Continuous Stairs Climbing Control Design for Bipedal Robots

An Undergraduate Thesis

Presented to The Department of Mechanical Engineering in Partial Fulfillment of the Requirements for Graduation with Distinction in Mechanical Engineering

By

Boyuan Wang

The Ohio State University

April 2024

Thesis Committee

Dr. Ayonga Hereid, Advisor

Dr. Haijun Su, Committee Member

Abstract.

While robotics technology advances and more real-world applications are used in the industry, legged robots, unlike robotic arms or wheeled robots, are not widely utilized despite their various benefits, such as improved movement performance in complex terrains. This could be due to several limitations in both hardware and software. This paper focuses on developing a reliable and efficient control algorithm for bipedal robots walking on long, continuous stairs. This enhancement aims to increase their movability in real-world scenarios, fully utilizing their advantages of better operation and relatively less space requirement in complex terrain compared to traditional wheeled robots. To enable bipedal robots to walk on continuous stairs, the paper proposes improvements to the traditional Angular Momentum Linear Inverted Pendulum (ALIP) planner. These improvements enable the 2D planar five-link walker robot, RABBIT, to handle vertical and horizontal displacement of the center of mass (COM) and to calculate foot displacement as a trajectory in the simulator. The findings suggest that the modified traditional dynamical model for bipedal robots requires fewer resources to generate control algorithms. As the working environment, stairs, a reparative environment, this offers an advantage with utilizing Bezier approximation to regulate the swing-foot and COM displacement trajectory onto the bipedal model itself. This work highlights the advantage from model-based approach that it takes less time and inputs to generate the control algorithms and the transparency during the simulation comparing to the black box-like process of the learning-based approaches, but with an inherent limitation of lacking versatility. And for LIP model specifically, without introducing other policies or methodologies, it is insufficient to handle the vertical displacement of the COM for bipedal robots.

Acknowledgements

The research was completed under the supervision of Dr. Ayonga Hereid. I am grateful for the opportunity to conduct undergraduate research and utilize the facilities at the Cyberbotics Lab. The key inspirations and solutions to challenges provided by Dr. Hereid at every stage of my project were invaluable.

I also wish to extend my gratitude to Zhaoyang Xiang for his continuous support and dedication throughout the completion of this project, complementing Dr. Hereid's guidance.

Special thanks go to Victor C. Paredes and Min Dai, whose work served as an inspiration.

Finally, I must express my deepest appreciation to my parents. Their unwavering support in my pursuit of an engineering dream has been the bedrock of my journey.

Table of Contents

List of Figures

List of Table

Chapter 1. Introduction and literature review

1.1 Induction on Control Algorithms for Bipedal Robots

Bipedal robots have many applications in various fields, such as industrial manufacturing and disaster rescue and management. Two primary methodologies are used for planning foot placement in bipedal robots: model-based approaches, like the Linear Inverted Pendulum (LIP) model and various branches based on it, such as Spring-Loaded Inverted Pendulum (S-LIP) or Hybrid Linear Inverted Pendulum (H-LIP) [1-7], and learning-based methods that use reinforcement learning [8-9]. The LIP model, as shown in Figure 1, includes a massless rod connecting the center of mass (COM) of the robot and the contact point on the ground. Its equation of motion is linear so then the simulation can be simplified based on that feature. Researchers often generalize robots' full-order dynamics into a 2D or 3D workspace to reduce complexity during the simulation process. The S-LIP model can represent complex biological locomotion dynamics like hopping and running, by assuming that the robot's legs are spring-loaded [3]. The LIP model will be used in this project, and it was used to represent the COM dynamics of the robot [2].

Speaking of the LIP model, its dynamic model with no external input torques can be described as the following equations [10]:

$$\ddot{y} = \frac{g}{z_c} y \tag{1}$$

$$\ddot{x} = \frac{g}{z_c} x \tag{2}$$

In this case z_c is a constant, which stands for the vertical distance from the stance foot to the COM of the robot; x and y are the distance of swing in the stance foot frame.

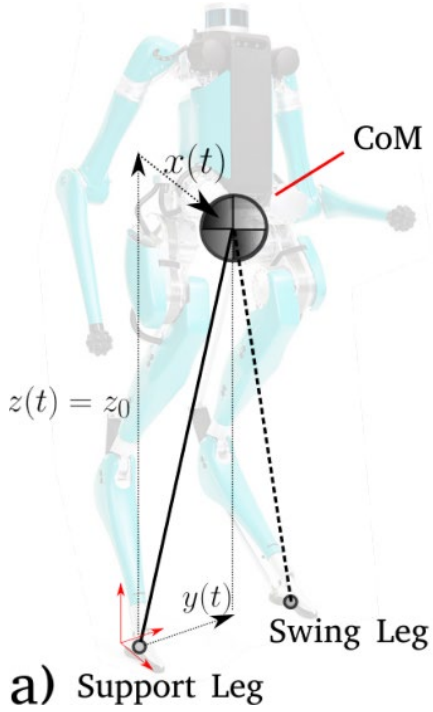

Figure 1: Diagram of a typical LIP model [2].

The COM plays a crucial role in positioning the robot within its environment, whereas the planning of foot placement necessitates a dedicated planner and controller to outline the swing foot trajectories for stable movement. Traditionally, the tracking of a robot's movement has predominantly relied on the linear velocity of the COM. However, there is evidence suggesting that focusing on the angular momentum around the contact point or the robot's torso could provide superior performance compared to solely depending on the COM's linear velocity, especially in navigating challenging terrains [2][7]. The integration of techniques such as Model Predictive Control (MPC) [5][6] and orbital energy [1][2] can further enhance these outcomes. This underscores the benefits of adopting a comprehensive approach to robot locomotion,

leveraging both linear and angular momentum considerations for improved stability and adaptability.

1.2. Literature Review

One key feature assessed by Victor in his work is that when walking on flat ground, the bipedal robot's COM would be at a constant height, the velocity of the COM would show a periodic gait pattern for walking in both forward and lateral directions given the step duration and desired velocity. The pattern and its boundaries can be described as Orbital Lines [2]. As linear dynamics, the states of the robot at any time, t, can be obtained with equation 1 given t and the initial state.

$$x(t) = \begin{bmatrix} cosh(\lambda t) & \frac{1}{\lambda} sinh(\lambda t) \\ \lambda\, sinh(\lambda t) & cosh(\lambda t) \end{bmatrix} x_0 \tag{3}$$

Note that $x(t) \coloneqq \left(\left(p_x(t), \dot{p_x}(t)\right)\right)$, x_0 stands for the initial state.

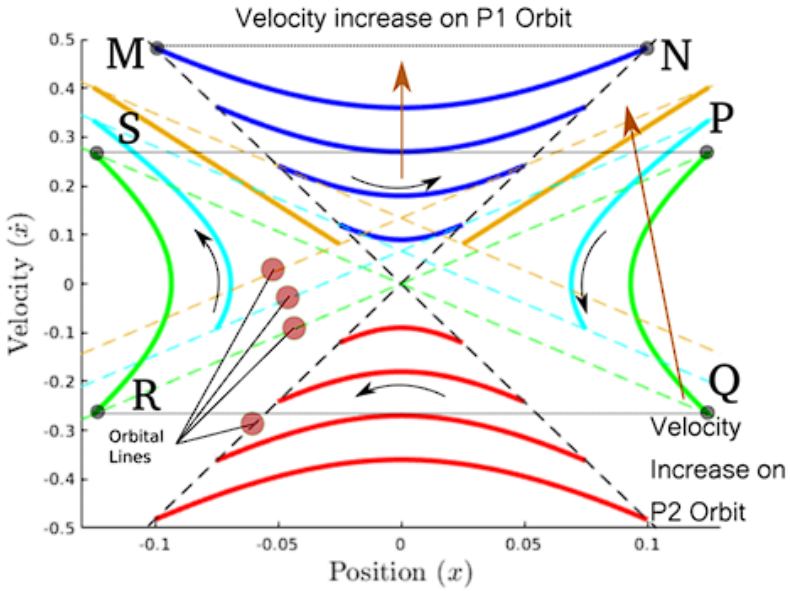

Figure 2: Example of orbital lines, P1 orbits stands for the forward velocity case [2]

According to the orbital lines, P1 for forward walking in this case, the velocity of COM at any point on the line can be defined by:

$$v_x = \pm\lambda \coth\left(\frac{T\lambda}{2}\right) p_x \tag{4}$$

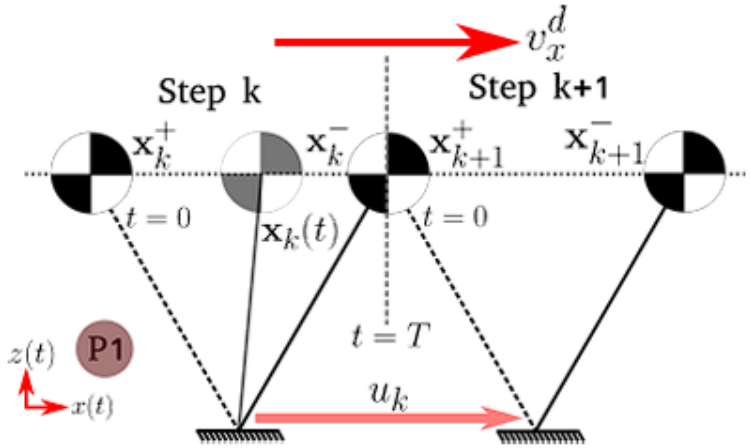

Figure 3: Gait example for walking forward, generated based on P1 orbit [2]

Ideally when walking steadily on flat ground, the initial velocity of a step should be the same as it at the end of this step, velocity at any point during the step can be calculated by the previous equation. With the desired velocity at the end of next step and the velocity at the end of the current step, the length of the next step can be executed by the LIP planner and delivered to the lower-level controller to perform the following steps. This is the mechanism named one-step ahead controller, which equation is derived from equation 2.

Since the LIP is an underactuated model, errors accumulated at each step will eventually collapse the system. The introduction of neural adaptation and QP-based inverse kinematics to the controller successfully produces a stable algorithm, enabling the robot to walk stably on flat surfaces by learning the residual dynamics. This framework also allows for high-efficiency real-time simulations by achieving fast computation of joint references.

With another approach, Min Dai's work, which uses the strategic regulation of angular momentum at the discrete impacts of walking, achieved through detailed control over the pre-impact vertical velocity of the COM. Employing the underactuated LIP model, this paper delineates a methodological framework that simplifies the complex dynamics of bipedal locomotion, setting the stage for an online optimization process fused with closed-form polynomials to dynamically generate and adjust walking patterns.

The key parameter that was monitored was the angular momentum about the stance foot of the LIP model, L_y, and Orbital energy, E [1].

$$\begin{bmatrix} \dot{x}_{com} \\ \dot{L}_y \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{z_{com}} \\ g & 0 \end{bmatrix} \begin{bmatrix} x_{com} \\ L_y \end{bmatrix} \quad (5)$$

$$E = \left(\frac{L}{z_{com}^{+}}\right)^2 - \frac{g}{z_{com}}(x_{com}^{+})^2 \quad (6)$$

Orbital energy and step duration would also be determined based on systems' initial conditions and with respect to the circumstances of the future steps, and then would be used in the optimization process. Following that the desired trajectories of the swing foot, in both x and z directions, would be obtained and expressed with a set of closed-form polynomials. After that would be an online optimization carried out by MPC to ensure that was solved at the same frequency as the QP-based lower-level controller, and then the processed data would be output to simulate the walking process of the robot.

1.3. Objective and Research Significance

Many current algorithms were built with a limitation, they have a restricted capability to handle a certain number of stairs, generally four to six, which are smaller in size compared to the regular stairs present in buildings. For example, even if a blind stair traversal algorithm was built upon the environment evolving the stairs with average size from the real world, there was still a limitation that the highest stair has eight steps [9]. This certainly does not hold true in real life; there will always be longer steps as well as more complex situations, and this presents a higher demand for better step planners. Researchers came up with a better approach based on the underactuated LIP model and online optimization to allow bipedal robots to walk on randomly generated constrained footholds [1]. This project aims to develop a simplified continuous stair-climbing methodology for bipedal robots with model-based approach in simulation environments.

1.4. Thesis overview

This thesis comprises four chapters. Chapter One introduces the previous model-based accomplishments achieved by other researchers in related fields, setting the stage for this work's context. Chapter Two describes the approaches adopted in this study to achieve its objectives, detailing the methodologies and theoretical frameworks employed. Chapter Three presents the results achieved and provides evaluations of these outcomes, offering a critical analysis of the work's impact and significance. Finally, Chapter Four offers a conclusion and recommendations for future work, suggesting directions for further research and potential improvements to the field.

Chapter 2. Methodology

2.1. Selection of robot model and environment

The robot model chosen for this project is RABBIT, a planar, underactuated five-link model consisting of a torso, left and right thighs, and left and right shins, demission listed in Table 1 [4]. It features four actuated joints: the left and right hip joints, and the left and right knee joints, but it lacks ankle joints due to its pointed feet design [4]. In this case, the system states of the robots are defined by the following seven parameters: x-coordinate of the COM, z-coordinate of the COM, torso orientation, torque at the right hip, torque at the right knee, torque at the left hip, and torque at the left knee. In conjunction with this model, FROST (Fast Robot Optimization and Simulation Toolkit) was employed. FROST is a trajectory optimization and simulation tool designed to generate model-based control algorithms specifically for this robot.

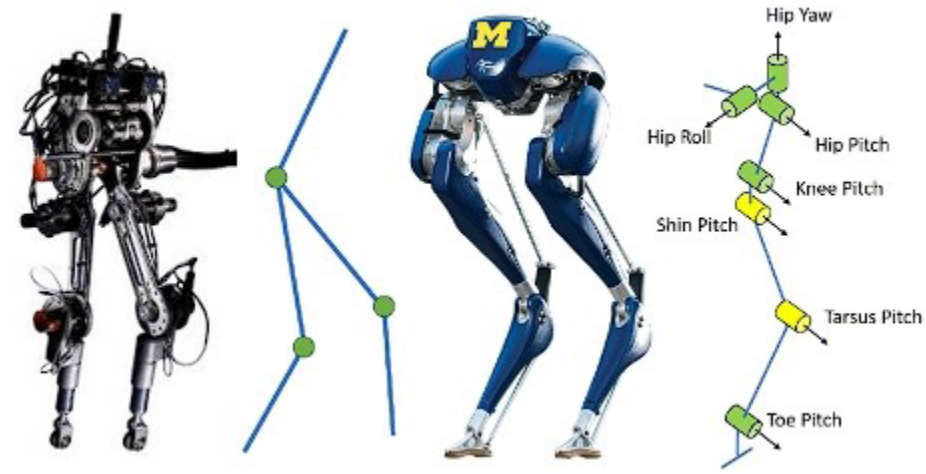

Figure 4: Genderized bipedal Rabbit 2D (left) and Cassie 3D (right) model, showing the structure including frames and joints [7].

Table 1: Description of the RIBBIT robot [4]

	Torso	Femur	Tibia
Length (m)	0.63	0.4	0.4
Mass (kg)	12	6.8	3.2
Inertia ($kg * m^2$)	1.33	0.47	0.2

2.2 Modification on Impact detection and step counting method

LIP model and its derivative models, including ALIP, were mainly designed and calibrated to help generating control algorithms to let bipedal robots walking on flat surfaces, when walking process involves vertical displacements, especially for COM displacement, aspects that need to be considered would change.

When considering the case of walking on flat surfaces, the completion of each step can be easily monitored. One key feature that was defined as an event function in Victor's work is identifying the time and position of the impact between the swing foot and ground happens [2]. Event function is a built-in feature in FROST that acts like a condition that will be triggered by certain discrete events, swing foot impacts in this case [11]. The impacts were assumed to be instantaneous and rigid. In this case, the event function will be triggered when the rigid impact between the swing foot and the ground happens, defined by the height of the swing foot equals the ground. When the impact happens, the robot will switch its stance and swing foot, increase the step count by one, and proceed to the following step. To correctly record the footstep number, another limitation was applied to the event function, the step count cannot be added if the height of the swing foot is the same as the ground within a 0.1 second range after that step was initialized.

As for walking on stairs, the event function and step counting mechanism must be modified. With the buffer time for obtaining a steady step state, the 0.1 second delay for counting the number of steps was kept, to avoid the unstable situation as the swing foot vibrates above and below the desired height frequently with small magnitude. The key change was made on the impact detection. The updated event function monitors both x and z coordinates of the swing foot, continuously updates and compares the location of the next stair and report the impact when both coordinates for the swing foot and stair matches, saying that swing foot x position locates between the start of the current stair and before the start of the next stair in the x direction, and its height matches current stair height. Assume the depth of the last stair is infinite.

2.4 Virtual constraints modification

2.4.1 Virtual constraint on torso orientation and COM trajectory

When building the environments and during the simulation process, generally some of the key state variables were chosen to define functions that modulate the robot links, to perform the desired behaviors [11]. These functions are called virtual constraints in FROST, and they are generally defined as the difference between the desired and actual outputs. The virtual constraints chosen in this work are the height of COM, torso orientation, and the position of the swing foot. The torso orientation in this case was defined with respect to a line that is perpendicular to the ground, which means that the desired state of the torso angle is the same as what is generally desired by the circumstances of walking on the flat ground.

When walking on flat ground, the trajectory of the COM of the robot can be simplified into a horizontal line, with a constant height measured with respect to the stance foot, eventually from the ground. Then its virtual constraint can be defined as a constant, meaning that in the

steady phase, the robot should be able to walk without vertical displacement of its COM. The original thought of adapting this virtual constraint towards stair climbing scenarios was to simplify the stairs into a slope and then apply the same definition, the COM will have a constant distance perpendicular to the slope surface. This idea was later proven wrong as the position of the COM was noted with respect to the stance foot, instead of the places the robot is walking on top. Defining a representation of the trajectory in this would be unnecessarily complicated as the horizontal and vertical position of the stance foot change whenever one step is finished. As the stance foot changes for each step, the virtual constraint of COM height now can be written into a break-down version. At this place a new concept is introduced, time-base:

$$\tau = \frac{T - T_0}{T_f - T_0} \tag{7}$$

Note τ is the phase variable, ranges from 0 to 1, as 0 stands for the start of one step and 1 stands for the end of it. T is the current time, T_0 is the initial time for this step, and T_f is the end time. Step duration was assumed to be constant for this work as $T_s = 0.43s$.

In this case, the virtual constraint can be defined as a line with the starting point h_r meters above the stance foot and ends at x_{des} meters in front and $z_{des} + h_r$ above. h_r is the vertical distance between COM and stance foot when the robot was standing still on flat ground, x_{des} and z_{des} are the coordinates of the desired foot placement with respect to the stance foot, equals to the depth and height of one stair in this case. At the beginning of the next step, the virtual constraint will reset to its initial state as the stance foot becomes the new swing foot, and repeat what happened in the previous step, since each stair was designed to be identical.

2.4.2 Swing foot trajectory modification

The trajectory of the swing foot contains two aspects, in x and z directions. When walking on flat ground, the trajectory in horizontal direction can be defined as a cubic curve, showing the nature of the velocity of the step: start from still, accelerate in the first half, and then decelerate to zero in the second half. The trajectory of the vertical displacement can be described as a parabola, the swing foot will first lift and reaches the highest point during the step when t = 0.5, and then drop to the ground during the second half of the step, the trajectory is symmetrical from left to right. Then virtual constraints for swing foot trajectories can be easily defined based on such simplification.

Then for walking on stairs, this work's initial thought was to continue with the virtual constraints defined for horizontal walking, but with a minor modification on only keeping the part of the second half of the step prior to the impact and finally ends after that instantaneous impact. No modification is needed for the trajectory in the x direction. However, this approach has a limitation that the tall arch for the trajectory in the z direction will not be so efficient to in cooperate with the desired horizontal displacement.

After reviewing Min Dai's work and comparing the swing foot trajectory in the z-direction, it can be concluded that a higher-order, specifically a fifth-order Bezier curve, provides a better representation. This approach significantly increases the manageable step length and produces a walking pattern more akin to human locomotion [1]. Two methods were employed to derive the representations of the swing foot trajectories:

The first method was to define six control points, a_0 through a_5, as a_0 is the initial state of one step of the robot; a_1 is derived from the first-order derivative of the initial state minus a_0; a_2 and a_3 are set to constrain the maximum height the swing foot can reach within a step,

calculated as 1.5 times the stair height. The desired velocity of the swing foot in the vertical direction at the end of each step can be obtained by subtracting a_5 by a_4, defined as -0.1 m/s to ensure the foot descends onto the stair from above avoiding unintended collisions. As illustrated in Figure 5a, the negative value added to the final control point was so large that it significantly lowered the endpoint of the fitted trajectory, potentially causing the foot to contact the stairs too early or to miss the step entirely. To address this issue, reducing the desired velocity at the end of the steps is effective. As shown in Figure 5b, the desired velocity was adjusted to -0.03 m/s, effectively mitigating the problem.

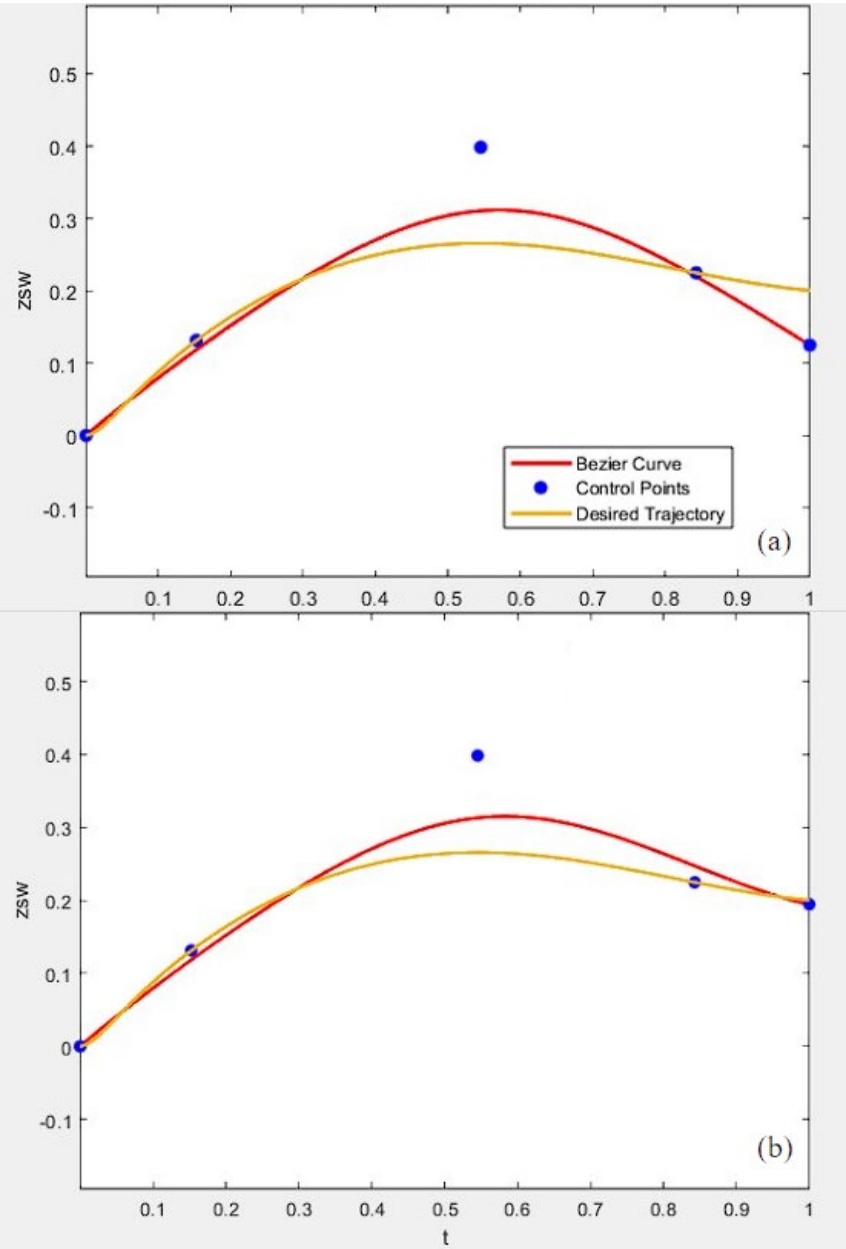

Figure 5: The effect of desired end-state velocity on Bezier curves.

The second method extracted the swing foot trajectories from Min Dai's work in both the x and z directions to generate desired trajectories, using a Bezier-fit function. The outputs of this process were subsequently utilized to define the virtual constraints. The principal distinction between these two approaches lies in their methods for constraining the maximum height of the swing foot trajectory and in whether the control points are explicitly defined. In Min Dai's

method, the Bezier coefficients a_1 through a_3 were defined in the same way by adding 0.1 to the height of the highest stair which is within a three-step range of the robot, encompassing the current step, one stair preceding it, and one stair following it. Similarly, a negative value was added to a_5 for the same purpose while a_4 is the height of the current stair. Unlike the previous approach, this method does not require matching or locating the control points with time, or the phase variable during a step, allowing the trajectories to be obtained independently.

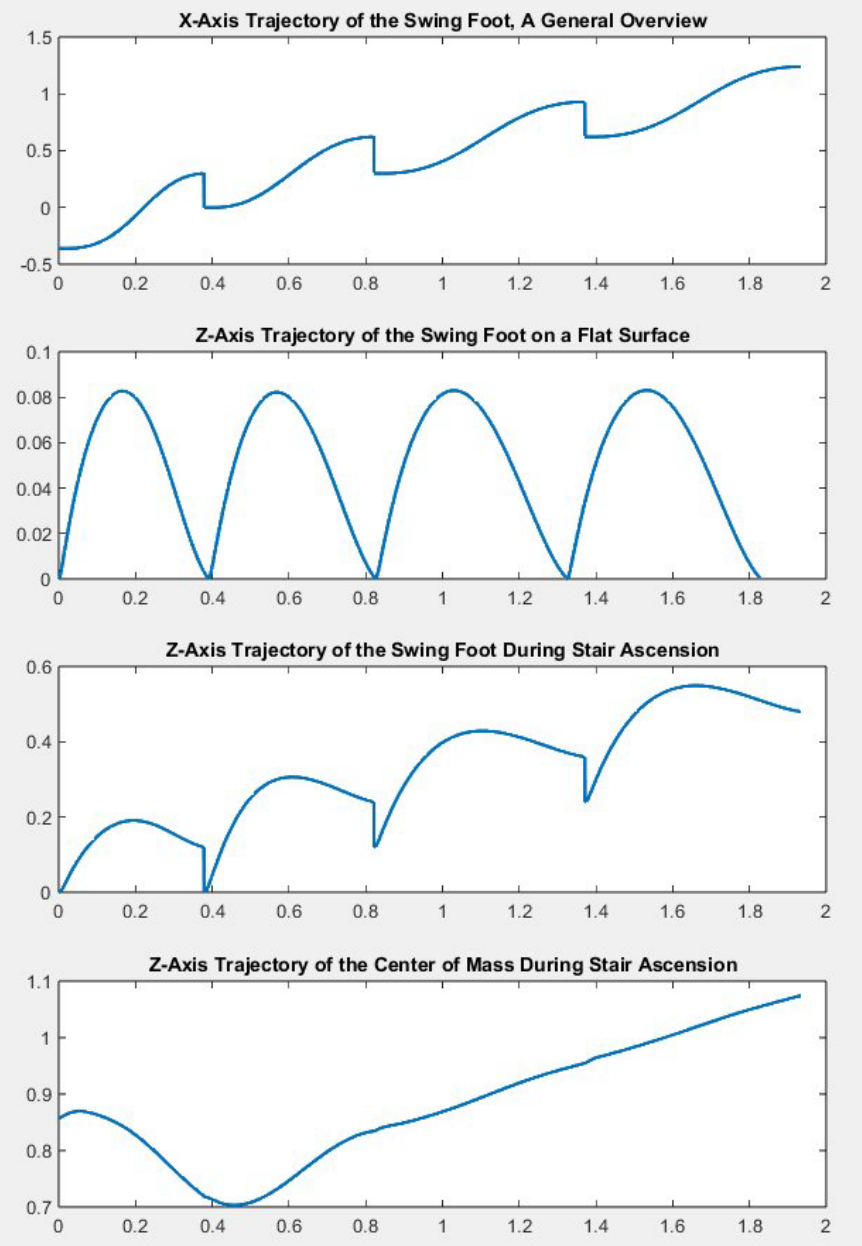

Figure 6: typical swing foot and COM trajectories

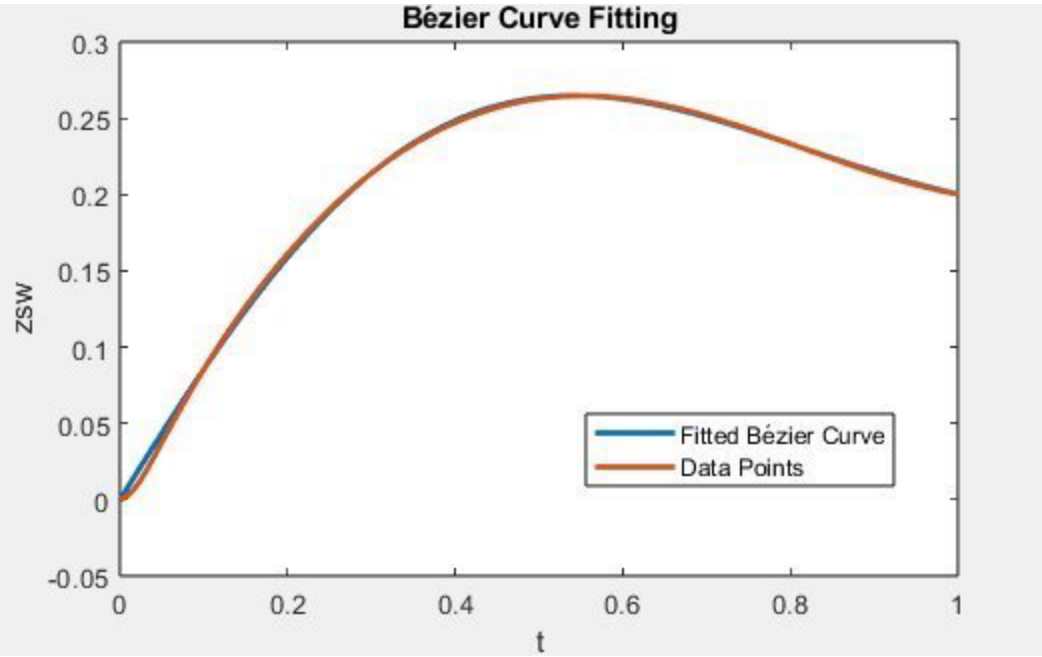

Figure 7: The desired swing foot trajectory in Z direction fitted by Bezier curve.

Chapter 3. Result and discussion

3.1 Results

According to Ohio's code on stairs, Rule 4781-6-03.8 specifies a maximum riser height of 8.25 inches (210 mm) [12], the testing environment was chosen with a depth of 0.2m and a height of 0.1m. Following the methodologies outlined in the previous sections, the simulation process yielded a result. When using the default stair parameters, one full step could be completed. However, starting with the second step, the actual trajectory of the swing foot, especially in the z-direction, failed to match the desired trajectory. This discrepancy led to a chain reaction that ultimately caused the entire system to collapse, especially for the COM trajectory, as shown in Figure 8 and 9. Referring to Figure 10 and 11, performance improved when the stair height was reduced to 0.02m, allowing four steps to be completed. The simulation failed at the fifth step because the desired trajectory of the swing foot in the x-direction shifted to

an unreasonable value, which could be due to an accumulating inherited error within the simulation process.

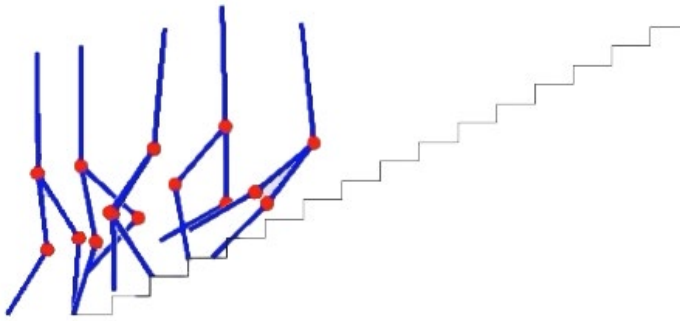

Figure 8: Simulation result for stair height as 0.1m

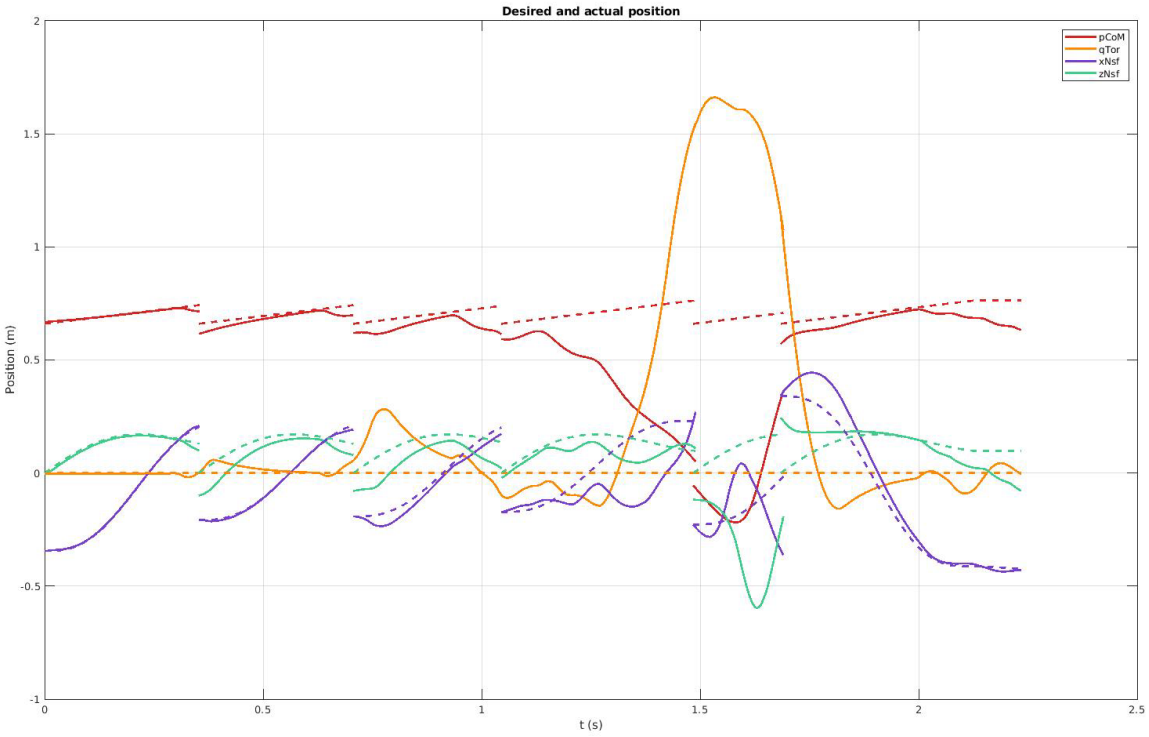

Figure 9: The desired and actual trajectories of the virtual constraints when stair height was 0.1m, desired ones are noted by the dashed lines while the actual output was represented by solid lines.

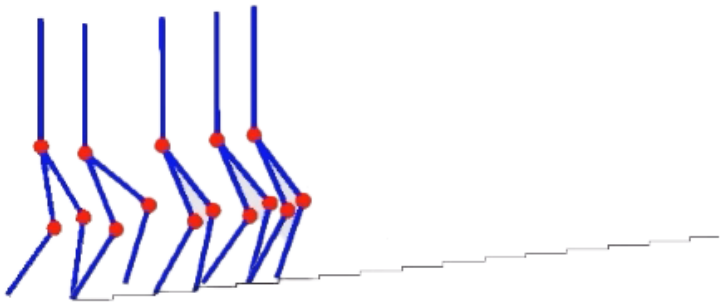

Figure 10: Simulation result for stair height as 0.02m

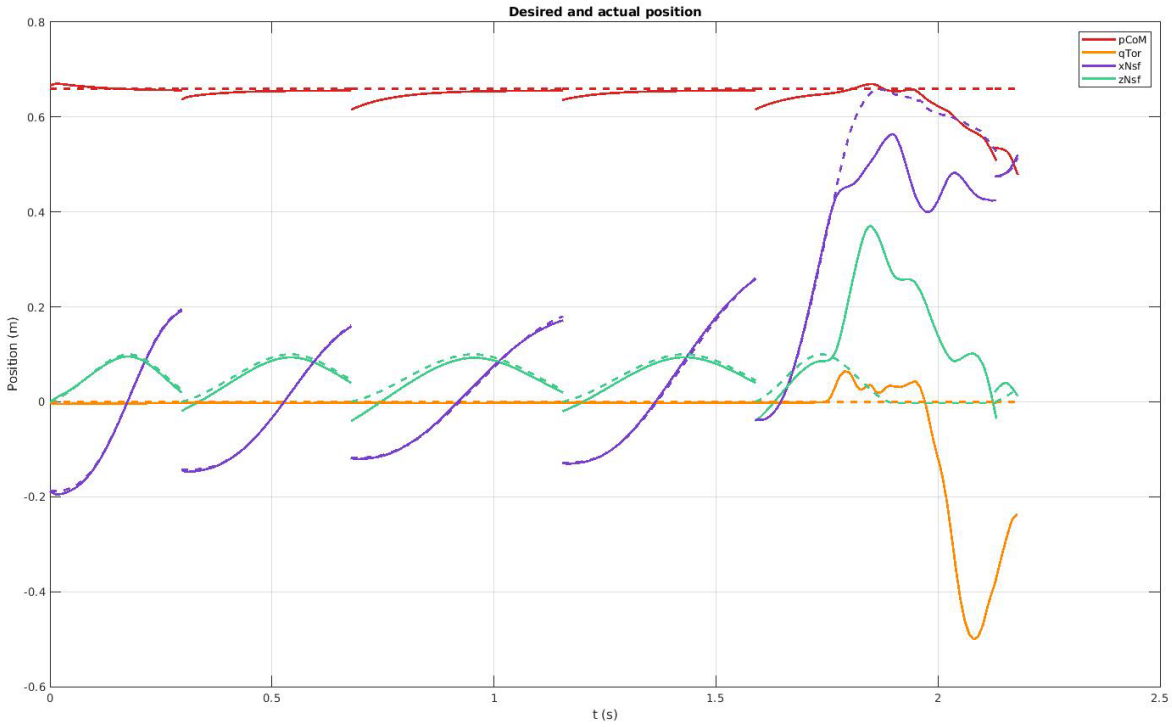

Figure 11: The desired and actual trajectories of the virtual constraints when stair height was 0.02m, desired ones are noted by the dashed lines while the actual output was represented by solid lines.

3.2. Possible failure reason analysis

As previously mentioned, the state variables used to monitor the dynamics of the robots are referred to as virtual constraints. In this context, they include the COM height, torso orientation, and the swing foot's trajectory in both the x and z directions. A failure in one or more of these constraints can lead to system collapse, as demonstrated in Figure 9 and 11. Initially, in the default settings, the COM height and swing foot trajectory in the z-direction failed to follow their intended paths. This deviation quickly resulted in a minor error in torso orientation. By the third step, all virtual constraints were impacted, causing the robot to lose its ability to walk stably. Furthermore, by the fifth step, the desired state of each parameter was significantly off target, especially for the step duration, which was much shorter than in the first two steps. This occurred despite the intention for it to remain constant across all steps. Combined with the observations in Figure 11, this indicates that maintaining consistency is an important aspect to consider in planner design.

The feature illustrated in Figure 8 also indicates a discrepancy between the velocity of the COM in the direction the robot is moving, particularly in the x-direction, and the movement of the robot's lower limbs. This suggests that the planned trajectory did not account for the scenario where the COM moves faster in the x-direction, which, in this case, causes the robot to fall forward. Such a feature, along with the desired trajectory for the last step shown in Figure 11, implies that this direct method of defining desired trajectories is insufficient for outlining the robot's overall motion during the simulation. A better planning strategy is required to address the velocity differences between the robot's upper and lower body.

Another notable example of a state variable critical to the simulation process, yet not included in the virtual constraints, is the knee angle of the robot during walking, as depicted in

Figure 12. The knee angle shown in Figure 12a represents the normal orientation expected for the RABBIT model. Figure 12b displays the state of the robot 0.544 seconds after the initial state depicted in Figure 12a, highlighting that the knee angle of the swing foot has increased to significantly more than 180°. Because forcing LIP model operating on the stairs makes the robot's state unstable. And there is no restriction on the joint angle inside the RABBIT model, so this deviation is also a phenomenon that the robot is not well controlled, which serves as another contributing factor to the collapse of the simulation process.

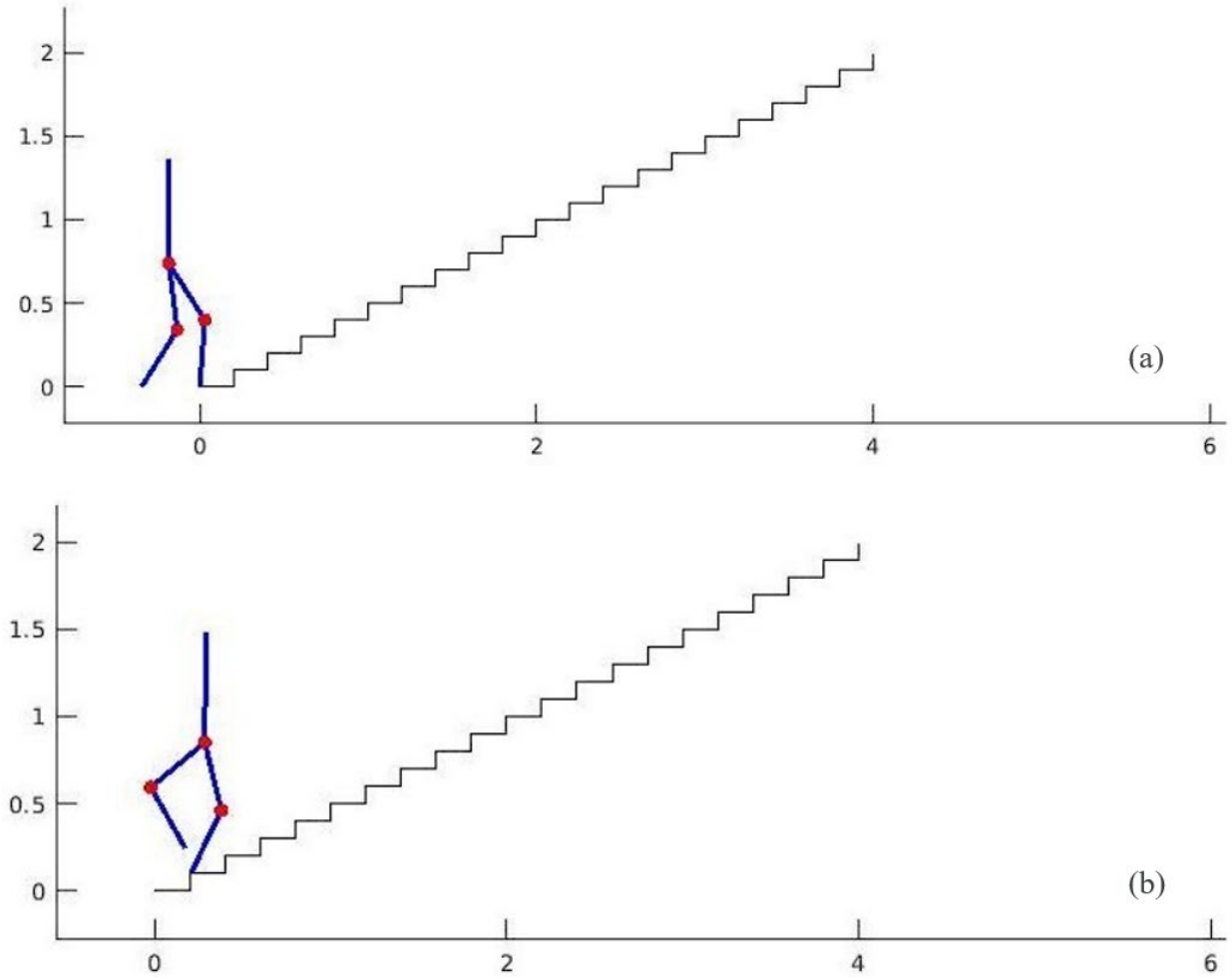

Figure 12: (a): default status of the RABBIT robot model; (b): the undesired status of the RABBIT robot model

Chapter 4. Conclusion and future recommendations

4.1 Contributions

This work demonstrates that, with modifications, the LIP model can partially manage vertical displacements. Newly defined methods for impact detection and step counting, along with COM and swing foot trajectory planning, functioned as intended. Success levels increase as stair height decreases.

However, the controller is also a factor that limited the model's ability to fully achieve the ideal state. Despite modifications to virtual constraints and crucial elements enabling trajectory generation for both level of controllers, COM vertical displacement and velocity in both directions is not fully regulated, indicating incomplete execution of planned motions. Further enhancements are necessary for both controller and planner to achieve desired outcomes. Additional features could be introduced to address these issues and enhance the versatility of the policies.

4.2 Future recommendations

The findings of this work were not entirely ideal. To fully address such challenges, three potential solutions are proposed. The first is to develop a new system dynamical model that inherently manages changes in both vertical and horizontal positions. Although feasible, this solution may introduce a complexity far beyond that of the currently utilized LIP model and its variants. This might also compromise the algorithm's efficiency due to increased system dynamics complexity. Another approach is to enrich the existing model with more state variables that can be monitored, such as knee angles, which may enhance the current model by providing additional parameters that can be manually regulated and defined in the planning state. And a

better trajectory planning methodology is also needed to deal with the coexistence of both actuated and underactuated parts in the model, as well as the state variables, such as the linear velocity of the COM in both x and z directions, that cannot be defined directly. Since the involved desired trajectories were planned with greater precision, this approach minimizes errors and uncertainties caused by multi-directional displacements. Involving features like MPC might also increase the efficiency and accuracy of the controller, which would enable more precise regulation of the foot and COM trajectories. Furthermore, it would allow for more efficient management of specific gait restrictions or requirements through optimization tools. The last solution, which I find to be the most feasible and would pursue if given the opportunity to continue this research, is to involve or shift to learning-based approaches. Machine learning algorithms, such as reinforcement learning, allow the system to operate more intuitively compared to the process of deriving control algorithms through model-based approaches. This could also provide a higher level of versatility.

Appendix A. Sample Flow Charts of Control Algorithms for Bipedal Robots

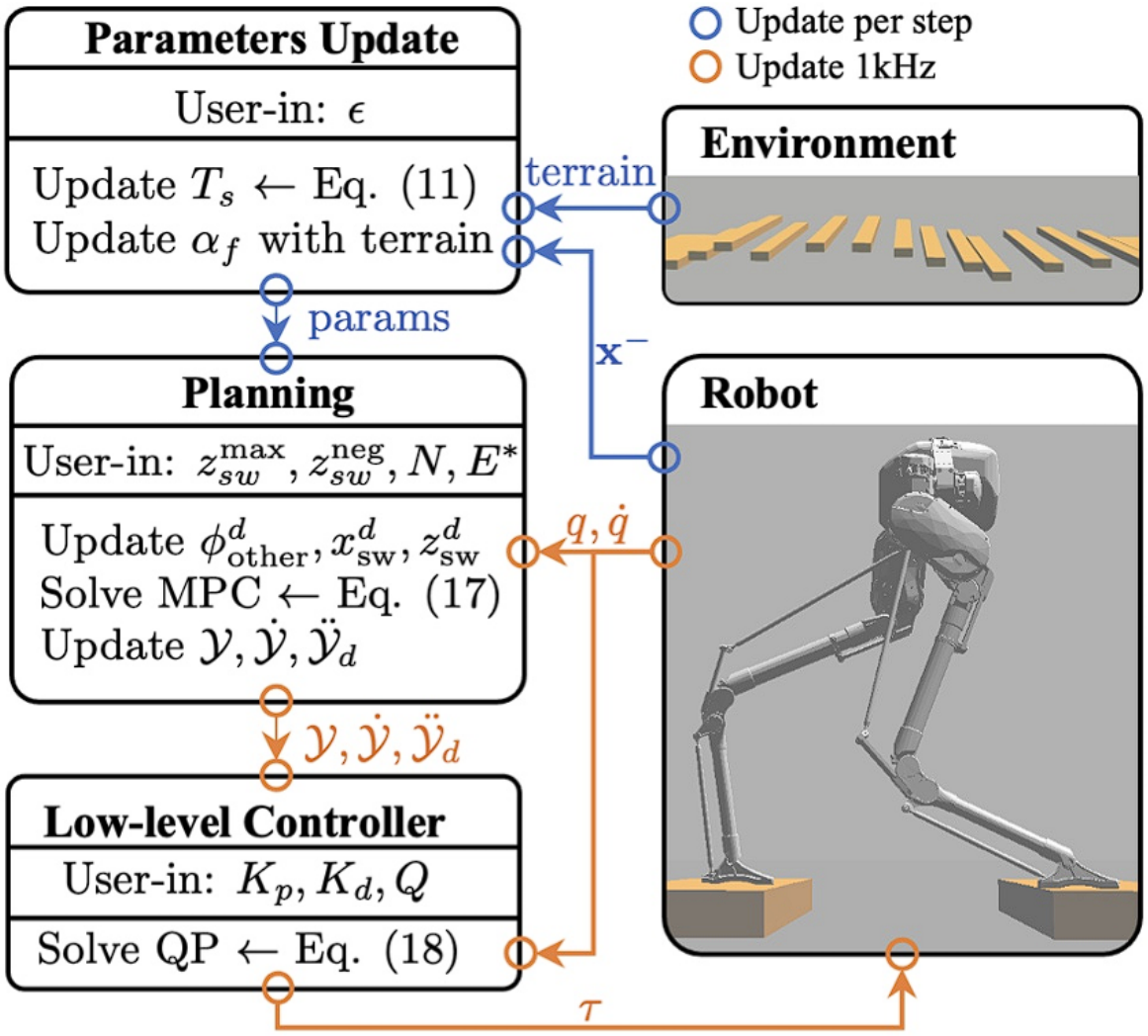

Figure 13: Walking synthesis overview [1]

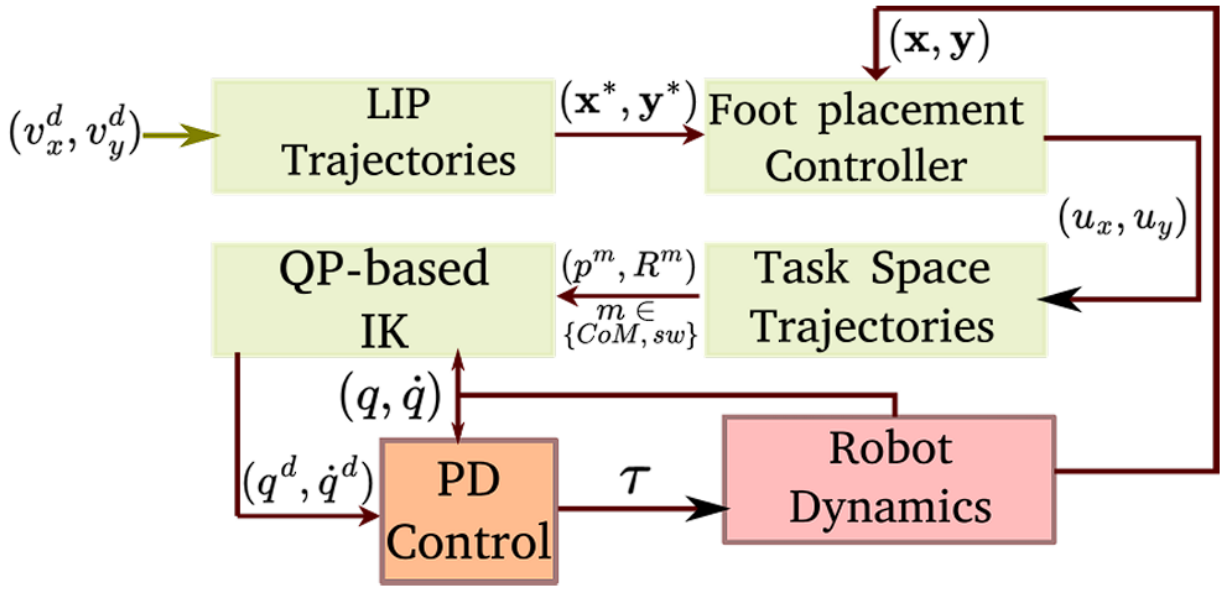

Figure 14: The resolved motion framework starts with the ideal LIP Trajectories [2]

References.

[1] Dai, Min, et al. “Bipedal Walking on Constrained Footholds: Momentum Regulation via Vertical COM Control.” 2022 International Conference on Robotics and Automation (ICRA), 2022, doi:10.1109/icra46639.2022.9812247.

[2] Paredes, Victor C., and Ayonga Hereid. “Resolved Motion Control for 3D Underactuated Bipedal Walking Using Linear Inverted Pendulum Dynamics and Neural Adaptation.” 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, doi:10.1109/iros47612.2022.9982009.

[3] Xiong, Xiaobin, and Aaron Ames. “Slip Walking over Rough Terrain via H-Lip Stepping and Backstepping-Barrier Function Inspired Quadratic Program.” IEEE Robotics and Automation Letters, vol. 6, no. 2, 2021, pp. 2122–2129., doi:10.1109/lra.2021.3061385.

[4] Castillo, Guillermo A., et al. “Reinforcement Learning Meets Hybrid Zero Dynamics: A Case Study for Rabbit.” 2019 International Conference on Robotics and Automation (ICRA), 2019, doi:10.1109/icra.2019.8793627.

[5] Dosunmu-Ogunbi, Oluwami, et al. “Stair Climbing Using the Angular Momentum Linear Inverted Pendulum Model and Model Predictive Control.” arXiv.Org, 11 July 2023, arxiv.org/abs/2307.02448. Accessed 25 Oct. 2023.

[6] Gibson, Grant, et al. “Terrain-Adaptive, ALIP-Based Bipedal Locomotion Controller via Model Predictive Control and Virtual Constraints.” arXiv.Org, 28 July 2022, arxiv.org/abs/2109.14862v3.

[7] Gong, Yukai, and Jessy Grizzle. “Zero Dynamics, Pendulum Models, and Angular Momentum in Feedback Control of Bipedal Locomotion.” arXiv.Org, 3 Feb. 2022, arxiv.org/abs/2105.08170v2.

[8] Y. Gong and J. Grizzle, "One-Step Ahead Prediction of Angular Momentum about the Contact Point for Control of Bipedal Locomotion: Validation in a LIP-inspired Controller," 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 2021, pp. 2832-2838, doi: 10.1109/ICRA48506.2021.9560821.

[9] Siekmann, Jonah, et al. “Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning.” Robotics: Science and Systems XVII, 2021, doi: 10.15607/rss.2021.xvii.061.

[10] Kajita, S., et al. “The 3D Linear Inverted Pendulum Mode: A Simple Modeling for a Biped Walking Pattern Generation.” Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the Next Millennium (Cat. No.01CH37180), 2001, doi:10.1109/iros.2001.973365.

[11] Hereid, Ayonga, and Aaron D. Ames. “Frost*: Fast Robot Optimization and Simulation Toolkit.” 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sept. 2017, doi:10.1109/iros.2017.8202230.

[12] “Rule 4781-6-03.8: Exterior and Interior Close-Up.” Rule 4781-6-03.8 - Ohio Administrative Code | Ohio Laws, codes.ohio.gov/-administrative-code/rule-4781-6-03.8. Accessed 25 Feb. 2024.