Optimizing end-effector force during the sit-stand task on the Talos humanoid bipedal robot

Luka Mišković[1,2], Rebeka Kropivšek Leskovar [1,2], Andrej Gams[1,2], Tadej Petrič[1,2]

[1]*Odsek za avtomatiko, biokibernetiko in robotiko, Inštitut "Jožef Stefan", Jamova cesta 39, 1000 Ljubljana*
[2]*Mednarodna podiplomska šola Jožefa Stefana, Jamova cesta 39, 1000 Ljubljana*
E-pošta: {luka.miskovic, rebeka.leskovar, andrej.gams, tadej.petric}@ijs.si

Abstract

In this paper, we present the demanding task of a humanoid bipedal robot Talos getting up from a squat posture by finding the optimal force applied to the robot's end-effector by pulling it. The corresponding force acts together with the Talos' own torque in the joints. First, we recorded demonstrations of sit-to-stand motions from human subjects using the Optitrack motion capture system and mapped the demonstrated motions to the humanoid robot. We derived a cost function that was minimized using fmincon and a genetic algorithm. The results are presented in the simulation using Matlab and MuJoCo. The work is part of the Switch project, whose goal is to enhance robot learning performance by role reversal between robots and humans, and to implement complex human dynamics into robot behavior. The first step is that the human helps the robot stand up, and the second step is for the robot to help the human to stand up. In this paper, the first procedure is elucidated.

1 Introduction

We humans are able to perform dynamic and dexterous movements with relative ease: walking, jumping, and dance are just one of a few examples. Standing up from a chair is one of these everyday tasks that appears to be easy, but is actually quite complex and dynamic and can become challenging and dangerous as we age. The success of this task requires whole body coordination, balance and high fidelity control. In the field of robotics, we are interested in movements such as standing up and seek to better understand their complexities, as this allows us to improve current state-of-the-art robotic systems. Traditional robots require users to possess programming skills, which makes robots inaccessible to the general public [1]. On the other hand, future generations of home and service robots, including humanoid robots, are expected to perform tasks in more natural and highly dynamic environments created for humans, which is why it is necessary for us to study the natural human movement and how to implement it into robot control systems.

One such method is learning from demonstration (LfD), which has been used to help robots independently implement manipulation behavior by observing movement performed by human demonstrators as seen in [2]. In LfD applications, example trajectories can be captured in various ways, e.g. kinesthetic demonstration, motion-sensor demonstration, teleoperation, and methods like visual observation [3] [4]. After example trajectories are recorded, Inverse kinematics is often used to calculate the joint angles [3], however such operations need relatively high computational cost. In [5] a particle filter is used for joint angle imitation instead, in order to realize a real-time imitation of a humanoid robot through observation of human demonstration.

Figure 1: The concept of humanoid helper robot in household environment.

In general, to achieve robot learning from demonstration, we need to address three challenges: the problem of correspondence, generalization, and robustness against perturbation [6]. The problem of correspondence occurs due to the robots and human differences in kinematics, i.e. they do not have links and joints in the same place. Due to this, learning by demonstration is only feasible when the demonstrated movement can be generalized, for instance with different target positions. However, accurate repetition of the observed movement is unrealistic in a dynamic environment, in which obstacles can suddenly appear [1]. Due to this fact, in LfD robustness against perturbation is needed.

With all the studies mentioned the robots in human-physical collaboration are still facing a shortcoming due to their limitations in observing and adapting to human dynamics. The goal of the Switch project is to develop methods that can address these known limitations and allow the robot to efficiently observe human dynamics in real-time and learn anticipatory models from demonstration. The first step achieving this goal is for the robot to be able to observe a human picking it up, before switching the roles. Due to safety precautions however, we must first find out the amount of force needed to safely pickup the robot. This is why the goal of this paper is to find the external forces needed to lift the Talos on its feet.

To do so, we use two Matlab optimization methods by first recording the movements of humans standing up from chairs and later transferring the recorded movements to the humanoid robot Talos. The paper is organized as follows. The section 2 presents the kinematics of Talos and the methods used. It is also explained how the cost function is derived. Section 3 shows the results obtained. Furthermore, the discussion follows in Section 4 after the result.

2 Methods

2.1 Humanoid bipedal robot Talos

The robot used in this study was the bipedal humanoid robot Talos. Its kinematics is explained in this chapter so as to better understand the difference between human. The bipedal humanoid TALOS, built by the company PAL-Robotics, has 32 degrees of freedom, of which two are in the head, seven per each arm, one per hand, two in the waist and six per each leg. TALOS is fully torque-controllable thanks to the torque sensor feedback in all joints, which enables powerful sensing and multi-contact motions. The closed loop torque control can be used for whole-body control inverse dynamics and safe interaction with the environment [7]. Figure 2 shows the kinematics of Talos.

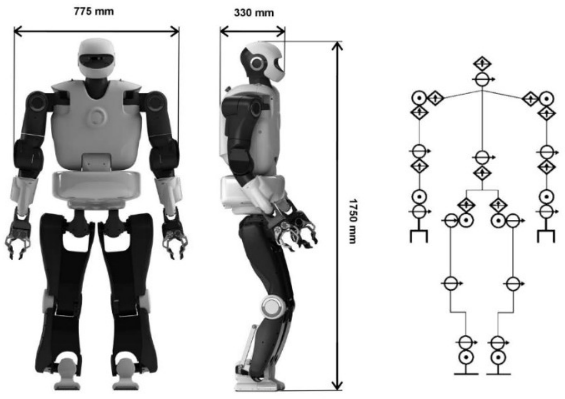

Figure 2: Kinematics of TALOS

2.2 Optimization

Force profile: In this section is described the process of finding the optimal force applied to the robot's end-effector by pulling the Talos from its sitting position. To describe a force we used a parametric description. In this description, force is defined as a linear combination of radial basis functions (RBF). As proposed in [8], among many possible radial basis functions, those with Gaussian kernels were used. They are defined as:

$$\Psi(x) = e^{-\frac{(x-c)^2}{2\sigma}}, \tag{1}$$

where c defines the center, σ defines the width of the kernel function and x is the phase parameter (see [8] for details).

To describe a force trajectory we use a parametric description:

$$f(x) = \frac{\sum_{j=1}^{m} w_j \Psi_j(x)}{\sum_{j=1}^{m} \Psi_j(x)}, \tag{2}$$

where m is the number of kernel functions, and w_j are the weights that define the path.

We define $\mathbf{\Phi}$ as a row vector with components:

$$\boldsymbol{\Phi_j}(x) = \frac{\boldsymbol{\Psi_j}(x)}{\sum_{j=1}^{m} \boldsymbol{\Psi_j}(x)}, \tag{3}$$

which yields:

$$\boldsymbol{f}(x) = \mathbf{\Phi}(x)\mathbf{w}, \tag{4}$$

where $\mathbf{w}$ is a vector with elements w_j.

The corresponding weights can be found by solving Eq. (4) using pseudoinverse:

$$\mathbf{w} = (\hat{\mathbf{\Phi}}^T \hat{\mathbf{\Phi}})^{-1} \hat{\mathbf{\Phi}}^T \hat{\boldsymbol{f}}, \tag{5}$$

where $\hat{\boldsymbol{f}}$ are the path values and $\hat{\mathbf{\Phi}}$ basis vectors for all path steps.

To get the optimal force profile the weights of radial basis functions need to be optimized. This is done by optimizing the cost function. If not stated otherwise we used $m = 6$ RBF functions.

Cost function: The cost function used in this study was composed of four quadratic functions. It is defined as:

$$\boldsymbol{J} = \sum_{n=1}^{m} K_n \boldsymbol{f}_n, \quad K_n \geq 0, \tag{6}$$

where $\boldsymbol{f}_n$ denotes the quadratic loss functions, K_n are the weights, which determine the importance of each quadratic loss function, and m is the number of quadratic loss functions.

The first quadratic loss function takes the sum of all the positions of the Talos' head in time and tries to bring them to the desired position. This ensures that in the end, the robot stands upright. Talos' positions and orientations were obtained in simulation during its movement, using MuJoCo dynamic library. The first quadratic loss function is defined as:

$$\boldsymbol{f}_1 = \sum_{t=0}^{t_m} (p_H - p_{H,d})^2, \tag{7}$$

where t_m denotes the total time during getting up, p_H is the measured head position, and $p_{H,d}$ is the desired head position.

To increase the stability of the robot, a second quadratic loss function was used to ensure that the pitch angle of the Talos' feet is minimal, or ideally zero, throughout the standing up process. Accordingly, the second quadratic loss function is derived as:

$$\boldsymbol{f}_2 = \sum_{t=0}^{t_m} \theta^2, \tag{8}$$

where θ is the pitch angle of the robot's feet.

Similarly, the third quadratic loss function minimizes the velocity of the torso, which as well increases the stability of the robot and is defined as:

$$\boldsymbol{f}_3 = \sum_{t=0}^{t_m} \dot{p_T}^2, \tag{9}$$

where $\dot{p_T}$ is the torso velocity.

The last part of the cost function finds the weights of the Gaussian kernel functions and is defined as:

$$\boldsymbol{f}_4 = \sum_{j=1}^{m} {w_j}^2, \tag{10}$$

where $\boldsymbol{w}$ is the vector with the weights of Gaussian kernel functions, which is found through the optimization process.

Finally, with the values of the weights w_n from the Eq. 6 adjusted, the lower and upper bounds of optimization and the initial guess defined, the optimization problem was solved by finding the solution vector $\boldsymbol{w}$ that minimizes the cost function value $\boldsymbol{f}_{val}$, defined as:

$$\boldsymbol{f}_{val} = \min_{\boldsymbol{w}} \boldsymbol{J}(w). \tag{11}$$

The cost function was optimized using two different Matlab functions: fmincon and the genetic algorithm. Fmincon finds a constrained minimum of a function of several variables through an iterative process. In our case, the 'active-set' algorithm was set in options. On the other hand, genetic algorithm generates a population of points at each iteration and modifies a population of individual solutions. Here, the best point in the population approaches an optimal solution [9].

Movement of human subject's six joints during sit-to-stand motion were captured using Optitrack motion capture system. These angles were mapped to the six joints of the humanoid robot: ankle, knee, hip, torso, shoulder, and elbow. The human subject was about the same height as the robot. This enabled the robot to imitate natural human movement during standing up as close as possible. The force applied to the robot's arms can be understood as an external perturbation force that acts on the robot's end-effector and helps it to stand up. The Talos' joints are controlled in position mode.

The resulting force calculated with the optimization is applied to the end effector of the Talos robot, i.e., the hands. By multiplying the force by the rotation matrix, we ensure that the direction of the force is aligned with the robot arms.

3 Results

When using fmincon solver, the value of the cost function converges faster compared to the genetic algorithm which we can see in Figure 3. By analyzing the time required for each method to find the optimal solution, we have found that fmincon takes about 41 seconds, while the genetic algorithm takes 4922 seconds on a typical desktop pc with i7 processor.

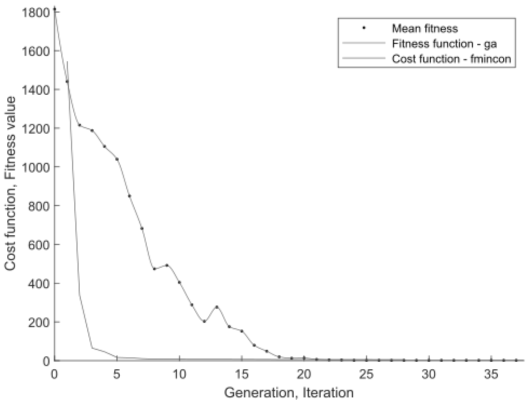

Figure 3: Comparison of convergence between the fmincon solver and the genetic algorithm.

The resulting optimized force profiles found using fmincon and genetic algorithm methods are shown in Figure 4. For comparising we also show the results of force optimization with $m = 20$ RBFs.

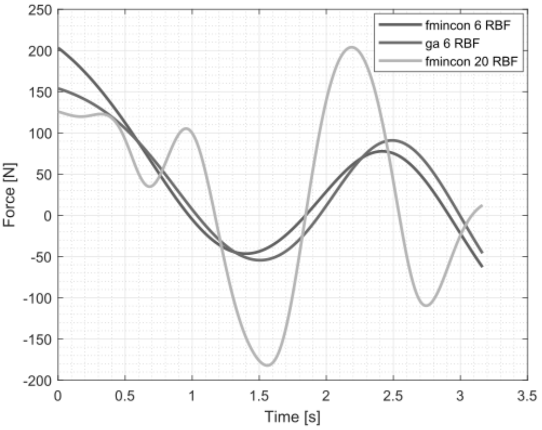

Figure 4: Optimized forces with different solvers and number of RBFs

Figure 5 shows the result of Talos standing up in simulation environment.

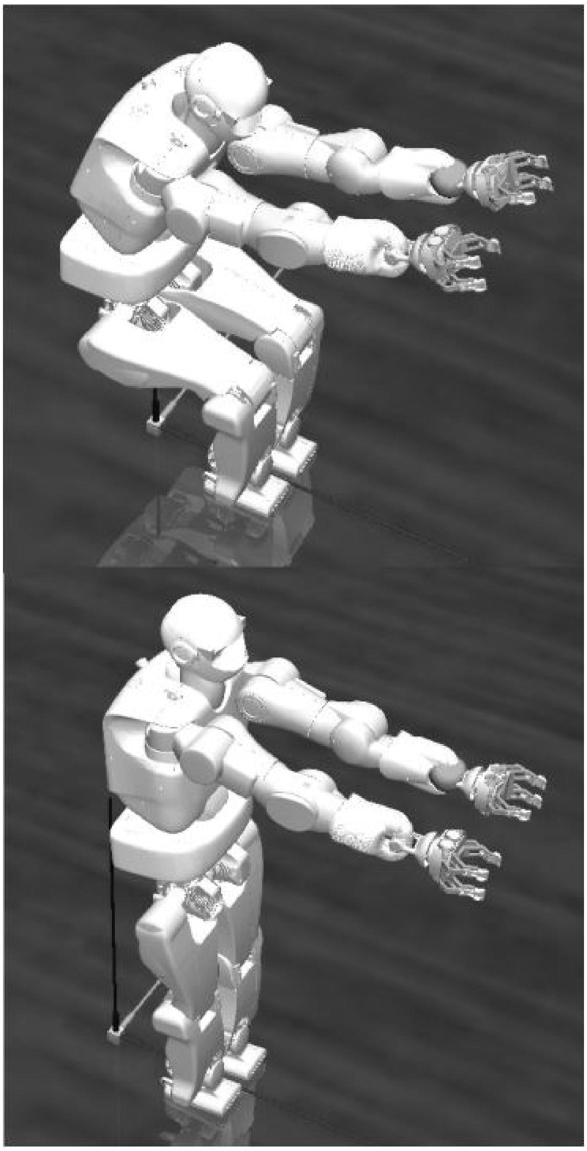

Figure 5: Talos robot in MuJoCo in sitting position (top) and standing position (bottom)

4 Discussion

In this paper, we presented a way to find the optimal force path needed to lift the bipedal mobile robot Talos to the standing position by pulling its hands, using two different optimization methods in Matlab. The results of the experiment show that the bipedal mobile robot Talos is able to stand up successfully using the methods proposed in this paper. If at least the approximate trajectory of the force is not known before real-life experiments, it could be a potential safety hazard.

Analyzing the graphs, we can see that both the fmincon solver and the genetic algorithm resulted in very similar solutions (force profile) when the weights of 6 RBFs were optimized. However, increasing the number of RBFs resulted in overfitting and more profile curvature, as can be seen in Figure 4.

We also found that the fmincon method takes less time to find the minimal solution compared to the genetic algorithm. This is to be expected since fmincon uses the local gradient to tune the parameters, which means it is faster, but it might get stuck in a local minimum, while the genetic algorithm generates a random trial and then shuffles them to generate a new generation that ends up finding a global optimum.

In general, both methods can be used for similar optimization problems, but in our case, due to the fact that both methods gave the same result, fmincon can be considered as the preferred method.

The work presented in this paper is an important step to get a first impression about the shape and amplitude of the profile. This allows us to gain an approximation of the force profile needed to safely lift Talos from a seated position in real life. With this knowledge, Talos will be able to help a person stand up in the future.

Acknowledgement: This work was supported by Slovenian Research Agency grant PR-10489 and N2-0153.

References

[1] Z. Zhu and H. Hu, “Robot learning from demonstration in robotic assembly: A survey,” *Robotics*, vol. 7, no. 2, 2018.

[2] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, “Recent Advances in Robot Learning from Demonstration,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 297–330, 2020.

[3] Y. Ou, J. Hu, Z. Wang, Y. Fu, X. Wu, and X. Li, “A Real-Time Human Imitation System Using Kinect,” *International Journal of Social Robotics*, vol. 7, no. 5, pp. 587–600, 2015. [Online]. Available: http://dx.doi.org/10.1007/s12369-015-0296-9

[4] R. Dillmann, “Teaching and learning of robot tasks via observation of human performance,” *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 109–116, 2004.

[5] Y. Kondo, S. Yamamoto, and Y. Takahashi, “Real-Time Posture Imitation of Biped Humanoid Robot Based on Particle Filter with Simple Joint Control for Standing Stabilization,” *Proceedings - 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems and 2016 17th International Symposium on Advanced Intelligent Systems, SCIS-ISIS 2016*, pp. 130–135, 2016.

[6] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, “Learning and generalization of motor skills by learning from demonstration,” pp. 763–768, 2009.

[7] O. Stasse, T. Flayols, R. Budhiraja, K. Giraud-Esclasse, J. Carpentier, J. Mirabel, A. Del Prete, P. Soueres, N. Mansard, F. Lamiraux, J. P. Laumond, L. Marchionni, H. Tome, and F. Ferro, “TALOS: A new humanoid research platform targeted for industrial applications,” *IEEE-RAS International Conference on Humanoid Robots*, pp. 689–695, 2017.

[8] L. Žlajpah and T. Petrič, “Unified Virtual Guides Framework for Path Tracking Tasks,” *Robotica*, vol. 38, no. 10, pp. 1807–1823, 2020.

[9] C. Mathworks, “Statistics and Machine Learning Toolbox$^{\text{TM}}$ User’s Guide R2020a,” *MATLAB Mathworks Inc*, vol. 2020a, no. A, p. 7984, 2020. [Online]. Available: www.mathworks.com