Explainable Robotics applied to bipedal walking gait development

Nico Roos and Zhenglong Sun

Data Science and Knowledge Engineering, Maastricht University, The Netherlands
roos@maastrichtuniversity.nl

Abstract. Explainability is becoming an important topic in artificial intelligence (AI). A well explainable system can increase the trust in the application of that system. The same holds for robotics where the walking gait controller can be some AI system. We will show that a simple and explainable controller that enables an energy efficient walking gait and can handle uneven terrains, can be developed by a well structured design method. The main part of the controller consist of three simple neural networks with 4, 6 and 8 neurons. So, although creating a stable and energy efficient walking gait is a complex problem, it can be generated without some deep neural network or some complex mathematical model.

1 Introduction

Stability and energy efficiency are two important aspects of a robot's waking gait. Without sufficient stability, the robot will regularly fall, which limits the applicability of the robot. Energy efficiency is important because the available energy stored in the robot's battery is limited and walking is a main source of energy consumption. A more energy efficient gait will increase the time the robot can operate without recharging.

How can we generate an energy efficient bipedal walking gait that is stable on uneven terrains, is an important question in robotics. If we would ask this question to todays students in artificial intelligence, then most most of them would propose some approach based on *deep reinforcement learning*; e.g.: [12, 13]. The use of deep neural networks is a hype and many seem to think that it is the solution to all problems. Traditional approaches are based on an abstract or detailed mathematical model of the robot [10, 9, 8, 1, 3, 2, 14] usually combined with the zero moment point (ZMP) stability criterium [18].

There are important differences between the two approaches. If sufficient training is possible, the deep neural networks based approach may reach a higher energy efficiency and and robustness when walking on uneven terrains. Collecting a large amount of training data using a real robot is however time consuming, and therefore high quality simulators are needed. A major disadvantage is that deep neural networks sometimes fail for unknown reasons.

The approaches based on mathematical models are well understood and provide clear conditions under which a stable walking gait can be realized. The most advanced mathematics-based approaches can realize a high degree of stability on many different types of terrains. The latter approaches require a detailed mathematical models of the robot. When the robot carries some object, model needs to be adapted. The mathematical models are also computational demanding. Energy efficiency is not consider since the focus is on stability.

Is there middle road between the two extremes outline above? An approach that guarantees a stable gait on different types of terrains, is energy efficient, and can easily be explained? Below we will see that the answer is 'yes'. We take inspirations from a *partially passive dynamic walkers* [4, 5]. These types of robots have body configuration that are designed to produce a stable gait with minimal control while they are unstable according to well known stability criteria such as the zero moment point. Their walking gait can be analyzed by a *Poincaré map* [6]. The idea behind the use of a Poincaré map is to look at the state of a robot at some fixed moment in successive steps. The heel strike is an often used moment in a step where we determine the state of the robot. So, at the heel strike of step i we determine the robot state $\mathbf{s}_i$. The Poincaré map $P(\mathbf{s}_i) = \mathbf{s}_{i+1}$ is a non-linear mapping between successive states. There exists a stable walking gait if the Poincaré map has a 'fixed point': $P(\mathbf{s}^*) = \mathbf{s}^*$ and is asymptotically stable in region around the fixed point $\mathbf{s}^*$. These types of stable walking gaits are denoted as *Limit Cycle Walking* [7]. Unfortunately, these robots have drawbacks such as a limited range of walking motions and inadequate robustness of control. They generally excel however in the energy efficiency compared to robots with full control over all degrees of freedom.

In the following sections we will see that it is possible to design a very simple but flexible gait controller that produces an energy efficient bipedal walking gait that is stable on uneven terrains for robots that have full control over all degrees of freedom. Like the partially passive dynamic walkers with specially designed body configurations, the gait is not stable according to well known stability criteria such as the ZMP but the corresponding Poincaré map is asymptotically stable. The approach consists of five steps.

1. Use a simple abstract model to identify how the length of the stance leg must change during a step to realize a stable and energy efficient gait.
2. Use the identified behavior of the stance leg to create a controller with control parameters implementing a gait on a bipedal robot[1]. Different parameters are used for each of the two legs.
3. Optimize the parameters of controller for different walking speeds on a flat ground using reinforcement learning by letting the robot walk in a simulator[2] or in the real world.

[1] We used the Nao robot produced by Softbank's Aldebaran Robotics company in our experiments.

[2] We used the robot simulator *Webots* initially developed at the Laboratoire de Micro-Informatique (LAMI) of the Swiss Federal Institute of Technology, Lausanne, and currently developed by Cyberbotics.

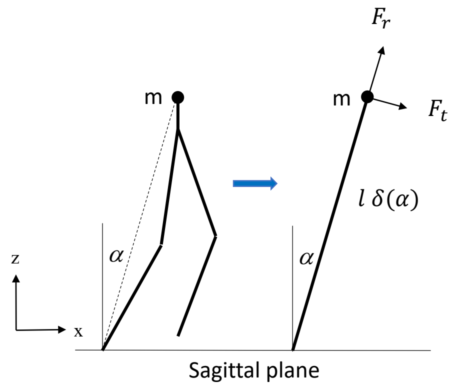

Fig. 1. The abstract Inverted Pendulum Model

4. View walking on an uneven terrain as walking on a slope. Consider slopes in the sagittal and the lateral direction (walking up or down hill, and walking perpendicular to direction slope). Finally optimize the parameters for the two directions and for different slope angles.
5. An analysis of the parameter values shows that we need to control the movement of the stance leg, and the force of the swing leg in the double support phase as function of the height difference of the two feet. Three simple neural networks (only one hidden layer) suffices to create a robust controller the enables an energy efficient gait on uneven terrains.

2 Inverted Pendulum Model

Srinivasan and Ruina [15] showed that the most energy efficient walking gait identified using an Inverted Pendulum Model with telescopic legs corresponds with the human walking gait. We investigate whether a similar model as described by Srinivasan and Ruina can be used for identifying an energy efficient gait for humanoid robots such as a Nao robot, despite differences between humans and humanoid robots. First, unlike muscles, motors of a robot do not behave like springs. Second, our experiments with a Nao robot show that the energy consumption of a motor mainly depends on the torques of joints and that the work that is done can be ignored. Srinivasan and Ruina only considered the work done. Third, a human need not bend the knee of the stance leg while walking because (s)he can toe-off using the foot and the calf muscle. In this way the human can increase the length of the leg without torque on the knee joint. A robot such as a Nao robot cannot toe-off, because it can not bend its foot. This difference makes it possible to ignore the torque in the human model, but not in models of certain robots such as a Nao robot.

Inverted pendulum model To reduce the total energy cost, we set the stiffness on both ankle joints to zero. Thus, the stance leg of the robot can freely rotate around the ankle joints, and the area of support reduces to a point. Apart from reducing the energy cost, zero ankle stiffness also allows the robot to adapt to an uneven floor.

To analyze the energy consumption, we use an Inverted Pendulum Model with telescopic legs [16, 17] (see Figure 1). This model, which is based on the work of [15], ignores the physical structure of both legs, and use an imaginary massless segment, the *virtual leg*, connecting the point mass to a ground contact point instead. Furthermore, this model allows the length of the virtual support leg to vary during a step. A leg-length policy $\delta : [-\frac{\pi}{2}, \frac{\pi}{2}] \rightarrow [0, 1]$ determines how much the virtual support leg will be shortened as a function of the angle α between the (virtual) stance leg and the vertical axis. The shortening of the stance leg is realized by bending the knee joint.

We use the leg-length policy of the stance leg to determine the radial force F_r^s on the mass m located and the CoM in the direction of the virtual leg, and tangential force F_t^s on m perpendicular to F_r^s. Note that F_t^s works perpendicular to the telescopic leg while the path of the mass m need not be perpendicular to the leg because the length of the leg may change. The superscript s in F_t^s refers to the **single** support phase. Using the force F_t^s, we can determine the second derivative of the position of x_t w.r.t. the time t which is given by: $F_t^s = ma = m\frac{d^2x_t}{dt}$. F_t^s is determined by the component of the gravity working perpendicular to the stance leg: $mg \sin \alpha$ and the friction: $b\frac{dx_t}{dt}$. The air friction and the ankle joint friction (if an ankle is present) reduce the forward speed of a walking robot. The air friction is quadratic in the speed but can assumed to be linear because of the low walking speed. The ankle joint friction depends on the connected gearbox and motor. This friction is also assumed to be linear in the speed. The constant b captures both types of friction. Hence, we get: $\frac{d^2x_t}{dt^2} + \frac{b}{m}\frac{dx_t}{dt} - g \sin \alpha = 0$. We can transform this equation into a differential equation of the angle α:

$$\frac{d^2\alpha}{dt^2} + \frac{1}{\delta(\alpha)}\frac{d\delta(\alpha)}{d\alpha}\left(\frac{d\alpha}{dt}\right)^2 + \frac{b}{m}\frac{d\alpha}{dt} - \frac{g}{\delta(\alpha)l}\sin \alpha = 0 \tag{1}$$

The above differential equation also enables us to determine the radial force F_r^s that needs to be generated by the stance leg.

$$F_r^s = mg \cos \alpha + ml\left(\frac{d^2\delta(\alpha)}{d\alpha^2}\left(\frac{d\alpha}{dt}\right)^2 + \frac{d\delta(\alpha)}{d\alpha}\frac{d^2\alpha}{dt^2}\right) \tag{2}$$

A stable gait (limited cycle walking) When the swing foot impacts with the ground at the end of a step, the direction in which the mass m is moving may change. We assume an *inelastic* collision of the swing foot with the ground. This implies that the speed of the mass m in the radial direction of the swing leg becomes equal to the radial speed $v_{b,r}$ of the stance leg at the beginning of a step. To ensure a constant walking speed, the tangential speed of the swing leg at the end of a step must be the same as the tangential speed $v_{b,t}$ of the stance leg at the beginning of a step. To compute the tangential speed of the swing leg at the end of a step, we first compute the tangential and radial speed of the mass m w.r.t. the stance leg: $v_t = l\delta(\alpha)\frac{d\alpha}{dt}$ and $v_r = \frac{dl\delta(\alpha)}{dt} = l\frac{d\delta(\alpha)}{d\alpha}\frac{d\alpha}{dt}$, and next the corresponding speeds in the Cartesian coordinate system: $v_x = v_t\cos(\alpha) + v_r\sin(\alpha)$ and $v_z = v_t\sin(\alpha) + v_r\cos(\alpha)$. The robot is walking at a constant speed if the following equation holds:

$$\boldsymbol{v_{t,b} = (v_{z,e}\sin(\alpha_b) + v_{x,e}\cos(\alpha_b))}$$

The subscript b and e denote the beginning and the end of a step respectively.

Heel strike The law of momentum conservation applies to the impact of the swing leg with the ground. Therefore, the sum of the moments before and after the foot impacts with the ground must be 0. This does not imply that the impulse generated by the reaction force of the leg when it impacts with the ground is 0. The inelastic collision implies that the impulse must stop the movement in the radial direction of the swing leg (the *new* stance leg). The impulse generated by the reaction force of the new stance leg is determined by the change in speed of m in the direction of the new stance leg: $I_r = v_{r,b}m - v_{r,e}m$. Here I_r denotes the impulse in the radial direction of the new stance leg, $v_{r,b}$ denotes the radial speed of the new stance leg at the beginning of a step, and $v_{r,e}$ denotes the radial speed of the swing leg at the end of a step. After the foot impacts with the ground, the speed in the radial direction of the *new* stance leg is completely determined by the leg-length policy because of the inelastic collision; i.e., no bounce occurs: $v_{r,b} = \frac{dl\delta(\alpha)}{dt}(t_b)$. So, the impulse produced by the leg becomes:

$$I_r = m\left(\frac{dl\delta(\alpha)}{dt}(t_b) - \Big(v_{x,e}\sin(\alpha_b) + v_{z,e}\cos(\alpha_b)\Big)\right)$$

The impulse also equals: $I_r = \int F_i\ dt$. In an ideal situation, the impact time t_{imp} with the ground is infinitely small implying an infinitely large reaction force produced by the leg on the mass m. We assume that the reaction force is constant during the impact. So, $F_i = \frac{I_r}{t_{imp}}$

Energy consumption and optimal leg-length policy To identify the leg-length policy that minimizes the energy consumption of a robot, we make use of the fact that the robot has to bend the knee in order to shorten the leg. The energy consumption is observed to be proportional to the torque of these joints in experiments with a Nao robot. So, a stretched leg requires a minimal amount of energy while a largely bent leg requires a maximum amount of energy. The experiments also showed that the contribution of the positive work can be ignored. We will use these observations to determine the energy consumption in the model. For each leg-length there is a corresponding bending of the knee joint, and a corresponding torque. The torque on the knee joint is determined by the radial force and arm of this force w.r.t. the knee joint. The arm is given by: $r = \frac{1}{2}l\sqrt{1-\delta(\alpha)^2}$. Ignoring the positive work, we define the energy consumption as:

$$E_{ic} = \lim_{t_{imp}\to 0} E \propto \frac{1}{2}l\sqrt{1-\delta(\alpha_b)^2}I_r + \int_0^{t_s}\frac{1}{2}l\sqrt{1-\delta(\alpha)^2}F_r^s dt$$

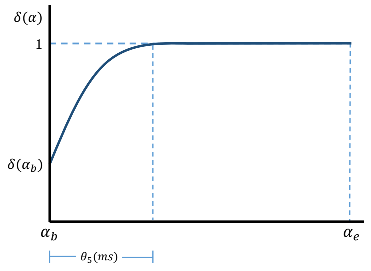

Fig. 2. The optimal leg-length policy

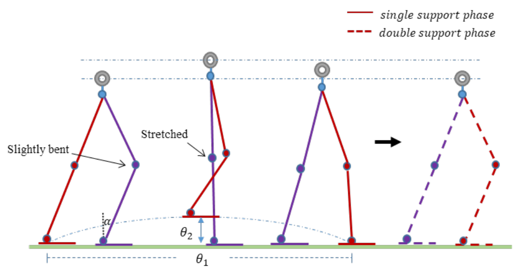

Fig. 3. The kinematics of the optimal sagittal motion

where t_s is the duration of a step.

The above described model has been used to identify an energy efficient gait by evaluating different leg-length policies. Figure 2 shows the leg-length policy $\delta(\alpha)$ as a function of the angle α from the beginning till the end of the step and Figure 3 right shows the realization using the 5-link model. We can see that the swing leg may touch the ground before it takes over the support of the robot. Since the swing leg only takes over the support of the robot at the end of a step, the knee joint of the swing leg is under-actuated during the double support phase in this planar model.

The effect of the double support phase on the leg-length policy In the double support phase (DSP) the robot has to shift its weight from the stance leg to the swing leg. In order to be balanced in the lateral direction at the end of the double support phase, the robot must put force on the swing leg to stop the lateral movement in time. This force also influences the movement in the sagittal plane.

The results of the new simulations that considered the force generated by the swing leg during the DSP, showed that the shape of the leg-length policy does not change, the stance (rear) leg stays fully stretched ($\delta(\alpha) = 1$) in the DSP.

Stability analysis The stability of a gait is the most important aspect. Since the ankle joins are under-actuated, stability criteria, such as the ZMP criterion, cannot be used to analyze the stability of the gait. In the identified energy efficient gait, the robot *falls* forward till the swing leg takes over the support. Therefore, to analyze the stability of the gait we will apply the Poincaré map.

The simulation described in this section determines a Poincaré map. The state variable in the Poincaré map is the speed v of the CoM at the start of the single support phase. The Poincaré map itself depends on the leg-length policy $\delta : [-\frac{\pi}{2}, \frac{\pi}{2}] \to [0, 1]$. The optimal leg-length policy δ^* determines a fixed point v^* of the Poincaré map: $P(v^*; \delta^*) = v^*$. In other words, the speed of the CoM at the start of the single support phase (the leftmost abstract representation in Figure 3) is the same as the speed at the start of the single support phase of the next step (the rightmost abstract representation in Figure 3).

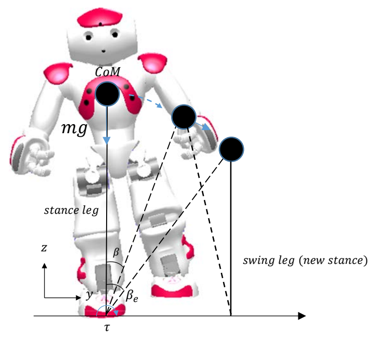

Fig. 4. The frontal plane during double support phase

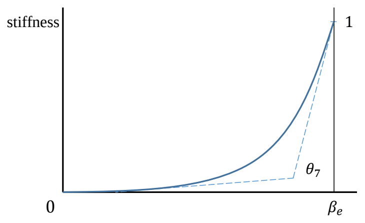

Fig. 5. A Quadratic Bezier Curve specifying the stiffness as a function of the angle β

To assess the stability of the fixed point solution, we varied the initial speed at the start of the single support phase while using the leg-length policy δ^* that corresponds with the fixed point solution. For small variations ε in the initial speed, the difference with the fixed point speed decreases after one step; $|P(v^* + \varepsilon; \delta^*) - v^*| < |\varepsilon|$. Hence, the walking gait is asymptotically stable.

3 Kinematics Model in Lateral Direction

For a simple forward step, it is insufficient to only consider 2D dynamics in the sagittal direction. We also need to consider the dynamics in the frontal plane (the lateral direction). We use an inverted pendulum model shown in Figure 4.

We assume that during the single support phase, the robot is perfectly balanced in the lateral direction. Therefore, at the beginning of the DSP, the CoM is vertically above the center of the stance foot, and in the frontal plane there is no torque to make the CoM rotate around the sole of the stance leg. So, the angle β between the virtual telescopic stance leg and the vertical axis in the frontal plane is equal to 0, as illustrated in Figure 4.

In order to balance the CoM in the lateral direction above the swing foot (the next stance foot) at the end of the DSP, we need a torque τ rotating the CoM in the frontal plane while the stance leg is fully stretched. So the angle β changes from 0 to β_e. To generate the torque τ, we manipulate the upper body to bend slightly inwards at the angle θ_8 for 100 ms. The bending θ_8 disrupts the balance enabling gravity to create a torque $\tau > 0$. We manipulate the force generated by the swing leg to control the rotation of the CoM with a non-zero angular velocity $\dot{\beta}$ and to stop the rotation at the position ($\beta = \beta_e$) where the robot can put its whole body weight on the new stance leg and keep it stable. The problem is to control the torque τ appropriately. We do this by controlling the force generated by the swing leg by means of a force policy $\gamma : [-\frac{1}{2}\pi, \frac{1}{2}\pi] \to [0, 1]$. We define the force policy with respect to the angle β of the telescopic leg with the vertical axis in the frontal plane.

We use the force policy $\gamma(\beta)$ to control the *stiffness* of the swing leg's knee joint, and thereby control the force generated by the swing leg on the CoM. At the beginning of the DSP, ideally, the force policy $\gamma(\beta)$ imposes *no* force on the swing leg. Therefore the torque τ generated by the slightly inward bending θ_8 of the upper body, is needed to start the lateral movement of the CoM. (In our experiments, the bending lasts for 100 ms.) As β increases, the force policy controls the force of the swing leg to gradually decrease the $\dot{\beta}(t)$, and stops the CoM movement when $\beta = \beta_e$. When $\beta = \beta_e$, the angular velocity $\dot{\beta}$ and acceleration $\ddot{\beta}$ become 0, therefore the CoM stabilizes above the swing leg (the new stance leg). The shape of the force policy is determined by means of a Quadratic Bezier curve, as illustrated in Figure 5. This Quadratic Bezier curve is defined by 3 points in the interval of the DSP. The start point and the end point are fixed, so we start with no force generated by the swing leg and stop with the full weight of the robot on the swing leg, after which it becomes the new stance leg. We assume a smooth transition between these two points which is determined by the middle point θ_7 of the Quadratic Bezier curve. So we have to determine the optimal point θ_7. The optimization of the point θ_7 will be discussed in the Section 4.

4 The Gait Controller

Gait parameters We designed a controller which implements the gait identified in the previous sections on an Aldebaran Nao robot. The controller for walking on flat ground has 9 basic parameters that are essential in controlling the gait. To enable walking on a slope, several parameters are split into a parameter for the left leg and a parameter for the right leg.

- *Step Length* (θ_1): Defines the distance over which the Nao moves in a singe step (sagittal).
- *Step Height* (θ_2): Defines the maximal altitude between the ground and a lifted foot. A high step height requires a faster movement of the swing leg, which may cause instabilities. A low step height increases the possibility of tripping and limits the step length.
- *Knee Bending* (θ_3^L, θ_3^R): Defines the maximum bending of the swing leg (left and right) at the beginning of the double support phase which determines the value of $\delta(\alpha_b)$, see Figure 2. This parameter determines the sagittal velocity and the energy cost.
- *Step Time* (θ_4): Defines how long a single step lasts. This parameter determines the sagittal walking velocity.
- *Stretch Time* (θ_5^L, θ_5^R): Defines how long it takes for the stance leg to stretch from θ_3^L and θ_3^R (angle of bent knee left and right) to its full length at the beginning of the single support phase, see Figure 2.
- *Torso Pitch Inclination* (θ_6^L, θ_6^R): Defines the maximum angle that the torso leans in the sagittal direction at the beginning of the first step. If positive, it will move the center of mass (CoM) in the sagittal direction. If it is not

set appropriately, a fall will occur. In our experiments, the inclination lasts for 200 ms.
- *Quadratic Bezier point* (θ_7^L, θ_7^R): Defines the magnitude of the middle points in Quadratic Bezier Curves, see Figure 5, which determines the force policy of the swing leg (left and right) introduced in Section 2.
- *Torso Roll Inclination* (θ_8^L, θ_8^R): Defines the maximum angle that the torso leans in the lateral direction. If positive, it will move the center of mass (CoM) towards the swing leg in the frontal plane as discussed in Section 2.
- *Proportion of single support duration* (θ_9^L, θ_9^R): Defines how long the single support phase lasts in one single step. The single support phase duration equals the product of this parameter and the step time θ_4.

All parameters except θ_1 (the step length) will be optimized in the experiments. We do not consider the optimization of the step length, because we determine the parameters for a fixed walking velocity. We manually set a different walking velocity v in each experiment and determined the optimal *Step Time* θ_4. The corresponding step length is given by: $\theta_1 = v\theta_4$.

Fitness function To optimize the parameters, we need a fitness function $\mathcal{F}$. The fitness function should address two objective, (1) the stability of the gait, and (2) energy consumption of the gait. We measure the stability of the gait by letting the robot walk a maximum distance D and measuring the actual distance d that the robot walks without falling. The energy consumption is evaluated by comparing the actual power consumption with the maximum power consumption of the robot. The stability has a preference over the energy consumption with a ratio of 7 to 3. We use $\mathcal{F}(\boldsymbol{\theta})$ to denote the fitness of the gait determined by the vector $\boldsymbol{\theta}$ of control parameter values.

Ankle stiffness Initial experiments with the Nao robot showed that it is not possible to set the ankle pitch stiffness to 0. The Nao robot starts to walk on its heels and has not enough grip to propel itself. An ankle pitch stiffness of 0.1 solved the problem. This low stiffness still allows free rotation around the ankle joint.

Learning the Controller Parameters We use a Policy Gradient Reinforcement Learning (PGRL) method presented by Kohl et al. [11] to optimize the gait parameters. We chose this version of PGRL because we cannot calculate the gradient exactly.

Optimal parameter values To identify the optimal gait parameters and validate the gait's performance, we uploaded the controller of our proposed gait together with an implementation of the policy gradient algorithm into the *Webots* simulator[3]. We learned an optimal set of parameter values starting from a randomly

[3] The Webots software was initially developed at the Laboratoire de Micro-Informatique (LAMI) of the Swiss Federal Institute of Technology, Lausanne, Switzerland (EPFL). The Webots robotic simulation software is currently developed by Cyberbotics.

Table 1. The optimized parameters of for walking on a flat ground at different velocities

Velocity	$\theta_1(cm)$	$\theta_2(cm)$	θ_3	$\theta_4(ms)$	$\theta_5(ms)$	θ_6	θ_7	θ_8	θ_9
2 cm/s	2.3	3.24	6.2°	1150	53	4.3°	(0.9,0.2)	6.8°	0.7
3 cm/s	2.9	3.24	8.7°	980	59	5.6°	(0.9,0.2)	6.8°	0.7
4 cm/s	3.2	3.24	11.5°	800	67	6.1°	(0.9,0.2)	6.8°	0.7
5 cm/s	3.4	3.24	12.3°	680	71	7.5°	(0.9,0.2)	6.8°	0.8
6 cm/s	3.6	3.24	14.2°	600	78	8.9°	(0.9,0.2)	6.5°	0.8
7 cm/s	3.7	3.24	15.6°	530	81	9.2°	(0.9,0.2)	6.5°	0.8
8 cm/s	3.8	3.24	16.7°	480	83	9.4°	(0.9,0.2)	6.3°	0.9
9 cm/s	3.8	3.24	16.7°	430	83	9.6°	(0.9,0.2)	6.3°	0.9
10 cm/s	3.8	3.24	16.7°	380	83	9.6°	(0.9,0.2)	6.3°	0.9

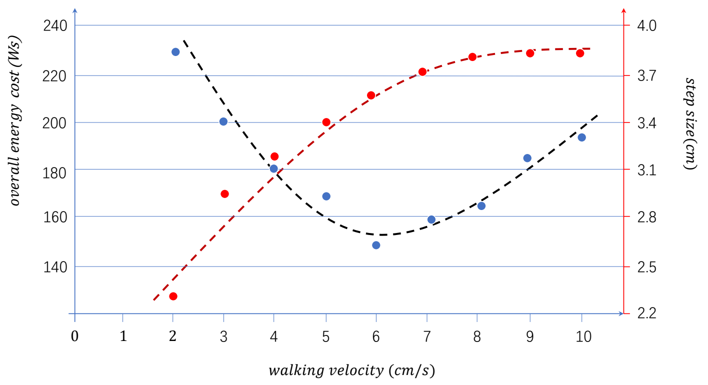

Fig. 6. The estimated energy consumption (the blue dots) for walking a distance of 75 cm, and the corresponding optimal step length (the red dots).

chosen parameter vector $\boldsymbol{\theta}$. We repeated this learning process 500 times, which took 3 weeks of computation time on a standard PC with CPU: Intel Core i5-2400 3.10GHz, and operating system: Windows 7 Enterprise. In all cases the algorithm converged to the same set of optimal parameter values. This is a strong indication that the parameter values represent a global optimum. Each evaluation required around 1200 learning steps.

Table 1 shows the parameter values for different walking speeds on a flat ground. We also verified the robustness of the identified parameter values. No parameter was on the edge of a stable region.

We computed the energy consumption of each gait. Figure 6 shows the total energy needed to walk a distance of 75 cm as well as the corresponding optimal step size of the gait. The figure shows that the most energy efficient walking speed for traversing a specified distance is 6 cm/s.

Experiments with the real Nao robot We evaluated the most energy efficient walking speed on a real Nao robot, version 5. The new walking gait enabled the Nao robot to walk on a wooden plank floor that is not completely flat, and

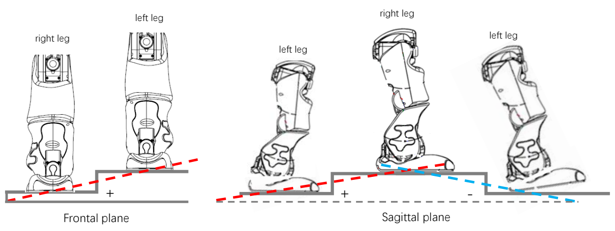

Fig. 7. A complex terrain can be modeled as the discrete slopes.

reduced the power consumption by 41% compared to standard gait of the Nao provided by Aldebaran. The accompanying video material[4] shows the Nao robot walking on a flat ground with our proposed gait controller at a speed of 6 cm/s.

5 Walking on uneven terrains

Since a step on uneven terrain causes a difference in altitude for the two feet of the biped robot, we assume that each step of the walk on an uneven ground can be viewed as a step on a (virtual) slope (see Figure 7). The angle of the slope is determined by the difference in altitude of both feet when they are firmly placed on the ground. In the sagittal direction, this corresponds to walking up or down a flat slope. In the lateral direction, this corresponds to walking perpendicular to the slope direction at a constant height. We define the slope angle to be positive if left foot is landed higher than the right foot in the frontal plane, or the new support foot is landed higher than the rear foot in the sagittal plane. Hence, the complex terrain with obstacles, including bumps, pits and slopes, can be modeled as slopes of variable angles.

We designed two series of experiments in the simulator *Webots* to obtain the optimal control parameters while walking on a specified slope type. In the experiments, also PGRL [11] is used to find the proper control parameters that can generate a stable walking gait on different slopes. In the first series of experiments the robot walks on slopes where the slope angle varies from -0.17 rad to 0.17 rad in the sagittal plane, which corresponds to -10° to 10°. We repeatedly let the robot walk 10 steps while running the reinforcement learning algorithm. We did not fixed walking speed of the robot as we did in the previous section. Instead, we choose to fix the step size θ_1 at 3.6 cm and the step time θ_4 at 650 ms, which determine the most energy efficient speed of 6 cm/s on a horizontal flat ground. Table 2 shows the learned parameter values for a stable walk.

In the second series of experiments required the robot to stand on slopes for which the tilt angle varied from 0 rad to 0.09 rad in the robot's frontal plane, which corresponds to -0° to 5°. Since in frontal plane, slopes with negative tilt angles are opposite to those with positive angles, the results of left leg on

[4] `https://project.dke.maastrichtuniversity.nl/robotlab/wp-content/uploads/naowalk.mp4`

Table 2. Experimentally learned control parameters for different slopes in the sagittal direction. p is the angle of the slopes. h is the height different of the feet, which is determined by $h = \theta_1 \sin(p)$. h is positive in an uphill step, negative in a downhill step. Parameter θ_1, θ_2, θ_4 and θ_9 are fixed.

p	h(cm)	θ_2(cm)	θ_3	θ_4(ms)	θ_5(ms)	θ_6	θ_7	θ_8	θ_9
-10°	-0.63	3.24	8.5°	650	64	0°	(0.92 0.28)	9.5°	0.8
-7°	-0.44	3.24	9.2°	650	68	1.4°	(0.91 0.27)	7.5°	0.8
-5°	-0.31	3.24	11.7°	650	70	4.1°	(0.91 0.26)	7.1°	0.8
-3°	-0.19	3.24	13.7°	650	75	5.7°	(0.90 0.22)	6.9°	0.8
0°	0.00	3.24	14.2°	650	78	8.9°	(0.90 0.20)	6.5°	0.8
3°	0.19	3.24	14.4°	650	76	10.2°	(0.86 0.16)	5.2°	0.8
5°	0.31	3.24	15.2°	650	75	13.5°	(0.85 0.14)	3.4°	0.8
7°	0.44	3.24	15.7°	650	74	14.9°	(0.82 0.14)	2.2°	0.8
10°	0.63	3.24	16.1°	650	73	16.3°	(0.78 0.12)	1.5°	0.8

slopes with negative angle are identical to the results of right leg on slopes with positive angle and vice versa. We repeatedly let the robot walk for 5 seconds while running the reinforcement learning algorithm. Table 3 shows the learned parameter values for a stable walk.

The Sagittal Controller The main task of the sagittal controller is to generate a leg-length policy that is able to adapt to unknown slopes. The leg-length policy is determined by two parameters: θ_3 (*Knee Bending*) and θ_5 (*Stretch Time*). The parameter θ_3 is determined by the bending of the knee of the swing leg at the end of the double support phase. So, the only parameter of the leg-length policy that can be controlled, is θ_5.

Although θ_5 depends on the slope angle p, we would like to make a more robust controller that can compensate for disturbances during a step; e.g., strong wind, collisions with other robots, etc., which may have a similar effect as walking on a slope. We therefore chose to use α (the angle between a virtual leg and its vertical line, see Figure 1), angular velocity $\dot{\alpha}$, angular acceleration $\ddot{\alpha}$ and θ_3 to determine the leg-length $l\delta(\alpha)$ of the stance leg directly. So, we no longer need the parameter θ_5. The inputs α, $\dot{\alpha}$ and $\ddot{\alpha}$, which can be determined using the Inertia Measurement Unit (IMU) of the Nao robot, implicitly identify the slope angle p. The mapping from $(\alpha, \dot{\alpha}, \ddot{\alpha}, \theta_3)$ to a knee-angle, which determines the leg-length, is realized by a simple neural network with only 8 neurons and 1 hidden layer.

Lateral Controller The main task of the lateral controller is to ensure that the robot is balanced during the double support phase. The lateral stability is determined by the parameters θ_7 (*Quadratic Bezier point*) and θ_8 (*Torso Roll Inclination*). Both parameters depend on the hight difference between the two feet (h in Table 2 and k in Table 3). *Observe* that the same height difference results in the same values for the parameters θ_7 and θ_8. To create a robust controller that can compensate for disturbances during a step; e.g., strong wind, collisions

Table 3. Experimentally learned control parameters for different slopes in the lateral direction. q is the angle of the slopes. h is the height different of the feet, which is determined by the distance of 10 cm between the feet and the angle q: $k = 10\sin(q)$. Parameter θ_1, θ_2, θ_4 and θ_9 are fixed.

q	k(cm)	θ_2^L(cm)	θ_3^L	θ_4(ms)	θ_5^L(ms)	θ_6^L	θ_7^L	θ_8^L	θ_9^L
0°	0.00	3.24	14.2°	650	78	8.9°	(0.90,0.20)	6.5°	0.8
1°	0.17	3.24	14.5°	650	76	10.1°	(0.86 0.16)	5.5°	0.8
2°	0.33	3.24	15.3°	650	75	13.5°	(0.84 0.14)	3.5°	0.8
3°	0.50	3.24	15.5°	650	74	15.1°	(0.80 0.12)	2.0°	0.8
4°	0.66	3.24	16.1°	650	72	16.8°	(0.78 0.12)	1.5°	0.8
5°	0.83	3.24	16.4°	650	70	17.3°	(0.76 0.12)	1.5°	0.8
q	k(cm)	θ_2^R(cm)	θ_3^R	θ_4(ms)	θ_5^R(ms)	θ_6^R	θ_7^R	θ_8^R	θ_9^R
0°	0.00	3.24	14.2°	650	78	8.9°	(0.90,0.20)	6.5°	0.8
1°	-0.17	3.24	13.5°	650	74	5.8°	(0.90 0.22)	6.7°	0.8
2°	-0.33	3.24	11.7°	650	70	4.0°	(0.91 0.26)	7.1°	0.8
3°	-0.50	3.24	9.6°	650	68	1.3°	(0.92 0.28)	8.5°	0.8
4°	-0.66	3.24	8.5°	650	65	0.0°	(0.92 0.29)	9.5°	0.8
5°	-0.83	3.24	8.2°	650	65	0.0°	(0.94 0.29)	9.5°	0.8

with other robots, etc., we generate the force policy directly instead of setting the parameter θ_7. We therefore use β' (the angle between the *swing leg* and the vertical axis in the frontal plane), angular velocity $\dot{\beta}$ and angular acceleration $\ddot{\beta}$, which can be determined by the IMU, as inputs for the lateral controller. The mapping from $(\beta', \dot{\beta}, \ddot{\beta})$ to a knee stiffness, which determines the force generated by the swing leg, is realized by a simple neural network with only 6 neurons and 1 hidden layer. Finally, we use θ_3, which encodes information about the height difference between the two feet, to determine the parameter θ_8. The mapping is realized be a neural network with 4 neurons and 1 hidden layer.

Controller design Figure 8 show the controller for walking on an uneven terrain. The accompanying video material[5] shows the Nao robot walking on an uneven terrain with our proposed gait controller in the simulator Webots

6 Conclusion

A design method for developing a controller that enables an energy efficient gait capable of walking on uneven terrains was presented in the paper. The resulting controller consists three simple neural networks. Based on these results we can draw the following conclusions:

– An energy efficient gait capable of walking on uneven terrains can be developed without making use of some complex mathematical model, and without some deep reinforcement learning approach, which is not always well understood and requires a large amount of training data. The proposed approach is

[5] `https://youtu.be/owHiGQm8WSg`

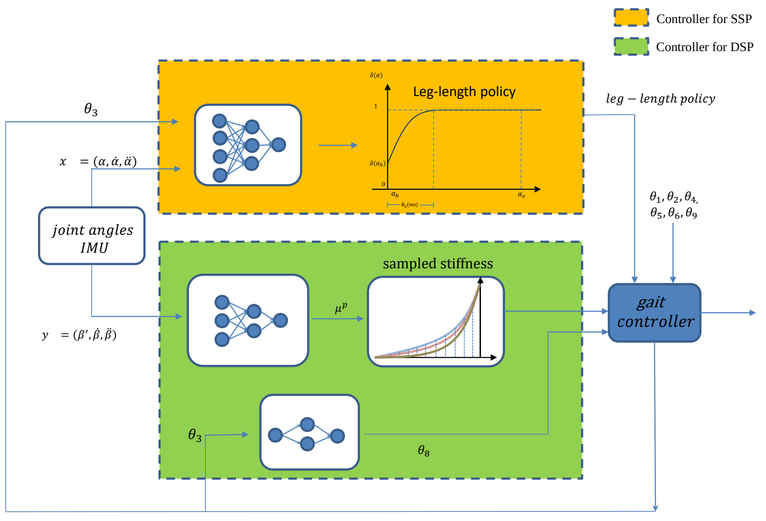

Fig. 8. Structure of the controller for the single and the double support phase. The top neural network outputs the sampled values of the stance leg's knee angle, the middle neural network outputs the sampled values of the swing leg's knee join stiffness, and the bottom neural network determines *Torso Roll Inclination*

well explainable. The controller exploits the physical properties of the robot without the special design of the partially passive dynamic walker.

– A simple abstract inverted pendulum model can be used to identify an energy efficient walking gait for a Nao robot. The gait can, after fine-tuning, also be used for other robots with a similar design. We conjecture that the approach can also be used for robots with a different design, such as robots that can toe-off, after adapting the function of the energy consumption.

Future research should focus on different types of robots, including those the can toe-off. Looking at the final controller, we observe that its simplicity is based on exploiting the physical properties of the robot. It would therefore be interesting to see whether the human neural network has a similar property. Since humans learn to walk quite easily, we conjecture that the answer is yes.

References

1. Chevallereau, C., Abba, G., Aoustin, Y., Plestan, F., Westervelt, E., de Wit, C.C., Grizzle, J.: RABBIT: a testbed for advanced control theory. IEEE Control Systems Magazine **23**(5), 57–79 (2003). https://doi.org/10.1109/MCS.2003.1234651
2. Chevallereau, C., Grizzle, J.W., Shih, C.L.: Asymptotically stable walking of a five-link underactuated 3-D bipedal robot. IEEE Transactions on Robotics **25**(1), 37–50 (2009). https://doi.org/10.1109/TRO.2008.2010366
3. Chevallereau, C., Grizzle, J.W., Shih, C.L.: Steering of a 3D bipedal robot with an underactuated ankle. In: 2010 IEEE/RSJ International Con-

ference on Intelligent Robots and Systems (IROS). pp. 1242–1247 (2010). https://doi.org/10.1109/IROS.2010.5648801

4. Collins, S., Ruina, A., Tedrake, R., Wisse, M.: Efficient bipedal robots based on passive-dynamic walkers. Science **307**(5712), 1082–1085 (2005). https://doi.org/10.1126/science.1107799
5. Collins, S.H., Ruina, A.: A bipedal walking robot with efficient and human-like gait. In: roceedings of the 2005 IEEE International Conference on Robotics and Automation. pp. 1983–1988 (2005). https://doi.org/10.1109/ROBOT.2005.1570404
6. Garcia, M., Chatterjee, A., Ruina, A., Coleman, M.J.: The simplest walking model: stability, complexity, and scaling. Journal of Biomechanical Engineering **120**(2), 281–8 (1998)
7. Hobbelen, D.G., Wisse, M.: Limit cycle walking. In: Hackel, M. (ed.) Humanoid Robots, chap. 14. IntechOpen, Rijeka (2007). https://doi.org/10.5772/4808
8. Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Biped walking pattern generation by using preview control of zero-moment point. In: 2003 IEEE International Conference on Robotics and Automation. vol. 2, pp. 1620–1626. IEEE (2003)
9. Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., Hirukawa, H.: The 3D linear inverted pendulum mode: A simple modeling for a biped walking pattern generation. In: 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). vol. 1, pp. 239–246 (2001). https://doi.org/10.1109/IROS.2001.973365
10. Kajita, S., Tani, K.: Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. In: 1991 IEEE International Conference on Robotics and Automation. pp. 1405–1411 (1991). https://doi.org/10.1109/ROBOT.1991.131811
11. Kohl, N., Stone, P.: Policy gradient reinforcement learning for fast quadrupedal locomotion. In: IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. vol. 3, pp. 2619–2624 (2004). https://doi.org/10.1109/ROBOT.2004.1307456
12. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. In: NIPS Deep Learning Workshop (2013)
13. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (02 2015). https://doi.org/10.1038/nature14236
14. Pratt, J., Chew, C.M., Torres, A., Dilworth, P., Pratt, G.: Virtual model control: An intuitive approach for bipedal locomotion. The International Journal of Robotics Research **20**(2), 129–143 (2001)
15. Srinivasan, M., Ruina, A.: Computer optimization of a minimal biped model discovers walking and running. Nature **439**, 72–75 (2006). https://doi.org/doi:10.1038/nature04113
16. Sun, Z., Roos, N.: An energy efficient gait for a nao robot. BNAIC (2013)
17. Sun, Z., Roos, N.: An energy efficient dynamic gait for a nao robot. In: 2014 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). pp. 267–272 (2014). https://doi.org/10.1109/ICARSC.2014.6849797
18. Vukobratovic, M., Borovac, B., Surla, D., Stokic, D.: Biped locomotion: dynamics, stability, control and application, Scientific Fundamentals of Robotics, vol. 7. Springer-Verlag (1990)