Experimental Results for Dexterous Quadruped Locomotion Planning with RoboSimian

Brian W. Satzinger, Chelsea Lau, Marten Byl and Katie Byl

Abstract RoboSimian is a quadruped robot inspired by an ape-like morphology, with four symmetric limbs that provide a large dexterous workspace and high torque output capabilities. Advantages of using RoboSimian for rough terrain locomotion include (1) its large, stable base of support, and (2) existence of redundant kinematic solutions, toward avoiding collisions with complex terrain obstacles. However, these same advantages provide significant challenges in experimental implementation of walking gaits. Specifically: (1) a wide support base results in high variability of required body pose and foothold heights, in particular when compared with planning for humanoid robots, (2) the long limbs on RoboSimian have a strong proclivity for self-collision and terrain collision, requiring particular care in trajectory planning, and (3) having rear limbs outside the field of view requires adequate perception with respect to a world map. In our results, we present a tractable means of planning statically stable and collision-free gaits, which combines practical heuristics for kinematics with traditional randomized (RRT) search algorithms. In planning experiments, our method outperforms other tested methodologies. Finally, real-world testing indicates that perception limitations provide the greatest challenge in real-world implementation.

Keywords Kinodynamic planning · Dexterous locomotion · Quadruped

B.W. Satzinger (✉) · C. Lau · M. Byl · K. Byl
Robotics Laboratory, University of California at Santa Barbara (UCSB),
Santa Barbara, USA
e-mail: bsatzinger@gmail.com

C. Lau
e-mail: cslau12@gmail.com

M. Byl
e-mail: marten.byl@gmail.com

K. Byl
e-mail: katiebyl@gmail.com

1 Introduction and Problem Statement

RoboSimian (Fig. 1) is a human-scale robot designed and built by JPL to compete in the DARPA Robotics Challenge (DRC) [1]. In this paper, we address the problem of planning desired joint reference trajectories for this high-dimensional quadruped to walk on rough terrain. This is an example of kinodynamic planning [2, 3], simultaneously considering kinematic constraints as well as dynamics. For RoboSimian, the primary kinematic challenges involve selecting among redundant solutions and avoiding collisions of the robot with terrain obstacles and with itself, while the main dynamic constraints are joint velocity limits and static balance requirements. For locomotion more generally, consideration of joint accelerations and allowable center of pressure (aka ZMP) location are also key considerations, but in practice, a low joint velocity limit (1.5 rad/s) makes these constraints relatively trivial to achieve for RoboSimian's current design.

Comparing with past work in planning quadruped locomotion on rough terrain for LittleDog [4–7], two particular challenges for RoboSimian are that it has seven degrees of freedom (DOFs) per limb, rather than three, and that perception relies solely on on-board sensing, rather than the use of motion capture (Vicon) along with saved (point-cloud) terrain maps.

Each of RoboSimian's four identical limbs consists of a kinematic chain of six rotational DOFs to define the (6 DOF) position and orientation of a lower leg segment, shown in green in Fig. 2, relative to the body frame. A final (7th) rotational joint simply allows the most distal end, or foot, of the lower leg to twist relative to the leg, so that the L-shaped lower leg segment itself can yaw while the foot remains fixed with respect to the ground. Even with only six actuators to set the 6-DOF pose of the lower leg, there are frequently redundant solutions. Qualitatively, each solution involves making one of two geometric choices (akin to "which way to bend
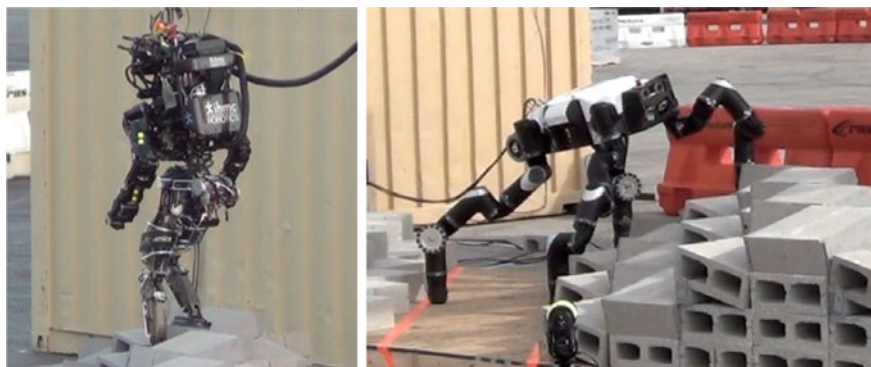

Fig. 1 Atlas humanoid controlled by IHMC (*left*) and RoboSimian (*right*) during the DARPA Robotics Challenge. The quadruped's larger footprint simplifies balance but makes kinematic planning a challenge, compared with the humanoid
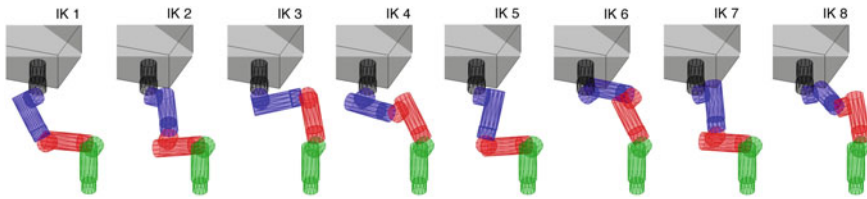
Fig. 2 Redundant inverse kinematic (IK) solutions for RoboSimian

an elbow") at each of three points along the chain: 2^3 results in a total of 8 IK families, as depicted in Fig. 2. The workspace and proclivity for self-collision of each family is different, and solutions for continuous trajectories in task space within a single family sometimes require discrete jumps in joint angles, so that kinematic planning is quite complex. In our problem formulation, we seek tractable methods to design trajectories for all 28 actuated joints, for slow walking with high-torque joints, given a set of candidate footholds on complex terrain.

2 Technical Approach

Our approach begins with a set of candidate footholds locations. We use a graph search to find a specific foothold plan, consisting of a series of steps that will be taken. Because our approach allows body motion during a step, we next search for body poses for the initial and final pose for each step, over a horizon of the next several steps. These poses are passed to our RRT-Connect implementation, which finds a path between the steps, respecting static stability, kinematic feasibility, and collision constraints. The motion plan is then executed on the robot (or a simulation). If more steps remain to be planned, control passes back to the RRT-Connect planner. Otherwise, control passes back to the pose finder. This process repeats until an error has occurred, or the goal has been reached. These phases are described in more detail below (Fig. 3).

2.1 Foothold Graph Search

Although A* search guarantees an optimal solution, it is suggested by [8] that suboptimal solutions may be found by related best-first search algorithms in much less time. In particular, they propose a K-best-first search algorithm that expands the K best nodes at once (A* corresponding to the special case of $K = 1$). We have implemented a modified approach where K threads expand nodes asynchronously. The empirical performance of this algorithm relative to alternative graph search algorithms is not a focus of this paper, but some performance data will be presented
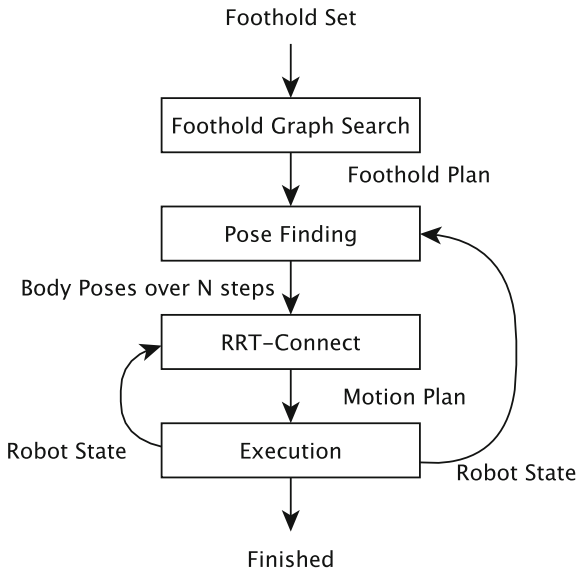
Foothold Set

Foothold Graph Search

Foothold Plan

Pose Finding

Body Poses over N steps

RRT–Connect

Motion Plan

Robot State   Execution

Robot State

Finished

Fig. 3 Planning phases in our approach

in Sect. 3.2 in the context of the entire system. We expect other foothold planning methods to be applicable as well [5–7, 9].

We search for a feasible sequence of steps to bring the robot to a goal location, using a pre-determined set of feasible foothold locations (e.g., based on perception and classification of the terrain, a priori knowledge of terrain shape as in DRC simulation, or manual selection). In our formulation, a node identifies a particular stance among the set of possible footholds. Our cost heuristic is the linear distance between the centroids of the footholds of two nodes (or between a node and the goal location). We expect future work to modify this cost heuristic to reflect other planning preferences, such as the preferential use of certain footholds over others.

The search can enforce a gait order or allow free gaits with steps in any order. In general, free gaits increase the complexity of the search because each node has more potential child nodes. However, for the DRC terrain the free gait search is advantageous because it enables the negotiation of difficult parts of the terrain. An experiment with gait order fixed did not find a solution.

2.2 IK Tables

Once a foothold plan is determined, we must choose body poses at the beginning and end of each step. This process requires an inverse kinematics solution that addresses

the difficulties inherent in high-DOF robot limbs. We used an IK table rather than an IK solver because, although an IK solver such as ikfast [10] can easily provide an arbitrary number of solutions to achieve a given 6-DOF pose with RoboSimian's 7-DOF limb, many such solutions result in joint reconfigurations once continuous limb motions are planned. Also, ikfast does not distinguish among "families" of kinematics when giving joint solutions; our grouping of solutions depends upon a customized but relatively slow IK solver, written in-house at UCSB. We precompute an IK table, in terms of only the relative 3-DOF position of a limb with respect to the body coordinate system. This exploits the fact that the lower leg need not be exactly normal to the ground during stance and greatly simplifies planning for body pitch and roll. Methods to produce and optimize IK table solutions for RoboSimian are described in more detail in [11].

2.3 Body Pose Search

Another advantage of pre-computing an IK table for the (x, y, z) coordinates of a limb is that we can also test potential body poses for feasibility very rapidly. Given a set of either 3 stance legs and 1 swing leg, we set a nominal body orientation (roll, pitch, and yaw) heuristically, to match the underlying foothold locations and heights. Then, we search numerically over a 6-DOF $7 \times 7 \times 9 \times 5 \times 5 \times 5$ (x, y, z, roll, pitch, yaw) grid of potential body poses centered on the heuristic pose. We search in an order that tests poses closer to the nominal pose first, and we terminate as soon as a single feasible pose is found. A feasible pose consists of one that is kinematically feasible for all four limbs, with static stability on a support triangle given by the 3 stance footholds.

In order to handle uncertainty in the terrain while planning, we also test that the swing foot will be able to reach above and below the planned foothold location while remaining kinematically feasible. This ground penetration distance can be set as a parameter. Choosing a larger value will allow greater uncertainty in the terrain, but also forces the planner to choose only conservative motions.

RoboSimian's four limbs together limbs account for roughly 60 % of its total mass. Because limb motions affect center of mass location significantly, testing a step for feasibility requires performing two body pose searches, one with the swing foot at the initial pose, and one with the swing foot at the final pose. This also allows us to plan steps with different body poses at the beginning and end. In this paper, we analyze on the effect of allowing body motion during a step on the volume of reachable footholds. Our results demonstrate that this capability does not have much advantage when planning a regular forward crawl gait on flat ground, but does significantly increase the reach on complex terrain with irregular steps or height changes.

We expect future work to address several shortcomings with this method of finding body poses. The approach is somewhat computationally expensive. In our results, we quantify the time spent searching for body poses in comparison to other planning

processes. The search also does not guarantee that the solution is better than other possible solutions or that it is far away from infeasibility. The authors plan to implement either a more sophisticated search or a subsequent pose optimization step to address one or both of these issues.

2.4 Motion Planning

Our general trajectory planning framework is described in more detail in [12]. We use RRT-Connect [13] to solve for a feasible path to a previously established goal position. Kuffner [14] has demonstrated this method to plan locomotion for a humanoid with 6-DOF limbs, but, in practice, this required a search over an apparently much smaller configuration space (e.g., C^3) than in our case (C^{16}). Several works plan locomotion by first searching over a graph and then filling in allowable motions [15–17]. In particular, Bretl [17] developed a non-gaited motion planner for the LEMUR quadruped, which has 3 DOF per limb. Hauser [16] solved for non-gaited motions on a 36-DOF humanoid by focusing on clever (contact-before-motion) sampling, but a single step still required several minutes, and a plan for climbing a ladder took a few hours, computationally.

As described in [12], our approach uses RRT-Connect to solve for paths between the initial and final goal. We will briefly summarize this to provide context for our experimental results. We parameterize the configuration space in a way that allows us to reduce the number of dimensions substantially when compared to a naive approach while simultaneously addressing kinematic closure of the stance legs and allowing the use of all degrees of freedom on the swing limb in order to allow dexterous motion.

Figure 4 illustrates this approach. During a swing motion, the complete pose of the robot can be specified by the 7 joint angles of the dominant limb and the 7 joint angles of the swing limb. We can also allow rotation at the contact between the dominant limb and the ground by introducing 2 additional degrees of freedom to give roll and pitch at this contact; recall that the most distal actuated joint already allows for yaw near the ground contact. This gives a total of 16 degrees of freedom.

Fig. 4 Cartoon sketch from [12] illustrating the design of our RRT-Connect configuration space parameterization for RoboSimian
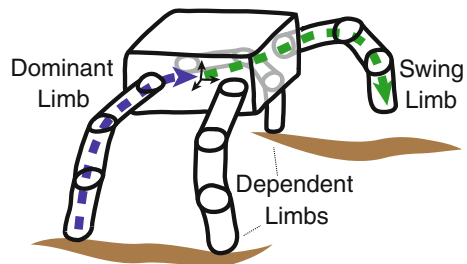
The remaining two dependent limbs are not directly represented in the parameterization. We determine their positions using IK table lookups. This requires knowing the location of the footholds relative to RoboSimian's body. The body position, in turn, is defined relative to the dominant foothold by the forward kinematics of the dominant limb. It is possible to generate poses where one or more dependent limbs do not have a kinematically feasible IK table solution. These poses are considered to be infeasible and are not used.

A similar approach can be used to allow a body shift with all four feet in contact with the ground along with different initial and final body poses. This still requires one dominant limb to determine the body pose and the remaining three limbs to be dependent limbs. This gives 9 degrees of freedom, including the roll/pitch contact with the dominant foothold.

All dimensions are given in radians, and angles are not wrapped at multiples of 2π. Although RoboSimian's actuators do not have hard joint limits and can continuously rotate, accumulating several rotations could damage cabling passing through the actuators. To avoid the accumulation of rotations during planning, we treat all joint angles (e.g., 0 and 2π) as distinct.

3 Results

In this section, we present results from various experiments that demonstrate the capabilities and benefits of our approach to trajectory planning for Robosimian.

3.1 Feasible Step Volume

In contrast with the planning approach used on RoboSimian during the 2013 DRC trials, our RRT configuration space design supports movement of the body during a step, from an initial pose to a final pose. We performed an analysis to consider the effects of this design decision on the ability for our system to plan steps on difficult terrain.

For the analysis, the robot's initial footholds are placed in a crawl-gait position corresponding to a steady state step length of 0.50 m. Then, we consider planning a step with one of the limbs from its initial position to a set of goal positions on a 3D grid. The search grid had 32 points in x and y, and 18 points in z. We compare the volumes reachable when the initial and final poses are forced to be the same, and when they are allowed (but not required) to be different. Therefore, the "same pose" volume S will always be a subset of the "different pose" volume D, and we can quantify the benefits of this approach by the size and shape of the relative complement of S in D.

The analysis found that 2541 of the destination foothold locations were likely reachable based on finding a feasible body pose for the end of the motion. Our
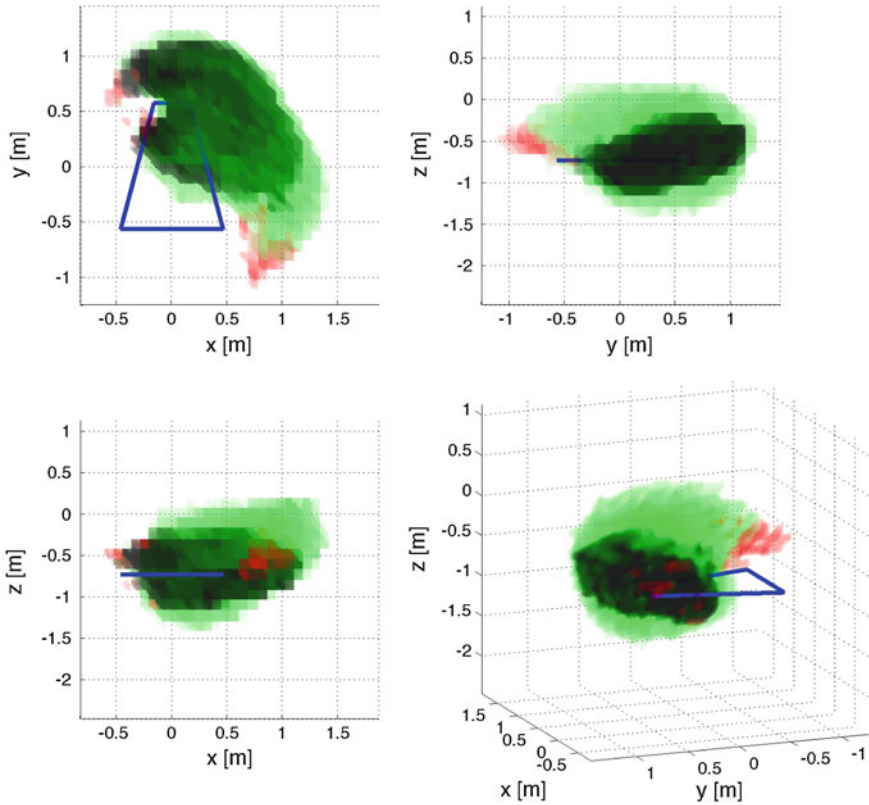
Fig. 5 Volumetric rendering of the reachable volume from several perspectives. *Black* shows the volume S, and *green* shows the relative complement of S in D (showing the region reachable only with body motion during a step). *Red* shows regions contained in S and D where the RRT solvers were not able to find a path, despite the existence of feasible initial and final poses. *Blue* shows the outline of the initial support region, with the initial swing foothold embedded in the reachable volume

RRT-Connect implementation was able to find paths to 2499 of them (98.3 %). The remaining 42 points where the RRT-Connect search failed to find a path are shown in red in Fig. 5. The 2499 successful paths represent 1.92 Km of total motion, with an average step length (linear distance from start to finish) of 76.8 cm, and a maximum step length of 169.7 cm.

In contrast, when the body position was required to be the same at the beginning and end of the step, only 1065 steps were possible, with an average step length of only 55.7 cm, and a maximum step length of only 130.6 cm.

As Fig. 5 shows, the reachable volume is substantially increased by allowing the body to move during a step. Notably, the maximum distance that it is possible to move the foot directly forward (positive x direction) does not vary much between S and D, so there is little benefit for a regular crawl gait on approximately flat terrain.

However, *S* is much larger in other directions, suggesting benefits for irregular gaits, especially on terrain with large changes in height, or when the robot is turning.

3.2 Simulated Traversal of DRC Terrain

We performed a simulation experiment to quantify the performance of our approach. A simulated DRC terrain was created by placing footholds on a $16 \times 16 \times 6$ inch grid, to match the spacing created by the cinderblocks used in the trial. An additional foothold was placed on the highest terrain level in order to allow the A* planner to find a feasible solution. The goal location was set 9 m from the starting location. In order to reduce the A* search space, only two rows of the terrain grid were populated with footholds, as this was sufficient to allow a solution (Fig. 6). We also did not consider the orientation of the footholds, which are on locally sloped surfaces for the second half of the terrain.

Although collision detection with the terrain is used during RRT planning, integration into body pose finding and A* planning is not complete as of writing. If the initial and final body poses chosen for a step happen to be in collision with the terrain, the RRT solvers will be unable to find a solution. Therefore, for this simulation, a terrain collision model was not used. However, checks for self collisions between different parts of the robot are always made.

The simulation was performed using RoboSimian's control software in a special offline mode, which provides the software interfaces, interpolation algorithms,

Fig. 6 RoboSimian straddles the second ridge on a simulated crossing of the DRC terrain
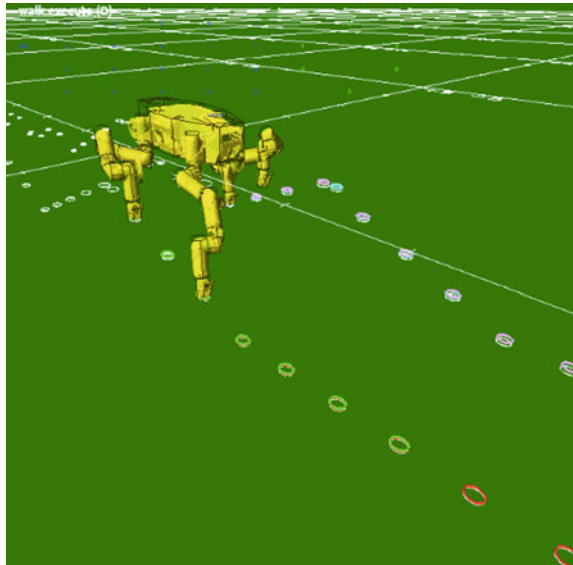
Table 1 Time spent in various subtasks (non-overlapping) while simulating DRC terrain crossing

Subtask	Time (s)	Time (%)
Foothold graph search	169.9	13.1
Pose finding	192.4	14.9
RRT-connect	212.8	16.5
Execution	647.2	50.0
Other	71.1	5.5
Total	1293.5	100.0

and error checking that are used with the robot hardware. Therefore, the planning/execution cycle occured in real time with representative communications overhead. All software involved in the simulation was run on a single laptop with an Intel i7-4900MQ 2.8 GHz CPU and 16 GB of memory. All planning was done autonomously, without operator input.

The A* search was executed once at the beginning for the entire terrain. This process was multithreaded using 7 threads to expand the search tree in parallel. Pose finding was performed every four steps, and chose 8 body poses (for the beginning and end of each step). The pose finding search used 8 threads.

RRT based pathfinding algorithms were used to find feasible and stable paths between the initial and final poses given by each step or body movement. Our RRT formulation gives a special role to one of the three (during a step) or four (during a body shift) stance limbs, which we call the 'dominant' limb. We call the RRT solver three (or four) times with each choice of dominant limb in turn. The solution with the shortest time (respecting the robot's actuator velocity and acceleration limits) is used. Because these calls are independent, in principle, they can be parallelized. However, for this experiment they were done serially.

For this offline experiment, we did not include uncertainty in the terrain height relative to the given foothold locations. This would have allowed us to use pipelining and, for example, call the RRT for the next step while the current step was still executing. However, when running on hardware we discover the true height of the terrain at the end of a step when contact is made with the ground and incorporate that knowledge into planned sequence of footholds. Therefore, we postpone calling the RRT solvers until just before the motion is to be executed so that motions are planned with the most up to date information possible (Table 1).

3.3 Obstacle Avoidance

In the prior simulation, we did not include a terrain collision model. In order to demonstrate the capability to avoid complex obstacle geometry, we generated a set of obstacles for Robosimian to avoid when taking one step forward. Boxes were arranged in the formation specified in Table 2 and shown in Fig. 7. A foothold plan

Table 2 Body poses, foothold locations, and box poses and sizes for the RRT demonstration in Fig. 7

	Position (m)			Rotation (°)			Size (m)		
	x	y	z	Roll	Ritch	Yaw	x	y	z
Body—entry	0.10	0.10	0.00	0	0	0	–	–	–
Body—exit	0.10	0.10	0.00	10	−5	10	–	–	–
Foothold—FR	0.51	0.51	0.65	–	–	–	–	–	–
Foothold—BR	−0.51	0.51	0.65	–	–	–	–	–	–
Foothold—BL (start)	−0.51	−0.51	0.55	–	–	–	–	–	–
Foothold—FL	−0.51	−0.51	0.65	–	–	–	–	–	–
Foothold—BL (end)	−0.01	−0.51	0.55	–	–	–	–	–	–
Box 1	−0.22	−0.50	0.56	0	0	0	0.04	0.61	0.19
Box 2	−0.29	−0.75	0.26	0	0	0	0.04	0.41	0.76
Box 3	−0.22	−0.64	0.31	−10	30	5	0.20	0.61	0.03
Box 4	−0.27	−0.30	0.56	0	50	0	0.49	0.20	0.03

The acronyms FR, BR, BL, and FL refer to the *front right limb*, *back right limb*, *back left limb*, and *front left limb*, respectively
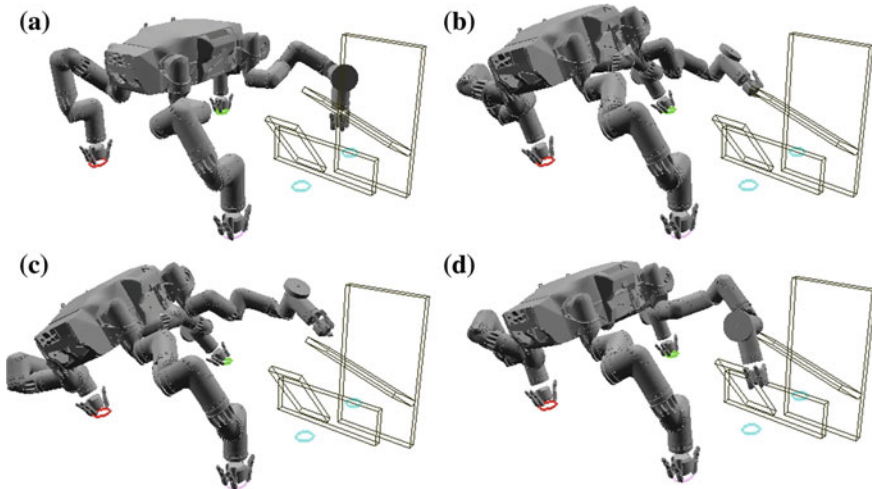

Fig. 7 Demonstration of an RRT generated swing trajectory with four *boxes* arranged as obstacles. RoboSimian's back *left* limb successfully avoids the *boxes* as it swings to its next foothold 0.50 m forward in the x-direction

was generated to move Robosimian's back left leg forward 0.50 m. Such a situation would likely be encountered on rough terrain, especially those involving fallen debris.

Because our pose finding algorithm does not yet account for terrain collisions, the initial and final body poses were chosen manually to keep Robosimian's center of pressure within the support polygon during swing. A rotation was added to the final body pose to allow for body adjustments required to step over the obstacles.

The footholds for the non-swing limbs were set in the crawl gait formation with end effectors approximately 1 m apart in both the x- and y-directions. The end effector of the swing leg was set 0.10 m above its nominal walk pose to account for contact behavior.

A swing trajectory to maneuver Robosimian's back left leg over the obstacles was successfully found using RRT-Connect. The generated trajectory included 241 waypoints with a playback time of 12.13 s. Such complicated maneuvers would not be possible using only IK table solutions since many of the poses required of Robosimian to swing its leg over the boxes are atypical. As demonstrated by this example, when traversing rough terrain, many situations require unique and complicated movements beyond those that are reasonable to store in an IK table. The combined RRT plus IK tables approach allows us to generate feasible solutions for unpredictable situations within an acceptable time frame. Computation time for the RRT to generate the swing trajectory depicted in Fig. 7 was 17.12 s. The RRT required 8227 nodes from the starting foothold and 7912 nodes from the goal foothold for a total of 16139 nodes.

Another important aspect of this approach to solving for swing trajectories is the flexibility of the body pose during swing. The movement of the body is integral to the RRT's ability to find a solution for the foothold as the contortions required of the swing limb without movement of the body would result in collisions between joints and the body.

3.4 DARPA Robotics Challenge

Our methods have been tested for short foothold plans in lab on RoboSimian in preparation for the DARPA Robotics Challenge (DRC). However, the most significant (and disappointing) result for us was that drift in perception of our world map made careful foothold planning a significant challenge. In practice, RoboSimian performed the locomotion task during the DRC using RRT-Connect for a set of heuristic
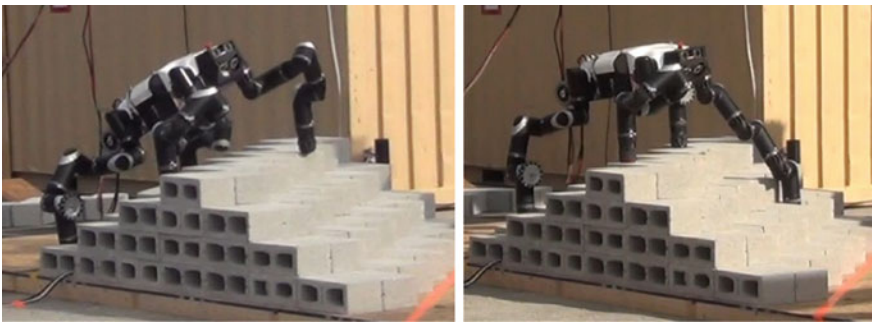

Fig. 8 RoboSimian pitches its body and stretches to a near-singular configuration to traverse terrain at the DRC

footholds planned blindly on terrain, using force feedback to detect ground contact. Results were still good enough to place 5th in the 2013 DRC trials and qualify for the final competition, scheduled for June 2015. Since then, we have successfully crossed the DRC terrain shown in Fig. 8 using our combined IK Table and RRT approach; however, practical implementation currently requires that an operator periodically corrects for drift in the estimation of body pose with respect to the world map. On this extreme terrain, combined computation and execution time allows RoboSimian to walk at about 1.2 ft/min, and walking speed increases to around 5 ft/min on milder portions of the DRC terrain. For reference, the fastest pre-planned walking gait we can currently obtain on RoboSimian goes about 15 ft/min on flat ground, which benchmarks limitations due to joint velocity limits, versus for computation and terrain roughness.

4 Experimental Insights and Future Work

Our key experimental insights are (1) that our approach blending IK tables and RRT-Connect provides a computationally practical kinodynamic planning method with a high rate of success, (2) that allowing for body motions during a swing leg motion significantly increases the set of reachable next footholds compared with sequentially moving either a leg or the body during a crawl gait, and (3) that adequate perception is a strong requirement for real-world implementation on RoboSimian. Specifically, although three pairs of forward-facing stereo cameras can view front limbs, rear limbs are often well over a meter behind the front limbs, requiring an accurate world map of previously viewed terrain. Future upgrades to the robot plan to incorporate LIDAR sensing to improve mapping and localization significantly.

References

1. Hebert, P., Bajracharya, M., Ma, J., Hudson, N., Aydemir, A., Reid, J., Bergh, C., Borders, J., Frost, M., Hagman, M., Leichty, J., Backes, P., Kennedy, B., Karplus, P., Byl, K., Satzinger, B., Shandar, K., Burdick, J.: Mobile manipulation and mobility as manipulation—design and algorithms of RoboSimian. J. Field Robot. (JFR), Special Issue on the DRC (to appear) (2014)
2. Donald, B., Xavier, P., Canny, J., Reif, J.: Kinodynamic motion planning. J. ACM **40**(5), 1048–1066 (1993)
3. Donald, B.R., Xavier, P.: Provably good approximation algorithms for optimal kinodynamic planning for Cartesian robots and open-chain manipulators. Algorithmica **14**(6), 480–530 (1995)
4. Byl, K., Shkolnik, A., Prentice, S., Roy, N., Tedrake, R.: Reliable dynamic motions for a stiff quadruped. Proc. ISER **54**(2009), 319–328 (2008)
5. Byl, K.: Metastable legged-robot locomotion. Ph.D. dissertation, MIT, 2008

6. Kolter, J.Z., Ng, A.Y.: The stanford littledog: a learning and rapid replanning approach to quadruped locomotion. Int. J. Robot. Res. (IJRR) **30**(2), 150–174 (2011)
7. Zucker, M., Ratliff, N., Stolle, M., Chestnutt, J., Bagnell, J.A., Atkeson, C.G., Kuffner, J.: Optimization and learning for rough terrain legged locomotion. Int. J. Robot. Res. **30**(2), 175–191 (2011)
8. Felner, A., Kraus, S., Korf, R.: Kbfs: K-best-first search. Ann. Math. Artif. Intell. **39**(1–2), 19–39 (2003). http://dx.doi.org/10.1023/A%3A1024452529781
9. Vernaza, P., Likhachev, M., Bhattacharya, S., Chitta, S., Kushleyev, A., Lee, D.D.: Search-based planning for a legged robot over rough terrain. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2009, pp. 2380–2387
10. Diankov, R.: Automated construction of robotic manipulation programs. Ph.D. dissertation, Carnegie Mellon University, Robotics Institute, Aug 2010
11. Byl, K., Byl, M., Satzinger, B.: Algorithmic optimization of inverse kinematics tables for high degree-of-freedom limbs. In: Proceedings of ASME Dynamic Systems and Control Conference (DSCC), 2014 (to appear)
12. Satzinger, B., Byl, K.: More solutions means more problems: Resolving kinematic redundancy in robot locomotion on complex terrain. Submitted to IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2014
13. Kuffner, J., LaValle, S.: RRT-connect: an efficient approach to single-query path planning. In: Proceedings IEEE International Conference on Robotics and Automation (ICRA), vol. 2, pp. 995–1001 (2000)
14. Kuffner, J., Kagami, S., Nishiwaki, K., Inaba, M., Inoue, H.: Dynamically-stable motion planning for humanoid robots. Auton. Robots **12**(1), 105–118 (2002)
15. Bouyarmane, K., Kheddar, A.: Humanoid robot locomotion and manipulation step planning. Adv. Robot. Int. J. Robot. Soc. Jpn, Special Issue on the Cutting Edge of Robotics in Japan 2012, **26**(10), 1099–1126 (2012)
16. Hauser, K., Bretl, T., Latombe, J.C.: Non-gaited humanoid locomotion planning. In: Proceedings of International Conference on Humanoid Robots. IEEE, 2005, pp. 7–12
17. Bretl, T.W.: Multi-step motion planning: application to free-climbing robots. Ph.D. dissertation, Citeseer, 2005