Journal of UAB ECTC
Volume 16, 2017
Department of Mechanical Engineering
The University of Alabama, Birmingham
Birmingham, Alabama USA

SURFACE IDENTIFICATION INSPIRED BY REINFORCEMENT LEARNING USING SOFT ROBOTS

Miranda Tanouye
University of Alabama
Tuscaloosa, Alabama, USA

Vishesh Vikas
University of Alabama
Tuscaloosa, Alabama, USA

ABSTRACT

A terrestrial soft robots' dynamic movement can be discretized and modelled using graph theory for different actuation or material. This representation allows for learning from the environment and generating control sequences to manipulate the robots' movements. Inversely, provided the knowledge of different environments (captured in the individual graphs), the robot has the ability to optimally identify the environment through experimentation and interaction. This paper presents a method for soft robots to process the information from known environments and apply it to new ones to optimize their movement. The soft robot comprises multiple limbs with accelerometers attached to the robot. In real time, data from the accelerometers can be compared to stored data. If all the data from each limb falls within a threshold for a known environment, then the robot is acting optimally based on previous experimentation for that environment. If not, then the robot may be in a new environment. Following a predetermined path, data can be collected from the sensors and compared to known data. From that, the robot may readjust based off the known data and retest to see the new reaction to the environment.

KEYWORDS: soft robotics, environment identification, graph theory, control

INTRODUCTION

Soft robotics is a growing field with many applications from manufacturing to health care. The design, manipulation, and control of these soft robots is very difficult due to their deformability and continuum nature, which makes them difficult to model. While their intrinsic nature makes them difficult, it also gives them extreme potential for instances where classic rigid body robots cannot be adapted, such as navigating structurally unstructured environments. This adaptability can be exploited to learn about the environment by comparing the robot-environment interaction with prior knowledge.

Soft robot modeling has been performed using continuum modeling [1]–[3] and fast-FEM numerical methods [4] for soft manipulator control. Modeling of environmental interaction has also been researched [5]; however, it is computationally expensive and very difficult for unstructured environments. Finally, the modeling of soft robot actuators – ranging from pneumatic actuation [6], [7], motor-tendon [8], [9] and shape memory alloys [10], is very difficult. A counter approach is to find motion primitives [11] in task space – discretize the factors that dominate the robot-environment interaction. Locomotion can be viewed as optimization of forces acting on a robot – maximizing them at one end and minimizing at the other to effect movement [12]. Hence, in principle, locomotion can be effected by manipulating 'relative' friction between two ends of a robot, rather than the absolute friction of each end. While analyzing robot locomotion, the factors that dominate robot-environment interaction primarily involve friction-manipulation. Friction manipulation has been implemented in soft material robots through relative changes in the frictional coefficients of materials of contact or directional friction [9], [10].

Quantification of the environment has not been explored by observing the interaction between a soft material robot and the surface/environment. A model-free control approach [13] for soft robots utilizes the feedback from robot-environment interaction for obtaining periodic control sequences to effect desired locomotion on a surface. The most interesting aspect of this approach is the adaptability; i.e. due to use of graph theory and discretization of motion primitives, the robot is able to learn about the "new environment" and is even fault tolerant—it can handle loss-of-limb scenarios. The presented research uses the inverse approach to learn about the environment – *given the knowledge (graph) for different environments, what is the environment the current robot is navigating?*

The paper is structured as follows – design and manufacturing of the SMA-actuated soft robot (Section 2), model-free control and graph theory (Section 3), experimental data and discussion about results (Section 4).

MANUFACTURING

The soft robot used in this study was designed to be small and modular with a tube-shaped body. Modularity is implemented by having interchangeable end pieces, from here on called grippers, to provide a variety of friction interactions between the environment and the robot. This is important for the robot's locomotion as it can perform more effectively in an environment based on the type of gripper attachment. It is also modular as several two limbed bodies can be joined at a common center piece to quickly create a multi-limbed robot. For this study however, the robot has a two limbed body with two modular grippers. Further studies will be conducted later for modular multi-limbed bodies.

Casting was the chosen manufacturing method because it is fast and cheap. The two main casting materials are the mold and casting material. A mold can quickly be designed using any solid modeling software and then 3D printed allowing for quick robot design or mold modifications as well as being reusable. The casting material is Smooth On Dragon Skin 10. The size of the robots also means that several can be produced with an extremely small amount of material, saving costs. Figure 1 shows the manufacturing process of the robot body.

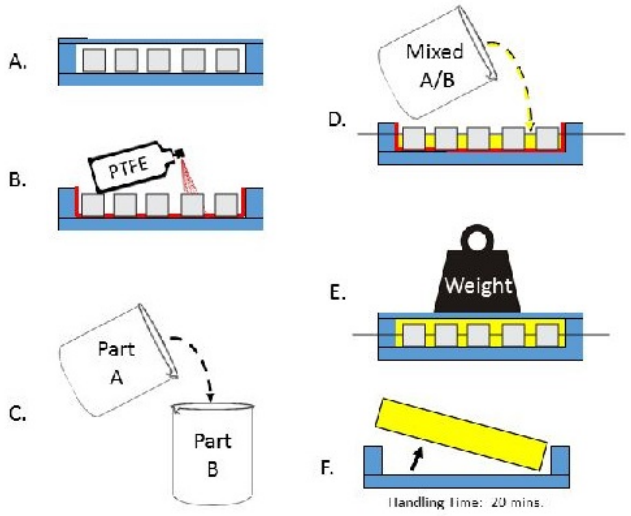

Figure 1. Manufacturing process of the soft robot

The mold has two components, the base and press. The base holds most of the material and forms the top half of the body. The press portion forms the legs of the main body. These legs are inspired by soft linear creatures such as caterpillars and centipedes. They provide grip as well as friction manipulation. Once the mold is printed and ready three metal rods, one 17 gauge and two 20 gauge, are inserted through the mold. Steps A to D in Figure 1 show the Dragon Skin 10 being prepared and then poured into the base. The press is then pushed down into the excess material as allowed by design, shown as step E. Once dry, the three rods and press are removed and then the final body is complete and removed as demonstrated by step F. Each rod creates a channel through the top portion of the body, set in a triangular pattern with the two smaller rods on bottom. Wires are run through the larger channel and shape memory alloy is placed in the two smaller channels to act as actuators. Rods were chosen to create these channels because of their placement in the body. The rods are reusable and easy to install and remove while casting. Other materials were tested, but were difficult to remove after the silicone cured.

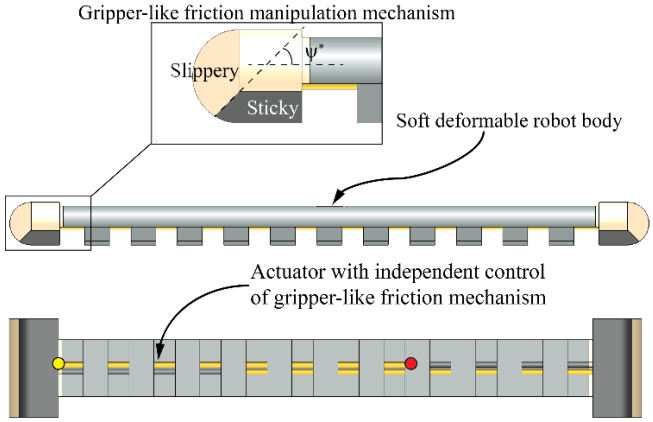

Figure 2. Two-limbed robot body where each end comprise of two different materials to manipulate frictional interaction with the surface of contact

GRAPH THEORY

Locomotion of this robot is modeled using graph theory. A graph for this study is a set of vertices, or nodes, with connecting edges. Each node represents a different binary state of the robot in question. The number of states (n) for this system is defined as

$$n = b^m$$

where b is the number of behaviors and m is the number of grippers

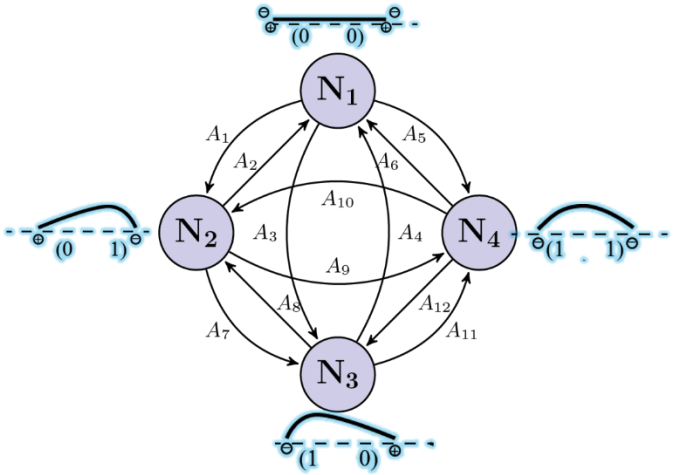

Figure 3. The directed graph indicating the robot shapes/states and the corresponding nodes. The arcs indicate the transition between these robot states.

Each gripper has two possible behaviors, on (0) or off (1). For a two limbed robot, such as in this study, there are two grippers meaning that there are four possible states which can be represented as (0,0), (1,0), (0,1), and (1,1). The edges connecting each node represent the transition between states, or in other terms, the movement of the robot. Since it is possible for the robot to move from one node to any other node except itself, the graph is considered complete and directed. By analyzing each edge and seeing its effect on the locomotion of the robot we can apply weights to them based on performance. Figure 2 shows a drawing of this graph as well as the states at each node.

If the goal is to move the body in the +X direction, the edges that aid this movement are weighted heavier than the rest and create a simple cycle within the graph. To test this each edge of the graph must be analyzed and observed. This is not difficult, as there are only twelve total edges for the robot in this study. However, the number of edges increases as the number of nodes increases, which makes testing each individually difficult and time consuming. The solution would be to find a path that crosses every edge once; this is called an Eulerian path.

Euler paths are any path in a graph which traverses every edge of that graph exactly once. Theorem 1.8.1 from Reinhard Diestel: Graph Theory states "A connected graph is Eulerian if and only if every vertex has even degree." The degree of a vertex refers to how many edges are connected to that vertex. Since each vertex of the graph in our study is connected to every other vertex twice, to and from, the degree of every vertex will always be even. From that we can conclude that the graph has an Eulerian path. There are two common algorithms for determining an Euler circuit within a graph, Fleury's and Hierholzer's. Fleury's algorithm starts at a vertex of odd degree within a given graph if there are at most two vertices of odd degree within the graph. If there are no vertices of odd degree then the path may start at any vertex. From there the path moves across edges and deletes the edges after crossing. The algorithm chooses which edge to cross based on whether or not the deletion of the edge would cause the graph to disconnect. If there is no such edge then it picks the remaining edge and continues this process until there are no edges left. The order in which these edges are chosen forms the Eulerian path. This algorithm however is inefficient time wise, as it detects which edges have been deleted at every node it visits. Hierholzer's algorithm then is our method of choice. This algorithm begins at an arbitrary vertex in a given graph and travels across edges until reaching the starting vertex. From there it will enter another vertex along the trail and follow the unused edges from this vertex until returning to itself. Using this algorithm a path was chosen that is viable for all graphs used in this study.

Starting at the first node the path moves clockwise to every other node until arriving back at the first node. From there it moves counter-clockwise to the last node, and then goes back and forth between that node and every other node besides the first. Once completing all of those edges, the path continues counter-clockwise to the second to last node and repeats the action omitting the edges already traversed. This action is repeated until it arrives at the second node which ends the path by traversing the edge that connects it back to the first node. The Eulerian paths can be represented as the order in which the nodes are passed, so for a four node graph the path is $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 1 \rightarrow 4 \rightarrow 2 \rightarrow 4 \rightarrow 3 \rightarrow 1 \rightarrow 3 \rightarrow 2 \rightarrow 1$

The pattern can be seen clearer when the path is shown in a matrix where the path starts at the first row, first column and the zeros are viewed as only placeholders.

$$\begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 4 & 1 & 4 & 2 & 4 \\ 3 & 1 & 3 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Here is a matrix of an Eulerian path for a three limbed robot illustrated in Figure 3, whose graph would contain eight nodes

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 8 & 1 & 8 & 2 & 8 & 3 & 8 & 4 & 8 & 5 & 8 & 6 & 8 \\ 7 & 1 & 7 & 2 & 7 & 3 & 7 & 4 & 7 & 5 & 7 & 0 & 0 \\ 6 & 1 & 6 & 2 & 6 & 3 & 6 & 4 & 6 & 0 & 0 & 0 & 0 \\ 5 & 1 & 5 & 2 & 5 & 3 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 1 & 4 & 2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Based on this pattern a MATLAB code was developed that accepts the number of nodes as input and returns the Eulerian path as the order in which to pass through the nodes of the graph. This is only one possible Eulerian path in these types of graphs, but it provides a tool that produces a known one quickly and easily.

Edges in graphs are assigned numerical values called weights, which can represent a variety of things. For example, if the nodes represent cities the edges can be weighted by the distance between each city and this provides a better representation of the relationship between each node. This is also useful when the goal is to use the graph to determine a shortest path. For an unweighted graph, the algorithm would treat each edge the same; however, as in the previous example that would be inappropriate.

In the case of observing the locomotion of a soft robot, the edges are weighted by how the edge affects the overall motion of the robot. If we are looking at the robot translating in the +X direction, then the edges are weighted based on the displacement and velocity they effect on the body along the +X axis as observed in the experiments. Taking several sets of data for the same surface environment provides an average weight and an expected threshold. This means that every time the robot is placed in that environment again, it should react in a predictable manner based on edge weights. This test can be repeated for any surface environment, and the data will be used for comparison between surfaces.

Given the graph remains the same, the graphs for different surfaces will only differ by the assigned edge weights. This comparison is the key for the robot determining which surface environment it is in. The robot collects data in real time from the sensors mounted to its body while traversing a preprogrammed path, which will be determined based on the largest edge comparison of the two paths. The robot can then determine which environment it is in based on which threshold the real time data lies in. If the thresholds of the two edges overlap, and the actual data falls within the overlap, the robot could then determine based on which of the average weights it is closest to. Once determining its environment, the robot would know which preset graph to follow.

EXPERIMENTS

The experiments were set up using a two limbed, two gripper robot controlled by an Arduino. A test stand was erected with aluminum slotted bars, and a camera was set up directly over the robot to record its movements. A video of the robot going through the Eulerian path was recorded then uploaded into the free Kinovea imaging software program for analysis of the robot's overall displacement and velocity. Table 1 shows the displacement results from two environments, a smooth surface and carpet.

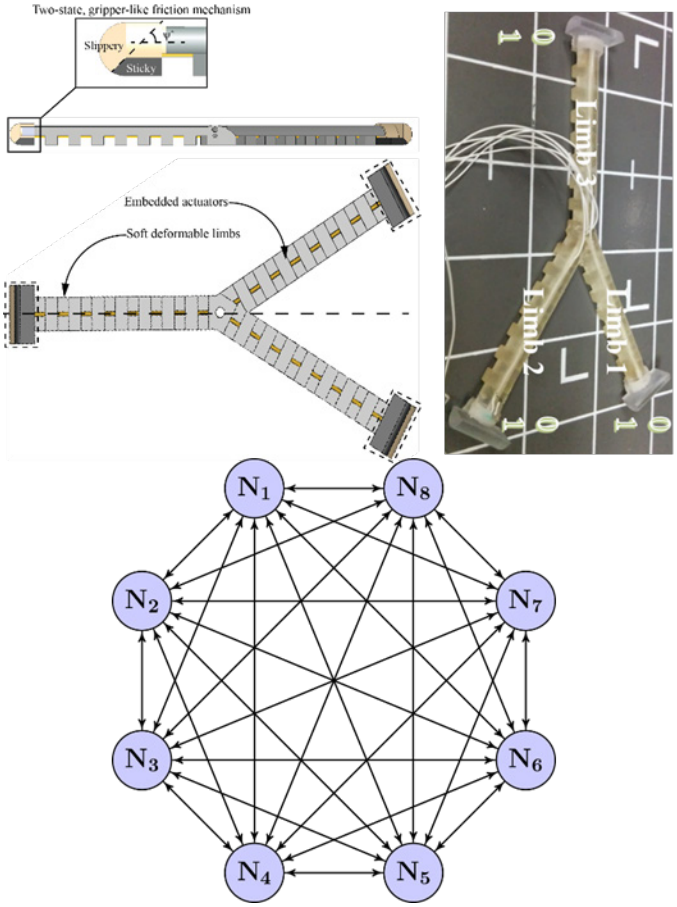

Figure 4. Three limbed soft robot that has correspondingly 8 nodes and the corresponding directed graph.

Path	Smooth		Carpet	
	X (in)	Y (in)	X (in)	Y (in)
1→2	0.04	0.17	0.01	0.15
2→3	0.02	0.11	0	0.1
3→4	0.01	0.2	0.07	0.08
4→1	0	0.09	0.07	0.02
1→4	0.03	0.03	0.07	0
4→2	0	0.42	0.11	0.13
2→4	0	0.1	0.08	0.09
4→3	0	0.03	0.05	0
3→1	0.01	0	0	0
1→3	0.01	0	0.01	0
3→2	0.04	0.55	0.12	0.2
2→1	0	0.14	0.02	0.08

Table 1. Eulerian path results of a two limbed soft robot shown as its displacement along the x and y axis in inches

From the results we can see first that there is a clear difference in displacement along the y-axis between the two environments. The soft robot could go along the simple cycle $3 \rightarrow 2 \rightarrow 4 \rightarrow 2$ as it goes across the two paths of the greatest difference along the y-axis ($3 \rightarrow 2$ and $4 \rightarrow 2$). Knowing this, the robot could collect data from sensors attached to its body and analyze it to determine which environment it is in.

CONCLUSION

In conclusion, this method of discretizing a soft robot's movements to learn about the surface of contact looks promising. Table 1 shows a clear difference in locomotion of the robot across different surface environments; however, only two surfaces were tested for this study. To further show the ability of this theory a larger variety of surfaces will need to be tested. Comparing a variety of surfaces may also show a relationship between paths and the locomotion of soft robots allowing for the control and prediction of the robots' movements. This method is inspired by reinforcement learning, and continues the study of optimizing the robot-environment interaction by taking it further through determination of its environment.

REFERENCES

[1] F. Renda, M. Cianchetti, M. Giorelli, A. Arienti, and C. Laschi, "A 3D steady-state model of a tendon-driven

continuum soft manipulator inspired by the octopus arm," *Bioinspir. Biomim.*, vol. 7, no. 2, p. 025006, Jun. 2012.
[2] I. D. Walker, H. Choset, and G. S. Chirikjian, "Snake-Like and Continuum Robots," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer International Publishing, 2016, pp. 481–498.
[3] D. Trivedi, A. Lotfi, and C. D. Rahn, "Geometrically Exact Models for Soft Robotic Manipulators," *IEEE Trans. Robot.*, vol. 24, no. 4, pp. 773–780, Aug. 2008.
[4] C. Duriez, "Control of elastic soft robots based on real-time finite element method," in *IEEE International Conference on Robotics and Automation*, 2013, pp. 3982–3987.
[5] K. Autumn *et al.*, "Robotics in scansorial environments," 2005, vol. 5804, pp. 291–302.
[6] R. F. Shepherd *et al.*, "Multigait soft robot," *Proc. Natl. Acad. Sci.*, vol. 108, no. 51, pp. 20400–20403, Dec. 2011.
[7] J. Bishop-Moser and S. Kota, "Design and Modeling of Generalized Fiber-Reinforced Pneumatic Soft Actuators," *IEEE Trans. Robot.*, vol. 31, no. 3, pp. 536–545, Jun. 2015.
[8] M. Cianchetti, A. Arienti, M. Follador, B. Mazzolai, P. Dario, and C. Laschi, "Design concept and validation of a robotic arm inspired by the octopus," *Mater. Sci. Eng. C*, vol. 31, no. 6, pp. 1230–1239, Aug. 2011.
[9] V. Vikas, E. Cohen, R. Grassi, C. Sözer, and B. Trimmer, "Design and Locomotion Control of a Soft Robot Using Friction Manipulation and Motor-Tendon Actuation," *IEEE Trans. Robot.*, vol. 32, no. 4, pp. 949–959, Aug. 2016.
[10] T. Umedachi, V. Vikas, and B. A. Trimmer, "Softworms : the design and control of non-pneumatic, 3D-printed, deformable robots," *Bioinspir. Biomim.*, vol. 11, no. 2, p. 025001, 2016.
[11] S. Schaal, "Dynamic Movement Primitives -A Framework for Motor Control in Humans and Humanoid Robotics," in *Adaptive Motion of Animals and Machines*, Springer, Tokyo, 2006, pp. 261–280.
[12] V. Radhakrishnan, "Locomotion: Dealing with friction," *Proc. Natl. Acad. Sci.*, vol. 95, no. 10, pp. 5448–5455, May 1998.
[13] V. Vikas, P. Grover, and B. Trimmer, "Model-free control framework for multi-limb soft robots," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 1111–1116.