International Journal of Advanced Robotic Systems

OPEN ACCESS
ARTICLE

Optimization of PI and Fuzzy-PI Controllers on Simulation Model of Szabad(ka)-II Walking Robot

Regular Paper

István Kecskés[1]* and Péter Odry[2]

1 Obuda University, Budapest, Hungary
2 College of Dunaujvaros, Dunaujvaros, Hungary
*Corresponding author(s) E-mail: kecskes.istvan@gmail.com

Received 01 November 2013; Accepted 18 July 2014

DOI: 10.5772/59102

Abstract

The *Szabad(ka)-II* 18 DOF walking robot and its simulation model is suitable for research into hexapod walking algorithm and motion control. The complete dynamic model has already been built, and is used as a black box for walking optimization in this research. First, optimal straight line walking was chosen as our objective, since the robot mainly moves in this mode. This case can be tested and validated as well on the current version of our robot. An ellipse-based leg trajectory has been generated for this low-cost straight line walking. Currently a simple new Fuzzy-PI controller with three input variables is being constructed and compared with an previously used PI controller. The purpose of defining the rules and its optimization are to obtain a controller that provides walking with higher quality. Both the compared controllers have been optimized together with the parameters of the leg trajectory. The particle swarm optimization (PSO) method was chosen from several methods with our benchmark-based selection research and the help of specific test functions; moreover the previous research (the comparison of genetic algorithm (GA) and PSO) also led to this conclusion.

Keywords Hexapod Robot, Optimization Method Selection, Walking Optimization, PSO, Fuzzy Control

1. Introduction

This research contributes to the development of the driving control optimization of the Szabad(ka)-II mechatronic device [1-7]. It deals with two issues: 1) finding a relatively quick optimization method for a non-differentiable and highly non-linear objective, i.e., the simulation model of the Szabad(ka)-II robot; 2) optimizing hexapod walking with the help of the simulation model in order to achieve an adequate trajectory curve and an effective Fuzzy motor control that can be later embedded into the robot's microcontrollers. However finding a certain optimal solution was not the focus; rather, the goal was to obtain a new procedure that can be generally applied for the tuning of robot controllers if the simulation model is available.

The improvement of the robot's walking on rough terrain is still in progress, including the designing of new legs containing ground contact sensors. In the current research

phase a relatively simple case - the straight-line tripod walking on flat ground - was selected first to develop the optimization system of the robot motion and control. In order to obtain optimal parameter values, which can be used for the building or design of a real robot, a proper robot model is required.

Figure 1. Szabad(ka)-II hexapod robot

The current simulation model was introduced in [3], which is a detailed kinematic and dynamic model of the real Szabad(ka)-II robot (Fig. 1). The aim of this model is to simulate the motion, and measure the walking quality (also known as performance) in comprehensive situations. Controller optimization with the help of the model is important because the system's performance mostly depends on the controller's efficiency [9] besides the structural parameters. Numerous research studies have defined the necessity and role of simulation, controller optimization and fitness function, for example in the conclusion of articles [8, 38]. The controllers and walking trajectories - both developed in this model - can be implemented into the microcontroller of the real robot. The model is validated by the dynamic measurements taken on the real robot (as mentioned in [40], and to be published in detail). Thus the expected deviation between the simulation and reality is known.

1.1 Reason for choosing fuzzy control

The previous research [1] constitutes the basis of this work, which compares two optimization methods (GA and PSO) on a robot-walking task with a PI controller. Some of the most successful applications of fuzzy control have been highly related to conventional controllers, such as proportional-integral-derivative (PID) controller [10]. Currently a Fuzzy-PI controller is being introduced and both the compared controllers are being tuned up with the selected optimization method. The literature provides several examples of the applicability of the fuzzy controller, and most of these also apply the optimization for tuning up Fuzzy parameters, for example: [8, 11-17]. Moreover one publication deals with PID controller optimization [9]. The main difference between fuzzy logic control (FLC) and conventional control is that the former is not based on a properly defined model of the system, but instead implements the same control 'rules' that a skilled expert would operate [10].

The Fuzzy inference system also proved to be suitable for the tuning of the sliding mode control [18], especially in the case of a non-linear system. Hexapod robots also have a strongly non-linear and variable dynamic character, thus the effective control with a Fuzzy type system is expected.

1.2 Fitness function

The most suitable optimum can be obtained primarily if the quality definition is correctly determined. The specific robot's walking optimality is measured by a certain fitness function (also known as cost- or objective function). In the previous research a fitness function was already defined and used for the same problem [1, 5].

The tripod type [19] straight-line hexapod walking on even ground is a simpler case. It has been assumed that in such a case the robot moves towards a farther target point without any manoeuvres and other operations. More energy would remain for the other walking modes if the energy consumption was minimized for straight line walking. Thus the most important task will be to achieve a fast and low-cost (low energy consumption) locomotion. The presented fitness function (1) expresses the quality measurement of these features. It is a concrete aggregation of multi-objectives resulting in a global criterion (F) based on the weighted product method. Generally the goals of robot walking are [5]:

a. achieving the maximum speed of walking with as little electric energy as possible, similarly to [20],

b. keeping the minimal torques on the joints and gears,

c. maintaining the currents of the motors as little discursive and spiky as possible, and

d. keeping the robot's body acceleration at a minimum in all three-dimensional directions.

In order to obtain the results in accordance with our demands, the following should be emphasized (1): In the fitness function the average velocity tag was squared in order to emphasize it as much as the small energy consumption and the accelerations, i. e., these two aspects influence the system oppositely.

$$F = \frac{100000 \cdot \overline{V}_X^2}{E_{WALK} \cdot F_{GEAR} \cdot F_{ACC} \cdot F_{ANG-ACC} \cdot (|Z_{LOSS}| + 0.03)} \quad (1)$$

Where $\overline{V}_X$ - the average walking speed (in direction X); E_{WALK} - electric energy is needed for crossing unit distance; F_{GEAR} - root mean square of the aggregated gear torques;

F_{ACC} - root mean square of acceleration of the robot's body; $F_{ANG-ACC}$- root mean square of angular acceleration of the robot's body; Z_{LOSS}- loss of height in direction Z during the walk. A more detailed description of this fitness function can be found in [4, 5].

1.3 Leg trajectory for straight-line walking

The tripod-type hexapod walking [19] is the most appropriate for a fast and low-cost locomotion. For this walking a three-dimensional ellipse-based trajectory curve was generated that defines the feet's desired cyclic movement in relation to the robot body. More detailed description of this deformed half-ellipse trajectory can be found in [4].

The trajectory curve and the driving motor controller's behaviour directly influence both the real or simulated movement. Since the change of the trajectory's parameters will influence the optimal values of the other parameters – that is, the parameters are not independent – the optimal parameter set should be found in the multi-dimensional space. Therefore the chosen motor controllers and the parameters of this trajectory (see Table 1) have been optimized together. The similar trajectory optimization attempt in [20] did not optimize the motor controller with this trajectory; this is what is different in the current research. The lower (min.) and upper (max.) bounds of these parameters were defined empirically in most cases, with the exception of the upper bound of the fourth 'length of the step' (T_B) parameter given by the structural dimension of the robot.

Parameter	Symbol	Min.	Max.
The cycle's time duration in second	T_{TIME}	0.9	1.7
Length of step – stride, in meter	T_B	0.1	0.18
Height of walk trajectory in meter	T_H	0.01	0.04
Lift (A) and cycle (A+B) ration	$T_{A/(A+B)}$	0.45	0.75
Lowpass FIR filter strength, order in millisecond, *(integer parameter)*	T_{FIR}	4	300

Table 1. Trajectory parameters and its bounds

1.4 Optimization issue

This paper further describes an effort to search for the best optimization method that can most effectively solve the mentioned problem (the best result in terms of performed time and achieved fitness value).

The optimization speed is very important for our system, because the simulation of one second with Szabad(ka)-II dynamic model takes four minutes in an up-to-date PC with a i7-2600K processor (i.e., Simulink solver with 0.2ms time step, model of 18 DC motor, 18 inverse dynamics, etc.). This means one optimization process lasts for several days, and searching for the best optimization method with adequate parameters would last several months.

Chapter 2 briefly presents the benchmarking of the optimization methods applied on different test functions. Multi-dimensional quick test functions have been selected for the benchmark, which have similar non-linear, discontinuous, integer, etc. characters as in the case of the walking simulation of the Szabad(ka)-II robot. Based on the research in Chapter 2 the best possible optimization method could be chosen for the current robot walking.

2. Selection of optimization methods

There are numerous multi-parameter optimization methods, and it is generally difficult to choose the best because the performance of each method is problem-dependent [21]. Based on our experience [1, 4, 5] and literature [8, 21-24] the heuristic and hybrid methods are promising for a non-linear, multi-parameter system. Table 2 lists the selected methods that are currently under test and comparison. Moreover while trying to select the best method the existence of public Matlab implementation [25-30] was taken into account in order to avoid algorithm implementation and obtain quick results:

- Genetic algorithm (GA) can be applied to solve problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, non-differentiable, stochastic, or highly nonlinear [30].
- Particle swarm optimization (PSO) is one of the most important swarm intelligence paradigms [12]. The PSO uses a simple mechanism that mimics swarm behaviour in bird flocking and fish schooling to guide the particles to search for globally optimal solutions [16]. There is no built-in PSO algorithm in Matlab, and thus external source exploration was needed. Considering the characteristics of the available implementations, [27] seems to be the good choice. It is easy to learn, has the ordinary Matlab-like syntax, and has only the necessary options.
- The pattern search (PS) algorithm supported in Global Optimization Toolbox by Matlab [29].
- Gravitational search algorithm (GSA) is a never-heuristic optimization method, which is constructed based on the law of gravity and the notion of mass interactions [31].
- Simulated annealing (SA) models the physical process of heating a material and then slowly lowering the temperature to decrease defects, thus minimizing the system energy [28].
- Teaching-learning-based optimization (TLBO) is a population-based, new, efficient optimization method, which works on the effect of the influence of a teacher on learners. [32]
- Tabu search (TS) is a heuristic method but is still very limited for dealing with continuous problems. The directed tabu search (DTS) is a continuous TS that also

uses the Nelder-Mead method and adaptive pattern search. [33]

- The new version of 'multistart clustering global optimization method' (called GLOBAL) utilizes the advantages offered by Matlab, and the algorithmic improvements increase the size of the problems that can be solved reliably with it [25].

Method	Symbol	Source
Own implementation of Genetic Algorithm	GA-IK	[5]
Genetic Algorithm in Global Optimization Toolbox by Matlab	GA	[30]
Particle Swarm Optimization	PSO	[27]
Particle Swarm Optimization with Pattern Search hybrid	PSO-PS	[27]
Gravitational Search Algorithm	GSA	[31]
Simulated Annealing	SA	[28]
Pattern Search in Global Optimization Toolbox by Matlab	PS	[29]
Teaching-learning-based optimization	TLBO	[32]
Directed Tabu Search	DTS	[33]
Multistart clustering global optimization method GLOBAL, with local search UniRandi	GLuni	[25]
GLOBAL with local search FminSearch	GLfmin	[25]
GLOBAL with local search BFGS	GLbfgs	[25]

Table 2. Selected optimization methods for the benchmark on test functions

2.1 Test functions

Not all the selected optimization methods with various configurations are worth running on the simulation model of the Szabad(ka)-II robot, because it would take half a year (see chapter 1.4). This led to the application of the methods' benchmark on faster test functions, and offered a kind of pre-selection of methods based on some key characteristic behaviours. The current dynamic model of hexapod walking - in view of character - is a multi-dimensional, highly nonlinear, non-smooth, and a slightly mixed integer problem, i.e., it:

- Has a minimum of seven dimensions: PI controller has seven dimensions (5 trajectory+2 PI parameters), while the Fuzzy-PI has 17 dimensions (5 trajectories+12 Fuzzy parameters).
- Has non-continuous behaviour due to walking on six legs and the ground contact.
- Has no random parts.
- Contains integer parameters, e.g., the trajectory parameter T_{FIR} is an integer type, see Table 1 and Table 4.

The ground contact model of the six legs – a critical part of the dynamic model - has a discontinuous character as it can be seen in formulae (2), (3) and (4). More details about this ground contact model can be found in [3, 7]. The backlash occurrence at the robot's links and gears also has a non-smooth feature.

$$F_Z = \begin{cases} -kz - cz & if \quad z < 0 \\ 0 & if \quad z \geq 0 \end{cases} \tag{2}$$

$$F_{NORM} = \begin{cases} -F_z & if \quad F_z > 0 \\ 0 & if \quad F_z \leq 0 \end{cases} \tag{3}$$

$$F_x = \begin{cases} -\mathrm{sgn}(v_X)\delta F_{NORM} & if \quad |v_X| - v_d > 0 \\ 0 & if \quad |v_X| - v_d \leq 0 \end{cases} \tag{4}$$

Therefore test functions have been selected based on the mentioned aspects in order to ensure the testing of these characters:

- smaller (marked with D4) and larger (D7) dimensions,
- continuous (C1) and discontinuous (C0),
- with integer (I1) and without integer (I0),
- with random (R1) and without random (R0).

Both of them can be seen in formulae (5) and (6); the rest assemble from the mixing of presented function tags, the exact optimum is known, and the various methods (run with the same conditions) make efforts to find this set as much as possible. Selected methods run as constrained optimization, and the test function has been scaled in order to support unified side constraints $-1 \leq x \leq 1$, except the integer parameter, which has $0 \leq x \leq 10$ ranges. These test functions can be downloaded from the webpage [34].

The discontinuous (C0) and seven dimension (D7) functions are more interesting for the present problem. Bearing in mind the previous facts and assumptions the *D7C0R0I1* function is the closest to this simulation system as the objective function. It is expected that the robot-walking problem will be effectively optimized with the methods providing better results for such a test function that has the same characteristics as the problem. This assumption was confirmed in this example.

2.2 Optimization benchmark on test functions

Each optimization method was run $N=100$ times with various configurations on all test functions. The configuration refers to some main parameters of a certain optimization method, which was randomly selected in each case (for example, in the case of GA: *generations, population, elite count, crossover type*).

Fig. 2 shows the results in case of four-dimension test functions, while Fig. 3 and Fig 4 illustrate the seven dimension cases. The left-bottom corners represent the best performance, i.e., the better fitness on the horizontal axis and the smaller number of function calls on the vertical axis. An acceptance condition was defined, and plotted with a magenta line. Different performance clouds can be seen in

cases of various types of functions. There are more methods reaching acceptable results for the four-dimension problem (Fig. 2). However, in case of seven dimensions (D7) and discontinuous (C0) benchmark only the PSO, the PSO-PS hybrid, and TLBO methods reach really acceptable results (Fig. 3 and 4). The following findings can be obtained from the clouds in Fig. 3 and 4:

- The PSO, PSO-PS, and TLBO methods provide the best stable results for all the discontinuous functions.
- The PSO-PS hybrid method contains the good performance of PSO and the stableness of PS. Thus this will be the best choice for higher dimension problems, especially in the case of *D7C0R0I1* function (left-top graph in Fig. 4), which is most similar to the current robot model.
- The GL*, PS and DTS methods reach almost the best results for the continuous functions without random tags, but in other cases give a lower performance.
- The GA, GA-IK, GSA and SA methods do not reach acceptable results in any case. The SA method seems to be the weakest for all types of functions.
- The GA methods reach a lower performance but keep roughly similar values for different test functions. This reinforces the problem-independent character of GA.
- The GSA method is excellent only for the continuous problems without an integer tag, but very weak for the others.

$$f_{D4C0R0I1}(x \mid x \in R^4) = \begin{cases} \left| \left| 1 + \dfrac{10}{round(x_1)/10 - 0.9} \right| + \log_2(|x_2 + 1.3| + 1) \right| \\ \quad + \dfrac{|round(x_3) - 8|}{10} + sign(x_4 + 0.4) & x_2 \geq -0.3 \\ \left| \left| 1 + \dfrac{10}{round(x_1)/10 - 0.9} \right| + \log_2(|1.8 - x_2| + 1) \right| \\ \quad + \dfrac{|round(x_3) - 8|}{10} + sign(x_4 + 0.4) & x_2 < -0.3 \end{cases} \tag{5}$$

$$f_{D7C1R1I0}\begin{pmatrix} x \mid \\ x \in R^7 \end{pmatrix} = \left| \left| 1 + \frac{1}{x_1 - 0.9} \right| + (x_2 + 0.5)^2 + \sqrt{|x_3 + 0.2|} + rand(1)\sqrt{|x_4 + 0.6|} \right| + \log_2(|x_5 + 2| + 1) + \frac{|x_6 - 8|}{10} + x_7 - 0.6 \tag{6}$$

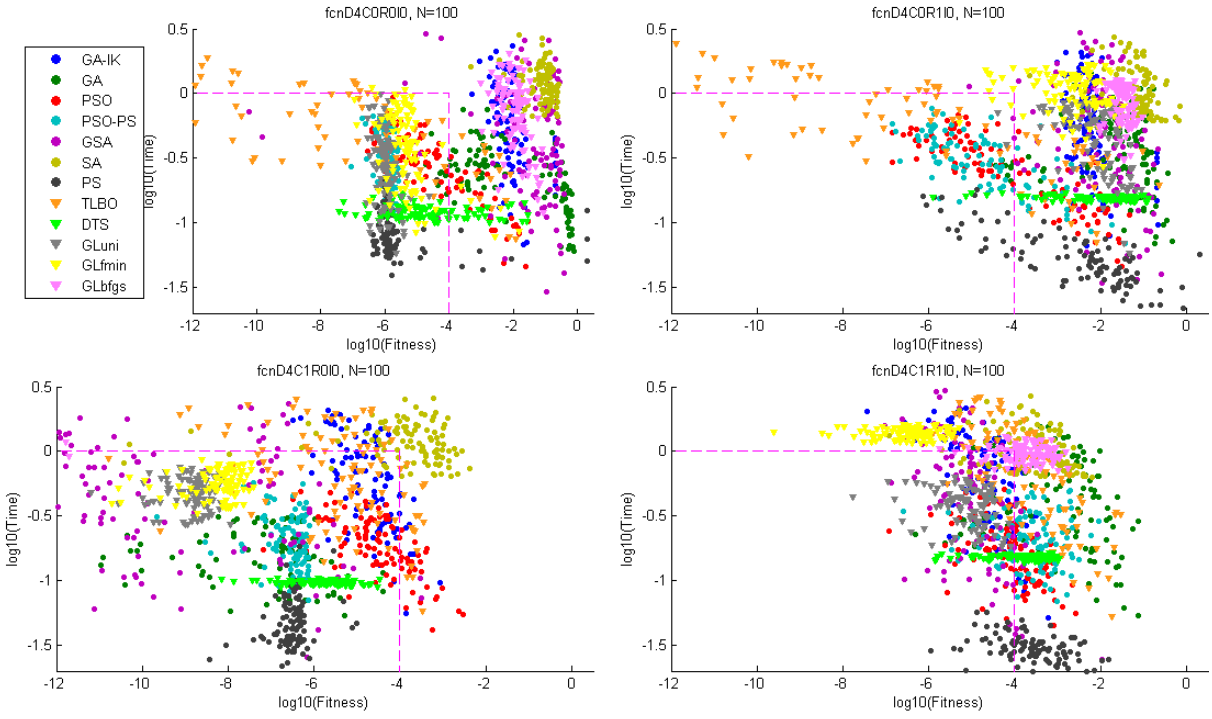

Figure 2. Optimization benchmark on functions of four dimensions (D4) and without integer (I0)

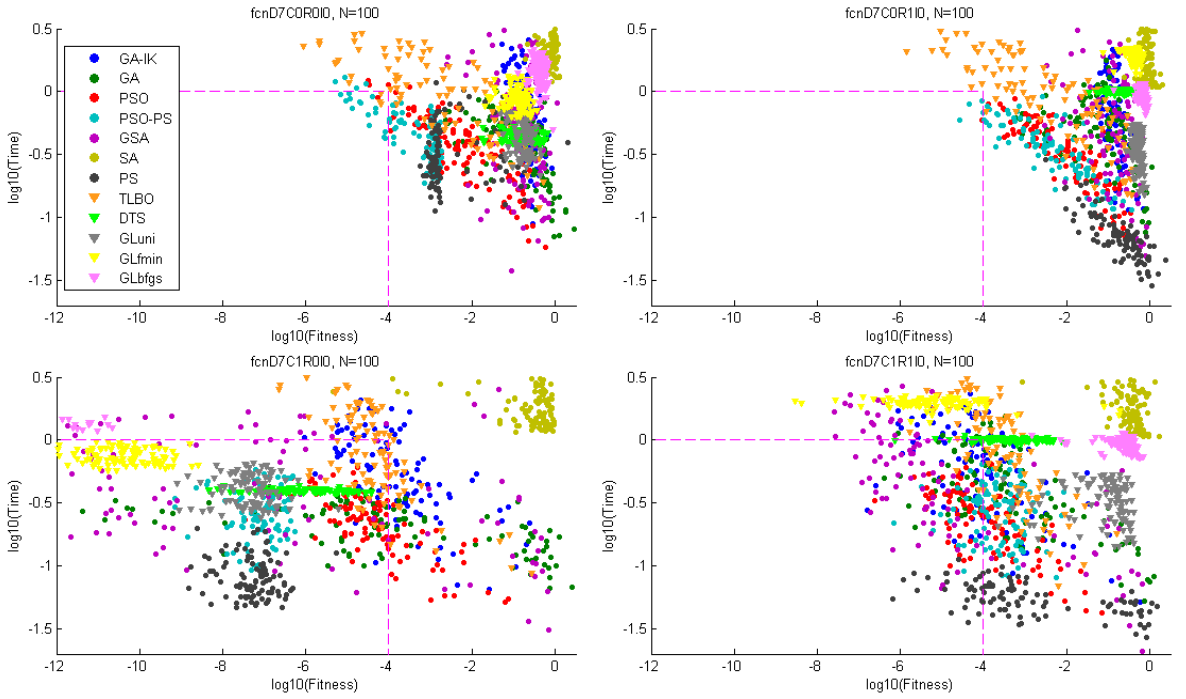

Figure 3. Optimization benchmark on functions of seven dimensions (D7) and without integer (I0)

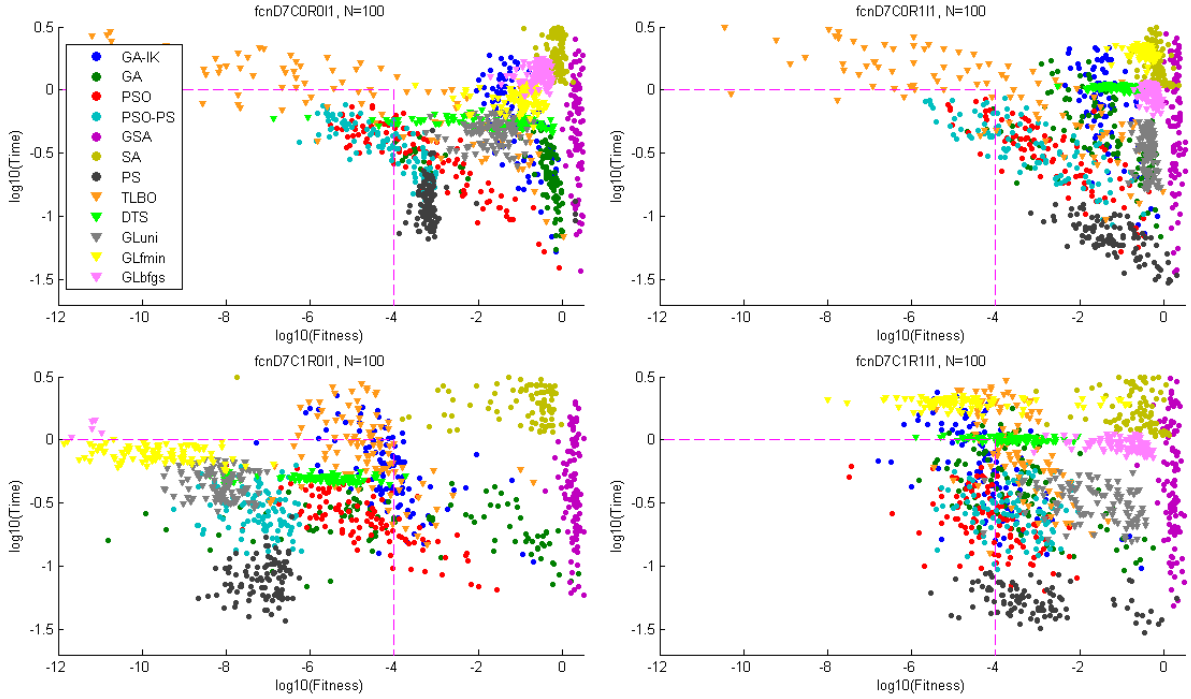

Figure 4. Optimization benchmark on functions of seven dimensions (D7) and with integer (I1)

The results of this benchmark contributed and confirmed the effectiveness of the selected PSO method as mentioned in papers [1, 11, 24, 15]. Similar benchmark efforts can be found in [22] where the PSO also reaches a very good performance level. In paper [11] a benchmark of optimization methods (ANFIS, PSO among others) was also applied on a Fuzzy controller, not on the test functions. It also confirmed the PSO usability for tuning the Fuzzy system. The pattern search (PS) can refine the result from PSO (compare PSO and PSO-PS clouds in Fig. 2, 3, 4). It runs

after PSO and is initialized by the best entities of PSO. Thus, the PSO-PS pair has become the best method for us.

3. Fuzzy-PI controller

First of all the design of a controller is primarily defined by the fact that it should be implemented into the microcontrollers of the real Szabad(ka) robot series. This means that the memory and calculation demand of the controller should be maintained within certain boundaries. From another aspect only just the available measured quantities can be used as input (of a Fuzzy controller) due to the given sensor interface.

Based on previous research [6, 35] a Fuzzy controller with some rules is enough for obtaining an improved result compared with the simple PID controller. One of the key things is the fact that Fuzzy can include more inputs, while the PID has only one (the error of control variable). In case of robot link control it is the angle error, i.e., the difference between the desired and measured angle. Besides this the absolute value of the motor current was put into the Fuzzy inputs; it was possible since the electronics on the Szabad(ka)-II measures this. A similar solution [10] was found in the literature; however the authors do not explain the role of this current feedback. In addition, if required, the derivative of angle error and the error of angle velocity could be used as input or the measured angle value might also be applied in case the controlling behaviour is different in a certain angle section.

3.1 Fuzzy inputs and outputs

The block diagram in Fig. 5 shows the designed Fuzzy-PI controlling cycle with the inputs and outputs. The three selected inputs are: error angle (A_{ERR}), error velocity (V_{ERR}), and motor current (I_{MA}), while the two outputs are: proportional tag of voltage ($FzzP$), and integrative tag of voltage ($FzzI$). A controller system with the same parameters and conditions should be provided for each 18 DC motor of the robot. Currently this controller has been implemented only on the dynamic model of Szabad(ka)-II robot, when it was tested and optimized.

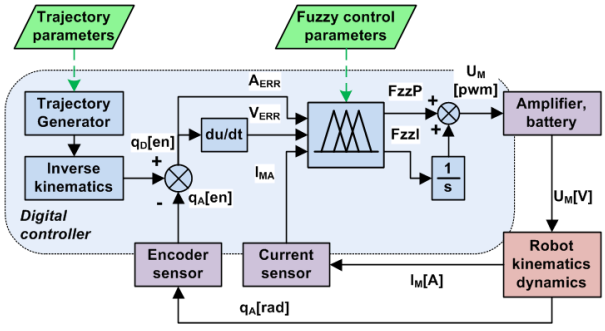

Figure 5. Fuzzy-PI motor control loop in the dynamic model of Szabad(ka)-II robot

3.2 Membership functions and rules

Fig. 6 presents the necessary membership functions (MFs) and the eight rules defined by the authors, which mostly determine the controlling character:

- The first rule refers to cases when there is no error angle and the outputs come near to zero.
- The second rule ensures that if the velocity error is small then the integration output tends toward zero.
- The third and fourth rules ensure the output activity in order to decrease the control (angle) error.
- The fifth and sixth rules have an opposite influence to the third and fourth rules, but only when the motor current is high. These rules ensure a softer feature of controlling when the currents or torques are great, and thus can protect against electrical and mechanical overload.
- The seventh and eighth rules reinforce the integrative output activity for decreasing the velocity error when the motor current is smaller.

The logic of these rules has been reinforced by earlier research [6], but on the other hand the optimization process should select the necessary or dominant rules by tuning up its weights.

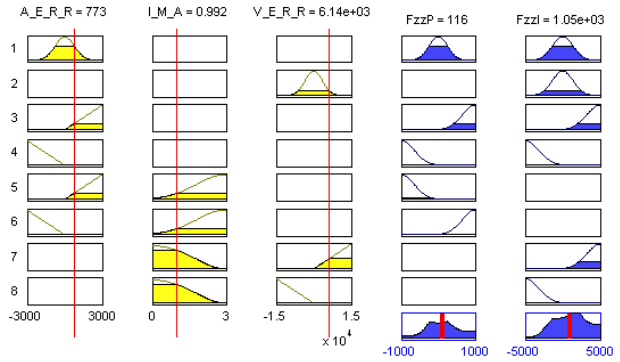

Figure 6. Rules of Fuzzy-PI controller: first input column is the error angle, second input column is the absolute motor current, third column is the error, first output column is the proportional tag, and second output column is the integrative tag.

Fig. 7 illustrates the output surfaces of the built Fuzzy-PI controller, where the aggregated effects of the previously described rules can be observed.

3.3 Selecting fuzzy-PI parameters for optimization

The number of Fuzzy controller parameters depends on the number of all MFs and rules. If it is assumed that the defined rules are suitable, then only the weight of them count as target parameters. Furthermore there is no need to count separate parameter values for the symmetric MFs and rules. According to this the current Fuzzy-PI controller has 37 parameters in all:

- 5 method type parameters: *AndMethod, OrMethod, ImpMethod, AggMethod, DefuzzMethod*
- 9 MFs x 3 parameters (2 scale values+1 function type value (*trimf* or *gaussmf*))
- weight of 5 rules (8 rules – 3 symmetry)

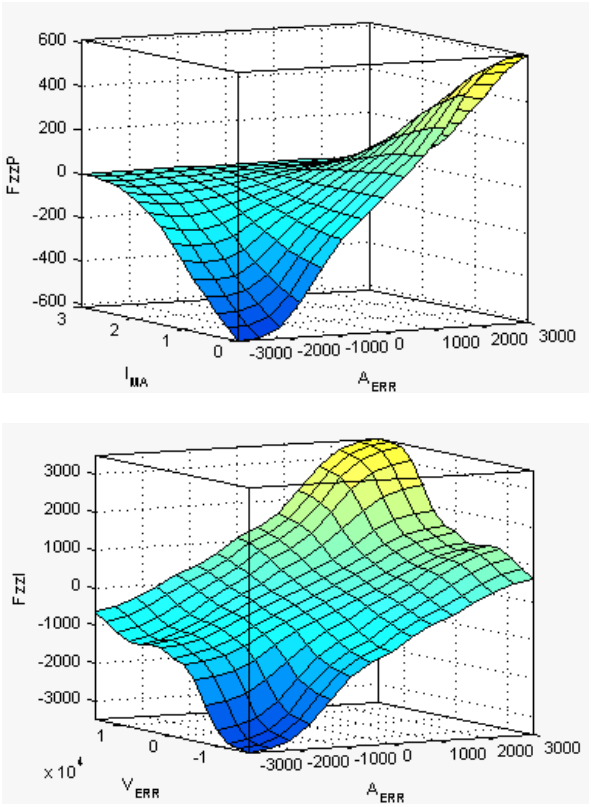

Figure 7. Outputs' surfaces of Fuzzy-PI controller, above the proportional output, below the integrative output

Despite this the parameters of MFs have been reduced by the following method: only the range values of inputs and outputs have been changed, thus the internal MFs do not change relative to each other. For the modification of the range values it is also necessary to convert the parameters of the Fuzzy membership functions, for which the Fuzzy Toolbox's **`strtchmf`** function can be applied. Additionally the MF types have been selected for optimization. The Matlab's built-in Fuzzy Toolbox supports more MF types; however, the converting of one MF type into a second type is not a trivial task if the character is to remain. The Fuzzy Toolbox's **`mf2mf`** function also cannot properly convert the MFs in all cases. From the original triangle MF (*trimf*) the gauss MF (*gaussmf*) can be converted in the easiest way, which is why only these two types were selected. The MF's own parameters could also be changed, but it is not applied now because it needs a more complex solution due to the incomparable parameters of the different MF types.

Table 3 contains the selected 12 main parameters of the current Fuzzy-PI controller with the target domains (*Min, Max columns*).

Parameter	Min	Max	Note
Input 1 (A_{ERR}) range	500	10000	Lower Upper
Input 2 (I_{MA}) upper range	1.0	6.0	
Input 3 (V_{ERR}) range	1000	30000	Lower Upper
Output 1 (*FzzP*) range	200	5000	Lower Upper
Output 2 (*FzzI*) range	500	10000	Lower Upper
Output 1 MF's type	1	2	1-tri
Output 2 MF's type	1	2	2-gauss
Rule 1 weight	0	1	
Rule 2 weight	0	1	
Rule 3 and 4 weight	0	1	
Rule 5 and 6 weight	0	1	
Rule 7 and 8 weight	0	1	

Table 3. Fuzzy-PI controller parameters and its target boundaries

The MF types of inputs have not been selected for optimization, partly because they only slightly influence the output surface, and partly because the selected shapes were intended. For example, the triangle shapes at positive MF and negative MF of angle error (A_{ERR}) are important for precise control.

4. Results and comparison

4.1 Simulation

The simulation model was introduced in [3], which is a detailed and 3D kinematic and dynamic model of the real Szabad(ka)-II hexapod robot, implemented in Simulink environment with the help of Robotics toolbox [39]. It includes the exact copy of the digital controller with the trajectory generator, DC motors and gears, rigid body dynamics, the ground contacts as an approximated model, ground surface, and in some degree the sensors (encoder, current measurement, and accelerometer).

In the initial state of the chosen simulation case, the robot's bottom point was *1mm* above the ground; the legs were set to the initial points of the desired trajectories. The simulation time was selected for only three seconds in order to hasten the runtime as much as possible, but at least to enable the simulation of one and a half walking cycles. The ground was even with a *0.9* friction constant. After the simulation the fitness function runs and calculates the specified quality of walking based on the simulation results (robot movements, motor currents, link torques).

4.2 Optimization results

The results of optimization can be seen in Table 4 for both controller types (PI and Fuzzy-PI), and for two optimization cases with PSO-PS and PSO methods. The PSO-PS method was selected for the current optimization case, the reason described in Chapter 2. The PSO-specific parameters were: the number of generation was selected to *NG=70*, the population size was *NP=40*, the inertia weight *w=0.9*, the cognitive attraction *c1=0.5*, and the social attraction *c2=1.5*. These parameters were selected partly from the literature [1, 11, 23, 21, 24, 15], partly from own experience.

Table 5 comprises the detailed partial results of the fitness evaluation (1). However, the Fuzzy-PI-PSO method seems to be the best one if only the lowest energy consumption and the fastest movement is considered. But the lower accelerations and stability are also important for the quality, and that is why our fitness function (1) takes into account all of these properties. In this respect the Fuzzy-PI-PSO-PS has the best fitness value (without any significant differences).

	Parameter	Fzz-PI PSO-PS	Fzz-PI PSO	PI PSO-PS	PI PSO
Trajectory	The cycle's time duration	1.740	1.733	1.808	1.696
	Length of step (stride)	0.163	0.161	0.168	0.142
	Height of walk trajectory	0.0364	0.0329	0.0397	0.0366
	Lift (A) and cycle (A+B) ration	0.564	0.544	0.574	0.577
	Lowpass FIR filter strength	12	33	93	9
PI	Proportional			0.454	0.340
	Integral			0.147	0.534
Fuzzy-PI	Input 1 range	5126	6369		
	Input 2 upper range	4.4	1.465		
	Input 3 range	11682	12103		
	Output 1 range	2227	2620		
	Output 2 range	2176	6362		
	Output 1 MF's type	2	2		
	Output 2 MF's type	1	2		
	Rule 1 weight	0.066	0.234		
	Rule 2 weight	0.996	0.281		
	Rule 3, 4 weight	0.215	0.496		
	Rule 5, 6 weight	0.385	0.258		
	Rule 7, 8 weight	0.783	0.502		

Table 4. Optimized walking parameters: trajectory and controllers

Fitness Property	Fzz-PI PSO-PS	Fzz-PI PSO	PI PSO-PS	PI PSO
Gear torques	8.71	9.09	8.82	9.12
Body acceleration	1.77	1.89	1.77	1.80
Body angular acceleration	16.09	17.09	16.39	17.2
Energy per meter	41.96	41.05	42.55	42.5
Loss of height	-3.8e-3	-5.5e-3	-7.7e-3	-7.3e-3
Mean velocity	0.156	0.163	0.152	0.152
Fitness value	**6.887**	**6.209**	**5.644**	**5.171**
Number of function calls	**3872**	**1442**	**1776**	**645**

Table 5. Fitness values and function tags in cases of optimized PI and Fuzzy-PI system. More details can be found in [1].

4.3 PI and fuzzy-PI comparison

Fig. 8 shows the time diagram of the robot movement (B_X), velocity (BV_X) acceleration (BA_{Mag}), and the summarized motor currents (I_{SUM}) for five optimized cases:

- Fuzzy-PI controller optimized with PSO-PS method (red)-the best Fuzzy solution, high fitness obtained by smaller energy consumption and smaller acceleration, see Table 5.
- Fuzzy-PI controller optimized with PSO method (yellow)-the Fuzzy reached a little faster movement than the PI besides a roughly same power consumption and body acceleration. This was the main reason for the higher fitness value.
- PI controller optimized with PSO-PS method (blue) – the best PI solution, the details can be seen in Table 5.
- PI controller optimized with PSO method (green) – an interesting trajectory can be observed, very similar to the best Fuzzy-PI solution
- PI controller optimized with GA previously [1, 5] (grey) – it is important to present the mentioned typical hexapod gait problem, i.e., the significant fluctuation of velocity.

Based on the comparison of these four optimized cases and the other results obtained during the development it can be concluded that some solutions reach the high fitness value by higher speed, while others reach this value by smaller energy and acceleration. In spite of the difference between the presented four cases, both control methods in all given solutions generate high quality walking: the fluctuation of velocity is relatively small compared to a typical inadequate hexapod walking, illustrated with grey in Fig. 8, and found in [1, 5, 6]. Mostly the motor currents have different curves due to the fact that the Fuzzy also includes the current in the control decision.

The three-dimensional leg trajectory can be seen in Fig. 9, related to the five mentioned optimized cases in Fig. 8. The

ellipse-based desired trajectory was also plotted with a little shift besides the simulated-regulated trajectory. The simulated-regulated angle curves of three robot leg links illustrated on the right follow the desired angle curves calculated with inverse kinematics. The explanation of the presented link numeration can be found in Fig. 3 in paper [36].

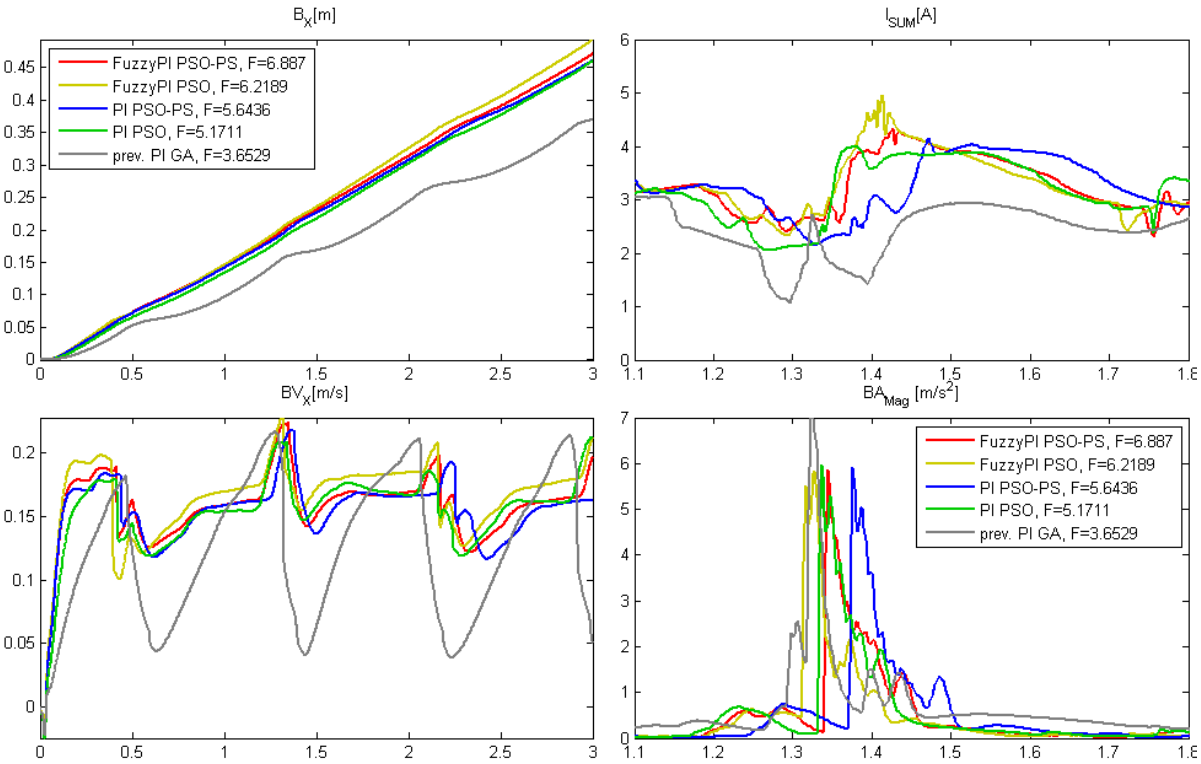

Figure 8. Comparison of optimized walking with PI and Fuzzy-PI controllers: movement (B_X) at top-left, velocity (B_{VX}) at bottom-left, summarized motor current (I_{SUM}) at top-right, and magnitude of 3D body acceleration (BA_{Mag}) at bottom-right

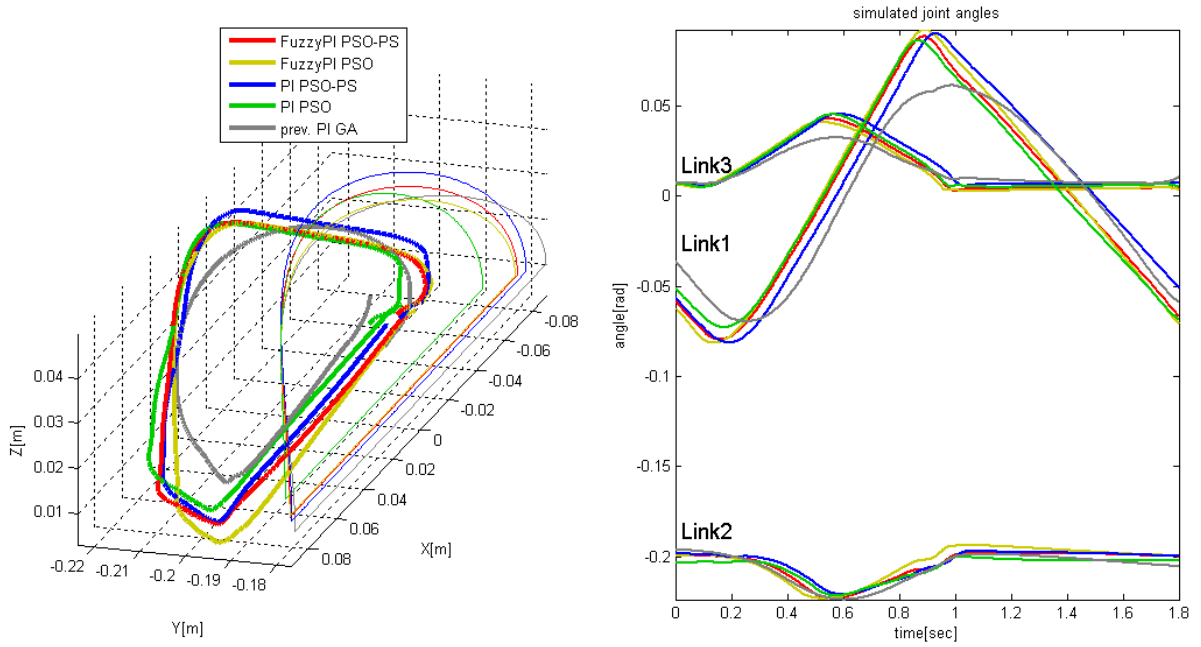

Figure 9. Leg trajectory curves of four optimized cases: the desired and simulated trajectory curves (left), simulated angles of three links (right)

5. Conclusion

During this research the authors gained experience in the field of robot control optimization. A new and more widely usable method was created for (pre)selecting the potentially best optimization method(s) used for a given problem. The test functions for a benchmark were created including those mathematical characteristics that are interesting or typically describe the examined objective function (chapter 2.1). The selection of these characteristics was the key point, since certain methods provide significantly different performance levels for different test functions with various characteristics (see chapter 2.2). It can be observed that the parameters of the optimization method also have an influence, because the random change of these parameters formed a cloud for each method (see coloured clouds in Fig. 2, 3, 4). The best set of these parameters can also probably be deducted from these benchmark results.

In the current demonstrated case the objective function is the straight-line walking quality of the Szabad(ka)-II robot with a Fuzzy controller in virtual simulation space using the dynamic model. The design vector (parameters to be optimized) of this system contains both parameters of the trajectory and the controller. If the optimization methods were tested directly on this simulation model, the computation would take some months (see chapter 1.4) due to the model's complexity i.e., it is specifically computationally expensive [3]. Contrary to benchmarking on fast test functions this takes a much shorter time: only a few hours. Thus the optimization of the robot model has been run only with the benchmark-selected methods. The PSO and PSO-PS methods were selected as best for the function having similar characteristics as the robot walking system. The PSO-PS method proves to be effective compared with the earlier optimization attempts [1, 4, 5], giving significantly better results, independent of the controller type (see chapter 4.2). Previously the GA optimization method was used for the same walking problem, and the results (fitness) were significantly worse $F=3.78$ [1]. This research pointed to the fact that a suitable method should be found for each optimization problem, thus reaching the best results quickly-hopefully the optimum.

Besides, this research also confirmed that a well-defined Fuzzy type controller is a more customizable motor controller than a simple PI controller. A relatively simple Fuzzy-PI controller was constructed based on previous experience [6] (see chapter 3) in order to implement it into the microcontrollers of the real robot without any resource problems. After the optimization procedures - run with similar conditions - the Fuzzy-PI controller reached nearly 22% better walking quality ($F_{FZZ}=6.88/F_{PI}=5.64 \cong 1.22$). Of course, the obtained controller itself is not sufficient to drive the robot with various gaits and on various terrains, and was not tested yet on the hardware. Nevertheless the current attempt shows the way to the optimal, Fuzzy-based robot motion controller.

The well-defined quality formulation and proper weighting of fitness function tags (objectives) are important according to our experience. The next major research task intends to focus on the following issues: which simulation cases and fitness functions can result in such a trajectory and controller that ensures similar effective functioning, which is robust for various environmental influences. These environmental parameters are the robot weight, walking direction, type and slope of the ground, etc.

6. Acknowledgements

This work/publication is supported by the TÁMOP-4.2.2.A-11/1/KONV-2012-0027 project. The project is co-financed by the European Union and the European Social Fund.

7. References

[1] István Kecskés, László Székács, János C. Fodor, Péter Odry (2013) PSO and GA Optimization Methods Comparison on Simulation Model of a Real Hexapod Robot. ICCC, pp. 125-130, ISBN: 978-1-4799-0060-2

[2] Ervin Burkus, Peter Odry (2008) Autonomous Hexapod Walker Robot 'Szabad(ka)'. Acta Polytechnica Hungarica, Vol. 5, No. 1, pp. 69-85, ISSN 1785-8860

[3] István Kecskés, Péter Odry (2008) Full Kinematic and Dynamic Modeling of 'Szabad(ka)-Duna' Hexapod. SISY 2009, pp. 215-219, ISBN: 978-1-4244-5348-1

[4] István Kecskés, Péter Odry (2009) Walk Optimization for Hexapod Walking Robot. CINTI, pp. 265-277

[5] Zoltán Pap, István Kecskés, Ervin Burkus, Fülöp Bazsó, Péter Odry (2010) Optimization of the Hexapod Robot Walking by Genetic Algorithm. SISY, pp. 121-126, ISBN: 978-1-4244-7394-6

[6] István Kecskés, Péter Odry (2010) Protective Fuzzy Control of Hexapod Walking Robot Driver in Case of Walking and Dropping. Springer, Vol. 313, pp. 205-217, DOI:10.1007/978-3-642-15220-7_17

[7] István Kecskés, Péter Odry (2013) Simple Definition of Adequate Fixed Time-Step Size of Szabad(ka)-II Robot Model. ICCC, pp. 315-320, ISBN: 978-1-4799-0060-2

[8] Radu-Emil Precup, Radu-Codrut David, Emil M. Petriu, Mircea-Bogdan Radac, Stefan Preitl, János C. Fodor (2013) Evolutionary Optimization-Based Tuning of Low-Cost Fuzzy Controllers for Servo Systems. Knowledge-Based Systems, Vol. 38, pp. 74–84

[9] Arturo Y. Jaen-Cuellar, Rene de J. Romero-Troncoso, Luis Morales-Velazquez, Roque A. Osornio-Rios (2013) PID-Controller Tuning Optimization with

Genetic Algorithms in Servo Systems. IJARS, Vol. 10, 324, DOI: 10.5772/56697

[10] Ming-Shyan Wang, Ying-Shieh Kung, and Yi-Ming Tu (2009) Fuzzy Logic Control Design for a Stair-Climbing Robot. International Journal of Fuzzy Systems, Vol. 11, Issue 3, p174-182

[11] Mahdi Aliyari Shoorehdeli, Mohammad Teshnehlab, Ali Khaki Sedigh (2009) Training ANFIS as an Identifier with Intelligent Hybrid Stable Learning Algorithm Based on Particle Swarm Optimization and Extended Kalman Filter. Fuzzy Sets and Systems 160, pp. 922–948, DOI:10.1016/j.fss.2008.09.011

[12] Ching-Chang Wong, Hoi-Yi Wang, Shih-An Li, and Chi-Tai Cheng (2007) Fuzzy Controller Designed by GA for Two-wheeled Mobile Robots. International Journal of Fuzzy Systems, Vol. 9, No. 1, pp. 22-30

[13] Pratihar Dilip. Kumar, Deb Kalyanmoy, & Ghosh Amitabha (2002) Optimal Path and Gait Generations Simultaneously of a Six-Legged Robot Using GA-Fuzzy Approach. Robotics and Autonomous Systems, Vol. 41, pp. 1–20

[14] Rong-Jong Wai (2003) Robust Fuzzy Neural Network Control for Nonlinear Motor-Toggle Servomechanism. Fuzzy Sets and Systems, Vol. 139, pp. 185–208

[15] Ching-Chang Wong, Hou-Yi Wang, and Shih-An Li (2008) PSO-based Motion Fuzzy Controller Design for Mobile Robots. International Journal of Fuzzy Systems, Vol. 10, No. 1, pp. 284-292

[16] Abraham Melendez, Oscar Castillo, Patricia Melin (2013) Evolutionary Optimization of the Fuzzy Controllers in a Navigation System for a Mobile Robot. ICIC International, pp. 451-464, ISSN 1349-4198

[17] Radu-Emil Precup, Hans Hellendoorn (2011) A Survey on Industrial Applications of Fuzzy Control. Computers in Industry 62, pp. 213–226, DOI: 10.1016/j.compind.2010.10.001

[18] Kemalettin Erbatur, Okyay Kaynakb, Asif Sabanovicc, Imre Rudas (1996) Fuzzy Adaptive Sliding Mode Control of a Direct Drive Robot. Robotics and Autonomous Systems, Vol. 19, No. 2, pp. 215–227

[19] Ganesh Ramanathan (2009) SCOOP for Robotics, Implementing Bio-Inspired Hexapod Locomotion, Available from: http://se.inf.ethz.ch/old/projects/ganesh_ramanathan/report.pdf Accessed on 23 Mar 2014

[20] Mustafa Suphi Erden (2011) Optimal Protraction of a Biologically Inspired Robot Leg. J Intell Robot Syst Vol. 64, pp. 301–322, DOI: 10.1007/s10846-011-9538-8

[21] Singiresu S. Rao (2009) Engineering Optimization: Theory and Practice, Fourth Edition. Hoboken, New Jersey, ISBN: 978-0470183526

[22] Luis Miguel Rios, Nikolaos V. Sahinidis (2013) Derivative-Free Optimization, A Review of Algorithms and Comparison of Software Implementations. Journal of Global Optimization, Vol. 56, Issue 3, pp. 1247-1293, DOI:10.1007/s10898-012-9951-y

[23] Pakize Erdogmus and Metin Toz (2012) Heuristic Optimization Algorithms in Robotics, Serial and Parallel Robot Manipulators-Kinematics, Dynamics, Control and Optimization, Dr. Serdar Kucuk (Ed.), ISBN: 978-953-51-0437-7, InTech DOI: 10.5772/30110. Available from: http://www.intechopen.com/books/serial-and-parallel-robot-manipulators-kinematics-dynamics-control-and-optimization/heuristic-optimization-algorithms-in-robotic. Accessed on 23 Mar 2014

[24] Magnus Erik Hvass Pedersen (2010) Good Parameters for Particle Swarm Optimization. Hvass Laboratories, Technical Report No. HL1001

[25] Tibor Csendes, László Pál (2008) The GLOBAL Optimization Method Revisited. Optimization Letters, Vol. 2, No. 4, pp. 445-454, DOI: 10.1007/s11590-007-0072-3

[26] www.mathworks.com/products/global-optimization/index.html. Accessed on 23 Mar 2014

[27] code.google.com/p/psomatlab/, or http://www.mathworks.com/matlabcentral/fileexchange/25986. Accessed on 23 Mar 2014

[28] www.mathworks.com/discovery/simulated-annealing.html. Accessed on 23 Mar 2014

[29] www.mathworks.com/help/gads/pattern-search.html. Accessed on 23 Mar 2014

[30] www.mathworks.com/discovery/genetic-algorithm.html. Accessed on 23 Mar 2014

[31] Esmat Rashedi, Hossein Nezamabadi-pour, Saeid Saryazdi (2009) GSA: A Gravitational Search Algorithm. Information sciences, Vol. 179, No. 13, pp. 2232-2248

[32] Ravipudi V. Rao, Vimal J. Savsani and D. P. Vakharia (2011) Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer Aided Design. Vol. 43, Issue 3, pp. 303–315.

[33] Abdel-Rahman Hedar, Masao Fukushima (2004) Tabu Search Directed by Direct Search Methods for Nonlinear Global Optimization. Available from: www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/go_files/DTS.pdf. Accessed on 23 Mar 2014

[34] www.szabadka-robot.com. Accessed on 23 Mar 2014

[35] István Kecskés, Péter Odry (2012) Fuzzy Route Control of Dynamic Model of Four Wheeled Mobile Robot, LINDI, 2012, 4th IEEE International Symposium on, pp. 215-220, ISBN: 978-1-4673-4520-0, DOI: 10.1109/LINDI.2012.6319490

[36] Ervin Burkus, János C. Fodor, Péter Odry (2013) Structural and Gait Optimization of a Hexapod Robot with Particle Swarm Optimization, SISY, 2013, IEEE 11th International Symposium on, pp. 147-152, ISBN: 978-1-4799-0305-4

[37] www.faulhaber.com. Accessed on 23 Mar 2014

[38] Andrew L. Nelson, Gregory J. Barlow, Lefteris Doitsidis (2009) Fitness Functions in Evolutionary Robotics: A Survey and Analysis, Robotics and Autonomous Systems, Vol. 57, pp. 345-370

[39] Peter I. Corke (2001) Robotics Toolbox for Matlab (Release 6), Manufacturing Science and Technology Pinjarra Hills, Australia, April, http://www.cat.csiro.au/cmst/staff/pic/robot. Accessed on 23 Mar 2014

[40] István Kecskés, Ervin Burkus, Péter Odry (2014) Swarm-Based Optimizations in Hexapod Robot Walking, SACI 2014 IEEE 9th International Symposium on, pp. 123-127, DOI: 10.1109/SACI.2014.6840048