Original Paper

Adaptive Behavior

Adaptive Behavior
2015, Vol. 23(2) 69–82

DOI: 10.1177/1059712314563620
adb.sagepub.com
SAGE

Transfer learning of gaits on a quadrupedal robot

Jonas Degrave[1], Michaël Burm[1], Pieter-Jan Kindermans[2], Joni Dambre[1] and Francis wyffels[1]

Abstract

Learning new gaits for compliant robots is a challenging multi-dimensional optimization task. Furthermore, to ensure optimal performance, the optimization process must be repeated for every variation in the environment, for example for every change in inclination of the terrain. This is unfortunately not possible using current approaches, since the time required for the optimization is simply too high. Hence, a sub-optimal gait is often used. The goal in this manuscript is to reduce the learning time of a particle swarm algorithm, such that the robot's gaits can be optimized over a wide variety of terrains. To facilitate this, we use transfer learning by sharing knowledge about gaits between the different environments. Our findings indicate that using transfer learning new robust gaits can be discovered faster compared to traditional methods that learn a gait for each environment independently.

Keywords

Quadrupedal robot, gait optimization, transfer learning, particle swarm optimization, evolutionary algorithms

1 Introduction

In recent years, compliant robots have increasingly gained interest in the scientific community. They use low-impedance mechanisms to exploit the robot's passive dynamics and nudge it into the desired behaviour, rather than enforcing a desired trajectory (Pratt, 2000). This approach is promising, as it is more closely aligned with biology, where even today animals show capabilities unmatched in robotics. Like in nature, we can make our robots more compliant by adding passive elements in the design. These can for instance be used for storing energy between steps, such as with the Lucy robot (Verrelst et al., 2005), or the added compliance can be used to have more robust control of the robot on difficult terrain (Raibert, Blankespoor, Nelson, Playter, et al., 2008).

Examples of compliant robots include for instance the robots with compliant actuators, such as the CoMaN (Tsagarakis, Li, Saglia, & Caldwell, 2011) and the Kuka-DLR lightweight arm (Bischoff et al., 2010). Notable examples of compliant robots in legged robotics are for instance the M2V2 (Pratt & Krupp, 2008), HyQ (Semini et al., 2011) and StarlETH (Hutter et al., 2012).

Despite their advantages, compliant robots are harder to control using linear control methods. Since their elastic elements show non-linear behaviour, control by classical paradigms requiring linearization of the problem prove to be less effective (Horn & Raibert, 1977). One approach to tackle this problem is to have the robot learn its behaviour and model using optimization techniques from machine learning (Bennewitz, Burgard, Cielniak, & Thrun, 2005; Deisenroth & Rasmussen, 2011; Dillmann, Rogalla, Ehrenmann, Zollner, & Bordegoni, 2000; Kohl & Stone, 2004; Peters, 2007; Waegeman, wyffels, & Schrauwen, 2012; Wolbrecht, Chan, Reinkensmeyer, & Bobrow, 2008). This way the robot inherently learns to deal with non-linearities and uncertainties in its own morphology. These uncertainties are an inherent problem in robotics, where the quality of contemporary sensors makes it impossible to know either the body of the robot or the environment in full detail.

Having said that, the main disadvantage with these techniques is that the robot needs to go through the optimization process for its behaviour repeatedly, every time the setting of the problem changes. This process takes up a lot of optimization time, and the time to test

[1]Electronics and Information Systems, Ghent University, Belgium
[2]Department of Software Engineering and Theoretical Computer Science, TU Berlin, Germany

Corresponding author:
Jonas Degrave, Sint-Pietersnieuwstraat 41, 9000 Gent, Belgium.
Email: Jonas.Degrave@UGent.be

a behaviour on the robot is usually far greater than the amount of time spent on the calculation of the optimization process. It would therefore be useful to make these optimizations more data-effective, reducing the number of behaviours to test and thus significantly reducing the amount of time needed for optimization.

A lot of research has already been done in transfer learning as a method for increasing the optimization speed in various methods of machine learning, such as neural networks and reinforcement learning (Taylor, Whiteson, & Stone, 2007). Overviews of the state of the art in many subdomains can be found in review articles by Taylor and Stone (2009) and Pan and Yang (2010). Notable results include the transfer learning of information in unlabelled images to image classification algorithms (Raina, Battle, Lee, Packer, & Ng, 2007) or the use for classification of texts (Do & Ng, 2005). Examples on the use of transfer learning in robotics include work on mobile robots (Thrun & Mitchell, 1995) and on the RoboCup soccer Keepaway problem (Taylor et al., 2007). We are unaware of research done on transferring knowledge from one gait to another.

However, the idea that transferring knowledge between tasks speeds up the optimization process is non-trivial, as transfer learning may hinder performance if the tasks are too dissimilar (Rosenstein, Marx, Kaelbling, & Dietterich, 2005). Hence, in this article we evaluate the hypothesis that the learning of locomotion in compliant robots can be speeded up by transferring knowledge from one gait motion to another in the learning process.

In order to illustrate this claim, we optimize a gait in different setups for the quadrupedal, compliant robot Oncilla (see Figure 1). In each of these setups, we use

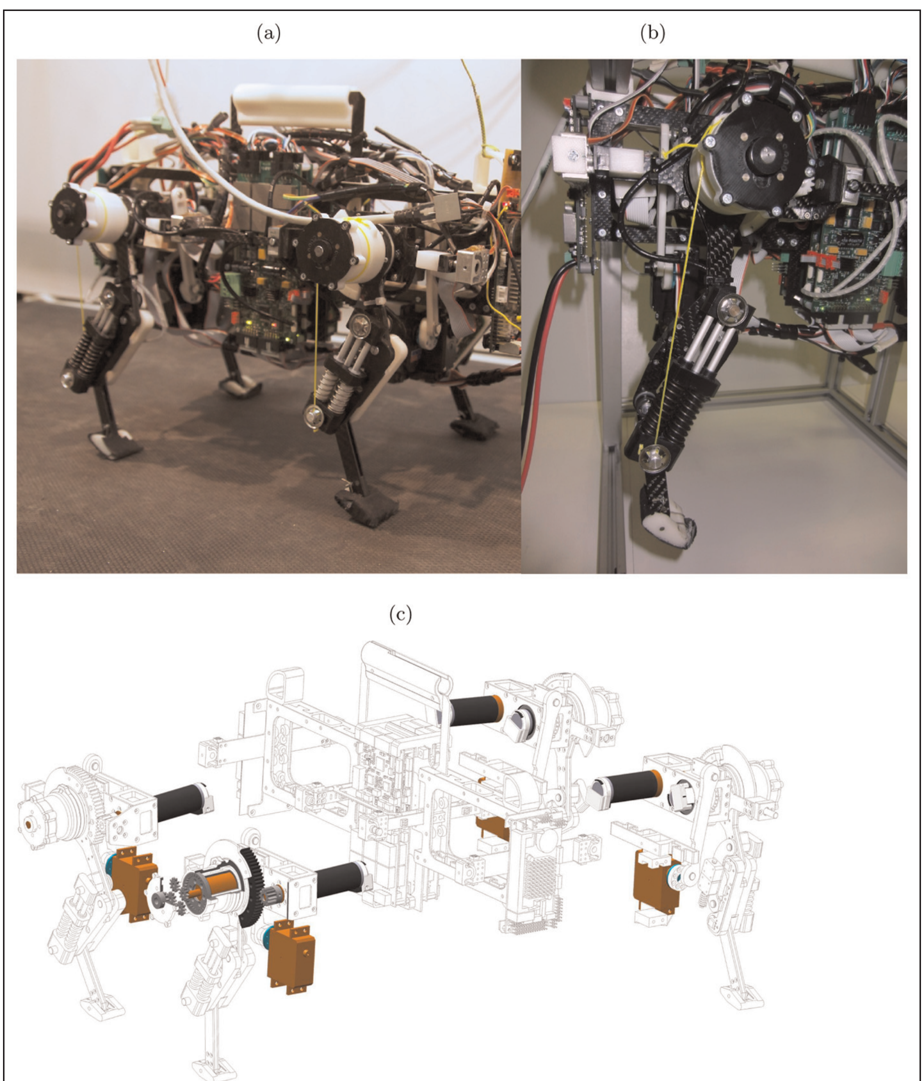


Figure 1. In (a), the Oncilla robot on the treadmill is shown. Notice the three-segmented pantographic legs and the cable mechanism actuating the knee in (b). In (c), an exploded view drawing is presented, with the three motors in each leg coloured.

particle swarm optimization (PSO) to evaluate the learning process with transfer learning, and compare it to the same learning process without the transfer learning. This way we have a baseline to compare with, and we can evaluate the effectiveness of transfer learning in increasing the learning speed.

In the following sections, we show the research we have done to evaluate whether this transfer of knowledge is beneficial. In the next section we describe the setup used for our experiments. Then we discuss the results of the experiments and find an answer to our hypothesis. Afterwards, we evaluate the results and show what this means for a broader range of applications. Finally, we conclude with a reiteration of the most interesting results and findings in this article.

2 Gait optimization

The problem of robot locomotion consists of finding the appropriate motor signals with respect to higher level constraints, such as speed or stability of the gait. There are various ways to generate motor signals. For one, it is possible to generate gaits through classical underactuated control theory. Various solutions have been developed in this terrain. The most influential example of this approach is the static balance method, in which the robot keeps the centre of gravity inside the support polygon (e.g. Rebula, Neuhaus, Bonnlander, John- son, & Pratt, 2007). A more advanced but equally important example is the dynamic balance method, such as the zero-moment-point-based technique (Byl, Shkolnik, Prentice, Roy, & Tedrake, 2009), in which the foot placement is chosen such that the resulting moment on the body becomes zero. These classical methods however have difficulties dealing with a compliant and thus uncertain morphology. They require a precise measuring of the state of the robot and its environment in order to provide accurate feedback in the motor signals. In order to avoid the problem of measuring the state of a compliant robot, we focus on open-loop gait generation in this paper, as this does not require feedback.

A first commonly used approach is to generate the motor signals by designing a path in the joint space of the robot (Iida & Pfeifer, 2004; Niehaus, Röfer, & Laue, 2007; wyffels et al., 2010) and optimizing the parameters of this path. This is the simplest approach, but we found in previous research that it is not very effective because the parameters are not very robust (Degrave, Burm, Waegeman, wyffels, & Schrauwen, 2013). Indeed, small changes in the parameters can make a gait go from stable to completely unstable.

A second approach is to use a biologically inspired approach to generate gaits (Dimitrijevic, Gerasimenko, & Pinter, 1998), based on central pattern generators (CPGs) in the joint space or (Ijspeert, 2008). These are very flexible, can create a big range of motions and they allow for a smooth gait transition between them. However, this flexibility comes at the price of an increased design complexity with respect to the parameters. In particular, each of the parameters performs multiple functions which increases the difficulty of interpreting them (Ijspeert, 2008). A single parameter of a CPG can for instance affect both the step frequency, step height and step length.

We have used a third, intermediate approach, based on previous experiments (Degrave et al., 2013). We parameterize the trajectories of the end-effector, and use inverse kinematics to generate the motor commands for these trajectories. In this way, we can use intuitive and robust parameters, while retaining the flexibility for generating a variety of gaits. Note that the inverse kinematics do not take the compliance of the robot into account, nor the interaction of its body with the environment. We only nudge our end-effector in the direction of this trajectory; we do not force it to follow that trajectory exactly. A similar approach has been done where the trajectory is generated with cycloids (Sakakibara, Kan, Hosoda, Hattori, & Fujie, 1990) and CPGs (Barasuol et al., 2013; Maufroy, Kimura, & Takase, 2010).

We have shown in previous research that cubic-Bézier-spline-based curves provide the fastest gaits we found on the Oncilla robot (Degrave et al., 2013) as well as being able to produce biologically plausible trajectories (Flower, Sanderson, & Weary, 2005; Shen & Poppele, 1995). Therefore we use these splines to generate the parameterized foot trajectories for each gait. In this approach, the parameters to be optimized (see Figure 2) are the following: the major axis $2a$, four control points of the Bézier curves P_0, P_1, P_2, P_3 and the relative position of this shape with respect to the hip of the robot (x_0,y_0). We only allow the $P_i(i = 1,\ldots,4)$ to move vertically in order to reduce the number of parameters. The lengths of the segments are denoted as $l_i(i = 1,\ldots,4)$, and l_2 and l_3 are the same for all four legs, such that the same stance trajectory is applied to each leg. In total, there are therefore 12 parameters to be optimized. Table 1 gives an overview of all the parameters that are optimized.

We generate the splines of adjacent feet in counterphase in order to have the leg pattern of a trot gait. This gait was chosen because it is stable and robust for quadrupeds (Kimura, Shimoyama, & Miura, 1989). These splines are subsequently limited to the range of the actuators. They spend an amount of time t_{stance} in the stance phase and t_{swing} in the swing phase in order to make a trajectory from these splines. These foot trajectories are then converted to motor signals using an inverse kinematics model of the robot's legs. In Figure 3, four examples of such trajectories are shown: three from the beginning of the optimization process

Table 1. An overview of the parameters to be optimized. Each particle is a vector with a value for each of these parameters. The trajectories of every gait in this paper are fully defined with these parameters. The distance of the step length is dependent on the maximal step length at the height x_0, such that the points A and B in the foot trajectory never go out of the reachable region of the robot. The horizontal distance y_0 is dependent on the step length a and on the height x_0 for the same reason.

Parameter	Description	Range
f (Hz)	The frequency of the gait	[1.5;2.0]
a (mm)	The step length	$[0;a_{max}(x_0)]$
x_0 (mm)	The vertical distance from the hip to the trajectory's centre	[126; 158]
y_{0f} (mm)	The horizontal distance from the hip to the trajectory's centre for the fore feet	$[y_{min}(x_0,a);y_{max}(x_0,a)]$
y_{0r} (mm)	The horizontal distance from the hip to the trajectory's centre for the hind feet	$[y_{min}(x_0,a);y_{max}(x_0,a)]$
t_{stance} (d.u.)	The fraction of the period of the gait spent in the stance phase	[0.4; 0.6]
l_{0f}, l_{1f} (mm)	The control points for the top of the trajectory for the fore feet	[0; 70]
l_{0r}, l_{1r} (mm)	The control points for the top of the trajectory for the hind feet	[0; 70]
l_2, l_3 (mm)	The control points for the bottom of the trajectory for all four feet	[0; 70]

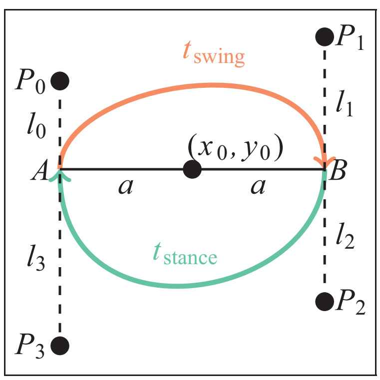


Figure 2. The foot trajectory is defined by two Bézier splines, controlled by four control points P_i. We only allow P_i to move vertically in order to reduce the number of parameters. The step length is controlled by a; t_{stance} and t_{swing} are the time in which the foot does the stance and swing part of the trajectory respectively; and x_0 and y_0 are the coordinates of the location of the centre point relative to the hip joint.

and one from the end of the optimization process on a flat terrain.

3 PSO

In order to optimize the parameterized trajectories for maximal speed, we use PSO. This evolutionary optimization algorithm uses a set of candidate solutions that move towards their own previous best solution as well as the global optimum found so far (Poli, Kennedy, & Blackwell, 2007). PSO was first intended to simulate social behaviour of human societies when processing knowledge (Kennedy, 1997). The algorithm was simplified, after which it was found to perform general optimization. By now, PSO has been used in hundreds of different applications (Poli, 2007), including robotics (Chatterjee, Pulasinghe, Watanabe, & Izumi, 2005; Pugh & Martinoli, 2008; Pugh, Martinoli, & Zhang, 2005; Qin, Sun, Li, & Ma, 2004). More closely related to this article, this technique has also been used for optimizing gaits on bipedal robots (Hemker, Stelzer, Stryk, & Sakamoto, 2009; Niehaus et al., 2007; Shafii, Aslani, Nezami, & Shiry, 2010). Evolutionary algorithms are often used in robotics because they are easy to understand and implement, because they do not require gradients, and because they are robust against noisy optimization landscapes (Kennedy & Eberhart, 1995). These benefits make them feasible for applications on real robots. We preferred PSO to other successful evolutionary algorithms such as covariant matrix adaptation evolution strategy (CMA-ES) (Hansen & Ostermeier, 1996) and genetic algorithms (GA) (Davis et al., 1991) because it places the fewest assumptions on the data. We are therefore confident that the results obtained with PSO could also be achieved by the more complex algorithms, whereas the reverse is less straightforward.

PSO models a set of particles on the fitness landscape, whose velocities perceive a noisy force towards both the particle's previous best solution, and the best solution found across all particles so far. After sufficient time, the particles have a tendency to converge to the optimum. Nevertheless, PSO is only a metaheuristic, and neither convergence nor global optimality is guaranteed. Fortunately, for optimizing gaits neither of these is necessary. Firstly, our time available on the robot is too limited to wait for full convergence as we want to have a more data-efficient approach. Secondly, since no mathematical properties of the fitness landscape are known a priori, global optimality is never guaranteed anyway.

In PSO, the update equations of particle $\mathbf{x}$, a vector containing the 12 parameters, at time step $n+1$ are defined as follows (Shi & Eberhart, 1998):

$$\begin{aligned}\mathbf{v}(n+1) &= \omega\mathbf{v}(n) \\ &\quad + \phi_p r_0 (\mathbf{p}(n) - \mathbf{x}(n)) \\ &\quad + \phi_g r_1 (\mathbf{g}(n) - \mathbf{x}(n)) \\ \mathbf{x}(n+1) &= \mathbf{x}(n) + \mathbf{v}(n+1)\end{aligned}$$

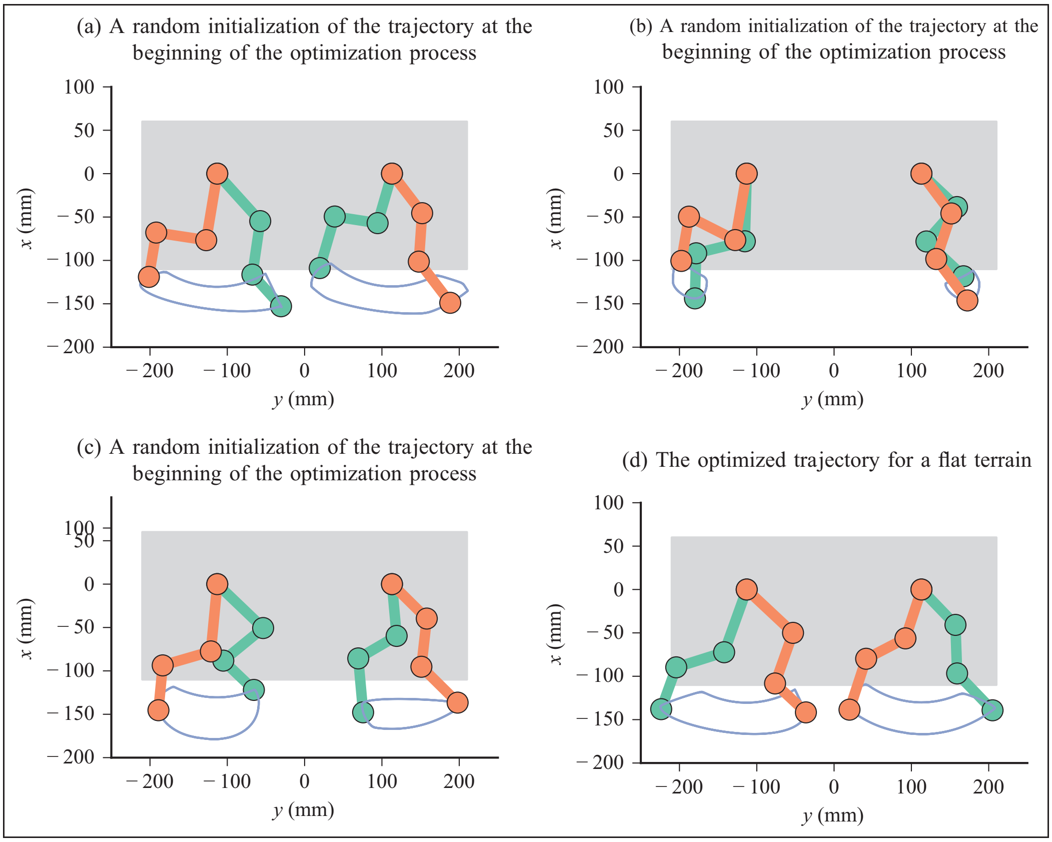


Figure 3. Example trajectories of the feet of the Oncilla robot: (a), (b), (c) a random initialization of the gait from the beginning of the optimization process; (d) the optimized gait for a flat terrain. Note that all four are idealized trajectories, determined using inverse kinematics and not taking effects of the robot's compliance into account.

At time step n, $\mathbf{x}(n)$ and $\mathbf{v}(n)$ are the locations and velocity of the particle and $\mathbf{p}(n)$ and $\mathbf{g}(n)$ are the particle's previous best solution and the global best solution over all past generations. The stochastic terms r_i are sampled from a uniform distribution between 0 and 1. This stochastic term ensures that sufficient exploration occurs (Kennedy & Eberhart, 1995). We do not copy any particles without randomization, since at the end of the process we will take the best solution found over all generations. In this paper, we use a population size of 20 particles, as our initial research showed that this amount found a good balance between the exploration and speed of the optimization.

The parameters ω, ϕ_p and ϕ_g determine the characteristics of the particle and are the meta-parameters of the algorithm. They determine the amount of exploration and the speed of convergence. Here, ω is the inertia of a particle, and ϕ_p and ϕ_g are the acceleration coefficients determining the magnitude of the random forces in the direction of the particle's personal optimum and the global optimum (Poli et al., 2007). These were set at the values $\omega = 0.66$, $\phi_p = 1.6$ and $\phi_g = 0.62$, which yield good results for our optimization problem with a 12-dimensional search space, small swarm size and limited number of evaluations, according to recent findings (Pedersen, 2010).

4 Transfer learning for PSO

We want to adapt the gait to various environments. Therefore, the robot will need to learn optimized gaits in each of these domains. However, since we optimize on our hardware and not on a model of the hardware, we need to restrict the number of trials. One way of achieving this is by reusing past knowledge in order to solve a new problem, an approach called 'transfer learning' (Pan & Yang, 2010). This way, knowledge the robot has gained of the data in a previous optimization is reused in order to spend less time optimizing.

In the original PSO algorithm, particles and their corresponding speeds are randomly initialized. To implement transfer learning, we initialize the particle population with the best particles from a previous optimization process of a similar problem. In our case, this means we take the best 20 particles over all generations from a previous optimization of a similar problem. We chose this approach over more complex approaches, where the values of the particles are taken into account to encourage dissimilar solutions, for simplicity's sake. We instead give these particles a small impulse in a random direction in order to have them start exploring new solutions ($r \in \mathcal{U}(-1, 1)$). We reckon that initializing the transferred particles with transferred impulses would excessively reduce the exploring of new solutions:

$$\mathbf{v}(0) = r$$
$$\mathbf{x}(0) = \underset{n}{\operatorname{argmax}}\,(score(\tilde{\mathbf{x}}(n)))$$

We hypothesize that this new initialization step will speed up the convergence of the optimization process by exploiting the similarity between gait optimization solutions. This is not trivial (Rosenstein et al., 2005), as it might increase the change of convergence to a local optimum because of the reduced exploration in the beginning. It is also possible that the previous gaits perform worse under the new conditions, and consequently that transferring them to this problem is disadvantageous to the optimization process. The question of whether transfer learning is beneficial to our problem of optimizing gaits is therefore the subject of this article.

5 The Oncilla robot

In order to evaluate the gaits, we use the quadrupedal, compliant robot Oncilla (Sproewitz et al., 2011). The robot has 12 degrees of freedom: each leg has a low inertia shoulder, for adduction and abduction, and hip actuator, for extension and flexion, and a third actuator actuating the knee through a cable mechanism. It has been developed for the AMARSi-project in a joint effort between EPFL in Lausanne, Switzerland and Ghent University, Belgium. The leg design was loosely based on that of a cat, using a three-segmented pantographic system to achieve dynamical properties similar to those of felines, as shown in Figure 1.

The robot is also equipped with various sensors. The hip and knee actuators are fitted with motor encoders. The heel, knee and hip-joint are equipped with magnetic encoder sensors. Additionally, the robot is equipped with a Sharp distance sensor on the front, to measure the distance to an object in front of the robot.

For the purpose of processing the signals on the robot, a Roboard RB-110 running Linux Ubuntu 10.04 with a realtime kernel is mounted on the front. Motor signals are calculated on an external computer next to the setup and communicated to the robot over ethernet. This allows us to send a new command to the robot every 10 ms. In the experiments, the robot was powered with an additional power cable loosely attached to an overhead rail. The Oncilla robot can operate autonomously as well. For autonomous use, the robot is powered using lithium polymer batteries, and the commandos are calculated on the Roboard RB-110. This mode of operation is however not suited to optimization, because the battery limits the operating time to 15 to 20 min on a 11.1 V, 1800 mAh battery, depending on the task and the gaits. This is not nearly enough, as

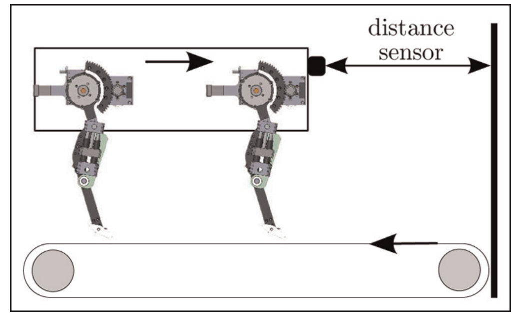


Figure 4. Schematic representation of the setup, showing the robot on the treadmill. The robot is equipped with a distance sensor to detect the wall at the end of the treadmill.

for the experiments in this article, the Oncilla robot ran nearly 6 km during the 4 hours it spent optimizing.

In order to evaluate the multitude of gaits without interruption, one for each particle in every generation, a treadmill is used, as shown in Figure 4. The measurements of the long-distance sensor on the robot are used to control the treadmill's speed, in order to keep the robot at a fixed distance from the front of the treadmill. This way, the robot can walk indefinitely and at various speeds, as long as it keeps walking in the forward direction. We test each gait for 4 s at a time, and have a smooth transition between these gaits for 2 s. Walking for more than an hour at a time poses little problem this way. In order to stop the robot from walking off the sides of the treadmill, a light thread has been added between the robot's head and an overhead rail, which limits the robot's lateral freedom of movement, while allowing it to freely move forwards. This thread is loose during normal operation. During the experiments, an assistant sits next to the track in order to intervene when the setup's safety is jeopardized by a very unstable gait.

6 Experimental setup

The goal of our experiments is to test our hypothesis that transfer learning speeds up the learning of locomotion in compliant, quadrupedal robots. To carry this out, we compare the performance of an optimization with transfer learning to the performance of an optimization without transfer learning. This way we can evaluate the benefit of transfer learning by comparing it to a baseline in an identical setting. We verified this on an experimental basis by evaluating learning speed in three different classes of problems often encountered in robotics:

1. Have a change in the environment of the robot, consisting of a different inclination of the treadmill;
2. Have a change in the front leg's stiffness, that is, a change in the robot's morphology;

3. Increase the noise in the environment of the robot, by having the robot walk over pebbly terrain.

The goal of our optimization is to maximize the average robot speed, measured over a period of 4 s. In this paper, we chose to only optimize for speed. Initial experiments showed that additionally minimizing body rotations did not have any effect on the optimization process. Since adding this stability measure to the fitness would add a meta-parameter weighing the contribution of this measure, while having no effect on the overall process, we chose to completely overhaul the stability as fitness and to only optimize for speed.

To evaluate these optimizations, we take a look at a number of metrics to evaluate the effectiveness of the optimization. Firstly, we compare the speed of the best gait in each optimization. This method compares the goals of the optimizations and is therefore necessary in the evaluation of the approaches. It is however limited, since it is very sensitive to outliers. This is particularly a problem in our application, since due to our limited population size we will have comparatively more outliers during the optimization process than if we had used a larger population size.

Therefore we also introduce a second metric, to evaluate whether all particles perform better on average than the particles of the same generation in the other optimization. To do this, we compare the speed of the gaits in each generation using a Wilcoxon rank-sum test, the non-parametric variant of the Student's t-test, which is to be used for unknown distributions and small sample sizes (Bridge & Sawilowsky, 1999). Subsequently we combine the obtained p-values using Fisher's method, a well-established method for doing so (Rosenthal, 1978). This way, we can test the significance of the claim that one optimization process is on average outperforming the other. This second method has the advantage of being less sensitive to outliers, as it uses more data and a non-parametric method. Note that we cannot say anything on the statistical significance of the experiment itself, only on the significance of the difference between the two optimization processes compared to a population of speeds randomly drawn from a population.

The data from these experiments is laid out in the following sections, and serves not only to scrutinize our hypothesis, but also to explain the mechanisms behind them.

6.1 Transfer learning for different inclinations

In the first experiment, we evaluate the effectiveness of transferring a gait from a flat terrain to a slope. We start by optimizing a gait on a flat terrain using PSO. After that we optimize a new gait on a slope twice, once with transfer learning from the gaits on a flat terrain, once without transfer learning. We compare the obtained results with each other in order to evaluate the effect of transfer learning. We perform these last two optimizations on two inclinations (a 9% and an 18% grade, or a 5.14° and a 10.2° slope), making up a total of four optimizations.

When we compare the achieved speeds of locomotion in Figure 5, it is clear that on both inclinations the parameters not only perform better during optimization when using transfer learning, but they also train faster. We compare the results of the particles per generation using a Wilcoxon rank-sum test. We find that the particles in the optimization with transfer learning on average outperform the particles without transfer learning significantly on both inclinations ($\rho < 0.001$).

As we observe in Table 2, each inclination requires a different gait for optimal locomotion. We notice that on steeper inclinations the step length is reduced while the robot spends more time with its feet on the ground. We observe as well that the robot moves its hind feet further from the body and to the back, in order to level the body more and keep the centre of mass within the support polygon.

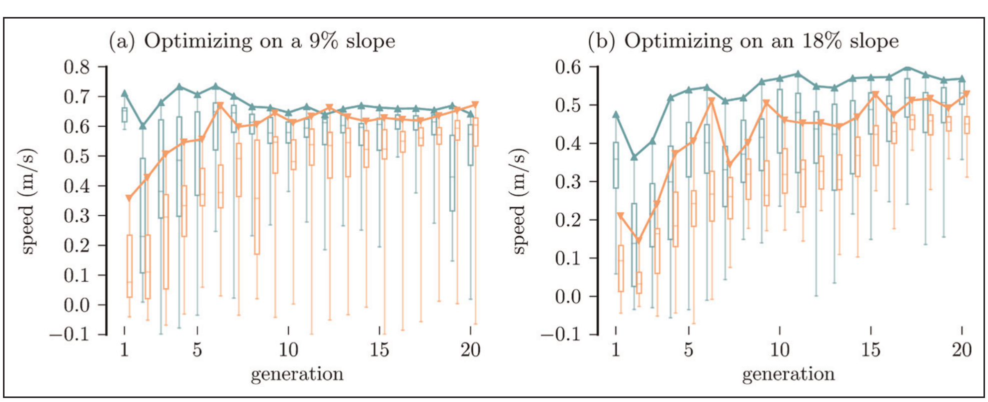


Figure 5. Visualization of the particle scores for each generation, in green (△) for with transfer learning; in orange (▽) for without. (a) The evolution of the fitness over a 9% slope. (b) Evolution of the fitness for an 18% slope. Note the generally better performance when transfer learning is used, especially when the slope is steeper and the problem is more difficult.

Table 2. The respective optimized parameters for the different terrains. You can see that the optimal step length of the robot reduces over higher slopes, while the robot moves its hind feet more to the back. Here you can see that robot spends more time in the stance phase with higher slopes, and moves its hind feet further from its body.

Grade	Transfer learning	f	a	x_0	y_{0r}	y_{0f}	t_{stance}	l_{0f}	l_{1f}	l_{0r}	l_{1r}	l_2	l_3	Speed (m/s)	Euclidean distance
Flat	×	1.98	95.3	137.0	−16.39	0.00	0.54	67.1	10.4	4.39	65.3	58.8	24.1	0.76	0
9%	×	2.00	87.9	141.8	−11.88	0.00	0.60	70.0	3.13	0.00	47.6	69.4	4.53	0.67	4.13
9%	✓	2.00	90.6	140.1	−11.71	0.00	0.54	65.7	4.06	23.9	70.0	70.0	26.2	0.74	1.72
18%	×	1.71	84.4	143.9	−8.87	0.00	0.56	52.6	34.8	0.00	51.6	66.2	4.51	0.53	6.64
18%	✓	2.00	86.3	142.8	−9.77	0.00	0.58	70.0	3.77	50.3	39.9	70.0	0.00	0.60	9.05

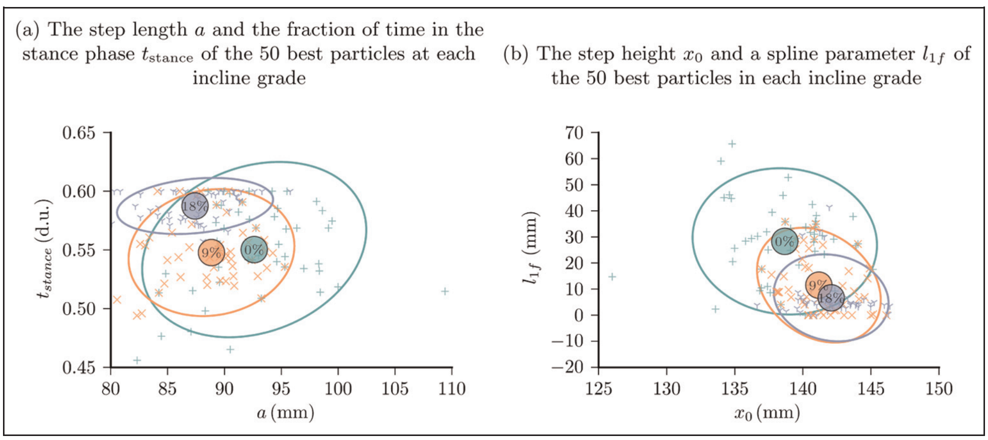


Figure 6. In these figures, we have plotted the parameters of the 50 best particles found in all our optimizations for each inclination. We have fitted a Gaussian distribution to these particles, and depicted the mean together with the two standard deviations ellipse. We see that the highest variance is found on the flat terrain, lowering with an increasing inclination. This indicates that walking on higher inclinations has a higher parameter sensitivity, and is thus more difficult to optimize. Secondly, we see that there is a relation between the parameters and the inclinations. With increasing slope, the best gaits have a longer stance phase, a shorter step length and the feet move further from the body.

When studying the best particles found during all of our optimizations, we found that there is a relation between the parameters found and the inclination on which the robot is running. As you can see in Figure 6, a higher inclination implies a lower variance in parameters. Thus, trotting on a steeper terrain requires more specific parameters. Additionally, we notice a correlation between the parameters on the different inclinations. This confirms our previous research on computer models of a quadrupedal robot (Kindermans, wyffels, Caluwaerts, Guns, & Schrauwen, 2012). The results depicted in this figure indicate that a hierarchical learning approach to gaiting, such as the one used in Kemp, Perfors, and Tenenbaum (2007), should prove fruitful. This is because the relation between the parameters can be learned, and this knowledge can subsequently speed up the learning process on intermediate or even higher inclinations. However, such an approach would lie outside the scope of this article.

In order to find out whether the method with transfer learning stays closer to the original particle, we have calculated the distances from the best particle on the flat terrain. In Table 2, we have included the Euclidean distance to the particle on the flat terrain, after normalization of the parameters. This shows that the result of the optimization with transfer learning is not necessarily closer to the original solution, which serves as an indication that enough exploration does happen. Apparently the distance from the optimal parameters on a flat terrain increases with increasing inclination as well. Looking at the speed of the optimized results, we can see that the best particles from transfer learning have a gait with a speed 9% and 14% faster on the 9% and 18% grades respectively.

These results confirm our hypothesis, namely that transfer learning increases the speed of the optimization process, resulting in better particles throughout the optimization process.

6.2 *Transfer learning for different leg spring constants*

In the second experiment, we have tested the effectiveness of transfer learning against changes in the robot's

Table 3. The respective optimized parameters for the different stiffnesses of the front leg springs.

	Transfer learning	f	a	x_0	y_{0r}	y_{0f}	t_{stance}	l_{0f}	l_{1f}	l_{0r}	l_{1r}	l_2	l_3	Speed (m/s)	Euclidean distance
Normal	×	1.98	95.3	137.0	−16.39	0.00	0.54	67.1	10.4	4.39	65.3	58.8	24.1	0.76	0
Compliant	×	1.73	96.2	136.4	−24.16	0.00	0.50	0.00	1.74	0.00	40.7	70.0	25.2	0.66	12.98
Compliant	✓	2.00	105.7	129.1	−5.27	0.00	0.60	70.0	0.00	0.00	70.0	70.0	70.0	0.75	7.16

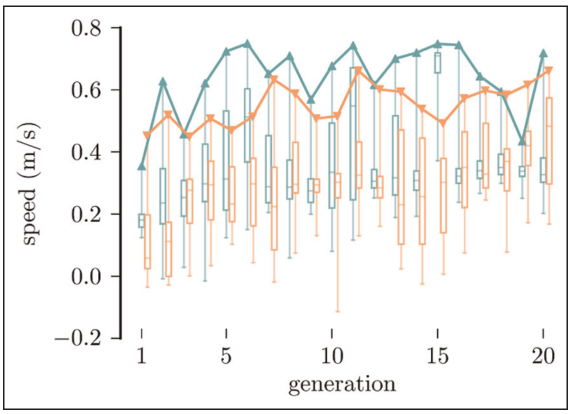


Figure 7. Visualization of the particle scores for each generation when optimizing for a reduced stiffness, in green (△) for with transfer learning, in orange (▽) for without. Notice that despite the original particles performing worse, the overall optimization is still better. This means that the better starting position of the transferred particles is not the main reason they perform better.

body parameters. To do so, we have reduced the spring constant in the front legs of the robot by removing one of the two springs in each of the robot's front legs. These springs can be seen in Figure 1(b). This reduction of the front leg stiffness results in a decrease of their pushing capacity. We use the same procedure as we did in the previous experiment with different inclinations, comparing the optimization results with and without transfer learning.

If we look at the data in Figure 7, we observe that the original transferred particles do not perform as well as the random particles in the new task. However, the improvement of the particles in the first generations is still higher than the one of the particles without transfer learning. This is remarkable, because in order to perform better, the transferred particles not only have to learn faster than their counterparts, but they also have to overcome the disadvantage of starting with a worse gait. We will offer an explanation for this observation in a dedicated subsection later in the paper.

Using the same statistical method as before, we find that the particles in the optimization process with transfer learning run significantly larger distances ($\rho < 0.05$).

If we compare the best parameters, as shown in Table 3, we can see that the best particle of the optimization with transfer learning has a 13% faster gait than in the optimization process without transfer learning. This is despite the fact that the solution is quite different from the solutions with the normal stiffness which were transferred originally, as can be seen in Table 3. The Euclidean distance of the optimized solution to the particle with the normal stiffness is relatively large, both with and without transfer learning. We will offer an explanation for this behaviour in Section 7.

Once more, these results confirm our claim that transfer learning increases the speed of the optimization process. Equivalently, the gaits perform better after a given time of optimizing. Moreover, this experiment shows that in transfer learning there is a mechanism at work besides having a head start.

6.3 Transfer learning for difficult terrains

As a third experiment, to further scrutinize our hypothesis, we have optimized in a noisier environment. In noisy problems, the PSO will produce more robust parameters in order to find gaits that perform well, because the same parameters do not always obtain the exact same result. An ill-placed pebble could potentially tip the robot over, while there might be no problem if the pebbles are arranged slightly differently. In order to do this, we had the robot walking over a flat surface covered in pebbles. We attached a pebble dispenser to our treadmill, which covered the treadmill in pebbles with a diameter of approximately 1 cm. A video of this setup is available online (http://youtu.be/kcBBdwwYmQA).

If we look at the results in Figure 8, we see that the original transferred particles perform well in this new task. However, once the PSO algorithm starts exploring, it is hard to return to these original good solutions. If we compare both optimization processes using the same method as before, we find that we cannot confirm our hypothesis. Over all generations together, the particles optimized with transfer learning do not perform significantly better than the particles without transfer learning ($\rho > 0.05$).

If we look at the best particles in Table 4, we see that the one from the transfer-learned optimization does not

Table 4. The respective optimized parameters on rocky and normal terrain, with and without transfer learning. We have also added the parameters of the second-best particle in the transfer learning case, because the best particle was part of the first generation.

Terrain	Transfer learning	f	a	x_0	y_{0r}	y_{0f}	t_{stance}	l_{0f}	l_{1f}	l_{0r}	l_{1r}	l_2	l_3	Speed (m/s)	Euclidean distance
Normal	×	1.98	95.3	137.0	−16.39	0.00	0.54	67.1	10.4	4.39	65.3	58.8	24.1	0.76	0
Pebbly	×	2.00	87.3	142.2	3.44	0.00	0.49	0.00	70.0	60.6	70.0	70.0	70.0	0.69	25.82
Pebbly	✓	2.00	89.3	141.0	−18.48	0.00	0.59	54.6	34.9	0.08	64.9	55.2	22.0	0.74	2.44
Pebbly	✓	2.00	94.8	137.3	−8.29	0.00	0.57	70.0	29.0	0.00	63.2	56.4	28.7	0.74	1.25

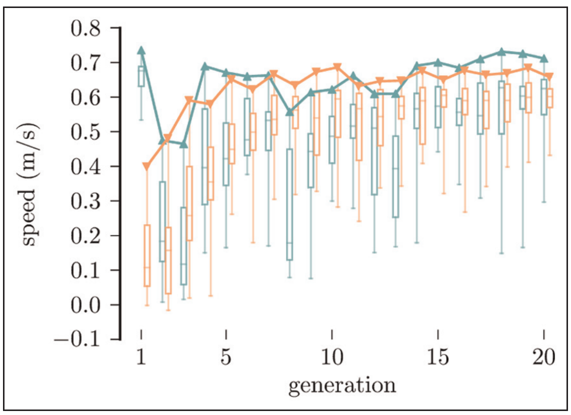


Figure 8. In this figure we visualize the scores of the particles for each generation when optimizing for a more difficult terrain, in green (△) for with transfer learning, in orange (▽) for without. Notice that despite all the original particles outperforming the random particles in the first generation, this head start is quickly lost during the rest of the optimization process.

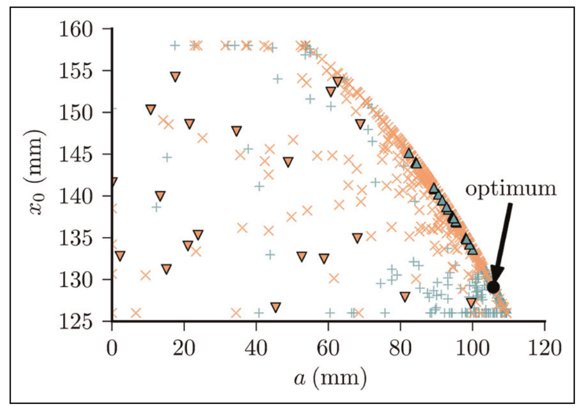


Figure 9. A chart showing the values of the step length a and the distance from the body x_0 evaluated during the PSO. The parameters of two optimization processes are shown: the green particles (+, △) have been optimized with transfer learning, and the orange particles (×, ▽) have not. The triangles (△, ▽) show the parameters at the initialization of the PSO. The rightmost edge of the search space is not vertical, because a_{max} depends on x_0 as explained by Table 1. Notice how the transferred parameters reduce the search space for the optimization algorithm, whereas the randomly initialized parameters still need to explore the entire parameter space before finding that maximizing the step length results in faster gaits. Note that most particles lie on the edge of the parameter range, which is also the end of the robot's range of motion. When the particles have been transferred, there is still a little exploration around the entire parameter space, but most particles are centred around the optimum.

differ a lot from the original optimization on the flat terrain. The Euclidean distance to the original particle is comparatively small. This is because the best particle found during the entire optimization was one of the transferred particles. If we look at the best particle in the same optimization, but exclude the transferred particles, we find that it is very similar to these solutions as well.

7 The mechanism behind the speedup

The previous results show that the optimization time decreases when using particles from similar but different problems in the initialization step. Our experiment with the changed stiffness indicates that this is not necessarily caused by the transferred particles scoring well in the new problem. The experiment on the pebbly terrain indicates the same thing. Even though these particles started off better, they did not outperform their counterparts without transfer learning. In this section we show an alternative mechanism behind the transfer learning, namely that the learning process is speeded up because the transferred particles indicate useful areas of the parameter space to explore.

To explain this, we plot all tested parameter combinations of the optimization on the 18% grade inclination. As you can see in Figure 9, it is more fruitful to maximize the step length a and have the feet moving at a medium distance from the body. You can see that in the transferred parameters this general idea is already contained in the original particles, while the random parameters still need to discover this relation. So even though the transferred particles do not perform well, they already contain this general idea. Therefore, the search space is reduced. This way, the optimization process has to spend less time on rediscovering this

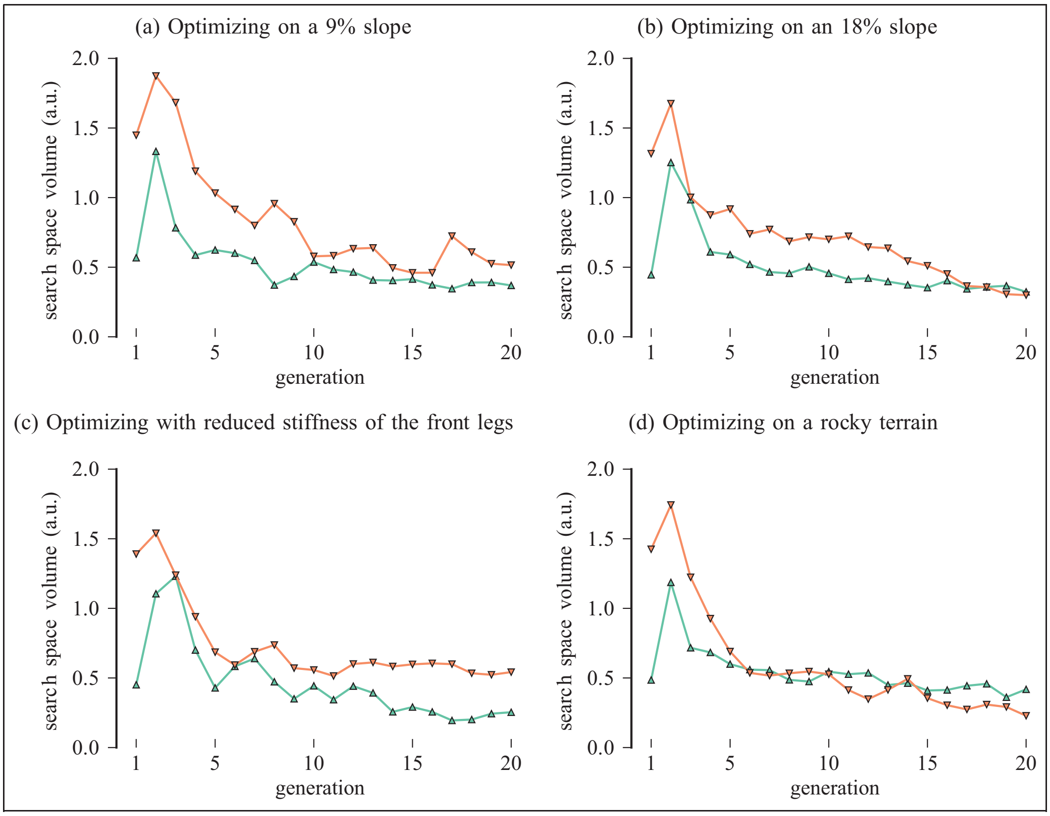


Figure 10. The size of the search space is plotted here as the geometric mean of the lengths of the principal component axis of the particle parameters for each generation, in green (△) for with transfer learning, and in orange (▽) for without. The lower this mean, the smaller the volume of the parameter space which is being explored in that generation. As you can see, the transferred particles initially occupy a smaller part of the parameter space, and this head start is not often lost during the optimization process. This is an important second mechanism through which the optimization speed is increased.

relationship, and can focus on the more fruitful areas of the parameter space.

To make this clearer, we have plotted in Figure 10 the mean size of the principal component axes of the particles throughout the optimization process. These axes are an indicator of the size of the part of the parameter space that is being explored in a particular generation. The size of the search space with transfer learning is on average 1.7 times smaller. Moreover, the transferred parameters start with a search space which is about the same size as at the end of the optimization without transfer learning. This indicates again that the increased learning speed when optimizing with transferred particles can be explained by the reduced search space in which the search starts.

8 Conclusion

In this article, we have found that transfer learning is beneficial for learning gaits on our legged robot. In order to test our hypothesis that transferring particles from previous optimizations improves the speed of learning new gaits on different problems, we compared these optimizations with optimizations starting from random samples. We optimized a gait for our robot in three different settings: first on a 9% and an 18% grade inclination, then with a different leg stiffness and finally on a pebbly terrain. We found that using transfer learning results in better gaits than when not using transfer learning in all tested cases, evaluated on a real robot. Everything considered, we conclude that transferring particles resulted in better gaits in the same amount of robot time in three out of four cases, while not harming the optimization speed in the fourth case.

We have also shown that the increased learning speed is caused by a reduction of the volume in the parameter space that is used in the exploration. Therefore, we believe that our approach is more generally applicable and will prove useful for further developments in walking robots.

How generally applicable this conclusion is cannot be reliably determined from this study alone, since only a single robot and a limited set of walking conditions was used to obtain the data. We did however identify a mechanism, alter the robot parameters and the robot surroundings to test the robustness of this method and found similarities between the results of the different optimizations, which gives us confidence for a more general applicability.

The research reported here indicates that this idea of transferring particles is more broadly applicable in PSO. In general, it would be interesting to verify whether the mechanism of the reduced search space is also the main mechanism when applying transfer learning in other optimization algorithms. Especially because of this mechanism, we believe that this approach might be beneficial for other learning problems in robotics as well, where the number of evaluations in an optimization is inherently limited and tasks are often similar, but solutions to one problem will not often work well in another.

Acknowledgements

We would like to thank Benjamin Schrauwen for the fruitful initial discussions and his practical advice during the experiments.

Funding

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013, Challenge 2: Cognitive Systems, Interaction, Robotics (grant agreement number 248311 – AMARSi), and from the Agency for Innovation by Science and Technology in Flanders (IWT).

References

Barasuol, V., Buchli, J., Semini, C., Frigerio, M., De Pieri, E. R., & Caldwell, D. G. (2013). A reactive controller framework for quadrupedal locomotion on challenging terrain. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2554–2561).

Bennewitz, M., Burgard, W., Cielniak, G., & Thrun, S. (2005). Learning motion patterns of people for compliant robot motion. *The International Journal of Robotics Research (IJRR)*, *24*(1), 31–48.

Bischoff, R., Kurth, J., Schreiber, G., Koeppe, R., Albu-Schäffer, A., Beyer, A., & …Hirzinger, G. (2010). The KUKA-DLR lightweight robot arm: A new reference platform for robotics research and manufacturing. In *41st international symposium on robotics (ISR) and 6th German conference on robotics (ROBOTIK)* (pp. 1–8).

Bridge, P. D., & Sawilowsky, S. S. (1999). Increasing physicians' awareness of the impact of statistics on research outcomes: Comparative power of the t-test and Wilcoxon rank-sum test in small samples applied research. *Journal of Clinical Epidemiology*, *52*(3), 229–235.

Byl, K., Shkolnik, A., Prentice, S., Roy, N., & Tedrake, R. (2009). Reliable dynamic motions for a stiff quadruped. In *Experimental robotics* (pp. 319–328).

Chatterjee, A., Pulasinghe, K., Watanabe, K., & Izumi, K. (2005). A particle-swarm-optimized fuzzy-neural network for voice-controlled robot systems. *IEEE Transactions on Industrial Electronics*, *52*(6), 1478–1489.

Davis, L., et al. (1991). *Handbook of genetic algorithms* (Vol. *115*). New York, NY: Van Nostrand Reinhold.

Degrave, J., Burm, M., Waegeman, T., wyffels, F., & Schrauwen, B. (2013). Comparing trotting and turning strategies on the quadrupedal Oncilla robot. In *IEEE international conference on robotics and biomimetics (ROBIO-2013)*.

Deisenroth, M., & Rasmussen, C. E. (2011). PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 465–472).

Dillmann, R., Rogalla, O., Ehrenmann, M., Zollner, R., & Bordegoni, M. (2000). Learning robot behaviour and skills based on human demonstration and advice: The machine learning paradigm. In *International symposium on robotics research (ISRR)* (Vol. *9*, pp. 229–238).

Dimitrijevic, M. R., Gerasimenko, Y., & Pinter, M. M. (1998). Evidence for a spinal central pattern generator in humans. *Annals of the New York Academy of Sciences*, *860*(1), 360–376.

Do, C., & Ng, A. (2005). Transfer learning for text classification. In *NIPS*.

Flower, F., Sanderson, D., & Weary, D. (2005). Hoof pathologies influence kinematic measures of dairy cow gait. *Journal of Dairy Science*, *88*(9), 3166–3173.

Hansen, N., & Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the IEEE international conference on evolutionary computation* (pp. 312–317).

Hemker, T., Stelzer, M., von Stryk, O., & Sakamoto, H. (2009). Efficient walking speed optimization of a humanoid robot. *The International Journal of Robotics Research*, *28*(2), 303–314.

Horn, B. K., & Raibert, M. H. (1977). Configuration space control. Massachusetts Institute of Technology.

Hutter, M., Gehring, C., Bloesch, M., Hoepflinger, M., Remy, C. D., & Siegwart, R. (2012). StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion. In *15th international conference on climbing and walking robot – CLAWAR* 2012.

Iida, F., & Pfeifer, R. (2004). Cheap rapid locomotion of a quadruped robot: Self-stabilization of bounding gait. In *Intelligent autonomous systems* (Vol. *8*, pp. 642–649).

Ijspeert, A. J. (2008). Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks*, *21*(4), 642–653.

Kemp, C., Perfors, A., & Tenenbaum, J. B. (2007). Learning overhypotheses with hierarchical Bayesian models. *Developmental Science*, *10*(3), 307–321.

Kennedy, J. (1997). The particle swarm: Social adaptation of knowledge. In *IEEE international conference on evolutionary computation* (pp. 303–308).

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of the IEEE international conference on neural networks* (Vol. *4*, pp. 1942–1948).

Kimura, H., Shimoyama, I., & Miura, H. (1989). Dynamics in the dynamic walk of a quadruped robot. *Advanced Robotics*, *4*(3), 283–301.

Kindermans, P.-J., wyffels, F., Caluwaerts, K., Guns, B., & Schrauwen, B. (2012). Towards incorporation of hierarchical Bayesian models into evolution strategies for quadruped gait generation. In *21st annual Belgian–Dutch* *conference on machine learning (BeNeLearn & PMLS 2012)* (p. 70).

Kohl, N., & Stone, P. (2004). Policy gradient reinforcement learning for fast quadrupedal locomotion. In *IEEE international conference on robotics and automation (ICRA)* (Vol. *3*, pp. 2619–2624).

Maufroy, C., Kimura, H., & Takase, K. (2010). Integration of posture and rhythmic motion controls in quadrupedal dynamic walking using phase modulations based on leg loading/unloading. *Autonomous Robots*, *28*(3), 331–353.

Niehaus, C., Röfer, T., & Laue, T. (2007). Gait optimization on a humanoid robot using particle swarm optimization. In *Proceedings of the second workshop on humanoid soccer robots (IEEE-RAS)*.

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Pedersen, M. E. H. (2010). *Good parameters for particle swarm optimization*. Technical Report HL1001, Hvass Laboratories, Copenhagen, Denmark.

Peters, J. R. (2007). *Machine learning of motor skills for robotics*. Unpublished doctoral dissertation, University of Southern California, CA.

Poli, R. (2007). *An analysis of publications on particle swarm optimization applications*. Technical Report CSM-469, Department of Computer Science, University of Essex, UK.

Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization. *Swarm Intelligence*, *1*(1), 33–57.

Pratt, G. A. (2000). Legged robots at MIT: What's new since Raibert? *IEEE Robotics & Automation Magazine*, *7*(3), 15–19.

Pratt, J., & Krupp, B. (2008). Design of a bipedal walking robot. In *SPIE defense and security symposium*.

Pugh, J., & Martinoli, A. (2008). Distributed adaptation in multi-robot search using particle swarm optimization. In: M. Asada, J. C. T. Hallam, J.-A. Meyer & J. Tani (Eds.), *From animals to animats 10* (pp. 393–402). New York, NY: Springer.

Pugh, J., Martinoli, A., & Zhang, Y. (2005). Particle swarm optimization for unsupervised robotic learning. In *Proceedings of IEEE swarm intelligence symposium (SIS)* (pp. 92–99).

Qin, Y., Sun, D., Li, N., & Ma, Q. (2004). Path planning for mobile robot based on particle swarm optimization. *Robot*, *26*(3), 222–225.

Raibert, M., Blankespoor, K., Nelson, G., Playter, R., et al. (2008). BigDog, the rough-terrain quadruped robot. In *Proceedings of the 17th IFAC world congress* (pp. 10,823–10,825).

Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007). Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th international conference on machine learning* (pp. 759–766).

Rebula, J. R., Neuhaus, P. D., Bonnlander, B. V., Johnson, M. J., & Pratt, J. E. (2007). A controller for the LittleDog quadruped walking on rough terrain. In *IEEE international conference on robotics and automation (ICRA)* (pp. 1467–1473).

Rosenstein, M. T., Marx, Z., Kaelbling, L. P., & Dietterich, T. G. (2005). To transfer or not to transfer. In *NIPS 2005 workshop on transfer learning* (Vol. *898*).

Rosenthal, R. (1978). Combining results of independent studies. *Psychological Bulletin*, *85*(1), 185.

Sakakibara, Y., Kan, K., Hosoda, Y., Hattori, M., & Fujie, M. (1990). Foot trajectory for a quadruped walking machine. In *IEEE international workshop on intelligent robots and systems* (pp. 315–322).

Semini, C., Tsagarakis, N. G., Guglielmino, E., Focchi, M., Cannella, F., & Caldwell, D. G. (2011). Design of HyQ – a hydraulically and electrically actuated quadruped robot. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*. Epub ahead of print 1 August 2011. DOI: 0959651811402275.

Shafii, N., Aslani, S., Nezami, O. M., & Shiry, S. (2010). Evolution of biped walking using truncated Fourier series and particle swarm optimization. In: J Baltes, M. G. Lagoudakis, T. Naruse & S. S. Ghidary (Eds.), *RoboCup 2009: Robot soccer world cup XIII* (pp. 344–354). New York, NY: Springer.

Shen, L., & Poppele, R. E. (1995). Kinematic analysis of cat hindlimb stepping. *Journal of Neurophysiology*, *74*(6), 2266–2280.

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. In *The 1998 IEEE international conference on evolutionary computation proceedings* (pp. 69–73).

Sproewitz, A., Kuechler, L., Tuleu, A., Ajallooeian, M., D'Haene, M., Moeckel, R., & Ijspeert, A. J. (2011). Oncilla robot: A light-weight bioinspired quadruped robot for fast locomotion in rough terrain. In *Proceedings of the fifth international symposium on adaptive motion on animals and machines*.

Taylor, M. E., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *The Journal of Machine Learning Research*, *10*, 1633–1685.

Taylor, M. E., Whiteson, S., & Stone, P. (2007). Transfer via inter-task mappings in policy search reinforcement learning. In *Proceedings of the 6th international joint conference on autonomous agents and multiagent systems* (p. 37).

Thrun, S., & Mitchell, T. M. (1995). *Lifelong robot learning*. New York, NY: Springer.

Tsagarakis, N. G., Li, Z., Saglia, J., & Caldwell, D. G. (2011). The design of the lower body of the compliant humanoid robot "cCub". In *IEEE international conference on robotics and automation (ICRA)* (pp. 2035–2040).

Verrelst, B., Van Ham, R., Vanderborght, B., Daerden, F., Lefeber, D., & Vermeulen, J. (2005). The pneumatic biped "Lucy" actuated with pleated pneumatic artificial muscles. *Autonomous Robots*, *18*(2), 201–213.

Waegeman, T., wyffels, F., & Schrauwen, B. (2012). Feedback control by online learning an inverse model. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(10), 1637–1648.

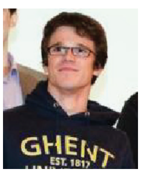
Wolbrecht, E. T., Chan, V., Reinkensmeyer, D. J., & Bobrow, J. E. (2008). Optimizing compliant, model-based robotic assistance to promote neurorehabilitation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *16*(3), 286–297.

wyffels, F., D'Haene, M., Waegeman, T., Caluwaerts, K., Nunes, C., & Schrauwen, B. (2010). Realization of a passive compliant robot dog. In *3rd IEEE RAS and EMBS international conference on biomedical robotics and biomechatronics (BioRob)* (pp. 882–886).

About the Authors

Jonas Degrave received an MSc in electronics engineering at Ghent University in 2012, where he is currently pursuing a PhD in computer science. His research interests are focused on machine learning and how it can be applied to robotics in order to generate more efficient locomotion.

Michaël Burm received an MSc in electronics engineering from Ghent University in 2013, after which he joined Reservoir Lab for a year of research in quadrupedal robots, improving on the research done in his Masters dissertation. He now works in industry, applying his interest in electronics and mechatronics to innovation and new product development.



Pieter-Jan Kindermans received an MSc in computer science from Ghent University in 2010 and received his PhD in computer science in 2014 on the topic of machine learning. He is currently a postdoctoral researcher at the Technische Universität Berlin.

Joni Dambre obtained an MSc in electronics engineering and a PhD in computer engineering at Ghent University, where she is currently a tenured professor. She specializes in brain-inspired computing techniques, both in analogue hardware and in software, building upon neural networks and the reservoir computing concept. Her research targets applications in a wide range of domains such as robotics, photonics, healthcare and the classification of video features. She is currently responsible for the coordination of the UGent Reservoir Lab in the electronics and information systems department.

Francis wyffels was born in Belgium in 1983. He received an MSc in computer engineering in 2007 and his PhD in 2013 from Ghent University on the topic of how to exploit properties of reservoir computing systems for robot locomotion and time series prediction. He is currently a postdoctoral researcher at the electronics information system department at Ghent University.