Trajectory Optimization for a Wheel-Legged System for Dynamic Maneuvers that Allow for Wheel Slip

Guillaume Bellegarda and Katie Byl

***Abstract*— Wheel-legged systems have the potential to combine the benefits of two heavily studied research areas: the speed and efficiency of locomotion on wheels with the versatility and agility to surmount various obstacles of legged systems. However, there have been limited results in dynamic locomotion taking advantage of both of these areas. In this paper, we present a trajectory optimization framework for a skating system on passive unactuated wheels, which has no means to locomote itself without exploiting tangential frictional forces. Research in planning for wheeled systems traditionally enforces non-slip (the velocity in the rolling direction is exactly equal to the wheel angular velocity multiplied by the wheel radius, without loss) and non-skid (no velocity perpendicular to the wheel roll direction) conditions, which will be frequently violated during any motion, especially over terrains with low coefficients of friction such as icy or slippery roads. By explicitly modeling friction, and specifically *not* enforcing these traditional non-slip and non-skid conditions, our framework naturally discovers dynamic maneuvers to avoid and/or exploit wheel slipping and skidding depending on the cost function. We illustrate several examples such as energy-efficient forward locomotion, skidding to a halt, and hybrid wheel-legged skating. A video showing these examples can be found at `https://youtu.be/uwXHU6vYq3E`.**

I. Introduction

Wheeled mobile robots (WMRs) have been studied extensively, as the benefits from speed and ease of deployment on flat terrain are clear. There have also been incredible advances in locomotion for legged systems such as bipeds and quadrupeds, which have the advantage of being able to traverse more hazardous and uneven terrain more reliably than WMRs. Wheel-legged systems seek to combine the advantages of both of these areas of research, namely the speed of locomoting on wheels over flatter terrain, with the added agility to overcome obstacles such as stairs and debris. Most research for wheel-legged systems until now has considered only static stability of the system, with fewer examples of dynamic or agile behaviors. Of note is the (usual) reliance on the no slip condition for the wheel (i.e. the planar velocity v in the wheel roll direction is exactly equal to its angular velocity ω multiplied by its radius r, $v = \omega r$, with no friction loss), and the no skid condition (zero velocity along the wheel's rotational axis, that is, no velocity perpendicular to its roll direction). These assumptions are not always valid, and in fact frequently violated, especially over terrains with low coefficients of friction such as icy and slippery roads. In this work, we show that by directly accounting for friction at wheel contact points, dynamic and realistic motions are found by our trajectory optimization framework.

This work was funded in part by NSF NRI award 1526424.

Guillaume Bellegarda and Katie Byl are with the Robotics Laboratory, Department of Electrical and Computer Engineering, University of California at Santa Barbara (UCSB). `gbellegarda@ucsb.edu`, `katiebyl@ucsb.edu`

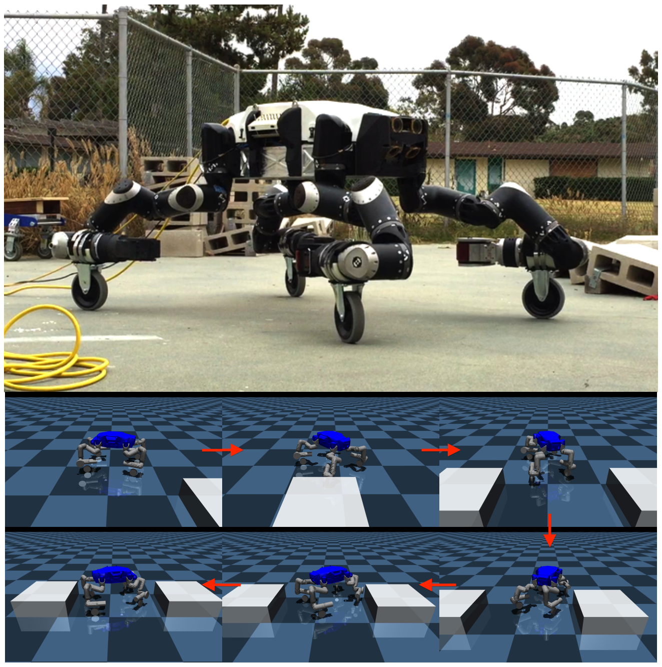

Fig. 1. Robosimian as a wheel-legged system on passive wheels. Top: Skating on flat ground with the real system. Bottom: Clockwise from top left, a dynamic parking skidding trajectory, in MuJoCo [7].

We specifically study JPL's Robosimian quadruped [1], [2], [3], [4], [5], which is shown in Figure 1. Robosimian has four identical limbs, each with seven degrees of freedom. A single, passive (unactuated), wheel with no sensing is mounted at each forearm to transform the robot into a wheel-legged system, while also leaving the option to reconfigure for exclusively walking gaits. Previous work in [6] evaluates the robustness of a variety of hand-designed skating motions, comparing specifically skating with three vs. four wheels. As these hand-designed trajectories do not account for varying frictional forces or disturbances in the environment, there is a mismatch between desired and real world states when executed open-loop. Thus, we seek a more rigorous and robust approach to skating locomotion.

Another approach we have taken to developing skating locomotion primitives is through reinforcement learning as in [8]. In this work we used a model-free reinforcement learning algorithm, Proximal Policy Optimization (PPO) [9], which means the agent indirectly learned properties about the environment through choosing actions in task space in the simulation environment, without explicitly learning or modeling friction or terrain. Thus the agent learns a policy that may or may not be taking advantage of wheel slip or skid to maximize rewards, without an explicit model or observation of these phenomena.

A. *Related Dynamic Wheel-Legged Work*

As general wheeled-legged motion in the static stability sense has been studied by many groups, we focus here on several recent works investigating dynamic wheel-legged locomotion. One such example is a trajectory optimization framework applied to ANYmal [10]. There the authors study a wheel-legged dynamic system with actuated wheels, and can generate agile motions combining walking and driving, including full flight modes. However, this framework enforces the no slip and no skid conditions, which will be frequently violated when driving in the real world.

Another family of wheel-legged systems are the Skaterbots [11], where the authors present a general optimization framework that works for locomoting a wide variety of wheeled-legged systems, including those that have passive wheels only. However in this work the authors explicitly model the wheel's full state, and enforce no-slip constraints when the wheels are in contact with the ground. This does not allow their trajectory optimization to account for inevitable skidding or slip of the wheel.

A third example of dynamic wheel-legged locomotion is Boston Dynamics' wheeled robot Handle [12]. Handle has actuated wheels and can perform a variety of impressive motions quickly and agilely. However since there is no publication to accompany this work, we can not be sure if wheel slipping or skidding are directly accounted for in their framework.

B. *Contribution*

The no slip and no skid conditions are frequently violated throughout any wheeled motion, especially over terrain with low coefficients of friction such as slippery and icy roads. As soon as some skid/slip occurs, the entire model and constraints are already violated a priori before running the aforementioned trajectory optimizations, which do not model such behavior. By explicitly modeling the frictional forces at the contact point and treating the wheel as an ice skate, dynamic and realistic motions that take advantage of this phenomena are discovered naturally by our trajectory optimization framework.

The rest of this paper is organized as follows. Section II describes modeling details for a simplified Robosimian system to be used in the trajectory optimization framework, the latter of which is outlined in detail in Section III. Section IV presents results from using this framework to efficiently skate forward, skid and turn around simultaneously, and skate with different gaits. A brief conclusion and future outlook is given in Section V.

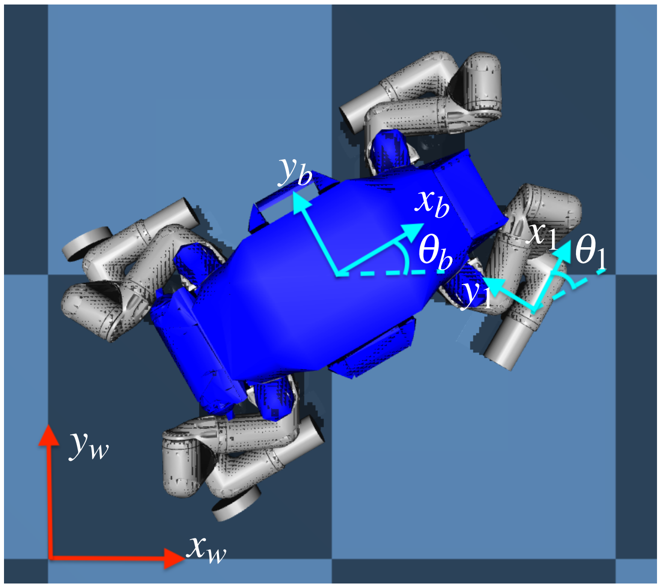

Fig. 2. Robosimian coordinates for optimization in MuJoCo [7], shown here for skate 1, where skate numbering is clockwise. θ is the rotation about the z axis, which points out of the page for the world, body, and skate frame coordinates.

II. Modeling

A. *Kinematics*

We first ignore all internal degrees of freedom for the full Robosimian model and consider a free floating body with floating skate end effectors, whose coordinate frames are as shown in Figure 2. Throughout this work we will assume no pitch or roll from any body.

In the figure, (x_w, y_w, z_w) are the fixed world frame coordinates for the global origin, and thus (x_b, y_b, z_b) are the robot's body center coordinates relative to the fixed world frame. θ_b represents the counterclockwise (CCW) rotation of the body x-axis, relative to the fixed world frame x-axis x_w.

Each skate's coordinates (x_i, y_i, z_i) are relative to the body frame, where $i \in [1,2,3,4]$, starting with the front right skate (shown), and proceeding clockwise (so skate 2 is the rear right, skate 3 is the rear left, skate 4 is the front left, when viewed from above). Each skate also has an additional degree of freedom to rotate about its z-axis (yaw), θ_i, which is relative to the body angle θ_b.

By not modeling the wheel angle position, we are treating the skate more as an "ice"-skate than a "roller"-skate. This decision comes from the fact that no sensing is available at the wheel, as they are latched onto the forearms, and since we are interested in dynamic maneuvers including those involving slipping and skidding, we hypothesize that the wheel's true position angle at a given time step may not be essential to locomotion.

B. *Dynamics*

The dynamics of this simplified system can be derived via a Lagrangian approach. The robot body has a point mass m_b and inertia J_b, and each skate has a point mass m_i and inertia

J_i. The equations of motion for the system can be written in the standard manipulator equation form as:

$$D(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q)+A(q)^T\lambda = B(q)u+F \tag{1}$$

where q are the generalized coordinates, $D(q)$ is the inertial matrix, $C(q,\dot{q})\dot{q}$ denotes centrifugal and Coriolis forces, $G(q)$ captures potentials (gravity), $A(q)^T\lambda$ are constraint forces (where λ are unknown multipliers a priori), $B(q)$ maps control inputs into generalized forces, and F contains non-conservative forces such as friction.

For this system, $q=[x_b,y_b,z_b,\theta_b,x_i,y_i,z_i,\theta_i]^T \in \mathbb{R}^{20}$ where $i \in [1,2,3,4]$. Since each skate's (x_i,y_i,z_i,θ_i) can be individually set with inverse kinematics in the real system, each skates' coordinates are actuated, for 16 total actuators, i.e. $u=[u_{x_i},u_{y_i},u_{z_i},u_{\theta_i}]^T \in \mathbb{R}^{16}$ where $i \in [1,2,3,4]$.

For general wheeled mobile robots, $A(q)^T\lambda$ typically contains constraints ensuring no slip (free rolling in the direction the wheel is pointing) and no skid (no velocity along the wheel's rotation axis perpendicular to the free rolling direction), which come from writing these constraints in Pfaffian form $A(q)\dot{q}=0$. λ can be explicitly solved for by differentiating $A(q)\dot{q}=0$ and substituting in $\ddot{q}$ from Equation 1. We note that our framework can also find trajectories enforcing these constraints when the assumptions of no slip and no skid are valid, but for the rest of this paper we set $A(q)^T\lambda=0$.

III. Trajectory Optimization

This section provides details on formulating the locomotion problem for the system as a trajectory optimization. We discretize the full nonlinear system, and use direct collocation along with backward Euler integration as suggested in [13] to generate the skating motions. More precisely, the problem is formulated as:

$$\begin{aligned}
\text{find} \quad & q,\dot{q},u,F_n,F_{fric} && (2)\\
\text{subject to} \quad & \text{minimize cost } J \\
& \text{State Constraints} \\
& \phi(q,\dot{q},u,F_n)=0 && (3)\\
& \psi(q,\dot{q},u,F_n)\geq 0 && (4)\\
& \text{Dynamics Constraints} \\
& D(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q) \\
& \qquad +A(q)^T\lambda = Bu+F_n+J(q)^T F_{fric} && (5)\\
& \text{Friction Constraints} \\
& \gamma\mathbf{e}+\mathbf{D}^T\mathbf{v}^{k+1}\geq 0, \quad \beta \geq 0 && (6)\\
& \mu F_n-\mathbf{e}^T\beta \geq 0, \quad \gamma \geq 0 && (7)\\
& (\gamma\mathbf{e}+\mathbf{D}^T\mathbf{v}^{k+1})^T\beta = 0 && (8)\\
& (\mu F_n-\mathbf{e}^T\beta)\gamma = 0 && (9)\\
& F_{fric}=\mathbf{D}\beta && (10)\\
& \text{ZMP Constraints} \\
& \text{Gait Constraints}
\end{aligned}$$

where each of the above constraints is detailed below, along with cost function considerations.

A. Objectives

The cost function J at minimum always contains a term penalizing energy use, similar to the modified Cost of Transport (COT) using the integral of the absolute power as in [14]:

$$J_0=\int_0^t \sum_{i=1}^N \sqrt{L+\varepsilon}\; dt \tag{11}$$

where $L=(\dot{q}^TU)^2$, with $U=[\mathbf{0}_{1\text{x}4},u_{x_i},u_{y_i},u_{z_i},u_{\theta_i}]^T \in \mathbb{R}^{20}$ for $i \in [1,2,3,4]$. $\varepsilon = 10^{-6}$ is a regularization term to help smooth the cost function as suggested in [15]. N is the number of sample points for the trajectory.

Note that the above cost function is for minimizing power of moving the end effector in the simplified skating model, and that the real power cost on the full system results from setting the joint positions with inverse kinematics, which may vary largely between the current desired configuration and the previous configuration. We are assuming here that such considerations are on average highly correlated, and that this is still a good measure to account for energy-efficient motion.

Additional cost function terms J_{c_i}, depending on the task, can also include distance measures between desired final and intermediate body positions and/or orientations.

B. State Constraints

The initial states q and $\dot{q}$ are constrained exactly based on the robot's current state. For the rest of the N time points, these are bounded by ranges for placing each skate with inverse kinematics. u is also bounded explicitly, as well as implicitly by $\dot{q}$ ranges. The normal force on each skate F_n is constrained based on the contact schedule as explained in Sec. III-F, and forced to be approximately evenly distributed between all skates currently in contact with the ground.

C. Dynamics Constraints

At each time step k, with $h=\Delta t$ the time step interval, the dynamics are constrained:

$$q_{k+1}=q_k+h\dot{q}_{k+1} \tag{12}$$
$$\dot{q}_{k+1}=\dot{q}_k+h\ddot{q}_{k+1} \tag{13}$$

with

$$\begin{aligned}\ddot{q}_{k+1}=D_{k+1}^{-1}(&Bu_{k+1}+F_{n_{k+1}}+J_{k+1}^TF_{fric_{k+1}}\\ &-C_{k+1}\dot{q}_{k+1}-G_{k+1})\end{aligned} \tag{14}$$

where we write $D(q_{k+1})$ as D_{k+1}, and similar for other terms.

D. Friction as a Linear Complementarity Problem

Determining the friction for each skate in the optimization can be posed as a set of Linear Complementarity Problems (LCPs), and we use the approach detailed by Trinkle and Stewart in [16]. They begin by approximating the circular friction cone by a polyhedral cone, but due to the nature of the skate, we take this one step further and consider a triangle, where friction occurs only perpendicular to the skate's rolling direction. The triangle $\mathscr{F}$ is defined as:

$$\mathscr{F}(\mathbf{q})=\{F_n\mathbf{n}+\mathbf{D}\beta \mid F_n\geq 0, \beta\geq 0, \mathbf{e}^T\beta\leq \mu F_n\} \tag{15}$$

where F_n is the magnitude of the normal contact force, $\mathbf{n}$ is the local unit z-axis representing the six dimensional unit wrench of the normal component of the contact force, the columns of $\mathbf{D}$ are direction vectors that positively span the space of possible generalized friction forces, $\mathbf{e} = [1,1]^T \in \mathbb{R}^2$ as there are two edges of the polyhedral approximation, and $\beta \in \mathbb{R}^2$ is a vector of weights. More concretely, $\mathbf{D} = [y_{s_i}, -y_{s_i}]$, where with $R(\theta)$ representing a 2x2 rotation matrix, $y_{s_i} = R(\theta_b)R(\theta_i)[0,1]^T$ is the unit vector perpendicular to skate i's rolling direction (i.e. skate frame y-axis).

For each contact at each time step, the following inequality constraints are enforced to produce the correct friction forces:

$$\gamma \mathbf{e} + \mathbf{D}^T \mathbf{v}^{k+1} \geq 0, \quad \beta \geq 0 \tag{16}$$

$$\mu F_n - \mathbf{e}^T \beta \geq 0, \quad \gamma \geq 0 \tag{17}$$

with the complementarity conditions:

$$(\gamma \mathbf{e} + \mathbf{D}^T \mathbf{v}^{k+1})^T \beta = 0 \tag{18}$$

$$(\mu F_n - \mathbf{e}^T \beta)\gamma = 0 \tag{19}$$

where γ can be interpreted as a scalar roughly equal to the magnitude of the relative tangential velocity at a contact, and $\mathbf{v}^{k+1}$ is the global 2D planar velocity vector of the contact point at the end of the next time step. From the above constraints the friction F_{fric} at a particular contact can then be recovered with:

$$F_{fric} = \begin{bmatrix} F_{fric_x} \\ F_{fric_y} \end{bmatrix} = \mathbf{D}\beta \tag{20}$$

F_{fric} for each contact is then mapped from global coordinates to generalized coordinates with $J(q)^T$ and input into the manipulator equation.

E. Zero Moment Point (ZMP)

The ZMP is defined as the point on the ground where the sum of all the moments of the active forces is equal to zero. Ensuring the ZMP remains in the support polygon created by the vertices of the skates in contact with the ground is crucial for stability so that the robot will not fall over. It can be calculated as in [17] for the three dimensional skating system with 5 rigid-body links, where we consider the wheel-less “ice” skating version. The x and y components of the ZMP located at point $\boldsymbol{p} = [p_x, p_y, p_z]^T$ are:

$$p_x = \frac{Mgx + p_z \dot{\mathscr{P}}_x - \dot{\mathscr{L}}_y}{Mg + \dot{\mathscr{P}}_z} \tag{21}$$

$$p_y = \frac{Mgy + p_z \dot{\mathscr{P}}_y + \dot{\mathscr{L}}_x}{Mg + \dot{\mathscr{P}}_z} \tag{22}$$

where M is the total mass of the system, g is gravity, (x,y) are the coordinates of the Center of Mass (COM), p_z is the height of the floor, $\mathscr{P}$ is the total linear momentum, and $\mathscr{L}$ is the total angular momentum.

At each time step, depending on the gait and configuration of the robot's skates in contact with the ground, a ZMP constraint is added to the optimization to ensure stability during that time interval. If all four skates are in contact with the ground, the ZMP is constrained to lie within the quadrilateral created from connecting the four skates. If three skates are in contact with the ground, the ZMP is constrained to lie within the triangle connecting these three skates. If only two skates are in contact with the ground, which in this work we assume would only happen with diagonal skates, the ZMP must lie on the line connecting these two skates.

F. Gait

A gait can be provided as input to the optimization in the form of a 4 by N schedule matrix of boolean values indicating which skates are in contact with the ground. This pattern directly determines the desired height z_i of each skate in the form of a spline that depends on the duration(s) a skate should not be in contact with the ground. In turn this pattern also directly sets the desired normal forces evenly for skates remaining on the ground.

G. Implementation Details

The trajectory optimization is implemented in both MATLAB and Python with CasADi [18], with future work entailing moving the code to C++ for additional speed. In particular, solving the NLP can be computationally intensive due to the equality constraints of the LCP. One idea to mitigate this convergence issue is to place the equalities into the cost function as in [11]. However, then we no longer have the guarantee they will be exactly 0, and care must be taken to properly weight the cost function terms to heavily penalize deviations from 0.

We also observe that warm-starting the optimization initial conditions with the solution of the previous run is a significant aid with convergence speed.

Since the trajectories found with the framework are for a simplified skating model ignoring internal degrees of freedom, the solutions must be mapped back to the full system with inverse kinematics. Due to the mismatch in dynamics between the simplified system and full system, we can run the trajectory optimization in a receding horizon fashion at each time step, using the previous solution as initial conditions for the next run. We verify the trajectories found on the full system using PD control on the joint positions with inverse kinematics to track the skate $(x_i, y_i, z_i, \theta_i)$ desired positions in MuJoCo [7].

IV. Results

To show the versatility of our framework and the sorts of dynamic trajectories it can generate, we consider the results from 3 desired tasks:

1) Skating forwards from rest optimally with respect to speed and efficiency
2) A dynamic skidding parking maneuver, showing the main strengths of our approach
3) Hybrid gaits for walk-skating with 2-3 wheels remaining in contact with the ground

The reader is encouraged to watch the accompanying video for more clear visualizations and simulations of the trajectories discussed.

A. *Efficient Skating Forwards*

For the first task we consider energy optimal locomotion with all four wheels in contact with the ground for the duration of the trajectory. The cost function is a weighted sum of the modified COT J_0 from Eq. 11 as well as a term maximizing the final x_b position of the body. We examine a horizon of 2 seconds, starting Robosimian from rest. A locally optimal trajectory for this scenario is shown in Figure 3. We note that for this trajectory, no slipping or skidding is occurring, as the optimization finds the friction forces and torques necessary to propel the Robosimian model forward.

B. *Dynamic Skidding Parking Maneuver*

Next we consider a task of parallel parking in a limited time horizon of 2 seconds. Specifically, we assume Robosimian is moving at 4 (m/s) in the x direction, with otherwise the same initial configuration at the origin. The "parking spot" is located at (4,-1), and we desire Robosimian to be parked there with body angle $\theta_b = \pi$ [rad]. Thus the cost function consists of weighted terms penalizing deviations from these final states for x_b, y_b, and θ_b. Intuitively, we note that with such a small horizon to complete the task, and with no means of braking due to the passive nature of the wheels, we should expect skidding on purpose to slow the system down. As can be seen in Figure 4, the trajectory optimization framework finds precisely such a solution exploiting the frictional forces acting perpendicular to the skate roll directions, and is able to effectively minimize the cost function, arriving at the desired end position.

Depending on the time horizon and energy costs, we might also have expected some sort of several-point turn similar to when turning around a vehicle on a narrow road.

As a slight variation, by changing the cost function to only reward final body position θ_b angle, where we now initialize $\dot{x}_b$ to 8 (m/s), we would anticipate some sort of spinning behavior, or at least the robot to skate around in the narrowest circle it can. Since we do not enforce the non-slip and non-skid conditions, which would limit the subspace of true available motions, the framework finds a trajectory that slips and skids throughout the 2 (s) interval, similar to how a stunt driver might perform a "donut" in a car.

C. *Walking-Rolling Gaits*

Our framework is able to generate trajectories for gaits with any pattern of three wheels in contact with the ground, as well as patterns of any two diagonal wheels in contact with the ground, and alternating combinations of these two types. Ensuring the ZMP is within the support polygon of the remaining contact wheels is crucial for stability of the system during the trajectory. As an example, we show one trajectory of static wheel-walking with skates 2,1,3,4 lifted off the ground, in that order, as in Figure 6, as well as a static trotting gait, with diagonal wheels alternating in the air, as in Figure 7.

The trajectory in Fig. 6 is for a 1.5 second horizon and gait for which each skate is lifted for 0.25 (s), in order for skates 2, 1, 3, 4; shown with '+' symbols in the figure. The ZMP is overlaid with the red dash-dot line, which is constrained to be within the support polygon of skates in contact with the ground. The cost function minimizes energy use, maximizes x_b final state, while attempting to keep final y_b and θ_b at 0.

The trajectory in Fig. 7 is for a 1.5 second horizon and gait where diagonal skates 2 and 4 are in the air from 0.5-0.75 (s) and diagonal skates 1 and 3 are in the air from 1.0-1.25 (s), which is shown by '+' symbols. The cost function minimizes energy while maximizing final x_b position (also minimizing y_b and θ_b offsets from 0). While a skate pair is in the air, the ZMP must lie on the line between the other two skates still on the ground. We note that when a skate is not in contact with the ground, it cannot produce any direct forces to locomote the system, and thus the final x_b position is not as large as when all four skates are on the ground throughout the trajectory as in Section IV-A.

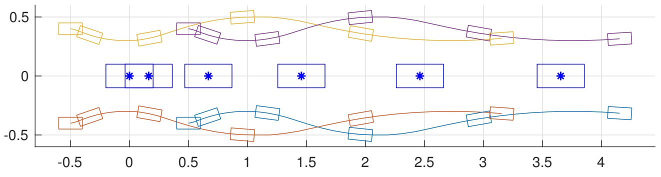

Fig. 3. Trajectory found for skating in a 2 second horizon from at rest, with snapshots every 0.4 (s). The cost function includes a term minimizing energy as well as rewarding motion in the x direction.

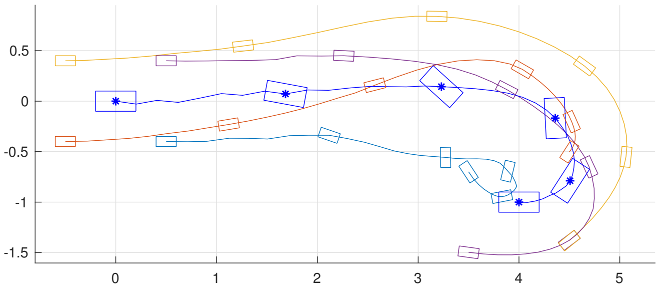

Fig. 4. Trajectory found for a 2 second horizon initialized with 4 (m/s) body velocity in the x direction, with snapshots every 0.4 (s). The cost function is minimizing squared final body orientation θ_b from π, and final body x and y offsets from (4,-1).

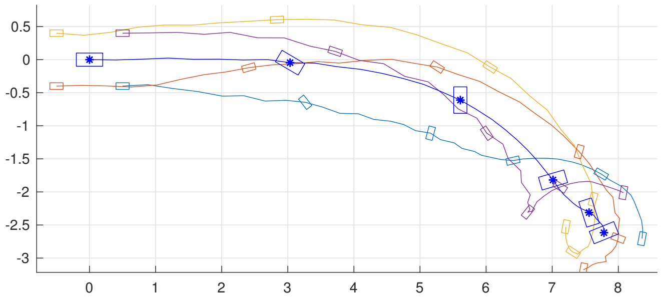

Fig. 5. Trajectory found for a 2 second horizon initialized with 8 (m/s) body velocity in the x direction, with snapshots every 0.4 (s). The cost function is minimizing negative squared final body orientation θ_b (so attempts to find large magnitudes for θ_b).

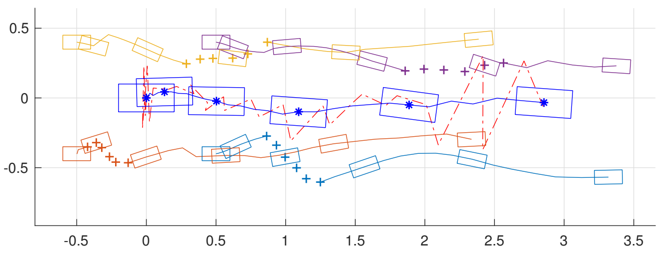

Fig. 6. Trajectory found for a 1.5 second horizon and predetermined rolling-walking gait where skates 2,1,3,4 are in order lifted off the ground each for 0.25 (s). Snapshots are taken every 0.3 (s). The '+' symbols show the sections of the trajectory where that skate is off the ground, and the red dash-dotted line is the location of the ZMP throughout the trajectory. The cost function is minimizing energy and maximizing final x offset from the origin, and minimizing final y_b and θ_b offsets.

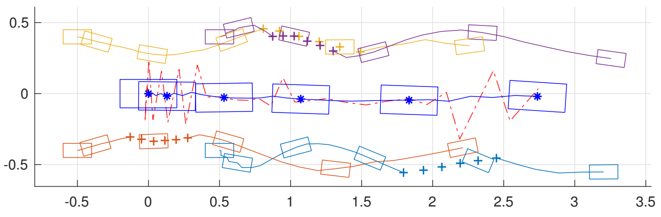

Fig. 7. Trajectory found for a 1.5 second horizon and predetermined static trotting gait where diagonal skates are always in contact with the ground, with snapshots every 0.3 (s). The '+' symbols show the sections of the trajectory where a skate is off the ground, and the red dash-dotted line is the location of the ZMP throughout the trajectory. The cost function is minimizing energy and maximizing final x offset from the origin, and minimizing final y_b and θ_b offsets.

V. Conclusion

In this work we have shown the benefit of not only accounting for, but also optimizing over frictional forces in wheel-legged systems to produce intuitive and dynamic motion primitives. As our system has passive wheels with no sensing, the only way for the system to locomote itself is to reconfigure its internal degrees of freedom to produce the required forces at each of its contact points. By not enforcing explicit non-slip and non-skid conditions that would be frequently violated in the real world, our trajectory optimization framework finds more realistic and dynamic movements.

Future work includes porting the framework to C++ for computational efficiency and speed, as well as developing a better whole-body controller for the full system to accurately track the desired trajectory that is not just a PD controller over the trajectory states. In particular, due to Robosimian's heavy limbs, the trajectories requiring lifting a skate off the ground inadequately simplifies the true dynamics. This mismatch in dynamics also causes drift between the desired trajectory and actual robot states in MuJoCo. Additionally, we would like to increase the modeled dimensions to include pitch and roll for each body, to take advantage of Robosimian's overactuated limbs, which may result in even more dynamic motions. We hypothesize these could include motions such as banking on turns to avoid slip and maintain greater stability at high speeds.

References

[1] B. W. Satzinger, C. Lau, M. Byl, and K. Byl, "Experimental results for dexterous quadruped locomotion planning with Robosimian," in *Proc. 2014 Int. Symp. on Exp. Robotics (ISER)*, 2016, pp. 33–46.

[2] ——, "Tractable locomotion planning for Robosimian," *The International J. of Robotics Research*, vol. 34, no. 13, pp. 1541–1558, 2015.

[3] K. Byl, M. Byl, and B. Satzinger, "Algorithmic optimization of inverse kinematics tables for high degree-of-freedom limbs," in *Proc. ASME Dynamic Systems and Control Conference (DSCC)*, 2014.

[4] P. Hebert, M. Bajracharya, J. Ma, N. Hudson, A. Aydemir, J. Reid, C. Bergh, J. Borders, M. Frost, M. Hagman, J. Leichty, P. Backes, B. Kennedy, K. Karplus, K. Byl, B. Satzinger, K. Shankar, and J. Burdick, "Mobile manipulation and mobility as manipulation–design and algorithms of Robosimian," *Journal of Field Robotics (Special Issue on the DRC)*, vol. 32, no. 2, pp. 255–274, 2015.

[5] S. Karumanchi, K. Edelberg, I. Baldwin, J. Nash, J. Reid, C. Bergh, J. Leichty, K. Carpenter, M. Shekels, M. Gildner, D. Newill-Smith, J. Carlton, J. Koehler, T. Dobreva, M. Frost, P. Hebert, J. Borders, J. Ma, B. Douillard, P. Backes, B. Kennedy, B. Satzinger, C. Lau, K. Byl, K. Shankar, and J. Burdick, "Team Robosimian: Semi-autonomous mobile manipulation at the 2015 DARPA Robotics Challenge finals," *J. of Field Rob.*, vol. 34, no. 2, pp. 305–332, 2017.

[6] G. Bellegarda, K. van Teeffelen, and K. Byl, "Design and evaluation of skating motions for a dexterous quadruped," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[7] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 5026–5033.

[8] G. Bellegarda and K. Byl, "Training in Task Space to Speed Up and Guide Reinforcement Learning," *arXiv e-prints*, p. arXiv:1903.02219, Mar 2019.

[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347

[10] Y. de Viragh, M. Bjelonic, C. D. Bellicoso, F. Jenelten, and M. Hutter, "Trajectory optimization for wheeled-legged quadrupedal robots using linearized zmp constraints," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1633–1640, April 2019.

[11] M. Geilinger, R. Poranne, R. Desai, B. Thomaszewski, and S. Coros, "Skaterbots: Optimization-based design and motion synthesis for robotic creatures with legs and wheels," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 160:1–160:12, Jul. 2018. [Online]. Available: http://doi.acm.org/10.1145/3197517.3201368

[12] Boston Dynamics. (2017, February) Introducing Handle. [Online]. Available: https://www.youtube.com/watch?v=-7xvqQeoA8c

[13] M. Posa and R. Tedrake, "Direct trajectory optimization of rigid body dynamical systems through contact," in *Algorithmic foundations of robotics X*. Springer Tracts in Adv. Robotics, 2013, pp. 527–542.

[14] G. Bellegarda, N. Talele, and K. Byl, "Nonintuitive optima for dynamic locomotion: The Acrollbot," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2018, pp. 3130–3136.

[15] M. Srinivasan, *Why walk and run: energetic costs and energetic optimality in simple mechanics-based models of a bipedal animal*. Cornell University Ithaca, NY, 2006.

[16] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with Coulomb friction," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2000, pp. 162–169. [Online]. Available: https://doi.org/10.1109/ROBOT.2000.844054

[17] S. Kajita and B. Espiau, "Legged robots," in *Springer Handbook of Robotics*. Springer, 2008, pp. 361–389.

[18] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.