Reinforcement Learning Methods to Enable Automatic Tuning of Legged Robots

Mallory Tayson-Frederick
Pieter Abbeel, Ed.
Ronald S. Fearing, Ed.

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2012-145
http://www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-145.html

May 31, 2012

Reinforcement Learning Methods to Enable Automatic Tuning of Legged Robots

Mallory Tayson-Frederick
Masters of Engineering in Electrical Engineering and Computer Science
University of California, Berkeley
Advisor: Pieter Abbeel

***Abstract* –** Bio-inspired legged robots have demonstrated the capability to walk and run across a wide variety of terrains, such as those found after a natural disaster. However, the survival of victims of natural disasters depends on the speed at which these robots can travel. This paper describes the need for adaptive gait tuning on an eight-legged robot, which will enable it to adjust its gait parameters to increase the speed at which it navigates difficult and varying terrains. Specifically, we characterize the robot's performance on varied terrains and use the results to inform the implementation of a finite-difference policy gradient reinforcement learning algorithm. We compare the robot's performance under hand-tuned policies with the performance under the reinforcement learning algorithm, and finally, suggest improvements to the presented policy search process.

I. INTRODUCTION

During a disaster situation, the goal of search and rescue operations is to rescue the greatest number of people in the shortest amount of time. In particular, time is the largest factor in determining the number of people rescued from a disaster site. As seen in Figure1, the less time a victim spends trapped in rubble, the higher their chances are for surviving [1]. In order to reduce time spent in search efforts and to minimize risk to human rescue forces, robots are being developed to aid in the search for disaster survivors.

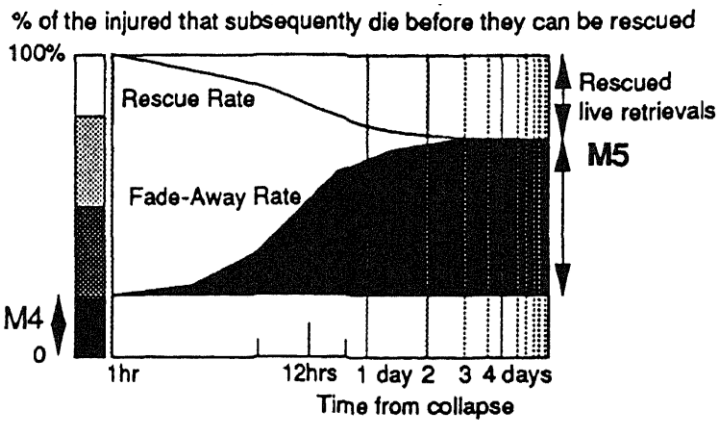

Fig. 1: Mortality rate of people trapped after a natural disaster [1]

Robots which can perform this role take many forms, and although existing platforms accomplish the necessary tasks, they do not do so optimally. These platforms are often remote operated, slower than a human rescuer, and can be prohibitively costly. Biomimetic legged robots such as the Octo- Robotic Autonomous Crawling Hexapod, or OctoRoACH, offer solutions to these problems. They cost a tenth of the price to manufacture than existing solutions and have demonstrated the ability to walk autonomously over a variety of terrains. However, the efficiency and speed of their locomotion vary greatly according to the timing of their gait. Because this timing is also affected by several external factors, it is time consuming, labor-intensive, and error-prone to tune by hand for each type of terrain the robot might encounter.

Thus, the challenge addressed here is to enable any OctoRoACH to automatically tune its gait timing while traversing different terrains, allowing the robots to travel to their destinations quicker, and ultimately perform their duties better. In order to accomplish this, we first confirm the need for adaptive gait control on the OctoRoACH through a characterization of its performance on different

terrains and with varying control schemes. Next, we describe a simple policy gradient reinforcement learning algorithm and its implementation on OctoRoACH which will enable a robot to continuously tune its own gait parameters depending on specific mechanical and environmental conditions. Along with other developments in robotic technology, the implementation of adaptive learning techniques will ultimately enable OctoRoACH and other biomimetic legged robots to aid in search and rescue operations.

The remainder of this paper is structured as follows. Section II discusses the development of OctoRoACH and previous work in the area of learning algorithms applied to legged robots. Section III describes our specific approach and methodology, while Section IV presents the results of our experimentation and the analysis of those results. The paper concludes in Section V with a summary of findings and recommendations for future work.

II. LITERATURE REVIEW

In recent years, there have been a variety of developments in the field of bio-inspired legged robots and adaptive algorithms to accompany them. These developments are mainly motivated by the fact that legged robots are often more successful than wheeled or track drive robots at traversing terrains which include features taller than the robot's hip height [2]. Despite these robots' successes in traversing terrain with large features, the variation in terrains a robot may encounter practically requires that the robot be able to modify its stride, or gait, to adjust to disturbances which may affect its operation. Therefore, adaptive control is an important addition to this category of legged robots.

We will discuss several relevant biomimetic robots here as well as the methods used to control them. It is also appropriate to understand the development of OctoRoACH and the design decisions that led to the current model.

The last four years have each seen a new generation of the robot family leading up to the current design of OctoRoACH. First, there was the miniRoACH [3]. Weighing in at only 2.4g, the miniRoACH qualifies as a millirobot and was the smallest and lightest autonomous legged robot produced at the time of its fabrication [3]. This basic design of RoACH included six legs and an alternating tripod gait. Because it was so small, the robot used shape-memory alloy (SMA) and opposing springs as actuators for the legs. In order to steer, RoACH's middle legs were made to be shorter than the outside legs, allowing the timing of the gait to dictate the heading. Although the fabrication (Smart Composite Manufacturing) and mechanical design of this robot provided proofs of concept in the design of milli-sized robots, other technologies did not yet exist at a level which could be integrated with the small size of the RoACH. Therefore, the next generation of robots, the Dynamic Autonomous Sprawled Hexapod, or DASH [1], was built with the same manufacturing process, but was made slightly larger (16g) to better understand locomotion dynamics and control of hexapedal robots [1]. Also with six legs, this DASH utilized the same alternating tripod gait seen in miniRoACH, by using a single DC motor. In order to steer, DASH was designed with SMA wire which could deform the robot's frame to bias one turning direction or the other. Overall, DASH was more effective at running straight than it was at making reliable turns at zero forward velocity [5]. In order to improve the turning dynamics, the next robot in the family, DynaRoACH (24g), featured an improved leg design and a new steering method while still relying on a single DC motor [4]. The new legs were designed in a semi-circle C shape and made out of a compliant rubber material rather than the stiff paddle-style legs used on DASH. In order to make turns, the middle leg on one side of DynaRoACH's body was switched out with a stiffer leg. Although the switching of the legs was done offline, it was also proven that they could be stiffened dynamically using SMA wire embedded in the leg itself [4]. Despite the improvements in turning ability, the DynaRoACH was still more effective at walking straight than making turns.

To accommodate the need for reliable turning, the most recent design in the RoACH family, OctoRoACH, was given a second DC motor so that each side of the robot could be driven individually, similar to the drive system on a tank [5]. With the added motor, the two sets of legs are no longer forced to be in synchronization with each other, allowing the possibility for instability in the robot's gait. OctoRoACH was given two extra legs to ensure stability. With the extra legs and motor, OctoRoACH is the largest robot of the RoACH family, weighing in at 35g [5]. A further description of the mechanical design of the robot can be found in Section III.

The design additions described above allowed for improved turning and steering ability in OctoRoACH, specifically during turning at low forward speeds [5]. With the improved functionality, two main steering methods have been implemented on the robot. The first method is simple leg velocity control which overcomes disturbances, such as change in terrain friction, using a basic rate-gyro control system. This is the method used here, and will be described in further detail below. The second method of steering involves mounting a tail to the robot using a modified servo [5]. While successful, it was found that the tail method of steering the robot was most useful when very quick turning was needed [5]. Since we are not interested in fast turning of the robot, the simpler leg velocity control method was deemed sufficient for the work done here.

In addition to the RoACH family, there have been numerous other legged robots developed recently. These include Sprawl [6], RHex [7], Whegs [8], and Aibo [9] to name a few. Here, we will review the most relevant robots and some of the adaptive gait control algorithms which have been implemented on them.

Also inspired by a cockroach, the Sprawl robot displays many similarities to OctoRoACH. It has six legs and uses an alternating tripod gait for locomotion. However, this robot uses a system of pneumatic pistons to actuate the legs individually rather than electric motors for actuation of a collection of legs [6]. For this robot, Cham et al derived an adaption law from a single-legged

vertical hopping model; the law attempts to maximize the length of the robot's stride by adjusting the stride period and the duty cycle of the actuating pistons. The adaption algorithm uses feedback from binary contact switches placed on the robot's feet [6]. It was found that the robot successfully optimized its locomotion parameters based on the adaption method described. Although the algorithm relies on the unique mechanics of Sprawl, making it inapplicable to OctoRoACH, this success further motivates the need for adaptive and autonomous gait control on biomimetic robots.

A robot which bears more mechanical similarity to OctoRoACH is RHex. Featuring only six legs, RHex also uses a C shaped leg design; however, these legs are all individually actuated and rotate in a complete circle [10]. Because the legs are individually actuated, the gait of RHex is much more complex than that of OctoRoACH and consequently has a high dimensional control policy. In order to reduce the complexity of this high-dimension space, the legs on RHex were grouped together to form the same alternating tripod gait used in previously described robots. To optimize the gait on RHex, a Nelder-Mead algorithm was implemented. This algorithm successfully optimized the robot's gait for both speed and endurance, but was found to provide unstable results when presented with poor initial conditions [10]. A second optimization tool implemented only in a simulation of RHex utilized the Bipedal Spring-Loaded Inverted Pendulum (BSLIP) model as a template for control of the robot [11]. Although it was not implemented on the actual robot, this method provides another example of how adaptive gait optimization can improve robot control.

The previously discussed robots bear the most mechanical similarities to OctoRoACH; however, the adaptive algorithms used on the Sprawl and RHex robots were specifically tailored to the mechanics of those robots. A more general application of reinforcement learning techniques can be seen in use on the Sony Aibo quadruped robots [9]. The Sony Aibo is a commercially available robot which resembles and moves like a dog; it uses four legs which are all actuated in multiple places. Using the Aibo, Kohl and Stone presented a policy gradient method of optimizing gait for speed which averages

the speed of the robot during several trials using different gait parameter settings. This policy gradient method was successful in finding gait settings for the robot which maximized forward speed.

Overall, the Sprawl and RHex robots provide good mechanical comparisons with the OctoRoACH, but the adaptive optimization algorithms implemented on these robots do not necessarily provide good models for algorithms on OctoRoACH. Reinforcement learning offers a promising approach which is easily applied and has been demonstrated to work well on other platforms.

III. METHODOLOGY

A. Robotic Platform

The robot used for this work, OctoRoACH [5], is pictured below in Figure 2. As discussed in Section II, the robot's mechanical design allows independent drive of its left and right sets of legs. The control and sensing hardware on the robot includes an inertial sensing unit (a three-axis accelerometer and a three-axis gyroscope) and an 802.15.4 wireless radio for communications. Figure 3 shows the block diagram for velocity control on the robot, where $\omega_{L,ref}$ and $\omega_{R,ref}$ are supplied by the user and the back-EMF for each motor is measured as during the off phase of the PWM cycle [5].

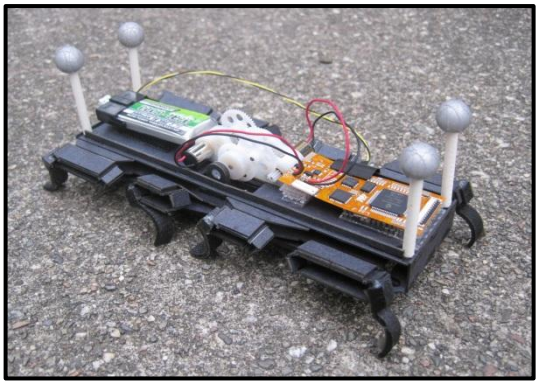

Fig. 2: OctoRoACH robot

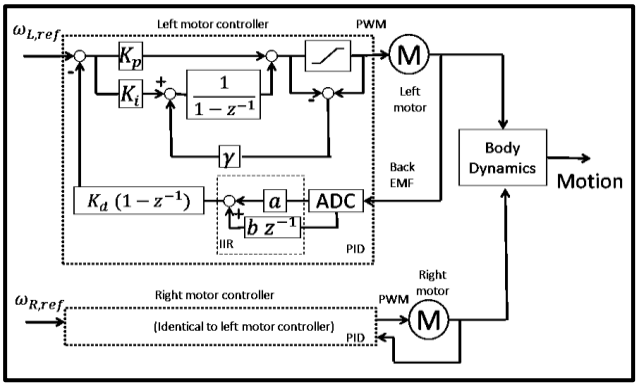

Fig. 3: Block diagram of velocity control loops using motor back-EMF [5]

B. Experiments

Experiments on the OctoRoACH crawler were split into three parts. In the first and second parts, open-loop and closed-loop tests were performed to confirm the need for reinforcement learning on the robot. In order to test this, two robots were run on three different surfaces with varying control policies, where a control policy $\Pi = (\theta_1, \theta_2)$ with θ_1 and θ_2 being the left and right motor thrusts. The surfaces tested were a smooth wooden surface, a carpeted surface, and gravel. The goal of finding an optimal policy was for the robot to travel in a straight line as fast as possible.

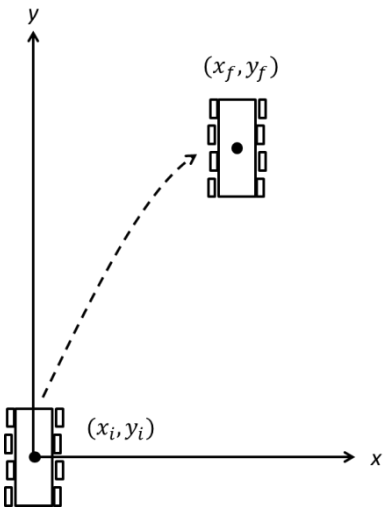

Fig. 4: Final and initial positions of the robot measured in the world coordinate system

As mentioned above, a specific control policy was considered better than others if the robot traveled in a straight line faster than any other run; therefore, performance metrics included speed and accuracy. These metrics were measured based only on the initial and final positions of the robot (see Figure 4), as well as the robot's running time. In both equations below, *x* and *y* are the horizontal and vertical positions of the robot respectively, and pose measurements were collected by hand through human observation, with accuracy of approximately 0.075 m.

The robot traveled relatively straight for each run, so speed was approximated as follows:

$$Speed = s(x, y, t) = \frac{\sqrt{(x_f - x_i)^2 + (y_f - y_i)^2}}{t} \quad (1)$$

Accuracy was used as a metric to provide an intuitive understanding of how straight the robot traveled on a particular run, and was calculated as follows:

$$Accuracy = a(x, y) = \sqrt{\frac{(y_f - y_i)^2}{(x_f - x_i)^2 + (y_f - y_i)^2}} \quad (2)$$

In order to hand tune each robot's control policy for a specific surface, we started by testing policies which optimized only for one performance metric, either speed or accuracy. However, it was found that a policy which optimized accuracy made the robot run slowly, and policies which optimized speed gave the robot low accuracy. Therefore, we tested a range of policies between the two extremes and chose policies which maximized both speed and accuracy as the optimal policy for a specific robot and surface.

After we found an optimal policy for each robot on all three terrains, we cross-tested policies across different robots and surfaces. Tables 1 and 2 show the full range of tests performed.

Table 1: Control Policies Tested on Robot 1

	Carpet Surface	Wood Surface	Gravel
Robot 1	R1 Optimal Carpet Policy	R1 Optimal Wood Policy	R1 Optimal Gravel Policy
	R1 Optimal Wood Policy	R1 Optimal Carpet Policy	R1 Optimal Wood Policy
	R1 Optimal Gravel Policy	R1 Optimal Gravel Policy	R1 Optimal Carpet Policy
	R2 Optimal Carpet Policy	R2 Optimal Wood Policy	R2 Optimal Gravel Policy
	R2 Optimal Wood Policy	R2 Optimal Carpet Policy	R2 Optimal Wood Policy
	R2 Optimal Gravel Policy	R2 Optimal Gravel Policy	R2 Optimal Carpet Policy

Table 2: Control Policies Tested on Robot 2

	Carpet Surface	Wood Surface	Gravel
Robot 2	R2 Optimal Carpet Policy	R2 Optimal Wood Policy	R2 Optimal Gravel Policy
	R2 Optimal Wood Policy	R2 Optimal Carpet Policy	R2 Optimal Wood Policy
	R2 Optimal Gravel Policy	R2 Optimal Gravel Policy	R2 Optimal Carpet Policy
	R1 Optimal Carpet Policy	R1 Optimal Wood Policy	R1 Optimal Gravel Policy
	R1 Optimal Wood Policy	R1 Optimal Carpet Policy	R1 Optimal Wood Policy
	R1 Optimal Gravel Policy	R1 Optimal Gravel Policy	R1 Optimal Carpet Policy

The closed-loop experiments followed the same general procedure as above; however, we wanted to know if the robot could perform better with simple sensory feedback integrated into the control. To do this, we used the onboard gyroscope and a PID controller which was provided with the platform. The control block diagram of this system is presented in Figure 5. The addition of a PID loop for steering control added four additional control parameters to the control policy. Therefore, when choosing an optimal policy for a specific robot on a specific surface, it was necessary to choose optimal PID parameters in addition to motor thrust values.

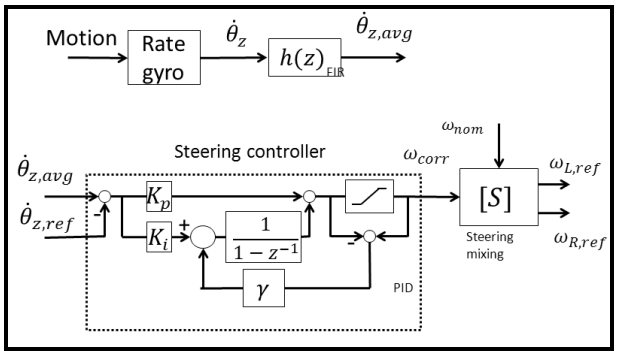

Fig. 5: Block diagram for steering control [5].

Once optimal policies were found for each robot on each surface, the policies were applied across robots and terrains in the same manner as for the open-loop policies.

After characterizing the robot's performance at different inputs, we implemented finite difference policy gradient. Based on the results of the open and closed-loop experiments, we found a direct relationship existed between speed and accuracy (see Section V). Due to this relationship, we determined that the harmonic mean of speed, *s(x, y, t)* and accuracy, *a(x, y)*, provided an intuitive reward function for the learning algorithm.

$$r(\Pi) = \frac{2\big(s(x,y,t)\big)\big(a(x,y)\big)}{s(x,y,t) + a(x,y)} \tag{3}$$

Therefore, the learning algorithm followed the procedure outlined below:

input: control policy $\Pi = \theta_1, \theta_2$

while not converged

for i = 1 **to** 6

generate policy variations $\Delta\theta_1, \Delta\theta_2$

perform trial with $\Pi^i = \left(\theta_1^i + \Delta\theta_1\ , \theta_2^i + \Delta\theta_2\right)$

record $s(x,y,t)$ and $a(x,y)$

calculate $r\left(\Pi^i\right)$

end for

estimate ∇r using least squares:

$$\begin{bmatrix} \theta_1^1 & \theta_2^1 \\ \theta_1^2 & \theta_2^2 \\ \theta_1^3 & \theta_2^3 \\ \theta_1^4 & \theta_2^4 \\ \theta_1^5 & \theta_2^5 \\ \theta_1^6 & \theta_2^6 \end{bmatrix} \nabla r = \begin{bmatrix} r(\Pi^1) \\ r(\Pi^2) \\ r(\Pi^3) \\ r(\Pi^4) \\ r\left(\Pi^5\right) \\ r(\Pi^6) \end{bmatrix}$$

$$\left.\begin{aligned} \theta_1 &= \theta_1 + \epsilon\, \nabla r_1 \\ \theta_2 &= \theta_2 + \epsilon\, \nabla r_2 \end{aligned}\right\}$$ Where ϵ was chosen such that $(\epsilon\, \nabla r_i)$ is on the order of five

end while

IV. RESULTS AND DISCUSSION

The results of the open and closed-loop hand tuning experiments provided useful insight into the characteristics of the robot. We found that no single control policy worked well across multiple terrains, or on different robots. For instance, Figure 6 indicates that control policies tuned for a specific surface performed well on that surface; they had high speed, high accuracy, or both. However, Figure 7 shows that those same policies do not perform well on different surfaces, as indicated by the wide spread of speeds and accuracies.

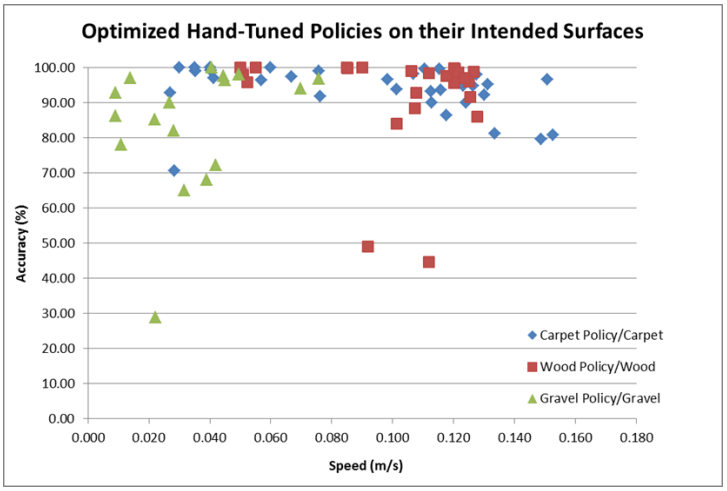

Fig. 6: Performance of OctoRoACH across different surfaces using open-loop policies tuned for those surfaces. The grouping of points in the upper regions of the plot indicates optimal policies were found.

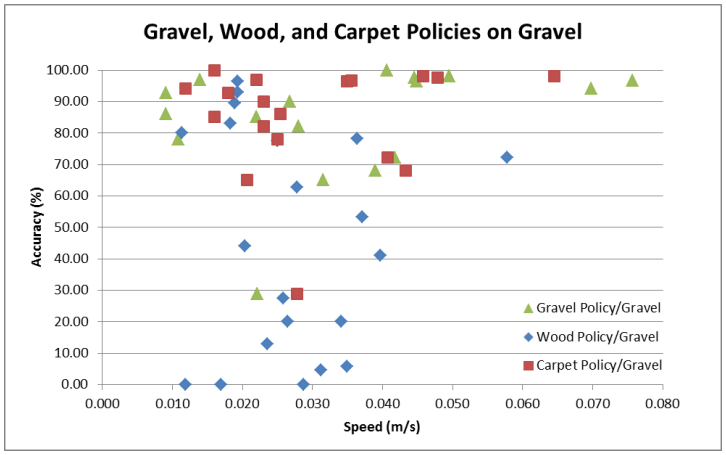

Fig. 7: Performance of OctoRoACH on multiple surfaces using open-loop policies which were not tuned for those surfaces. The wide spread of points indicates that policies did not perform well across surfaces.

Also, the open-loop tests provided insight into the relationship between speed and accuracy. We found that policies which gave the robot high speed also caused low accuracy, and policies with high accuracy caused low speed. This relationship motivated our choice of the harmonic mean as a reward function in the reinforcement learning algorithm because it gives a high reward to policies which maximize both speed and accuracy, rather than just one.

Because we found that no open-loop policy worked well across multiple surfaces, we integrated sensory

feedback into the control policy to understand if that could remedy the control policy. As expected, the integration of the rate-gyro sensor data increased the accuracy of the robot considerably, even across multiple terrains. However, because the closed-loop only provided feedback about the accuracy of the robot, it could not affect the speed at which the robot traveled. Figure 8 shows the performance of a closed-loop policy on the three different terrains; while the robot had high accuracy during all the runs, its speed decreased as the surface changed. Therefore, the closed-loop control policy provided an improvement over the open-loop policies, but did not solve the problem of robot performance across varying terrains.

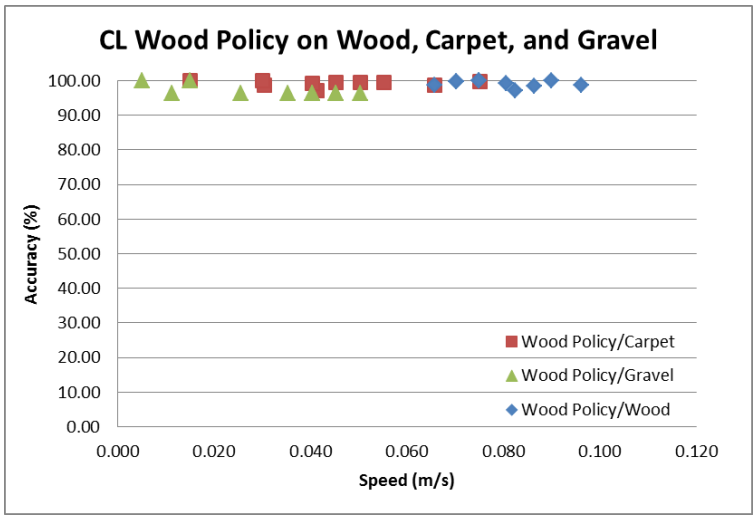

Fig. 8: Performance of OctoRoACH on multiple surfaces using a closed-loop steering policy

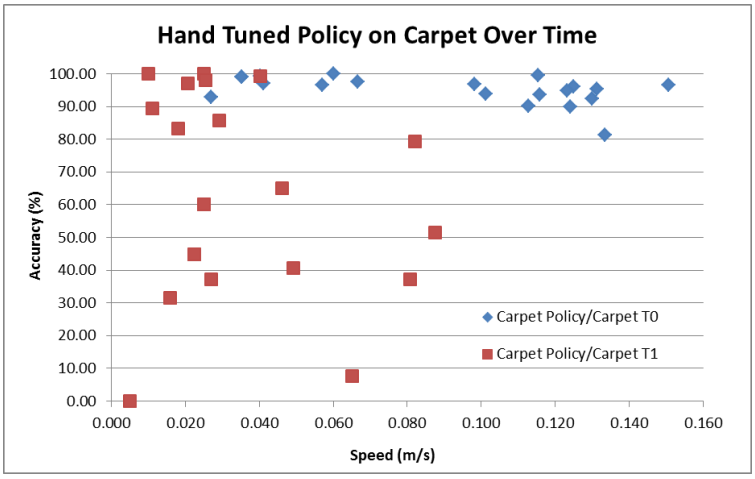

Fig. 9: Performance of OctoRoACH over time

Additionally, we can see in Figure 9 that mechanical wear on the robot significantly decreased its performance. At the initial time, policies chosen for carpet offered high performance for both speed and accuracy; however, after approximately three hours of continuous running, those same policies performed poorly on their intended surface.

Although we were able to apply the insights gained through tuning the OctoRoACH by hand to parts of the reinforcement learning implementation, we were not able to

successfully execute the policy gradient algorithm described above in Section III. As the algorithm iterated through policies chosen by following the gradient of the reward function, it chose policies which performed worse rather than better in terms of speed and accuracy. This result could be attributed to several factors of the experimentation. For instance, the robot used for the experiments had worn considerably from use and displayed increasing randomness in its functionality. Furthermore, our understanding of the robot's performance with respect to its control inputs may not have been as thorough as we believed, and may have affected the reward function we chose to optimize over. In order to more fully understand the robot's behavior, we performed an extended set of experiments to characterize the distance an OctoRoACH travels as a function of the motor thrust inputs. For these experiments, we varied motor input to the robot and tracked its position at a frequency of 100 Hz using a Vicon Motion Capture system. The results of these experiments can be seen in Figures 10 and 11. These plots indicate that the distances traveled on wood are similar to those traveled on carpet if the robot's left and right thrust values are increased by about 75 and 50 counts respectively. This relationship indicates that the policy search implementation may be reduced in dimensionality.

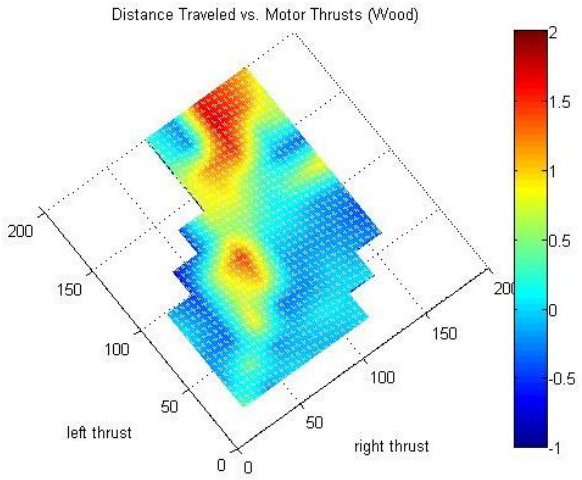

Fig. 10: Relationship between motor thrusts and forward distance traveled on wood surface

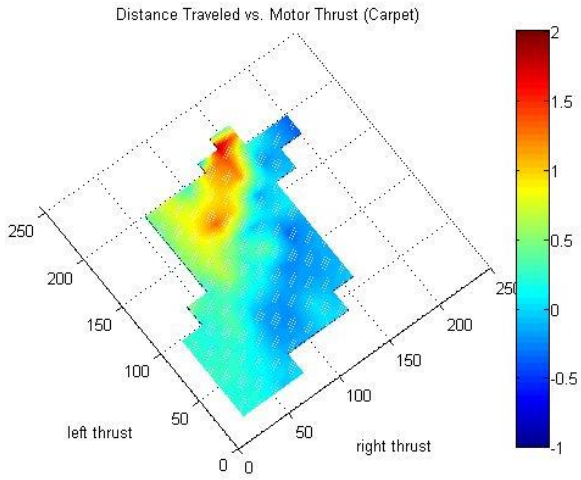

Fig. 11: Relationship between motor thrusts and forward distance traveled on carpet surface

V. CONCLUSIONS AND FUTURE WORK

Overall, it was found that reinforcement learning is a necessary tool in efficient locomotion for the OctoRoACH; however, more work is necessary to perfect its implementation. When tuning the robot's gait by hand, our initial hypotheses were validated. In an open-loop setting where the gyroscopes did not provide feedback to the controller, it was difficult to find a control policy which worked well on one robot for a given surface. This difficulty increased when trying to apply the same policy across different terrains. Therefore, no control policy worked well on all robots and across multiple terrains in the open-loop setting. Although adding sensory feedback made it easier to find a policy which worked well for one robot on a single surface, it did not solve the problem of finding a global policy which works well on multiple terrains.

Implementing an effective policy search method on OctoRoACH will require further understanding about the robot's behavior. As presented in the previous section, this understanding may begin with a characterization of the robot's behavior across different surfaces, as well as behavior across multiple robots. If consistent relationships can be determined, the policy search problem may be made less complex.

Although these results are interesting, there remains more to be learned about adaptive gaits on the OctoRoACH. First, other types of reinforcement learning could be better suited for OctoRoACH. Specifically, continuous learning should be implemented on the robot. Currently, the robot must perform several runs across a specific terrain before the control policy converges to an optimal policy. In order to perform better in a real world situation, the robot should be able to update the control policy as it performs its higher level tasks, such as exploring an area. As the robot's on-board sensing capabilities improve, this goal will become achievable.

References:

[1] A.W. Coburn, R.J.S. Spence, A. Pomonis, “Factors determining human casualty levels in earthquakes: Mortality prediction in building collapse,” Earthquake Engineering, Tenth World Conference, Balkema, Rotterdam, 1992.

[2] P. Birkmeyer, K. Peterson, and R. S. Fearing, “DASH: A dynamic 16g hexapedal robot,” in IEEE International Conference on Intelligent Robots and Systems, St. Louis, MO, 2009

[3] M. Hoover, E. Steltz, and R. S. Fearing, “Roach: An autonomous 2.4g crawling hexapod robot,” in IEEE International Conference on Intelligent Robots and Systems, Nice, France, Sept. 2008.

[4] A. Hoover, S. Burden, X.-Y. Fu, S. Sastry, and R. Fearing, “Bioinspired design and dynamic maneuverability of a minimally actuated six-legged robot,” in IEEE International Conference on Biomedical Robotics and Biomechatronics. BioRob 2010., Sep. 2010.

[5] A.O. Pullin, N.J. Kohut, D. Zarrouk, and R.S. Fearing, “Dynamic turning of 13cm robot comparing tail and differential drive,” to appear IEEE Int. Conf. Robotics and Automation, May 2012.

[6] J. Cham, J. Karpick, M. Cutkosky, “Stride Period Adaptation for a Biomimetic Running Hexapod,” International Symposium on Robotics Research, Lorne, Victoria, Nov. 2001

[7] U. Saranli, M. Buehler, D. Koditschek, “RHex: A Simple and Highly Mobile Hexapod Robot,” The International Journal of Robotics Research, vol. 20 no. 7, July 2001.

[8] J. M. Morrey, B. Lambrecht, A. D. Horchler, R. E. Ritzmann, and R. D. Quinn, “Highly mobile and robust small quadruped robots,” in Intl Conf on Intelligent Robots and Systems, vol. 1, 2003, pp. 82–87.

[9] N. Kohl, P. Stone, “Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion,” International Conference on Robotics and Automation, May 2004.

[10] J. Weingarten, M. Buehler, R. Groff, D. Koditschek, “Gait Generation and Optimization for Legged Robots,” IEEE Conference of Robotics and Automation, Sep. 2003.

[11] U. Saranli, D. Koditschek, “Template Based Control of Hexapedal Running,” IEEE International Conference on Robotics and Automation, Sep. 2003.