Exploiting the Dynamics of Tensegrities Via Morphological Computation

James M. Boggs John Rieffel (Advisor)
Computer Science Department, Union College

Abstract

Tensegrity robots are a class of soft robots which are comprised of rigid struts connected by springs under tension in such a way as to maintain a flexible resting form. These robots have dynamic and complex physical properties which make it impossible to predict how an actuation will affect the system. We are attempting to utilize, rather than mitigate, this physical dynamism by using morphological computation, which treats the physical system as implicitly performing computations. Specifically, we will implement a simple spiking neuron on each strut and treat the system as a whole as a spiking neural network, with communication between neurons implicit in the vibrational activity of the strut and looking for emergent behavior. Ultimately, we hope that the physical system can be used to produce a robust central pattern generator to generate an effective forward gait for the robot.

Introduction

Tensegrity robots are composed of rigid struts connected to each other by tensile links (usually springs) such that the opposing tensile forces of the links result in the structure taking on and maintaining a flexible form. Union's VVValtr, seen in Figure 1, is an example of a tensegrity robot.

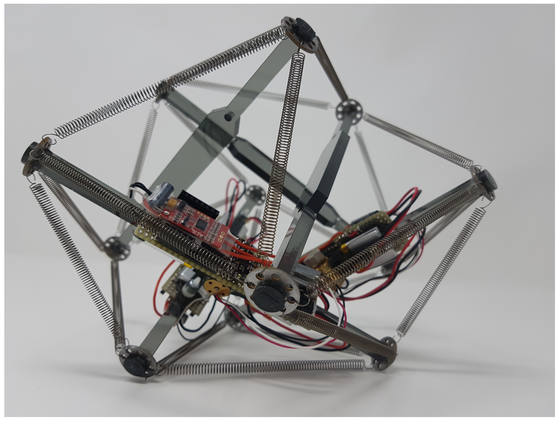

Figure 1: A roughly life-sized picture of our current tensegrity robot, VVValtr (VVireless Vibrationally Actuated Tensegrity Robot.

We use vibration to induce locomotion, which has been shown to be effective in [1]. We will use the complex physical dynamics of the tensegrity to perform implicit computations, as described and shown in [2] and [3]. In particular, we will use morphological computation to run a spiking neural network implementation of a central pattern generator. This will produce a steady pattern of vibrations which will hopefully move the tensegrity forward effectively.

Goal

We will demonstrate that the complex physical dynamics of a tensegrity system can be used to offload computational load, to allow for the decentralization of a control algorithm, and to add robustness to the control algorithm. Specifically, we will be treating each strut as a independent spiking neuron, and we will attempt to entrain a CPG in the tensegrity as a whole. Thus, the neural network itself will not be running on any computer, reducing computation and improving the robustness of the control system by decentralizing it. Additionally, we suspect that even after being perturbed by an obstacle, the system will return to a stable and effective gate since the tensegrity itself will return to its resting physical state.

Methods

We will design a simple spiking neuron in `Arduino`, which takes as input acceleration and gyroscopic motion in the x-, y-, and z-axes and output a frequency for the attached motor to spin at. The code will also listen for Bluetooth communication which can start or stop the neuron and adjust its weights. Each strut with a motor and controller will be running this code. A `Python` program running on the central computer will evolve effective weights for each neuron (i.e., strut) by running the neuron for a short period of time and tracking the movement of the tensegrity as a whole, as seen in Figure 2.

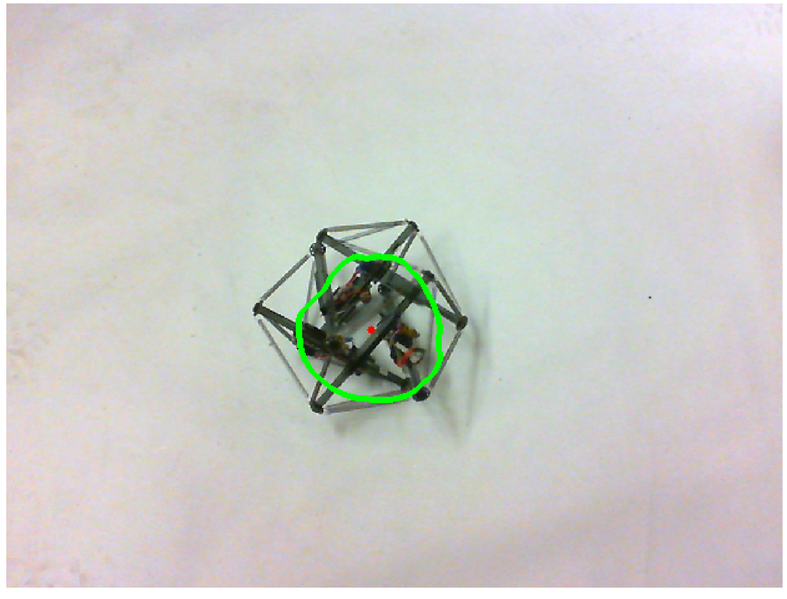

Figure 2: The view of the tracking code looking at the tensegrity. The code thresholds a black and white image to find the tensegrity, the reports the center.

Current Progress and Next Steps

We have finished designing and building wireless tensegrities with independent strut controllers. Next term we will first use this to get data on the effectiveness of other methods of generating tensegrity gaits such as genetic algorithms and Bayesian optimization. Next, we will implement the neuron code in `Arduino` and write `Python` experimental code capable of evolving the neurons' weights, then test this code and see how it compares to previous attempts in terms of its ability to generate an effective forward gait. Finally, if we can find a set of neuron weights which produce an effective gait, we will test the robot's ability to handle obstacles in its path. The neuron code will look something like Figure 3.

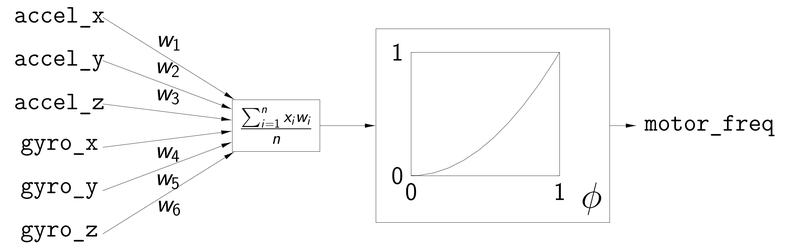

Figure 3: Diagram of the expected neuron code for each strut. Acceleration and gyroscopic data is weighted and averaged, and a linear function will output an appropriate motor frequency.

Acknowledgments

I would like to thank Kentaro Barhydt, Mitchell Clifford, Tommy Sipple, and Katerina Petridou for designing and building VVValtr.

References

[1] M. Khazanov, B. Humphreys, W. D. Keat, and J. Rieffel, "Exploiting dynamical complexity in a physical tensegrity robot to achieve locomotion.," in *ECAL 2013*, pp. 965–972, Union College, 2013.

[2] J. A. Rieffel, F. J. Valero-Cuevas, and H. Lipson, "Morphological communication: exploiting coupled dynamics in a complex mechanical structure to achieve locomotion," *Journal of the royal society interface*, vol. 7, no. 45, pp. 613–621, 2010.

[3] K. Caluwaerts, M. D'Haene, D. Verstraeten, and B. Schrauwen, "Locomotion without a brain: physical reservoir computing in tensegrity structures," *Artificial life*, vol. 19, no. 1, pp. 35–66, 2013.

[4] C. Paul, F. J. Valero-Cuevas, and H. Lipson, "Design and control of tensegrity robots for locomotion," *IEEE Transactions on Robotics*, vol. 22, no. 5, pp. 944–957, 2006.

[5] A. Agogino, V. SunSpiral, and D. Atkinson, "Super ball bot-structures for planetary landing and exploration," *NASA Innovative Advanced Concepts (NIAC) Program, Final Report*, 2013.

[6] A. Iscen, A. Agogino, V. SunSpiral, and K. Tumer, "Controlling tensegrity robots through evolution," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pp. 1293–1300, ACM, 2013.

[7] A. Iscen, A. Agogino, V. SunSpiral, and K. Tumer, "Flop and roll: Learning robust goal-directed locomotion for a tensegrity robot," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 2236–2243, IEEE, 2014.