The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Evolution of Artificial Muscle-Based Robotic Locomotion in PhysX

Kyrre Glette and Mats Hovin
University of Oslo, Department of Informatics
P.O. Box 1080 Blindern, 0316 Oslo, Norway
{kyrrehg,matsh}@ifi.uio.no

Abstract— **This paper investigates advanced features of the PhysX physics simulation engine for simulated robotic evolution, with the goal of applying the results to a real world soft robotic system which is under construction. The cloth feature in PhysX has the potential of taking into account complex dynamics while at the same time being accelerated by a graphics processing unit. As an initial approach, muscle-shaped structures are simulated with the cloth feature and employed as actuators in a robotic structure where both morphology and control parameters are subject to optimization by a genetic algorithm. A linear and a spring-damper-based model have also been applied for reference. Stable locomotion has been successfully evolved, however, attention to simulation parameters has been necessary in order to avoid simulator instability.**

I. Introduction

By applying evolutionary algorithms to robot control, interesting and efficient behavior has been obtained, especially in the case of robot locomotion [1], [2], [3]. An evolutionary setup on real robots would typically involve some mechanism restricting the movement such that the robot does not get stuck in the evaluation area, which would give different solutions different conditions for success. By introducing a software physics simulation in the evolutionary design of robots, more flexibility can be had in terms of both easier evaluation and the flexibility of being able to evolve the robot morphology. The use of such simulators has resulted in interesting virtual robots, such as the life-like locomotion of the evolved creatures in [4], and the developmentally created results from [5]. It has also been investigated how such virtual robots can be transferred to the real world through the use of rapid prototyping [6]. However, moving from a simulator with a limited model of real-world features to the real world will necessarily change the conditions for the solution, and is often referred to as the "reality gap" [7]. This motivates the need for an adaptation process for the solutions after being transferred to the real world, possibly in form of evolution adjusting locomotion control parameters.

A typical feature of robotic behavior is the rigid, abrupt nature of movements. In addition, such a movement, often caused by electrical motors, is often accompanied by noise. It would be advantageous to have more smooth, natural movements, as these could help in making robots more suitable for interaction with humans. With the improving features of physics simulators, such as soft body simulation and hardware acceleration, it becomes more possible to investigate the evolution of "soft" robot features. One example of such an investigation is shown in [8], where the authors take advantage of the graphics processing unit (GPU) hardware acceleration of soft bodies in the Nvidia PhysX [9] physics engine. It should be noted that the commonly available physics engines are primarily designed for use in computer games, and thus the simulation techniques employed are often optimized for speed and visually plausible results, rather than physical accuracy. It is therefore an open question whether the size of the reality gap for solutions evolved with such simulators is small enough to be useful for transfer to real-world designs.

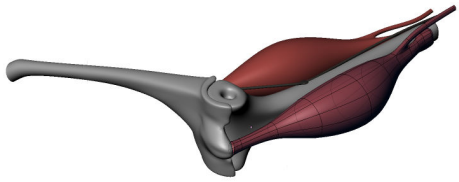


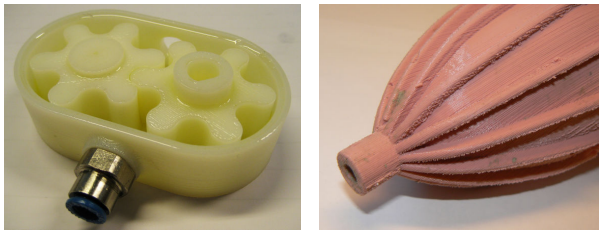
Fig. 1: Bio-inspired robotic limb. Artist's illustration.

We would like to investigate the possibilities for a more natural, biologically-inspired actuation of robotic limbs through the use of custom shape muscle-like structures. An illustration of this concept can be seen in Fig. 1. The muscle-like structures are to be made of an elastic material and filled with liquid for contraction and thus actuation of limbs. Other work on prefabricated fluidic muscles advocates the advantages of a hydraulic approach [10]. Before conducting experiments on real world structures, we investigate as a first step the use of the PhysX physics engine for simulated evolution of robot and muscle structures. The first task for the evolved robots will be forward locomotion, analogous to earlier experiments conducted with a real robot in [11].

The paper is structured as follows. Section II describes the status of the real robot design process, while Section III describes the developed robot simulator. The experimental setup and results are given in Section IV and discussed in Section V. Finally, Section VI concludes the paper.

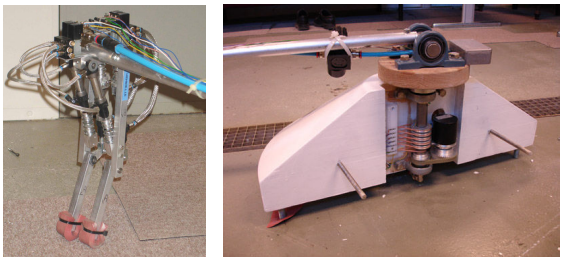
II. Real Robot Prototype

This section describes the actuator technology considered as well as the robot setup. With the real robot prototype not ready at the time of writing this section will give an overview of the achievements so far and the planned result.



(a) (b)

Fig. 2: Early prototypes of pump (a) and artificial muscle(b).



(a) (b)

Fig. 3: Mechanical setup for gait evolution. The robot (a) (here: the pneumatic version employed in earlier experiments [11]) is attached to the end of a balancing rod which is connected to a central hub (b) (here: cross section).

A. Artificial Muscle Actuator

We would like to manufacture artificial muscles from elastic materials, shaped as a container which can be filled with liquid. As the pressure in the container rises the muscle will be deformed and, with the right design, a contraction should be obtained along a desired axis. These muscles would then work in protagonist/antagonist pairs, since the substantial force will be obtained in the contraction and not the extension phase. It is envisioned that a miniature gear pump will be able to pump the liquid between the protagonist and the antagonist muscle containers. A preliminary design of the pump can be seen in Fig. 2a. At the moment, the experimental production process consists of using a 3D printer for making molds which are then filled by a silicone polymer material, resulting in an elastic shape as seen in Fig. 2b. In future work we plan to employ the Connex500 from Objet, a 3D printer capable of printing elastic materials, for direct manufacturing of the muscle. This could also make it possible to automatically integrate the robot bones with the muscles, ligaments, tendons, and hydraulic pipes all in one printer pass.

B. Biped Robot and Evolutionary Setup

As a first step, we would like to apply the muscle actuator technology to a biped robot, inspired by the robot system described in [11]. See Fig. 3. Having already developed the central hub for measuring fitness with the help of a rotation sensor, this is believed to be a good evolutionary setup before more advanced robot morphologies and fitness measurement devices are developed. One advantage of this setup is that the robot cannot get stuck in corners, nor does it require a reset to a starting position for each fitness evaluation. In addition, the rod connecting the robot to the central hub allows the evolutionary search to concentrate on a gait resulting in forward motion, without considering balance. Combined with a relatively low number of joints and therefore a low number of required muscle actuators, finding a satisfactory solution is hoped to be feasible. It could be imagined that this setup would be the first stage of an incremental evolutionary setup, where the evolution of gradually more advanced and self-sufficient locomotion is performed.

III. Simulation

This section presents the developed simulation environment and the simulation of the robot and its actuators.

A. Robot Simulation Environment

A robot simulator has been developed, based on the PhysX physics library and using OpenGL for visualization, see Fig. 5b for a screenshot. The PhysX library is primarily developed for real-time applications such as computer games, and some features (*cloth, soft bodies, fluids*) can be hardware accelerated by a GPU through the CUDA framework.

B. Muscle Actuator Models

This section describes the different muscle simulation models proposed. Since the prototypes of the artificial muscles are not yet ready it is unknown how well these models will relate to the real life behavior. A more careful design and tuning of the models will have to be performed when properties of the real life muscles are ready. It is desirable to have results from the simulation dictate parameters for the production of the real life muscles.

1) Distance Joint Model: The simplest muscle actuator model consists of a PhysX *distance joint* where the constraint is a fixed distance between points on two different dynamic objects. By updating this fixed distance, muscle contraction can be simulated. This type of actuator does not take into account the more complex dynamics of a soft hydraulic muscle, but is included as a reference actuation method. For stability reasons we chose to not have a protagonist/antagonist pair when applying this actuator, since the distance is a hard constraint and simulation instability can occur if two such forces are working against each other. In addition, this type of actuator is able to perform both contraction and extension. The contraction amount of the actuator is given in a percentage of the distance between the points, in our experiments set to 4.5%.

2) Spring Joint Model: The second muscle actuator model extends the *distance joint* by adding spring behavior to the target distance constraint. This increases the complexity of the dynamics and makes the actuator behave in a softer fashion. Setting the desired contraction amount is thus in this case not immediately followed by the same change in

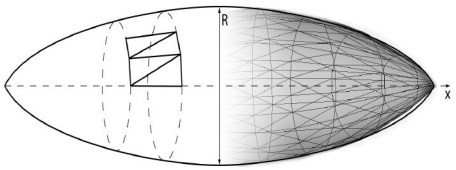


Fig. 4: Simulated muscle cloth mesh structure.

distance between the end points. The spring properties are controlled by *spring* and *damper* values. For our experiments these are set to 5000 and 0.5, respectively, which have been considered to give a relatively soft behavior.

3) Cloth and Pressure Model: The third muscle actuator model employs the cloth simulation feature of PhysX which is a position based dynamics approach [12]. A cloth is in this case a patch defined by a number of particles which are connected to each other in a triangular fashion, and the constraints which are applied to these connections. The main constraints are bending and stretching stiffness values. When having a closed cloth mesh, interior pressure can be simulated which deforms the structure. One can take advantage of this to construct an artificial muscle cloth mesh which is able to contract between the muscle endpoints. To make this contraction possible one needs to make the structure expand/stretch more in the plane which is orthogonal to the contraction axis. In the PhysX cloth model one method of achieving different stretching properties in different directions is to vary the number of cloth particles in the different directions. Few particles in one direction give a higher stiffness. These properties have been employed in order to construct a muscle-shaped closed cloth composed of N particles in the direction of the contraction axis and a higher number of particles, M, in the circumference. See Fig. 4. The radius of the shape along the contraction axis x can be defined as $r = R \times sin(x)$ where $0 \leq x \leq \pi$ and R is the radius at the widest point. In our experiments we have chosen $N = 6$, $M = 32$, $R = 1.4$, and set the *bending* and *stretching* constraints to the maximum values. The applied pressure varies between 1.0 and 2.5 atmospheres. Controlling the muscle contraction is in this case equivalent to setting the internal cloth pressure, which will in turn make the muscle shape deform, and eventually contract the end points.

C. Robot Model

The structure of the simulated biped robot is illustrated in Fig 5a. The limbs are simulated as dynamic rigid bodies, and are constrained by revolution joints which are limited to rotation along one single axis. The morphology of the robot is determined by the values of various parameters: Muscle attachment positions, resting angle of the joints, and lengths of the body parts. The two legs have the same structure.

The biped robot is then connected to a central hub by two radial rods, see Fig. 5b, restricting the robot to always stand straight up. The central hub allows the radial rods to rotate such that the robot can walk in a circle, which makes conditions similar to the real world setup. A very high number (50) of solver iterations has been applied to the bodies in order to avoid problems with joint instabilities.

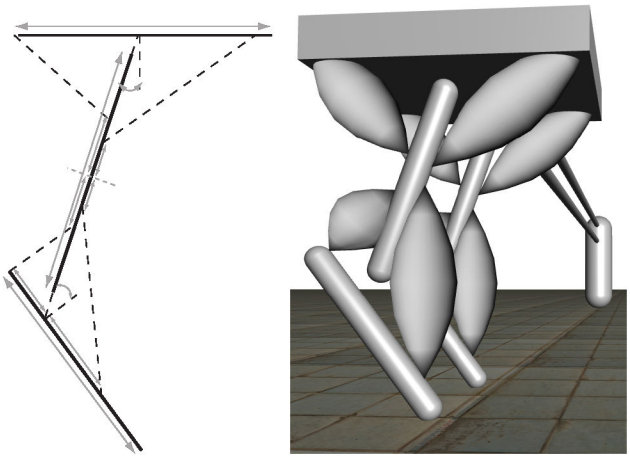


Fig. 5: The biped model. (a) Schematic side view illustrating the morphology parameters with grey arrows. (b) Simulation setup for evolution, showing the best evolved configuration.

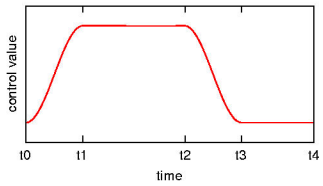


Fig. 6: Example period of the controller output.

D. Controller Model

A relatively simple muscle controller model has been chosen for producing the biped gait. An illustration of a period of the controller curve for one joint can be seen in Fig. 6. The *attack* parameter decides the time between t_0 and t_1, *pause* the time between t_1 and t_2, and *decay* the time between t_2 and t_3. The time between t_3 and t_4 is then the remaining period time given by the *frequency* parameter. The controller then repeats the same curve in a cyclic fashion. The controllers for the top and bottom joints share the *frequency* parameter, but have different *attack*, *decay* and *pause* parameters. In addition, the bottom controller is phase shifted by ϕ_0. The second leg has the same parameter values, however an additional phase shift ϕ_1 is applied, relative to the first leg.

IV. Experiments

This section describes the experiments and the results.

A. Evolutionary Algorithm Settings

For the evolutionary runs, the GAlib [13] library has been employed, running the "simple" genetic algorithm (GA) as described in [14]. The evolution runs have been run for 50 generations, with a population size of 50 and mutation

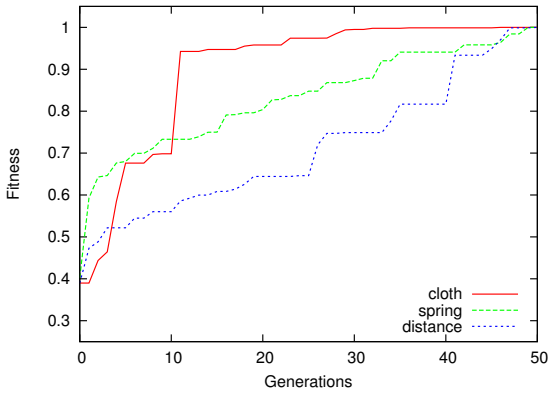


Fig. 7: Elite fitness curves for the different actuator models. Average of 3 runs and normalized for each model, i.e. the absolute fitness values differ.

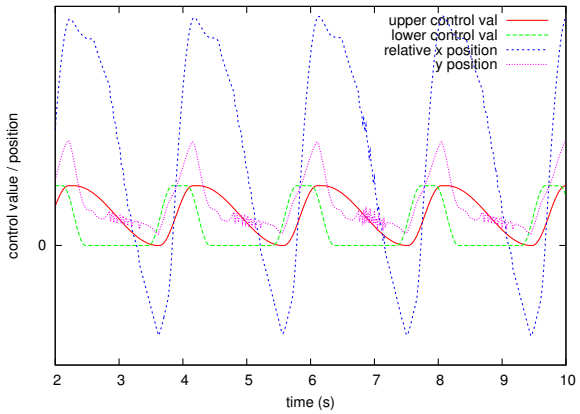


Fig. 8: Evolved distance controller and leg movement.

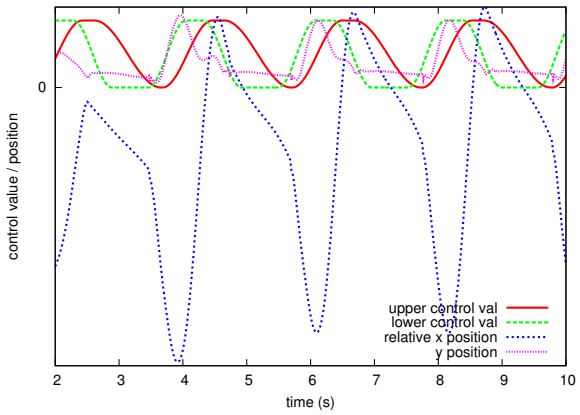


Fig. 9: Evolved spring controller and leg movement.

and crossover probabilities of 0.005 and 0.1 respectively. The genome, with a total length of 111 bits, consists of binary-coded numbers which are scaled to decimal numbers in desired parameter ranges for the robot morphology and controller.

The fitness function is based on the total distance covered by the biped robot in the positive walking direction over 700 simulation steps (11.7 s). However, a number of cut-off criterions have been introduced in order to be able to stop evaluation early and give zero fitness, such as when the robot walks backwards or stands still. In addition, certain unwanted behaviors are suppressed by stopping evaluation if certain constraints are exceeded. These constraints can be the robot moving too fast or the platform being too high or too low, and can be related to the robot breaking down or exploiting some unknown simulator feature which is unlikely to happen in the real world.

B. Evolution Runs

Evolution runs with the goal of producing a stable symmetrical biped walking gait were initiated for each of the different muscle actuator models. In this case the controller frequency was fixed with only a little adjustment margin in order to have controlled and more comparable results, and ϕ_1 had a short range around 0.5. Also, the maximum speed was limited in order to avoid jumping or similar behavior. The elite fitness curves can be seen in figure 7. During the cloth evolution runs, several invalid solutions were encountered and suppressed, most notably the cases where the structure collapsed because of the cloth stretching too much, and the robot getting stuck with both legs in a fully backwards position while still having a bouncing forward movement.

The computer running the experiments was equipped with an Intel Xeon 5160 CPU running at 3 GHz and a mid-range GPU: Nvidia GeForce GTS 250. The software simulation ran at a speed of 2260, 2167, and 100 simulation steps per second for the distance, spring, and cloth actuated robots, respectively. One simulation step corresponds to 1/60 s of simulated time. When enabling GPU hardware acceleration, the cloth actuated robot simulation runs at 234 steps per second, ie. a 2.3 times speed increase. The distance and spring based simulations do not support GPU acceleration in the current version of PhysX (2.8.3). However, GPU acceleration was disabled for the experiments due to the difficulty of finding stable parameters for the cloth simulation. A full evolution run for spring based individuals took 13 min, while a cloth based run took 131 min.

C. Evolved Gaits

The evolved gaits from the runs described in the previous section are presented in figures 8, 9, and 10. Here, the control values for the actuators as well as the resulting movement for a point on the tip of the first leg are combined in the same figure. The x position is relative to the center of the robot body, whereas the y position is relative to the ground. The absolute fitness values of the differently actuated robots are not comparable since these are highly dependent on the different actuator properties, and are thus not reported. By qualitative observation in the simulator all of the gaits seem functional and stable after an initial run-in period. It is possible to observe that the leg of the distance actuated robot steps slightly through the ground plane when pushing down and backwards. The difference of the x position in the

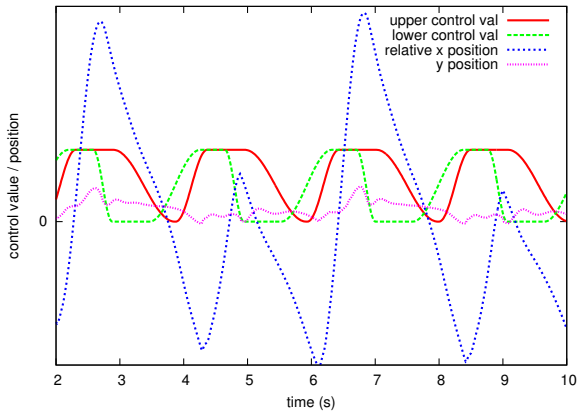


Fig. 10: Evolved cloth controller and leg movement.

TABLE I: Result of applying the evolved solutions on different simulation models.

evolved for	applied: *distance*	applied: *spring*	applied: *cloth*
distance	good	backwards	collapse
spring	very slow	good	very slow
cloth	ok	very slow	good

cloth actuated robot for every two control periods, as seen in 10, is less striking by observation in the simulator. The best evolved cloth actuator configuration can be seen in Fig. 5b.

The evolved solutions were then evaluated qualitatively when employing the other simulation models than they were evolved for. The results are summarized in Tab. I. When the cloth model was applied to the evolved distance muscle solution, the robot collapsed and could not walk, due to the muscles not being strong enough to support the evolved structure. The solution evolved for the cloth model had the overall best stability when transferred to the other models.

Another evolutionary run was conducted for the cloth actuator, where a slightly higher maximum speed was allowed in the fitness function. This resulted in a more asymmetric and unstable gait which corresponded less to the applied control values, see Fig. 11

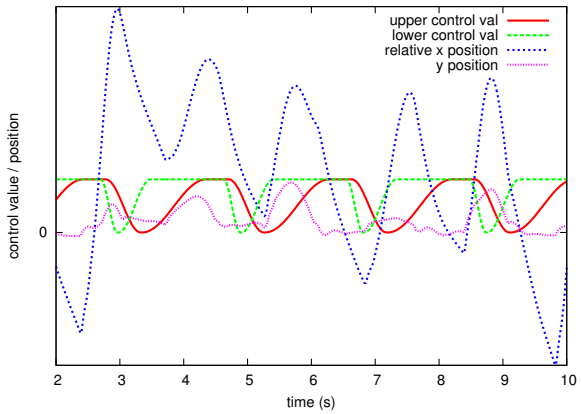


Fig. 11: Evolved cloth controller and leg movement, causing jumping behavior.

V. Discussion

This section discusses the experimental results.

A. Evolution Runs and Results

From observing the plot of the evolutionary runs for the different actuation models (Fig. 7), it seems like the advances in the elite fitness tend to be more frequent for the distance and spring actuators, but in shorter steps, than for the cloth actuator. This could indicate that the fitness landscape is more difficult for the cloth actuator, due to the complex dynamics of the cloth, and that the search thus is less directed. In addition, the large improvement steps of the cloth search may be related to the potential contraction/extension power inherent in the current cloth muscle actuator setup. This may also seem connected to the fact that the cloth actuated robot was the only reaching the ”speed limit” in the fitness function. It is unknown whether all of this actuator power is a natural phenomenon or some artifact of the simulation model, and it may seem sensible to keep the cloth parameters, such as the pressure, within a limited range in order to avoid unnatural behavior.

By limiting the frequency of the controller we have been able to produce relatively stable, but slow-paced gaits. The simulation inaccuracy causing the y position artifact for the distance actuated robot (Fig. 8) is probably due to a failure of the simulator to satisfy collision constraints when submitted to the hard constraint of the distance actuator. The disparity of the x position in the first and second controller periods of the cloth robot (Fig. 10) makes it the only model with which a ”normal” gait has not been found, and indicates more complex dynamics in the employed cloth model. This becomes even more apparent in Fig. 11, where the increase of the speed limit allows even more complex locomotion and it becomes difficult to evaluate the validity of the solution. The fact that the evolved cloth based solution is the one that performs best when changing to the other actuator models, as seen in Tab. I, may indicate that the complex dynamics of the cloth muscle has led to a more robust solution. In order to improve the smoothness of the gaits, and also possibly reduce energy consumption, the fitness function should be extended to take such qualities into account.

Although there were variations in the evolved morphologies, no particular trends have been apparent. However, from the experiment reported in Tab. I, it became apparent that the cloth based actuators are weaker and that the distance actuator based solution would collapse when using cloth actuators. This signifies that the evolved cloth solution is more stable in terms of requiring less force to hold the robot together. As these experiments primarily have focused on achieving functional solutions in the first place, it would be an interesting path for future work to explore more variations in morphology not only for the robot bodies but also the actuator shapes.

B. Simulation Models

The cloth muscle simulation model, although being a considerably more computationally intensive model than the

spring model, gives a relatively high (higher than real-time) frame rate for the simulation. This is particularly the case when GPU acceleration can be applied. A more accurate finite element model would be computationally more intensive and this would pose constraints on the number of solutions that can be evaluated. We believe it is an advantage to have "interactive" speeds for the evolution. Not only in terms of having faster evolution, but also in terms of being able to observe the behavior and progress of the evaluated solutions in an intuitive way. It also a clear advantage, for practical reasons, to be able to use only one physics library for simulating all behaviors, being both the soft actuators as well as the standard rigid bodies and dynamics.

Using the cloth feature of PhysX intuitively seems like a suitable model for an artificial hydraulics-based muscle made from an elastic material. However, it has been difficult to control the behavior of the cloth, in particular to achieve a high muscle stiffness without causing instabilities in the simulation. Increasing the pressure would seem intuitive but has the side effect of creating ghost forces which eventually make the robot hover above ground. On the other hand, increasing the mass of the cloth particles resulted in a higher perceived stiffness and actuation force, but caused instability and particle explosions when running on the GPU. It is therefore unclear whether this model will in all cases be a better choice than a spring-based muscle model. The cloth model has the potential however to take into account interaction of the muscle with the morphology of the robot, although for simplicity and stability this has been avoided to a certain degree in the current simulation by turning off muscle - rigid body collisions. A focus of future research should be to improve the simulation setup such that also this behavior can be modeled in a stable way. Also, when a prototype of the real artificial muscle system has been finished, effort should be taken in order to make the simulation models behave in a similar way, and do further investigations on which models are most appropriate.

A finite element-based simulation could be advantageous for determining the optimal shape properties of the muscle, as this would give a higher level of accuracy and control, but it could be impractical to incorporate it into the robot simulator for the evolution of morphology and controller parameters, both for speed and compatibility reasons. However, one approach could be to evolve muscle shape and parameters based on a finite element model and then transfer the properties of the resulting muscle into simpler models, such as spring or cloth, in the simulator.

The advantage of being able to apply GPU acceleration to the simulation of the cloth is clearly an advantage which speeds up the evolution significantly. However, we have not yet found good parameters for the GPU-based cloth which are robust for all individual evaluations and still stiff enough. We believe this may be related to the solver accuracy in the GPU implementation and that there are chances that such problems will be reduced in future versions of the library and/or GPU hardware. The employed GPU in these experiments, the Nvidia GTS 250, does not have double precision floating point support in CUDA, however it is unknown whether this is the reason for the instability.

VI. Conclusion and Future Work

We have explored employing an advanced feature from a modern commercial physics engine in an evolutionary robotics simulation, with transfer to real world robots as a future goal. The cloth feature seems like a promising way for the evolution of advanced soft locomotion, however further research is needed on the relation between the simulation model and the real world behavior. In addition, the stability of the simulation model is highly dependent on the cloth parameters and it has been difficult to achieve stable GPU-accelerated evolution. Future work includes, in addition to improving the stability of the simulation, a more advanced control scheme, such as recurrent neural networks, and possible feedback from bio-inspired sensors. Furthermore, it is desirable to investigate a highly automated prototyping process through the application of advanced 3D printing technology, such that a second phase in the evolution can be performed on a real-world robot.

References

[1] P. Dittrich, A. Burgel, and W. Banzhaf, "Learning to move a robot with random morphology," *Lecture Notes in Computer Science*, vol. 1468, pp. 165–178, 1998.

[2] P. Augustsson, K. Wolff, and P. Nordin, "Creation of a learning, flying robot by means of evolution," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, 2002, pp. 1279–1285.

[3] V. Zykov, J. Bongard, and H. Lipson, "Evolving dynamic gaits on a physical robot," in *Proceedings of Genetic and Evolutionary Computation Conference, Late Breaking Paper, GECCO*, vol. 4. Citeseer, 2004.

[4] K. Sims, "Evolving virtual creatures," in *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1994, pp. 15–22.

[5] J. Bongard and R. Pfeifer, "Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontogeny," in *Proceedings of The Genetic and Evolutionary Computation Conference, GECCO-2001*, 2001.

[6] H. Lipson and J. Pollack, "Automatic design and manufacture of robotic lifeforms," *Nature*, vol. 406, no. 6799, pp. 974–978, 2000.

[7] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *Advances in Artificial Life: Proc. 3rd European Conference on Artificial Life*. Springer-Verlag, 1995, pp. 704–720.

[8] J. Rieffel, F. Saunders, S. Nadimpalli, H. Zhou, S. Hassoun, J. Rife, and B. Trimmer, "Evolving soft robotic locomotion in physx," in *GECCO '09: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, 2009, pp. 2499–2504.

[9] NVIDIA, "PhysX SDK," http://developer.nvidia.com/object/physx.html.

[10] I. Boblan, R. Bannasch, H. Schwenk, F. Prietzel, L. Miertsch, and A. Schulz, "A human-like robot hand and arm with fluidic muscles: Biologically inspired construction and functionality," *Lecture notes in computer science*, pp. 160–179, 2004.

[11] L. M. Garder and M. E. Hovin, "Robot gaits evolved by combining genetic algorithms and binary hill climbing," in *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2006, pp. 1165–1170.

[12] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *Journal of Visual Communication and Image Representation*, vol. 18, no. 2, pp. 109–118, 2007.

[13] M. Wall, "GAlib: A C++ library of genetic algorithm components," http://lancet.mit.edu/ga/.

[14] D. Goldberg, *Genetic Algorithms in search, optimization, and machine learning*. Addison–Wesley, 1989.