ABSTRACT

This paper investigates the problem of rapidly transitioning the pose of a limbed robot while remaining balanced. In particular, we consider motions where rotational accelerations may significantly affect the center of pressure location within a limited base of support. We consider solutions for high-impedance robots with stiff, high-torque actuators that essentially provide accurate, position-control outputs at the joints. We present and compare three methods for generating joint trajectories to achieve fast yet feasible dynamic motions for such systems while maintaining a safety margin for the center of pressure location, toward robust balance. We focus on development of theory and intuition for each method and quantify performance in terms of achievable speed of transition and required joint velocity limits.

1 INTRODUCTION

This work presents and compares three methods for planning motion trajectories for a limbed robot to transition from a horizontally-aligned, quadrupedal stance to an upright, bipedal one. In particular, we are interested in applying results to a new, simian-inspired robot, RoboSimian, shown in Figure 1, which was developed at Jet Propulsion Labs (JPL) for the DARPA Robotics Challenge (DRC). The goal of the DRC is to have a teleoperated robot complete a range of tasks typical of those required in a rescue scenario. In the DRC, robots must operate in a degraded environment that is designed for human kinematics.

For example, a robot should successfully deal with doorways, ladders, and various handheld tools during the Challenge. Also, communication between the robot and an operator is noisy and intermittent, so that increasing the level of autonomy of the robot during tasks is anticipated to be highly advantageous.

FIGURE 1. RoboSimian at the DARPA Robotics Challenge [DRC], with the body in a tilted posture. During legged locomotion, RoboSimian is oriented like a typical quadruped, with the body prone. To perform manipulation and climbing tasks, however, it is advantageous to reorient the body to a more humanoid, upright posture.

With these goals in mind, one particular challenge is to plan a fast and stable transition from a quadrupedal stance, which is practical as a means of remaining balanced while walking...
across rubble or other rough terrain, to an upright posture, in which two limbs are free for manipulation tasks and the reachable workspace for the robot is similar to that of standing human. Within robotics, this is a classic example of kinodynamic planning, in which both kinematics and dynamics must be considered simultaneously for a task [1].

Two key challenges in transitioning RoboSimian to an upright pose reliably are balance constraints and actuator limits. To balance, we want to provide a large safety margin for the center of pressure (CoP) location within the support polygon during motion. On flat ground, the support polygon is defined as the convex hull of all contact points of the robot with the ground, and maintaining the CoP strictly within this region avoids generation of tipping moments to destabilize the base of support.

The dominant actuation constraint for RoboSimian is its joint velocity limit. RoboSimian is designed with high gear ratio motors, so torque and acceleration limits are not nearly as important in motion planning as is the velocity limit. RoboSimian is currently designed for a 3 rad/sec velocity limit, but for JPL’s current prototype, named “Clyde”, we currently enforce a limit of around 1 rad/sec. We are particularly interested in the trade-offs between required joint speed, safety margins for balance, and overall speed of kinodynamically feasible stand-up trajectories in part toward revising velocity and torque limits in future design revisions of the robot.

RoboSimian has a mass of about 100 kg. More relevant to balance, the robot’s center of mass (CoM) is approximately one meter above the ground when upright, while the base of support extends only about +/- 0.12 meters fore and aft of the projection of the CoM onto the ground when standing in the final, upright pose. For robustness, we would like to constrain the CoP to remain within a region with a safety margin of about 0.07 m inside the true support polygon, to account for modeling simplifications and uncertainty. This allows the CoP to vary by only +/- 5 cm (i.e., about 1/20 the CoM height) about the center of the support polygon, making balance quite challenging.

For a particular class of motion, this balance challenge has already been well-addressed by researchers on humanoid walking. Specifically, when there is minimal net change in angular momentum during walking or standing, the dynamics of a humanoid can often be well-approximated by that of a point mass traveling at a constant height. When planning such motions, robotics researchers frequently use the term “zero moment point” (ZMP) rather than CoP, although these terms are simply synonyms. Over ten years ago, Kajita et al. [2] presented a method to plan reference trajectories to track desired ZMP locations by adapting an existing preview control method [3]. Kajita [2] draws an analogy between humanoid robot dynamics and those of a cart moving side to side on a tall, narrow table, using this simplified “cart-table model” for motion planning. The ZMP preview control method is now widely used (and cited) and has also been extended to plan transient (underactuated) motions which do include significant rotation of the robot [4, 5].

A typical assumption in ZMP (aka CoP) planning is that accelerations of masses and inertia within a walking robot are ultimately due to gravity and ground reaction forces; i.e., that these are the only non-negligible external forces on the system. Thus, if one pushes on the world with a force, then \( F = ma = \sum m_i \ddot{x}_i \) ultimately describes the dynamics.

Correspondingly, if one plans a particular set of internal motions of the system that are compatible with a particular trajectory of the ZMP on the ground that is always within the support polygon of the robot, then executing those internal motions must result in the planned ZMP locations over time (again, since \( F = ma \)). This is a key concept in walking and balance research, and it is a basic assumption in our work here.

In this work, the ZMP preview control method previously mentioned [2] is not directly applicable, because the effects of rotating inertias are so significant. Previous work has addressed control for the general problem of a humanoid with significant rotating inertia by modeling the dynamics as an inverted pendulum with a reaction wheel [6, 7], toward remaining upright and rejecting perturbations to the system. Other work has previously investigated transitioning between a quadrupedal and bipedal mode of operation [8–11]. By comparison, this paper presents methods to plan joint trajectories that enable fast yet balanced transitions in robot body pose, which are somewhat unique requirements set by RoboSimian’s unique design and functionality. Finally, we note our work here is related to but distinct from several recent kinodynamic planning problems we have investigated recently for RoboSimian, to develop inverse kinematics tables for fast limb motions [12] and to employ new algorithms to plan collision-free locomotion on rough terrain [13, 14].

2 Problem Statement and Overview

![FIGURE 2. Schematic of RoboSimian just before transitioning from a horizontal to a vertical body pose. At left, a side view illustrates the planar nature of the problem. At right, a 3D view more clearly shows the “feet” (in green) used to provide a support base. Motions are fully actuated (not underactuated), so long as the ZMP is kept strictly within the base of support.](image)

---

1While this is typically a good assumption, strong winds during the DRC in Florida in 2013 provided non-negligible external forces on some humanoids.
Figure 2 gives a schematic of the configuration we consider in this paper for a stand-up maneuver. Each of the four limbs on RoboSimian is identical by design. In our problem statement, we keep the front limbs rigid with the body of the robot and only move the rear legs together synchronously. The lowest limb segments on the rear legs are bent to serve as “feet”, with one pointing forward and one backward, to form a parallelogram as a base of support. We represent the 3D system in Figure 2 with a base of support. As a cart rolls back and forth on a tall table, the GRF (shown in yellow) within which we constrain all planned ZMP locations captures the basic dynamics of an arbitrary collection of masses and inertias within a standing robot. Namely, the model has four DOFs, which map to the following four parameters: $x$, $y$, $\dot{x}$, $\dot{y}$, $J$, $\theta$, $\dot{\theta}$, and $\ddot{\theta}$.

Figure 5 provides a planar model of a multi-link system in motion, to illustrate various points about the relationship between ground reaction force (GRF) and the dynamics of the system. In the figure, there are two point masses, three rotational degrees of freedom DOFs, and a prismatic joint that changes the distance between the two masses. There is also a limited base of support, or foot, shown in blue at the ground. The schematic captures the basic dynamics of an arbitrary collection of masses and inertias within a standing robot. Namely, the model has four DOFs, which map to the following four parameters: $x$ and $y$ coordinate of the center of mass (CoM), inertia $J$ of the system about its CoM, and angle $\theta$ of this inertia. The location, direction and magnitude of the GRF can then be written as a function of $x$, $\dot{x}$, $y$, $\dot{y}$, $J$, $\theta$, and $\ddot{\theta}$.

Three different situations are illustrated in Figure 5. At left, the GRF ($F_y$) vector has a positive component, $F_{gy}$, in $x$, and the acceleration of the CoM is $\ddot{x} = \frac{1}{m} F_{gy}$. Also, the GRF points to the
right of the CoM, so that a counter-clockwise torque is imparted onto the system. By contrast, the GRF in the middle image points exactly at the CoM, imparting no moment to the body. Finally, $F_{gx} = 0$ in the rightmost diagram, and a net clockwise torque on the system is generated. If we visualize the GRF acting at different points along the base of support, it is clear we can achieve a desired \( \dot{x} \) by setting the CoP at any arbitrary location. In particular, it should be obvious that: (1) it is not necessary for the GRF in the \( \dot{x} = 0 \) case to lie directly beneath the CoM, (2) we can plan for trajectories in which the GRF always acts at the center of the support base, by synchronizing the changes in \( x, y, \theta \), and \( J \) appropriately over time, and (3) if the joint between the foot and the first joint is unactuated, the GRF must act at this pivot point. These are key insights toward planning motion with a limited base of support.

The rest of this paper is organized as follows. Section 3 details the three trajectory planning methods we investigate, while Section 4 summarizes results, focusing on the total time required for a transition and the peak joint velocity. Section 5 presents discussion points and topics for future work, and Section 6 summarizes with a brief conclusion.

### 3 Methodologies

We explore three different methods for planning fast motions of a body from some initial pose of interest to a final pose, using only a limited base of support for balance. All three approaches to control for the fully-actuated system are novel, to our knowledge, although the first method builds from recent work in our lab group on methods to transition RoboSimian from bent to upright while using an additional contact point on a wall for extra support [15]. All three methods are described briefly below.

The first method we present linearly interpolates between the specified start and end poses to define intermediate poses and then uses bang-bang control to “play back” this sequence of poses as rapidly as possible while ensuring the ZMP never exits a particular Region of Trust (RoT) within the support polygon, to avoid destabilizing the base of support. Unlike traditional bang-bang control, however, the quantity we saturate here is the location of ZMP on the ground, which depends on accelerations in the system and can correspondingly jump instantaneously between the (bang-bang) limits of the RoT.

The second method first simulates a (fictitious) underactuated version of the system, using a linear quadratic regulator (LQR) to drive the system from its initial pose to the upright equilibrium position, and then uses the resulting joint angles of the underactuated system over time as reference trajectories for the fully-actuated system model. This automatically results in a solution in which the ZMP/CoP will be at the first pivot point, which (by symmetry) is exactly at the center of the support polygon, providing the maximal safety margin possible for balance.

The third method essentially combines the other two. Bang-bang control is again used here, to allow the ZMP to vary within the RoT, but the reference poses that are executed are those resulting from the LQR simulation, rather than a simple linear interpolation between the start and end configurations. Intuitively, this provides poses that are in some sense more “dynamically compatible” with a centrally-located ZMP location while also exploiting the additional benefits of shifting the GRF location on the ground to speed up the motion. Our results show that all three methods work to produce feasible trajectories, while the last method arguably provides the best overall trade-off in speed and stability.

#### 3.1 Bang-bang Trajectory Playback

The first method we explore employs bang-bang control on a trajectory that linearly interpolates all joint angles from a given start pose to a given end pose. That is, we first define each joint angle as a function of a dummy variable, \( \tau \), and we next determine \( \tau(t) \) such that the entire trajectory is played back as quickly as feasible, given constraints on the allowed CoP location.

This problem formulation is closely related to the classic minimum-time control problem for a mass, given limits in the magnitude of the control input (force), \( u \), that can be applied.

\[
\dot{x} = u, \quad \text{s.t.} \quad u \in [-1, 1]
\]

(1)

In a more general optimization formulation, one minimizes a cost that depends both on the final state, \( x_{\text{final}} \), and the path by which the system gets there. In our case, we wish only to minimize the time to arrive at \( x_{\text{final}} \), so our cost is \( J = T_{\text{final}} \). In the framework of optimal control, this is described by the well-known general and specific cost definitions below:

\[
J[x,u] = h(x_{\text{final}}) + \int_0^{T_{\text{final}}} g(x,u) dt, \quad \text{s.t.} \quad h = 0, \quad g(x,u) = 1,
\]

(2)

The well-known solution is bang-bang control: the mass is either accelerated or decelerated by using an extremum value of \( u \) at all times during the optimal trajectory. This optimal control solution can be rapidly shown via Calculus of Variations [16]. First, we construct the Hamiltonian, \( H \), for the system. From Pontryagin’s maximum principle, optimal control maximizes \( H \).

\[
H(x,u,p) = <p, \dot{x}> - g(x,u) = p_1 \dot{x}_1 + p_2 \dot{x}_2 - g(x,u)
\]

(3)
where $p_1$ and $p_2$ are co-states to $x_1$ (position) and $x_2 = \dot{x}_1$ (velocity), the states of the dynamic system. For the optimal control solution, being at the minimal cost solution requires that the time derivatives of the elements in the adjoint vector, $p$, equal the negative of the partial derivatives of the Hamiltonian, $H$, with respect to the elements in the state vector, $x$, i.e., $p_1^* = \frac{-2H}{\dot{x}_1} = 0$, and $p_2^* = \frac{-2H}{\dot{x}_2} = -p_1^*$. This yields the solutions: $p_1^* = c_1$, and $p_2 = -c_1 + c_2$, so:

$$H(x,u,p) = c_1 \dot{x}_1 + (c_2 - c_1 t) u - 1$$

which is maximized when $u$ is at either a maximal or minimal value, depending on the sign of “$c_2 - c_1 t$”. In other words, the solution for the fastest trajectory to $x_{final}$ involves starting with either the min or max possible choice for $u$ and switching at most once (if at all) before arriving at the goal state.

If we now apply this approach to move the cart in Figure 4 along the table surface from some initial state, $(x_i, \dot{x}_i)$, to some final state, $(x_f, \dot{x}_f)$, a key modification is that “$u$” (analogous to $F_{gs}$ here) is no longer limited to a constant range of values. Instead, the set of possible $F_{gs}$ values depends on the current state of the robot. Here, $u = F_{gs}$ is proportional to the $x$ distance between the CoM and the ZMP on the ground: $F_{gs} \propto x_{com} = x_{zmp}$. As a result, there are now combinations of initial and final state for which control solutions do not exist, which is not true of the system described in Eqn 1. However, solutions will exist for all cases in which a) $\dot{x}_i = \dot{x}_f = 0$, and b) both $x_i$ and $x_f$ are within the bounds of the support polygon in $x$.

For simplicity we always define $\tau$ as going from 0 (at the start pose) to 1 (for the end pose) in magnitude. If we prefer the form of Equation 6 over 5, we can first define trajectories in terms of the joint angles, $q_1(\tau)$, $q_2(\tau)$, and $q_3(\tau)$ and then derive $x_i(\tau)$ and $y_i(\tau)$ for each of 6 representative point mass locations. We are then left with the problem of determining how quickly it is feasible to “play back” the variable $\tau$ over time. From Equation 5,

$$0 = M \left( (x - x_{zmp}) \left( \ddot{y}_i + g \right) - y_i \dot{x}_i \right) + 2I\ddot{\theta} + J\ddot{\theta}$$

For the three-link system shown in Figure 3, each link inertia can be represented by two point masses, so that $N = 6$ in Equation 6.
we can rewrite a single equation of motion for the system, which describes how the dynamics by which our “clocking” variable $\tau$ evolve over time:

$$
[a_1 (x - x_{cmp}) + a_2] \ddot{\tau} + [b_1 (x - x_{cmp}) + b_2] \dot{\tau} + c \dot{\tau}^2 + [g (x - x_{cmp})] = 0
$$

where:

$$
a_1 = y' \\
b_1 = y'' \\
a_2 = -yx' + J\theta' \\
b_2 = -yx'' + J\theta'' \\
c = 2J \theta' \\
g = 9.81 [m/s^2] \ (i.e., \ gravity)
$$

Here, the $'$ operator indicates a derivative with respect to $\tau$, i.e., $\theta' = d\theta/d\tau$, $\theta'' = d^2\theta/d\tau^2$. Equations 8 and 9 are derived by plugging the relationships given below in Equation 10 into the force balance in Equation 5.

$$
\dot{\theta} = \theta' \dot{\tau} \\
f = J \dot{\tau} \\
\ddot{y} = y'' \dot{\tau}^2 + y' \ddot{\tau} \\
\ddot{x} = \chi' \dot{\tau}^2 + \chi' \ddot{\tau} \\
\ddot{\theta} = \theta'' \dot{\tau}^2 + \theta' \ddot{\tau}
$$

To solve numerically for the optimal trajectory, we generate a discrete-time (DT) approximation for the system dynamics by selecting a small sampling time, $T$. We start with two initial conditions, one in which $\tau = 0$ and $\dot{\tau} = 0$, and the other in which $\tau = 1$ and $\dot{\tau} = 0$. Simulating forward from $\tau = 0$ (and backward, from $\tau = 1$), we solve for the dynamics assuming $\dot{\tau}$ remains constant during each time step, $T$. The value of $\dot{\tau}$ selected at each time step is the maximum one feasible (or minimal one, when simulating backward from $\tau = 1$) given the limited set of locations for $x_{cmp}$. This value is estimated numerically by calculating the required $x_{cmp}$ for a discrete set of possible values for $\tau$ and interpolating to find a value of $\dot{\tau}$ that provides less than one micron of error in $x_{cmp}$. As long as the ZMP does not exit the support polygon, new value of $\tau$ can then be calculated (via simple calculus) as:

$$
\ddot{\tau}_n = u_n \\
\ddot{\tau}_{n+1} = \ddot{\tau}_n + T u_n \\
\tau_{n+1} = \tau_n + T \ddot{\tau}_n + \frac{1}{2} T^2 u_n
$$

The effect of the DT approximation is that the true $x_{cmp}$ value will drift away from its original value slightly during the time interval $T$. Recalling, however, that we restrict the ZMP to a Region of Trust within the true support polygon, this error is not significant.

For the general trajectory-optimization case, one final issue worth noting is that the solution to the ZMP bang-bang solution for an arbitrary set of joint trajectories may in general require multiple switching points. However, such cases appear to be only pathological exceptions in our experience. See Section 5 for further discussion on this issue.

### 3.2 LQR-Based Method, via Underactuated Solutions

Our second, LQR-based method takes an entirely different approach to the problem. Here, we restrict the ZMP to the exact center of the base of support, to maximize the safety margin within the support polygon. For a point-mass system, this restriction does not allow the system to start and stop in $x$, i.e., considering Figure 4 once again, a point contact requires either $x_{com} = x_{cmp}$ for all time (an unstable fixed point) or that $x_{com}$ varies monotonically for all time (so the system can never stop, once it begins moving). Therefore, this approach is not feasible unless we consider systems with distributed inertia(s). Even for such systems, finding (fully-actuated) joint trajectories over time that obey this safety constraint is potentially quite challenging.

We address this problem by first considering a related case in which the first joint is an unactuated point contact with the ground. The resulting system then becomes an underactuated, $n$-link serial chain, in which the first joint is passive while the remaining $n - 1$ links are actuated. This type of underactuated system is well-studied as a classic nonlinear control problem [17], and the two-link version is commonly called the acrobot [18].

Nonlinear equations of motion for such a multi-link system can be generated using the Lagrangian method and written in the form:

$$
M(q) \ddot{q} + C(q, \dot{q}) + G(q) = T(u)
$$

where $M$ is a configuration-dependent inertia matrix, $C$ is a vector of torques due to Coriolis and centripetal effects, $G$ contains torques due to gravity, and $T$ gives the actuator input contributions for the set of equations represented by 12.

Dynamics and control for the three-link version have been studied in several works, and full equations of motion for this system can be found in [19] and [20].

For this underactuated system, the GRF must obviously always act exactly at this point contact. If we then use the resulting joint trajectories over time as reference trajectories for our fully-actuated system, the GRF must also remain at the same point over time, since there will be a unique solution for the force required to achieve the desired positions, velocities, and accelerations over time. In other words, since $F = ma$, uniquely, at each
moment in time, where the only external force on the system is the GRF, and our control enforces the righthand side, in the limit as references trajectories are accurately followed.

The advantage of this second method is that it increases the safety margin for the ZMP within the base of support, while the advantages of the first method is that it exploits a reduction in safety toward improving speed.

3.3 Combining LQR and Bang-Bang

Finally, we use the LQR solution to define a set of waypoints as a function of our dummy variable, $\tau$. We can then potentially vary the ZMP within the RoT in the support polygon to speed up overall motion. We apply the same bang-bang control algorithm described in Section 3.1 to these new joint references, toward determining a faster path in phase space to move from the start pose ($\tau = 0$) to the end pose (defined at $\tau = 1$).

The intuition for doing this is that since the LQR-based solution can be performed without moving the ZMP at all, it may result in a more natural coupling between the translational and rotational movement of the joint inertias (than a linear interpolation does) that can in turn exploit the limited base of support to move even faster than either of the previous approaches described.

Indeed, this approach does increase motion speed significantly when compared with both the original bang-bang algorithm (linearly interpolating between the start and end pose to define $q_1(\tau)$, $q_2(\tau)$ and $q_3(\tau)$) and the LQR-based trajectory method. While this faster trajectory is dynamically feasible for the limited base of support, however, it does require higher joint velocities, as well. Detailed results are given in the next section.

4 Results

Below are more detailed results for each of the three methods described in Section 3, above.

4.1 Bang-bang Results, on Linearly-Interpolated Solutions

Figure 7 shows a phase plot for evolution of position versus velocity for the dummy variable $\tau$. Recall that for this method, the pose of the robot system for any $0 \leq \tau \leq 1$ is a linear interpolation between the start and end poses. For our three-link system, this means $q_i(\tau) = \tau q_i(1) + (1 - \tau) q_i(0)$ for $i \in \{1, 2, 3\}$.

In Fig. 7, the forward bang-bang solution, starting at $\tau = 0$, starts at ($\tau = 0$, $\dot{\tau} = 0$) and is shown in solid blue, while the solution starting at $\tau = 1$ is shown as a red dashed line. Just as in Figure 6, all bang-bang trajectories, corresponding to piecewise-constant $\tau_{mp}$ locations, are symmetric about the x axis of the phase plot. Thus, we reflect the red dashed line across the x axis to generate the fastest possible trajectory approaching $\tau = 1$, given all joint angles over time are controlled via feedback to follow their desired $q_i(\tau)$ values. As in the original bang-bang formulation, the optimal trajectory is formed by using the intersection between the forward and backward trajectories as the switching point for control.

Quantitatively, the overall time required to execute the full stand-up motion using this method is $T_{final} = 1.72$ seconds. Qualitatively, the trajectory on the phase plot for the forward solution “flattens out” significantly for this solution, compared with the point-mass solution in Figure 6, in which $\tau$ steadily increases up until the switching point. With $\tau$ remaining low in magnitude, there is hope the overall speed can be improved significantly by using a different sequence of intermediate poses during stand-up.

Figure 8 shows the three joint trajectories, $q_i(\tau)$, and joint velocities, $\dot{q}_i$ over time, again showing how velocities approach near-constant values over the middle portion of the trajectory.
Also, the peak joint velocity magnitude required for the motion is about 1 rad/sec.

### 4.2 LQR-Based Results

Two important open questions regarding the linear quadratic regulator (LQR) implementation are (1) whether one can simply use a regulator, rather than a reference trajectory, to allow the system to transition from the initial bent-over pose to its upright equilibrium orientation at all, and (2) if the resulting solution avoids collision with terrain. Interestingly, the answer is affirmative in both cases. Figure 9 illustrates a sequence of intermediate poses during LQR stand-up. Although the lowest limb comes close to the ground, it remains strictly above the ground plane at all times. Note that the system drops further away from the final upright configuration in the early portion of the trajectory (e.g., “pose 2” in Fig. 9), much as a weight-lifter naturally drops her center of mass before leveraging up a heavy load.

A second issue in the LQR implementation is how best to select the (quadratic) cost matrices, $Q$ (penalizing squared errors in states) and $R$ (penalizing squared actuator effort). We explore a range of values for $Q$ and $R$ to determine the trade-off between settling time to upright end pose and required peak actuator velocity. Figure 10 plots these two criteria against one another, to illustrate this. There is a distinct “knee” in the lower left of each scatter plot, where one can achieve an intuitively good compromise between settling time and required joint speed. For this work, we select a location near the knee of the curves, requiring peak actuator velocity of about 2 rad/s. This results in a 5% settling time of 2.25 seconds and a 1% settling time of about 3.14 seconds. This corresponds to diagonal $Q$ and $R$ matrices with values of 1 for all $Q$ values penalizing position errors, 0.1 for $Q$ weights on all joint velocity element, and $1e-6$ for all three actuator penalties within $R$.

![FIGURE 9. Animation of the LQR solution, transitioning from prone to upright. Several snapshots of intermediate poses are shown, numbered from 1 (initial pose) through 6 (fully upright end state).](image)

![FIGURE 10. Data comparing setting time versus peak required joint velocity, for the same initial and final pose but given a variety of relative cost coefficients in the $Q$ and $R$ matrices used to derive LQR gains.](image)

![FIGURE 11. Phase portraits for each of the 3 joints of a typical LQR solution, showing position versus velocity.](image)

Figure 11 shows the phase portraits for each of the three joints for the LQR solution, which initially jump rapidly onto smooth trajectories that are close to being critically damped. Finally, Figure 12 shows the actual $q_i(t)$ joint trajectories over time. Note that the peak velocity jumps very rapidly to its value of about 2 rad/sec and that velocities do not “flatten out”, as in the bang-bang solution for linearly interpolated poses, but they also do not “sharply peak” as in the point-mass bang-bang solution.

### 4.3 Bang-Bang Playback of LQR Trajectories

Our last method uses the joint trajectories obtained in the previous section (shown in Figure 12) to set $q_i(\tau)$ poses and then executes this sequence of poses via bang-bang control on the ZMP. Figure 13 show the phase portrait ($\tau$ vs $\dot{\tau}$) for this method. We note it demonstrates the same peak-topped signature characteristic of the point-mass bang-bang example, and that the feasible transition time is much faster now, taking only 0.49 sec.
FIGURE 12. LQR solution for underactuated 3-link model. Because there is no torque applied at the first joint in the simulation, the ground reaction force is automatically constrained to act only at the point contact on the ground, yielding the maximum safety margin for the ZMP.

compared with the 3.1 sec 1% settling time when using the original LQR trajectory as a reference without the bang-bang implementation (i.e., 1/6 total time required). As the lower subplot in Fig. 14 shows however, the peak joint velocity magnitude is now 8 rad/sec, which is a factor of 4 larger than for the LQR case.

FIGURE 13. Phase portrait for dummy variable $\tau$.

5 Discussion

We note first that our bang-bang playback of the LQR solution can be recalculated to use an even smaller range of allowable ZMP values, which would allow one to adjust peak velocity and time for stand-up. Also, one could easily incorporate a strict joint velocity limit into the algorithm, in future work.

Also, it is possible to construct trajectories that require multiple switching points. In other words, the ZMP may have to jump repeatedly from one edge of the support polygon to the other, rather than switching only once, as in the classic bang-bang problem. A trivial example is a cart that repeatedly changes direction (left vs right) as $\tau$ evolves, but other cases can be constructed in which $x(\tau)$ and $y(\tau)$ evolve monotonically but the multiple switches are required due to rotational effects (such as a flywheel mounted on the robot reversing directions multiple times). In practice, the trajectories we have analyzed have not displayed this problem, but investigating necessary and sufficient conditions for a trajectory to require exactly one switching point is a topic for future work.

As a final comment, note that the limit in overall speed may, in extreme cases, be set by the fact that the ground contact provides a unilateral constraint: the robot can only push (not pull) against the ground plane: $\ddot{y} < 9.81 \text{ [m/s}^2\text{]}$. This unilateral constraint was never close to being violated in this work, but we wish to remind the reader that it should be considered if these methods are used.

6 Conclusion

In this paper, we present three different methods for producing joint trajectories to reorient body pose of a legged robot such that resulting motions are both fast and dynamically stable. We first demonstrate a method to apply bang-bang control to a class of problems in which the external force applied to the system due to the ground reaction force has limits that are a function of the system state. This method first defines all actuated DOF’s as function of a single dummy variable, $\tau$, and then determines the minimum time to “playback” $\tau(t)$, under ZMP balance constraints. A second method designs reference trajectories over time for the fully-actuated three-link system by first simulating a (fictitious) underactuated three-link system, thus automatically generating solutions in which the GRF must act solely
at the center of the support polygon on the ground. Finally, we demonstrate a method to blend these two approaches by using the LQR trajectory to define $q_i(\tau)$ values, then determining the time-optimal $\tau(t)$ via bang-bang control.

For our case example, the last method provides the fastest time to stand-up compatible with ZMP constraints, but (not surprisingly) it also requires correspondingly higher peak joint velocity. Specifically, the original bang-bang method on poses that are linearly interpolated from start to end, as described in Sec. 3.1 and 4.1, requires a finite, total time of $T_{f_{\text{final}}} = 1.7$ seconds. The LQR method (Sec. 3.2 and 4.2) approaches an upright pose exponentially (rather than in finite time), with a (slower) 1% settling time of $T_{1\%} \approx 3.1$ seconds (i.e., to within 1 degree of upright). However, the LQR-generated poses can safely be played back as described in Sec. 3.3 and 4.3 in a finite transition time of $T_{f_{\text{final}}} = 0.49$ seconds. This work provides a basis for future development of more general optimization algorithms, for example, to iteratively improve sequences of poses for a stand-up motion optimizing a cost function that includes both trajectory time and required actuator velocity.

ACKNOWLEDGMENT

This work is supported in part by the DARPA Robotics Challenge. We wish to thank the entire RoboSimian team at JPL, Caltech, Stanford and UCSB for their efforts.

REFERENCES