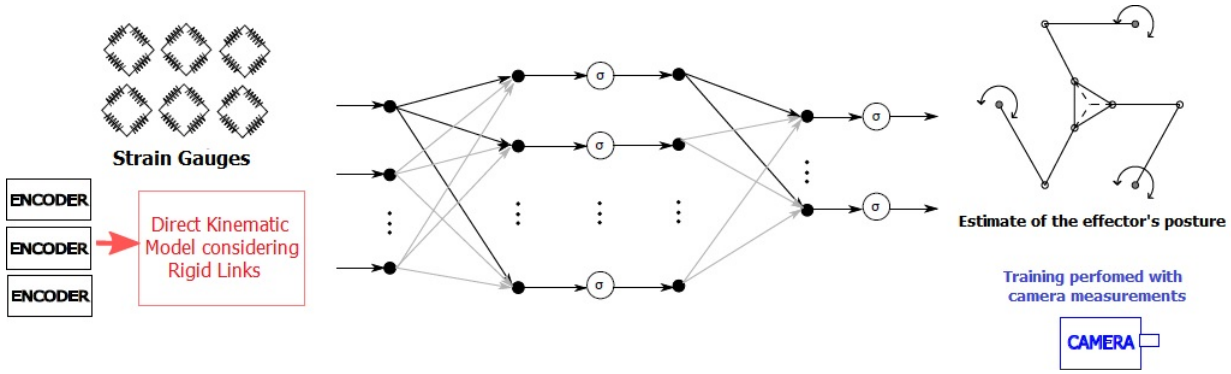

Figure 2. Neural network being used for estimation of effector posture

Results are presented and discussed in Section 4, while conclusions are drawn in Section 5.

2. ARTIFICIAL NEURAL NETWORKS

An artificial neural network (ANN) architecture is defined as how its various neurons are arranged with each other. These arrangements are structured through the synaptic connections of neurons. The topology of a neural network refers to the different arrangements that it can assume (da Silva *et al.*, 2017). For example, you can have two topologies of the same architecture, one consisting of 10 neurons and the other with 20 neurons, or two topologies with different activation functions. The main architectures of a neural network can be divided into single-layer feedforward networks, multi-layer feedforward networks (see Figure 3), and recurrent networks (Haykin, 2009). For this work, the architecture chosen is multi-layer, also called multi-layer perceptron or MLP.

An MLP is a neural network connecting an input, an output, and hidden layers. Figure 3 illustrates a MLP with p inputs, one output and a single hidden layer with L neurons. In this work, the input data are composed by the estimated end-effector's pose ($[x\ y\ \alpha]^T$) and six strain gauges measurements ($p = 9$), described as f_i where $i = 1 \dots p$ in Fig. 3. The end-effector's pose is estimated using the kinematic model of the 3$\underline{\text{R}}$RR considering rigid links and the encoders' measured data ($[\theta_1\ \theta_2\ \theta_3]^T$). This model is summarized in Section 3.1.

The output and hidden layers are composed of neurons and weighted summations. The inputs for the neurons of the hidden layer are described by u_j where $j = 1 \dots L$ can vary according to the number of neurons of this layer. The input of the output layer is described as ${}^{o}u$. These inputs are composed of weighted sums, where ${}^{*}w_{ij}$ are the weighting factors. The outputs of the neurons of the hidden and output layers, described by y_j where $j = 1 \dots L$ and o, are derived using hyperbolic tangent and linear activation functions, respectively. The hyperbolic tangent activation function is described as:

$$y_j = \sigma(u_j) = \frac{1 - e^{-\beta u_j}}{1 + e^{-\beta u_j}}. \tag{1}$$

where $j = 1 \dots L$.

Three MLPs are trained in this work. The output layer of each MLP is responsible for estimating the $[x\ y\ \alpha]^T$ based on the actual camera's measurements. In this work, an offline supervised learning algorithm is used with a Levenberg-Marquardt Backpropagation training algorithm for adjusting the synaptic weights used for deriving the inputs of the neurons. In this procedure, the data is separated between training and testing sets. The former is used for the ANN learning process and the latter for the estimator's performance verification. The training algorithm consists of minimizing the error between the outputs of the training data set (the actual camera's measurements) and the output layer. In addition, a consistent analysis was carried out regarding the errors associated with the test and training subsets to overcome situations of underfitting and overfitting. This analysis is necessary to prevent the learning algorithm from choosing to use a larger number of neurons while the ratio between the test and training subset error increases.

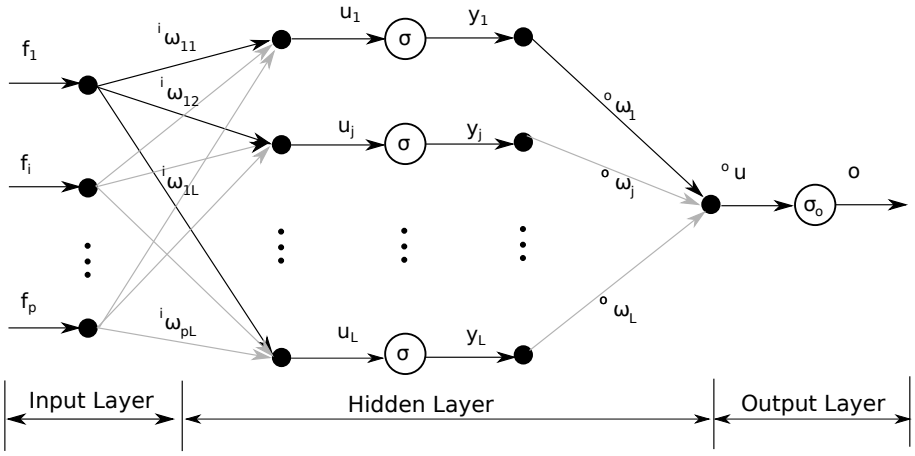

Figure 3. Illustration of a MLP with a single hidden layers of neurons

3. 3$\underline{\text{R}}$RR PLANAR PKM

In this section, the kinematic model of the 3$\underline{\text{R}}$RR, a planar PKM, is firstly summarized. Secondly, the experimental prototype is described.

3.1 Kinematics

The 3RRR is a planar parallel mechanism that has three kinematic chains connected to the A_iB_i effector, where the subscript $i = 1, 2, 3$ is according to the kinematic chain. Each of these chains has three rotating joints, one active (R) and two passive (RR). As shown in Figure 4, the three active joints of the manipulator transmit the movement to the end-effector, with center at the point D. The global coordinate system fixed to the end-effector O_{xoyo} is located at the point D and has an orientation, relative to this frame, of angle α. The angular orientation of the A_iB_i links is given by the angle θ_i, and β_i is the angular orientation for the B_iC_i links. The end-effector position vector $\mathbf{X} = [x\ y\ \alpha]^T$, is relative to the fixed coordinate system and is the set of points that will be estimated, given a trajectory that the manipulator will execute.

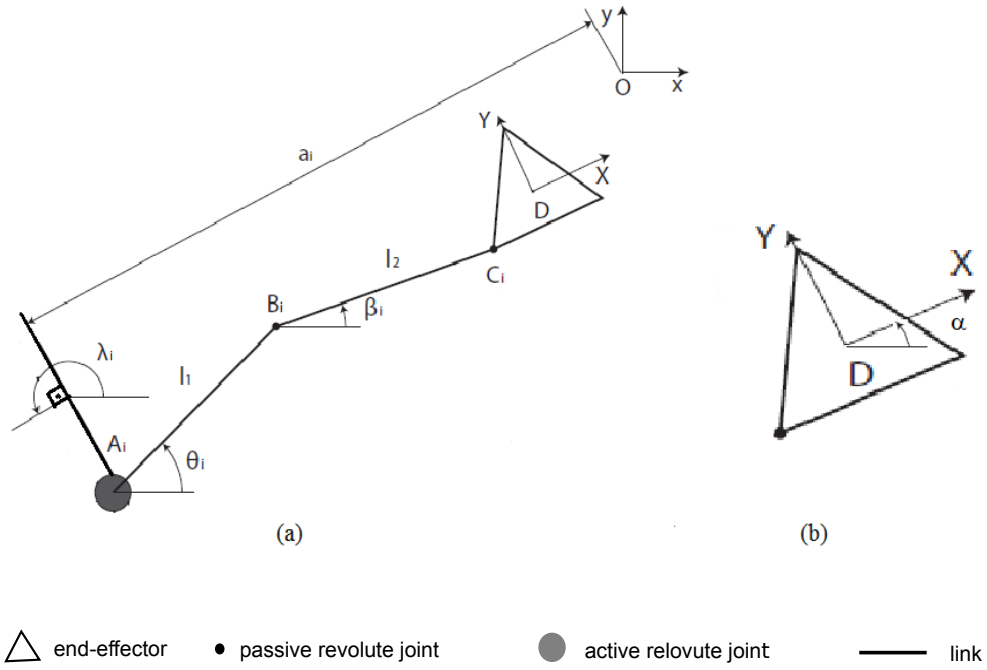

Figure 4. (a)Illustration of a kinematic chain of 3RRR and (b) Details of the manipulator's end-effector

According to Figure 4, it is possible to describe the position of the joints A_i, B_i and C_i, in relation to the end-effector's coordinate system as

$$\mathbf{r}_{B_i} = \mathbf{r}_{A_i} + l_1 \begin{bmatrix} \cos(\theta_i) \\ \sin(\theta_i) \end{bmatrix}, \quad \text{and} \tag{2}$$

$$\mathbf{r}_{C_i} = \mathbf{r}_{B_i} + l_2 \begin{bmatrix} \cos(\beta_i) \\ \sin(\beta_i) \end{bmatrix} \quad . \tag{3}$$

The geometric constraint equation of the links $\left\|\overrightarrow{B_iC_i}\right\| = \|\mathbf{r}_{C_i} - \mathbf{r}_{B_i}\| = l_2$, allows us to derive an equation for each angle θ_i (Fontes and da Silva, 2016), where $e_{i1} = -2l_1\rho_i$, $e_{i2} = -2l_1\mu_i$, $e_{i3} = \mu_i^2 + \rho_i^2 + l_1^2 - l_2^2$, $\mu_i = x_D + h_i cos(\lambda_i + \alpha) - a_i cos(\lambda_i)$ and $\rho_i = y_D + h_i sin(\lambda_i + \alpha) - a_i sin(\lambda_i)$. Eq. 4 is also called inverse kinematics for the 3RRR manipulator.

$$\theta_i = 2\tan^{-1}\left(\frac{-e_{i1} \pm \sqrt{e_{i1}^2 + e_{i2}^2 - e_{i3}^2}}{e_{i3} - e_{i2}}\right). \tag{4}$$

To determine active joint velocity, $\dot{\mathbf{\Theta}} = [\dot{\theta_1}\dot{\theta_2}\dot{\theta_3}]^T$, and the end-effector's velocity $\dot{\boldsymbol{X}} = [\dot{x}\dot{y}\dot{\alpha}]^T$, we use the geometric constraint equation commented above and derive it with respect to time. Knowing the relationships described by equations

2 and 3, and also assuming that the λ_i is constant, it is possible to arrive at the following equation after some mathematical manipulations

$$l_2\ cos(\beta_i)\ \dot{x}_P + l_2\ sin(\beta_i)\ \dot{y}_P + l_2\ h\ \dot{\alpha}\ sin(\beta_i - \alpha - \lambda_i) = l_1\ l_2\ \dot{\theta}\ sin(\beta_i - \theta_i). \tag{5}$$

Thus, it is possible to rewrite Eq.5 in matrix form to determine the relationship that maps velocities in joint space $\dot{\boldsymbol{\Theta}}$, with velocities in task space $\dot{\boldsymbol{X}}$ as being $\boldsymbol{A}\dot{\boldsymbol{X}} = \boldsymbol{B}\dot{\boldsymbol{\Theta}}$, where

$$\mathbf{A} = \begin{bmatrix} l_2\ cos(\beta_1) & l_2\ sin(\beta_1) & l_2\ h\ sin(\beta_1 - \alpha - \lambda_1) \\ l_2\ cos(\beta_2) & l_2\ sin(\beta_2) & l_2\ h\ sin(\beta_2 - \alpha - \lambda_2) \\ l_2\ cos(\beta_3) & l_2\ sin(\beta_3) & l_2\ h\ sin(\beta_3 - \alpha - \lambda_3) \end{bmatrix} \tag{6}$$

$$\mathbf{B} = \begin{bmatrix} l_1\ l_2\ sin(\beta_1 - \theta_1) & 0 & 0 \\ 0 & l_1\ l_2\ sin(\beta_2 - \theta_2) & 0 \\ 0 & 0 & l_1\ l_2\ sin(\beta_3 - \theta_3) \end{bmatrix} \tag{7}$$

Therefore, the Jacobian matrix that maps the velocities in joint space to the task space is such that $\boldsymbol{J} = \boldsymbol{A}^{-1}\boldsymbol{B}$ if $\boldsymbol{A}$ is invertible. The development of the kinematic model carried out in this section is used later to adjust and correct the angular position values of the active joints obtained by the encoders.

3.2 Experimental Prototype

In this section, the communication and instrumentation of the 3RRR prototype are described. The motors that perform the linear displacements δ_i, illustrated in Figure 4, are not covered in this work, so the joints related to motors $M1$, $M2$ and $M3$, shown in Figure 5 are considered as passive. Thus, the only active joints that provide movement to the manipulator are the $M4$, $M5$ and $M6$ motor joints. These joints are powered by brushless Maxon $EC60$ flat motors, with $100W$ of power and a current rating of $2,3A$, coupled to $GP52C$ planetary reducers with a reduction of $3.5:1$, providing a nominal rotation of $1200RPM$ and nominal torque of $0,82Nm$. Each of the motors has a Maxon $EPOS250/5$ controller with a power supply up to $50Vdc$ and a current of $5A$.

A diagram of the communication can be seen in Figure 5. The communication of these motors with the $DSPACE1103$ is done via CAN protocol with a transmission rate of $250kbit/s$. Therefore, both the motor drive and the acquisition of the encoder data are carried out in this way. HBM $350-E$ strain gauges arranged with full-bridge were attached to each link of the manipulator to perform the deformation measurements of each link when the manipulator executes a task. The strain gauge information needs to be read by the DSPACE A/D inputs, and for this reason, it goes through an HBM HB40 CLIPX signal amplifier.

To directly measure the position of the manipulator end-effector, an $oCam\text{-}5CRO\text{-}U$ camera with USB 3.0 interface with CMOS image sensor and rolling shutter is used, at a resolution of $640x480$ pixels and a maximum framerate of $120fps$. The images obtained by the camera are processed in a computer using a filter that will detect the most relevant points of the manipulator: the center of the end-effector, D_i, and the fixed points A_i (see Figure 4). This filter is implemented in Visual Studio and uses the Computer Vision library called OpenCV, developed in C++. Finally, the transmission of the data measured by each frame captured by the camera is done using serial communication through the $RS\text{-}232$ standard between the computer and the DSPACE acquisition board, with a frequency of $60fps$.

With direct measurement data from the camera, strain gauge strains and angular position from the encoders, DSPACE sends this dataset to a central computer where it is possible to capture all this information simultaneously. It is necessary to treat the strain gauges and the camera data with a low-pass digital filter since the obtained signal has high-frequency noise.

4. RESULTS

As described in the previous sections, an ANN algorithm using a multi-layer feedforward network architecture was developed to estimate the end-effector's pose. The end-effector's pose, x, y, and α, was acquired using the images obtained by the camera. An evaluation regarding the number of neurons to be used by the ANN was carried out. The objective was to find the number of hidden layer neurons to minimize the error between the actual data (captured by the camera) and the estimated data (by the ANN). The number of neurons in the hidden layer was varied from 1 to 151 at a fixed pace of 5 neurons. In each training step, the same architecture with the number of neurons was trained five times to prevent the learning algorithm from selecting a training data set that, coincidentally, is at a local minimum in the error minimization process. Table 1 shows the best mean squared errors (MSE) regarding each ANN (x, y, and α) and the number of neurons of the hidden layer.

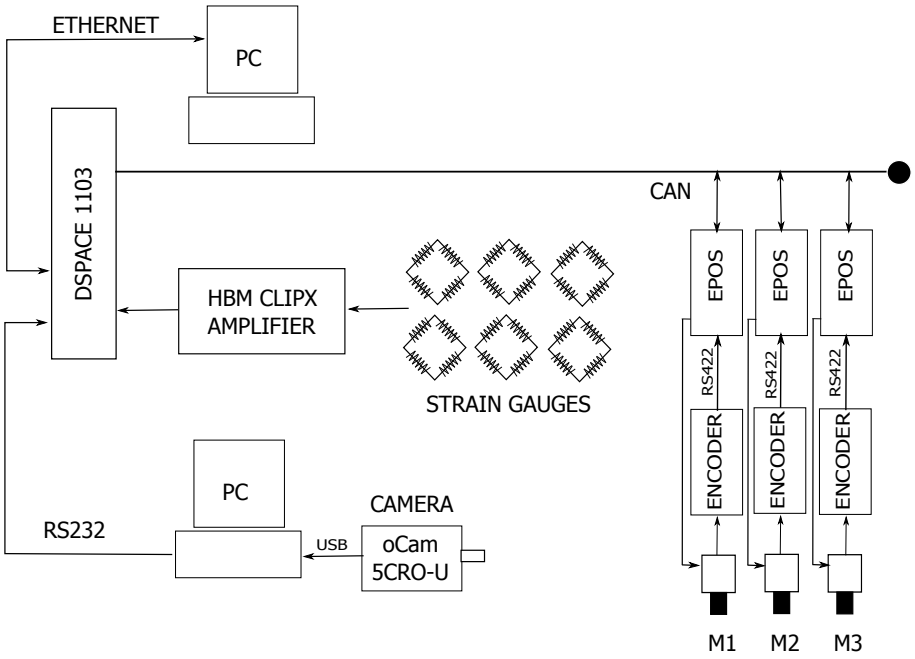

Figure 5. Illustration of instrumentation and its communication scheme

Table 1. Result of the best neural network after training from 1 to 150 neurons with a step of 5.

Estimated	**Performance based on MSE**	**Numbers o Neurons**
Coordinate X	$1.5727 * 10^{-7}$	141
Coordinate Y	$1.2362 * 10^{-7}$	136
Angle α	$1.5283 * 10^{-6}$	126

To avoid situations of overfitting and underfitting, it was necessary to build error behavior graphs between training subsets and test subsets. The importance of this analysis is because the training algorithm does not choose a neuron architecture that is memorizing and memorizing the responses against the data introduced in the inputs. Figure 6(a) illustrates the behavior of these errors during the training of the network to estimate the x-position of the manipulator's end-effector. Likewise, Figure 6(b) is for the estimate of the y-position and Figure 6(c) for the estimate of the angle position α of the end-effector. It is important to note the behavior of the dots in blue and red for the stipulated maximum number of neurons in these figures. The difference between the error values does not seem to increase as the number of neurons increases. This type of behavior could be observed if the overfitting situation was occurring in the network training process.

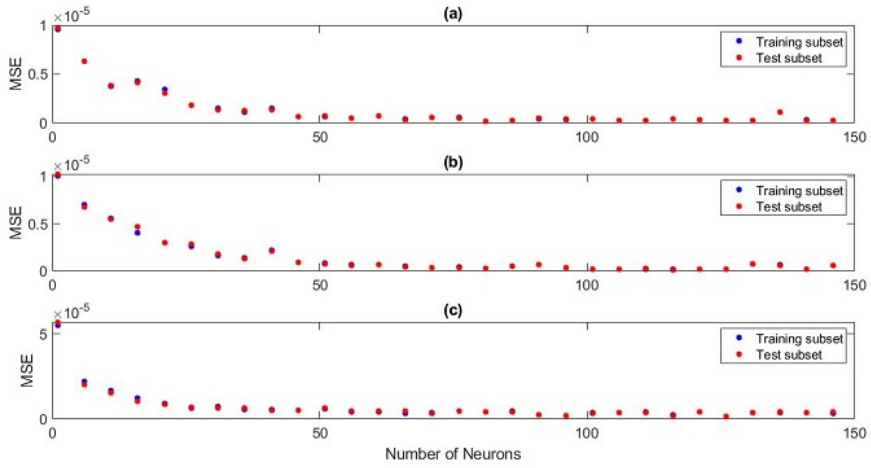

Figure 6. Relationships between test and training subset error as a function of the number of neurons in the hidden layer.

The result of the best training performed to estimate the x-position of the manipulator end-effector can be seen in

Figure 7. The solid blue curve is the data acquired by the camera data during the execution of a ten-second task. The red dotted curve shows the data generated by the ANN with 141 neurons in the hidden layer. This architecture resulted in a performance of $1.5727 * 10^{-7}$ as shown in Table 1.

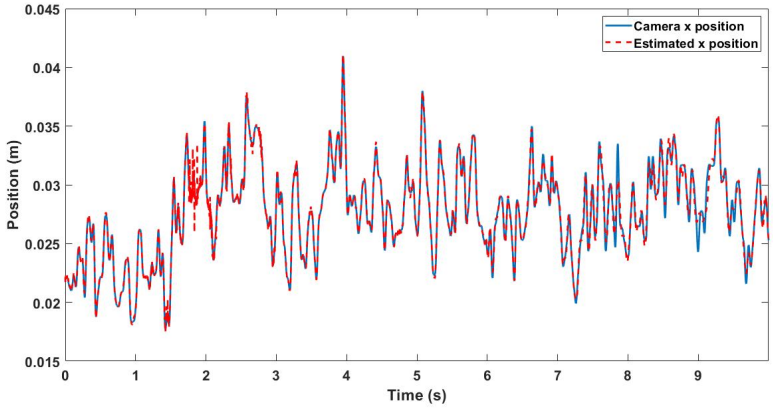

Figure 7. Comparison between the x-position captured by the camera and the x-position estimated by ANN.

To estimate the y-position curve, Figure 8, the learning process chose an ANN with 136 neurons in the hidden layer resulting in a performance of $1.2362 * 10^{-}7$ also shown in Table 1. Figures 7 and 8 shows that both estimation of x and y position present noise. Further studies are required to eliminate this issue.

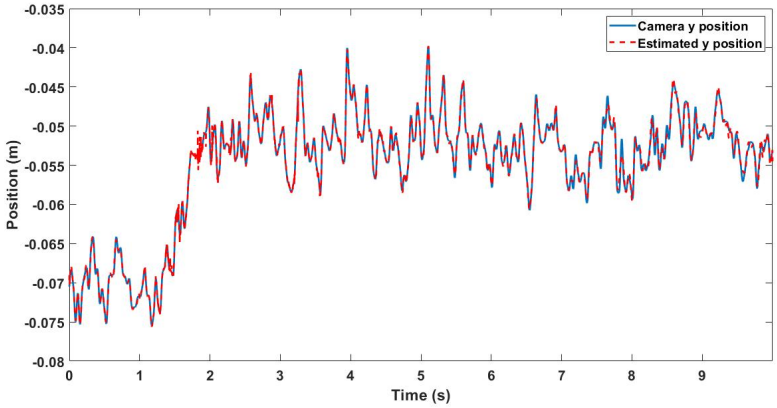

Figure 8. Comparison between the y-position captured by the camera and the y-position estimated by ANN.

Finally, the estimation of the angular position α of the manipulator's end-effector can be seen in Figure 9. The continuous curve in blue illustrates α captured by the camera during the task execution, while the dashed curve in red illustrates the ANN estimative. For this, an architecture with 126 neurons, resulted in a performance of $1.5283 * 10^{-6}$, was used.

In general, the estimated data generated by the learning algorithm achieved satisfactory results. In Figs. 7 and 8, we can see some instability in the estimation around the two seconds. This instability happens because the task execution command was carried out approximately at this moment. The response plots from strain sensors show this effect clearly.

5. CONCLUSIONS

The design of parallel kinematic manipulators with flexible links proved to be a task to perform with a high level of care and attention to resolve most situations that may trigger unsatisfactory results. The inertia reduction of the manipulator's mobile components proposed by several researchers revealed the essential step to be studied and fully understood due to the degree of complexity that the project starts to take. The flexibility now built into its structure makes it challenging to acquire and determine the current position and how the end-effector behaves along a trajectory stipulated by a given task.

To solve this situation, the proposal to develop an Artificial Neural Network Algorithm using a Multi-layer Perceptron proved to be effective and produced satisfactory results. Furthermore, the method of cross-validation by random sampling was implemented during the development phase of the learning algorithm. This method forces the learning process to use input data for the learning and validation, and testing phases. Furthermore, analyses regarding the prevention of

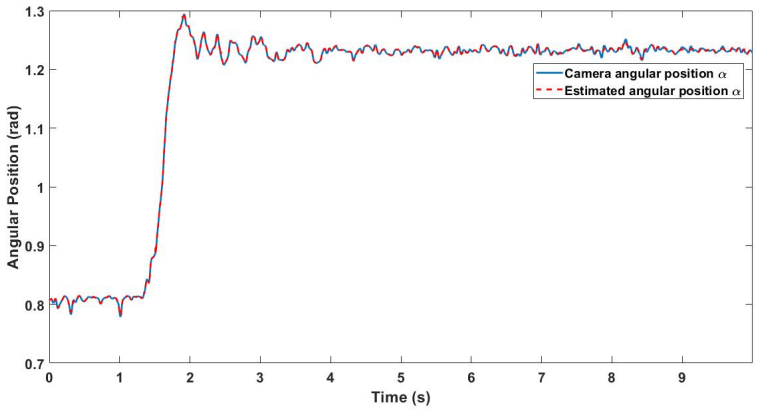

Figure 9. Comparison between the α-position captured by the camera and the α-position estimated by ANN.

overfitting/underfitting were widely addressed and implemented to avoid memorizing the responses of the system outputs to be estimated.

Therefore, this work presents a contribution in the fields of PKM regarding the estimation and identification of systems using Artificial Neural Networks. Furthermore, the proposal has achieved good performance determined by the learning algorithm during its training process. The trained ANN can be used in control loops.

6. ACKNOWLEDGEMENTS

This research is supported by FAPESP 2018/21336-0. Fábio, Guilherme and Maíra M. da Silva are thankful for their scholarships CAPES and CNPq 301338/2018-3, respectively.

7. REFERENCES

Bellakehal, S., Andreff, N., Mezouar, Y. and Tadjine, M., 2011. "Vision/force control of parallel robots". *Mechanism and Machine Theory*, Vol. 46, No. 10, pp. 1376–1395. doi:10.1016/j.mechmachtheory.2011.05.010. URL `https://doi.org/10.1016/j.mechmachtheory.2011.05.010`.

Colombo, F.T., de Carvalho Fontes, J.V. and da Silva, M.M., 2019. "A visual servoing strategy under limited frame rates for planar parallel kinematic machines". *Journal of Intelligent & Robotic Systems*, Vol. 96, No. 1, pp. 95–107. doi:10.1007/s10846-019-00982-7. URL `https://doi.org/10.1007/s10846-019-00982-7`.

da Silva, I.N., Spatti, D.H., Flauzino, R.A., Liboni, L.H.B. and dos Reis Alves, S.F., 2017. *Artificial Neural Networks*. Springer International Publishing. doi:10.1007/978-3-319-43162-8. URL `https://doi.org/10.1007/978-3-319-43162-8`.

de Carvalho Fontes, J.V., Colombo, F.T. and da Silva, M.M., 2021. "Model-based joint and task space control strategies for a kinematically redundant parallel manipulator". *Robotica*, Vol. x, No. x. doi:10.1007/xxx. URL `https://doi.org/xxx`.

de Carvalho Fontes, J.V., Santos, J.C. and da Silva, M.M., 2018. "Numerical and experimental evaluation of the dynamic performance of kinematically redundant parallel manipulators". *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, Vol. 40, No. 3. doi:10.1007/s40430-018-1072-1. URL `https://doi.org/10.1007/s40430-018-1072-1`.

Fontes, J.V. and da Silva, M.M., 2016. "On the dynamic performance of parallel kinematic manipulators with actuation and kinematic redundancies". *Mechanism and Machine Theory*, Vol. 103, pp. 148–166. doi: 10.1016/j.mechmachtheory.2016.05.004. URL `https://doi.org/10.1016/j.mechmachtheory.2016.05.004`.

Haykin, S., 2009. *Neural Networks and Learning Machines*, Vol. third. Person Education, Upper Saddle River, NJ.

Merlet, J.P., 2006. *Parallel Robots*. Springer Netherlands.

Mohan, S., Mohanta, J.K., Huesing, M. and Corves, B., 2017. "Dual-loop motion control for geometric errors and joint clearances compensation of a planar 2-PRP1-PPR manipulator". In *Mechanisms, Transmissions and Applications*, Springer International Publishing, pp. 171–180. doi:10.1007/978-3-319-60702-3$_1$8.URL