Trajectory Optimization of a Kneed Compass Gait on Rough Terrain

George Chen
Dept. of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, MA
gcfchen@mit.edu

Silvia Knappe
Dept. of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA
seknappe@mit.edu

***Abstract*—To explore the area of actuated bipedal walkers, this research project presents an optimization-based method for a bipedal robot to navigate a rough terrain. Using a kneed compass gait as the robot model, we were able to formulate its kinematic and dynamic constraints, and plan a minimal-energy trajectory through an uneven terrain in simulation. After successfully solving the trajectory optimization problem using the Sparse Nonlinear OPTimizer (SNOPT) in Drake, we were able to compare the compass gait's limit cycle in flat terrains to that in rough terrains, and visualized the generated trajectory in MeshCat.**

***Index Terms*—trajectory optimization, limit cycle, legged locomotion, rough terrain navigation**

I. Introduction

Bipedal robots enjoy the advantage of their agility and their versatility of two freed hands, but their stability and controllability introduce unique challenges to research. There is an abundant amount of existing research [1] [2] on quadrupedal robots, which include justifying their stability and using different control policies to plan their behavior. The societal applications that bipedal robots can be applied to, such as contactless delivery as well as search and rescue, both of which benefit from the bipedal robot's free arms, are of great interest in the field of robotics research.

In existing literature on rough terrain navigation, quadrupedal robots are more commonly used compared to bipedal robots. One potential reason is that quadrupeds are easier to balance on rough terrain and would be less likely to tip over. However, bipedal robot research has also been growing significantly and there are many examples of bipedal robots being used to traverse rough terrain. One example makes a minimal controller for a compass-gait type bipedal robot walking over rough terrain [1], but does not take into account sensory feedback. Other bipedal robots use force sensing in their controls scheme [2], but the speed of biped robots traversing rough terrain is still quite slow (0.5 m/s). A review of underactuated bipedal robots also confirms that walking speeds for bipedal robots across rough terrain is an aspect that can be improved, as well robot performance on inclined terrains [3]. Overall, the problem of bipedal legged locomotion over rugged terrains is a worthwhile research area.

II. Kneed Compass Gait Dynamics

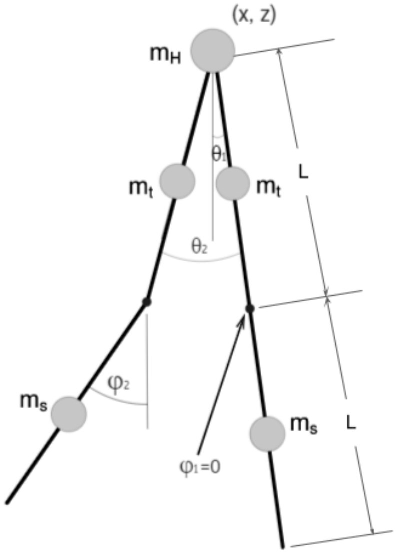

Fig. 1. Kneed compass gait kinematics.

Consider the kneed compass gait model shown in Figure 1. The robot is modeled as a 4-link system, each with a point mass at its center, and a hip connected at the top via a continuous joint. The position of the hip has coordinates (x, z), and the stance leg has an absolute angle of θ_1 with respect to the vertical axis. The rest of the coordinates are relatively defined. Let θ_2 be the relative angle between the stance leg and the swing leg. Let φ_1 and φ_2 be the angles at the knee joints (i.e. the relative angle between the lower leg and the upper leg) of the stance and swing legs, respectively. A URDF model of the kneed compass gait model is adapted from Michael Posa's work from DAIR Lab of UPenn [4], with modified physical parameters shown in Table I.

The configuration vector is therefore defined as a 6-dimension vector $\mathbf{q} = [x, z, \theta_1, \theta_2, \varphi_1, \varphi_2]^T$, and the system state vector is $\mathbf{x} = \left[\mathbf{q}^T, \dot{\mathbf{q}}^T\right]^T$. The nonlinear rigid body dynamics of the kneed compass gait system can be succinctly

Parameter	Symbol	Value
upper leg length	L	0.5
lower leg length	L	0.5
hip mass	m_H	10
thigh / upper leg mass	m_t	2.5
shank / lower leg mass	m_s	2.5

TABLE I
PHYSICAL PARAMETERS OF THE KNEED COMPASS GAIT MODEL.

described by the manipulator equation [5],

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} = \tau_g(\mathbf{q}) + \mathbf{B}\mathbf{u} + \sum_i \mathbf{J}_i^T(\mathbf{q})\lambda_i \tag{1}$$

where $\mathbf{M}$ captures the system's inertial properties, $\mathbf{C}$ the Coriolis forces, and τ_g the gravity vector. λ_i are the constraint forces at both feet (i.e. $i \in \{1,2\}$), modeled as point contact, and $\mathbf{J}_i$ are the constraint Jacobians. The actuation matrix $\mathbf{B}$ maps control inputs $\mathbf{u}$ at the knee joints and the hip into generalized forces.

The limit cycle of the kneed compass gait to consists of a two-link phase and a three-link phase, as illustrated in Figure 2. The stance leg is modeled as one rigid link like a pendulum. At the start of the step, the knees starts out locked (i.e. $\varphi_1, \varphi_2 = 0$); its dynamics is equivalent to that of a two-linked compass gait. Its swing leg then bends its knee to make a stride, makes a knee strike and lands again with the knee locked, returning to the dynamics of a two-link pendulum. After the heel strike with the ground, the stance and swing legs switch instantaneously, and the limit cycle repeats.

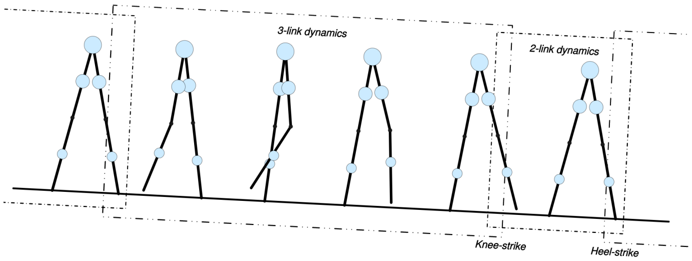

Fig. 2. Limit cycle trajectory of a kneed compass gait. [5]

III. Optimization

In order to extract the dynamical properties of the given URDF model, we constructed a `MultibodyPlant` in Drake [6] and utilized the `MathematicalProgram` to solve for the trajectory optimization problem such that the robot would navigate across the terrain.

We implemented an L2-norm cost on actuation effort so that the robot could achieve its goal with minimal energy while obeying all constraints. Mathematically, the optimization problem can be written as

$$\min ||\mathbf{u}||^2 \textit{ s.t. dynamical constraints}$$

where the dynamical constraints are discussed below.

The first portion of the constraints relates to the dynamics of the robot. With the mass matrix M, Coriolis forces C, gravity τ_g, and actuation matrix B obtained with Drake from the model of the robot, we imposed a constraint that the residuals of the manipulator equation must equal zero. We also assumed inelastic collisions between foot and ground, and reset the velocity after every heel strike accordingly. The swing- and stance- feet, when in contact with the ground, must reside within the friction cone. Implicit Euler dynamics described in Equation (2) and Equation (3) were also implemented to calculate position and velocity at the next timestep.

$$\mathbf{q}[t+1] = \mathbf{q}[t] + \mathbf{h}[t]\dot{\mathbf{q}}[t+1] \tag{2}$$

$$\dot{\mathbf{q}}[t+1] = \dot{\mathbf{q}}[t] + \mathbf{h}[t]\ddot{\mathbf{q}}[t] \tag{3}$$

The next portion of constraints relates to the compass gait kinematics. At the beginning of a step, we constrained the robot to start with both feet on the ground. After the first step, in order to maintain continuity between steps, we put a stricter requirement that the feet had to start in the position as the previous step. This means that after the robot swung its right leg forward to take a step, for the next step, the right leg should start where it finished in the previous step. Although perhaps unlikely given the cost we implemented, we also put constraints on the limits of the angles that the joints of the robot could form. For example, we constrained the hip joint such that the legs of the robot could at maximum be 180 degrees apart,

$$-\frac{\pi}{2} \leq \mathbf{q}[t,2] + \mathbf{q}[t,3] \leq \frac{\pi}{2}, \forall t \in [0,T) \tag{4}$$

We also constrained the knee joints to only bend in the positive direction, and limited them also to 180 degrees of mobility; the lower leg of the robot should not rotate past the upper leg.

$$0 \leq \mathbf{q}[t,x] \leq \frac{\pi}{2}, \forall t \in [0,T) \tag{5}$$

where x represents the index corresponding to the knee that belongs to the swing leg. In order to take multiple steps, we switched the swing and stance legs instantaneously after each step was taken.

By definition, the stance foot of the robot was constrained to be touching the ground at all times during the step, and the knee in that leg was constrained to be locked out and straight. In accordance with the 2-link/3-link transition described in Figure 2, both the stance and swing legs were also constrained to be straight at the beginning and end of a step. This constraint was mostly kept for simplifying reasons; constraining the swing leg to be straight at the end of its step means it's already straight when the next step starts and the swing leg becomes the plant leg, so no additional work is needed to straighten out the swing leg.

We elaborate further on the ground in the simulation section, but in our ground made up of blocks, we make a simplifying assumption that the robot advances 1 block per step. This means that the swing foot starts out 1 block behind the stance foot, and moves 1 block in front of the stance foot. Given this assumption, we constrained the final position of the swing foot to land within the next block relative to the stance leg.

Lastly, we needed to make sure the legs didn't penetrate the ground and that the swing leg didn't collide with the ground on taking a step. To do this, we implemented 2 helper functions. One for getting the (x, z) distance from stance foot with respect to the ground, and one for getting the (x, z) distance of the swing foot with respect to the ground. We use the stance foot helper to ensure that the distance between the stance foot and the ground is always 0. For the swing leg, we constrained the foot to have some clearance distance between it and the ground so that the leg didn't scuff on the ground while walking, and so that the leg didn't collide into the ground on the step. We did this by making use of the helper function that got the distance between the swing foot with respect to the ground and made sure it was some clearance distance above the ground. The distance we ended up using was the height of the blocks we made the ground out of.

In order to successfully solve the optimization problem, we did allow the robot some slack room in our constraints. Everywhere where we constrained the foot position of the robot to be 0, our realistic constraint was $0 \pm \epsilon$. This is because the function used to get the distance between the foot and the ground (`CalcPointsPositions`) never returned 0 exactly when the foot was on the ground, which is likely due to discretization error.

IV. Simulation and Results

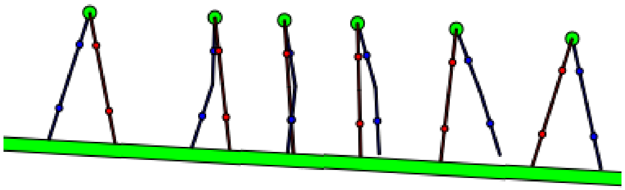

Fig. 3. HTML animation of the kneed compass gait walking on a smooth slope.

Our simulation environment consists of the robot model as described in Posa's URDF and the rough terrain ground model. As a preliminary step, we investigated the compass gait's behavior when walking on a flat slope (Figure 3). After imposing a periodic constraint and all dynamical constraints (less the foot clearance constraint) stated above, the optimization problem was solved using the SNOPT solver in Drake [6]. From the resulting motion, it is evident that the angles of the legs are continuous in the compass gait's limit cycle. The phase portrait of its states has two vertical jumps, indicating that the next step starts at the angle at which the previous step ends, but with a discontinuous angular velocity. Refer to Figure 4 below for a visualization.

The rough terrain model composes of 10 rectangular blocks that are 0.5 in width and 0.1 in height, rotated by a random angle $\alpha \sim \mathcal{U}(0, \pi/8)$ (a uniformly distributed random variable between 0 and $\pi/8$). A visualization of the environment is illustrated in Figure 5. To simplify our implementation, we have me some assumptions for our environment:

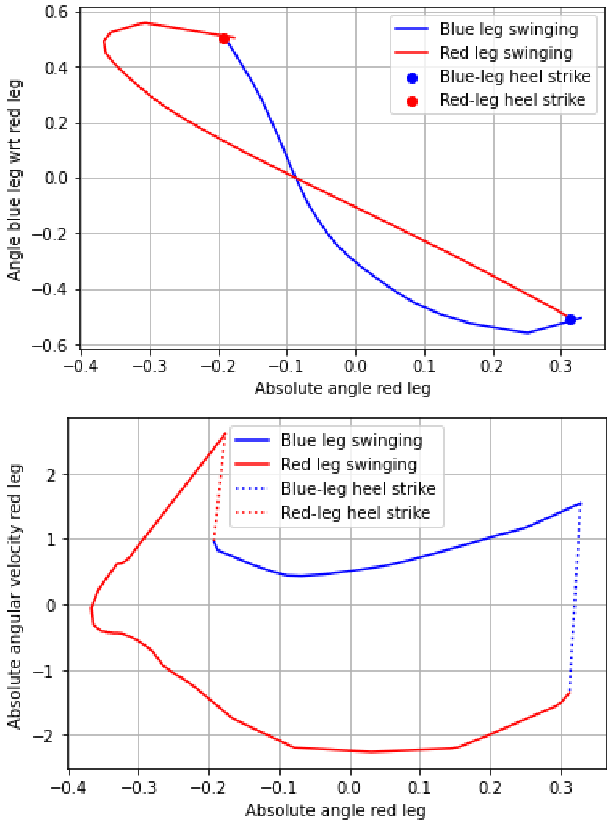

Fig. 4. Limit cycle of the kneed compass gait walking on a smooth slope. The blue trace corresponds to the stance leg and the red trace the swing leg.

- We have perfect knowledge of our uneven terrain, i.e. the position and orientation of each block.
- The blocks do not overlap one another.
- The robot takes one step per block, as mentioned in the optimization section.

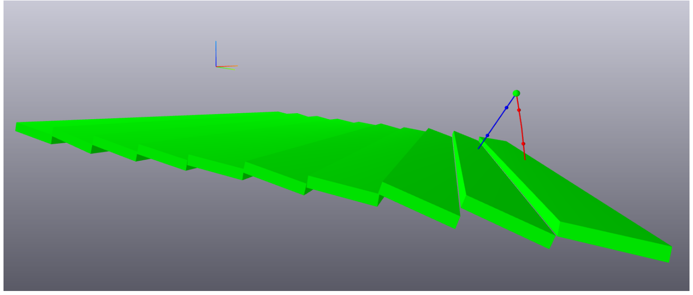

Fig. 5. Simulation environment setup.

These assumptions do not completely limit our approach. The results presented in this paper serve as a proof of concept, but if we were to implement a more robust approach to plan trajectories in uneven terrains, blocks may be overlapping and the third assumption will be relaxed.

The resulting motion is shown in Figure 6. As expected, the robot tends to bend its knee more and raise its foot higher when the ground is more uneven. However, an interesting ob-

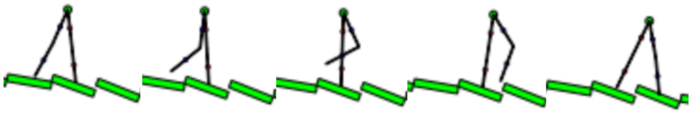

Fig. 6. One step in HTML animation.

servation is that, with our optimization approach, the compass gait prefers to use its hip actuator more due to the higher cost of additionally actuating its knees. An attempt to recover the compass gait's "limit cycle" is shown in figures below. In contrast to walking on flat grounds, the compass gait's motion is aperiodic when walking on uneven terrains. As evident in Figure 7, the hip angles at the start of a step do not line up with the configuration at which the heel strike occurred.

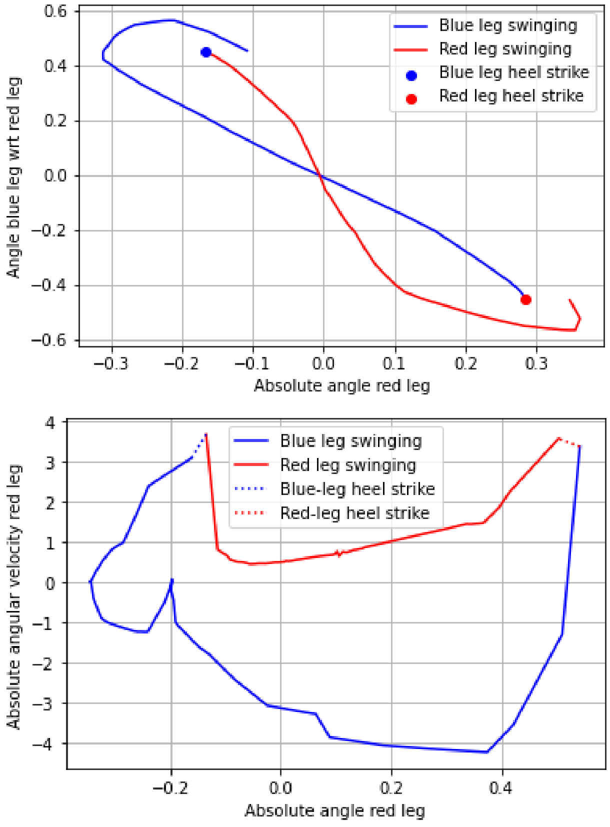

Fig. 7. Limit cycle of the kneed compass gait navigating a rough terrain. The blue trace corresponds to the swing leg and the red trace the stance leg. Note the knee strike occurred on the red trace in the bottom graph.

To confirm the results we visualized in the simulations, plots of the state variables were generated to be able to quantitatively visualize the trajectory of the robot. These plots confirm that the stance knee is kept straight throughout the step. Overall, the robot is moving forwards at a steady speed, as can be seen in the x-hip plot, and it is lifting its knee to take a step, as can be seen be the increase in angle in the swing knee plot.

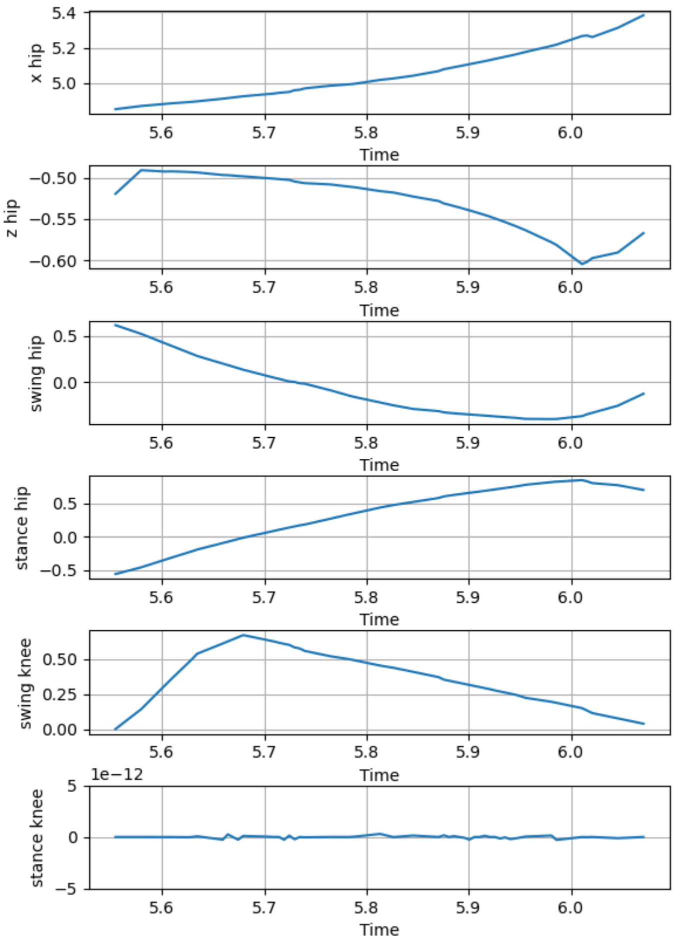

Fig. 8. State variables during a step.

V. Future Work

Our piecewise linear representation of the ground may allow us to naturally extend this project to design trajectories for kneed compass gaits to climb up and down flights of stairs. On the other hand, our simulation admittedly has an environment with many simplifying assumptions and therefore cannot be directly applied to a real-world environment. First, the simulation assumes perfect information about the rough terrains. However, even with the state-of-the-art extroceptive sensors, robotic systems in the real world can neither measure nor model perfectly the unknown stochasticity in the environment. Further, the designed trajectory from the solver could be stabilized using trajectory stabilization techniques such as the finite-horizon linear quadratic regulator (LQR), which will significantly better our understanding of the system's dynamics.

References

[1] F. Iida and R. Tedrake, "Minimalistic control of biped walking in rough terrain," *Autonomous Robots*, vol. 28, no. 3, pp. 355–368, Apr. 2010. [Online]. Available: https://doi.org/10.1007/s10514-009-9174-3

[2] F. Sygulla and D. Rixen, “A force-control scheme for biped robots to walk over uneven terrain including partial footholds,” *International Journal of Advanced Robotic Systems*, vol. 17, no. 1, p. 1729881419897472, 2020. [Online]. Available: https://doi.org/10.1177/1729881419897472

[3] S. Gupta and A. Kumar, “A brief review of dynamics and control of underactuated biped robots,” *Advanced Robotics*, vol. 31, no. 12, pp. 607–623, 2017. [Online]. Available: https://doi.org/10.1080/01691864.2017.1308270

[4] M. Posas. (2021) Planarwalker.urdf. https://github.com/DAIRLab/dairlib/blob/master/examples/PlanarWalker/PlanarWalker.urdf.

[5] R. Tedrake, *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832)*, downloaded on May 20, 2021 from http://underactuated.mit.edu/.

[6] “Model-based design and verification for robotics.” [Online]. Available: http://drake.mit.edu/