Learning the combined Dynamics of a Pushing agent and a block using an Autoregressive Model

Alrick Dsouza
ME department, Personal Robotics Lab
University of Washington
adsouz@uw.edu

Matt Schmittle
Personal Robotics Lab
University of Washington
schmittle@cs.washington.edu

Christoforos Mavrogiannis
Personal Robotics Lab
University of Washington
cmavro@cs.washington.edu

Abstract

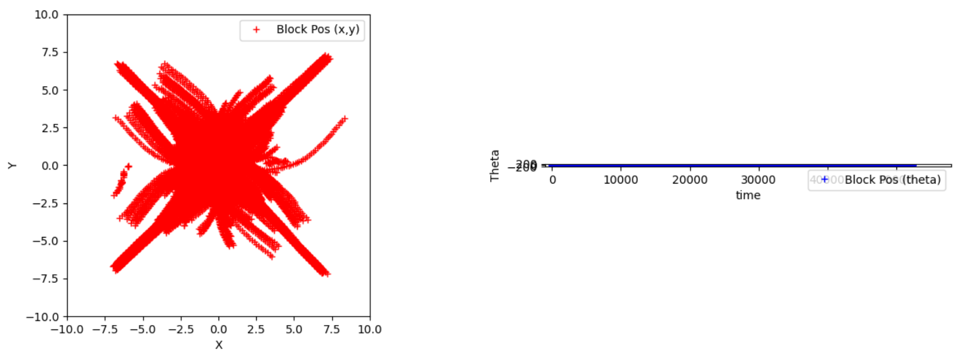
Push based manipulation has been studied for a long time due to its ubiquitous nature and ability to simplify complex robotic tasks [6]. However, Pushing in cluttered environments is relatively unexplored and challenging at the same time due to the pushed objects uncertainty in motion. The aim of my thesis is to develop an algorithm to manipulate objects of cubic geometry using a nonholonomically constrained autonomous vehicle, and the aim of this project is to learn a robust model of the dynamics of the combined pusher-pushed agent system which can be used for path-planning, multi-step prediction and obstacle avoidance.
An Autogressive model is used to learn the dynamics of the model and at the same time add stochasticity, which is useful for testing in the real-world. The results indicate that the LSTM has superior performance over the GP and regression model. It is observed that the test performance of the LSTM increases with decreasing sequence length. This observation is not expected for multi-step prediction and higher noise levels where larger sequence lengths might prove to be more meaningful.

1 Introduction and Motivation

Pushing is an essential primitive motion that can extend a robot's manipulation capabilities dramatically. Since pushing is a non-prehensile action (doesn't require a gripper), it can be performed by non-prehensile robots. A lot of work has been carried out in pushing using non-prehensile robots ([1], [2], [3], [4]) using analytical ([5], [6]), data-driven [7] and deep learning ([8], [9]) approaches. However, we plan to execute multiple simultaneous pushing tasks in crowded environments which would introduce us to a unique set of constraints from multi-agent coordination like executing the pushing task within a small-confined region and completing the pushing task within a fixed time frame.

The goal of my thesis is to develop an algorithm which enables an autonomous, nonholonomically constrained robot vehicle with a flat-surfaced bumper attached to its front side to manipulate objects of cubic geometry to a target configuration in a confined space. This work is a part of the parent project called Pixel Art, where a cluster of mobile robot cars under careful planning will push blocks (objects of cubic geometry) to target configurations to make 2D art.

Learning a robust model of the dynamics of the combined car and block system is necessary for path planning, multi-step prediction and avoiding dynamic and static obstacles. In addition, the learned dynamics can be also used for path planning in the real world with minimal training.



(a) X-Y distribution (b) Theta distribution

Figure 1: Data distribution

These approaches will be tested on the MuSHR racecar [10], developed at the Personal Robotics Lab at UW.

2 Approach

Inspired by Bauza et al. [2], the lab first used a Gaussian process (GP) to learn the dynamics of the combined car-block system. There were several issues here, since the GP is a Non-parametric learner it has an upper limit on the size of the dataset it can train on. Also data collection became tricky as you want to incorporate various different trajectories into limited data.

Given course requirements, and the ability of Generative Models to deal with stochasticity, Autoregressive Models seemed like a good fit to learn the forward dynamics of the car-block system.

The basic model architecture is to encode our position data X = (x, y, theta) to a higher dimensional vector, pass this data to the LSTM network and learn separate decoder layers to output x,y and theta. To add stochasticity, we first find the value of x and input it to decoder y along with the LSTM output, and similarly input y to the decoder of theta. Prior to this, the data is split into batches and further the batches into sequences of fixed length, this way the LSTM can update its hidden and cell state from prior data.

3 Experiment setup and data collection

The experiment setup consists of the MuSHR racecar and a cubic block of length equal to .1 units in MuJoCo. For data collection, particularly three Control policies of interest are used, namely Straight path, Constant curve and Random controllers. The plots in Figs 1(a), 1(b) display the block distribution in MuJoCo space and few example trajectories.

The data is divided into train, test and unseen test categories. Unseen tests are trajectory positions beyond a circle of radius 4 units (meters in MuJoCo). However, since we train the models using a relative position of the block w.r.t. to the car, the relative value of the block pose w.r.t. to the car may not be unseen by the model. Also it is observed, mostly Straight Path policies cross the circular boundary and hence Unseen test performances are higher than Seen test performances, as from different experiments we learned that Straight Path policies are easier to learn.

4 Results

Since the accuracy of the model predictions is inversely proportional to the difference in actual block position and predicted block position, the Average Mean Squared Error (MSE) is a good evaluation

Table 1: Model Comparison

Model	Test seen MSE	Test unseen MSE
Simple Regression	0.4757	0.1945
Gaussian Process	7.5612-0.3845	2.2062-**0.0911**
Autoregressive Model	**0.3452**	0.2307

Table 2: LSTM Sequence length analysis

LSTM Sequence length	Test seen MSE	Test unseen MSE
1	**0.3452**	0.2307
10	0.3553	**0.1105**
100	0.3562	0.1397
500	0.3945	0.1567
1000	0.3844	0.2136

metric. Table 1 consists of a comparison between a Simple Regression (SR) model, Autoregressive (AR) model and previously used Gaussian Process (GP).

The expected dynamics of the combined car-block system is supposed to be independent of earlier time-steps, and we see that the performance of the Simple regression model is not bad. The regression model consists of only full-connected layers and ReLU non-linearity.

The Gaussian Process being non-parametric can provide satisfactory performance if a good sample of the dataset is selected for training, as given by the upper and lower range of MSE observed while training the GP.

Using an LSTM as an Autoregressive Model, provides both low MSE and consistent performance across all trajectories in the test set. A further analysis on the effect of sequence length is provided in Table 2, where in we observe an increase in test performance with a decrease in sequence length. The LSTM was trained with varying sequence length, and all other parameters namely, batch size, learning rate and epochs constant.

Note: Training and testing plots for the LSTM are attached in the Appendix.

5 Conclusion and Discussion

The performance of LSTM is superior to the simple regression model as well as the Gaussian Process given by the MSE Evaluation metric, but the visual trajectories of the Simple regression model look more accurate. Also, table 2 shows an increase in test performance of the LSTM with decreasing sequence length, suggesting that a more complex regression model might be able to overcome the performance gap.

One reason for Simple regression performing very well could be that the noise in the Data and inputs to the model is quite low. The reason for not adding additional noise to the data initially is because the data collection is not perfectly denoised either. For example, it was observed that there were slight differences in the displacement of the block w.r.t. to the car in the direction of the normal of the flat surfaced bumper while executing Straight Path policies. However, the difference was in the order of 10^{-5}, which the regression model seems to have been able to overcome.

In the next phase of experiments, additional noise will be added to the data, magnitude of which will be identical to the tracking system in the Personal Robotics Lab. Also noise will be added to speed and steering angle values as well.

Furthermore, the next phase of experiments will include multi-step prediction, i.e. (predicting the trajectory of length N given starting state and N control inputs) where larger sequence lengths is expected to provide more meaning and perform better than regression models.

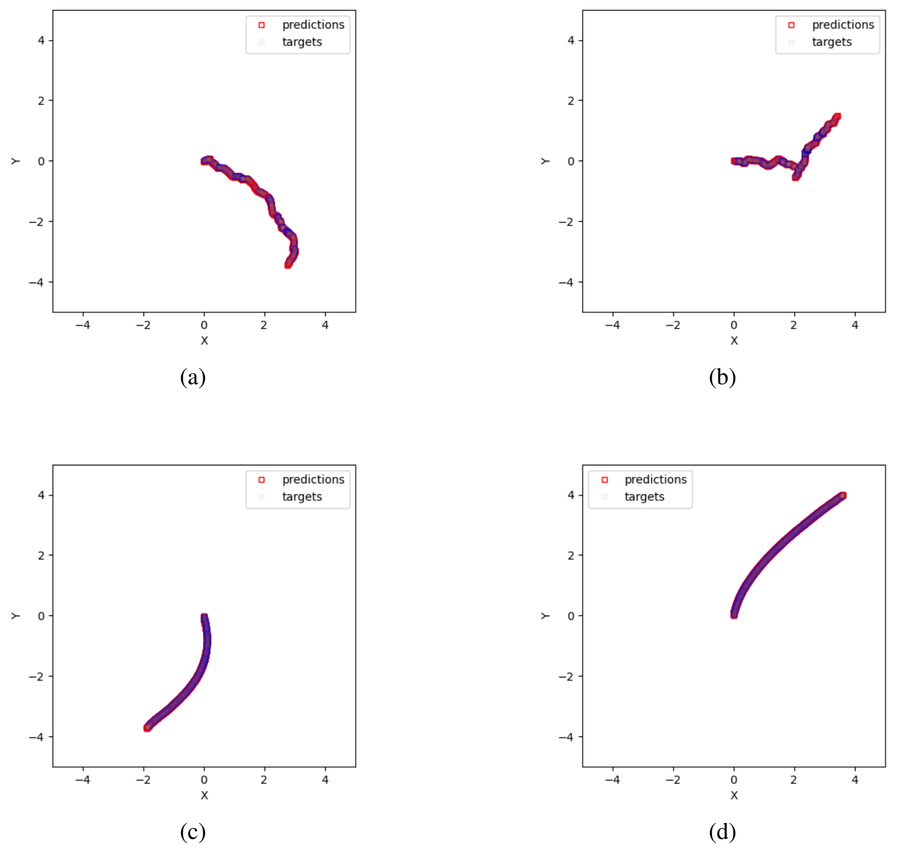


Figure 2: Trajectory visuals

References

[1] Agarwal, P. K., Latombe, J. C., Motwani, R., and Raghavan, P. (1997). *Nonholonomic path planning for pushing a disk among obstacles,* in Proceedings of International Conference on Robotics and Automation, Vol. 4 (Albuquerque, NM), 3124–3129.

[2] Bauza, M., Alet, F., Lin, Y.-C., Lozano-Perez, T., Kaelbling, L. P., Isola, P., et al. (2019). *Omnipush: accurate, diverse, real-world dataset of pushing dynamics with rgb-d video,* in International Conference on Intelligent Robots and Systems (IROS) (Macau).

[3] de Berg, M., and Gerrits, D. H. P. (2010). *Computing push plans for disk-shaped robots,* in 2010 IEEE International Conference on Robotics and Automation (Anchorage, AK), 4487–4492.

[4] Emery, R., and Balch, T. (2001). *Behavior-based control of a non-holonomic robot in pushing tasks,* in Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164), Vol. 3 (Seoul), 2381–2388.

[5] M. T. Mason, *Manipulator Grasping and Pushing Operations,* Jun. 1982, Accessed: Sep. 21, 2020. [Online]. Available: https://dspace.mit.edu/handle/1721.1/6853.

[6] M. R. Dogar and S. S. Srinivasa, *Push-grasping with dexterous hands: Mechanics and a method,* in 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct. 2010, pp. 2123–2130, doi: 10.1109/IROS.2010.5652970.

[7] S. Krivic, E. Ugur, and J. Piater, *A robust pushing skill for object delivery between obstacles,* in 2016 IEEE International Conference on Automation Science and Engineering (CASE), Aug. 2016, pp. 1184–1189, doi: 10.1109/COASE.2016.7743539.

[8] J. Li, W. Sun Lee, and D. Hsu, *Push-Net: Deep Planar Pushing for Objects with Unknown Physical Properties,* presented at the Robotics: Science and Systems 2018, Jun. 2018, doi: 10.15607/RSS.2018.XIV.024.

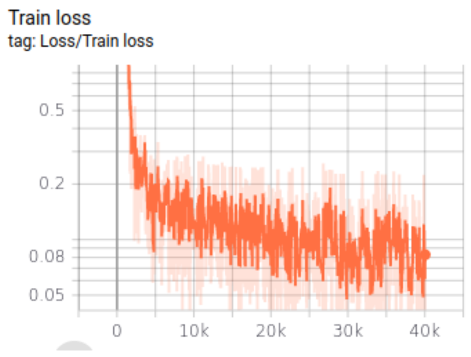
[9] C. Finn, I. Goodfellow, and S. Levine, *Unsupervised Learning for Physical Interaction through Video Prediction,* in Advances in Neural Information Processing Systems 29, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 64–72.

[10] S. S. Srinivasa, Siddhartha S. and Lancaster, Patrick and Michalove, Johan and Schmittle, Matt and Summers, Colin and Rockett, Matthew and Smith, Joshua R. and Chouhury, Sanjiban and Mavrogiannis, Christoforos and Sadeghi, Fereshteh, *MuSHR: A Low-Cost, Open-Source Robotic Racecar for Education and Research,* CoRR, vol. abs/1908.08031, 2019.

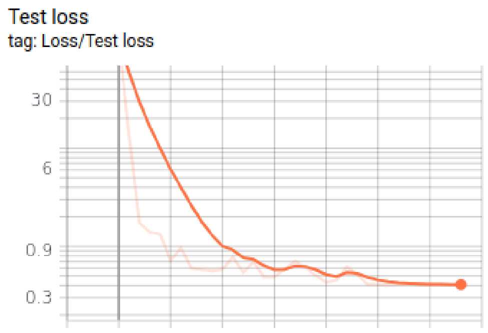
6 Appendix

The Model architecture couldn't fit in the report. To visualize the model architecture, go through README.pdf.

The tensorboard file can be found in "mushr_push_sim/train/TensorboardVisuals/". These are the training and testing plots of LSTM with sequence length 1000. (I deleted the sequence length 1 plots by mistake, and did not have much time to run it again.)



(a) Training



(b) Test seen

Figure 3: Training and testing plots