Modeling a Transformable Wheel Mobile Robot With a Simulator Neural Network

Anthony J. Clark

Missouri State University, Springfield, MO 65804
anthonyclark@missouristate.edu

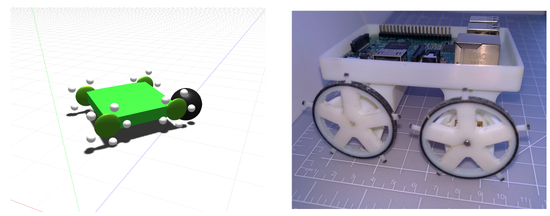
Abstract

It is difficult to model kinematics of a legged-wheel robot. Complex interactions between their irregularly-shaped wheels and the ground make it difficult to derive an accurate mathematical model. Yet, for many applications it is vital to have such a model. For example, to predict the current velocity of the robot. We propose using a neural network to model the kinematics of a transformable wheel mobile robot. We use a physical simulation of our robot to generate training data. The training data is then used to optimize a neural network that can predict changes to the robot's pose based its current control commands. The neural network simulation is better able to predict the location of the physically simulated mobile robot when compared to a differential drive model. Using the trained network, we next evolved a simple controller to navigate a series of way-points. The evolved control parameters were then transferred to the simulated robot where nearly identical behaviors were observed. Our results show that a simulator neural network can be effective in prediction the movement of a transformable wheel mobile robot.

Introduction

Robots are frequently used in remote and unpredictable environments. For example, in *search and rescue* a robot is designed to aid first responders search for victims by traveling over highly varied terrain (Graf et al., 2017). One solution to this problem is to use an unmanned aerial vehicle (UAV). However, UAVs can typically only operate for short periods of time (roughly 30 minutes to one hour).

More recently, lightweight mobile robots with transformable wheels have been developed for search and rescue. As shown in Figure 1, a transformable wheel robot is capable of extending struts radially outward from the center of each wheel. These struts help the robot climb over obstacles that are roughly the same size as the robot (Clark et al., 2018). These devices have the benefits of normal wheels (e.g., increased stability and less vibration) and legged-wheels (e.g., increased mobility), and they have a longer operating duration compared to UAVs.

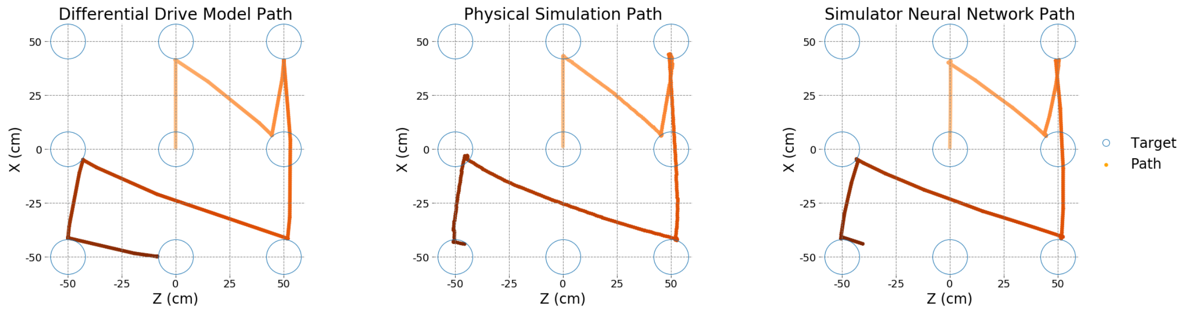


(a) Physical Simulation (b) Prototype

Figure 1: A mobile robot with transformable wheels (see the following an interactive visualization: `https://review.github.io/`).

A drawback of using a transformable wheel mobile robot (or a legged-wheel robot) is that they are difficult to model. Without an accurate model of the robot, it is difficult to determine when wheels should be transformed from normal wheels into a legged-wheels. Specifically, a model can be used to calculate the *expected* velocity of the robot, and this quantity can be compared to the *measured* velocity (using sensor readings). If these two quantities differ by some threshold amount, then the robot should infer that it is stuck or slipping and it should extend its wheel struts. Without an accurate model of the robot kinematics, this process cannot work.

Related work. In prior work, we used a differential drive model to calculate *expected* velocity (Clark et al., 2018). This process was error prone. The model did not account for strut extension, wheel slippage (i.e., differing friction properties of different terrains), uneven ground, or that the wheels might rotate at a different rate than commanded. In this study, we develop a simulator neural network (SNN) similar to that described by Pretorius et al. (2014), where they train a neural network so that control signals map to changes the the pose of the robot.



(a) (b) (c)

Figure 2: Example paths taken by the robot (a) as predicted by the differential drive model, (b) during a physical simulation, and (c) as predicted by a trained SNN. Blue circles indicated way-points that the robot is trying to reach in a predefined sequence. The robot moves on to its next way-point once it gets within 10cm of the current way-point. Paths are indicated in orange, and lighter shades indicate earlier an earlier time (the robot starts at the origin).

Training a Simulator Neural Network

The goal of our study is to develop a SNN that is able to determine a new pose for our mobile robot based on the input control signals. To train the neural network, we first need some way to collect training data.

Collecting training data. We built a physical simulation of our mobile robot using DART (Lee et al., 2018). The simulation can be run with different values for the robot's input control signals: the speeds of the left and right wheels as well as the strut extension amount. The simulation did not include any obstacles or We ran the simulation with 9012 different combinations of these input signals and collected the resulting change in position and heading.

Training the SNN.

noisy measurements uneven terrain

Comparing the SNN model.

running times (phys vs snn) 7.5s .28

difference between SNN and diff will be exaerbated

One example. Others are more or less dramatic

evolve then show behavior in phys vs differential drive model

Evolving a simple controller.

2 minutes vs 1 hour

Use cases: (1) module onboard (2) evolve high-level strategy control parameters

Pretorius et al. (2014) more efficient (30×)

Evolving a simple controller.

future - obstacles - uneven terrain

References

Clark, A. J., Cissell, K. A., and Moore, J. M. (2018). Evolving Controllers for a Transformable Wheel Mobile Robot. *Complexity*, 2018:1–12.

Graf, M., Poy, M., Bischof, S., Dornberger, R., and Hanne, T. (2017). Rescue path optimization using ant colony systems. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7.

Lee, J., X. Grey, M., Ha, S., Kunz, T., Jain, S., Ye, Y., S. Srinivasa, S., Stilman, M., and Karen Liu, C. (2018). DART: Dynamic Animation and Robotics Toolkit. *The Journal of Open Source Software*, 3(22):500.

Pretorius, C. J., du Plessis, M. C., and Gonsalves, J. W. (2014). A comparison of neural networks and physics models as motion simulators for simple robotic evolution. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 2793–2800, Beijing, China. IEEE.