Genetic Programming Applied to Biped Locomotion Control with Sensory Information

César Ferreira[1], Pedro Silva[1], João André[1], Cristina P. Santos[1] and Lino Costa[2]
[1]*Industrial Electronic Department, University of Minho, Azurem Campus, Guimarães, Portugal*
[2] *Production Systems Department, University of Minho, Gualtar Campus, Braga, Portugal*

Keywords: Biped Locomotion, CPG (Central Pattern Generator), Sensory Information.

Abstract: Generating biped locomotion in robotic platforms is hard. It has to deal with the complexity of the tasks which requires the synchronization of several joints, while monitoring stability. Further, it is also expected to deal with the great heterogeneity of existing platforms. The generation of adaptable locomotion further increases the complexity of the task.
In this paper, Genetic Programming (GP) is used as an automatic search method for motion primitives of a biped robot, that optimizes a given criterion. It does so by exploring and exploiting the capabilities and particularities of the platform.
In order to increase the adaptability of the achieved solutions, feedback pathways were directly included into the evolutionary process through sensory inputs.

1 INTRODUCTION

There is an increasing interest in building autonomous systems to aid humans performing tasks in a wide variety of situations. Ranging from space and deep ocean exploration, or rescue missions in hazardous environments, to in everyday tasks, such as cleaning the house or taking care of the elderly. In most of these cases, legged locomotion may provide for an advantage over wheeled or tracked robots. It offers a higher level of flexibility required in a wide variety of terrains and the ability to deal with harsher terrain features, e.g. stairs, obstacles, uneven or irregular terrain. Particularly, biped locomotion provides the flexibility to a world shaped for humans. The control and generation of biped locomotion for the ever improving biped robots is a very demanding task, addressing complex problems as the generation of the movements and coordination between many degrees of freedom, balancing, perception and planning, and disturbance rejection.

Typical solutions to the problem of biped locomotion make extensive use of the knowledge of the robot and environment. Generally a plan of the path and foot placement sequence is determined, then the required motions are computed using the robot's kinematical model, respecting determined constraints established through some stability criterion, as the popular Zero-Moment Point (Vukobratović and Borovac, 2004). However, such approach requires a good perception of the environment which may hamper the general application to different dynamic environments.

Alternatively to these typical solutions, bio-inspired approaches have been researched and proposed with quite successful results. One of these approaches uses the concept of Central Pattern Generators (CPGs), exploiting the interesting characteristics of intraspinal neural networks in vertebrates (Ijspeert, 2008). These generate rhythmic activation for walking motor patterns. The main characteristic that motivates for the application of CPGs in the generation of robotic legged locomotion is the ability to adapt and correct the locomotion by the integration of sensory feedback pathways (Kim et al., 2011). This provides the ability for the robots to tackle unexpected disturbances and not completely known environments. However, there is no established framework for designing CPG solutions and such sensory feedbacks.

Previously, we have proposed a CPG based solution for biped locomotion (Matos and Santos, 2012). It combines a small set of motion primitives within CPGs driven by phase oscillators, producing basic but very capable biped walking for the DARwIn-OP humanoid robot. Despite the simplicity of the solution, the expansion of the repertoire of motion prim-

Ferreira C., Silva P., André J., P. Santos C. and Costa L..
Genetic Programming Applied to Biped Locomotion Control with Sensory Information.
DOI: 10.5220/0005062700530062
In *Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2014), pages 53-62
ISBN: 978-989-758-039-0

itives to broaden the locomotor behaviors has proven complex, as well as the design of feedback mechanisms for the adaptation and correction of locomotion. Some authors tackle this problem through imitation and learning from demonstration (Nakanishi et al., 2004), optimization of parameterized trajectories (Kim et al., 2009) or reinforcement learning (Sugimoto and Morimoto, 2011). In this work, we take a distinct approach, where the goal is to apply Genetic Programming to the automatic exploration of: 1) the motion primitives within the CPG, and 2) the integration of sensory inputs into feedback mechanisms for the adaptability to the environment.

Evolutionary Computation (EC) algorithms rely on the concept of Darwin's evolution theory to find optimized solutions for a target problem, such as Genetic Algorithms (GA) and Genetic Programming (GP). The former considers a control policy whose configuration is evolved as a string of chromosomes - configuration parameters for a target problem. The latter evolves a complete control program for the task at hand. These methods use a fitness function that evaluates the candidate solutions, or individuals, and whose value is used as quality measure for a set of evolutionary operators (selection, crossover and mutation).

Candidate solutions in GP, or individuals, can fully describe the solution to the target problem, not requiring any a-priori structure. Therefore, although the complexity of the search space is increased, it is expected to generate more adequate solutions to a particular problem.

GP has proven to be useful in the generation of locomotion for very different types of robotic platforms, thus showing its efficiency in finding solutions for problems with a high level of complexity. In (Gritz and Hahn, 1997) a generic controller for an animated physically plausible 3D character was created: an articulated lamp. In (Tanev et al., 2005) a locomotion controller for an articulated, snake like robot was created. GP was also employed to generate a legged locomotion controller for a quadruped robot in (Andersson et al., 2000).

This method has also been applied in the generation of biped locomotion. In (Ok et al., 2001), GP was applied in the automatic generation of feedback neural networks for the control of a simulated 3D biped model with 32 muscles that controlled rigid segments of the legs, body and arms. The model was able to generate locomotion during only four steps. In (Ok and Kim, 2005), these results were improved by applying an enhanced adaptive mutation operator that reduced the search space and improved the evolution results, increasing the generated steps to 10. Although this work yielded interesting results, it is applied on a very specific model, which physical and mechanical properties do not fit common biped robotic platforms.

Other works address the generation of controllers to robotic platforms through the use of GP. In (Wolff and Wahde, 2007), Linear Genetic Programming (LGP) was used to generate a locomotion controller with feedback pathways, for a robust and anthropomorphic biped robot model. The model is simulated but physically plausible. There was no a-priori knowledge about the mechanical or physical properties of the body. Instead, the evolution uses feedback from several sensor modalities (e.g. joints and several accelerometers in the body and in the legs) to successfully achieve biped locomotion.

The work proposed in (Wolff and Nordin, 2003) presents the generation of robot legged locomotion in flat ground using LGP. A primary solution generated in simulation would be passed on to a physical robot. However, the achieved solution could not be executed in the physical platform.

We intend to use GP to automatically search the solution landscape and find solutions that rely on a set of motion primitives. We also explore the use of feedback pathways as a means to enable adaptation to the environment features, particularly to adapt the locomotion to walk up and down slopes in the environment. We are particularly interested in the impact of sensory inclusion in the robot behavior herein assessed considering Center of Mass (CoM) trajectory. This provides for an understanding of how feedback enhanced the locomotion skills of a biped robot. Results demonstrate the smooth locomotion achieved by the proposed GP mechanism and the added adaptability to the environment, provided by the inclusion of feedback pathways directly onto the controller. Therefore, movement is generated in entrainment with the environment.

The paper is organized as follows. The following section presents the locomotion model used to control the target platform. Then in section 3 the GP evolution mechanism is presented, where the individuals for the current evolution process and the evolution configuration are defined. Lastly, the results are presented in section 4, followed by a discussion in section 5 and conclusions and future work in section 6.

2 BIPED LOCOMOTION MODEL

The basis of the locomotion controller used in this work was previously presented in (Matos and Santos, 2012), where we proposed a Central Pattern Generator (CPG) integrated with local sensory feedback,

capable of generating bipedal locomotor behaviors, such as walking forward/backwards and turning.

The proposed CPG controls a single leg, divided in rhythmic and unit motion pattern generators. The use of a phase oscillator as a rhythmic generator allowed for a simple contralateral coupling between the left and right CPGs, maintaining the correct coordination between the generated locomotor trajectories in both legs by producing each a driving rhythmic signal ϕ_i in strict alternation (i for left/right leg).

Motion pattern generators receive this rhythmic input and produce the corresponding joint trajectory, z, in a synergistic approach of modular motion primitives encoded as a set of non-linear dynamical equations with well defined attractor dynamics, similarly to other works (McSharry et al., 2003; Nakanishi et al., 2004; Ijspeert et al., 2002). Basic parameterized motion primitives, e.g. sine and bell-shaped motions (implemented as Gaussians), were considered.

The joint angle value $z_{i,j}$, for leg i and joint j, is given by

$$\dot{z}_{i,j} = -\alpha\left(z_{i,j} - O_{i,j}\right) + \sum f_j^m\left(z_{i,j}, \phi_i, \dot{\phi}_i\right). \quad (1)$$

The final motor program in a single joint results from the sum of rhythmic motion primitives f_j^m around a center point $O_{i,j}$. α is a relaxation parameter for the offset. j specifies the joint: hip roll (hRoll), hip yaw (hYaw), hip pitch (hPitch), knee (kPitch), ankle roll (aRoll) and ankle pitch (aPitch); and i specifies the left or right leg.

This approach, despite simplistic, allowed a small humanoid DARwIn-OP robot to perform stable locomotion (Matos and Santos, 2012). The simplicity of the approach is also its weakness. For instance, if more complex motor programs are desired we are unaware of which motion primitives f_j^m should be employed. This problem motivates us to use automatic optimization in the present work, where the motor program is given as a result from GP evolution

$$\dot{z}_{i,j} = E_{i,j}\left(z_{i,j}, \phi_i, \dot{\phi}_i\right). \quad (2)$$

The biped robot DARwIn-OP has 6 joints per leg, hip roll, pitch and yaw, knee pitch and ankle pitch and roll. Since the motor programs are valid for both legs, and because we use the same motor programs of the hip and knee joints in the ankle joints to maintain the feet parallel to the ground at all times, we only perform the search in four motor programs for the hip roll, pitch and yaw, and for the knee pitch. This is depicted in Fig. 1.

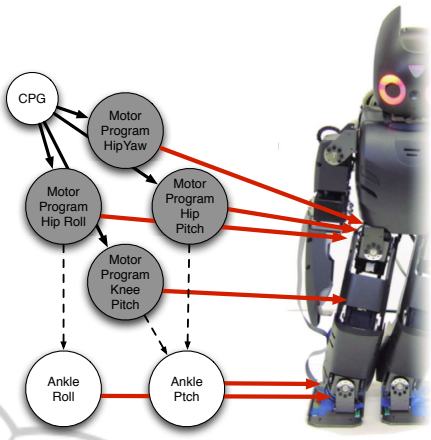

Figure 1: Schema for the control DarwinOP using only 4 motor programs (grey dark circles).

3 EVOLUTION CONFIGURATION

The evolution process aims to optimize the locomotion of a given platform, in this case the DarwinOP biped robot. Different platforms could have been used instead, under adequate configuration. Additionally, it is also desirable to provide adaptation to the locomotion through feedback pathways. Therefore, two specific goals are defined: 1) to improve the locomotion efficiency for the target platform comparatively to an initial hand-tuned solution by exploring different motion primitives for each joint; and 2) to explore the search and optimization of feedback pathways to achieve adaptability to changing features of the environment.

Two different controllers will be proposed in order to address each of these goals: *controller 1* and *controller 2*. *Controller 1* intends to generate biped locomotion to a target platform, and evolves in open-loop without including any sensory information from the environment. *Controller 2* intends to generate biped locomotion to the same target platform but in a closed-loop fashion. Therefore, sensory information was directly included into the movement generation in order to achieve adaptability to the environment. This last controller should therefore provide for better results when adaptability is required.

3.1 Individuals

Locomotion is generated according to four mathematical expressions E, eq. 2. Each individual, a candidate solution for the locomotion problem, is therefore composed by 4 chromosomes that correspond to the

four E equations, that will drive the robot joints, as depicted in Fig. 1.

Each chromosome is a mathematical expression under the form of a tree. The nodes of the tree are defined by functions whose branches correspond to the inputs, and that can be either other nodes or just leaves. The leaves are variables, constants and sensory inputs, that will input the lower level tree nodes.

The function set is specified as:

$$S_F = \{\sin, \cos, *, +, -, /, \exp, \text{sigmoid}, \text{step}\}. \quad (3)$$

It is assumed that this set is sufficient to create rhythmic motions for the achievement of the biped locomotion pattern. These functions were used in (Matos and Santos, 2012) to implement the basic parameterized motions primitives, sine and bell-shaped motions, to achieve biped locomotion.

Further, the functions sigmoid and step were included as relevant functions for the control process. The sigmoid function was defined as follows:

The step function according to :

$$\text{sigmoid}(x) = 2\frac{1}{1+\exp^{-x5}} - 1 \quad (4)$$

Both functions were though as interesting functions to enable the interaction between functions and between function and inputs (e.g. variables or sensory inputs).

$$\text{step}(x) = \begin{cases} 1 & \text{if} x > 0, \\ -1 & \text{if} x < 0. \\ 0 & \text{otherwise}, \end{cases} \quad (5)$$

The terminal set is defined specifically for each of the two proposed controllers.

3.1.1 Controller 1

For this controller the terminal set is defined as follows:

$$S_T = \left\{\phi, \dot{\phi}, z, -\frac{\pi}{6}, -\frac{\pi}{4}, -\frac{\pi}{3}, -\frac{\pi}{2}, -\pi, \frac{\pi}{6}, \frac{\pi}{4}, \frac{\pi}{3}, \frac{\pi}{2}, \pi, [-60, 60], 0\right\}, \quad (6)$$

where ϕ, $\dot{\phi}$ and z are the controller inputs as defined in section 2. The constants were chosen as angles and real values thought to be relevant for the purpose. Although the angles, such as $\frac{\pi}{2}$, are within the defined interval of real numbers, $[-60, 60]$, their specification in the terminal set, increases the probability of being chosen. This is important for the locomotion generation due to their adequacy to define phase relations between the different joints. The real interval, $[-60, 60]$, was chosen as possible values for the amplitudes of the different movements.

3.1.2 Controller 2

In order to be able to seek for adaptation to the environment such that locomotion is generated accordingly, sensory feedback can be directly included in the search space.

In this work, sensory feedback is provided by a three axis accelerometer, a_x, a_y and a_z; a three axis gyroscope, g_x, g_y and g_z; and the touch sensors of each leg, t_l and t_r. Both the accelerometer and the gyroscope values are normalized and then fed to the controllers. The touch sensors indicate if a given leg is in contact with the ground or not, yielding a boolean value indicating the state.

The terminal set for evolutions is thus defined as follows:

$$S_{T_f} = S_T \cup \{a_x, a_y, a_z, g_x, g_y, g_z, t_l, t_r\}. \quad (7)$$

3.2 Evaluation

The criterion used in the evaluation of the individuals is the forward distance traveled by the robot during a certain amount of time. Besides this, the fitness takes into account different and penalizable results as follows:

$$f = \frac{1}{N}\sum_{i=1}^{N}\left[\Delta_z^{(i)} - |\Delta_x^{(i)}| - c_{\text{fall}}^{(i)} v_{\text{fall}}^{(i)}\right] - c_{\text{nan}} v_{\text{nan}}, \quad (8)$$

where Δ_z is the forward displacement and Δ_x is the lateral displacement. The lateral displacement is removed from the forward displacement in order to compensate for possible asymmetric sliding of the platform. v_{fall} and v_{nan} are flags that indicate if the controller caused the robot to fall, or produced impossible joint positions, respectively. The constants c_{fall} and c_{nan}, are the penalizing coefficients for the corresponding situations. The undesirable situations of the robot falling or attempting an impossible joint position, as well as the lateral displacement, Δ_x, penalize the corresponding individual's fitness.

The evaluation process of each individual is divided into N stages. i indicates the stage. Each stage yields different displacements and possible falls that have to be averaged to count for the final fitness.

3.3 Architecture

The evolution process was implemented using the OpenBeagle Framework, (Gagné and Parizeau, 2006) as shown in Fig. 2. The initial population is generated with individuals that are evaluated using the Webots simulator, to guarantee that are feasible solutions, that is, they do not generate impossible joint positions.

The scenario may vary and yields the fitness values accordingly. The fitness of each individual is used by the genetic operators, selection, crossover and mutation, in the process of generating a new population. The new population is then evaluated and the loop goes on until the termination criterion is satisfied. In the present work, the termination criterion is defined by a maximum number of generations.

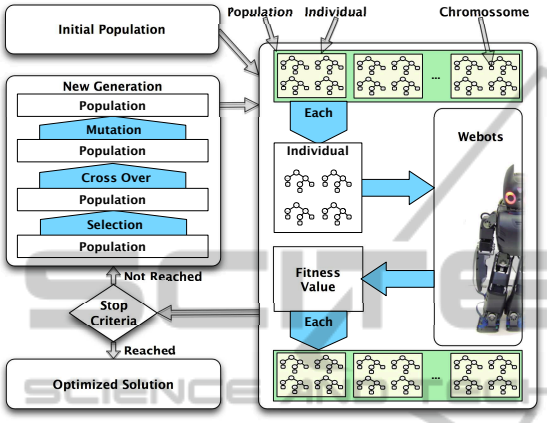

Figure 2: Evolution architecture. On the left, a new population is generated. On the right, the evaluation process in which each individual is tested in Webots simulator.

The initial population was specified using 10 seed individuals, as variations of a hand-tuned solution from our previous work in (Matos and Santos, 2012). These seed solutions were defined by small variations in the parameters of eq. 1, whether in the motion primitives, or in the offsets of the joints ($O_{i,j}$).

The remaining individuals were generated randomly using the ramped Half-and-Half method - where half the individuals are randomly generated trees until a variable depth is reached, and the other half until a variable size is reached. All share the same structure so that their information will be spread through the population during the evolution process, and coupled with different structures to generate novelty and diversity.

Other parameters of the evolution process configuration are listed in table 1. The crossover and mutation rates, as well as the selection sizes, were selected by trial and error, so that the evolution process yields optimized solutions.

The penalizing coefficient $c_{\text{nan}} = 10$, so that the genetic information that generates impossible joint positions is quickly discarded by the evolutionary process.

The number of stages, N, as well as the penalizing coefficient, c_{fall}, change for the different developed controllers. Therefore they will be specified in the text when required.

Table 1: GP evolution parameters.

Parameter	Value
Tournament Size	10
Cross over Rate	0.8
Mutation Rate	0.3
Population Size	500
# Generations	100
Max Depth	25

4 RESULTS

This section intends to show the obtained results for the two developed controllers.

Firstly, we demonstrate the adequacy of both controllers to produce different walking locomotion with improved performance over the initial hand-tuned one, according to the specified criteria. Secondly, we explore the inclusion of sensory feedback pathways as a means to provide for adaptability and as such achieve locomotion with better performance when climbing and descending slopes.

The evaluation of both controllers is performed in three different scenarios. *Flat Ground* experiment is the simplest scenario. *Slope Ground* experiment is a scenario in which the robot has to climb or to descend a sloped ground. *Up-Slope and Down-slope Ground* experiment is a scenario in which the same solution has to cope with up and down slopes.

During the first 10 s of an individual evaluation, the robot movements are linearly increased from an initial posture up to the specified values. This provides for a smooth and stable slow start of the robot's locomotion. At the end of each individual evaluation, the robot is set to its initial position and rotation, such that initial conditions are equal for the evaluation of all individuals of all populations.

Results were obtained in an Intel i7-2600k 3.4Ghz Linux (8 GB of RAM) PC.

4.1 Flat Ground Scenario

In this scenario, the goal of the evolved solutions is to generate a controller that enables the robot to go as further as it can, while drifting laterally as little as possible, during 30 s.

The robot always moves in the same flat ground. Thus only one stage is considered ($N = 1$). The fall penalty factor is set as $c_{\text{fall}} = 1$, so that the evolution process is forced to select individuals that do not cause the robot to fall, over the ones that do.

Fig. 3 depicts the forward and lateral displacements achieved for the 10 seed individuals (star

marker) and the best 4 solutions out of 4 independent evolutions for *controller 1* (square marker) and *controller 2* (triangle marker). Each evolution took approximately 30 hours to simulate.

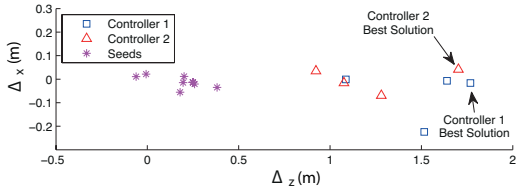

Figure 3: Displacements achieved by 10 seed individuals (magenta stars) and 4 solutions of *controller 1* (blue squares) and *controller 2* (red triangles), in flat ground. Initial position is (0,0).

The forward displacement, Δ_z, achieved by the evolved solutions of both controllers was far better than the one achieved by the seed individuals. The lateral displacement, Δ_x, varied within the solutions. The best solution was the one named of *Best controller 1* and highlighted in Fig. 3, with $\Delta_x \approx 2$ cm and $\Delta_z \approx 1.75$ m. Comparatively to the best hand-tunned one with $\Delta_x \approx 3.5$ cm and $\Delta_z \approx 38$ cm it improved 360% in z and 43% in x.

In the overall both controllers presented similar displacements. This suggests that the inclusion of feedback in the evolution process further increases the complexity, and that in case there is no explicit need for adaptation, that complexity could be discarded.

Fig. 4 shows the evolution of the fitness function values of the best solution for *controller 1* and *controller 2*. This figure allows to see the variation and the degree of learning of the best solution for controllers. It is possible to observe that the increasing of the fitness is slightly faster for *controller 2* when compared with *controller 1*. This seems to indicate that the inclusion of feedback can initially speedup the search. However, the fitness of the best solutions for both controllers tends to nearly identical values in the end of the search.

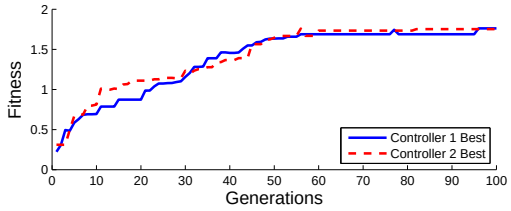

Figure 4: Fitness values for best solutions of *controller 1* (solid blue line) and *controller 2* (dashed read line) in flat ground.

Fig. 5 shows the Center of Mass (CoM) trajectory (red solid line) alongside with the feet position (blue polygons) during the evaluation procedure, for the solutions highlighted in Fig. 3 as *Best controller 1* (top) and *Best controller 2* (bottom). We can observe that

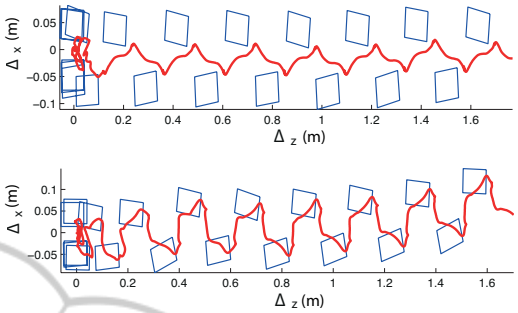

Figure 5: Trajectories of CoM (solid red line) and feet positions (blue polygons) in flat ground for *controller 1* (top) and *controller 2* (bottom).

lateral displacement is larger for the *controller 2* solution and makes the robot turn left. However, the inclusion of sensory information resulted in a larger excursion of the CoM trajectory. It is now projected within the bounds of the robot feet thus generating a more stable locomotion. Further, the robot weight moves towards the center of the foot which facilitates vertical clearance of the unloaded leg during the swing phase of the step. The result is a locomotion more entrained with the robot dynamics and the environment conditions.

4.2 Adaptation to Slopes

This experiment is intended to verify the adequacy of the proposed controllers to adjust the generated trajectories to the current environment, sensed by the described sensors. This adaptation to environmental features was evaluated considering slopes, that the robot is expected to climb and descend.

In order to obtain a path with smooth up and down slopes, these were generated using a sinusoidal function. The slope for each evaluation task are shown in Fig. 6, a and b, for the climb and descending tasks, respectively. Each tasks is composed of a single stage, $N = 1$. During the evolution, the robot starts 10 cm away from the slope, both when it will climb or descend. Then, it advances and needs to adapt to the changing inclination level, whose maximum value is 9.8 degrees, midway through the slope. This corresponds to a maximum height of 5.5 cm and an extension slope of 50 cm.

From the starting point to the end of the slope (in both cases of climbing and descending), the robot needs to walk over 60 cm. As in preliminary evolution tests the achieved velocities in such conditions

were lower than the ones on flat ground, the evaluation time was increased to 40 s, instead of the previous 30 s. The initial solutions to this task were bound to fall over, since the seed had no *a priori* knowledge of how to adapt the locomotion to the slope. Therefore, the fall penalty factor was set as $c_{\text{fall}} = 0.2$. This way, the controllers that caused the robot to fall were still penalized, but those that were able to reach the slope and only fell while in the slope, could persist in the populations and generate solutions that adapt the locomotion to the slope.

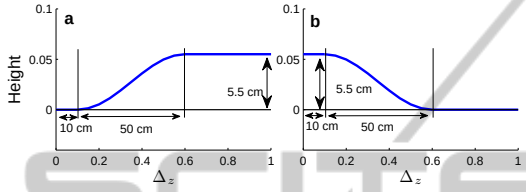

Figure 6: Up-Slope (a) and Down-slope (b) Ground stages.

In these scenarios, both controllers were evolved two times. Each of which took approximately 35 hours to simulate.

The results are presented in Fig. 7, top and bottom, for the up and down slopes, respectively. The achieved Δz displacements are similar for *controller 1* (square markers) and *controller 2* (triangle markers) in the up slope. However, in the down slope, *controller 1* seemed to achieve a slightly bigger Δz displacement. Also, the climbing seems to be harder than descending since Δz is much smaller.

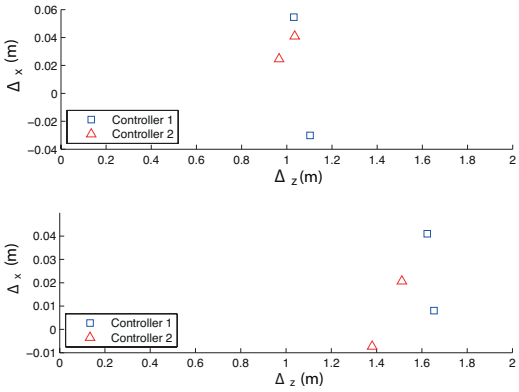

Figure 7: Forward and lateral displacements when the robot climbs (top) and descends (bottom) the slopes, for *controller 1* (square markers) and *controller 2* (triangle markers).

All solutions were tested in the converse scenario, i.e., if they were evolved to climb the slope, they were tested to descend it, and vice-versa. In all those cases, the robot fell. This suggests that the achieved adaptation to the slope did enable the robot to fully adapt only to the specific environment conditions. This denotes a static adaptation for a specific task. However, a solution that is able to climb (descend) a specific slope is also able to climb (descend) smaller slopes. This denotes a certain adaptability to the environment conditions.

4.3 Up-Slope and Down-slope Ground

In this scenario, the robot has both to climb and descend smooth slopes. Thus, two stages, $N = 2$, are considered. Firstly the robot attempts to perform the climbing stage, secondly, it is placed and reseted in order to attempt the descending stage. By forcing the robot to be able to face both these stages, the controller needs to adapt the locomotion to the slope, rather then statically adapt the robot's posture during locomotion. The goal is to verify if sensory inclusion provides for the required ability to adapt the generated locomotion to the environment, as well as to compare the results of both approaches. Further, we are interested in verifying the impact of sensory inclusion on the robot behavior.

In this scenario, the evolution stop criteria for each stage is 40 s and/or a maximum forward displacement of $\Delta_{z_{\text{max}}} = 0.8$ m. Thus, at most both stages take 80 s. This way, the over specialization on one of the stages is prevented. The fall penalty factor is set to $c_{\text{fall}} = 0.2$.

Three evolutions were performed for both the proposed controllers. Each took approximately 65 hours to simulate.

The obtained results are listed in table 2. We can observe that the solutions found without the inclusion of sensory inputs (*Controller 1*) were not able to fulfill the desired task. The robot fell during the climbing stage. On the other hand, *Controller 2* solutions were all able to both climb and descend the slopes. These results show that the inclusion of feedback enabled the robot to perform the task in both stages, thus adapting the locomotion as required.

Figs. 8 and 9 show the CoM trajectory (red solid line) alongside with the feet position (blue polygons) during the climbing (top) and the descending (bottom) stages for *Controller 1* and *Controller 2*, respectively, for the highlighted solutions in table 2. These are the solutions with higher fitness for both controllers. This simple functional gait analysis shows that besides finding solutions that do not fall, *Controller 2* was able to find solutions in which the CoM movement smoothly oscillates between the center of both feet with less abrupt oscillations. Again, this shows up that the proposed framework managed to enhance

Table 2: Relevant information for Up-Slope and Down-slope Ground stages for both controllers. Distances are expressed in meters.

	Up	Down	f	Disabled feedback	Full slope
Controller 1	Δ_z: **fall** Δ_x: **fall**	Δ_z: **0.8** Δ_x: **0.03**	**0.29**	-	**no**
	Δ_z: fall Δ_x: fall	Δ_z: 0.8 Δ_x: 0.11	0.27	-	no
	Δ_z: fall Δ_x: fall	Δ_z: 0.61 Δ_x: 0.24	0.1	-	no
Controller 2	Δ_z: **0.75** Δ_x: **-0.08**	Δ_z: **0.77** Δ_x: **0.1**	**0.67**	**no**	**yes**
	Δz: 0.6 Δx: 0.05	Δz: 0.65 Δx: 0.07	0.56	no	yes
	Δ_z: 0.61 Δ_x: 0.09	Δ_z: 0.64 Δ_x: 0.07	0.54	no	no

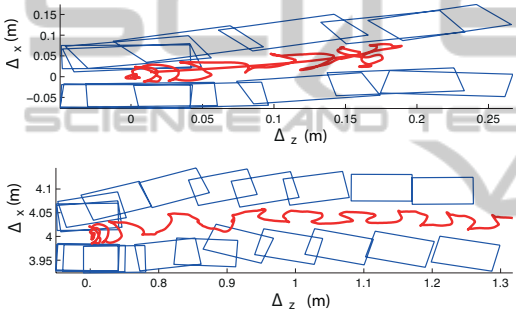

Figure 8: Trajectories of CoM (solid red line) and feet positions (blue polygons) during the climbing (top) and the descending (bottom) stages of *Controller 1* with highest fitness.

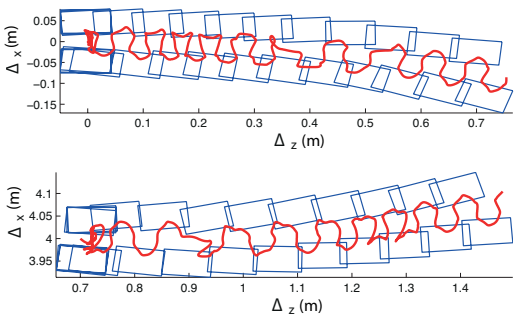

Figure 9: Similar to Fig. 8 but for *Controller 2*.

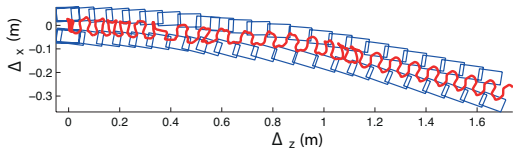

Figure 10: Trajectory of *Controller 2* solution with higher fitness in the full slope scenario.

the locomotion skills of the biped robot.

A relevant question emerges. What if the sensory feedback pathways were set to zero in the determined *Controller 2* solutions when performing the up-down slope? This would show the need for feedback inclusion in the controller, when driving the robot through the defined task.

The obtained results are presented in table 2 in the column labeled *Disabled feedback*. An *yes* indicates if the solution was still able to walk over the scenario. In all cases feedback was mandatory for the *Controller 2* solutions to perform both tasks.

The inclination level used in this work, was out of the reach of *Controller 1* solutions, but it is achievable through the use of feedback pathways. All found solutions were also able to perform locomotion in slopes with lower inclination values.

In order to verify the solutions' generality a different scenario was used. During 80 s, the robot is faced with a complete sine curve, a down slope immediately followed by an up slope, in a continuous fashion. This scenario is intended to verify if the generated solutions are able to cope with the overall path. It requires the controller to continuously adapt the locomotion as it progresses through the slope.

The achieved results are listed in the last column of table 2, labeled *Full Slope*, for each of the *Controller 2* solutions. An *yes* indicates if the solution was able to walk over the scenario. Note that only one solution was not able to walk over the overall path.

Fig. 10 shows the achieved trajectories for the *Controller 2* solution with highest fitness (highlighted in table 2). Despite a slight final lateral displacement, it successfully achieved a forward displacement $\Delta_z \approx 1.7$ m.

5 DISCUSSION

In case of flat ground, the use of feedback pathways showed no specific advantage, since the solutions were very similar. These results were the expected ones, since there was no explicit need for feedback, and the inclusion of sensory feedback in the search space further increased the complexity of the optimization problem. However, considering other metrics for the performance of the generated movement, such as the movement of the COM, one can see that robot motion seems to be more entrained with the environment and the robot model when in closed-loop. This needs more attention in future work.

Evolution results in the slope scenarios demonstrated the ability to generate solutions capable to climb and descend slopes, both with and without the inclusion of feedback inputs. However, the achieved solutions could only perform in the tasks for which they were evolved to. Otherwise, the robot fell. For instance, the solutions that were evolved to climb the slope would fall if tried to descend that same slope. These results suggest that the robot's posture was adapted to the slopes during evolution, rather than its locomotion generation, as it goes through the slope.

In order to prevent this from happening, a different scenario composed of up and down stages was evaluated, in which each solution evolved in both stages. The results showed that only the solutions with feedback inclusion achieved adaptation to the environment in both stages. When no feedback was considered the robot fell during the climbing stage. Therefore, the use of feedback was required in order to enable adaptation to the ground's slope.

More importantly was to verify the impact that sensory information inclusion brings to the robot performance and how it enhances the locomotion skills of the biped robot. This was assessed through a simple functional gait analysis considering CoM trajectories. The resultant CoM trajectory was enlarged and smoothly oscillated between the center of both feet. Thus, the generated locomotion was more stable and thus entrained with the robot dynamics and involving environment.

6 CONCLUSIONS

In this paper, we have proposed a gait optimization system for a biped robot using GP. Further, we explore the inclusion of feedback pathways to achieve locomotion adaptation to the environment. We based this work in (Matos and Santos, 2012), in which a CPG based solution using a combination of a small set of motion primitives was able to generate biped walking for the DARwIn-OP humanoid robot.

Two controllers were developed. *Controller 1* generates biped locomotion for a target platform, and evolves in open-loop without including any sensory information from the environment. *Controller 2* generates biped locomotion for the same target platform but in a closed-loop fashion. Therefore, sensory information was directly included into the movement generation by the GP evolution process, in order to achieve adaptability to the environment.

The obtained results have shown the adequacy of both controllers to produce different walking locomotion with improved performance over the initial hand-tuned one, according to the specified criteria. In fact, the achieved displacement was up to four times larger. Further, the inclusion of sensory feedback pathways provided the required adaptability to achieve locomotion with better performance when climbing and descending slopes. These solutions were tested against the disabling of the feedback (replacing by a null value), and against scenarios not used in the evolution stages, demonstrating the generalization of the solution. The solutions were able to continuously adapt to the changing inclination of a complete sine curve like slope, and also were able to walk over slopes of lower inclination levels.

In the overall, the obtained results emphasize the fact that the inclusion of feedback has enabled a smoother locomotion, with less undesired oscillations. The obtained locomotion was more entrained with the robot dynamics and the involving environment. Future work includes to extend this functional gait analysis to quantify how the proposed framework managed to enhance the locomotion skills of the biped robot.

Additionally, experiments were performed on a physical platform. However, a great difference in the physical setup caused the robot to fall. Such problem is often referred to as Reality Gap. As future work, this problem will be addressed similarly to the proposed approach in (Koos et al., 2010), so as to provide an efficient locomotion controller for a real DarwinOP robot.

ACKNOWLEDGEMENTS

This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the Project Scope PEst OE/EEI/UI0319/2014.

REFERENCES

Andersson, B., Svensson, P., Nordahl, M., and Nordin, P. (2000). On-line evolution of control for a four-legged robot using genetic programming. *Real-World Applications of Evolutionary Computing*, pages 322–329.

Gagné, C. and Parizeau, M. (2006). Open beagle: a c++ framework for your favorite evolutionary algorithm. *SIGEVOlution*, 1:12–15.

Gritz, L. and Hahn, J. (1997). Genetic programming evolution of controllers for 3-d character animation. *Genetic Programming*, 97.

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

Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2002). Learning Rhythmic Movements by Demonstration Using Nonlinear Oscillators. In *In Proceedings of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS2002)*, volume 2002, pages 958–963.

Kim, J.-J., Lee, J.-W., and Lee, J.-J. (2009). Central pattern generator parameter search for a biped walking robot using nonparametric estimation based particle swarm optimization. *International Journal of Control, Automation and Systems*, 7(3):447–457.

Kim, Y., Tagawa, Y., Obinata, G., and Hase, K. (2011). Robust control of cpg-based 3d neuromusculoskeletal walking model. *Biological cybernetics*, pages 1–14.

Koos, S., Mouret, J.-B., and Doncieux, S. (2010). Crossing the reality gap in evolutionary robotics by promoting transferable controllers. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, GECCO '10, pages 119–126, New York, NY, USA. ACM.

Matos, V. and Santos, C. P. (2012). Central pattern generators with phase regulation for the control of humanoid locomotion. Business Innovation Center Osaka, Japan.

McSharry, P., Clifford, G., Tarassenko, L., and Smith, L. (2003). A dynamical model for generating synthetic electrocardiogram signals. *Biomedical Engineering, IEEE Transactions on*, 50(3):289 –294.

Nakanishi, J., Morimoto, J., Endo, G., Cheng, G., Schaal, S., and Kawato, M. (2004). Learning from demonstration and adaptation of biped locomotion. *Rob. and Aut. Systems*, 47(2-3):79 – 91.

Ok, S. and Kim, D. (2005). Evolution of the cpg with sensory feedback for bipedal locomotion. *Advances in Natural Computation*, pages 428–428.

Ok, S., Miyashita, K., and Hase, K. (2001). Evolving bipedal locomotion with genetic programming-a preliminary report. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 2, pages 1025–1032. IEEE.

Sugimoto, N. and Morimoto, J. (2011). Phase-dependent trajectory optimization for cpg-based biped walking using path integral reinforcement learning. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 255–260. IEEE.

Tanev, I., Ray, T., and Buller, A. (2005). Automated evolutionary design, robustness, and adaptation of sidewinding locomotion of a simulated snake-like robot. *Robotics, IEEE Transactions on*, 21(4):632–645.

Vukobratović, M. and Borovac, B. (2004). Zero-moment point—thirty five years of its life. *International Journal of Humanoid Robotics*, 1(01):157–173.

Wolff, K. and Nordin, P. (2003). Learning biped locomotion from first principles on a simulated humanoid robot using linear genetic programming. In *Genetic and Evolutionary Computation—GECCO 2003*, pages 199–199. Springer.

Wolff, K. and Wahde, M. (2007). Evolution of biped locomotion using linear genetic programming. *Climbing and Walking Robots, Towards New Applications*.