2013 IEEE International Conference on Robotics and Automation (ICRA)
Karlsruhe, Germany, May 6-10, 2013

Evolutionary Co-Optimization of Control and System Parameters for a Resonating Robot Arm

Jurren Pen*, Wouter Caarls†, Martijn Wisse† and Robert Babuška*
*Delft Center for Systems and Control, Delft University of Technology
Mekelweg 2, 2628 CD Delft, The Netherlands. Email: r.babuska@tudelft.nl
†Delft Biorobotics Lab, Delft University of Technology
Mekelweg 2, 2628 CD Delft, The Netherlands. Email: {w.caarls, m.wisse}@tudelft.nl

***Abstract*—In this paper we simultaneously optimize the parameters describing the morphology of a robot arm and the parameters of its nonlinear controller. A novel concept of a pick-and-place robot arm is considered, which is called the *resonating arm* (RA). It uses a nonlinear spring mechanism to generate pick-and-place motions without the need for powerful actuators. This improves energy efficiency, cost and weight of the robot arm. Because of the complex interactions of the spring mechanism and the controller, we use evolutionary co-optimization to optimize the RA system as a whole. The results reveal that evolutionary co-optimization yields near optimal solutions for a 1 degree of freedom (1-DOF) RA, which require 43% less torque than the solution found through a separate optimization of the system and the control parameters. In case of a 2-DOF RA, evolutionary co-optimization resulted in credible solutions as well, but with less consistency.**

I. Introduction

The Resonating Arm (RA, [1]) is a novel concept for a robot arm with low-power actuators designed for pick-and-place tasks. These tasks are in general highly repetitive, which favors the use of robots above human workers. Therefore robot arms are already widely used to perform pick-and-place tasks in numerous industries (e.g. the food handling industry). These robot arms are typically equipped with high-power actuators to meet the fast handling speeds required by the industry, making them inefficient and unsafe. The resonating arm is being developed to reduce this need for heavy, powerful actuators while being fast enough to be used in an industrial environment.

In order to reduce the actuator power, the resonating arm uses a spring mechanism to move the arm between the pick and place areas. An electric actuator is only used to overcome friction and to perform precise placement within those areas. The spring mechanism has many parameters, making it hard to optimize by hand. Furthermore, the performance can only be measured when the arm is properly controlled, and the control parameters depend on the mechanism. In this paper, we therefore use evolutionary co-optimization to simultaneously optimize the control and system parameters.

We will start by reviewing related work in the areas of efficient robotic arms and evolutionary optimization in Section II. Next, we will explain the resonating arm mechanism and the optimization methods in Sections III and IV, respectively. The results are presented in section V and discussed in Section VI. Finally, we conclude with Section VII.

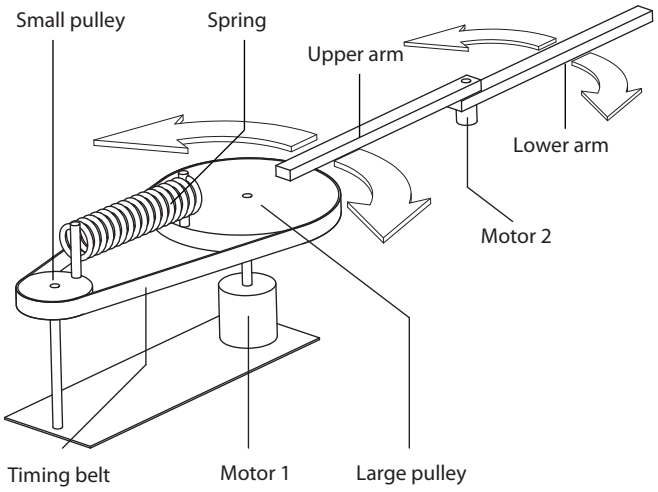


Fig. 1. A drawing of the controlled mechanical system, called resonating arm, which will be optimized in this research. An important part of the resonating arm is its spring mechanism, represented by the two pulleys, the spring and the timing belt connecting them. Furthermore the systems consists of an upper and lower arm with two motors to control both degrees of freedom. From [1].

II. Related work

The idea of exploiting the natural motions of a system has been applied to manipulators before. Williamson investigated control strategies for natural oscillating arms [2]. However, those oscillations were not created mechanically, and therefore did not decrease the required actuator power. The work most strongly related to the resonating arm we consider in this paper is that by Babitsky and others [3] who researched mechanically resonant robotic systems and designed mechanisms for those robots. The drawback of these mechanisms is that they lock into place at pre-determined positions. For practical applicability, there is a need for freedom to deviate from the pre-determined locking positions.

It appears that evolutionary co-optimization [4], [5] has mainly attracted the attention of researchers in the field of autonomous robots, while its benefits are generally applicable to the development of other types of robots, such as manipulators. The software Darwin2K by Leger [6] is one of the few applications concerning the optimization of robot manipulators. This software is able to synthesize the robot morphology

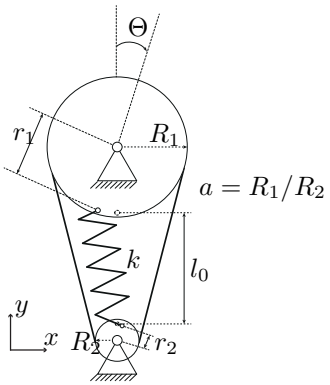


Fig. 2. The spring mechanism driving the resonating arm. This mechanism consists of two pulleys interconnected by a timing belt and a spring. The most important variables describing this mechanism are the angular position of the large pulley Θ and the morphological parameters l_0 r_1, r_2, a and k, respectively the original length of the spring, the radii at which the spring ends are attached to the large and small pulley, the ratio between the outer radii of the pulleys (R_1/R_2) and the spring constant.

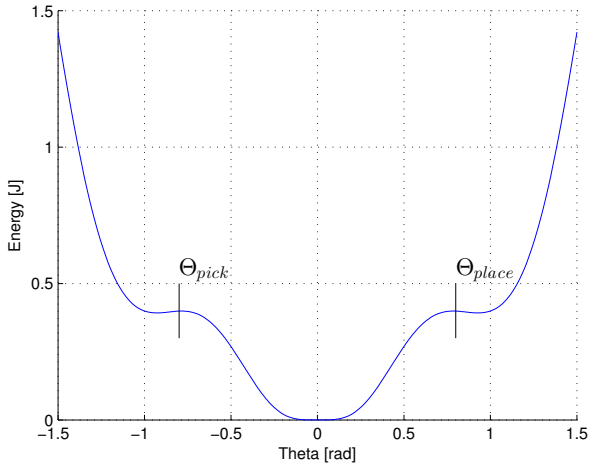


Fig. 3. Potential energy as a function of angle Θ of the upper arm for morphological values of the prototype. The spring mechanism in the resonating arm yields the two flat plateaus in the potential energy curve near the desired pick angle $\Theta_{pick} = -0.8$ rad and desired place angle $\Theta_{place} = 0.8$ rad at which energy is stored in the system, but the derivative of the potential energy curve is close to zero. This implies that at these angles the torque applied by the spring mechanism is close to zero. This is ideal when the system has to be kept at these positions using a low-power actuator, with a limited maximum torque.

and optimize the system and control parameters. In order to keep the complexity of the problem manageable, simple local control strategies (i.e. PID) are applied. Unfortunately, these simple local controllers also limit the search for an optimal solution. Therefore this research concerns the optimization of more complex and global controllers.

III. Resonating robot arm

The reduction of actuation power without a loss of handling speed can only be achieved by designing a clever mechanism that does not rely on actuation power to generate the high accelerations needed; the RA is such a mechanism.

The basic layout of the RA is equal to a SCARA type robot. It has a parallel-axis joint layout in which the arms can move in the X-Y plane but are rigid in the Z-direction. Figure 1 shows a drawing of the RA with the most important parts of the system indicated. The core of the RA is a spring mechanism that consists of a spring attached to two pulleys with different diameters, connected by a timing belt which creates a ratio between the angular displacement of both pulleys. The upper arm is connected to the larger pulley, which is driven by a motor for positioning. The lower arm is connected to the upper arm and actuated by a second motor which is placed at the pivot point between both arms. The end effector is not shown, but would be located at the end of the lower arm.

In Figure 2 the spring mechanism itself is depicted together with its defining system parameters. The mechanism creates a nonlinear relation between spring elongation and angular position of the upper arm. Figure 3 plots the relation between the potential energy and the angle. The morphological values used are the ones used in the design of the first prototype [1], namely: $r_1 = 0.1$ m, $r_2 = 0.02$ m, $a = 5$, $l_0 = 0.1$ m and $k = 150$ N/m. Also the desired pick angle $\Theta_{pick} = -0.8$ rad and desired place angle $\Theta_{place} = 0.8$ rad of the prototype are indicated.

Two specific characteristics of the spring mechanism can be observed in Figure 3. First we see the expected behavior that the potential energy increases when the system moves away from the rest position at Θ = 0 due to the spring elongation. But more important are the two plateaus in the potential energy curve at which the change in potential energy is relatively small compared to angles outside these plateaus. At these angles the derivative of potential energy E_p over the angle Θ (which is equal to the spring mechanisms torque τ_s) is close to zero. This enables the system to efficiently stay at the pick or place positions until a product is fully grasped or placed and then use the stored energy to quickly accelerate the system in the direction of the other pick or place position.

IV. Methods

We use fuzzy control and evolutionary optimization to find the best controller and morphological parameters of the spring mechanism with regard to the maximum actuator torque that is needed to stay at the pick and place locations (torque-to-stay), as well as move between them (torque-to-move).

A. Fuzzy control

In order to control the resonating arm system we require a nonlinear feedback controller. Nonlinear since the RA is a highly nonlinear system for which linear control approaches will not be able to generate an optimal control. And feedback is required to control the system from different starting positions. We use fuzzy control [7] because it makes it much easier to understand the control behavior and to distill general control rules. This also makes it possible to incorporate prior knowledge into the controller to initialize the optimization and increase the speed of convergence.

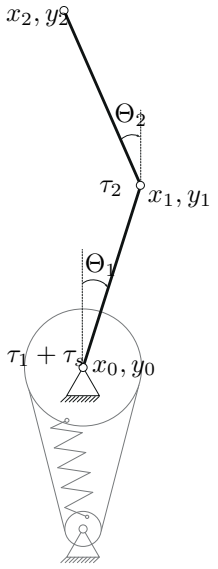


Fig. 4. Top view of the resonating arm model with the two degrees of freedom of the system Θ_1 and Θ_2 and the torques acting on the joints with τ_1 and τ_2 the actuator torques and τ_s the torque caused by the spring mechanism.

We use fuzzy variants of the linear proportional-derivative (PD) controllers used in many control applications, which determine their control output $\mathbf{u}$ based on the error $\mathbf{e}$ and the error derivative $\dot{\mathbf{e}}$

$$\mathbf{u} = f(\mathbf{x}) \tag{1}$$

with

$$\mathbf{x} = [\mathbf{e}\ \dot{\mathbf{e}}]^T \tag{2}$$

where in fuzzy control the function $f(\cdot)$ is determined by the fuzzy rules and membership functions, which are to be optimized in conjunction with the mechanism parameters.

The number of inputs for the controllers is equal to four, defining the errors between the desired angular positions and speeds of both arms as

$$\mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \Theta_{1d} - \Theta_1 \\ \Theta_{2d} - \Theta_2 \end{bmatrix} \tag{3}$$

$$\dot{\mathbf{e}} = \begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \end{bmatrix} = \begin{bmatrix} \dot{\Theta}_{1d} - \dot{\Theta}_1 \\ \dot{\Theta}_{2d} - \dot{\Theta}_2 \end{bmatrix} \tag{4}$$

where Θ_{1d}, Θ_{2d}, $\dot{\Theta}_{1d}$ and $\dot{\Theta}_{2d}$ represent the desired angles and their derivatives, respectively.

The output $\mathbf{u}$ is defined as

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{5}$$

with τ_1 and τ_2 the control torques acting on the upper and lower arm, see Figure 4.

The $\mathbf{e}$ and $\dot{\mathbf{e}}$ variables are scaled by a factor between 1/3 and 3 to bring the real inputs into the normalized domain used by the fuzzy controller (between -1 and 1), and to provide the optimization algorithm with a variable that has a global influence on the relative widths of all membership functions. The same holds for the outputs which are also scaled by a factor between 1/3 and 3.

The positions of the membership functions of the fuzzy controller are variable and need to be optimized by the EA. The membership functions are multidimensional Gaussian functions with zero covariance. Gaussian functions are used since it has been shown that these yield a good approximation [8].

Each membership function is associated with a fuzzy rule that outputs a desired torque vector $\boldsymbol{\omega}$. The outputs of all rules are combined using the weighted average method:

$$f(\mathbf{x}) = \frac{\sum_{j=1}^{N} \boldsymbol{\omega}_j \left(\Pi_{i=1}^{k} \mu_{A_j^i}(x_i) \right)}{\sum_{j=1}^{N} \left(\Pi_{i=1}^{k} \mu_{A_j^i}(x_i) \right)} \tag{6}$$

with

$$\mu_{A_j^i}(x_i) = \exp\left(-\frac{(c_j^i - x_i)^2}{2(\sigma_j^i)^2} \right) \tag{7}$$

where the centres c_j^i, widths σ_j^i and consequent torque vectors $\boldsymbol{\omega}_j$ are to be optimized. Note that each rule has its own c and σ.

B. Evolutionary optimization

The problem considered in this paper is characterized by its high dimensionality, complexity and constraints causing a large and nonlinear search space with many peaks and discontinuities, making it ideally suited for evolutionary optimization. We analyze and compare the performance of two state-of-the-art algorithms, CMA-ES and CoSyNE.

1) CMA-ES: CMA-ES [9] was developed with the idea that the evolutionary operators of selection, recombination and mutation implicitly define a distribution from which the next generation is sampled. Therefore an evolutionary search in its simplest form can be described by a three step process of sampling, evaluation and an update of the distribution parameters.

The sample distribution used in CMA-ES is a multi-variate normal distribution $\mathcal{N}(\mathbf{m}, \mathbf{C})$ that is uniquely determined by its mean value $\mathbf{m} \in \mathbb{R}^n$ and its symmetric positive definite $n \times n$ covariance matrix $\mathbf{C}$. This distribution can be geometrically interpreted as an iso-density ellipsoid, which shape is determined by the covariance matrix $\mathbf{C}$. Its position is determined by the mean value $\mathbf{m}$ around which the distribution is symmetric.

The mean value $\mathbf{m}$ of this distribution is updated after each generation by selecting the best points from the evaluated population. The weighted average of these points is then used to determine the updated mean value $\mathbf{m}$, where the weight of each point is determined by its ranking position. Also the covariance matrix $\mathbf{C}$ is updated by finding a covariance matrix that fits the distribution of the best points of the new populations best. Since CMA-ES uses a non-elitist selection only the newly created solutions are used for these updates.

The step-size determines how far new solutions are sampled from the mean value. This step-size is also automatically updated in CMA-ES by using a cumulative step-size adaptation, which increases the step-size when the mean vector $\mathbf{m}$ is updated in the expected direction based on the previous update directions. When this is not the case the step-size is reduced in order to facilitate a finer search.

2) CoSyNe: CoSyNE [10] belongs to the family of cooperative co-evolutionary algorithms which are inspired by natural ecosystems where several species cooperate for their survival. In this class large optimization problems are decomposed into sub components, which are subject to local evolutionary processes. This decomposition and evolution of sub solutions is expected to create EAs which are capable of optimizing problems with high dimensionality. Since the co-optimization of control and system parameters often results in high dimensional search spaces, the cooperative co-evolutionary approach is expected to be beneficial to optimize both the control and system parameters of the RA.

The cooperating species or sub populations in the CoSyNE algorithm are the different sub components of a complete solution. Since the fitness function cannot determine a fitness score for a single sub component, complete solutions are assembled using one member from each species. The fitness scores at the species level is defined in terms of the fitness of the complete solutions in which the species members participated. The evolution of each sub population is then handled independently by a standard evolutionary process of selection, reproduction, mutation and replacement [22]. To explain these steps more clearly a pseudo-code of the CoSyNE algorithm is presented in Table I.

TABLE I
COSYNE ALGORITHM

1: $P \leftarrow$ `initialize`$(P^1, P^2, P^3, \ldots, P^n)$ {initialize all n subpopulation with m variables}
2: **repeat**
3: **for** $j = 1$ to m **do**
4: $P_j \leftarrow (P_j^1 P_j^2 P_j^3 \ldots P_j^n)$
5: `evaluate`(P_j)
6: **end for**
7: $P_{\text{parents}} \leftarrow$ `SelectBestQuarter`(P)
8: $P_{\text{offspring}} \leftarrow$ `Offspring`$(P^1_{\text{parents}}, P^2_{\text{parents}}, \ldots, P^n_{\text{parents}})$
9: $P_{\text{new}} \leftarrow$ `ReplaceWorstQuarter`$(P, P_{\text{offspring}})$
10: $P \leftarrow$ `Permute`$(P^1_{\text{new}}, P^2_{\text{new}}, P^3_{\text{new}}, \ldots, P^n_{\text{new}})$
11: **until** end criteria

First (line 1), a population P consisting of n sub populations P_i $i = 1, \ldots, n$ is initialized randomly, where n is the number of variables needed to define a complete solution. Each sub population is initialized to contain m real numbers chosen from a uniform probability distribution in the interval $[0, 1]$. Starting with these initial sub populations the CoSyNE algorithm loops through a sequence of generations until the stopping criteria are met (lines 2-11). Each generation starts by constructing a complete solution by combining one individual from each sub population. In CoSyNE this is done by taking the jth row of the population matrix P (line 4). The complete solution is now evaluated and in this way a fitness score is determined for all m combinations (line 5). The fitness score of an individual is set equal to the fitness score of the complete solution it was part of. A quarter of the best performing solutions is selected as parents and used to create offspring by applying the variation operators mutation and crossover (line 7-8). The newly created individuals are now used to replace the worst performing individuals (line 9). In order to incorporate co-evolution into the algorithm the sub populations are permuted uniformly so that each individual forms part of a potentially different solution in the next generation (line 10).

The user-defined parameters of both the CMA-ES and CoSyNE algorithms are given in Table II

TABLE II
THE USER DEFINED PARAMETERS OF THE COSYNE AND CMA-ES ALGORITHM.

Parameter	CoSyNE	CMA-ES
population size	40	40
crossover type	uniform	-
mutation	Gaussian noise ($\sigma = 0.15$)	-
mutation probability	0.3	-

C. Stepwise optimization

To get a baseline against which to compare the evolutionary optimization results, we used a stepwise optimization procedure in which the system parameters and controller are optimized sequentially. To decouple the two problems, we optimized the system parameters for minimal torque-to-stay (flat area around Θ_{place} in Figure 3). The spring constant k and moment of ineria I_0 do not influence this plateau, and were set at the maximum and minimum bounds, respectively. The rationale is that a higher spring constant and lower inertia will decrease the required torque-to-move. It is the maximum of the torque-to-stay and torque-to-move that is optimized by the evolutionary algorithm. Table III summarizes the results, comparing them to the prototype.

TABLE III
STEPWISE OPTIMIZED PARAMETERS DESCRIBING THE DYNAMIC BEHAVIOR OF THE 1-DOF RESONATING ARM.

Symbol	Prototype value	Optim. bounds	Optim. value	Units
r_1	0.1	$[0.05 - 0.20]$	0.20	m
r_2	0.02	$[0.01 - 0.04]$	0.0389	m
a	5	$[2.5 - 10]$	5.1153	m/m
l_0	0.1	$[0.05 - 0.20]$	0.05	m
k	150	$[0 - 200]$	200	N/m
I_0	0.16	$[0.16 - 0.32]$	0.16	Nm^2
Torque-to-stay	0.0817		0.0218	Nm

To optimize the controller with respect to the required torque-to-move, we applied an iterative approach in which a minimum-time problem is solved [11] for a decreasing set of control torque bounds. When the control bounds are set too high the minimal-time problem will yield an optimal time below the time constraint of 1 second. By iteratively decreasing the bounds until the optimal time is equal to 1 second the minimal-maximum-torque problem can be solved.

The stepwise optimization was only performed for the 1-DOF system (without the lower arm), because the minimum-time optimal control problem could not be solved analytically for the 2-DOF case.

V. Results

The goal of the optimization is to find the fuzzy control and system parameters that minimize the maximum actuator torque required to perform certain pick-and-place tasks. This actuator torque should allow the fuzzy controller to move the system to a predefined region around the place position in the state space and it should be large enough to stay at all positions in this region (place range). For the 1-DOF system, we compare the results to those found using the conventional stepwise optimization. Unless otherwise indicted, all results were generated using the CMA-ES algorithm.

A. One degree of freedom

The fitness function used for the optimization of the 1-DOF RA is shown in Table IV. Before the fitness is determined using the maximum used torque during the simulation, two checks are present to make sure that the solutions can still be given a score when the simulation failed or if the system did not converge to the desired state after 1 second.

TABLE IV
Fitness function of the 1-DOF RA optimization

if simulation fails at some time t_{fail} **then**
  fitness $\leftarrow \left(1 - \frac{t_{fail}}{t_{simulation}}\right) + 2$
else if error ϵ at $t_{simulation} >$ allowed error **then**
  fitness $\leftarrow \left(1 - \frac{1}{1+\epsilon^2}\right) + 1$
else
  $\tau_m \leftarrow$ max. abs. torque to move from pick position to place range
  $\tau_p \leftarrow$ max. abs. torque to stay in place range
  fitness $\leftarrow \left(1 - \frac{1}{1+\max(\tau_p, \tau_m)}\right)$
end if

The evolutionary algorithms optimized the 6 system parameters and 38 control parameters (7 membership functions) simultaneously. A first comparison between the co-optimized solution and the stepwise-optimized solution is presented in Figure 5, where the dashed lines correspond with stepwise-optimized solution and the solid lines correspond with the co-optimized solution. From this figure we have to conclude that the co-optimized solution is able to use a lower maximum torque than the stepwise-optimized solution. In fact, when compared to the stepwise-optimized solution, the co-optimized solution is able to reduce the required maximum absolute control torque by 43%, from 1.338 Nm to 0.768 Nm.

Figure 6 shows how this reduction is possible by plotting the potential energy and torque curves for both the stepwise-optimized solution and the co-optimized solution. Although the co-optimized solution uses the nonlinearity of the spring mechanism to minimize the torque at the place range $[\Theta_a, \Theta_b]$, it did not minimize these torques to a minimum. Instead the maximum torque at the place range is close to the torque used to move the system between the pick-and-place positions.

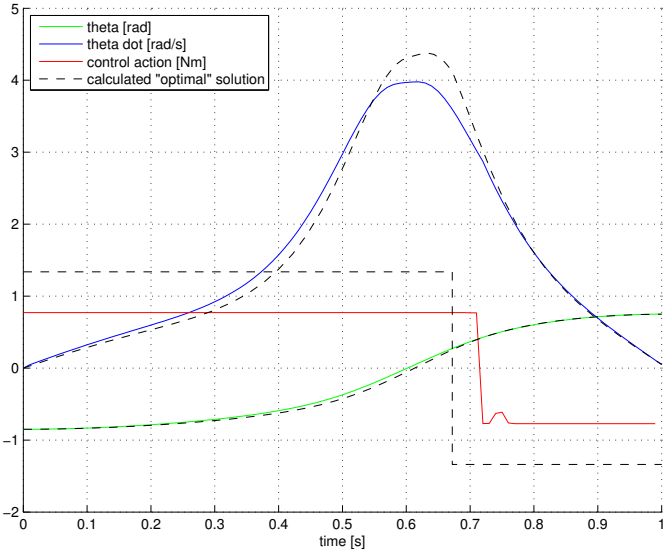
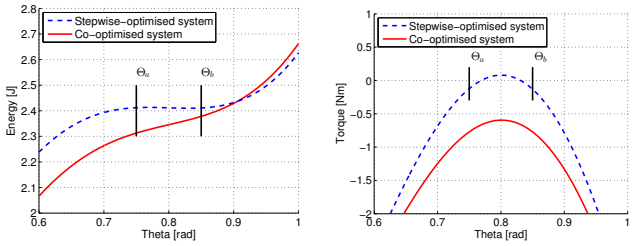


Fig. 5. The control output and state response of the solution found by the evolutionary algorithm starting from $[\Theta_{\text{init}}, \dot{\Theta}_{\text{init}}] = [-0.85, 0]$. The dashed lines represent the optimal control solution belonging to the stepwise optimized system. It can be seen that the solution found by the evolutionary algorithm is able to use a maximum torque, which is less than the solution found in the stepwise optimization, from this result we can conclude that the evolutionary algorithm found better system parameter values, which allow a lower torque.

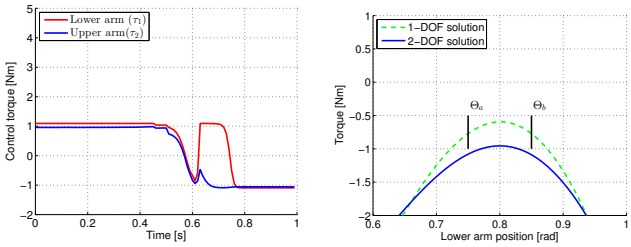


(a) Potential energy curve (b) Torque curve

Fig. 6. The potential energy and torque curves of the spring mechanism with parameter values obtained from a stepwise optimization and an evolutionary optimization. The evolutionary algorithm seems to prefer a bigger torque at the pick-and-place ranges than the minimized torque found in the stepwise optimization.

B. Two degrees of freedom

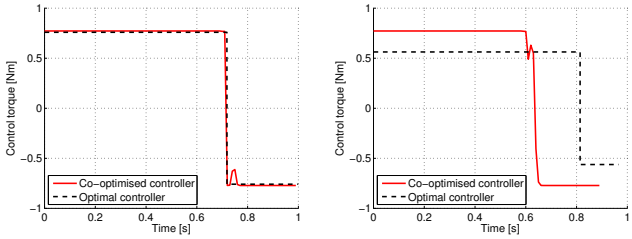
The fitness function used for this optimization is similar to the one used in the 1-DOF RA optimization, which was presented in Table IV. The only changes are the fact that the error now consists of four values and that two torques are used to move the system from the initial positions to the desired positions. The maximum of the two torques-to-move and the torque-to-stay is used to determine the fitness score, this means that the overall maximum torque used in the system is minimized and no distinction is made between the upper and lower arm actuators. The evolved controller had 17 membership functions, resulting in a total of 182 optimization parameters.



(a) The control outputs of the actuators from an initial state of $\Theta_1 = -0.85$, $\dot{\Theta}_1 = 0$, $\Theta_2 = 0$ and $\dot{\Theta}_2 = 0$.

(b) The torque curves at the place location corresponding to the best solutions found for the 1-DOF system and the 2-DOF system.

Fig. 7. Comparison of torque-to-stay and torque-to-move for the 2-DOF system



(a) Initial state: $\Theta = -0.85$, $\dot{\Theta} = 0$

(b) Initial state: $\Theta = -0.75$, $\dot{\Theta} = 0$

Fig. 10. The control output for the 1-DOF RA starting from the outer two initial states. The dashed lines represent the optimal control solution derived in Section IV-C. From the plots we can see that the evolutionary algorithm converged to a control solution at which an equal torque is used for both initial states, this torque equals the minimal torque needed to bring the system from the furthest state to the desired state in the desired time as can be seen in (a). In (b) the used torque is higher than the optimal maximal torque and therefore the system converges faster.

While in the 2-DOF case we do not have a baseline to compare against, we can still assess the performance of the optimization by looking at Figure 7(a), which plots the control output for one of the initial states. The maximum absolute torque used during the pick-and-place movement is 1.1 Nm and the maximum absolute torque-to-stay at the pick-and-place locations is also approximately 1 Nm as shown in Figure 7(b). The fact that these torques have the same size indicates that the optimization did indeed optimize the system and control parameters to an internally consistent solution, by finding the right balance between the torque-to-move and the torque-to-stay.

C. Comparison of CMA-ES and CoSyNe

In Figure 8 the convergence of both algorithms is presented when optimizing the 1-DOF RA with both types of controllers. The dotted lines indicate that the average fitness does not equal the average maximum torque used since some of the simulations obtained a penalty for not converging to the desired states. When the convergence line is solid the values correspond with the average maximum torque of all optimization runs.

Figure 9 shows the same data for the 2-DOF case. Although the consistency of the solutions found after 10 000 generations is considerably lower than the 1-DOF case, it can also be seen that after 5000 generations already 50% of the optimization runs yielded in solutions with a fitness score close to the final fitness score.

For all experiments the best results were found by the CMA-ES algorithm and it therefore appears to be the most effective algorithm for the optimization of nonlinear controllers and system parameters.

VI. Discussion

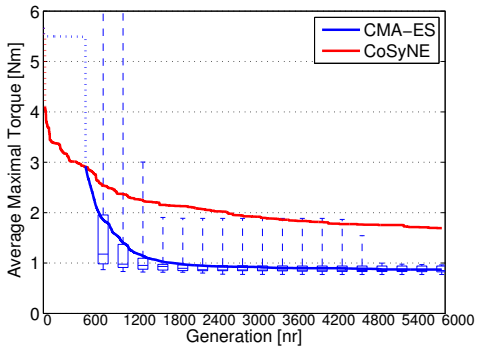
A. One degree of freedom

It was shown that the co-optimized solution was able to decrease the overall maximum torque by increasing the torques at the place range. The co-optimized solution uses the increased spring mechanism torques to accelerate the arm directly from the start and therefore less actuation torque is needed to perform the movement in time. This is a valuable insight, which was not discovered before the evolutionary co-optimization was applied.

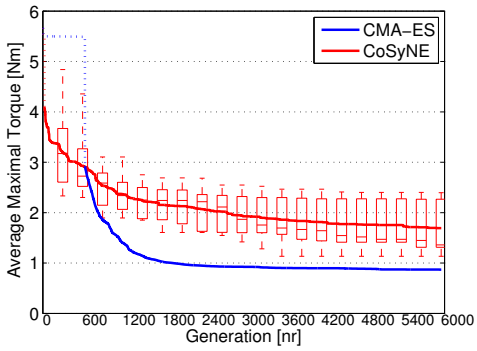
Eventually, this insight could also have been found through a further analysis of the mechanism. However, even when this was the case, this would result in an optimization problem that is difficult to solve in a stepwise optimization. This is caused by the fact that the optimal torque at the pick-and-place positions has to be equal to the optimal torque used to move between the pick-and-place positions. Therefore the dependency of the system parameters on the control solution is even stronger, which would have to be solved through a cumbersome iterative process of system optimization and control optimization.

Another effect of the necessity to balance the torques can be seen by looking at the control solutions for different initial states. In Figure 10 the actual control outputs and the optimal control outputs are shown. It can be seen that for both initial states the same maximum torque is used, even though a lower torque would have been able to bring the system to the desired state within the time constraint of 1 second. Moreover, the maximum torque used for all initial states is about equal to the optimal torque needed to bring the system from the furthest state to the desired state.

The reason why the EA only optimized the control for the furthest initial state can be found by analysing the optimization goal stated at the beginning of this experiment. This goal demands a minimized actuator torque that allows the controller to move the system to a predefined region in the state space and is at the same time large enough to keep the system at all positions in this predefined region. The initial state furthest from the desired region requires the highest actuator torque and the EA will reduce this torque-to-move by increasing the torque needed to stay at the place range. The minimal actuator torque for that initial state is found when the torque-to-move and the torque-to-stay have the same size. This torque-to-stay will now dominate the minimal actuation torque of the other initial states, and therefore a further minimization of their torque-to-move does not influence the optimization outcome. After optimizing the fuzzy controller this resulted in equal

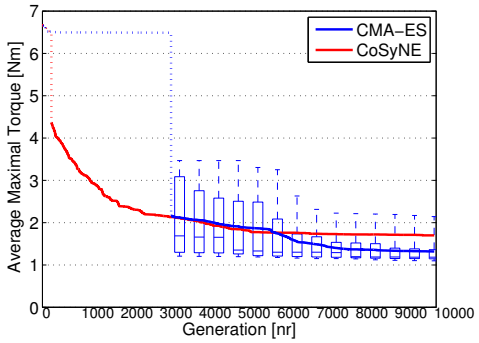


(a) CMA-ES

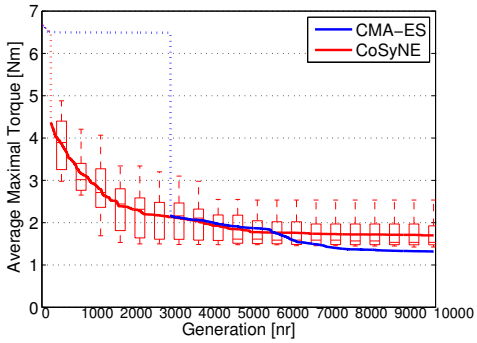


(b) CoSyNE

Fig. 8. The convergence of the 1-DOF RA experiment presented in box plots. The plotted line represents the median of 30 runs and the box represents the lower quartile and the upper quartile, the whiskers show the smallest and largest sample. Although CoSyNE shows a faster convergence initially, the CMA-ES algorithm is able to converge to a better solution.



(a) CMA-ES



(b) CoSyNE

Fig. 9. The convergence of the 2-DOF RA experiment presented in box plots. The plotted line represents the median of 30 runs and the box represents the lower quartile and the upper quartile, the whiskers show the smallest and largest sample. As in the 1-DOF experiments, CoSyNE shows a faster convergence initially, while the CMA-ES algorithm is able to converge to a better solution.

torques-to-move for all initial states.

From a practical point of view this outcome is acceptable since the highest minimal maximum torque for one of the initial positions will set the requirements for the actuators used in the robot arm.

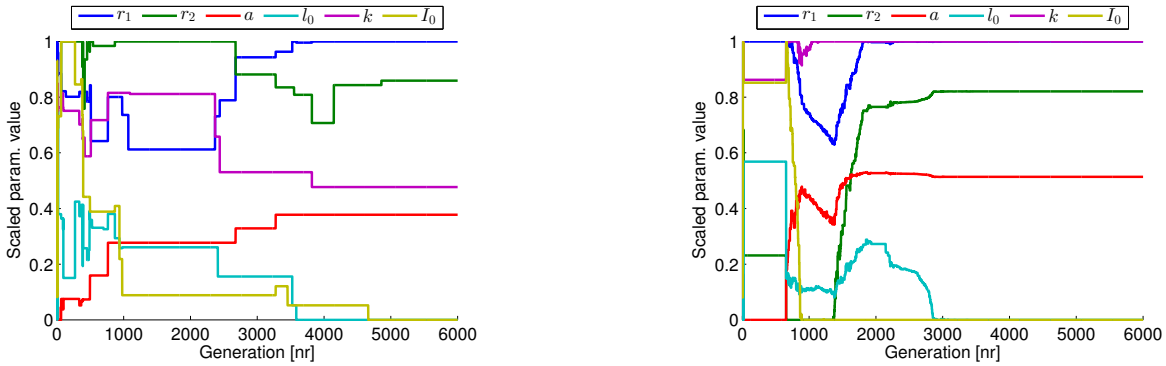
B. Two degrees of freedom

When looking at Figure 7(b), the torque curve corresponding to the 2-DOF system is plotted and compared with that of the 1-DOF system. Here it can be seen that the level of potential energy stored in the optimized 2-DOF RA is considerably lower than the potential energy stored in the optimized 1-DOF RA. This may be the result of a sub optimal convergence, but it is also possible that a higher level of potential energy will not allow a lower torque-to-move. This latter explanation finds support in the findings of Plooij using the original prototype [1], who observed that the sudden acceleration of the upper arm caused by the spring mechanism increases the torque required to control the lower arm.

C. Comparison of CMA-ES and CoSyNe

The best solutions for both the 1-DOF and 2-DOF system were obtained using the CMA-ES algorithm. The convergence of the CMA-ES algorithm to lower torques can be explained by the fact that the CMA-ES algorithm seems to be better in optimizing the system parameters. In Figure 11 the normalized parameter values are plotted and one can see that the CMA-ES algorithm is constantly finding better systems while the CoSyNE algorithm changes the best found system parameters less often. This is assumed to be the result of the adaptive step size used in the CMA-ES algorithm, which allows the algorithm to change the system parameters more gradually.

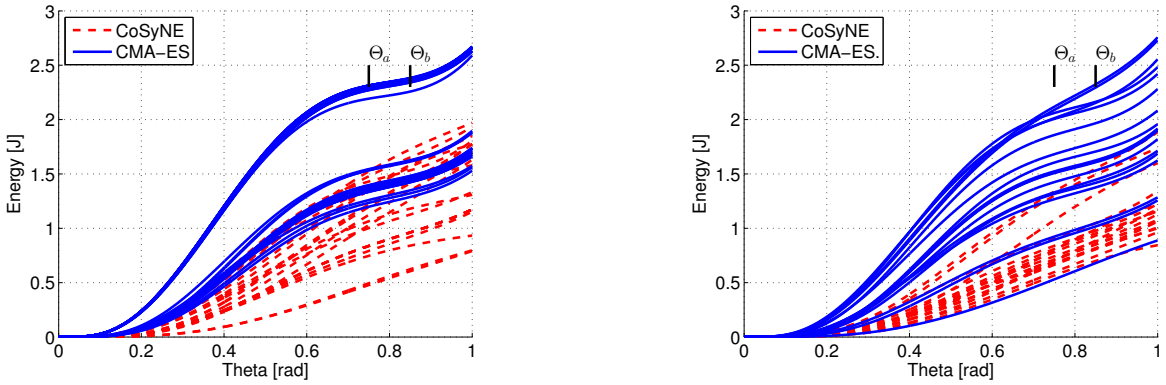
In Figure 12(a) the potential energy curves corresponding to the different system parameter solutions found by the CMA-ES and CoSyNE algorithm for the 1-DOF RA are presented. This plot shows that the CMA-ES algorithm gives a more consistent convergence of the system parameters than the CoSyNE algorithm. For the 2-DOF RA this consistency is lower as can be seen in Figure 12(b), however the CMA-ES algorithm is still better than the CoSyNE algorithm in reducing the slope of the potential energy curve in the place range, which might explain why it is able to reduce the torque further than the CoSyNE algorithm.



(a) System parameters 1-DOF - CoSyNE

(b) System parameters 1-DOF - CMA-ES

Fig. 11. The normalized system parameters of one optimization run plotted over time where 0 represents the lower bound and 1 the upper bound. The figures show that the CMA-ES algorithm is able to give a faster convergence to a system parameter solution while the CoSyNE algorithm is changing the system parameters with bigger steps and does not converge as quickly.



(a) 1-DOF RA

(b) 2-DOF RA

Fig. 12. The potential energy curves of all 30 runs found after optimizing the 1-DOF resonating arm (a) and the 2-DOF resonating arm (b) with the CMA-ES and CoSyNE algorithm. We can see that the CMA-ES algorithm gives more consistent results in comparison to the CoSyNe algorithm when optimizing the 1-DOF resonating arm. If we consider the results of the 2-DOF resonating arm we see that the both algorithms lacked consistency but the CMA-ES algorithm performed better in shaping the potential energy curve to decrease the torques at the pick-and-place positions.

VII. Conclusion

This paper introduced the use of evolutionary co-optimization of control and system parameters in the development of the Resonating Arm (RA); a novel concept for a low-power pick-and-place robot arm. We conclude that evolutionary co-optimization is an effective approach to find near optimal solutions for the RA with one degree of freedom (1-DOF). The 1-DOF RA solution found through evolutionary co-optimization required 43% less torque to perform the pick-and-place tasks when compared to the stepwise optimized solution. However more research is needed to effectively use it in the optimization of the 2-DOF RA, where the results were less consistent. Between the two EAs, CMA-ES yielded the best and most consistent solutions for all experiments.

References

[1] M. Plooij and M. Wisse, "A novel spring mechanism to reduce energy consumption of robotic arms," in *Proceedings of the 2012 International Conference on Intelligent Robots and Systems*, October 2012.

[2] M. M. Williamson, "Robot arm control exploiting natural dynamics," Ph.D. dissertation, Massachusetts Institute of Technology, 1999.

[3] V. Babitsky and A. Shipilov, *Resonant Robotic Systems*. Springer-Verlag, 2003.

[4] K. Sims, "Evolving 3D Morphology and Behavior by Competition," *Artificial Life*, vol. 1, no. 4, pp. 353–372, Jan. 1994. [Online]. Available: http://www.mitpressjournals.org/doi/abs/10.1162/artl.1994.1.353

[5] W. L. Hallam, J. Lund, and H.H., "A hybrid GP/GA approach for co-evolving controllers and robot bodies to achieve fitness-specified tasks," in *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, 1996, pp. 384–389.

[6] C. Leger, *Darwin2K: an evolutionary approach to automated design for robotics*. Kluwer Academic Publishers, 2000.

[7] T. Takagi and M. Sugeno, "Fuzzy Identification of Systems and its Applications to Modeling and Control," *IEEE Transactions on Systems, Man and Cybernetics*, vol. Volume No., pp. 116–132, 1985.

[8] T. Poggio and F. Girosi, "Networks for approximation and learning," in *IEEE 78*, 1990, pp. 1479–1481.

[9] A. Ostermeier, A. Gawelczyk, and N. Hansen, "Step-size adaptation based on non-local use of selection information," in *Parallel Problem Solving from Nature - PPSN III*. Springer, 1994, pp. 189—-198.

[10] F. Gomez, J. Schmidhuber, and R. Miikkulainen, "Accelerated Neural Evolution through Cooperatively Coevolved Synapses," *J. Mach. Learn. Res.*, vol. 9, pp. 937–965, 2008. [Online]. Available: http://portal.acm.org/citation.cfm?id=1390681.1390712

[11] D. S. Naidu, *Optimal Control Systems*. New York: CRC Press, 2003.