Feasibility and Optimization of Fast Quadruped Walking with One- Versus Two-at-a-Time Swing Leg Motions for RoboSimian

Peter Ha and Katie Byl

***Abstract*— This paper presents two planning methods for generating fast walking gaits for the quadruped robot RoboSimian and analyzes their feasibility and performance as a function of joint velocity limits. One approach uses a two-at-a-time swing leg motion gait, generated using preview control of the zero moment point (ZMP) on a narrow, double-support base of support. The second method uses a one-at-a-time swing leg crawl gait. We employ and compare both value iteration and gradient descent methods to generate the motions of the swing leg through free space for the crawl gait. For this gait, our optimization strategies exploit redundant degrees of freedom in the limb, which proves to increase walking speed significantly. By contrast, the ZMP-based gait requires nearly symmetric motions of the (fixed) stance legs and (free) swing legs with respect to the body, resulting in limited ability to improve the joint trajectories of the limbs through clever planning, and so an inverse kinematics (IK) table solution is used for all limb kinematics here. Our ZMP-based planning is only feasible if joint velocity limits are sufficiently large. Also, both the overall step size and speed of this gait vary nonlinearly as velocity limits increase. Lastly, comparing the two methods, we show that walking speed is faster using two-at-a-time swing leg motions when the joint velocity limit is between 1.62 and 3.62 rad/sec and that the crawl gait is optimal otherwise.**

I. INTRODUCTION

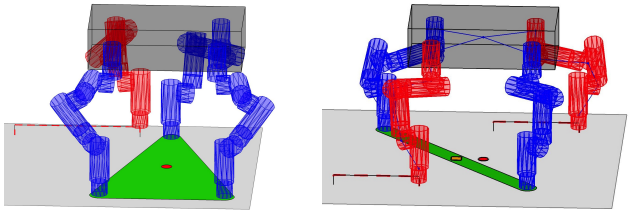
This work studies the problem of increasing gait speed for a quadruped robot with dexterous limbs and with actuators designed to favor high torque and (relatively) low speed.

The feasibility of dynamic gaits for legged robots inherently depends upon joint actuator characteristics. Obviously, a fast running or jumping robot must produce sufficient ground contact forces, in turn requiring sufficient joint torque. However, sufficient *velocity* limits are also required. More specifically, a robot with acutators that are "too slow" will never become airborne, no matter what static force it can produce, and is thereby destined to walk, rather than run.

One solution to improve speed for quadruped walking robots is to move two legs at once, instead of moving legs one-at-a-time. This requires using only two legs in stance ("double support"), as illustrated in Figure 1. A ZMP preview control method has successfully been adapted to execute such gaits with high repeatability, as demonstrated by Byl et al. [6] in work with the point-footed LittleDog quadruped during DARPA's Learning Locomotion program. Figure 2 shows LittleDog mid-stance in such a double-support gait, with only two feet in contact with the ground.

*This work was supported by DARPA.

P. Ha and K. Byl are with the Robotics Laboratory, University of California, Santa Barbara, CA 93106, USA {petersha90, katiebyl}@gmail.com



(a) (b)

Fig. 1: Two- vs. One-at-a-time swing leg gait. The blue limbs indicate the stance legs and the red limbs indicate the swing legs. Also, the green polygon represents the support polygon and the red dot on the ground shows the COM while the orange square shows the ZMP.

In the present work, we seek to develop fast walking gaits for a new quadruped, RoboSimian, which was developed by Jet Propulsion Labs (JPL) to participate in the DARPA Robotics Challenge (DRC). We explore two general questions in planning fast walking gaits for RoboSimian. First, we quantify the required joint velocities to enable use of our preview control for two-at-a-time swing leg motions. Second, we explore the extent to which additional degrees of freedom in the limbs enable faster walking in a traditional crawl gait, in which swing legs move one-at-a-time. In particular, LittleDog has only three actuators per limb, while RoboSimian's limbs have seven degrees of freedom (DOFs).

In a crawl gait, each swing leg moves in free space while the body remains stationary or moves relatively slowly, compared with the end effector. Planning for RoboSimian's high-DOF limb in free space allows for a wide range of possible trajectories to move the swing leg to a new foothold. With more DOFs, we hypothesize there will be large variability in the time required to complete any given path, meaning there is potentially an equally large opportunity to improve performance significantly through careful planning.

By contrast, our double-support gait requires that the two limbs in contact with the ground during a step move backward with respect to the body as the two swing legs simultaneously move forward. Also, the stance legs must remain relatively upright in orientation and on a fixed position on the ground, which limits the opportunities to improve overall speed by improving kinematic planning. More specifically, only the yaw of the stance leg varies as the body moves. This is possible because the last joint of RoboSimian allows the foot to remain fixed while the rest of the leg rotates about the vertical axis, and we use an

existing (heuristic) in-house kinematic law to set this yaw as a function of end effector position.

Additionally, the (forward) motion of the body during our ZMP-based, double-support gait is prescribed by the inverse dynamics solution generated by our implementation of the preview control approach, as previously developed for and tested on LittleDog [5], [6]. This solution in turn is a function of the height of the center of mass and the step length, so that both the required joint velocities and overall gait speed can be calculated as functions of step length, enabling us to analyze required joint velocity as a function of gait speed.

In our work, we analyze the speed of feasible ZMP-based walking as a function of actuator velocity limits, and we employ two approaches toward optimizing walking speed for a more traditional crawl gait. One search limits the DOFs of the end effector to enable use of a global search, via value iteration. The second search uses a local gradient search over a larger set of DOFs, and we explore the possibility of using the value iteration solution as an appropriate local starting point for our local search.

We find that allowing for additional degrees of freedom improves walking speed so that the gradient search significantly outperforms the reduced-dimension value iteration result. Also, our ZMP-based gait is not feasible below a certain actuator limit and only out-performs the static crawl gait over with a particular "window" of actuator velocity limits.

A. Fast Walking

High speed dynamic robots such as Boston Dynamic's Cheetah and WildCat [1] rely on constant movement when walking / running to ensure stability. Depending on the capabilities and design of a robot, there are many different approaches that have been used for maintaining stability while walking quickly. The most popular approaches for planning humanoid locomotion focus on regulation of the center of pressure, referred to as the so-called zero moment point (ZMP) which was introduced by Miomir Vukobratović in 1968. The ZMP represents the point on the ground where the horizontal moments become zero. In traditional ZMP methods, motions for a robot are planned such that the resulting ZMP lies strictly within the convex hull of the robot's contacts with the ground, called the support polygon, thereby ensuring there are no toppling moments about any edge of the support polygon for the planned motion. This approach is used widely for bipedal robots [2], [3], [4]. For a legged robot, the ZMP trajectory can be designed to jump instantaneously from being under one foot to the other foot. This is possible because the ZMP is a function of both the positions and accelerations of the distributed masses in the robot, and because we assume we can instantaneously apply torques to set accelerations at the joints. For quadruped robots, ZMP planning has also been used to perform double-support maneuvers such as lunging and "trot-walking" [5], [6] which requires two limbs swinging at once (Figure 1(b)). However, ZMP planning for double-support gaits requires sufficient actuator characteristics. In particular, if joint velocity limits are prohibitively low, performing such dynamics motions will no longer be feasible, and other approaches must be use to improve locomotion speed.

Fig. 2: Littledog in a double-support phase (top) and RoboSimian at the DARPA Robotics Challenge [DRC] (bottom). The two-at-a-time gaits we analyze for RoboSimian are based on proven techniques [6] employed to achieve fast walking in the point-footed LittleDog quadruped. To visualize relative scaling, note that the center of mass of LittleDog is approximately 15 cm above the ground during walking, while RoboSimian's is roughly 57 cm high in our work here.

In such cases, a more practical method of increasing walking speed may be to optimize the trajectory while moving only on limb at a time Figure 1(a). One method is to use third-order spline interpolation for the swing leg trajectory [7], [8]. As mentioned, we explore the performance of value iteration and gradient descent methods in maximizing walking speed for single limb motions in this paper.

B. RoboSimian

Robosimian is a quadruped robot developed at Jet Propulsion Laboratory (JPL) with design support from Stanford. The robot is designed for disaster response and is participating in the Defense Advanced Research Projects Agency (DARPA) Robotics Challenge (DRC). Figure 2 shows a picture of RoboSimian. We use in-house software developed at the University of California Santa Barbara Robotics Lab to solve for the inverse kinematics (IK, as in Fig. 3) used for the analysis presented here.

Since the limbs of RoboSimian have 7-Degrees of Freedom (DOF), there are in general an infinite number of IK solutions for each feasible end effector position. The

7th joint, at the distal-most end of the limbs in Figure 3, is a wrist that simply allows the lower part of a stance leg/ankle to rotate while the actual foot remains fixed on the ground. This leaves 6 joint angles to set the 6-DOF position and orientation of the last rigid limb segment, shown in green. Even for this 6-DOF reduced model, there can still be multiple redundant solutions. These redundant IK solutions can be divided into 8 different families depending on the configuration of the joint angles. At a high level, the kinematic solutions for RoboSimian involve three geometric choices, akin to deciding whether to bend an elbow forward or backward. Each of these solutions has its own feasible workspace. To achieve smooth limb trajectories, the robot must either remain within a particular kinematic family or intentionally pass through a singularity, and even when remaining in a particular family, it is possible for discrete jumps in joint angles, so testing of IK trajectories for smoothness is always a necessity.

For this paper, we study only the 3rd and 8th IK solution families (Figure 3), because these two of the eight IK solutions provided us with the largest continuous workspace for a near-vertical end effector to contact the ground while avoiding self-collisions with the robot. Figure 3 illustrates the difference between resulting leg configurations for each family for one, particular end effector pose.

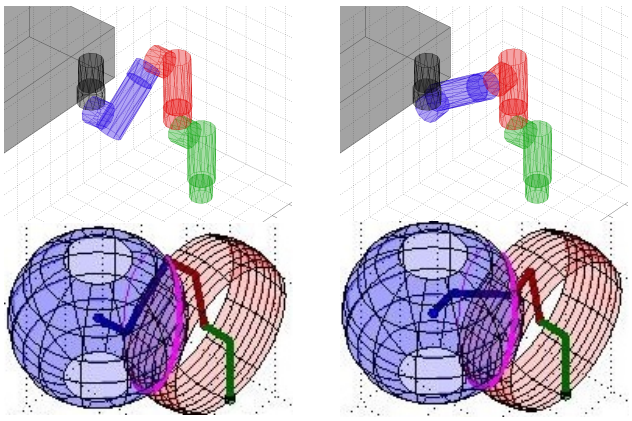


Fig. 3: 3^{rd} IK solution family (left) and 8^{th} IK solution family (right). For a fixed, 6DOF position and orientation of the most distal (green) limb segments, there exist up to 8 kinematic solutions. Through extensive IK testing, these two solutions have proven to yield the largest collision-free and feasible continuous workspace for rough terrain walking.

The partial blue sphere shows the reachable workspace of the point where the blue and red limb segments meet, assuming the base of the kinematic chain for the robot is fixed at the center of the blue sphere. The partial red sphere shows the reachable workspace of this same point in the kinematic chain when the end effector location and orientation (shown in green) are fixed. The actual location of the end of the second link must lie somewhere on the magenta arc(s) where these two spherical surfaces intersect, and it must also result in an orthogonal intersection between the blue and red links that meet at this point.

1) Limitations on RoboSimian: The most significant motion planning limitation with RoboSimian is that it currently has a velocity limit at each joint of 1 rad/sec, which limits the ability of the robot to perform dynamic motions. This limit can potentially be modified in future hardware revisions, however, which in part motivates this entire work. The joint velocity limits affect both the speed with which the body can be moved while the stance feet are planted on the ground and the speed with which the swing leg can be advanced. For a full gait cycle, all four legs and the body must advance one full step length and all four legs must move up and down a desired step clearance height. In the case of the one-at-a-time crawl gait, leg swing motions account for more than 80% of the gait cycle time. Thus, the swing leg motions dominate the forward speed of the robot during a steady, repeating gate. Also, since there are infinitely many continuous, non-colliding paths for the swing limb from start to end pose, it is not obvious which path produces the fastest motion. This motivates the use of a search algorithm to find the optimal path for the limb to move.

II. PLANNING METHODS

A. Assumptions

For this paper, we assume that the robot walks directly forward on flat ground with no obstacles. At the beginning (end) of each swing leg motion, we require the robot to lift (lower) a limb straight up either 10 cm or 6 cm, to give a safety margin for obstacle clearance and terrain uncertainty. During swing, the end effector is not allowed to dip below this prescribed terrain standoff height. Also, for two-at-a-time walking, the end effector of each stance limb has no pitch or roll, so that it remains perpendicular to the ground at all times, to avoid resting on the edge of the foot. For the one-at-a-time crawl gait, we calculate a single end effector trajectory using the front right leg and its resulting joint angle motions. The full gait involves symmetric motions (mirrored in space and/or time) for the other 3 limbs. Stance leg motions to propel the body come from an in-house IK table. Note that any local limb position throughout this paper is defined for the front right limb's end effector position with the center of body (COB) at the origin.

B. ZMP Based

Using a planning approach for ZMP by Shuuji Kajita, et al. [9] based on preview control [10], we were able to calculate a trajectory for the center of mass (COM) for a single mass at a constant height, given a desired ZMP trajectory. For a point-mass model, the ground reaction force must always point directly toward the COM, which greatly simplifies the calculation of ZMP location on the ground. The ZMP of a cart and table model is found by using the dynamics:

$$\ddot{x} = \frac{g}{z}(x_{com} - x_{zmp}) \tag{1}$$

where, x_{com} is the displacement of the COM in the x direction, g is gravity, z is the constant height of the COM, and x_{zmp} is the x location of the ZMP on the ground. The

preview control algorithm from Kajita solves for a COM trajectory minimizing a cost function combining squared errors in desired ZMP and on control (i.e., jerk).

As RoboSimian has limbs that together account for approximately 60 percent of the entire mass, the COB and COM locations vary significantly as the robot moves, requiring care in planning. Below, we describe the general process of planning desired, stable walking behaviors.

FIrst, we set a desired ZMP trajectory using diagonal pairs of limb end effector positions. Any ZMP trajectory that remains strictly inside the support polygon RoboSimian would be a stable planning choice. For simplicity, we plan for the ZMP to be along the mid-point of the two stance leg end effector positions. An example of desired end effector and associated ZMP locations is shown in Figure 4.

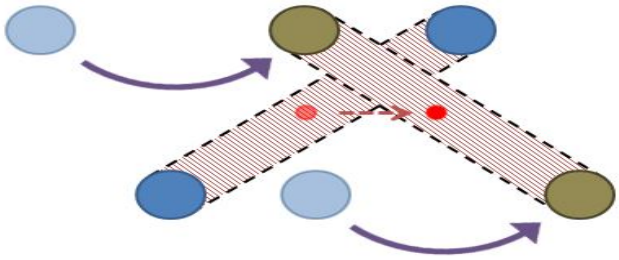


Fig. 4: ZMP trajectory. Above, two limbs move from the light blue footholds to the brown footholds while the ZMP is centered between the remaining two stance feet. Once all four limbs contact the ground, the ZMP is transitioned to the next, upcoming support polygon (shaded area).

From Figure 4, also note the ZMP moves $\frac{1}{2}$ the step length during each of two motions per cycle.

We then obtain a COM trajectory using Kajita's cart and table model and preview control. The key concept in this method is that by driving the system to track an appropriate, desired COM reference, the ZMP (i.e., center of pressure) is correspondingly driven to follow a particular, planned ZMP path. To track the COM, we first calculate joint trajectories for a nominal double support motion, as swing legs move two-at-a-time, in terms of the COB coordinate (which is fixed to the body frame) and record corresponding COM location along the way. This monotonic function (from COB to COM) is then inverted to obtain the required COB trajectory over time, as a function of desired COM.

This approach still neglects the effects of rotational accelerations on the ZMP, since we have assumed a point mass until now. Once joint reference trajectories are set, we then calculate the actual ZMP of our modeled motion plan by representing each of 33 rigid link elements in the robot as an equivalent-inertia set of six point masses. The true ZMP for our RoboSimian model is then:

$$x_{zmp} = \sum_{n=1}^{198} x_i(g + \ddot{z}_i) - z_i\ddot{x}_i \qquad (2)$$

Estimated errors between the actual ZMP and point-mass approximation are relatively small. Specifically, there is only a sub-mm bias in the point-mass estimate, and the standard deviation in error is about 1.8 cm, which should be acceptable for planning within a 5.5 cm radius foot base.

C. Value Iteration and Gradient-Based Optimizations

To find trajectories for a 7-DOF limb, one now-common approach in robotics is to use a Rapidly-exploring Random Tree (RRT) search algorithm [11]. However, an RRT does not guarantee the optimality of the solution. This method is useful when the speed of the limb is not crucial and when avoiding collisions between the limbs and nearby obstacles is important. While obstacle avoidance is important when traversing complex environments, we are focused on first getting RoboSimian to walk quickly across flat terrain. To find a trajectory for the swing limb that will optimize the speed of a static gait for RoboSimian we used an approach based on value iteration along with a gradient search. Value iteration is also known as backward induction. It is a process that derives the optimal policy and corresponding "cost-to-go" from any initial state to the final state. This algorithm requires: states, S, value of each state, V, actions that can be taken from each state, A, and a one-step cost, C, for transitioning from S to the next state S'.

The initial values of the states are initially set to zero and converge to their optimal values, which relate directly to time-to-go to complete a swing-leg motion. The values of the states are updated every iteration according to Equation (3).

$$V(s) = \min_A \left[C(S, A, S') + \gamma V(S')\right] \qquad (3)$$

where, γ denotes a discount factor, which was equal to 1 for our implementation, and where the goal state is uniquely defined with a fixed end cost of zero.

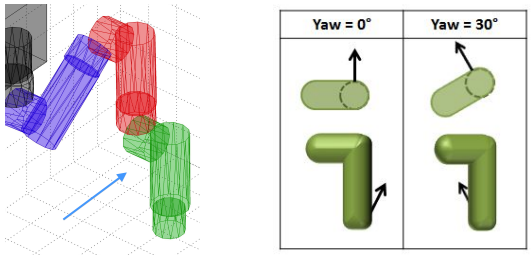


Fig. 5: Yaw Definition. At left is a close-up of the right, front limb, as illustrated at left in Fig. 3, with the blue arrow showing the forward direction of the body and the yaw set to zero in this configuration. The lower links (green) form at L-shape unit, which defines the yaw of the end effector, as illustrated in the key shown at right.

To allow for a tractable size mesh, our implementation fixes the pitch and roll of the end effector to be perpendicular to the ground. The states, S, were then defined by a 4D-mesh consisting of x, y, z, and yaw (defined as in Figure 5) for the end effector. The meshing region and particular start and end values for the swing leg in y and z were chosen by

calculating required joint velocities for end effector motions in the reachable workspace, i.e. by calculating the Jacobian across the workspace, and focusing on regions where the joint velocities are lowest when moving the limb. Actions were chosen using a combination of different distance, azimuth, elevation, and change in yaw. These actions place the reachable S' on spheres of different radii around S, and the one-step cost was the time to perform that action. Since the joint velocity limit of the RoboSimian is 1 rad/sec, time was measured by always saturating the currently-fastest joint at 1 rad/sec throughout a trajectory.

Our value iteration algorithm assumes that any trajectory must begin and end by lifting the limb straight up solely in the z-direction either 10 cm or 6 cm, representing a safety margin for terrain clearance. The time for lifting and lowering the foot is thus a fixed cost for a particular start and end state, added after the algorithm runs. Also, in addition to fixing the y and z values for the goal state, care was taken to ensure all mesh points were kinematically feasible, and any actions resulting in an infeasible end effector pose were reduced in magnitude to end at feasible mesh points.

In our gradient-based search, we start with sampled waypoints of the trajectory from value iteration. Rather than following the steepest path of descent (gradient descent), we use a randomized search, inspired directly by the REINFORCE algorithm of Williams [12], that follows some negative-gradient direction at each step, but not (in general) the path of steepest descent. This results in multiple possible end solutions for any given initial state of the search, and we find that iterating with randomization improves performance, compared with a strictly steepest descent search.

In this gradient-based search, we iteratively pick a random direction in the 5D space (now also including pitch on the end effector) for each waypoint at each step of the algorithms. Since we need to ensure this yields a smooth trajectory, we use a spline to connect the new waypoints and to calculate the resulting speed of the limb. If the resulting speed is faster than the original trajectory, the new trajectory becomes the base trajectory. Pseudo code is shown in Algorithm 1.

Algorithm 1 Pseudo code for gradient descent

```
function GRADIENT DESCENT(waypoints)
    speed_best ← GET SPEED(waypoints);
    for i = 1 : 100 do
        new waypoints ← waypoints + random dir;
        speed ← GET SPEED(new waypoints);
        if speed > speed init then
            waypoints ← new waypoints;
            speed_best ← speed;
        end if
    end for
return waypoints, speed_best;
end function
```

III. RESULTS

Unless stated otherwise, all gait speeds in our results are presented in feet/min, in keeping with the convention adopted by JPL and UCSB during the DARPA Robotics Challenge. For reference, 50 ft/min = 0.57 mph = 0.254 m/s. The gait speed for one-at-a-time swing leg gaits is calcualted using (4).

$$GaitSpeed = \frac{StepSize}{4 * SwingTime + ShiftBodyTime} \quad (4)$$

Here, the total time taken consists of the time to swing four legs and the time to move the body forward. The time to move the body was calculated by fixing the absolute (x,y,z) position of the each stance foot and allowing yaw to vary based using x vs yaw from the optimized swing leg trajectory as a rule-based function for stance legs, as well. To keep the COM safely within the support polygon for each swing leg motion while also avoiding limb-limb self-collisions, the body repositions before each footstep, so that the total swing leg motion can use the same relative path with respect to the COB. The full crawl gait is illustrated in Figure 6.

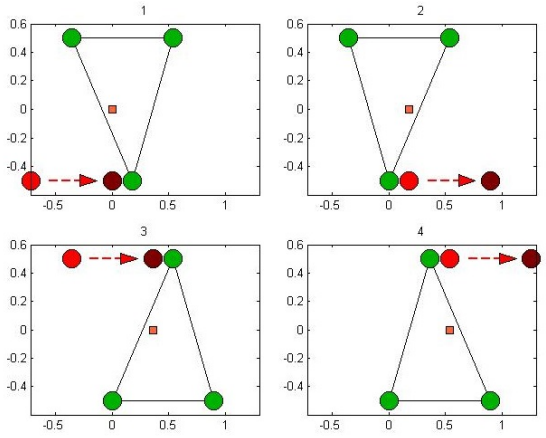


Fig. 6: Movement of center of mass. Before the swing leg (red circle) moves, the COB is repositioned to the same, relative x location (e.g., $x = 0$ in the case illustrated above).

A. ZMP Planning for Two-at-a-Time Gait

To test the feasibility of performing dynamic motions with RoboSimian, we calculated maximum required joint velocity (dq/dt) as a function of step size for the ZMP gait. These results are shown in Figure 7. We also calculated the net forward speed an a function of step size. This allows us to replot the same results to show forward walking speed versus dq/dt limit, as later summarized in Fig. 12.

From Figure 7, we conclude that RoboSimian requires a joint velocity limit of 1.1 rad/sec to perform a two-at-a-time gait. Of note, the required joint speed is not strictly monotonic with step length,due to the time required to lift and lower the leg. Specifically, as the step size gets smaller than 0.1 m, required joint velocity actually gets larger, resulting in a minimum required joint velocity for a step length of about 0.1 m.

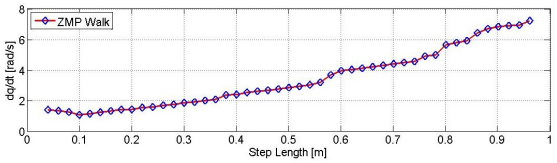


Fig. 7: ZMP step size vs. Maximum joint velocity needed

B. *Value Iteration for One-at-a-Time Gait*

Using value iteration, we determined what the approximate optimal step size and swing leg trajectory would be for Robosimian. We can claim only approximate optimal results here due to some simplifying assumptions to reduce the dimensionality of planning for a 34-DOF robot. The primary assumptions are summarized below.

The value iteration algorithm was implemented by first calculating solutions for each of a set of beginning and finishing x-coordinates for any given step size. Then, the best (i.e., fastest) start-end combination for any given step length is selected, to characterize speed for that step length. The y-coordinate for the start and finish leg positions in value iteration was fixed at y = -0.5, to limit the number of iterations. This choice is based on our Jacobian analysis of the workspace, which also led us to set the z-coordinate at a fixed height (z = -0.6 or -0.64 cm) for each simulation.

As explained earlier, there are 8 different inverse kinematics solutions to set the 6-DOF pose of the lower, L-shaped unit (Fig. 3), and we used the 3^{rd} and the 8^{th} family of IK solutions. For the different step sizes, we compared gait speed for different combinations of IK solution and step height (10 cm or 6 cm).

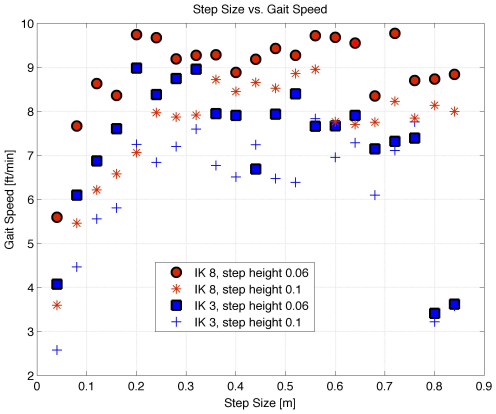


Fig. 8: Gait speed depending on the IK solution family and step height. The plot shows that reducing step height has a nonlinear effect in increasing overall gait speed, and the 8^{th} IK solution family always produced a faster swing than the 3^{rd} IK solution family for each tested step length.

Figure 8 presents data for both a smaller (6 cm) and larger (10 cm) step height, toward quantifying how sensitive speed is to the step height. The figure shows that the 8^{th} IK solution family with a step height of 6 cm yields the fastest gait speeds for any given step size, and that the fastest overall crawl gait solution uses a step size of 0.72 m. We correspondingly use this combination of IK solution, step height, and step length for the rest of our study here on improving crawl gait speed.

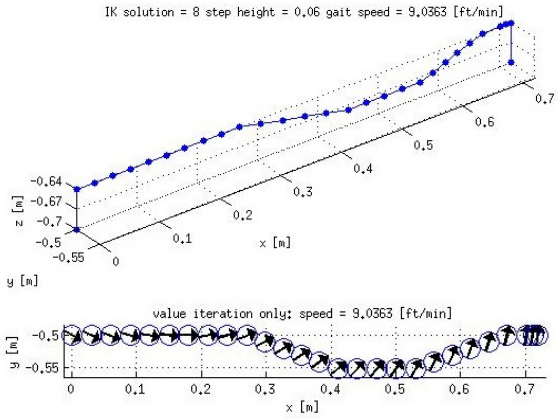


Fig. 9: Value iteration solution. Fastest swing trajectory when using the 8^{th} IK solution family with a step height of 6 cm (top) and an overhead view of the same path with the arrows representing the yaw of the end effector (bottom).

The fastest value iteration solution for swing trajectory, illustrated in Figure 9, produces a gait speed of 9.04 ft/min (0.10 mph or 0.046 m/s) using Equation (4). Note that the end effector trajectory swings outward, rather than moving in a straight line, to achieve faster swing time.

C. *Gradient-Based Improvement of One-at-a-Time Gait*

Next, we use the approximate optimal trajectory from value iteration to initialize a randomized, gradient-based search which allows both pitch and the start-end y coordinate of the swing leg to vary. For RoboSimian, allowing pitch of the end effector is essential for achieving very large (e.g., 0.95-meter) step lengths, since tilting the lower leg increases the reachable workspace. We hypothesize that pitch may also allow improve speed for less extreme step lengths, such as our 0.72-meter steps.

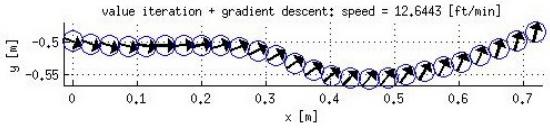


Fig. 10: Swing trajectory modified to include pitch, using a gradient-based search that is initialized using the value iteration solution. Points shown lie on a plane 6 cm above ground. The foot is now allowed to travel a few mm in y as it lifts off or set down on the ground, so that the start and end y coordinates need not match exactly here.

As Figure 10 shows, the speed of the gait has increased to 12.64 ft/min. This is due to the extra degrees of freedom allowed for the end effector and increased resolution, compared to a solution on a value iteration mesh.

Finally, we repeat our randomized gradient search using a heuristic swing leg trajectory to initialize the algorithm. Here, the initial trajectory that had zero yaw and pitch, constant z, curved y (Figure 11), and a monotonically increasing x. This resulted in a gait speed of 15.04 ft/min, which is surprisingly faster than the previous gait initialized from the value iteration solution.

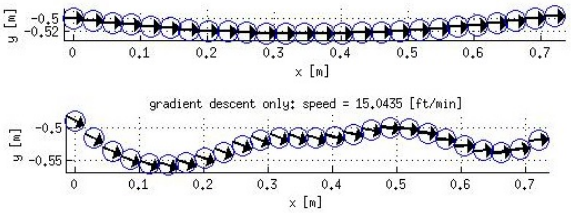


Fig. 11: Initial trajectory for using gradient descent only (top) and result of using gradient descent only (bottom)

Gradient-based methods provide only locally-optimal solutions, meaning this result is likely not the fastest possible gait for the robot. However, the result is still significant for two reasons. First, we note that this is a dramatic improvement oer our previous RRT-based gait planning, which achieved max speed of about 6 ft/min. Second, our primary goal here is in determining if double-support motions provide a significant speed improvement over one-at-a-time gaits for the given kinematic configuration of this robot, and our results are sufficient to make some general conclusions on this second point, as discussed below.

D. When to Use One- or Two-at-a-Time Planning

Having analyzed the two different methods for planning, we now summarize which of our modeled gaits is faster, as a function of joint velocity limit.

For the crawl gait that moves swing legs one-at-a-time, we use the best solution found (15.04 ft/min) for the current 1 rad/sec joint velocity limit. We assume speed for this motion simply increases linearly with the dq/dt velocity limit of the joints, over a certain range of (sufficiently low) velocities. Data for the two-at-a-time ZMP-based gait speed as a function of $dqdt$ come direction from Fig. 7, where each step length (x-axis in Fig. 7) corresponds to a particular overall gait speed, plotted on the y-axis in Figure 12.

From Figure 12, we see that if the joint velocity limits are between 1.62 and 3.62 [rad/s], using the ZMP based two-at-a-time swing leg method produces a faster walking gait than the fastest one-at-a-time swing leg gait found. Overall, the gait speeds are surprisingly similar, unless RoboSimian has a joint velocity limit greater than 5 rad/sec.

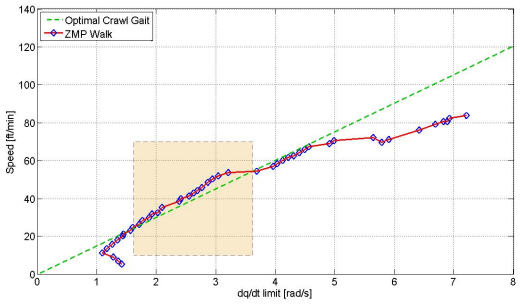


Fig. 12: ZMP based dynamic walk vs. Static walk. Shaded area indicates where ZMP method produces a faster gait speed.

IV. CONCLUSIONS AND FUTURE WORK

We present both ZMP-based planning for two-at-a-time swing leg motions and value iteration and gradient-based planning for a one-at-a-time swing leg motions in order to develop faster gaits for RoboSimian. First, we model ZMP based planning to determine the joint velocity limit needed to perform different sized steps. We predict that RoboSimian cannot achieve our dynamic double-support gait, no matter how small we set the step length, unless the velocity limit is at least 1.1 rad/sec.

We also investigate one-at-a-time swing leg motions for a more traditional crawl gait. Our methods allow for approximate optimization, i.e., optimization after introducing some constraints to the full problem. Our fastest gait speed for a 1 rad/sec joint limit is approximately 15 ft/min.

Comparing the two methods, we conclude it is faster to use the ZMP based two-at-a-time swing leg gait only if RoboSimian has a joint velocity limit of 1.62 to 3.62 rad/sec. Interestingly, unless the joint velocity limits exceed 5 rad/sec, there is no significant difference in predicted gait speeds between the two methods.

This general result contradicts previous work with the LittleDog robot, in which double-support gaits significantly improved overall walking speed. We conclude that additional degrees of freedom in the limbs are responsible for the non-intuitive result that a two-at-a-time leg motion is not significantly faster than a crawl for RoboSimian, and may in fact be a slower option across all ranges of joint velocity limits, if our crawl gait can be improved further. Specifically, LittleDog has only 3-DOF legs, while RoboSimian has 7-DOF limbs that can exploit yaw, pitch and (perhaps) roll of the end effector to reposition swing legs more rapidly.

The results from this paper are currently being applied to the actual RoboSimian robot at UCSB to confirm our simulations. One future task is to determine the maximum realistic joint velocity limit, dq/dt, to keep the ZMP a safe margin from within the support polygon. Also, we have included no roll in the end effector so far. Since our fastest swing leg trajectory solutions move side to side (in y), it would be interesting to quantify how allowing for roll might improve speed, just as pitch helped. A final extension to this work is to find optimal trajectories when not on even ground. For example, if RoboSimian is to climb a flight of stairs or other step-like terrain, we could search over a wider range in the z and use a body pose that pitches up or down, to match the average slope between terrain footholds.

REFERENCES

[1] CHEETAH Fastest Legged Robot, [Online]. Available: $http$: $//www.bostondynamics.com/robot\_cheetah.html$, [Nov. 18, 2013].

[2] S. Kagami *et al*., '"A Fast Dynamically Equilibrated Walking Trajectory Generation Method of Humanoid Robot," *Autonomous Robots*, vol. 12, pp. 71-82, 2002.

[3] S. Kajita *et al*., "Biped Walking Pattern Generator allowing Auxiliary ZMP Control," in *Proc. Int. Conf. on Intelligent Robots and Systems*, Beijing, China, 2006, pp. 2993-2999.

[4] K. Nishiwaki *et al*., "Online Generation of Humanoid Walking Motion based on a Fast Generation Method of Motion Pattern that Follows Desired ZMP," in *Proc. Int. Conf. on Intelligent Robots and Systems*, Lausanne, Switzerland, 2002, pp. 2684-2689.

[5] K. Byl. "Metastable Legged-Robot Locomotion," Ph.D. dissertation, Dept. Mech. Eng., Massachusetts Institute of Technology, Cambridge, 2008.

[6] K. Byl, A. Shkolnik, S. Prentice, N. Roy and R. Tedrake, "Reliable Dynamic Motions for a Stiff Quadruped," in *Proc. of the 11th Int. Symposium on Experimental Robotics (ISER)*, 2008.

[7] Z. Tang, C. Zhou, and Z. Sun, "Trajectory Planning for Smooth Transition of a Biped Robot," in *Proc. ICRA*, Taipei, Taiwan, 2003, pp. 2455-2460.

[8] H. Dong *et al*., "Gait Planning Of Quadruped Robot Based On Third-Order Spline Interpolation," in *Proc. Int. Conf. on Intelligent Robots and Systems*, Beijing, China, 2006, pp. 5756-5761.

[9] S. Kajita *et al*., "Biped Walking Pattern Generation by using preview control of Zero-Moment Point," in *Proc. ICRA*, Taipei, Taiwan, 2003, pp. 1620-1626.

[10] T. Katayama, T. Ohki, T. Inoue, and T. Kato. "Design of an optimal controller for a discrete-time system subject to previewable demand," *International Journal of Control*, vol. 41, no. 3, pp. 677699, 1985.

[11] A. Perez *et al*., "Asymptotically-optimal Path Planning for Manipulation using Incremental Sampling-based Algorithms," in *Proc. IROS*, San Francisco, CA, 2011, pp. 4307-4313.

[12] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229-256, 1992.