22

Control of Robot Interaction Forces Using Evolutionary Techniques

Jose de Gea and Yohannes Kassahun
Robotics Group, University of Bremen
Germany

Frank Kirchner
Robotics Group, University of Bremen,
DFKI (Robotics Innovation Center), Bremen
Germany

1. Introduction

For a robot manipulator to interact safely and human-friendly in an unknown environment, it is necessary to include an interaction control method that reliably adapts the forces exerted on the environment in order to avoid damages both to the environment and to the manipulator itself. A force control method, or strictly speaking, a direct force control method, can be used on those applications where the maximum or the desired force to exert is known beforehand. In some industrial applications the objects to handle or work with are completely known as well as the precise moment on which these contacts are going to happen. In a more general scenario, such as one outside a well-defined robotic workcell or when an industrial robot is used in cooperation with a human, neither the objects nor the time when a contact is ocurring are known.

In such a case, indirect force control methods find their niche. These methods do not seek to control maximum or desired force, but they try to make the manipulator compliant with the object being contacted. The major role in the control loop is given to the positioning but the interaction is also being controlled so as to ensure a safe and clear contact. In case contact interaction forces have exceeded the desired levels, the positioning accuracy will be diminished to account and take care of the (at this moment) most important task: the control of the forces. Impedance control (Hogan (1985)) is one of these indirect force control methods. Its aim is to control the dynamic behaviour of a robot manipulator when contacting the environment, not by controlling the exact contact forces but the properties of the contact, namely, controlling the stiffness and the damping of the interaction. Moreover, the steady-state force can be easily set to a desired maximum value. The main idea is that the impedance control system creates a virtual new impedance for the manipulator, which is being able to interact with the environment as if new mechanical elements had been included in the real manipulator.

First industrial approaches were focused on controlling the force exerted on the environment by a direct force feedback loop. A state-of-the-art review of the 80s is provided in (Whitney (1987)) and the progress during the 90s is described in (Schutter et al. (1997)). In many in-

dustrial applications, where objects are located in a known position in space and where the nature of the object is also familiar, the approach is well-suited since it prevents the robot from damaging the goods. If a detailed model of the environment is not available, the strategy is to follow a motion/force control method obtained by closing a force control loop around a motion control loop (De Schutter & van Brussel (1988)). If controlling the contact force to a desired value is not a requirement, but rather the interest is to achieve a desired dynamic behaviour of the interaction, indirect force control methods find their application. This would be the case when the environment is unknown and the objects to manipulate have non-uniform and/or deformable features. In this strategy, the position error is related to the contact force through mechanical stiffness or with adjustable parameters. This category includes compliance (or stiffness) control (Paul & Shimano (1976)), (Salisbury (1980)) and impedance control (Hogan (1985)), (Caccavale et al. (2005)), (Chiaverini et al. (1999)) and (Lopes & Almeida (2006)).
Several schemes are proposed to regulate the robot-environment contact forces and to deal with model uncertainties. In (Matko et al. (1999)) a model reference adaptive algorithm is proposed to deal with the uncertainty of the parameters that describe the environment. In (Erol et al. (2005)) an artificial neural network-based PI gain scheduling controller is proposed that uses estimated human arm parameters to select appropriate PI gains when adapting forces in robotic rehabilitation applications. In (Jung & Hsia (1998)) a neural network approach is also used to compensate both for the uncertainties in the robot model, the environmental stiffness, and the force sensor noise. Similarly, in (Seraji & Colbaugh (1993)) and (Lu & Meng (1991)) adaptive impedance control schemes are presented to deal with uncertainty of the environmental stiffness as well as uncertainty in the parameters of the dynamical model of the robot or the force measurement. These methods adapt the desired trajectory according to the current scenario, though using cumbersome or unclear methodologies for the selection of impedance parameters. Moreover, some of them might not be applied where the environmental properties are of non-linear nature (Seraji & Colbaugh (1993)).
This chapter aims at describing the use of evolutionary techniques to control the interaction forces between a robot manipulator and the environment. More specifically, the chapter focuses on the design of optimal and robust force-tracking impedance controllers. Current state-of-the-art approaches start the analysis and design of the properties of the impedance controller from a manually-given set of impedance parameters, since no well-defined methodology has been yet presented to obtain them. Neuroevolutionary methods are showing promising results as methods to solve learning tasks, especially those which are stochastic, partially observable, and noisy. Evolution strategies can be also used to perform efficient optimization, as it is the case in CMA-ES (Covariance Matrix Adaptation - Evolution Strategy) (Schwefel (1993)).
Neuroevolution is the combination of neural networks as structure for the controller and an evolutionary strategy which in the simplest case searches for the optimal weights of this neural network. The weights of this neural network represent the policy of the agent, in control engineering terms known as the control law. Consequently, the weights of this neural network bound the space of policies that the network can follow. In more complex strategies, the evolutionary strategy evolves both the weights and the topology of the neural network. In optimal control, one tries to find a controller that provides the best performance with respect to some measure. This measure can be for example the least amount of control signal energy that is necessary to bring the controller's output to zero. Whether in classical optimal control or in neuroevolutionary methods, there is an optimization process involved and we show in

this chapter that neuroevolutionary methods can provide a good alternative to easily design optimal controllers.
In this case study, an impedance controller represented as an artificial neural network (ANN) will be described, whose optimal parameters are obtained in a simple way by means of evolutionary techniques. The controller will regulate the contact forces between a robotic manipulator (a two-link planar arm) and the environment. Furthermore, it will be generalised and provided with force tracking capabilities through an on-line parameter estimator that will dynamically compute the weights of the ANN-based impedance controller based on the current force reference.
The resulting controller presents robustness against uncertainties both on the robot and/or the environmental model. The performance of the controller has been evaluated on a range of experiments using a model of a two-link robotic arm and a non-linear model of the environment. The results evidenced a great performance on force-tracking tasks as well as particular robustness against parametric uncertainties. Finally, the controller was enhanced with a steady-state Kalman filter whose parameters were learned simultaneously with the weights of the ANN. That provided robustness against the measurement noise, especially important in the force measurements.

2. System Description

The system's control architecture (Fig. 1) used for the experiments and implemented under MATLAB is composed of the following submodules: Trajectory Generation module, Impedance Controller (neural network-based controller), Direct and Inverse Kinematics modules, Dynamical Controller module, Two-link Arm Dynamical Model, and Environment model.

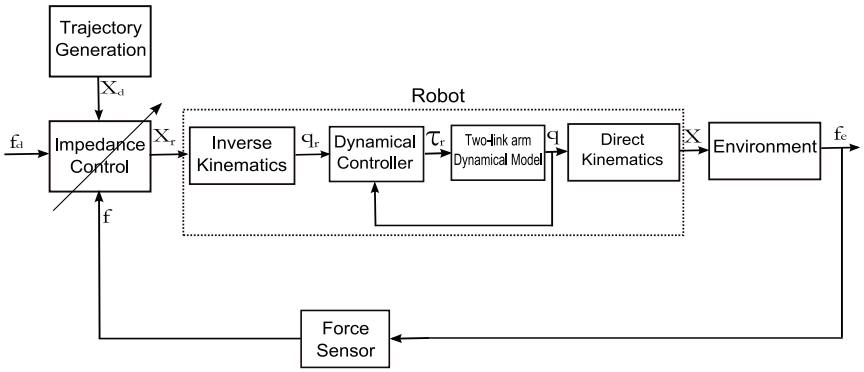


Fig. 1. System's control architecture

2.1 Evolution Strategy

Evolution Strategies (ESs) are a class of Evolutionary Algorithms (EAs) which use nature-inspired concepts like mutation, recombination, and selection applied to a population of individuals containing candidate solutions in order to evolve iteratively better and better solutions. These ESs were introduced by a (back then) unofficial workgroup on Evolution Techniques at the Technical University of Berlin in the late 1960s (Rechenberg (1973)). In contrast to

genetic algorithms, which work with discrete domains, evolution strategies were developed to be used in continuous domains, which make them suitable for continuous-space optimization problems and real-world experiments.

2.1.1 CMA-ES

CMA-ES is an advanced form of evolution strategy (Schwefel (1993)) which can perform efficient optimization even for small population sizes. The individuals are in this algorithm represented by n-dimensional real-valued solution vectors which are altered by recombination and mutation. Mutation is realized by adding a normally distributed random vector with zero mean, where the covariance matrix of the distribution is itself adapted during evolution to improve the search strategy. CMA-ES uses important concepts like *derandomization* and *cumulation*. Derandomization is a deterministic way of altering the mutation distribution such that the probability of reproducing steps in the search space that lead to better individuals is increased. A sigma value represents the standard deviation of the mutation distribution. The extent to which an evolution has converged is indicated by this sigma value (smaller values indicate greater convergence).

2.2 Impedance controller

The classical impedance controller (Fig. 2) is described by Eq. (1):

$$H(s) = \frac{E(s)}{F(s)} = \frac{1}{M_T s^2 + D_T s + K_T} \tag{1}$$

where M_T, D_T and K_T are the inertia, damping, and the stiffness coefficients, respectively, e is the trajectory error, defined as $e = x_d - x_r$ where x_d is the desired trajectory input, and x_r will be the reference trajectory for the next module (the inverse kinematics), corrected depending on the value of the contact force f. The parameters M_T, D_T and K_T will define the dynamic behaviour of the robot that could be compared to the effect of including physical springs and dampers on the robot.

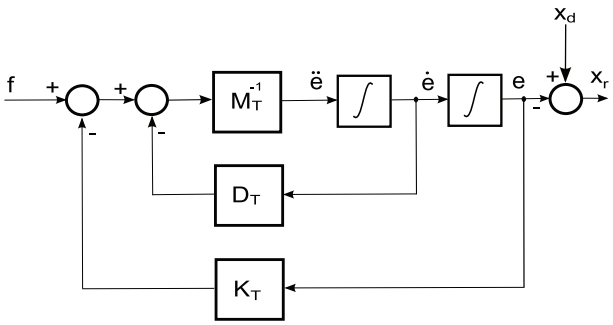


Fig. 2. Classical impedance controller

Starting from (1), the impedance controller can be discretized for its implementation in a computer. Using the bilinear transformation, $H(z) = H(s)\,|_{s=\frac{2}{T}\frac{z-1}{z+1}}$, the discrete version of the impedance controller is obtained.

$$H(z) = \frac{E(z)}{F(z)} = \frac{T^2(z+1)^2}{w_1 z^2 + w_2 z + w_3} \tag{2}$$

where

$$w1 = 4M_T + 2D_T T + K_T T^2 \tag{3}$$
$$w2 = 2K_T T^2 - 8M_T \tag{4}$$
$$w3 = K_T T^2 + 4M_T - 2D_T T \tag{5}$$

From the discretized controller we can generate the difference equation which the filter will be implemented with in the computer:

$$E(n) = \frac{1}{w_1}(F(n)T^2 + 2T^2F(n-1) + T^2F(n-2) - w_2E(n-1) - w_3E(n-2)) \tag{6}$$

Following Eq. (6), it can be clearly seen that the impedance controller can be represented as a neural network as in Figure 3. That means that each classical impedance controller can be implemented as a one-neuron neural network with 5 inputs, 1 output, and only 3 weights.

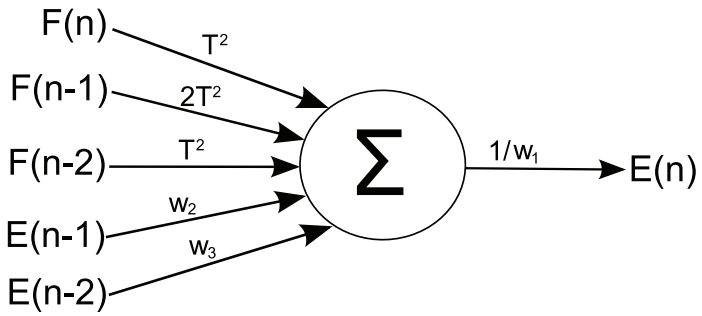


Fig. 3. Neural network representation of the impedance controller

3. Evolving the ANN-based impedance controller

3.1 Single-force reference controller

The weights of the neural network in Fig. 3 are obtained by using the CMA-ES evolutionary technique. In order to do so, the closed-loop system shown in Fig. 4 is used. The ANN-based impedance controller modifies the desired Cartesian position trajectory for the robot (x_d) and creates a new reference trajectory (x_r) based on current sensed forces. The block named *Robot* includes the blocks enclosed under the dotted-line rectangular box in Fig. 1: a dynamical model-based controller that translates the Cartesian positions into the necessary torques for the robot, and forward/inverse kinematics formulations to translate from/to a Joint reference frame to/from a Cartesian frame. The contact forces exerted by the environment onto the robot (f) are fed back to the controller in order to regulate the robot-environment interaction.

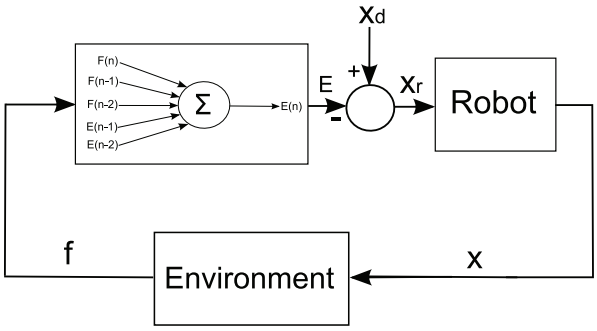


Fig. 4. Close-loop system used to evolve the parameters of the ANN-based controller

The evolutionary algorithm searches for the optimal parameters M_T, D_T, and K_T, and the weights of the neural network are then computed using Eqs. (3), (4), and (5). A fitness function needs to be defined that drives the search and in this case was defined as to minimise the following force error criterion:

$$h = \frac{\sum_{k=1}^{N} \left| f_{ref} - f_k \right|}{N} \tag{7}$$

where f_{ref} is the force reference to be tracked, f_k is the actual force at time step k, and N is the number of samples. A first set of controllers were evolved using only this criterion. By doing that, a controller with fast response is obtained. On the other hand, there are situations where stability on the contact is of outmost priority. To include this additional measure on the evolution of the controller, the following criteria was used for a second series of controllers. The contact stability criterion described in (Surdilovic (1996)) is applied on each individual in order to be selected as final solution. This criterion ensures that the contact with the environment is stable and no oscillations occur at the contact. A significantly overdamped impedance behaviour is required to ensure a stable contact with a stiff environment. If a relative damping coefficient is defined such as

$$\xi_T = \frac{D_T}{2\sqrt{M_T K_T}} \tag{8}$$

and the stiffness ratio is defined as

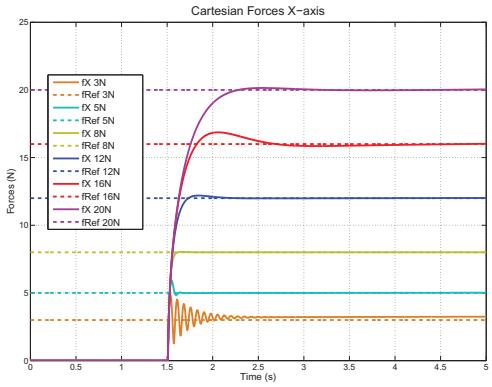
$$\kappa = \frac{K_E}{K_T} \tag{9}$$

where K_E is the stiffness of the environment, then to ensure contact stability we have to satisfy the following criterion:

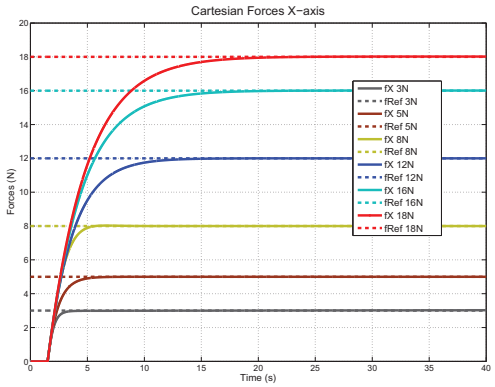
$$\xi_T > 0.5(\sqrt{1+2\kappa} - 1) \tag{10}$$

CMA-ES was initialised to start the search at [0.5,0.5,0.5], initial vector for M_T, D_T, and K_T, respectively. The initial global-step size for CMA-ES was set to $\sigma_{(0)} = 0.5$ and the system was evaluated 1000 times. The population size was chosen according to $\lambda = 4 + \lfloor 3\ln(n) \rfloor$, where n is the number of parameters to optimize and the parent number was chosen to be $\mu = \lfloor \lambda/4 \rfloor$.

A series of single-force reference controllers were evolved under this setup. Each of these controllers obtained as a result of the evolutionary process the optimal weight values for a given force reference. Figure 5(a) shows the results of the controllers evolved without being strict on the contact stability, whereas Figure 5(b) shows the results where the controller has to obey the condition given by Eq. (10). Clearly, the latter offers a safer response at the price of making the system slower.
To summarise, each single-force controller possesses three weights and their optimal values are found for a particular reference force. In a given scenario, the evolved controller is able to control the interaction forces to the desired value and with the desired dynamical characteristics. Provided the current state-of-the-art on selecting the impedance parameters, this solution is a novelty in terms of providing a simple methodology to obtain the optimal impedance parameters for a given task.



(a)



(b)

Fig. 5. Responses of the single-force controllers evolved with CMA-ES for different force reference inputs: (a) without contact stability criterion, (b) with contact stability criterion

3.2 Generalised force tracking controller

In this section, a more general force-tracking controller is designed that is able to adapt to different force references. To attain this goal, an additional block is added to the control scheme: the *Paramater Estimator*, a module that will generate estimations for $\widehat{M}_T$, $\widehat{D}_T$, and $\widehat{K}_T$ based on the current reference force. The complete control scheme can be seen in Fig. 6. The force-to-weights data sets obtained in the previous section (Fig. 5(a)) were used to generate a function that estimates the weights for the controller for any given force reference. By doing that, the input space of the force controller is generalised. Using a 6-th order polynomial as in Eq. (11) for each parameter (M_T, D_T, K_T), a function is generated that estimates the particular parameter for a given input force reference.

$$\widehat{y} = \sum_{i=0}^{n} a_i (f_{ref})^i \tag{11}$$

where $\widehat{y} = \{\widehat{y}_M, \widehat{y}_D, \widehat{y}_K\}$, are the estimation functions for each of the three parameters (M_T, D_T, K_T), respectively, and $n = 6$. The optimal coefficients a_i are again obtained using the CMA-ES evolutionary strategy.

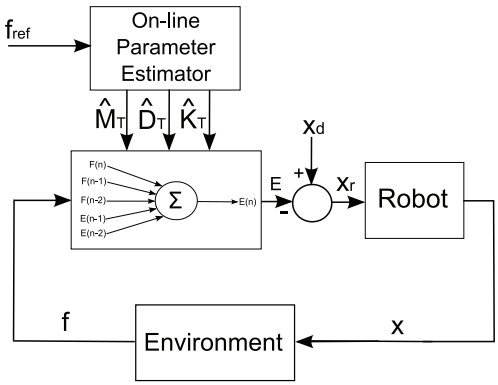


Fig. 6. Structure of the complete control scheme

The procedure is the following: CMA-ES is given the polynomial structure as in Eq. (11) and a set of force-weights training points. These points are the ones depicted in Fig. 7 for each of the parameters (inertia, damping, and stiffness) and relate an input force k with an output parameter. The vector k of input forces was $k = \{3,5,8,12,16,20\}(N)$. Note that for the sake of clarity, damping and stiffness curves have been appropriately scaled in order to be shown on the same graph. The task for the CMA-ES algorithm is to find the parameters of the polynomial that best fit through the corresponding training points. The result is a function that estimates the inertia, damping, and stiffness coefficients for any given reference force. Thus the controller will adapt its weights dynamically as the force reference requirements change. As shown in Fig. 7, the estimated curves precisely pass through the training points (the *measured* force-to-weights relationships). CMA-ES was set to stop the search for the optimal a_i coefficients when the error between the training points and the values of the curves at force k was below $1 \cdot 10^{-10}$.

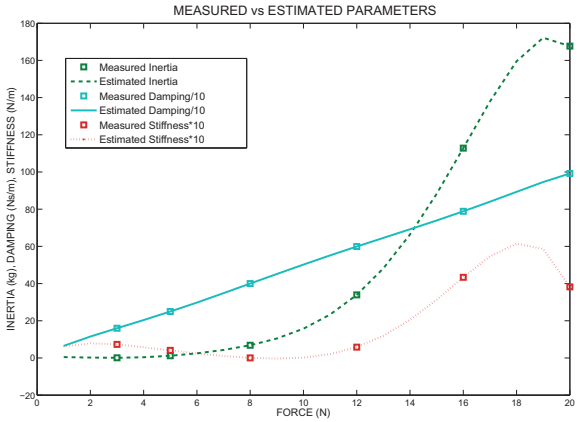


Fig. 7. Estimation functions for each of the parameters of the impedance controller

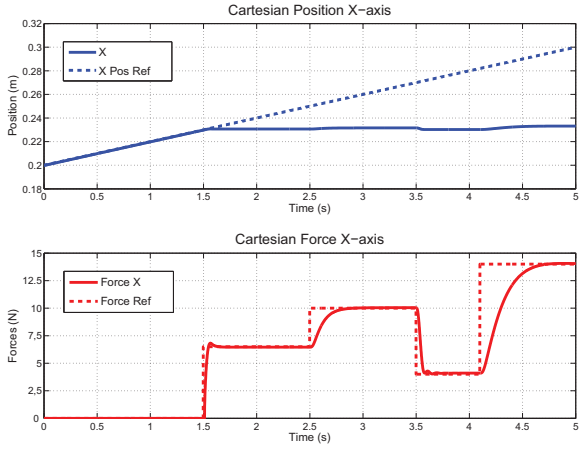
4. Experiments and Results

A series of experiments were conducted using a simulated two-link planar robotic arm (*Two-link Arm Dynamical Model* in Fig. 1) to test the performance of the ANN-based impedance controller. The robot's mechanical model is composed of two revolute joints and two bodies. The module receives torques as inputs and outputs joint angles. The masses are considered to be concentrated at the end of each link to simplify the modelling tasks. The lengths of the body links were set to $a_1 = a_2 = 0.2m$, and their masses to $m_1 = m_2 = 10kg$. A dynamic model-based controller (*Dynamical Controller* in Fig. 1) is used to cancel-out the non-linearities present on the dynamic model of the robot and to decouple the system. After this linearisation and decoupling process, a simple linear PD controller can be used to control the joint positions. The parameters K_p and K_v of the PD controller were set to $K_P = 10000N/m, K_V = 100Nms/rad$. The environment (*Environment* in Fig. 1) is modelled following a non-linear Hunt-Crossley relation (Diolaiti et al. (2005)) instead of the classical linear Kelvin-Voigt model (or spring-like model) since it achieves a better physical consistency and allows to describe the behaviour of both stiff and soft objects. Moreover, it is computationally simple to be computed on-line. The model obeys the following relation:

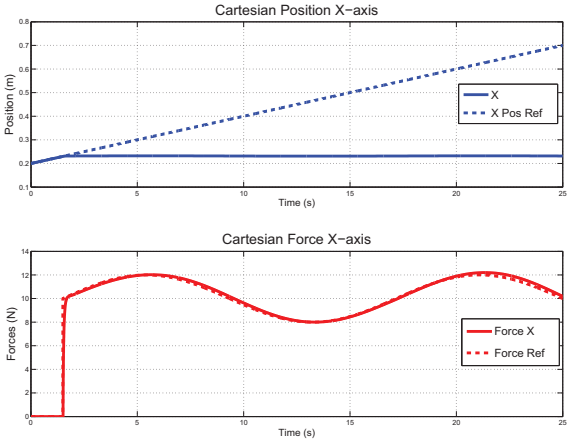
$$F(t) = kx^n(t) + \lambda x^n(t)\dot{x}(t), x \geq 0 \tag{12}$$

where n is a real number that takes into account the geometry of the contact surfaces. For these experiments, the environmental parameters were set to $k = 250N/m$, $n = 0.5$, and $\lambda = 0.0072$. For all the experiments, the robot is commanded to follow a desired position trajectory in the Cartesian space: $x(t) = 0.02t + 0.2$. This desired trajectory will be eventually modified by the impedance controller to create a new reference trajectory that complies with the current force requirements.

4.1 Response to changes on force reference



(a)

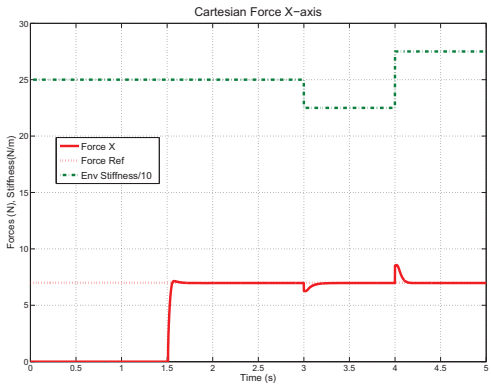


(b)

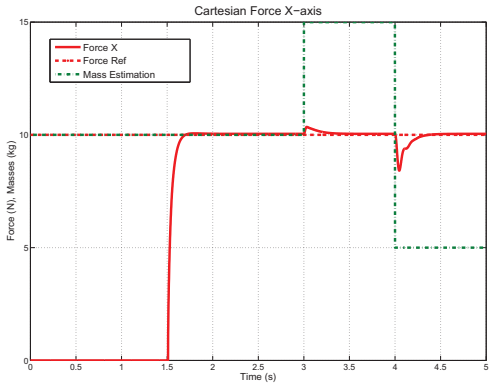
Fig. 8. Robot's response to changes on the reference force: (a) step response, (b) sinusoidal reference

To test the performance of the controller when dealing with force reference changes, two experiments were conducted. A first experiment presents multiple step changes on the reference force for the controller (Fig. 8(a)). The upper part of the figure shows the Cartesian position on the X-axis for the tip of the robot. The robot moves along that axis until it contacts a wall, placed at $x_e = 0.23m$. The bottom part of the figure shows the robot's force re-

sponses. The reference force after contacting the environment is modified and set sequentially to $\{6.5,10,4,14\}(N)$. Note that none of these values were used in the designing phase of the controller (Fig. 7). The robot is able to switch accurately between force references while keeping both a nearly-zero steady-state force error and a stable contact with the surface.
A second experiment was performed where the force reference is a sinusoidal signal (Fig. 8(b)). In this case, a sinusoidal waveform of amplitude 2N is superimposed to the reference of 10N, i.e., the reference force to be tracked is $f_{ref} = 10 + 2sin(\pi t)\ (N)$. As it can be seen on the bottom part of the figure, the robot tracks the sinusoidal force reference accurately.



(a)



(b)

Fig. 9. Robustness of the controller: (a) change on stiffness of the environment, (b) change on mass of the robot's links

4.2 Robustness against uncertainties

The following experiments aimed at testing the robustness of the controller for changes on both the environment and the robot's model. A robust controller has to be able to cope with uncertainties, especially those related to uncertainties in the parameters of the models.

4.2.1 Variations on the environmental stiffness

The first experiment modifies the stiffness of the environment during a stable contact. As previously stated, the stiffness of the environment in the Hunt-Crossley model was set to $k = 250N/m$. For this experiment, the stiffness is modified as $k_m = 250 \pm 10\%k$ (N/m). Figure 9(a) shows the behaviour of the controller in consequence of the changes on the environmental stiffness. The robot is able to recover and set back to the original force reference of 7N in a short time, despite of the fact that the stiffness is kept constant to a value below or over the nomimal.

4.2.2 Variations on the robot's model

A second experiment was conducted where the masses of the links of the robot were modified during a contact situation. As previously stated, the masses of the robot's links were set to $m = m_1 = m_2 = 10kg$. For this experiment, the estimated masses used on the dynamical model of the robot are modified as $m_m = 10 \pm 50\%m$ (kg). Figure 9(b) shows the behaviour of the controller to the changes on links' masses. The robot is again able to recover and set back to the original force reference in a short time, despite of the fact that the masses are kept constant to a value below or over the nomimal.

4.3 Robustness against noise

A final series of experiments aimed at testing the controller against the inherently-present measurement noise, especially important in the force measurement. The purpose of these experiments is twofold: on the one hand, to test whether the algorithm is able to find a solution using real-world noisy signals and, on the other hand, to enhance the evolved controller with a zero-delay noise filter using a Kalman filter. The filter is included on the evolution process in order to generate a one-step solution that takes into account noisy signals. In other words, the optimal parameters of the Kalman filter will be searched using the CMA-ES evolution strategy while simultaneously the controller's parameters are learned.

The Kalman filter (Kalman (1960)) estimates the state of a linear dynamical system that is perturbed by a gaussian noise. Formally, the filter addresses a general problem of estimating the true state $x \in \mathcal{R}^n$ of a discrete linear time system governed by

$$x_k = A_k x_{k-1} + B_k u_{k-1} + w_{k-1}, \tag{13}$$

where A_k is an $n \times n$ state transition matrix, B_k is an $n \times m$ control input model matrix, $u_k \in \mathcal{R}^m$ is the control vector, and w_k is the process noise which is assumed to be drawn from a zero-mean multivariate normal distribution with covariance matrix Q_k of size $n \times n$. The measurment (observation) $z_k \in \mathcal{R}^l$ of the true state is modelled by

$$z_k = C_k x_k + v_k, \tag{14}$$

where C_k is an $l \times n$ matrix representing the measurement model and v_k is the measurment noise which is again assumed to be drawn from a zero mean multivariate normal distribution with covariance matrix R_k of size $l \times l$.

The Kalman filter recursively estimates the current state based on the current measurement and the estimate from the previous state. The filter has basically two distinct phases: *predict* and *update*. Let $P_{k|k-1}$ and $\hat{x}_{k|k-1}$ be the *a priori* estimate of the error covariance matrix and the true state at timestep k, respectively, and $P_{k|k}$ and $\hat{x}_{k|k}$ be the *a posteriori* estimate of the error covariance matrix and the true state at timestep k, respectively. The filter starts with initial estimates for the true state $\hat{x}_{k-1|k-1}$ and the error covariance matrix $P_{k-1|k-1}$, and then

repeatedly executes its *predict* and *update* phase routines. Refer to (Welch & Bishop (1995)) for a more detailed introduction to the Kalman filter.

4.3.1 The Steady-state Kalman Filter with Constant Velocity Model

The Kalman filter used in our implementation is a particular type of the general Kalman filter in which a constant velocity model is assumed. The constant velocity model is usually used in tracking applications (Kalata (1992); Bar-Shalom et al. (2001); Perez-Vidal et al. (2009)) and is also known as an $\alpha\beta$ filter. Since we assume that the system's velocity does not change dramatically, we are able to assume a constant velocity model. The steady-state version of the Kalman filter is used in cases where the time required to compute the algorithm is an important constraint. For a given system, one can let the Kalman filter run for several cycles and record the Kalman gains K in steady state. These will be constant, so the computation can easily be sped up by always using these constants instead of updating K each cycle (which requires a matrix inversion computation). The equations that describe the steady-state Kalman filter are:

$$\hat{x}_{k|k-1} = A \cdot \hat{x}_{k-1|k-1} \quad (15)$$

$$\tilde{y}_k = z_k - C \cdot \hat{x}_{k|k-1} \quad (16)$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K \cdot \tilde{y}_k \quad (17)$$

where $\hat{x}_{k|k-1}$ represents the estimate of x at time k given observations up to and including time $k-1$. z_k is the measurement at time k, A is the state transition matrix, C is the output array and K is the steady-state Kalman gain. Expression (16) computes the innovation factor that allows the predictions to be updated after new measurements have been obtained. Given the assumption of a constant velocity model, the filter will choose two weighting coefficients (α and β) that will weight the differences between predictions and new measurements when updating the current prediction to find a new estimate. To better illuminate this, consider the classical tracking equations for the $\alpha\beta$ filter:

$$x_p(k) = x_s(k-1) + v_s(k-1)T \quad (18)$$

$$v_p(k) = v_s(k-1) \quad (19)$$

$$x_s(k) = x_p(k) + \alpha(z_k - x_p(k)) \quad (20)$$

$$v_s(k) = v_s(k-1) + (\beta/T)(z_k - x_p(k)) \quad (21)$$

where $x_p(k)$ and $v_p(k)$ are the predicted position and velocity at time k, $x_s(k)$ and $v_s(k)$ are the smoothed position and velocity at time k, T is the sampling time, and α and β are the weighting coefficients. After calculating $x_p(k)$ and $v_p(k)$ (Eqs. 18 and 19), the calculation of the smoothed parameters only requires the proper selection of values for α and β. The optimal values for α and β have been derived by (Kalata (1992)), and depend on the assumed variance of both measurement and process noises (σ_v and σ_w):

$$\gamma = \frac{T^2 \cdot \sigma_v}{\sigma_w} \tag{22}$$

$$r = \frac{4 + \gamma - \sqrt{8 \cdot \gamma + \gamma^2}}{4} \tag{23}$$

$$\alpha = 1 - r^2 \tag{24}$$

$$\beta = 2 \cdot (1 - r)^2 \tag{25}$$

$$K = \begin{bmatrix} \alpha \\ \beta / T \end{bmatrix} \tag{26}$$

The state transition matrix A is initialized the with a constant velocity model:

$$A = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{27}$$

and the output array C is

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{28}$$

where the '1' in the first column indicates that we have measurements from the force, and the '0' in the second column indicates that we have no information about the force change with respect to time (derivative of the force).

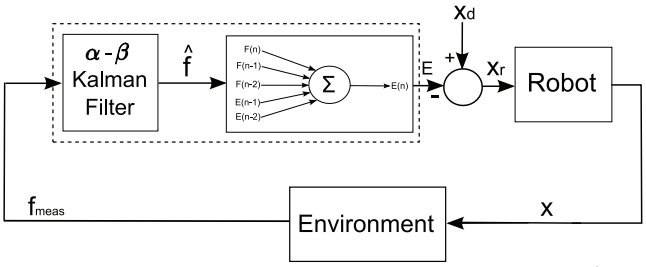
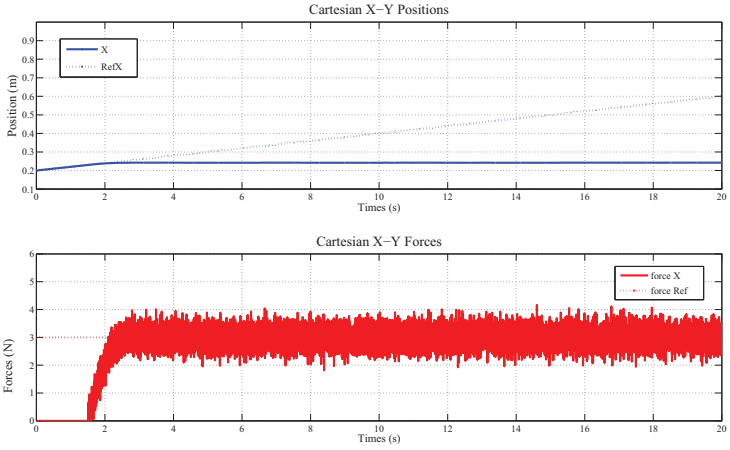


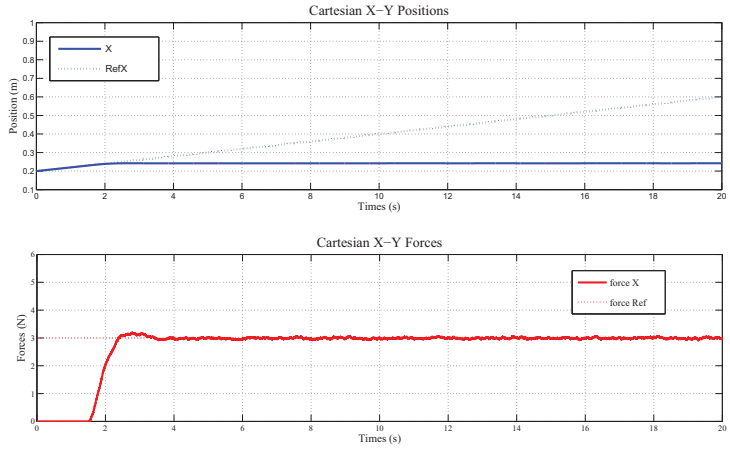
Fig. 10. The $\alpha - \beta$ Kalman filter is used to estimate the sensor value $\hat{f}$ from the measured (noisy) value f_{meas}. The parameters of the blocks enclosed under the dotted lines are obtained using evolution strategies

In our experiment, we use one Kalman filter for the force measurement. The measurement noise that is introduced to the system is a Gaussian signal with zero mean and standard deviation $\sigma_v = 10^{-1}$. This noise is added to the input f_{meas} (the force measurement) of the Kalman filter depicted in Figure 10. In this experiment, the same neural network structure was used as in the previous experiments, i.e. the one-neuron feedforward neural network with 5 inputs, 1 output, and 3 weights. Additionally, the optimization of the Kalman filter was incorporated into the evolutionary process, where optimal values for the parameters σ_v and σ_w were searched for using CMA-ES (along with the weights for the neural network). Because the problem is simulated, the standard deviation of the measurement noise we are introducing is known and thus the initial value for σ_v in the Kalman filter can be set to this value. In the case of a real system, however, a set of real measurements could have been collected, the mean and standard deviation of the data set calculated, and the standard deviation used as the value

for σ_v. In the case of process noise, manual tuning is typically used due to the complexity of determining the value of the noise. The Kalman filter, however, usually performs well with only a rough estimate of σ_w.



(a)



(b)

Fig. 11. Robustness against noise: (a) evolving the controller without a Kalman filter, (b) weights of the ANN-based controller and the Kalman filter evolved simultaneously

Figure 11 shows the results of the experiments with noisy signals. Figure 11(a) depicts the case of learning to track a specific force reference with a highly-noisy force measurement. As it can be seen, the algorithm is able to, despite the noise, learn a proper solution in order to

achieve the reference force. However, the controller would be useless in a practical scenario since the robot would oscillate at high frequency around the contact point. In order to provide a compact solution, the Kalman filter presented previously is included in the evolution process. By doing that, we obtain a solution in one step: both the weights of the neural network and the parameters of the Kalman filter are obtained simultaneously, without requiring of a pre-processing of the measurement data. Figure 11(b) shows the response obtained with the system depicted on Figure 10. The robot is able to reach the targetted reference force and, at the same time, imperceptible noise remains on the force response of the robot, i.e. no oscillations occur on the contact.

5. Conclusions

The work presented describes the design of an ANN-based impedance controller by using evolutionary techniques. The impedance controller is first discretized and represented as a neural network. The use of evolutionary techniques provides a simple methodology to evolve the controller requiring only the definition of a proper performance criteria to be optimised. Currently, unclear or cumbersome methodologies are found to select impedance parameters. The proposed approach obtains optimal parameters given a task to perform. Besides, it is shown how the classical impedance controller can be described as a single-neuron neural network with 5 inputs, 3 weights, and 1 output. Since the weights of the neural network bound the policy space of the controller, and in this case they are only three, the space of the possible inputs is unique. To generalise the controller for any given force reference input, an on-line estimator has been designed that estimates the weights for the current force reference. Using the values of a series of single-force controllers, the parameters of a polynomial are obtained that estimate the proper neural network weights for the current scenario. The resulting controller is able to track a great range of force reference inputs, a quality that is not intrinsically present on a classical impedance controller. Moreover, the robustness of the controller is demonstrated by modifying both the robot and the environmental model parameters. The controller is able to set back to the current reference force after abrupt changes on the environmental stiffness, even when it is constantly kept to values 10% below or over the nominal one. Similarly, abrupt changes on the estimated masses of the robot links of up to 50% of the nominal value are absorbed by the controller, which is able to keep track of the current reference force. Finally, the controller is enhanced with a Kalman filter to improve the controller's robustness against the measurement noise. Both the controller and the parameters of the Kalman filter are evolved simultaneously, thus providing a one-step solution which does not require a pre-processing of the measurement data used to learn the solution.

6. References

Bar-Shalom, Y., Li, X. & Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation*, John Wiley & Sons, New York, USA.

Caccavale, F., Natale, C., Siciliano, B. & Villani, L. (2005). Integration for the next generation - embedding force control into industrial robots, *IEEE Robotics and Automation Magazine*, pp. 53–64.

Chiaverini, S., Siciliano, B. & Villani, L. (1999). A survey of robot interaction control schemes with experimental comparison, *Mechatronics, IEEE/ASME Transactions on* **4**(3): 273–285.

De Schutter, J. & van Brussel, H. (1988). Compliant robot motion II: A control approach based on external control loops, *Int. J. Rob. Res.* **7**(4): 18–33.

Diolaiti, N., Melchiorri, C. & Stramigioli, S. (2005). Contact impedance estimation for robotic systemss, *IEEE Transactions on Robotics*, pp. 925–935.

Erol, D., Mallapragada, V. & Sarkar, N. (2005). Adaptable force control in robotic rehabilitation, *IEEE Int. Workshop on Robots and Human Interactive Communication*, pp. 649–654.

Hogan, N. (1985). Impedance control-an approach to manipulation, *Journal of Dynamics Systems, Measurement, and Control-Transactions of the ASME* **107**: 8–16.

Jung, S. & Hsia, T. (1998). Neural network impedance force control of robot manipulator, *IEEE Transactions on Industrial Electronics*, pp. 451–461.

Kalata, P. R. (1992). Alpha-beta target tracking systems: A survey, *American Control Conference*, pp. 832–836.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems, *Transactions of the ASME-Journal of Basic Engineering* **Series D**: 35–45.

Lopes, A. M. & Almeida, F. (2006). Acceleration based force-impedance control, *MIC'06: Proceedings of the 25th IASTED international conference on Modeling, indentification, and control*, ACTA Press, Anaheim, CA, USA, pp. 73–81.

Lu, W.-S. & Meng, Q.-H. (1991). Impedance control with adaptation for robotic manipulations, *IEEE Transactions on Robotics and Automation*, pp. 408 – 415.

Matko, D., Kamnik, R. & Badj, T. (1999). Adaptive impedance force control of an industrial manipulator, *Proc. IEEE Int. Symposium on Industrial Electronics*, pp. 129–133.

Paul, R. & Shimano, B. (1976). Compliance and control, *Proceedings of the 1976 Joint Automatic Control Conference*, pp. 694–699.

Perez-Vidal, C., Gracia, L., Garcia, N. & Cervera, E. (2009). Visual control of robots with delayed images, *Advanced Robotics* **23**: 725–745.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*, Frommann-Holzboog, Stuttgart.

Salisbury, J. K. (1980). Active stiffness control of a manipulator in cartesian coordinates, Vol. 19, pp. 95–100.

Schutter, J. D., Bruyninckx, H., Zhu, W.-H. & Spong, M. W. (1997). Force control: A bird's eye view, *In B. Siciliano (Ed.), Control Problems in Robotics and Automation: Future Directions*, Springer Verlag, pp. 1–17.

Schwefel, H.-P. P. (1993). *Evolution and Optimum Seeking: The Sixth Generation*, John Wiley & Sons, Inc., New York, NY, USA.

Seraji, H. & Colbaugh, R. (1993). Adaptive force-based impedance control, *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1537 – 1544.

Surdilovic, D. (1996). Contact stability issues in position based impedance control: Theory and experiments, *Proc. IEEE ICRA96* pp. 1675–1680.

Welch, G. & Bishop, G. (1995). An introduction to the Kalman filter, *Technical Report TR95-041*, University of North Carolina at Chapel Hill, Department of Computer Science, USA.

Whitney, D. (1987). Historical perspective and the state-of-the art in robot force control, *International Journal of Robotics Research* **6**: 3–14.

Factory Automation

Edited by Javier Silvestre-Blanes

ISBN 978-953-307-024-7
Hard cover, 602 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

Factory automation has evolved significantly in the last few decades, and is today a complex, interdisciplinary, scientific area. In this book a selection of papers on topics related to factory automation is presented, covering a broad spectrum, so that the reader may become familiar with the various fields, and also study them in more depth where required. Within various chapters in this book, special attention is given to distributed applications and their use of networks, since it is one of the most relevant subjects in the evolution of factory automation. Different Medium Access Control and networks are analyzed, while Ethernet and Wireless networks are looked at in more detail, since they are among the hottest topics in recent research. Another important subject is everything concerning the increase in the complexity of factory automation, and the need for flexibility and interoperability. Finally the use of multi-agent systems, advanced control, formal methods, or the application in this field of RFID, are additional examples of the ideas and disciplines that experts around the world have analyzed in their work.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821