2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Qualitative approach for inverse kinematic modeling of a Compact Bionic Handling Assistant trunk

A. MELINGUI, R. MERZOUKI, J. B. MBEDE, C. ESCANDE, B. Daachi, and N. BENOUDJIT

***Abstract*— Compact Bionic Handling Assistant (CBHA) is a continuum manipulator, with pneumatic-based actuation and compliant gripper. This bionic arm is attached to a mobile robot named Robotino. Inspired by the elephant's trunk, it can reproduce biological behaviors of trunks, tentacles, or snakes. Unlike rigid link robot manipulators, the development of high performance control algorithm of continuum robot manipulators remains a challenge, particularly due to their complex mechanical design, hyper-redundancy and presence of uncertainties. Numerous studies have been investigated for modeling of such complex systems. Such continuum robots, like the CBHA present a set of nonlinearities and uncertainties, making difficult to build an accurate analytical model, which can be used for control strategies development. Hence, learning approach becomes a suitable tool in such scenarios in order to capture un-modeled nonlinear behaviors of the continuous robots. In this paper, we present a qualitative modeling approach, based on neuronal model of the inverse kinematic of CBHA. A penalty term constraint is added to the inverse objective function into Distal Supervised Learning (DSL) scheme to select one particular inverse model from the redundancy manifold. The inverse kinematic neuronal model is validated by conducting a real-time implementation on a CBHA trunk.**

I. INTRODUCTION

In recent years, continuum manipulators ([1], [2]...) have been the subject of intensive research ([3], [4], [5]) due to their dexterity, and ability to adapt dynamically to the manipulation in unstructured environments. Such classes of robots have often a high number of passive joints associated with links designed with soft materials. Hence, their use for practical applications requires modeling and development of real-time efficient algorithms to extract their full physical potential.

Hyper-redundant robots such as CBHA are those having a higher degree of freedom, involving more mobility and reachability of targets citeantonelli2009stability. This redundancy can be exploited for obstacle avoidance, singularities elimination, various criteria performance enhancing, and as well as for smooth motion tasks achieving. Focusing on their kinematic modeling, unlike of rigid manipulators where the pose of any point in robot workspace can be fully defined by links lengths and joint angles, the kinematic of continuum robots remains difficult to obtain with high accuracy, because, they are under-determined systems with high number of parameters. Their modeling is carried out with continuum mechanics approach. Therefore, their kinematic equations can be solved by using integral resolutions, which are time consuming for autonomous systems.

The most widely proposed methods for solving the inverse kinematic problem for redundant manipulators are based on Jacobian pseudo-inverse citemayorga1995fast, [8]. The latter uses the forward kinematic transformation and quaternion representation of the orientation matrices. Afterward, a computer simulation is performed to evaluate the efficiency of the Jacobian in converting joint velocities into Cartesian velocities and to investigate the accuracy of Jacobian pseudo-inverse for various sampling times. But in the case of hyper-redundant manipulators with high degrees of freedom, the computational burden of pseudo-inverse Jacobian becomes prohibitive, despite of proposed improvements [9], [10].

Some researchers have investigated how continuum robots behavior can be analytically modeled [3], [11], [5] but these approaches lead often to less accurate models due to considered assumptions (constant curvature bending, no gravity force...). Hence, they cannot capture the full complexity of continuous deformations. Because continuum deformations are potentially infinite-dimensional, since the entire arm's material is deformable. This kind of deformation could only be reconstructed using redundant sensors. In such scenarios, learning approach becomes a suitable tool in order to capture un-modeled nonlinear behaviors of the continuous robots.

The learning approach has been investigated by some researchers. In [12], [13], the authors tackle the redundancies by partitioning the configuration space in a set of local regions, and building a global solution from these local regions. But the use of local experts requires an oracle determining model responsibilities, which may become difficult to obtain for hyper-redundant robots. In [10], authors used an adaptive MLP neural network to control a 6 DOF parallel robots in force/position. An interesting approach is presented in [14], [15], the authors proposed the goal babbling approach to solve the inverse kinematic problem of Bionic Handling Assistant (BHA). They referred to the successful bootstrapping of some motor skill by repeating the process to accomplish multiple goals related to that skill. Starting from a particular goal, a new goal is randomly drawn from a set of target positions and the endpoints are linearly interpolated. The

A. MELINGUI, R. MERZOUKI, and C. ESCANDE are with LAGIS, (UMR), (CNRS) 8146, Ecole Polytechnique de Lille, 59655 Villeneuve d'Ascq, France (achille.melingui@etudiant.univ-lille1.fr, rochdi.merzouki@polytech-lille.fr, and coralie.escande@googlemail.com)
J. B. MBEDE is with Electrical Engineering and Telecommunications Department of Ecole Nationale Superieure Polytechnique. The University of Yaounde I, P. O. Box: 8390 Yaounde, Cameroon (mbede@neuf.fr)
B. Daachi is with the LISSI laboratory, France (daachi@u-pec.fr)
N. BENOUDJIT is with Department of Electronics, University of Batna, Algeria (nbenoudjit@yahoo.com)

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

current inverse kinematic is used to estimate the current posture. However, if this approach has given excellent results for BHA trunk [14], its implementation to its compact version (CBHA) requires some improvements, because their control architecture involves several interactive loops which can increase significantly the computational time. In addition, the CBHA manipulator is designed to be used as an extension to mobile robot named Robotino which is supposed to be reactive with autonomous navigation. Braganza et al. [16] implemented a low-level joint controller of a soft extensible manipulator by using neural networks to compensate for the dynamic uncertainties. Giorelli et al. [17] used a feedforward neural networks to approximate the inverse kinematic model of a non constant curvature soft manipulator driven by three cables. Thus, a geometrical model of the manipulator has been used for sample data pairs generation and a direct inverse learning approach has been used to approximate the IKM.

In contrast to Giorelli et al. [17] approach, where a direct inverse learning have been used to approximate the IKM of a 3 inputs/3 outputs system, in this paper, due to redundancies, a squared penalty term is incorporated in Distal Supervised Learning (DSL) scheme to select one particular inverse model from the redundancy manifold. Jordan et al. [18] proposed a DSL approach to determine the inverse model of a controlled system based on its forward model. In the literature, Stitt et al. [19] used the DSL approach to control biped robot movements, and Howard et al. [20] used it to build a system that can learn to mimic speech using its own vocal tract.

In this work, MLP and RBF Neural Networks are integrated in DSL scheme to determine the inverse kinematic of a CBHA manipulator, using only the information of the Cartesian position of the CBHAâĂŹs effector. This paper is structured as follows: Section 2 presents the inverse kinematic problem formulation. While the direct inverse, the DSL learning, and the prediction by with neural networks is presented in Section 3. Section 4 provides experimental results and discussions. Section 5 gives the conclusions and future works.

II. INVERSE KINEMATIC PROBLEM FORMULATION

The CBHA depicted in Fig. 1 is attached to an omni-directional mobile robot platform called Robotino to form the RobotinoXT (Fig. 2). It comprises two main segments each with three pneumatic-actuated bellows. A ball-joint as wrist, controlled using two actuators, and two compliant jaws constituting the gripper, controlled by one actuator. Each actuator can be controlled separately. The venting of the backbone tubes allows resetting its shape; and supplying it with compressed air leads to its expansion. The bionic trunk composes of nine sensors; six wire-potentiometers, installed on the surface of each flexible backbone tubes to measure their actual elongations. Two sensors are used for the rotating part and the last one to detect the gripper status.

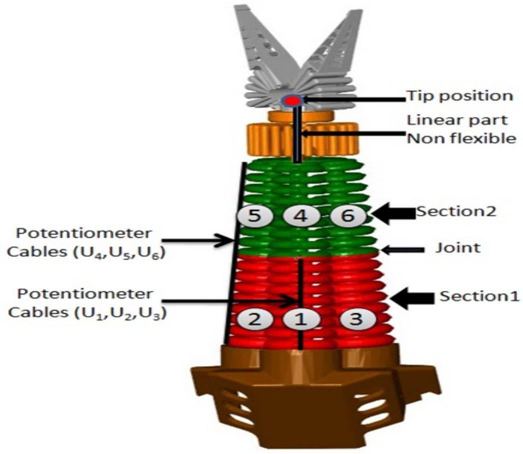

Fig. 1. CBHA manipulator

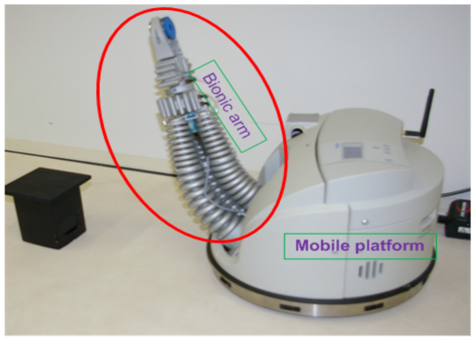

Fig. 2. Robotino XT platform

Nowadays, the CBHA placed of the Robotino mobile robot platform is controlled in an open-loop configuration using a joystick interface. The problem is to keep this control autonomous and in closed-loop scheme. The main difficulty is the establishment of an accurate IKM allowing obtaining the relationship between the Cartesian coordinates of the tip of the arm and the tube-lengths (non-linearities, uncertainties, and non-uniqueness of the inverse kinematic function). Thus, this is our main interest in this work. We study the IKM of the two jointed segments (red and green segments of the Fig. 1), so that the Cartesian coordinates and the tube-lengths are respectively used as inputs and outputs of the neural network. Note that, we have opted to IKM model (the relationship between the Cartesian coordinates of the tip of the arm and the tube-lengths), because frictions and hysteresis related to CBHA structure can cause largely different postures when applying the same pressure several times. Since pressure does not provide reliable information about the robot position and movement in space, reaching solely concerns the geometric information (length sensors). This geometric information (length sensor values) can be controlled by dynamically adjusting the pressure in each actuator.

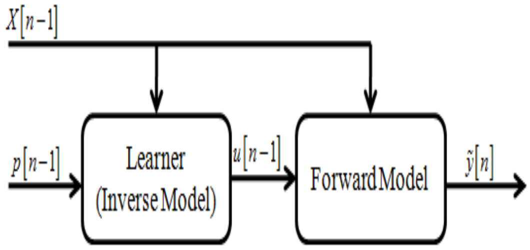

Fig. 3. Composite learning system [18]

III. DIRECT AND DISTAL SUPERVISED LEARNING

A. Direct Supervised Learning

The idea in direct inverse modeling is to observe the input/output behavior of the environment and to train the inverse model directly by reversing the roles of the inputs and outputs. Although the excellent results obtained [17], [21] in direct supervised learning, we note two drawbacks that limit its usefulness: First, when the environment is characterized by a many-to-one mapping from actions to sensations, the inverse mapping will map more than one images to a given point. The particular manner in which the inconsistency is resolved depends on the form of the cost function; the use of the sum-of-squared error yields an arithmetic average over points that map to the same endpoint (centroid). If the centroid does not belong to the manifold of the images, the non-linear many-to-one mappings can yield non convex inverse images. The second drawback with direct inverse modeling is that it is not "goal-directed." The algorithm samples in action space without regard to particular targets or errors in the sense space. That is, there is no direct way to find an action that corresponds to a particular desired sensation. To overcome the two problems, Jordan et al. [18] proposed a new architecture for control that they called Distal Supervised Learning.

B. Distal Supervised Learning approach

The distal supervised learning consists in composing a learning system as depicted in (Fig. 3). The current state of the environment is $X\left[n-1\right]$. The intention is $p\left[n-1\right]$, the action is $u\left[n-1\right]$, and the predicted outcome from the forward model is $\tilde{y}\left[n\right]$. We will also refer to the actual outcome as $y\left[n\right]$ and the desired output as $y^{*}\left[n\right]$. The forward model is a model that predicts the outcome of the environment given the current state and the action. The forward model can be learned by applying actions and comparing actual outcomes $y\left[n\right]$ with predicted outcomes $\tilde{y}\left[n\right]$. The idea of Jordan et al. [18] was to avoid the direct inverse modeling entirely. They used the fact that the composition of the inverse and forward models must yield the identity function. They proposed training first a neural network to model the forward kinematics, and to use this network to train indirectly the inverse model. The composite learning system can be trained by any supervised learning algorithm (back-propagation algorithm, generalized delta learning rule...); however, the learning algorithm must not alter the forward model (fixed forward weights) while the composite system is being trained. The inverse model will be eventually learned if the training input-output pairs stand for the identity function. In this way, the effect is that only one of the possibly many solutions is chosen for a given target point. But, without additional information about the particular structure of the input-to-action mapping there is no way of predicting which of the possibly infinite set of inverse models the procedure will find. Moreover, further virtue of the distal learning approach is the possibility to incorporate additional constraints in the learning procedure. In this work, a squared penalty term is added to the objective function of the inverse neural network into DSL scheme to select one particular inverse model from the redundancy manifold.

C. Prediction procedure based on MLP neural networks

A multi-layer perceptron neural network is composed of a large number of highly interconnected units (neurons) working in parallel and organized in layers with a feed-forward information flow. The architecture of the MLP is structured as follows: the signals flow consecutively through the different layers from the input to the output layer. The intermediary layers are known as hidden layers. For each layer, each elementary unit calculates a scalar product between a vector of weights and the output vector given by the previous layer. A transfer function is subsequently applied to the result to produce an input for the next layer. A common transfer function for the hidden layers is the sigmoid function:

$$f(x) = \frac{1}{1+\exp(-x)}. \tag{1}$$

Arriving at the neuron of the output layer, other transfer function can be used; for example, the identity function (simple linear activation) can be used for regression problems. MLP neural networks (MLPNNs) are trained by the error back-propagation (EBP) algorithm, optimized according to a predefined criterion [22].

D. Prediction procedure based on RBF neural networks

RBFNN is composed of three layers (input, a hidden, and an output layer). Input neurons just propagate input variables z_j to the next layer. Each neuron in the hidden layer is associated with a kernel function φ_j (usually a Gaussian function) characterized by a center c_j and a width σ_j.

$$\varphi_j(\|z-c_j\|) = \exp\left(-\frac{1}{2}\left(\frac{\|z-c_j\|}{\sigma_j}\right)^2\right). \tag{2}$$

The output layer consists of one neuron which is the target to be predicted. The output function is given by:

$$f(z) = \sum\nolimits_{j=1}^{P} \lambda_j \varphi_j\left(\|z-c_j\|\right). \tag{3}$$

Where P and λ_j are respectively the number and the weight of the radial functions. For more details about artificial neural networks (RBF and MLP), we refer the reader to

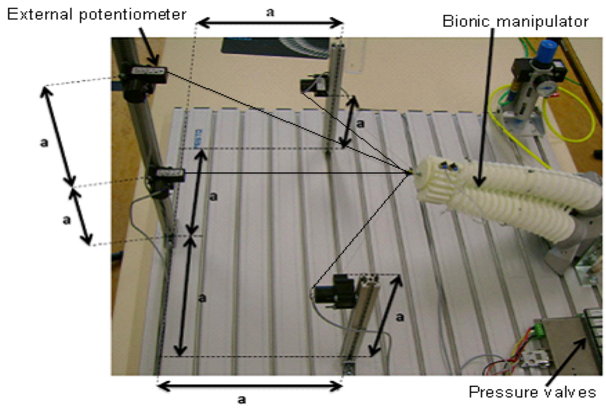

Fig. 4. Trilateration process

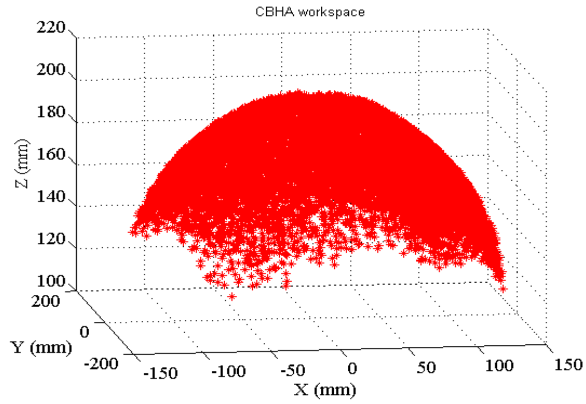

Fig. 5. CBHA workspace

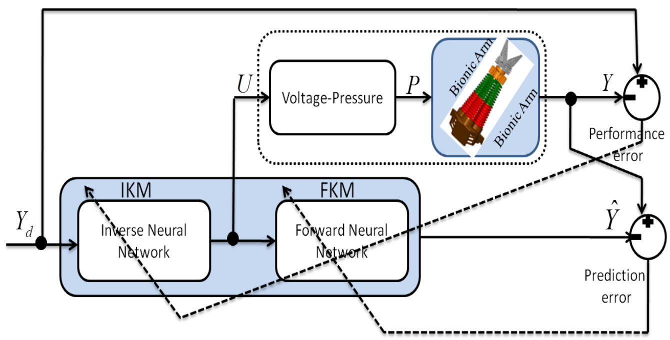

Fig. 6. Inverse Neural Network learning

[23], [24], [25], [22]. We use an online learning rule that makes incremental changes to the parameters of the two neural networks based on the instantaneous value of the cost functional (mean square error MSE) achieved on the training set and defined as follows:

$$MSE = \frac{1}{N_T}\sum_{q=1}^{N_T}\sum_{n=1}^{L}\left(\hat{f}\left(x_q^n\right) - y_q^n\right) \quad (4)$$

Where N_T is the number of training samples; L is the number of output, $\hat{f}\left(x_q^n\right)$ is the value predicted by the model and y_q^n is the measured value.

IV. Experiments and Results

To verify the performances of the proposed approach, the validation of the inverse kinematic model based on MLP and RBF Neural Networks has been implemented in real-time on a bionic arm manipulator. In this section, we first provide a description of the sample data acquisition followed by the application of the distal learning approach for the identification of the inverse kinematic model of the CBHA system. Finally, the learning phase results and the real-time experimental results are described, which illustrate the effectiveness of the proposed approach for the case of the CBHA system.

A. Data acquisition

To build the learning data base, the CBHA's tip position is evaluated experimentally by means of a trilateration system (Fig. 4) developed in [26]. The test bench consists of:

- 1 profiled metallic platform,
- 4 external proportional potentiometers
- 6 wire-potentiometers.

From external potentiometers values and using the simple trigonometry transformation, we can evaluate the CBHA's tip position with an accuracy of about $\pm 0.003m$. The learning base is built as followed: The CBHA posture (wire-potentiometer values) is varying proportionally with the pressure used to control tube-lengths. The pressure in each tube is controlled using internal PID-control. The range of each pressure is $[0; 1.5]$bars. By using a step size of 0.5, each tube can be controlled by one of these values $(0; 0.5; 1; 1.5)$. With 6 controlled inputs, we get a learning base of $4^6 = 4096$ samples. Regardless of the type of exploration that is used to generate the learning base, two examples with the exact same effector pose will rarely be found. Resolving inconsistencies solely based on the samples are therefore hardly possible. The better way to resolve inconsistencies is to consider the example generation method itself or the learning algorithm, instead of considering isolated examples. In this work, we assure that samples with ambiguous solutions to the inverse kinematic mapping are included in the learning data base in order to evaluate the capacity of the DSL scheme to deal with redundancies. The resulting workspace of the CBHA is represented in Fig. 5.

B. Distal learning approach for CBHA inverse kinematic

The DSL approach is used for the approximation of the inverse kinematics model of the CBHA. In DSL approach, the Forward Neural Network (FNN) is first learned to approximate the forward kinematic model of the CBHA (For more details about the forward kinematic model of the CBHA, we refer the reader to [28]). In the second phase, a particular inverse solution is obtained by placing the Inverse Neural Network (INN) and FNN in series, and by replacing the (Voltage-Pressure system + CBHA) by the forward kinematic model that had been trained previously (Fig. 6). At this stage, the composite learning system can be trained by any supervised learning algorithm (back-propagation algorithm in this work). As we have stated in the introduction, for a reliable CBHA positioning, it is not sufficient to control the pressure alone, because friction and hysteresis related to CBHA structure can cause largely different postures when applying the same pressure several times. Hence, the kinetics model (a mapping between the CBHA's tip position and the corresponding pressures) cannot be predicted with accuracy. In this paper, an inverse kinematics model (a mapping between the CBHA's tip position and the corresponding voltages) is developed. $Y_d = [X_d, Y_d, Z_d]^T$ denotes the desired CBHA's tip position, and $Y = [X, Y, Z]^T$ is the real CBHA's tip position. $U = [U_1, U_2, ..., U_6]^T$ is the predicted wire potentiometer voltage, and $\hat{Y} = \left[\hat{X}, \hat{Y}, \hat{Z}\right]^T$ is the predicted CBHA's tip position. The FNN is consisted of 6 inputs (U), and 3 outputs ($\hat{Y}$), while the INN is consisted of 3 inputs (Y_d), and 6 outputs (U). The both neural networks regressors (MLP and RBF) were trained on their corresponding training set. The learning data base is divided as follows; 70% for training set, 15% for validation, and 15% test sets. The training set is used during learning phase and the test set is only used to evaluate the performances of the neuronal models. The validation set is used during the learning phase to avoid the over-fitting. By observing the CBHA's workspace, the latter can be reconstructed by setting more sigmoid (or Gaussian) functions (with variable parameters: centres and widths...) in series. Such that, each point in the CBHA's workspace can be computed using a linear combination of CBHA's length-sensor values, activated by a sigmoid (or Gaussian) function. Hence, the sigmoid function and the Gaussian function are used respectively for MLP activation function and RBF kernel function. In order to minimize the mean square error calculated in the training set, the weight matrices are adjusted by using the back-propagation descent method including the momentum term. In order to empirically select the best model for each regressor, the value of each parameter is varied in a given predefined range according to a grid search over the validation set. We tested the MLP with 2 up to 80 neurons in the hidden layers. Concerning the RBF model, we varied the number of neurons in the hidden layer from 2 to 90 and the width of the Gaussian kernel from 0.01 to 2. A step size of 2 is used for the number of neurons, while it is 0.01 for the width of the Gaussian kernel. For a good generalization of neural network models and to avoid over-fitting, the early-stopping method for training is implemented. The latter requires that after a period of training (epochs) using the training set, the weight matrices of the neural network are fixed, and the neural network is operated in the forward mode using the validation set. The process is repeated until the MSE on the validation set reaches its minimum value. The prediction error ($Y - \hat{Y}$) and the performance error ($Y_d - Y$) are respectively used for forward and inverse neural networks learning.

To select a particular inverse kinematic function, a squared penalty term is added to the objective function of the inverse neural network. The cost functional yields:

$$J = \frac{1}{2}\left(Y_d - Y\right)^T\left(Y_d - Y\right) + \lambda\frac{1}{2}\left\|U\right\|^2 \qquad (5)$$

With $\|.\|$ the Euclidean norm. It has shown that ([29]), the larger the coefficient λ is, the smaller U becomes. The penalty term λ provides a possibility to effectively control the magnitude of U. Thus, a particular inverse solution can be easily selected. In this work, an inverse function which minimizes the Euclidean norm of wire-potentiometer voltages is selected ($\lambda = 0.001$ in the results presented). The assessment of the trained regressors in terms of MSE (Mean Square Error) on the test samples yielded the values reported in Table. I. The first column presents the neural network topologies, while the second column shows MSE obtained in the validation set. The third column depicts the MSE obtained in test set, and the last column presents the different models approximated.

TABLE I

RESULTS ACHIEVED BY EACH NEURAL NETWORK MODEL ON THE TEST SAMPLES

NN topologies	MSE (Validation set)	MSE (Test set)	
MLP (2, 16 neurons)	$2.6e^{-5}$	$3.7e^{-5}$	FKM
RBF ($\sigma = 0.98$,82)	$3.2e^{-5}$	$5.2e^{-5}$	
MLP (2, 16)	$7.6e^{-5}$	$1.1e^{-4}$	IKM
RBF ($\sigma = 0.22$,74)	$2.1e^{-4}$	$4.1e^{-4}$	

In the whole, the results obtained on the test set are satisfying. The MSE is of the order of 10^{-5} for forward model and 10^{-4} for inverse kinematic model.

C. Real time implementation

In addition to the offline validation (test samples), an implementation in real-time has been conducted. In fact, the Robotino XT (mobile platform +CBHA arm) has to move autonomously in dynamic and unstructured environments, while grasping objects. Hence, developed IKM models have to be performed in real-time. However, due to physical limitations of the pneumatic actuators, the CBHA arm needs a certain time to get to mechanical equilibrium (about 5 seconds). Thank to easily implementation (simple matrices manipulating), the neural networks become a suitable tool in such case rather than interactive methods [17], [16]. Thus, by following the control architecture depicted in Fig. 9, we have conducted several real-time experiments, but, due to the limitation of the number of pages, only one experiment is presented in this paper. By using the Robotino XT Matlab toolbox, the set of grasping object trajectories is recorded using the robotino XT set pressure function. The step size is reduced to (0.1) in order to evaluate once more the generalization capacity of the neural network models. The voltages of the length-sensors are recorded. The task is to control the bionic arm with the same recorded length-sensors voltage and compare each time the current robot posture with the recorded postures (vector of potentiometer voltages).

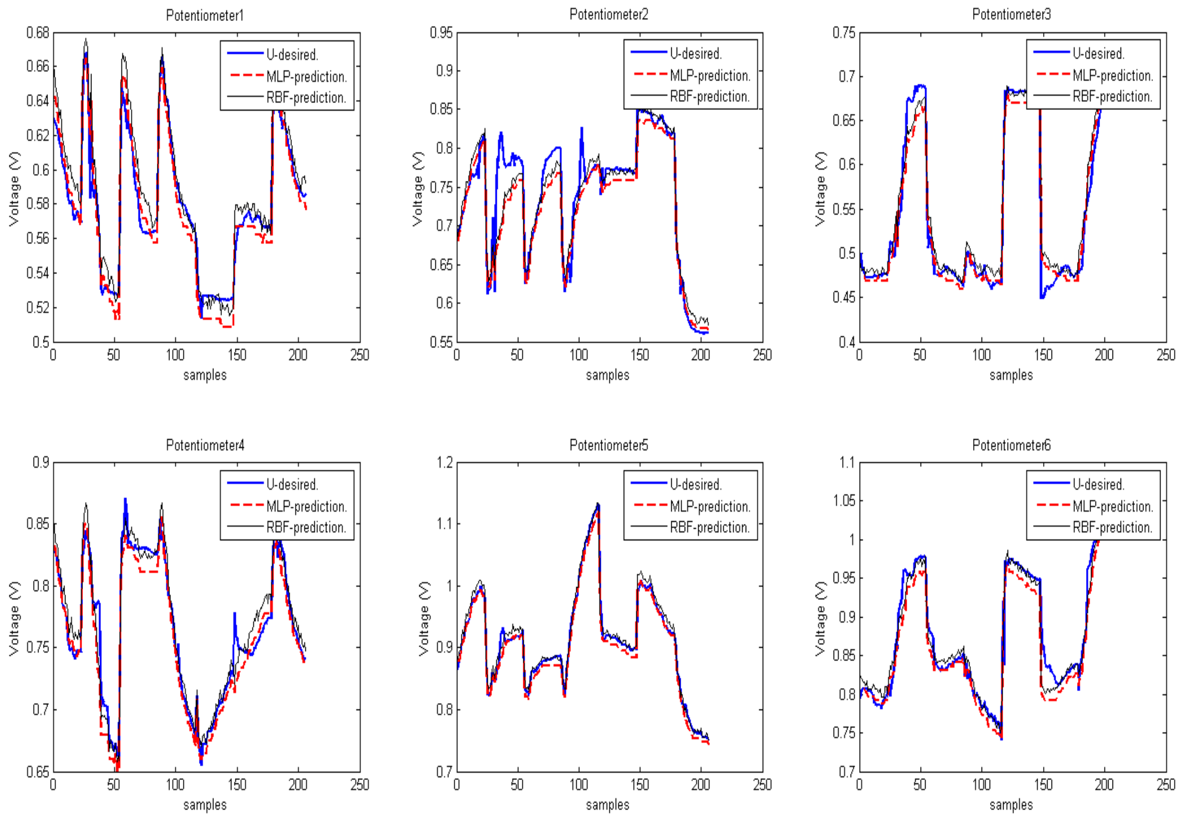

Fig. 7. Prediction voltages provide by neuronal models (MLP, RBF)

Note that, the length-sensors are given in volt, because they are provided by the potentiometer cables. By simple transformation, these voltages are transformed into lengths. As it is depicted in Fig. 9, the set of recorded voltages are used as inputs of the forward kinematic model. The positions generated by the forward kinematic model are used as inputs of the inverse kinematic model. The voltages generated by the inverse kinematic model are transformed into pressures. The set of pressures is applied to bionic arm. Finally, length-sensors provide the corresponding voltages. To transform the voltage to pressure, we use another MLP neural network with 2 hidden layers of 36 neurons. This neural network is not developed in this paper, because the present paper develops the inverse kinematic model. Fig. 7 and Fig. 8 depict respectively the prediction voltages provided by each neuronal model and the associated Euclidean error. From this architecture, we evaluate the performances of the forward model and the inverse model. Note that, the performances of the forward and inverse model in the present control scheme (Fig. 8) are related to those of the transformation Voltage-to-pressure. The implementation is conducted by using a Intel® Core™ i7-2670QM CPU at 2.20GHz.

D. Discussions

This subsection presents the results with discussions. Table 1 shows the performances achieved by neuronal model on the validation and test sets. The results on the test set are satisfying; the MSE is of the order of 10^{-5} for forward model and 10^{-4} for inverse kinematic model. We notice that the estimated and desired potentiometer voltages are close. However, we observe some peaks which are due to undesired and perturbing actuator venting. During a change of the trajectory, the bionic trunk tries to return to its initial configuration. This leads a slight venting of pressure contained in each tube. We obtain an average error of $0.02V$ corresponding to an elongation of about $4mm$. One of the downsides of continuum manipulators morphology is that even minimal changes in the actuated lengths can lead to large changes in the effector position. However, this error remains negligible, because the robot runs in open loop scheme without a controller. The control of the CBHA's effector require the design of an adaptive controller.

The actual work provides two contributions. In the first one, we show that without the assumptions like the constant curvature, the toroidal deformations, the MLP and RBF neural networks can approximate in real-time, the tool centre position with a good degree of accuracy (on the

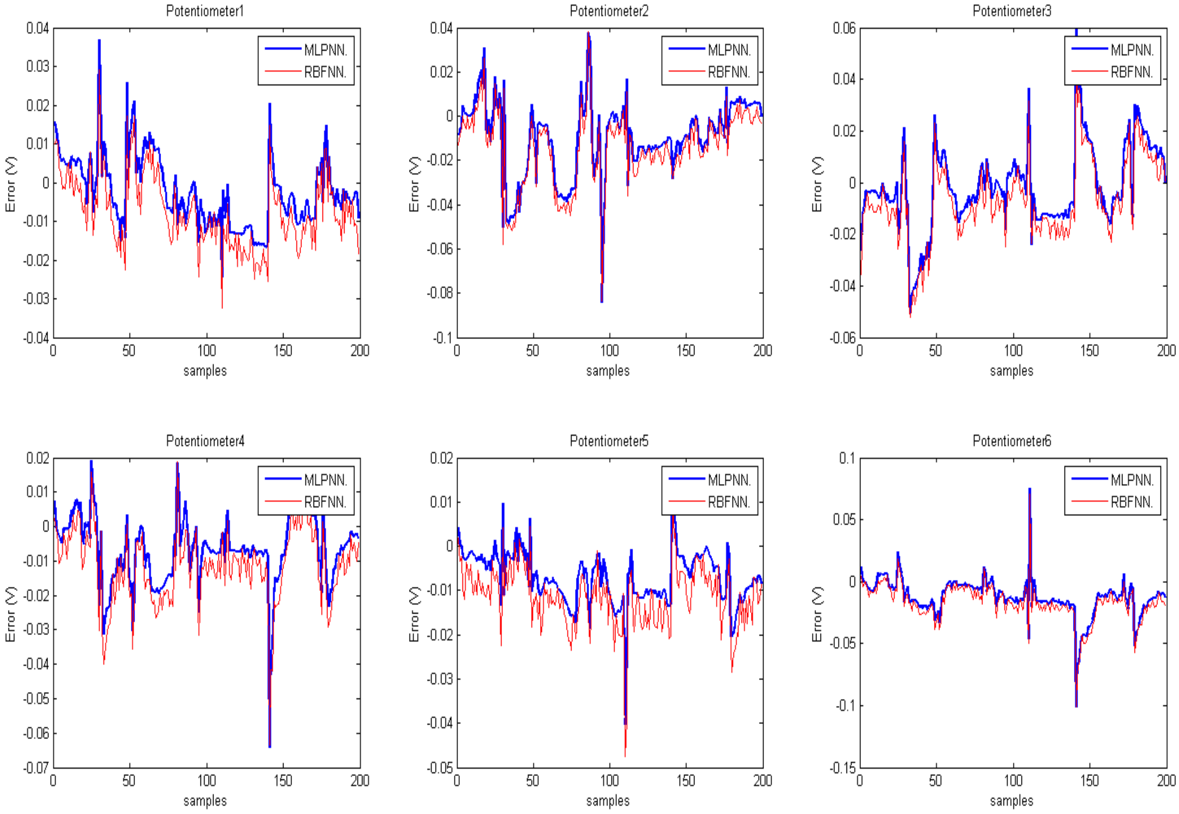

Fig. 8. Imprecisions observed in each neuronal model

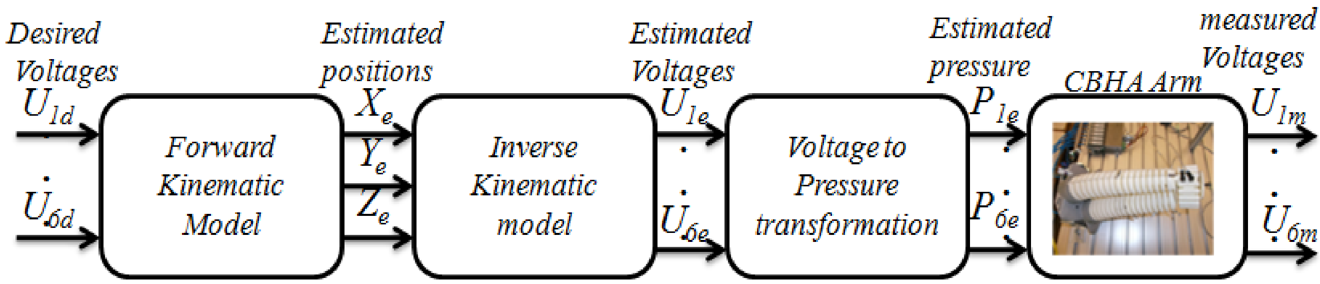

Fig. 9. Real-time implementation control scheme

test set, MSE(MLP)=3.7e-5 and MSE(RBF)=5.2e-5) while dealing with geometry singularities and stretched positions. Thereby, our approach compared to those developed in [3], [5] gives an improvement in the forward model estimation, with consideration of the undesired non-linearities of the bionic trunk. The second contribution is the development of the inverse kinematic model. From a Cartesian coordinates of the centre of the tool, with consideration of the bionic trunk non-linearities (shape memory effect, sensor noises,...), the inverse kinematic neural network model can capture in real-time, the bionic trunk pure elongations with good degree of accuracy (on the test set: MSE(MLP)=1.1e-4 and MSE(RBF)=4.1e-4) while dealing with geometry singularities and stretched positions. In the view of the results obtained, we notice that, the MLPNNs outperform their RBFNNs counterparts in term of performances achieved. However, we do not notice a significant difference in the results obtained in real-time.

If many researchers have attempted to find a solution to the inverse kinematic problem of continuum robots such the case in [3], [4], [5], few contributions have used neural network in distal learning scheme [20], [19]. Where they have used the DSL scheme to control biped robots movements [19], and to build a system that can learn to mimic speech using its own vocal tract [20]. Knowing that in [14], [5], [7] developed computational architectures, involving interactive loops. For our approach, this latter is outperformed using a computational real-time based on neural network weight matrices obtained from a DSL scheme.

V. Conclusion

In this paper, MLP and RBF neural networks with distal learning approach are used to solve the inverse kinematic problem of a CBHA manipulator. In this approach, MLP and RBF neural networks are trained to approximate the forward kinematic model. This model is incorporated into the distal learning scheme to obtain the inverse kinematic suitable model. Numerous experiments have been performed using the CBHA trunk to validate the effectiveness of the proposed neuronal models. It is demonstrated that by using an inverse neural network, it is possible for a given desired target point to track in real-time, a potentiometer voltage vector of the tubes elongation in the presence of uncertainties.

In future work, it is planned to estimate the whole inverse kinematic model of the trunk and its mobile-omnidrive platform, with considering the TCP orientation.

References

[1] P. E. Dupont, J. Lock, B. Itkowitz, and E. Butler, "Design and control of concentric-tube robots," *Robotics, IEEE Transactions on*, vol. 26, no. 2, pp. 209–225, 2010.

[2] J. Ding, K. Xu, R. Goldman, P. Allen, D. Fowler, and N. Simaan, "Design, simulation and evaluation of kinematic alternatives for insertable robotic effectors platforms in single port access surgery," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1053–1058.

[3] M. Rolf and J. J. Steil, "Constant curvature continuum kinematics as fast approximate model for the bionic handling assistant," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 3440–3446.

[4] R. J. Webster and B. A. Jones, "Design and kinematic modeling of constant curvature continuum robots: A review," *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1661–1683, 2010.

[5] B. A. Jones and I. D. Walker, "Kinematics for multisection continuum robots," *Robotics, IEEE Transactions on*, vol. 22, no. 1, pp. 43–55, 2006.

[6] G. Antonelli, "Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems," *Robotics, IEEE Transactions on*, vol. 25, no. 5, pp. 985–994, 2009.

[7] R. V. Mayorga, F. Janabi-sharifi, and A. K. Wong, "A fast approach for the robust trajectory planning of redunant robot manipulators," *Journal of robotic systems*, vol. 12, no. 2, pp. 147–161, 1995.

[8] S. Tzafestas, A. Zagorianos, and T. Pimenides, "A solution to the velocity control of redundant robots," *Mathematics and computers in simulation*, vol. 41, no. 3, pp. 201–208, 1996.

[9] S. R. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," *IEEE Journal of Robotics and Automation*, vol. 17, 2004.

[10] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy," *The International Journal of Robotics Research*, vol. 10, no. 4, pp. 410–425, 1991.

[11] I. S. Godage, D. T. Branson, E. Guglielmino, G. A. Medrano-Cerda, and D. G. Caldwell, "Shape function-based kinematics and dynamics for variable length continuum robotic arms," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 452–457.

[12] S. Yahya, M. Moghavvemi, and H. A. Mohamed, "Geometrical approach of planar hyper-redundant manipulators: Inverse kinematics, path planning and workspace," *Simulation Modelling Practice and Theory*, vol. 19, no. 1, pp. 406–422, 2011.

[13] A. D'Souza, S. Vijayakumar, and S. Schaal, "Learning inverse kinematics," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 1. IEEE, 2001, pp. 298–303.

[14] M. Rolf, "Goal balbing for and effcient bootstrapping of inverse models in high dimensions," Ph.D. dissertation, Bielefeld University, 2012.

[15] M. Rolf, J. J. Steil, and M. Gienger, "Goal babbling permits direct learning of inverse kinematics," *Autonomous Mental Development, IEEE Transactions on*, vol. 2, no. 3, pp. 216–229, 2010.

[16] D. Braganza, D. M. Dawson, I. D. Walker, and N. Nath, "A neural network controller for continuum robots," *Robotics, IEEE Transactions on*, vol. 23, no. 6, pp. 1270–1277, 2007.

[17] M. Giorelli, F. Renda, G. Ferri, and C. Laschi, "A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space," *in proc. IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 5033–5039, 2013.

[18] M. I. Jordan and D. E. Rumelhart, "Forward models: Supervised learning with a distal teacher," *Cognitive science*, vol. 16, no. 3, pp. 307–354, 1992.

[19] S. Stitt and Y. F. Zheng, "Distal learning applied to biped robots," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*. IEEE, 1994, pp. 137–142.

[20] I. S. Howard and M. A. Huckvale, "Training a vocal tract synthesizer to imitate speech using distal supervised learning," in *Proc. SPECOM*, 2005, pp. 159–162.

[21] S. S. Chiddarwar and N. Ramesh Babu, "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, 2010.

[22] C. M. Bishop, *Neural networks for pattern recognition*. Oxford university press, 1995.

[23] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[24] N. Benoudjit and M. Verleysen, "On the kernel widths in radial-basis function networks," *Neural Processing Letters*, vol. 18, no. 2, pp. 139–154, 2003.

[25] T. Xie, H. Yu, and B. Wilamowski, "Comparison between traditional neural networks and radial basis function networks," in *Industrial Electronics (ISIE), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 1194–1199.

[26] C. Escande, P. M. Pathak, R. Merzouki, and V. Coelen, "Modelling of multisection bionic manipulator: Application to robotinoxt," in *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*. IEEE, 2011, pp. 92–97.

[27] M. Kawato and H. Gomi, "A computational model of four regions of the cerebellum based on feedback-error learning," *Biological cybernetics*, vol. 68, no. 2, pp. 95–103, 1992.

[28] A. Melingui, C. Escande, B. Nabil, R. Merzouki, and J. Mbede, "Qualitative approach for forward kinematic modeling of a compact bionic handling assistant trunk," *Submitted to 19th World Congress of the Int. Federation of Aut. Control, Cape Town, South Africa, 24-29 August 2014*, 2013.

[29] W. Wu, H. Shao, and Z. Li, "Convergence of batch bp algorithm with penalty for fnn training," in *Neural Information Processing*. Springer, 2006, pp. 562–569.