Position Estimation for Control of a Partially Observable Linear Actuator

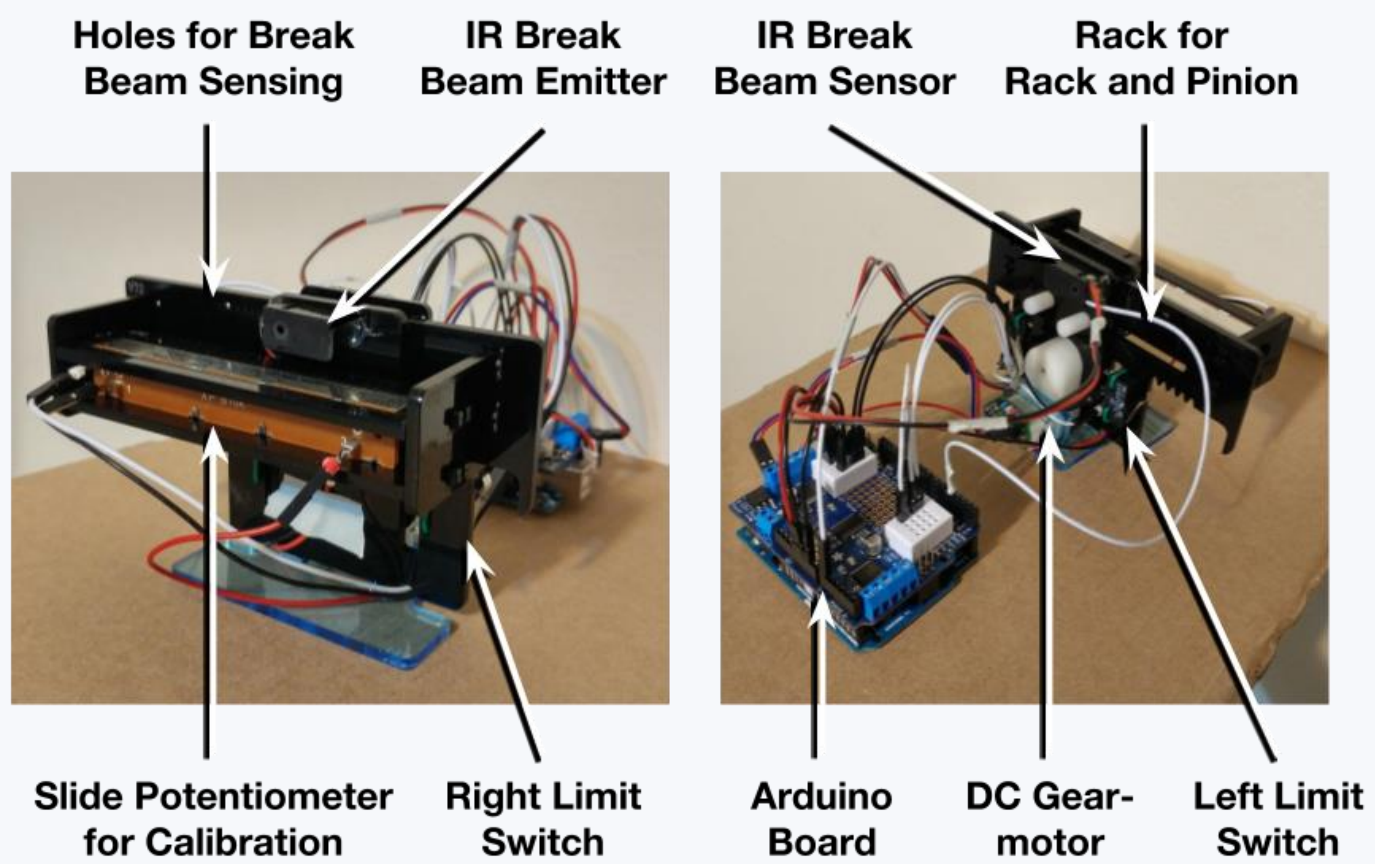
Ethan Li — ethanli@stanford.edu
Arjun Arora — aarora52@stanford.edu
Jay Whang (CS 238) — jaywhang@stanford.edu

Goals & Motivation

- Implement position control of a linear actuator with only partially observable state, specifically:
- Evaluate regression methods for estimating position from noisy and sparse sensor data:
  1. Linear Regression
  2. SVMs
  3. Boosting
- Evaluate models in real-time control of a hardware prototype

Data Collection and Methodology

- We collected a train/dev set by recording timeseries data at 100 Hz from 23k episodes of control with a baseline position estimator and controller from a hardware prototype:



- Each episode lasted for ~1 second
- We excluded episodes in which the edge sensor was found to have miscounted edges and split the remaining 3M examples by 80/20 train/dev split on episodes
- Ground truth position was measured by slide potentiometer
- Position estimators were compared by prediction RMSE
- Final evaluation was performed by using position estimator as feedback input to proportional-derivative controller, scored as squared error accumulated over episode duration

References
Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

Feature Sets

- Collected sensor inputs consisting of accumulated edge counts from optical edge sensor, limit switch states, motor direction and duty, and event timers from sensors
- Standard: all collected sensor inputs
- Minimal: exclude event timer features
- Engineered: add inverses of event timers, motor velocity
- Interaction: add all two-feature interaction terms between Engineered features

Experiments

- Linear Regression with Standard features
- Ridge Regression with Interaction features
  Alpha = 10
- Support Vector Regression with Minimal features
  Linear Kernel
- Gradient Boosting Regression with Engineered features
  400 regression trees of depth 6
- Precise Gaussian Noise Oracle: 99.7% of predictions within 0.5 mm of ground truth
- Imprecise Gaussian Noise Oracle: 99.7% of predictions within 1 mm of ground truth
- Naïve Baseline: simple hand-crafted formula

Results

Experiments	*Train Mean Error (mm)*	*Train RMSE (mm)*	*Dev Mean Error (mm)*	*Dev RMSE (mm)*
Precise Gaussian Oracle	N/A	N/A	0	0.167
Imprecise Gaussian Oracle	N/A	N/A	0	0.334
Naïve Baseline	N/A	N/A	-0.216	0.751
Linear Regression	0	0.615	0.008	0.601
Ridge Regression	0	0.427	0.015	0.398
Support Vector Regression	-0.105	0.642	-0.068	0.603
Gradient Boosting Regression	0	0.306	0.029	**0.316**

Table 1. Train/dev errors from position estimation experiments. Experiment with lowest dev RMSE highlighted in bold.

Controller Evaluation

Position Estimator	*Controller*	*Mean Score (mm^2)*	*Final Position Error RMSE (mm)*
Ground Truth	High Gain PD	-19571	1.85
Ground Truth	LQG	-20044	2.09
Ground Truth	Low Gain PD	-21882	1.17
Linear Regression	**Low Gain PD**	**-21951**	**1.45**
Linear Regression	LQG	-22909	6.46
Baseline	Hysteresis	-44974	10.9

Table 2. Mean scores of control episodes and RMSEs of final positions for various combinations of position estimators and controllers.

Analysis

- Gradient Boosting (best model) matched Imprecise Gaussian Oracle
- Every model predicted within 1.2 mm of ground truth position >95% of the time
- With feature engineering to add nonlinear features, linear models capture enough nonlinear dynamics to do well
- With current prototype, full speed actuation caused optical sensor errors; low gain PD tuned to avoid errors
- Difference between high gain PD and low gain PD shows that hardware design limits controller performance
- Similar scores of low gain PD show position estimator doesn't limit controller performance on current hardware

Conclusion & Future Work

- Position estimation is viable approach to address partial observability
- Hardware design must be improved for improved sensor reliability at high actuation speeds to take advantage of more precise position estimators and improve control
- Deep learning and state estimation approaches (Kalman filtering) may achieve improved position estimation