Using Kinesthetic Input to Overcome Obstacles with Snake Robots

Matthew Tesch, Alexander O'Neill, and Howie Choset
Robotics Institute, Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA, USA
mtesch@cmu.edu, oneill.alexander.b@gmail.com, choset@cmu.edu

Abstract — Snake robots have enormous potential to thread through tightly packed spaces and relay knowledge to search and rescue workers which is currently unattainable during the first hours of rescue operations. However, the existing approaches to snake robot locomotion in three dimensions is primarily limited to cyclic *gaits*, which lose effectiveness as the ratio of obstacle size to robot size or the irregularity of the environment increase. To this end, this work investigates a kinesthetic input approach to developing joint angle trajectories for overcoming these obstacles for which gaits are inadequate. The second contribution of this paper is the presentation and validation of a method to simplify these trajectories so that they can be easily stored, parameterized, and adjusted. Finally, we demonstrate that a simple sensor deviation filtering and thresholding approach can be used to quickly detect failure when overcoming an obstacle.

Keywords: *snake robot, kinesthetic input, search and rescue, parameterized trajectories, failure detection*

I. Introduction

In urban search and rescue applications, it is important to locate potential victims, and to do so without undue danger to rescue workers. In particular, this may require searching through unstable collapsed buildings, damaged infrastructure, and rubble. Robotic systems have the potential to be useful in such applications, because they can enter sooner (they need not wait until buildings have been declared safe enough for rescue workers) and they are smaller (enabling them to enter spaces where rescue workers and animals cannot fit).

In particular, the snake robots developed by Choset et. al [1] have notable benefits over other robotic systems. In particular, they have a very small cross-sectional diameter (just over 5 cm), theoretically enabling them to enter small crevices and void spaces in a collapse. In addition, they can articulate their long body to give an operator different vantage points of the scene, or to move over obstacles taller than the robot's diameter.

However, these robots (and to the authors' knowledge, other snake robots of the same form factor, such as [2], [3]) have as of yet had difficulty in overcoming such obstacles. Previously, the primary developments in the science of locomotion for these and other snake robots have been in *gaits*, or cyclic joint inputs that locomote the robots over fairly steady state terrain [4], [5]. The existing work on obstacle-aided snake locomotion [6], [7] involves two-dimensional setups which do not extend easily into three dimensions, and often require motion capture setups that are impractical for use outside of a laboratory setup. Other approaches for locomotion over obstacles have focused on different robot form factors ([8], [9]); these systems have different size, complexity, and mobility tradeoffs not addressed in this work.

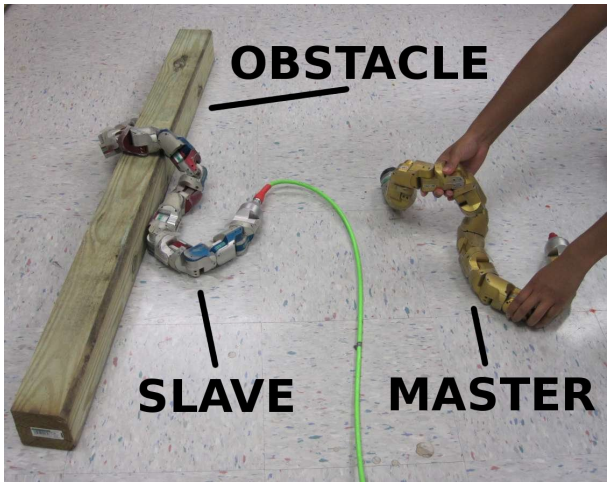
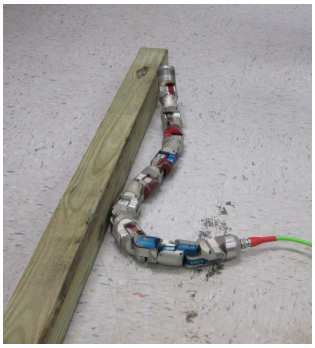


Fig. 1: In the kinesthetic control approach, a user moves an input snake (often termed the "master" or a "dolly"), and the controlled "slave" robot moves accordingly. We have found this to be an effective way to locomote the snake in challenging situations, and in this paper discuss how this information can be represented so as to most easily be reused and generalized to other tasks.

In this paper, we present a real-time master-slave "kinesthetic" control system for snake robots (see Figure 1), inspired by this idea for other robotic systems, such as large treaded articulated robots [10] or remote welding equipment [11]. This novel approach to overcoming obstacles with a snake robot allows us to learn open loop, acyclic commanded angle trajectories (non-gait motions) which move the robot over these obstacles and can later be replayed when these obstacles are encountered in the field.

In addition to extending the locomotive capabilities of these mechanisms, we also present a novel framework for analyzing and parameterizing these trajectories to enable **(a)** easier storage and implementation, **(b)** more intuitive understanding (and therefore tuning) of the trajectories, **(c)** learning methods



(a) 4x4 Beam (3.5x3.5 inches)

(b) Ledge (15 cm tall)

Fig. 2: The two obstacles used for demonstration of the kinesthetic input methods and subsequent trajectory simplification. Both of these represent obstacles that we have encountered during tests at TEEX Disaster City®, and were unable to overcome with the existing gait-based control methods. Also shown in these images are the starting positions used for the trials. The goals were to **(a)** move the robot completely over the 4x4 and **(b)** to move the robot so it is only supported by the ledge, without touching the ground.

to optimize these trajectories for robustness, and **(d)** generalization of these trajectories to a range of similar obstacles.

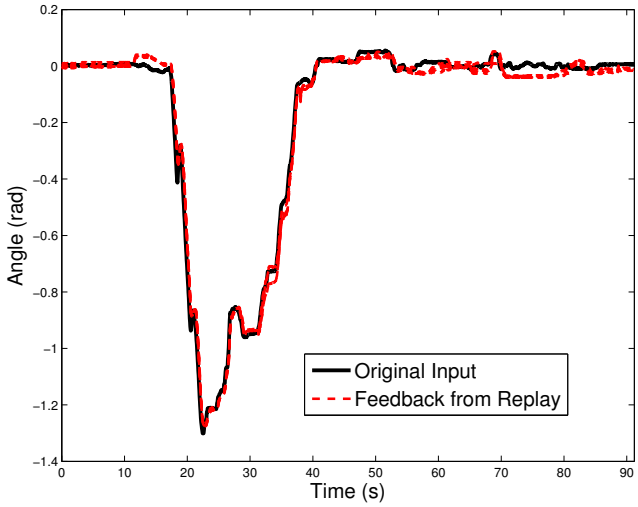
Finally, we show that through simple methods we can detect failure to overcome the obstacle, even before the replay of the trajectory is complete. This could be used to stop the playback and take corrective action or alert the operator.

The authors believe that the capabilities demonstrated here are only the first step to unlocking the full potential of snake robot control using kinesthetic input methods. We envision that the techniques we present can be used along with optimization algorithms to generate robust, adaptive, failure-aware controllers to improve one of the critical weaknesses of using snake robots for search and rescue – the ability to overcome obstacles. However, we believe even in their present form, these methods can increase the usefulness of these robots for searching damaged buildings and infrastructure.

II. Collecting Data through Kinesthetic Input

The kinesthetic input approach to data collection allows for dynamic collection of data through direct input and playback. This is enabled by the controllable motor current limits on our snake robots. When these limits are turned to zero, the snake is easily backdriveable. Because the modules are still on, the magnetic encoder continues to provide feedback.

This mode of input can be simultaneous - using one snake as a master, sending commanded angles to a slave snake - or pre-recorded - using one snake as both master and slave by recording a motion, resetting the snake to the original position of input, and playing back the most recent recording. An example of the simultaneous approach is shown in Figure 1, where an operator is steering the robot over an obstacle by moving a different robot through the desired shapes. When the recorded motions are played back on the snake, the commanded angles and sensor feedback values are stored in a log file which can later be analyzed in MATLAB.



(a) Module 7, "Head to Tail" approach to a 4x4

Fig. 3: The reference trajectory for a single module in one trial is shown as the solid black line; the dotted red lines show the joint angle feedback captured when playing back this reference trajectory on a robot attempting to move over an obstacle. This illustrates typical deviation from commanded angle trajectories.

We chose two obstacles for this study, a 4x4[1] solid wooden beam, illustrated in Figure 2a, and a 15.24 cm (6 inch) tall ledge, illustrated in Figure 2b, each of which represented a unique challenge. We had previously encountered both of these obstacles in some form during disaster response practice scenarios at TEEX Disaster City®. We were unable to make progress over these obstacles using our existing controllers in the field, but later had success using kinesthetic approaches in the lab to move over them.

To generate a corpus of data for this paper, we created five different approaches for climbing each of these two obstacles. Using the kinesthetic input approach, we saved a reference trajectory of the joint angles for each approach on each obstacle. The robot then attempted 25 trials on each obstacle, replaying each recorded reference trajectory five times to generate unique outputs. A typical example of the consistency of replaying the reference trajectory is shown in Figure 3.

At each obstacle the robot started in a common position, illustrated in Figure 2. A trial was considered a success if the robot moved completely over the beam or into a position where it was only supported by the ledge.

It should be noted that although these replays were run at real time for the purpose of this data collection, we are able to replay these faster; a speedup factor of 10x was attempted a number of times and presented no issues.

The data was recorded in a table, consisting of the type of obstacle, the method to overcome the obstacle, the trial number for the snake, the success of the trial and any notes concerning the trial. A summary of this data is shown in Table I.

[1]A 4x4 is a standard US dimensional lumber size measuring 3.5 x 3.5 inches in cross section.

TABLE I: Results of Replaying Recorded Trajectories

Obstacle	Approach	Successes/Trials
4x4	Flop	4/5
	Roll	4/4
	Head to Tail Progression	5/5
	Tail to Head Progression	5/5
	Pinch	5/5
15 cm Ledge	Flop	5/5
	Roll	4/5
	Head to Tail Progression	4/5
	Tail to Head Progression	5/5
	Pinch	5/5

The basic approaches taken were the following:

- **Flop:** Lift the snake's head up high, and then flop it down onto the obstacle.
- **Roll:** Roll the snake onto the obstacle. Main method of movement is rolling.
- **Head to Tail:** Progress a kink through the snake from the head to the tail to get over the obstacle.
- **Tail to Head:** Progress a kink through the snake from the head to the tail to get over the obstacle.
- **Pinch:** Lift up the middle first then the sides.

III. Low Dimensional Parameterization of Data

The data collection methods in the previous section have proved useful, enabling repeatable locomotion of the snake over the two selected obstacles. However, in real search and rescue scenarios there will be variation in the terrain and obstacles, leading to the need to extend and improve the trajectories that are recorded using kinesthetic input. We seek to increase their robustness (percentage of successful trials) in the presence of noise and small variations. We also hope to generalize these joint angle trajectories so that they can adapt to a wider range of obstacles – ledges at different heights, for example.

One difficulty with optimizing the controllers for robustness or identifying parameters that enable adaptation to different size obstacles is the large number of variables required to represent the controller. The recorded inputs from the slave snake in these trials is simply a list of joint angles, received at around 20Hz over a period of 60 to 180 seconds. This leads to a matrix of size k x n, where k is the number of joints (16 for the tests conducted here), and n is the number of timesteps (usually 2000 to 3000).

We seek to condense this large number of data points to a smaller representation that is also more understandable. This process does not have to enable a perfect reconstruction of the input, but should capture the intent of the original so that the simplified trajectory can still be used, without loss of effectiveness, to move the robot over an obstacle. Ideally, this processing would also remove unneccessary noise and jitter introduced while recording the input, potentially even improving the effectiveness of the paths. Furthermore, having a simpler parameterized description of the trajectories enables them to be more easily stored, used, and compared.

A. Method

We propose the following method to produce such a parameterized reduction of the joint angle trajectories generated in the previous section. First, we define the form of the reduction as ordered endpoints for a piecewise linear function, one set for each joint on the robot. When obtaining this reduction, we wish to minimize the number of endpoints needed, while maintaining some notion of representational error below a given threshold (ensuring we maintain a reasonable level of representational fidelity to the original data).

To find the parameters of the reduction for a given joint, we initially sample points that densely interpolate that joint's trajectory and use these as the endpoints for the piecewise linear function. Incrementally points are removed, selecting at each step to remove the point resulting in the smallest increase in error. This is repeated until a specified error threshold has been reached.

Optionally, the fidelity of the resulting piecewise linear function is improved by a secondary optimization step. Each remaining point is allowed to vary in time and angle, relaxing the initial assumption that these points exactly interpolate the data. Our implementation involved a simple gradient descent on the representational error. This optimization is done one point at a time, but cycles through all points until convergence is reached. Figure 4 shows the result of both the reduction and the optional optimization step for three different joint trajectories, using a threshold of $.004\,\text{rad}^2$ for each.

More formally, let the original input trajectory be defined by a vector of timestamps $T = [t_1, t_2, ...t_n]^T$, and the joint angles corresponding to these timestamps as

$$\Theta = \begin{bmatrix} \theta_1^1 & \dots & \theta_1^k \\ \vdots & \ddots & \vdots \\ \theta_n^1 & \dots & \theta_n^k \end{bmatrix}$$

We use the convention of subscripts for timestep index, and superscripts for joint number index.

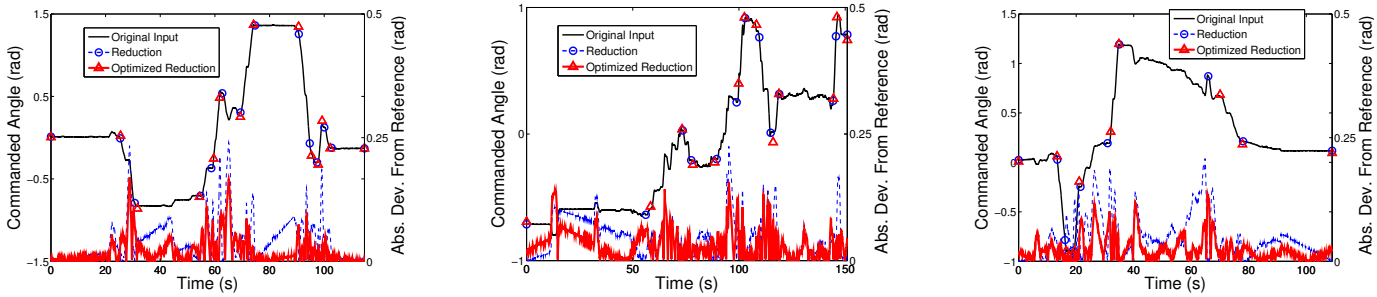
Our lower dimensional reduced trajectory can be defined as $\hat{\Theta} = \{(\hat{t}^i, \hat{\theta}^i) \,|\, 1 \leq i \leq k\}$. $\hat{t}^i$ is a column vector of the times for the endpoints of the piecewise segments for joint i, and $\hat{\theta}^i$ is a column vector with the corresponding joint angle at each of these times.

A slight overloading of notation allows us to use $\hat{\Theta}$ as a function as well; in this case, $\hat{\Theta}(t, i)$ returns the angle produced by this reduced trajectory for time t and joint i. This can be computed via linear interpolation between the surrounding points.

Given this notation, we can define the mean-squared error of our the reduced representation as the normalized sum of squared distances from the original input, or

$$\text{error} = \frac{1}{kn} \sum_{i=1}^{k} \sum_{t=1}^{n} \left(\hat{\Theta}(T(t), i) - \Theta(t, i) \right)^2, \qquad (1)$$

where parenthesis are overloaded to index into the matrices T and Θ in row-major order. Given the more rigorous definition



(a) Module 5, "Pinch" approach to a 4x4

(b) Module 12, "Roll" approach to a 15 cm ledge

(c) Module 3, "Flop" approach to a 4x4

Fig. 4: These plots show the fidelity of a piecewise linear approximation to the original kinesthetic input for three randomly selected joint angle trajectories. The original trajectory is shown by the solid black line. Simply reducing from a dense interpolation of the input (here, the mean-squared error threshold was $0.004\,\text{rad}^2$) gives a fair approximation to the input; the circular markers indicate the linear segment endpoints and the dotted blue line indicates the deviation of this approximation from the original (right axis scale). The secondary gradient descent optimization over the linear segment endpoints provides a noticable improvement (triangular segment endpoints and red solid line for deviation). Note that by moving to a piecewise linear parameterization, the amount of information that needs to be stored for these trajectories reduce from 2774, 3652, and 2636 points in $\mathbb{R}^2$ to 14, 13, and 10, respectively.

of the error function, the reduction and tuning/optimization step can now be completed as described above.

B. Summary

Overall, this reduction was run on each of the 10 recorded trajectories (five approaches each to overcoming two obstacles), and has resulted in typical reductions from initial T of around 3000 points, and Θ that are 3000 x 16, to $\hat{\Theta}$ reductions that are typically defined by around $10-14$ time/angle pairs for each of the 16 modules. This amounts to a typical reduction factor around 120, resulting in around 360 parameters (around 24 per joint).

While typical reductions of $3000 \times 16 = 48000$ values to around 360 is significant, this obviously still results in a fairly high dimensional space. However, in many cases the actual values that one might modify for optimization would be much lower. One might assume timestamps to be fixed, reducing to 180 parameters. If only certain modules are of interest, then this might further reduce to 45. Additionally, one can couple keyframes across multiple modules, tolerate increased error, use wavelet decomposition, form a grammar from common symbols, or apply other analysis with expert knowledge to decrease the number of parameters for optimization into a reasonable range $(10-30)$, depending on the exact task.

The exact processes which this further reduction would involve are likely to be highly task dependent. We leave this as an area of future work, but remain convinced that the reductions demonstrated here are a necessary first step in generalizing and best using the kinesthetic input that has recently proved so effective.

IV. Testing Reduced Trajectories

To validate the usefulness of the reduction methods described above, we must test the resulting joint trajectories to verify that they can actually accomplish the task with a similar rate of success as compared to the originals.

TABLE II: Results of Simple Parameterized Trajectory Controllers

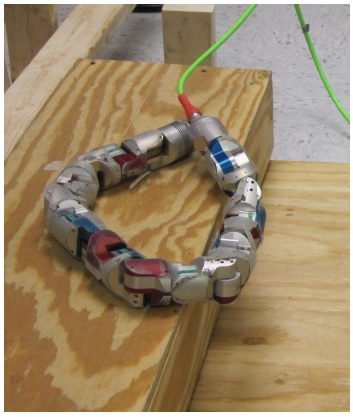
Obstacle	Approach	Successes/Trials		
		Original	Reduced	Optimized
4x4	Flop	4/5	5/5	5/5
15 cm Ledge	Roll	4/5	5/5	5/5
15 cm Ledge	Head to Tail	4/5	4/5	4/5

For a optimization threshold of $0.004\,\text{rad}^2$, Table II shows the performance of the reductions as compared to the original commanded trajectories for three selected obstacle/approach combinations. In no case is there a decrease in rate of success; in fact the rate of success increases for two obstacles. We hypothesize that this increase is due to a reduction in the high frequency signal (mostly noise during the original data collection), resulting in more repeatable motions.

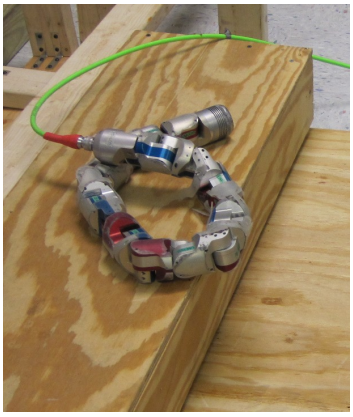
Interestingly, these results do not show a noticable improvement between the simple reduction and the extra optimization step. However, while running the tests there was indication that this optimization significantly increased the faithfulness of the representation. In Figure 5, the ending position for the "Head to Tail" approach to moving onto a ledge is compared for the original trajectory, the reduction, and the optimized reduction. In both the original and optimized reduction, there is very little of the robot hanging over the ledge, whereas the simple reduction leaves considerable more of the robot over the ledge.

Although more conclusive tests will be done in future work, these results give strong supporting evidence that the given reduction and parameterization methods result in simple, effective controllers for overcoming obstacles. In particular, these results indicate that there is no noticable degredation in performance between replaying the original trajectories and the piecewise linear optimized reductions.

(a) Original Controller



(b) Initial Reduction



(c) Optimized Reduction

Fig. 5: The ending position for the "Head to Tail" approach to climbing on top of a six inch ledge. The optimized reduction is noticably more faithful to the original controller, as indicated by the fewer modules overhanging the ledge resulting in a more stable ending configuration.

V. Detecting Controller Failure

When operating the robots in a true rescue or response scenario, it is critical to reduce the workload of the operator. To this end, it is important that the operator can issue a high-level command, such as "move forward over this obstacle", and the robot will take care of the execution without requiring human supervision. This split autonomy requires the ability to detect and respond to failures in the controllers.

A simple approach to this failure detection is to compare the sensor data feedback from the robot to expected sensor data output. In this section, we demonstrate the effectiveness of such an approach.

In Figure 6, we have compared the x, y, and z accelerometer data from a single module on the robot for a successful versus an unsuccessful trial, as well as from two successful trials. At one point (around 90 seconds) in the unsuccessful trial a marked deviation from the reference occurs.

Noting this occurance, a simple detection scheme can be invoked. The acceleration signal for an active trial is continually monitored, and passed through a simple 5 timestep averaging window. The resulting vector in $\mathbb{R}^3$ is subtracted from the expected acceleration vector at that timestep for a recorded, low-pass filtered successful trial. The magnitude of this difference gives an error reading. Whenever this error reading surpasses a set threshold, failure has been detected and appropriate responsive action can be taken. Figure 7 shows this error for two pairs of failed and successful trials encountered in the initial data collection.

Of course, this simple method leaves room for improvement, but demonstrates the efficacy of such an approach. In particular, sensor data from more modules can be used to detect deviation more quickly and robustly.

In addition, once failure has been detected the system must appropriately respond to this event. This response is outside the scope of this paper, but is an area of active research to increase the readiness of these robots for the purpose of actual rescue deployments. Note that other sensors, such as tactile sensors, could be used similarly if available.

VI. Conclusion and Future Work

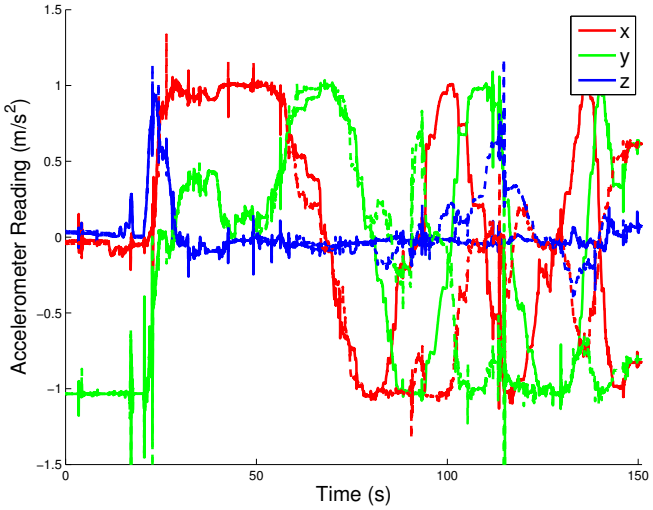
The methods described in this paper have demonstrated a path forward for improving snake robot locomotive performance in the presence of obstacles. This involved using a master-slave *kinesthetic* setup to generate desired joint angle trajectories, a reduction/parameterization step to simplify these trajectories, and a failure detection scheme. In addition to developing these tools, we have increased the capabilities of our robot by creating a set of robust approaches for overcoming two different obstacles.

Given the success of these preliminary findings, our future work is focused on extending them so that they can readily be used in a fieldable system. In particular, we will (1) extend the simplification to further reduce the parameterization via methods such as wavelet analysis, (2) use this reduction to compare and generalize trajectories of the same approach over a parameterized family of obstacles, (3) optimize over this parameter space to improve robustness of the trajectories (especially with respect to initial position relative to the obstacle), and (4) extend the failure detection scheme to close the loop and apply corrective action in real-time. Other obstacles we will investigate include those commonly found in USAR scenarios, such as gaps, narrow conduits, pipe junctions, etc.

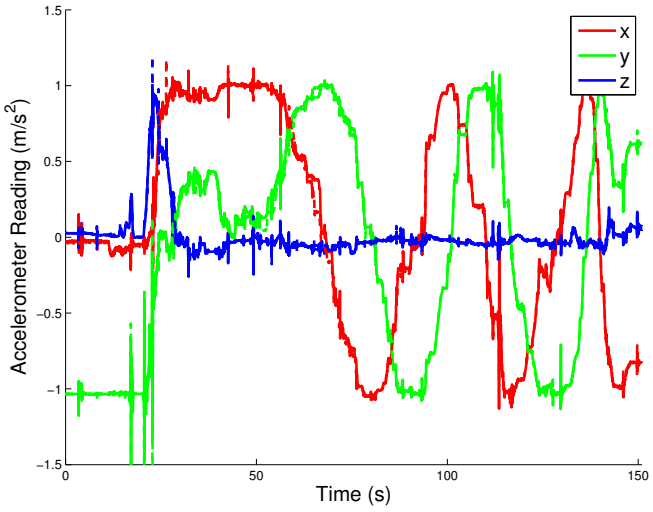
Finally, the driving goal of this work is to create a system which can be used to help rescue workers extend their reach. In the future we hope to integrate these solutions into an easy-to-use semi-autonomous system which reduces the mental workload of the operator, while effectively locomoting over difficult terrain.

Acknowledgment

The authors would like to thank all of the members of the Biorobotics lab at CMU, especially the work of Florian Enner and Dave Rollinson who contributed key software

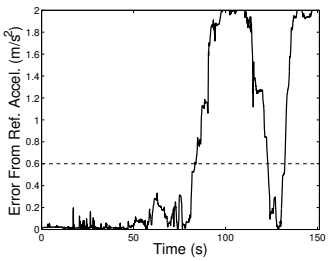


(a) Failed Trial

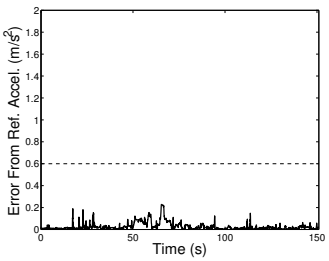


(b) Successful Trial

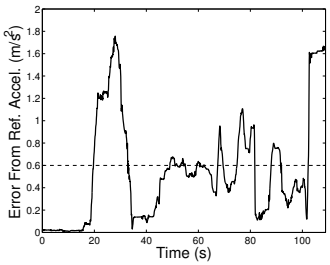
Fig. 6: Three-axis accelerometer data from the first module in the snake robot while attempting to climb onto a 15 cm ledge using the "roll" method. On the top, we compare our baseline trial (number 2 of 5) to the failed trial on this obstacle (3 of 5). The solid line represents the baseline values and the dotted line represents the failed trial; a clear separation can be seen around 80-90 seconds into the trial. On the bottom, the baseline is compared to a succesful trial (4 of 5). This results in a much better match.



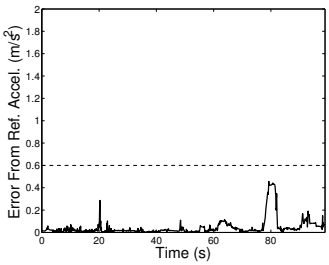
(a) Failed "Roll" onto Ledge



(b) Successful "Roll" onto Ledge



(c) Failed "Flop" over 4x4



(d) Successful "Flop" over 4x4

Fig. 7: Computing the magnitude of the difference in the accelerometer vectors generates a measure of the error from the expected sensor readings. If this signal goes above a simple threshold (dotted line), we can detect that the trial is likely to have failed. The top row compares this error from a failed trial with that of a successful trial; these correspond to the same trials as shown in Figure 6. The lower row presents this error signal for the "flop" approach to moving over a 4x4. The baseline is trial 4, and the error signal is shown for the failed trial 5 and successful trial 3. The baseline, successful, and failed trials were randomly selected from these obstacles; other trials showed similar results. Intuitively, as accelerometers provide a drift-free estimate of pitch and roll, failure is likely to correspond with a significant change in pitch or roll from that observed in the reference trajectory.

developments to make this research possible, Glenn Wagner who provide useful insights, and Priya Deo who was crucial in the process of creating the figures in this paper.

References

[1] C. Wright, A. Buchan, B. Brown, J. Geist, M. Schwerin, D. Rollinson, M. Tesch, and H. Choset, "Design and Architecture of the Unified Modular Snake Robot," in *2012 IEEE International Conference on Robotics and Automation*, (St. Paul, MN), 2012.

[2] H. Yamada and S. Hirose, "Study on the 3D shape of active cord mechanism," *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2890–2895.

[3] H. Ohno and S. Hirose, "Design of slim slime robot and its gait of locomotion," *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems.*, pp. 707–715, 2001.

[4] M. Tesch, K. Lipkin, I. Brown, R. Hatton, A. Peck, J. Rembisz, and H. Choset, "Parameterized and Scripted Gaits for Modular Snake Robots," *Advanced Robotics*, vol. 23, pp. 1131–1158, June 2009.

[5] J. Gonzalez-Gomez, H. Zhang, E. Boemo, and J. Zhang, "Locomotion capabilities of a modular robot with eight pitch-yaw-connecting modules," in *9th international conference on climbing and walking robots*, Citeseer, 2006.

[6] P. Liljeback, K. Pettersen, O. Stavdahl, and J. Gravdahl, "Experimental Investigation of Obstacle-Aided Locomotion With a Snake Robot," *Robotics, IEEE Transactions on Robotics*, vol. PP, no. 99, pp. 1–8, 2011.

[7] P. Liljeback, K. Y. Pettersen, O. y. Stavdahl, and J. T. Gravdahl, "A hybrid model of obstacle-aided snake robot locomotion," in *2010 IEEE International Conference on Robotics and Automation*, pp. 675–682, IEEE, May 2010.

[8] K. Hatazaki, M. Konyo, K. Isaki, S. Tadokoro, and F. Takemura, "Active scope camera for urban search and rescue," *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2596–2602, Oct. 2007.

[9] M. Neumann, P. Labenda, T. Predki, and L. Heckes, "Snake-like, tracked, mobile robot with active flippers for urban search-and-rescue tasks," in *15th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR)*, 2012.

[10] J. Baker and J. Borenstein, "The Joysnake A Haptic Operator Console for High-Degreeof-Freedom Robots," in *Robotics*, pp. 12–15, 2006.

[11] T.-J. Tarn, S.-B. Chen, and G. Fang, eds., *Robotic Welding, Intelligence and Automation*, vol. 88 of *Lecture Notes in Electrical Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.