Mechanical and Walking Optimization of a Hexapod Robot using PSO

Ervin Burkus*, Péter Odry**
*Óbuda University, Budapest, Hungary
**College Dunaújváros, Dunaújváros, Hungary
ervinbur@app-dsp.com, odry.peter@mail.duf.hu

Abstract—In this paper a novel solution is introduced for determining the dimensions and construction of a walking robot using an optimization method.

During our previous research and development several hexapod walking robots have been built. The latest model called Szabad(ka) II is a complex, servo motor driven, multiprocessor device made of aluminum and steel. Prior to its construction sophisticated modeling was carried out, with special attention to the smallest physical, mechanical and electrical parts. With this model the functionality of the robot could be checked before the manufacturing process.

For the time being, modeling was used only for verifying the construction, but as the next step it will be used for defining the construction and dimensions. Manually this is practically impossible, therefore the process requires automatization. The definition of the optimal parameters can be solved with optimization methods. For the task the Particle Swarm Optimization (PSO) was chosen.

I. THE CURRENT AND NEW MODEL

The Szabad(ka) II robot (Fig. 1) is described in detail here [5], [6]. The robot model contains its dynamical description, the ground contact, the behavior of the gears and joints as well as the model of the motors and control electronics. The constructed robot was later used to verify and improve the model [1], [2], [3], [4]. This improved model would be included in the total optimization of the mechanical structures of future robots. For the verification the motor currents, voltages, the real and theoretical joint angles as well as the body’s 3D accelerations were measured.

The mechanical structure of the robot was designed in SolidWorks. To create the SimMechanics model first the SolidWorks model was transferred to the Simscape/SimMechanics environment. Because the SolidWorks model was defined paying attention even to the smallest details, the model imported into the SimMechanics had the accurate dimensions, masses, and inertias of every part. The model also required adding the servo system’s electronic and mechanical modules, backlash of the joints, ground contacts, control and other modules.

Some factors such as the combined behavior of the motors and planetary reducers, and the damping parameters of the feet, could be achieved only after the robot had been constructed and concrete measurements were performed.

This model was perfectly suitable for the verification and walking simulation of Szabad(ka) II. This method can be applied in the process of designing the next robot. However in the construction of a new device the modeling should be used not only for checking and simulating but for defining the structure.

The structure of the robot can be defined by its dimensions. These are the parameters which have to be specified by the optimization. The most significant dimensions define the positions of the legs and the lengths of the leg parts.

In order to achieve this, the parameters must be easily modifiable and the simulation has to be fast. The previous model of Szabad(ka) II did not meet these requirements. Due to the high level of its complexity it is difficult to assign parameters to it and it requires high processing resources.

These reasons made it necessary to create a new model (Fig. 2) intended to be used for the optimization and entirely designed in SimMechanics. Instead of the First Generation, here the Second Generation of SimMechanics elements were used because these latter are clearer, optimized and faster [7].

The components of the robot are represented by simple primitives instead of complex parts. The parameters of these components (dimensions, density) can be set easily and the corresponding inertias are calculated by the software. It is not possible to create a realistic model, but for designing the structure that is not needed.

The model used for optimization does not have electronic modules and the control is simpler however this will not affect the usefulness of the results (torque, power), because only in their proportions are important.

This does not mean that the final model will not be realistic, merely the design of the structure does not need precise results.
II. OVERVIEW OF THE LITERATURE

In [12] a multi-criteria optimization of a hexapod machine is presented. With the help of modeling, the authors proposed several alternative designs of a hexapod machine and made an investigation with the aims to reduce flexibility and to eliminate singular kinematic configurations. In the model, no dynamic forces and torques were considered and, there was no closed-loop control.

The authors of [13] used recurrent neural networks with symbiotic species-based particle swarm optimization to evolve gaits of a hexapod robot. Here the optimization of the structure of the robot is not addressed, but the work in the field of the evolution of the gaits is remarkable.

The [14] addresses the gait synthesis in hexapod robots. Here genetic algorithms were chosen to perform the task. The paper describes the staged evolution of a complex motor pattern generator (CPG) for the control of the leg movements of a six-legged walking robot.

In [15] an interesting work is presented which may be related to our research. The authors made a robot, which can simultaneously optimize its structure and control system. This is done using an evolutionary algorithm. The main difference between this process, and the method presented in this paper is that in [15] an already constructed wheeled robot can alter the links of its mechanism on which the wheels are mounted, while in the case presented here, the robot’s structure can be changed before the manufacturing process.

III. OPTIMIZATION ALGORITHMS AND SWARMOps

Solutions to some problems are not simply correct or incorrect but are rated in terms of quality. Such problems are known as optimization problems because the goal is to find the candidate solution with the best, i.e. optimal quality.

The optimization method known as Particle Swarm Optimization (PSO) works by having a swarm of candidate solutions called particles, each having a velocity that is updated repeatedly and added to the particle’s current position to move it to a new position. [11]

Let \( \mathbf{x} \) denote the current position of a swarm particle. Then the particle’s velocity \( \mathbf{v} \) can be updated as follows (1):

\[
\mathbf{v} \leftarrow \omega \mathbf{v} + \varphi_p \mathbf{r}_p (\mathbf{p} - \mathbf{x}) + \varphi_g \mathbf{r}_g (\mathbf{g} - \mathbf{x})
\]  

(1)

where \( \omega \) is a user-defined parameter called the inertia weight and \( \varphi_p \) and \( \varphi_g \) are weights on the attraction towards the particle’s own best known position \( \mathbf{p} \) and the swarm’s best known position \( \mathbf{g} \). These are also weighted by the random numbers \( r_1, r_2 \approx U(0,1) \). In addition to this, the user has to determine the swarm-size \( S \).

Once the agent’s velocity has been computed it is added to the agent’s position (2):

\[
\mathbf{x} \leftarrow \mathbf{x} + \mathbf{v}
\]  

(2)

SwarmoPS, is an implementation of the swarm optimizations. It contains source libraries which can be used in Matlab or GNU Octave. A detailed description about Swarm algorithms can be found in [8], [9], [10].

![Figure 2. The SimMechanics model](image)

Besides SwarmoPS several other swarm optimization algorithms are available. SwarmoPS is chosen because it is fast, and its parameters are easy to set.

A comparison of genetic, swarm and other algorithms, and their usage in robotic applications are shown in [8].

IV. SETTING THE PARAMETERS OF THE SWARMOPS ALGORITHM

Before using the SwarmoPS algorithm, its parameters have to be set.

The number of the search space dimensions defines the number of the fitness function’s input variables. In this case the number of the inputs is two, this way the problem will stay relatively simple. The lower and upper bounds of the search space define the limits of the input variables. These were obtained from the robots mechanical limits.

The nontrivial parameters are:

- \( S \) - swarm size,
- \( \omega \) - inertia weight
- \( \varphi_p \) - weights on the attraction towards the particle’s own best known position 
- \( \varphi_g \) - weights on the attraction towards the swarm’s best known position

These parameters were determined by trial and error. The PSO algorithm ran several times on a designated test function while the parameter values were continuously changing.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Swarm-size</th>
<th>Omega</th>
<th>( \text{phiP} )</th>
<th>( \text{phiG} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>1 – 200</td>
<td>-1 – 1</td>
<td>-1 – 3</td>
<td>0 – 4</td>
</tr>
<tr>
<td>Chosen</td>
<td>40</td>
<td>0.2</td>
<td>0.5</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE I.
As the result parameters those parameter values were chosen with which the least amount of evaluations were needed to reach the optimal results of the test function.

As the test function, the Matlab peaks() function was used which is relatively complex, and has multiple minima and maxima points. During the testing, at a time only one parameter was changed and determined. The result had to be found with the accuracy of two decimal places.

The tests were run 100 times with each parameter and the mean of the results was considered.

Table I. shows the parameter range used in finding the optimal results and the selected values.

V. THE CONCRETE USE OF THE OPTIMIZATIONS

Developing the structure and defining its dimensions is a very broad set of problems. To demonstrate the optimization method's legitimacy a specific example was selected. However, this example has a specific benefit namely that its results can already be used in designing a new robot.

In the case of the previous robot its symmetrical arrangement caused some problems while it was walking using the tripod gait [5]. At tripod gait different forces are applied to the legs. While on one side of the robot the front and rear legs are at the same time on the ground till on the other side only the middle foot contacts the ground.

Because of this, the motors are loaded unevenly and the motors driving the femur part of the middle legs have to be much stronger. This phenomenon can be seen in the simulations results (Fig. 4).

This is the reason why the middle legs of Szabad(ka) II are different from the others.

The fact that different legs are used makes the design, calculations and manufacturing more complex and of course, increases the cost.

One solution to this problem is to change the placement of the legs, and the length of one part, for example the length of the femur. This requires a process which is able to determine the appropriate sizes. Swarm optimization is perfectly suited for this task.

All the calculations and simulations results (Fig. 4).

The fitness function needed for the optimization calls the robot simulation, in which the distance between the legs (alpha-alpha distance measured in the y direction) and the length of the femur can be set (Fig. 3). The algorithm can alter both the leg distance and femur length by +/- 50 millimeters. The simulation is about 10 seconds long, during which time the robot makes exactly two equal steps. The output of the simulation is the average of the torques needed to move the joints. From this the difference between the torques of the center and outer theta joints is calculated.

The optimization – based on experimental experience – needed maximum 400 evaluations to find best solution, at which the distribution of the load are even.

The results of the simulation can be seen on Fig. 5, 7, 8.

It can be concluded, that if the outer leg pairs are closer to each other, and if the femur parts are shorter, then the load difference between the theta2 joints is smaller (Fig. 6, 9).

The optimization algorithm was repeated several times. The best solutions (at which the difference between the torques was close to zero) were around -4, 4 centimeters (Table II).

As it can be seen on Fig. 4 optimal results can be found with several parameters.

The femur part significantly affects the walking speed therefore the optimization parameters have been refined. In the new cases the femur length varied between smaller limits. The algorithm successfully identified parameters resulting zero torque difference when the femur was reduced with 30 and 20 millimeters as well. The results are summarized in Table II.

After determining the optimal structural parameters of the robot, the walking can also be optimized. That may have several objectives, such as minimizing the movement of the legs, involved work, or uneven walking. In this case, the goal was walking the longest path per time unit, with minimum energy input. In the case of a real robot the actual use of this is prolonging the battery life.

The fitness function is the ratio between the sum of the average joint torques and the distance traveled by the robot. While mowing, the alpha, theta1 and theta2 joints can turn the same amount, and the variable parameters are added to their angles. The limits within which the joint angles may be modified are +/- π/8. The other terms and conditions are the same as they were at the previous optimization. The end result was the same even after multiple runs (Fig. 10, 10, 12).

As it can be seen on the figures the optimal solution (0.2279) is in the point where both theta1 and theta2 joints are rotated with π/8. With changing the limits or the fitness function even better results can be achieved.
VI. CONCLUSION

The above presented tests have shown how a robot structure could be defined using modeling and optimization techniques. These technologies will be definitely used prior to the construction of the next robot. The presented model cannot perform the total verification of the mechanics but, it is possible to create a detailed model which is capable of the robot’s realistic simulation. In the case of building the complete model, appropriate results can be achieved, if the optimization is fine-tuned (the parameters, and limit values are exactly specified).

ACKNOWLEDGMENT

Grateful acknowledgments to Obuda University for financing the conference registration fee of this publication.

REFERENCES


