Learning to Gesticulate by Observation Using a Deep Generative Approach

Unai Zabala, Igor Rodriguez(✉), José María Martínez-Otzeta, and Elena Lazkano

Computer Science and Artificial Intelligence, Faculty of Informatics, UPV/EHU, Manuel Lardizabal 1, 20018 Donostia, Spain
igor.rodriguez@ehu.eus
http://www.sc.ehu.es/ccwrobot

Abstract. The goal of the system presented in this paper is to develop a natural talking gesture generation behavior for a humanoid robot, by feeding a Generative Adversarial Network (GAN) with human talking gestures recorded by a Kinect. A direct kinematic approach is used to translate from human poses to robot joint positions. The provided videos show that the robot is able to use a wide variety of gestures, offering a non-dreary, natural expression level.

Keywords: Social robots · Motion capturing and imitation · Generative Adversarial Networks · Talking movements

1 Introduction

Social robotics [4] aims to provide robots with artificial social intelligence to improve human-machine interaction and to introduce them in complex human contexts. The demand for sophisticated robot behaviors requires to model and implement human-like capabilities to sense, to process, and to act/interact naturally by taking into account emotions, intentions, motivations, and other related cognitive functions.

Talking involves spontaneous gesticulation; postures and movements are relevant for social interactions even if they are subjective and culture dependent. Aiming at building trust and making people feel confident when interacting with them, socially acting humanoid robots should show human-like talking gesticulation. Therefore, they need a mechanism that generates movements that resembles humans' in terms of naturalness. A previous work [24] made use of gestures selected from a set of movements previously compiled. Those gestures

This work has been partially supported by the Basque Government (IT900-16 and Elkartek 2018/00114) and the Spanish Ministry of Economy and Competitiveness MINECO/FEDER (RTI 2018-093337-B-100, MINECO/FEDER, EU). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

M. A. Salichs et al. (Eds.): ICSR 2019, LNAI 11876, pp. 666–675, 2019.
https://doi.org/10.1007/978-3-030-35888-4_62

were then randomly concatenated and reproduced according to the duration of the speech. That approach was prone to produce repetitive movements and resulted in unnatural jerky expression.

The goal of the system presented in this paper is to develop a natural talking gesture generation behavior for a humanoid robot. At this step we aim to give a step forward by training a Generative Adversarial Network (GAN) gesture generation system with movements captured directly from humans. A Kinect sensor is used to track the skeleton of the talking person and a GAN is trained to generate a richer and more natural talking gesticulation.

Gestures (head, hands and arms movements) are used both to reinforce the meaning of the words and to express feelings through non-verbal signs. Among the different types of conversational movement of arms and hands synchronised with the flow of the speech, beats are those not associated with particular meaning [18]. References to talking gestures of the present work will be limited to beats.

The robotic platform employed in the performed experiments is a Softbank Robotics Pepper robot[1]. Currently, our robot is controlled using the *naoqi_driver*[2] package that wraps the needed parts of NAOqi[3] API and makes them available in ROS[4].

2 Related Work

According to Beck et al. [3], there are three main robot motion generation approaches: manually creating motion, motion capturing, and motion planning; for manual creation, it is required to set each joint position of the humanoid robot for each key frame (time step); the motion capture-based approach tries to mimic human gestures, recording human movements and mapping these data to a humanoid robot [22]; and motion planning approach relies on kinematics and/or dynamics equations to solve a geometric task. They found that the motion capturing approach produces the most realistic results, because the robot reproduces previously captured human movements.

Motion capturing and imitation is a challenge because humans and robots have different kinematic and dynamic structures. Motion capture (MoCap) is the process of recording motion data through any type of sensor. Applications of MoCap systems range from animation, bio-mechanics, medicine to sports science, entertainment, robotics [21,31] or even study of animal behavior [27]. MoCap systems rely on optical technologies, and can be marker-based (e.g. Vicon[5]) or markerless like RGB-D cameras. While the former ones provide more accurate results, the latter ones are less prone to produce gaps (missing values) that need

[1] https://www.ald.softbankrobotics.com/en/robots/pepper.
[2] http://wiki.ros.org/naoqi_driver.
[3] http://doc.aldebaran.com/2-5/naoqi/index.html.
[4] http://www.ros.org.
[5] https://www.vicon.com/.

to be estimated [19,29]. Many approaches make use of the Kinect sensors due to its availability [1,9,20].

No matter the motion capture system being used, there is a need to transfer human motion to the robot joints. This can be done by direct kinematics, adapting captured human joint angles to the robot. Or alternatively, inverse kinematics calculates the necessary joint positions given a desired end effector's pose.

On the other hand, generative models are probabilistic models capable of generating all the values for a phenomenon. Unlike discriminative models, they are able to generate not only the target variables but also the observable ones [28]. They are used in machine learning to (implicitly or explicitly) acquire the distribution of the data for generating new samples. There are many types of generative models. For instance, Bayesian Networks (BNs) [7], Gaussian Mixture Models (GMMs) [8] and Hidden Markov Models (HMMs) [23] are well known probability density estimators.

Focusing on generative models used for motion generation, in [14] the authors propose the combination of Principal Component Analysis (PCA) [30] and HMMs for encoding different movement primitives to generate humanoid motion. Tanwani [28] uses HSMM (Hidden Semi-Markov Models) for learning robot manipulation skills from humans. Regarding on social robotics, some generative approaches are being applied with different objectives. In [17] Manfrè et al. use HMMs for dance creation and in a later work they try variational auto-encoders again for the same purpose [2].

Deep learning techniques have also been applied to generative models, giving rise to deep generative models. A taxonomy of such models can be found in [10]. In particular Generative Adversarial Networks (GANs) [11] are semi-supervised emerging models that basically learn how to generate synthetic data from the given training data. GANs are deep generative models capable to implicitly acquire the probability density function in the training data, being able to automatically discover the internal structure of datasets by learning multiple levels of abstraction [15]. Gupta et al. [12] extend the use of GANs to generate socially acceptable motion trajectories in crowded scenes in the scope of self-driving cars. In [26] GANs showed to overcome other generative approaches such as HMM and GMM when confronted to the task of motion generation. In that work, movements produced synthetically (using choregraph) were used to train the different generative approaches. Instead, in this paper we feed the GAN with movements obtained by observing and capturing human talking gestures.

3 Developed Approach to Enhance Robot Spontaneity

The GAN used in the current approach takes as input only proprioceptive joint position information. In order to feed the GAN with natural motion data, a motion capturing approach is employed. Thus, these two aspects are exhaustively described here on.

3.1 Human Motion Capture and Imitation

In [25], direct kinematics was used to teleoperate a NAO robot. Human skeleton obtained with a Kinect was tracked and arm movements were replicated by the robot, while walking motions were commanded by using different spatial key movements. As the goal was to teleoperate the robot, there was no need for subtle and continuous motion since the arms only required to reach single poses when demanded by the operator. On the contrary, gesticulation requires continuous arm motion and involves also hands, head and fingers. Although the present work makes use of a similar motion capture and mapping system, the presented system has been enriched to cover all the aspects involved.

The Kinect uses structured light (depth map) and machine learning to infer body position [16]. The OpenNI based `skeleton_markers`[6] ROS package is able to extract in real time the 15 joints associated to the human skeleton.

Mapping Human's Arms into Robot's Space

Human arms have 7 degrees of freedom (DoF): a spherical joint at the shoulder, a revolute joint at the elbow and a spherical joint at the wrist. On the contrary, our humanoid's arms have 5 DoF: two at the shoulder (pitch and yaw) and elbow (yaw and roll), and one at the wrist (yaw) (Fig. 1). Thereby, the movement configurations of human and robot arms differ.

To transform the Cartesian coordinates obtained from the Kinect into *Pepper*'s coordinate space a joint control approach was employed. Note that the transformations are performed to the reference system of each individual joint, not to a robot's global reference frame. On the following explanation we will focus on the left arm. The analysis of the right arm is similar and it will be omitted here.

Pepper's left arm has five joints[7] (see Fig. 1): shoulder roll (LS_α) and pitch (LS_β), elbow roll (LE_α) and yaw (LE_γ) and wrist yaw (LW_γ). The `skeleton_markers` package can not detect the operator's hands' yaw motion and thus, LW_γ joints cannot be reproduced using the skeleton information. We chose another approach for LW_γ, that will be explained later on.

In order to calculate the shoulder's roll angle (LS_α) we use the dot product of the distance vector between both shoulders ($\overline{LRS}$) and the length vector between the shoulder and the elbow ($\overline{LSE}$). Note that, before computing that product, $\overline{LRS}$ and $\overline{LSE}$ vectors must be normalized. The LS_α angle is calculated in the Kinect's coordinate space, therefore, it must be transformed into the robot's coordinate space by rotating it $\frac{-\pi}{2}$ radians (Eq. 1).

$$\begin{aligned} LS_\alpha &= \arccos\left(\overline{LRS} \cdot \overline{LSE}\right) \\ LS_\alpha^{robot} &= LS_\alpha - \frac{\pi}{2} \end{aligned} \tag{1}$$

[6] http://wiki.ros.org/skeleton_markers.
[7] http://doc.aldebaran.com/2-8/family/pepper_technical/joints_pep.html.

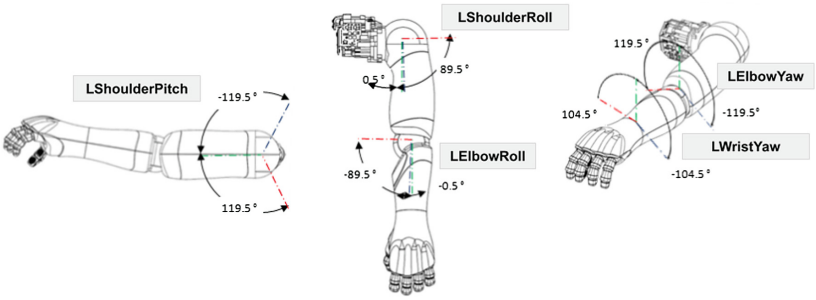

Fig. 1. Pepper's left arm joints and actuators (from Softbanks official Pepper's user guide (see Footnote 7)).

Elbow's roll (LE_α) angle is calculated in the same way as shoulder's roll angle (LS_α) but the length vector between the shoulder and the elbow ($\overline{LSE}$) and the length vector between the elbow and the hand ($\overline{LEH}$) are used instead. Again, those vectors need to be normalized and transformed to the robot's space, in this case by rotating it $-\pi$ radians.

With respect to elbow's yaw angle (LE_γ) calculation we use only the y and z components of the $\overline{LEH}$ vector. After normalizing $\overline{LEH}$, Eq. 2 is applied to obtain the LE_γ. Lastly, a range conversion is needed to get LE_γ^{robot} (from $[\frac{\pi}{2}$, $\pi]$ to $[-\frac{\pi}{2}, 0]$ and from $[-\pi, -\frac{\pi}{2}]$ to $[0, \pi]$).

$$\begin{aligned} LE_\gamma &= \arctan \frac{\overline{LEH}_z}{\overline{LEH}_y} \\ LE_\gamma^{robot} &= rangeConv(LE_\gamma) \end{aligned} \tag{2}$$

To conclude with the joints, the shoulder pitch angle (LS_β) is calculated by measuring the angle between the shoulder to elbow vector and the z axis. $z = 0$ occurs with the arm extended at 90° with respect to the torso. Thus, lowering the arm produces negative pitch angle while raising it above the shoulder produces positive angular values.

The LS_β can be defined as:

$$\begin{aligned} \|A\| &= LSE_z \quad \text{(by definition)} \\ \sin(LS_\beta) &= \frac{\|A\|}{\|\overline{LSE}\|} = \frac{\|A\|}{1} \\ LS_\beta^{robot} &= \arcsin(LSE_z) \end{aligned} \tag{3}$$

where LSE_z is the Z coordinate of the shoulder to elbow vector.

Mapping Human's Hands into Robot's Space

Hands movements are common in humans while talking. We do rotate wrist and open and close hands, move fingers constantly. Unfortunately, the skeleton capturing system we are using does not allow to detect such movements. It is possible though to capture the state of the hands using a different approach.

The developed solution forces the user to wear coloured gloves, green in the palm of the hand and red in the back (Fig. 2). While the human talks, hands coordinates are tracked and those positions are mapped into the image space and a subimage is obtained for each hand. Angular information is afterwards calculated by measuring the number of pixels (max) of the outstanding color in a subimage. Equation 4 shows the procedure for the left hand. N is a normalizing constant and $maxW_\gamma$ stands for the maximum wrist yaw angle of the robot.

$$\begin{cases} LW_\gamma^{robot} = max/N \times maxW_\gamma & \text{if } max \text{ is palm} \\ LW_\gamma^{robot} = \frac{max-N}{N} \times maxW_\gamma & \text{otherwise} \end{cases} \tag{4}$$

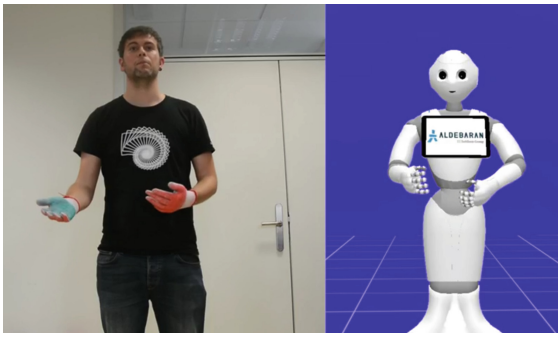

Fig. 2. Snapshot of a data capture session.

In addition LE_γ is modified when humans palms are up (subimage has only green pixels) to easy the movement of the robot.

Regarding the fingers, as they cannot be tracked, their position is randomly set at each skeleton frame.

Mapping Human's Head into Robot's Space

Humans move the head while talking and thus, head motion should also be captured and mapped. The robot's head has 2 DoFs that allow the head to move left to right (yaw) and up and down (pitch) as shown in Fig. 3.

The Kinect skeleton tracking program gives us the (neck and) head 3D positions. The approach taken for mapping the yaw angle to the robot's head consists of applying a gain K_1 to the human's yaw value, once transformed into the robot space by a $-\frac{\pi}{2}$ rotation (Eq. 5).

$$H_\gamma^{robot} = K_1 \times H_\beta \tag{5}$$

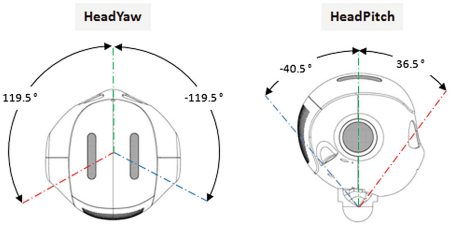

Fig. 3. Pepper's head joints and actuators (from Softbanks official Pepper's user guide (see Footnote 7)).

In order to approximate head's pitch angle, the head to neck vector ($\overline{HN}$) is calculated and rotated $-\frac{\pi}{2}$ and then, its angle is obtained (Eq. 6). Note that robot's head is an ellipsoid instead of an sphere. To avoid unwanted head movements a lineal gain is applied to the final value.

$$H_{\beta}^{robot} = \arctan\left(rotate(\overline{HN}, -\frac{\pi}{2})\right) + |K_2 * H_{\gamma}| \tag{6}$$

3.2 Generative Model

GAN networks are composed by two different interconnected networks. The *Generator* (G) network generates possible candidates so that they are as similar as possible to the training set. The second network, known as *Discriminator* (D), judges the output of the first network to discriminate whether its input data are "real", namely equal to the input data set, or if they are "fake", that is, generated to trick with false data.

The training dataset given to the D network contained 2018 unit of movements (UM), being each UM is a sequence of 4 consecutive poses, and each pose 14 float numbers corresponding to joint values of head, arms, wrists and hands (finger opening). These samples were recorded by registering 5 different persons talking, about 9 min overall.

The D network is thus trained using that data to learn its distribution space and its input dimension is 56. On the other hand, the G network is seeded through a random input with a uniform distribution in the range $[-1, 1]$ and with a dimension of 100. The G intends to produce as output gestures that belong to the real data distribution and that the D network would not be able to correctly pick out as generated. Figure 4 depicts the architecture the generator and discriminator networks.

GAN has been trained for 2000 epochs and its hyper-parameter have been tuned experimentally; we set up a batch size of 16, a learning rate of 0.0002, Adam [13] as optimization method, and $\beta_1 = 0.5$ and $\beta_2 = 0.999$ as its parameters.

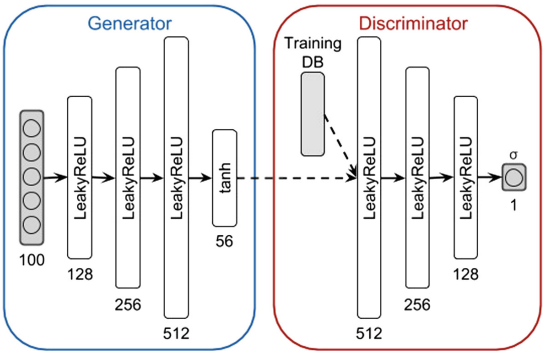

Fig. 4. GAN setup for talking gesture generation.

4 Results

The obtained robot performance can be appreciated in the following videos:

1. The first video[8] shows some instants recorded during the process of generating the database of movements captured through the motion capturing and imitation mechanisms. On the left side the participant talking and gesticulating is displayed, while the simulated robot mimicking the movements in real time (without GAN) is shown in the right side.
2. A second video[9] shows the evolution of the robot behavior during different steps of the training process. The final number of epochs was empirically set to 2000 for the model that has been integrated into the gesture generation system.

Notice that the temporal length of the audio intended to be pronounced by the robot determines the number of units of movement required to the generative model. Thus, the execution of those units of movements, one after the other, defines the whole movement displayed by the robot.

5 Conclusions and Further Work

In this work a talking gesture generation system has been developed using a GAN feeded with natural motion data obtained through a motion capturing and imitation system. The suitability of the approach is demonstrated with a real robot. Results show that the obtained robot behavior is appropriate, and thanks to the movement variability the robot expresses itself with naturalness.

As further work, we intend to improve the skeleton capture process by using more robust systems, such as OpenPose [5,6] or wrnchAI[10] that allow to capture more detailed movements. In this way, the speakers would not need to wear the

[8] https://www.youtube.com/watch?v=iW1566ozbdg.
[9] https://www.youtube.com/watch?v=1It_Y_AEnts.
[10] https://wrnch.ai/.

gloves, that somehow are conditioning them. Moreover, speakers tend to behave in an constricted way when recorded. A more powerful skeleton tracker system would allow to use recorded videos from real talks and to build a more objective database. With respect to the mapping process, in [20] direct kinematics is compared with two inverse kinematics approaches and the neuro fuzzy approach seems to improve the direct one. The use of a more effective method to translate human poses to robot poses could also produce better movements.

The work presented here pretends to be the starting point to acquire a richer gesture set, such as emotion-based gestures or context related gestures. Moreover, a generator conditioned on the sentence/word itself would correspond to how humans use their gestures to emphasize their communication.

References

1. Alibeigi, M., Rabiee, S., Ahmadabadi, M.N.: Inverse kinematics based human mimicking system using skeletal tracking technology. J. Intell. Robotic Syst. **85**(1), 27–45 (2017)
2. Augello, A., Cipolla, E., Infantino, I., Manfrè, A., Pilato, G., Vella, F.: Creative robot dance with variational encoder. CoRR abs/1707.01489 (2017)
3. Beck, A., Yumak, Z., Magnenat-Thalmann, N.: Body movements generation for virtual characters and social robots. In: Judee, K.B., Nadia, M.-T., Maja, P., Alessandro, V. (eds.) Social Signal Processing, pp. 273–286. Cambridge University Press, Cambridge (2017)
4. Breazeal, C.: Designing sociable robots. In: Intelligent Robotics and Autonomous Agents. MIT Press, Cambridge (2004)
5. Cao, Z., Hidalgo, G., Simon, T., Wei, S.E., Sheikh, Y.: OpenPose: realtime multi-person 2D pose estimation using part affinity fields. arXiv preprint arXiv:1812.08008 (2018)
6. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: CVPR (2017)
7. Castillo, E., Gutiérrez, J.M., Hadi, A.S.: Learning Bayesian networks. In: Expert Systems and Probabilistic Network Models. Monographs in computer science. Springer-Verlag, New York (1997). https://doi.org/10.1007/978-1-4612-2270-5_11
8. Everitt, B., Hand, D.: Finite Mixture Distributions. Chapman and Hall, New York (1981)
9. Fadli, H., Machbub, C., Hidayat, E.: Human gesture imitation on NAO humanoid robot using kinect based on inverse kinematics method. In: International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA). IEEE (2015)
10. Goodfellow, I.: NIPS tutorial: generative adversarial networks. ArXiv e-prints, December 2017
11. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
12. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social GAN: socially acceptable trajectories with generative adversarial networks. CoRR abs/1803.10892 (2018). http://arxiv.org/abs/1803.10892
13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

14. Kwon, J., Park, F.C.: Using hidden markov models to generate natural humanoid movement. In: International Conference on Intelligent Robots and Systems (IROS). IEEE/RSJ (2006)
15. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
16. MacCormick, J.: How does the kinect work?. http://pages.cs.wisc.edu/ahmad/kinect.pdf. Accessed 3 June 2019
17. Manfrè, A., Infantino, I., Vella, F., Gaglio, S.: An automatic system for humanoid dance creation. Biologically Inspired Cogn. Architect. **15**, 1–9 (2016)
18. McNeill, D.: Hand and Mind: What Gestures Reveal About Thought. University of Chicago press (1992)
19. Mehta, D., et al.: VNect: real-time 3D human pose estimation with a single RGB camera. ACM Trans. Graph. **36**(4), 44:1–44:14 (2017)
20. Mukherjee, S., Paramkusam, D., Dwivedy, S.K.: Inverse kinematics of a NAO humanoid robot using Kinect to track and imitate human motion. In: International Conference on Robotics, Automation, Control and Embedded Systems (RACE). IEEE (2015)
21. Okamoto, T., Shiratori, T., Kudoh, S., Nakaoka, S., Ikeuchi, K.: Toward a dancing robot with listening capability: keypose-based integration of lower-, middle-, and upper-body motions for varying music tempos. IEEE Trans. Robot. **30**, 771–778 (2014). https://doi.org/10.1109/TRO.2014.2300212
22. Poubel, L.P.: Whole-body online human motion imitation by a humanoid robot using task specification. Master's thesis, Ecole Centrale de Nantes-Warsaw University of Technology (2013)
23. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE. **77**, 257–286 (1989)
24. Rodriguez, I., Astigarraga, A., Ruiz, T., Lazkano, E.: Singing minstrel robots, a means for improving social behaviors. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2902–2907 (2016)
25. Rodriguez, I., Astigarraga, A., Jauregi, E., Ruiz, T., Lazkano, E.: Humanizing NAO robot teleoperation using ROS. In: International Conference on Humanoid Robots (Humanoids) (2014)
26. Rodriguez, I., Martínez-Otzeta, J.M., Irigoien, I., Lazkano, E.: Spontaneous talking gestures using generative adversarial networks. Robot. Auton. Syst. **114**, 57–65 (2019)
27. Schubert, T., Eggensperger, K., Gkogkidis, A., Hutter, F., Ball, T., Burgard, W.: Automatic bone parameter estimation for skeleton tracking in optical motion capture. In: International Conference on Robotics and Automation (ICRA). IEEE (2016)
28. Tanwani, A.K.: Generative models for learning robot manipulation. Ph.D. thesis, École Polytechnique Fédéral de Laussane (EPFL) (2018)
29. Tits, M., Tilmanne, J., Dutoit, T.: Robust and automatic motion-capture data recovery using soft skeleton constraints and model averaging. PLOS One **13**(7), 1–21 (2018)
30. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemometr. Intell. Lab. Syst. **2**(1–3), 37–52 (1987)
31. Zhang, Z., Niu, Y., Yan, Z., Lin, S.: Real-time whole-body imitation by humanoid robots and task-oriented teleoperation using an analytical mapping method and quantitative evaluation. Appl. Sci. **8**(10), 2005 (2018). https://www.mdpi.com/2076-3417/8/10/2005