Watschel - Making a 4 DoF Robot walk

Jonas Lewandrowski[1] and Björn Plonka[2]

***Abstract*— Despite advances in the field of humanoid locomotion, walking is inherently challenging due to the need for stability, dynamic motions, and balancing on one leg during each step. One popular model, the 3D Linear Inverted Pendulum (3D LIP), has been instrumental in simplifying bipedal walking by approximating the robot's dynamics. However, when applying this, the gap between simulations exercising 3D Walking Patterns and real-world steps continues to be a major hurdle. This discrepancy arises from factors such as unmodeled dynamics, approximations, variability in ground friction, and other inaccuracies. This paper details our strategy to adapt a walking pattern based on 3D LIP to obtain a walking gait for Watschel, a small 3D-printed 4 DoF biped.**

I. Introduction

Humanoid robots are expected to mirror human behavior, which includes bipedal walking. To enable natural and efficient walking gaits, they must obtain the ability to perform dynamic motions.

In this paper, we detail the method we employed to obtain a walking gait for our self-built bipedal robot Watschel.

One possibility to approximate the movement of a walking robot is using the *3D Linear Inverted Pendulum* (3D LIP) [1]. It models the entire robot only by its center of mass (CoM) and the feet, which come into contact with the ground (compare the 3D plot in the top right in Fig. 1). Given a set of target steps and additional parameters, the *3D Walking Pattern Generation* which is based on the 3D LIP calculates the optimal foot placement to achieve the most similar steps possible, considering the dynamics of the entire robot.

We will go on to describe how we applied inverse kinematics to calculate the required joint configurations to achieve the given positioning of the feet. Watschel has 4 joints, resulting in 4 internal degrees of freedom and 10 degrees of freedom in total. These joint angles as well as the inverse kinematic variables can be manipulated in a GUI as seen in the top left of Fig. 1. We also demonstrate how we identified keyframes to replicate a motion inspired by the 3D LIP executing the 3D walking pattern. A keyframe is defined by the joint configuration, that is to be achieved at a given time. These keyframes play a vital role in our method, as we use ROS to communicate between the robot and the user interface. However, given that the timing of communication varies, it is not viable to send the current joint states, as the motion is highly dynamic and milliseconds determine if the the robot walks or falls.

[1] jonas.lewandrowski@student.kit.edu
[2] bjoern.plonka@student.kit.edu
Jonas Lewandrowski and Björn Plonka are with Karlsruhe Institute of Technology

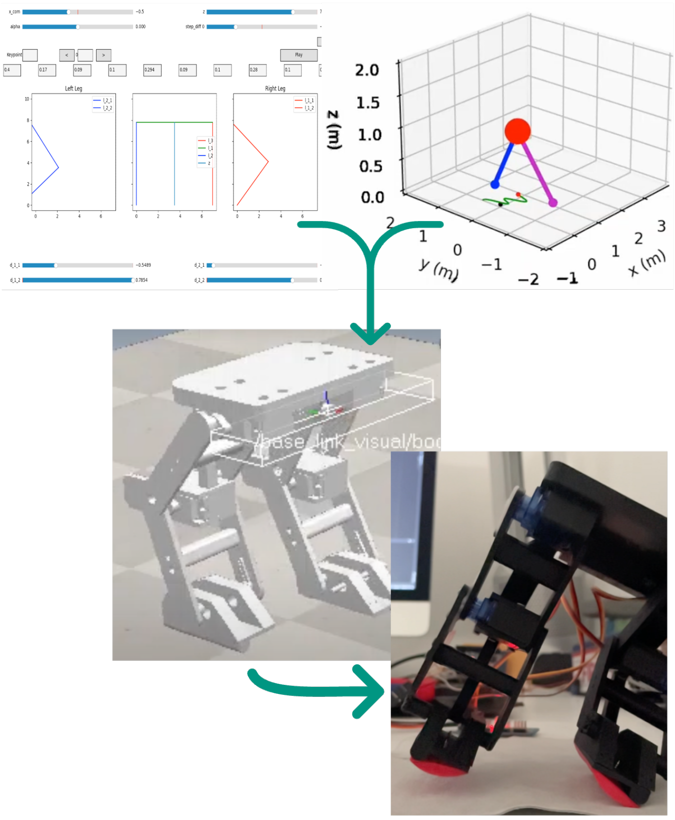

Fig. 1. Using Inverse Kinematics and the 3D Linear Inverted Pendulum allows us to preview the behavior of Watschel in the simulation before we execute the motion on the real robot.

Additionally, we build a full URDF model of Watschel (see Fig. 10) and a MuJoCo simulation. For this, we demonstrate how to find values for mass, inertia, and joint behavior without the use of expensive testing rigs. From this, we build a simulation environment in MuJoCo and explore control via ROS to control Twatschel, the digital twin of Watschel (see the middle image in Fig. 1). Using a ROS node allowed us to transmit the keyframes to the simulation as well as the real robot using the same software. Having a digital copy of the real robot permits us to test the correctness of the approach without depending on the real robot.

Moreover, we present how our approach deals with the simulation-to-reality (sim-to-real) gap. It exists as a result of several reasons, mostly modeling assumptions. Having optimized each keyframe digitally, we adapt the time of each keyframe on the real robot, so that Watschel can take steps (see the bottom right image in Fig. 1).

We furthermore show the influence of different foot designs. Our first designs were made out of hard plastic (PLA) and walked directly on the table. As Watschel suffered from a lot of slip and instability, we increased the friction by using a more flexible plastic (TPU) and walking on a softer surface.

Finally, we evaluate the effectiveness of our approach by comparing the trajectory of the center of mass (CoM) of Twatschel in the simulation and the initial walking pattern with the 3D LIP.

II. Approach

Our approach is split up into several topics. We start in II-A by reviewing the choices that influenced the design and components of Watschel. We continue in II-B by explaining the kinematics of Watschel and detail in II-C the theoretical model of a walking pattern. In II-D and II-E, we go into making Twaschel and it having its first walk, while explaining how to get Watschel to take steps in II-F. We end this chapter in II-G with our study on foot designs.

A. Design of the robot

Our design is conceptually similar to a robot proposed in the article 'A Non-Anthropomorphic Bipedal Walking Robot with a Vertically Stabilized Base' ([2]) (see: 2).

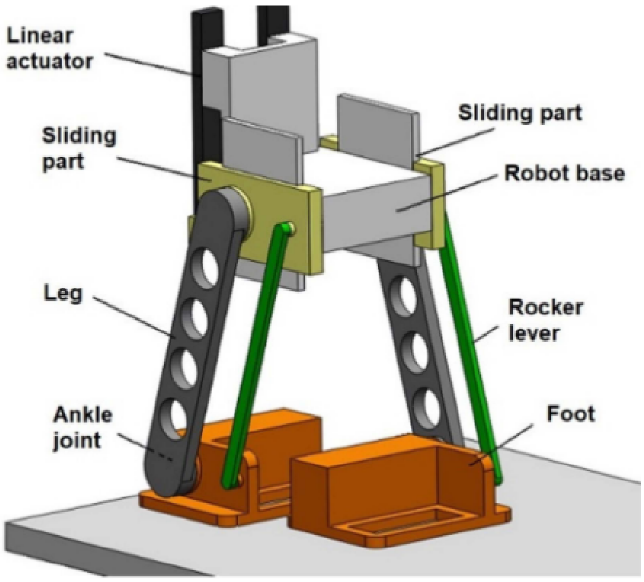

Fig. 2. Figure 1 from [2] - Design of a statically stable bipedal robot

However, instead of static stability, we use a design that is more dynamic and therefore better suited for "lifelike" movements. Through a rocking motion and a convex foot surface, the center of mass is shifted in such a way, that the airborne leg of the robot can be moved freely within the period of dynamic stability.

Furthermore, the design of the robot is heavily influenced by walking robots, which were first shown in [3]. The design of these bird-like robots was conceptually adapted by [4]. The main reasons we chose to adapt this design include the minimal amount of servos needed for the robot to walk, as well as the simple geometries, reducing the complexity of the system. Only having 4 actuated joints, results in only having to purchase 4 servos. While each of the servos used in [4] costs upwards of 400€, scaling the model down allowed us to reduce costs significantly. Choosing the servos is a complex task: A choice between maximum angular velocity and force as well as weight has to be made. Given our very light robot, we chose light and fast 9g servos that could move at sufficient speeds while being able to lift the robot.

The model contains several features, which allow for an easier walking gait. One of these is the parallel leg segments, ensuring that the motor in the knee as well as the foot always stays parallel to the ground. This is well-suited for walking on even surfaces like a tabletop, as we do not need a third actuated joint to control the angle of the ankle relative to the ground. This reduces weight in the foot as well as complexity for the entire system. A different geometric feature, enabling Watschel to take steps not only in the Simulation, is the design of the feet. While requiring multiple iterations of redesign during real-life testing, the main concept is shared between all designs: The contact between the ground and the foot is a line. This has the advantage that we do only have to worry very little about the robot falling to the front or the back, even during the steps. Compared to the square contact area, as seen in Fig. 2, this design does not hinder the rotation to the sides in order to manipulate the y-coordinate of the CoM. Additionally, a line allows the foot to have more friction and thus makes the steps more stable, compared to having two point-shaped contacts that could span the same contact area.

B. Inverse Kinematics

In this chapter, we will go into the details of Watschel's kinematics.

The kinematics of Watschel consist of 4 internal degrees of freedom (IDoF) and 10 degrees of freedom (DoF) in total. However, given the geometry of the feet and the way that the legs are constructed, while at least one foot touches the flat ground, the rotational axis around the y and z-axis is restricted. This results in 5 actual DoF when 1 foot is in contact with a flat floor and 3 when both are in contact. We define every variable describing a part of the robot on the right with a one as the first index and all on the left with a two. Furthermore, we index similar variables from top to bottom. Thus l_{11} is the length of the length of the upper link on the right side, while l_{22} stores the information about the length of the lower link on the left side. We furthermore define δ_{ij} for $i, j \in \{1, 2\}$ to represent the angle of the joint as seen in Fig. 3.

In order to calculate all the joint angles we need a fully defined system of equations. We therefore denote the lean angle as α, the height of the middle of the hip as hip_z. Finally, we define steppDiff_i with $i \in \{1, 2\}$ to represent the distance on the x-axis between the left and right foot and the base respectively (cf Fig. 4).

Using trigonometry, we can define the length of the right and left leg as

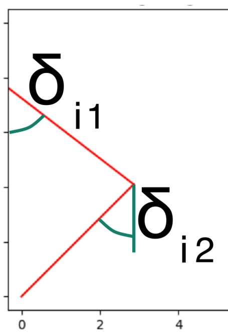

Fig. 3. We define the joint angles δ_{ij} for $i, j \in \{1, 2\}$ for each of the two feet, such that the higher angle has index 1 and the lower one index 2 as well as the right leg as the first one and the left leg as the second.

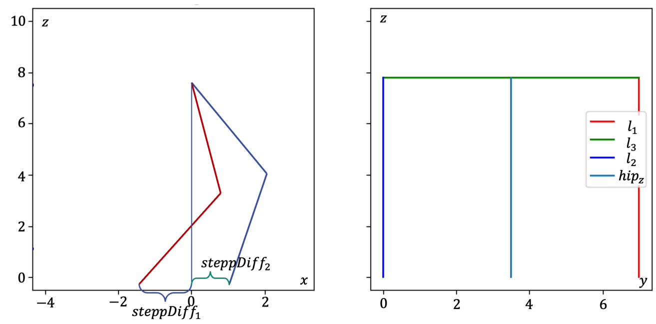

Fig. 4. Definition of the variables to create a fully defined system of equations for the inverse kinematics.

$$l_i = l_{i1}\cos(\delta_{i1}) + l_{i2}\cos(\delta_{i2}) \tag{1}$$

with $i \in \{1, 2\}$ respectively, where the length is defined as the distance between the foot and the corresponding end of the hip. Defining the length of the base as l_3, we can now set the entire system of equations for the forward's kinematics up, where the given joint angles δ_{11} to δ_{22} allow us to calculate the state variables as follows:

$$\text{steppDiff}_1 = l_{11}\sin(\delta_{11}) + l_{12}\sin(\delta_{12}) \tag{2}$$
$$\text{steppDiff}_2 = l_{21}\sin(\delta_{21}) + l_{22}\sin(\delta_{22}) \tag{3}$$
$$\alpha = \arctan\left(\frac{l_2 - l_1}{l_3}\right) \tag{4}$$
$$\text{CoM}_z = l_1\cos(\alpha) + \frac{l_3}{2}\sin(\alpha). \tag{5}$$

However, as we are given the x and y values of each of our steps, we are now interested in which joint angles allow us to fulfill the requirements. It should also be noted, that the current design of the robot only allows for finding joint configurations, that fulfill the required x value, as the robot does not have a joint to adapt the y coordinate of the foot.

Therefore we start by rewriting 5 as

$$l_1 = \frac{\text{CoM}_z - \frac{l_3}{2}\sin(\alpha)}{\cos(\alpha)}. \tag{6}$$

We can now insert 1 into 6 as follows:

$$\begin{aligned} & l_{11}\cos(\delta_{11}) + l_{12}\cos(\delta_{12}) = \frac{\text{CoM}_z - \frac{l_3}{2}\sin(\alpha)}{\cos(\alpha)} \\ \Leftrightarrow\ & l_{12}\cos(\delta_{12}) = \frac{\text{CoM}_z - \frac{l_3}{2}\sin(\alpha)}{\cos(\alpha)} - l_{11}\cos(\delta_{11}) \\ \Leftrightarrow\ & \delta_{12} = \arccos\left(\frac{\frac{\text{CoM}_z - \frac{l_3}{2}\sin(\alpha)}{\cos(\alpha)} - l_{11}\cos(\delta_{11})}{l_{12}}\right). \end{aligned} \tag{7}$$

Having a formula for δ_{12}, which only depends on δ_{11} as well as the known parameters, we can now rewrite 2 as

$$\begin{aligned} & \text{steppDiff}_1 = \\ & \quad l_{11}\sin(\delta_{11}) + \\ & \quad l_{12}\sin\left(\arccos\left(\frac{\frac{\text{CoM}_z - \frac{l_3}{2}\sin(\alpha)}{\cos(\alpha)} - l_{11}\cos(\delta_{11})}{l_{12}}\right)\right), \end{aligned} \tag{8}$$

resulting in a non-linear equation, only depending on δ_{11} as a variable. Knowing the boundaries of each joint, which are 0 and $\frac{\pi}{4}$, we can now search in this one-dimensional interval for the optimal solution for δ_{11}.

The same approach can be taken for the left side as well. Therefore we assume, that δ_{11} is already found and thus δ_{12} is also calculated, which in return means, that l_1 is also a known parameter now. We can then start by rewriting 4 as

$$l_2 = l_3\tan(\alpha) + l_1. \tag{9}$$

Inserting 1 into 9 results in

$$\begin{aligned} & l_{21}\cos(\delta_{21}) + l_{22}\cos(\delta_{22}) = l_3\tan(\alpha) + l_1 \\ \Leftrightarrow & l_{22}\cos(\delta_{22}) = l_3\tan(\alpha) + l_1 - l_{21}\cos(\delta_{21}) \\ \Leftrightarrow & \delta_{22} = \arccos\left(\frac{l_3\tan(\alpha) + l_1 - l_{21}\cos(\delta_{21})}{l_{22}}\right), \end{aligned} \tag{10}$$

where δ_{22} is now only depending on δ_{12} as well as known parameters. Replacing δ_{22} in 3 results in

$$\begin{aligned} & \text{steppDiff}_2 = \\ & \quad l_{21}\sin(\delta_{21}) + \\ & \quad l_{22}\sin\left(\arccos\left(\frac{l_3\tan(\alpha) + l_1 - l_{21}\cos(\delta_{21})}{l_{22}}\right)\right), \end{aligned} \tag{11}$$

where δ_{21} is the only variable. A search in the interval $[0, \frac{\pi}{4}]$ results in an optimal constraint value for δ_{21}. This can be used in return in 10 so that all values are found. The approximation by searching for the variables in the interval by using a grid search does not play a major factor in the accuracy of the algorithm as the servos are using a PWM Signal, only taking integer values in a range of 500 steps. Using a grid with a finer resolution will not improve the accuracy of the real robot while taking more time to compute. The big advantage of this approach is not relying on an optimizer, which might get stuck in a local optimum, and rather certainly arriving at the global optimum.

C. 3D LIP and 3D Walking Pattern Generation

In this section, we will recap the theory behind the theoretical approach to finding optimal foot placements as detailed in [1]. The model assumptions are as follows:

- the robot is modeled by the CoM
- a massless leg connects the CoM and the supporting point.
- the pendulum can rotate freely about the supporting point
- a leg can change its length using force
- the CoM always remains at a constant height

When comparing these modeling assumptions, we can see several deviations between Watschel and the model. While the representation of the robot by its CoM works quite well, it can not freely rotate about the supporting point mainly because it is supported by a line, restricting 2 of the 3 rotational axes. We add the rotation around the y-axis back by manipulating the distance between the foot and the hip along the x-axis. Watschel can change the length of a leg, however, given the limitations of Watschel's joints, we were unable to force the CoM to stay at a constant height while also taking a step. This model can now be used to determine a 3D Walking Pattern. Therefore, a constant step time is assumed, after which a switch of the supporting foot occurs. Fig. 4.25 in [1] explains how optimal foot placements can be found to minimize the distance between the target step and the modeled step, taking the dynamics of the moving CoM into account. It works by concatenating individual walk primitives which describe each individual step. The terminal position of the CoM determines each primitive due to the symmetric nature of its trajectory.

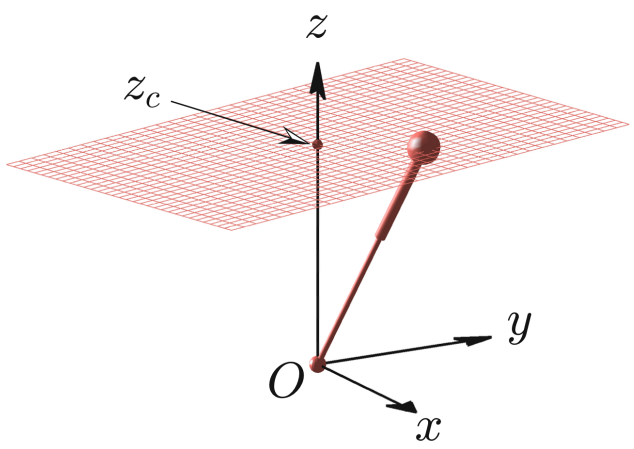

Fig. 5. The linear inverted pendulum as the CoM constrained to a specific height z_c, connected to the supporting point via a length-changing, massless leg

For Watschel, we set the height of the CoM to be at 1 unit, resulting in the following corresponding step length:

n	1	2	3	4	5
$s_x^{(n)}$	0.0	0.25	0.25	0.25	0
$s_y^{(n)}$	1.25	1.25	1.25	1.25	1.25

Additionally, we assume the initial position of the CoM at $(0.0, 0.0, 1)$ with an initial velocity of 0.04 along the x axis and 1.33 along the y axis. With $(0.0, -0.625, 0)$ and $(0.0, 0.625, 0)$ being the initial position of the left and right foot respectively, and the hyperparameters $a = 1$, $b = 1$, we can calculate the corresponding optimal foot placements. While the hyperparameters resulted from trial and error testing, the initial values mimic the positions of Twaschel after lifting the first foot off the ground. These parameters lead to the following positions of the feet:

n	1	2	3	4	5
$s_x^{(n)*}$	0	0.242	0.265	0.25	-0.02
$s_y^{(n)*}$	1.25	1.27	1.26	1.25	1.25

The resulting trajectory of the ground projection of the center of mass (gCoM) can be seen in Fig. 6.

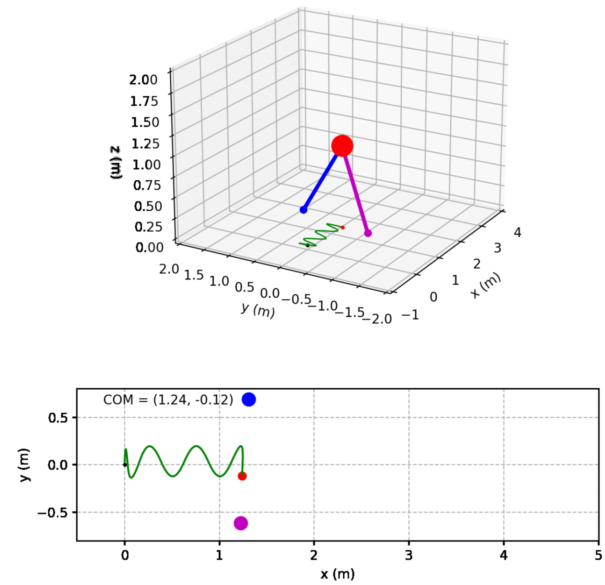

Fig. 6. Given a set of starting conditions, we can compute the optimal position of the feet and the resulting trajectory of the ground projection of the CoM (see the green line)

D. Simulation

Twatschel was realized in CoppeliaSim [5], a proprietary simulation environment that implements several physics engines, including MuJoCo. We chose MuJoCo for our implementation because of its strong support for dependent joints (also known as mimic joints). The real-world kinematics are modeled as two active and six mimic joints for the leg structure (four of them are for the support structure since physics engines generally only support loop-free kinematic chain topologies) (see Fig. 7).

The goal of the simulation is to validate ideas for walking gaits and mechanical alterations as well as enable further introspection into the physics of Watschel (see chapter III).

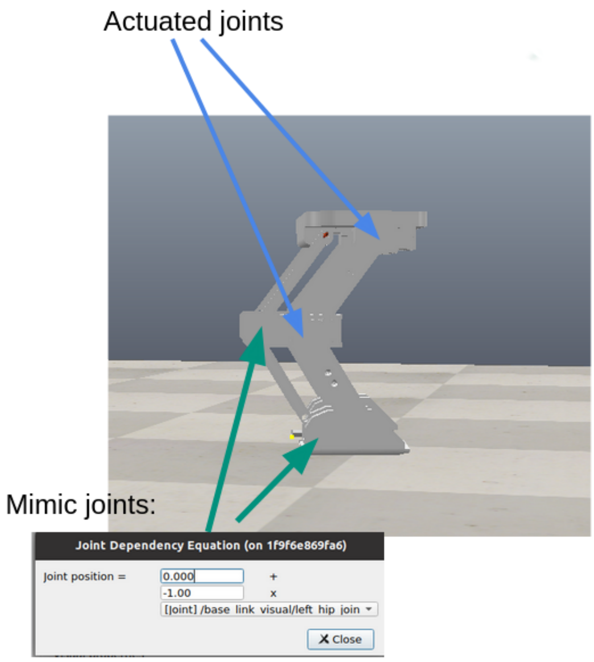

Fig. 7. The actuated and dependent joint of the model

1) Obtaining values: The size of the segments as well as their relative positions were obtained from CAD software. For their mass, we weighed the real-world pieces. Using the stl library from PyPI we derived the ixx, iyy, and izz inertia values to construct a simplified inertia tensor. From this we created a URDF [6] file (See fig 10).

E. Finding keyframes

We decided to define our walking gait through keyframes, which we constructed utilizing inverse kinematics (See II-B). As shown in Fig. 4, the robot is fully definable in four variables: The height of the hip, the sideways tilt of the hip, and the difference between the two feet to hip respectively. With this, we define a set of 8 keyframes:

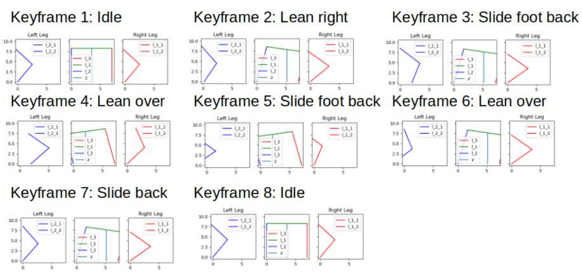

Fig. 8. Keyframes that compose the walking gait of watschel

Watschel begins and ends in a static idle position. It should be noted that all keyframes are statically stable, but

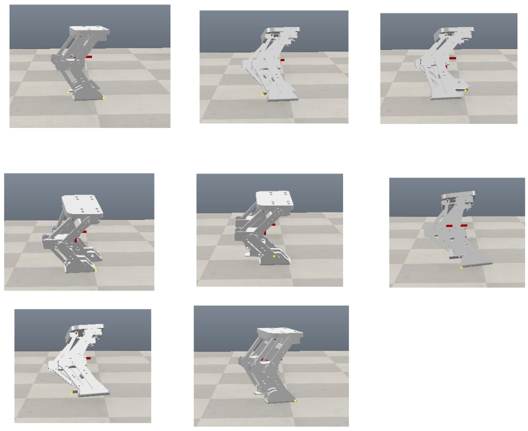

Fig. 9. Walking gait found experimentally for watschel

would not facilitate locomotion without the proper timing. Locomotion is possible by the following process: The robot tips from side to side, such that the gCoM does not cross the contact line of the supporting foot over the entire period of the walking gait. As the gCoM stays on one side of the contact line, it exerts a torque that reverses the tipping motion. Because of this, it will inevitably return to a static pose, where both feet touch the ground. However, the inertia exerted by the tipping motion is enough to temporarily lift the opposing leg. During this time, the opposing leg is pulled back. When it touches the ground again, the center of mass will have shifted backward, and the robot will have moved. The friction of the planted foot absorbs the rotational torque created by the pulling back of the opposing foot. The robot then tips to the other side and repeats the behavior. After this, the walking gait can be repeated. Alternatively, the robot can return to its statically stable idle position. The key problem is finding the correct timing. We were able to create working walking gaits both in the simulation and on the real robot with these keyframes. The sim-to-real-gap (See II-E.1) necessitates different timings. For the timings used in the real world, we performed extensive trial and error (See: II-F

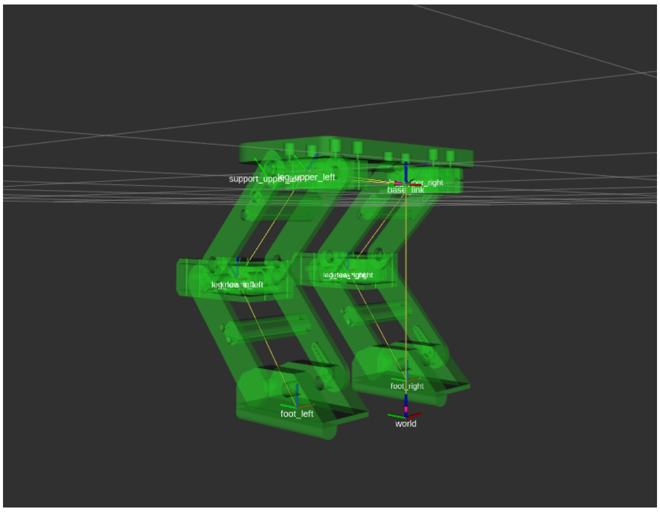

Fig. 10. The URDF model visualized in Rviz

For numerical stability, all units were multiplied by 100. For example, for the main hip segment, we used a mass of 17.2 kg instead of 172 grams. For this report, all values are therefore scaled down 100 times from what the simulation is using. Since the dependent joints in the upper and lower legs meant that joint limits in the lower segments were dependent on the position of the upper segments, we decided to choose conservative joint limits of 0 to 45 degrees. For the effort and force of the servo motors, we were not able to find satisfactory data. Generally, modeling electric motors has been a challenge in many simulation implementations. For example the paper "Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning" [7] proposes training an LSTM to replicate the behaviour of the used actuators. Instead, we decided to find the exact values experimentally. For the joints, we found 0.2Nm of peak force and a maximum speed of 50 degrees per second to replicate the analog servos best.

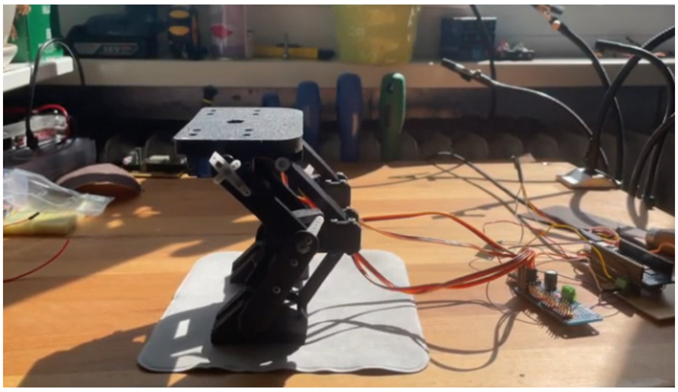

Fig. 11. Watschel being pushed back by cables

1) Sim-to-real-gap: Because of our limited resources, we were not able to extensively verify the values of our model. Especially the inertia and the performance of the servo motors would have required expensive testing rigs to properly model. As a consequence of this, walking gaits that work on the real robot do not work on the simulated model and vice versa. Further sources of discrepancies are for example the servo cables as seen in Fig. 11, which exert a small but noticeable force on the robot, as well as non-adjustable ground friction forces.

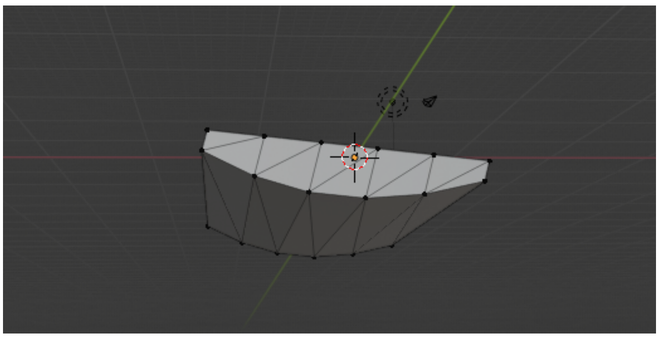

Fig. 12. The collider model used for the feet of Watschel

For reasons of stability and performance, we also decided to model the feet of Watschel with 10 faces as ground contacts as opposed to a continuous curved surface in real life (see Fig. 12).

In summation, these deviations amount to a behavior that is, while similar, not identical to the real-world Watschel. These differences mean that a different timing from the one used for the real-world robot was necessary.

F. Working Principle in the real world

Together with the defined keyframes, we need to transmit the data to the real robot. Given the complexity of the inverse kinematic and the additional controllability, we chose to use ROS. It enables us to easily communicate between a controlling PC and the real robot as well as the simulated one. However, it comes with the inconvenience, that it is not real-time capable. This means that we can not send a keyframe at the required time and be sure, that the robot will move into the correct pose at the right time. To overcome this obstacle, we transmit a message containing all the keypoints, including the times. Once the robot receives the message, it can use linear interpolation to calculate the corresponding joint states at each time.

Despite the sim-to-real gap, we found out that the joint states at each keyframe can be transferred from the Simulation to Watschel. Thus it was only a question of adapting the times of each keyframe. This was a time-consuming part as the difference of a millisecond determined whether Watschel took a step successfully or fell over. This further shows why transmitting the keyframes at the required time was not feasible.

G. Foot design

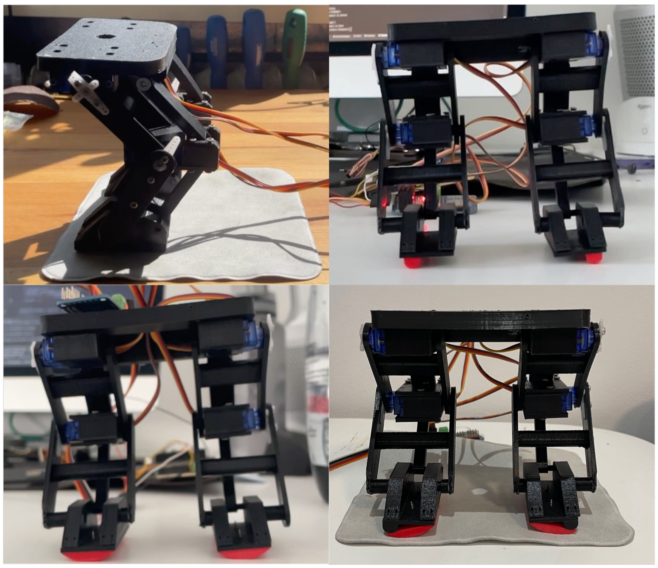

Fig. 13. The evolution of the foot design.

Once we had the pipeline to test the keyframes on the real robot it became obvious that our initial foot design was not suited for the task. While the robot had enough force to push itself over the foot, when trying to perform the first lift-off motion, the robot suffered from a lot of slip at the theoretically stable foot. Our first solution was to place a soft mat between the table and the robot, which helped in reducing the vibrations as well as adding some friction and thus stability (see the top left image in Fig. 13). However, it was apparent, that the friction was still not optimal. We continued using a new more flexible 3D-printable plastic: TPU (the red feet in Fig. 13). The advantages of this new material are identical to the mat, except not being quite as good at reducing vibrations. Our first idea was to replicate the same design of the feet but now in TPU (see top left image in Fig. 13). While it improved the stability, the structure of these feet made it very hard to get the ground projection of the center of mass (gCoM) close to the line of support without overshooting. We continued to test a different design of the feet, which were symmetrical to the foot (see the bottom left image in Fig. 13). While this design improved the friction even more, it worsened the stability. As the gCoM approaches the line of support, the robot is pulled towards it, ensuring falling over. Finally, we tested combining all individual aspects of the prior designs: We combined the rigid structure of the initial feet with the symmetric TPU feet, which walked on the soft mat (see the bottom right image in Fig. 13). The resulting unsymmetrical feet added in slowing the robot down, as the gCoM approached the line of support, allowing for more possible acceleration and thus bigger steps. The added acceleration was also enabled by the combination of the TPU on the soft mat, which provided enough friction to not slip.

III. Experiments and Evaluation

We seek to analyze the performance of our robot through a set of experiments aimed at qualitatively testing our walking gait.

A. Comparing the 3D Walking Pattern with Twatschel

In the following section, we will evaluate the steps performed by Twatschel by comparing the executed trajectory of the gCoM to the optimal trajectory as given by the 3D Walking Pattern [1].

Fig. 14 shows in blue the trajectory of the ground projection of the center of Mass (gCoM) of Twatschel while performing the steps. The trajectory of the gCoM of the 3D LIP in combination with the optimal step positions as given by the 3D Walking Pattern Generator is shown in red. It is easy to see that Twaschel's overall motion follows the theoretical trajectory as the simulation peaks overlap the theory's maxima. However, given the more realistic movements of Twatschel, we encounter a lot of vibrations, when the feet hit the ground. This manifests itself in the jitter around the peaks, when the supporting feet switch. This behavior can also be observed in the video of Watschel walking (cf. Video of Watschel).

It is also very interesting to compare the isolated trajectory of the gCoM along the y-axis between the simulation of Twatschel and the 3D LIP. While the combined trajectory

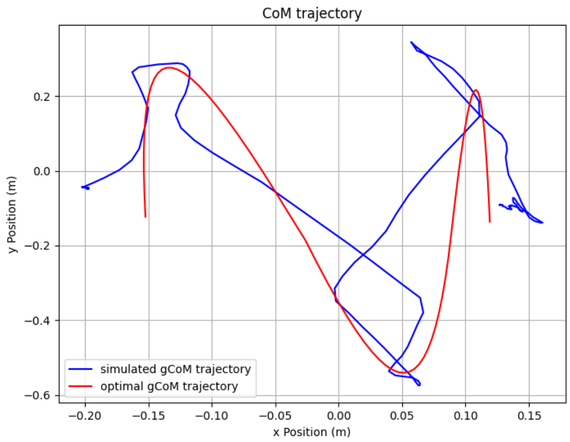

Fig. 14. Comparing the target trajectory of the ground projection of the center of mass (gCoM) with the trajectory of the gCoM as observed in the simulation, following the found keypoints.

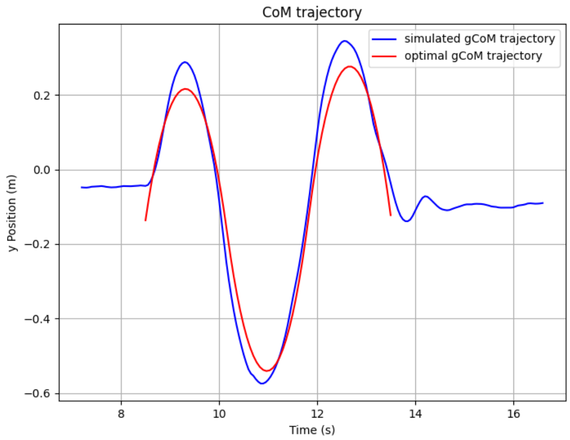

Fig. 15. The trajectory of the gCoM along the y-axis

shows a lot of deviation due to vibration, the isolated y-part of the trajectory matches the optimal trajectory nearly perfectly. It deviates only in the fact, that Twatschel has slightly larger amplitudes and a very minor longer period for its steps. This was however by design, as we wanted to give Watschel the best chance at taking real steps, which included larger amplitudes, as they allow for more ground clearance for the moving foot. With the larger amplitudes, a longer period cycle was emergent. This means, however, that the big deviations between the optimum and the adaptation lie in the movement along the x-axis.

We can also see that the movement of the center of mass along the z-axis (See Fig. 16) is rather stable, which is consistent with the expected behavior of the 3D walking pattern.

B. Determining the walking speed

Looking at the x trajcetory of the gCoM over time, we can deduce a theoretical maximum walking speed from a single

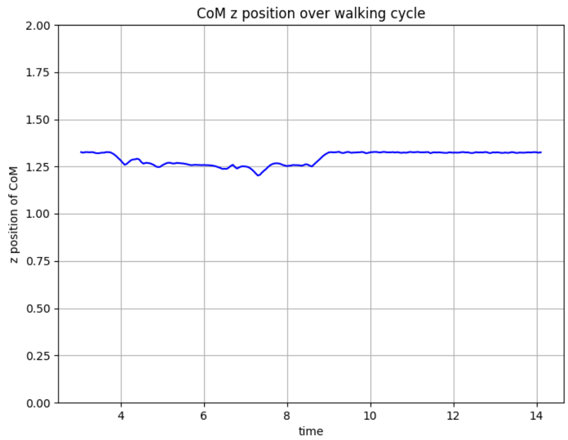

Fig. 16. The z position of gCoM

walk cycle.

We can see in 14 that we travel a total of 0.35 meters - Though it should be noted that the simulation is scaled up by a factor of 100. This gives us a theoretical distance for the episode of 3.5 centimeters in real life. Combined with the episode length of 5 seconds (8.5 to 13.5 seconds as seen in Fig. 15) this gives us a theoretical walk speed of 0.7cm per second. When Watschel is taking the same steps, we can measure a travel distance of roughly 3cm (see Fig. 17).

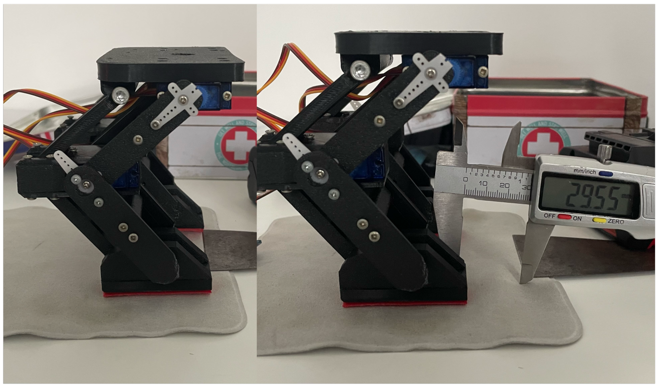

Fig. 17. Measuring the length of the steps as performed by Watschel

IV. Conclusions and Outlook

There are certainly many ways to improve upon our original designs and software stack. However, we feel that our goal of proving that bipedal locomotion with only 4 degrees of freedom, however limited, is possible. Further work should focus on improving the controllability. In theory, full control of the walk should be possible, including slow turning and therefore navigation. However, to achieve this, we would first need to design an energy delivery system, that would be fully contained on the robot, so that it would not be so influenced by the cables. Furthermore, while the softer TPU and the mat minimized vibration when the feet touched back on the ground, it would be interesting to test thicker and less dense feet that could improve upon vibration and thus stability even when not walking on the mat.

A further consideration we made for future improvements, is the use of a gyroscope. This sensor should enable the microcontroller to adapt the timing of the keyframes to take more stable steps by recovering from minor deviations. Additionally, this would minimize the sim-to-real gap, as the simulation would know the current orientation and the relative position of the real robot.

References

[1] S. Kajita, H. Hirukawa, K. Harada, K. Yokoi, Introduction to Humanoid Robotics, Springer Publishing Company, Incorporated, 2014.

[2] I. Virgala, L. Mikova, T. Kelemenova, M. Varga, R. Rákay, M. Vagaš, J. Semjon, R. Janosc, M. Sukop, P. Marcinko, P. Tuleja, A non-anthropomorphic bipedal walking robot with a vertically stabilized base, Applied Sciences 12 (9). `doi:10.3390/app12094108`.
URL `https://www.mdpi.com/2076-3417/12/9/4108`

[3] W. D. Imagineering, Droids in training: Imagineers conduct playtest at star wars: Galaxy's edge.
URL `https://www.youtube.com/watch?v=qNfRgZMWyCI&t=7s`

[4] J. Bruton. [link].
URL `https://youtu.be/XHjnWm7hGH8`

[5] E. Rohmer, S. P. N. Singh, M. Freese, V-rep: A versatile and scalable robot simulation framework, in: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 1321–1326. `doi:10.1109/IROS.2013.6696520`.

[6] D. Tola, P. Corke, Understanding urdf: A dataset and analysis (2023). `arXiv:2308.00514`.
URL `https://arxiv.org/abs/2308.00514`

[7] N. Rudin, D. Hoeller, P. Reist, M. Hutter, Learning to walk in minutes using massively parallel deep reinforcement learning, CoRR abs/2109.11978. `arXiv:2109.11978`.
URL `https://arxiv.org/abs/2109.11978`