Simulated Data Generation Through Algorithmic Force Coefficient Estimation for AI-Based Robotic Projectile Launch Modeling

Sajiv Shah[1*], Ayaan Haque[1*], Fei Liu[2] (*equal contributions)

[1]Saratoga High School, [2]University of California, San Diego

Overview

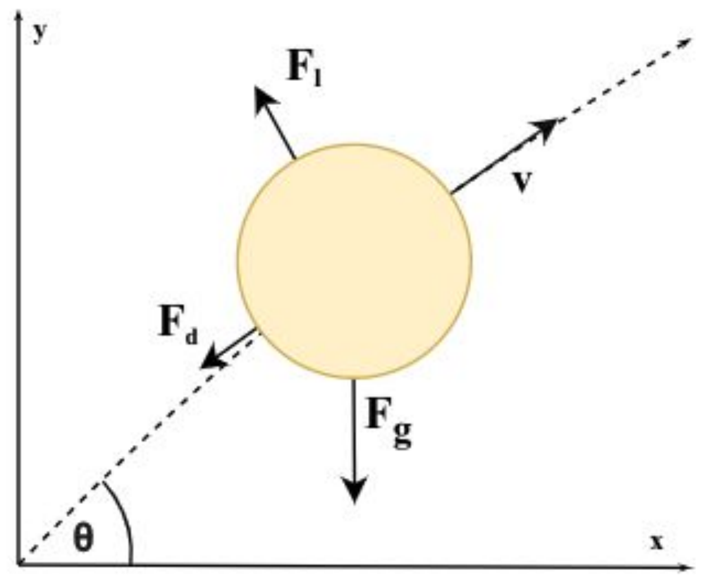
Problem

- Launch modeling with physics-based models does not consider all dynamics, creating inaccuracies.
- Unknown aerodynamic force coefficients can only be estimated using expensive equipment.
- Using data-driven AI solutions requires large data pools.

Solution: FCE-NN

- Combine a small set of collected data with a large pool of generated data, which is derived from a physics-based simulator, and then use neural networks to model projectile flight trajectory.
- ArXiv: https://arxiv.org/abs/2105.12833; Code: https://github.com/ayaanzhaque/FCE-NN

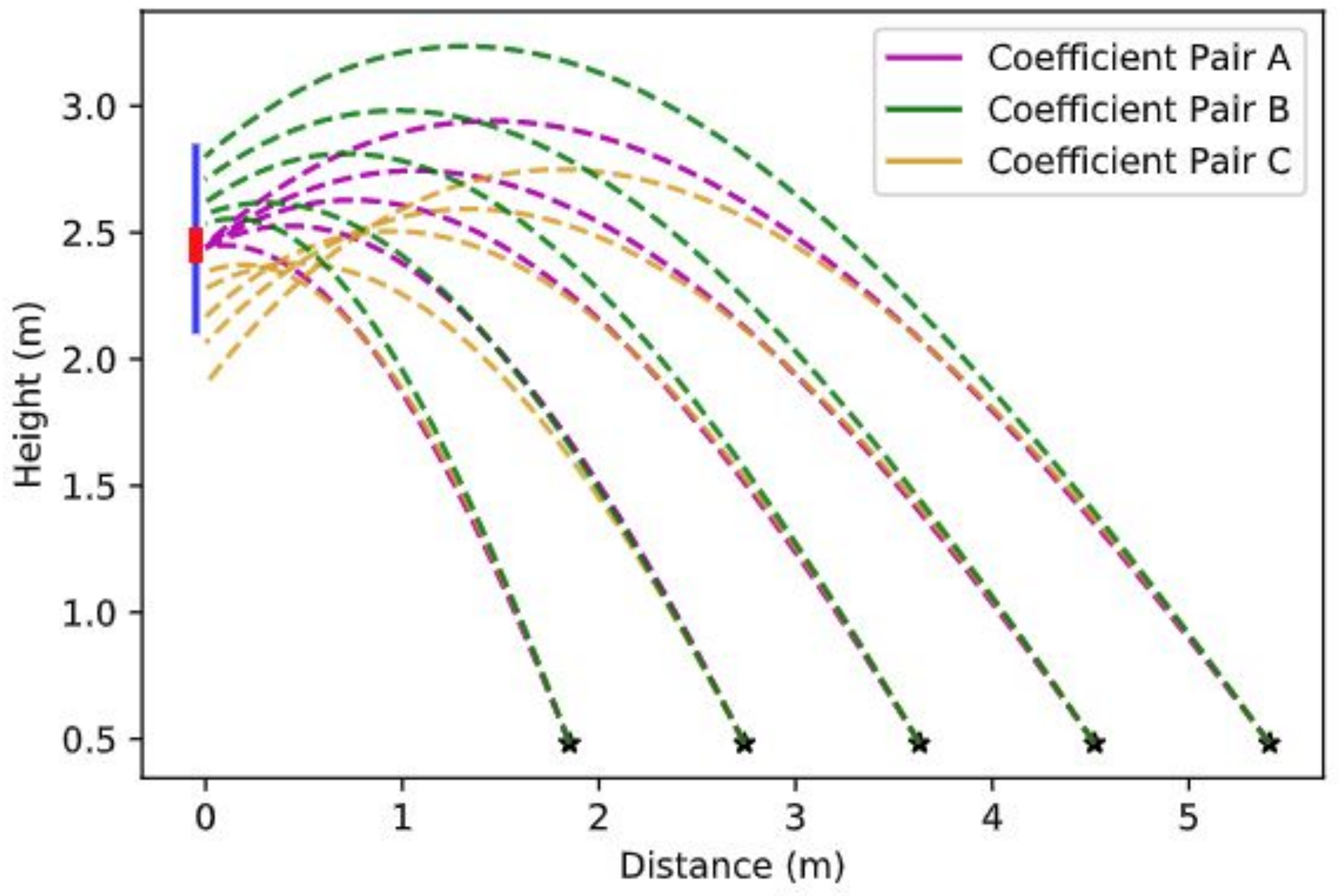
Simulator Design



$$F_g = mg$$
$$F_l = C_l \frac{4}{3}(4\pi^2 r^3 s\rho v)$$
$$F_d = \frac{C_d}{2} A v^2 \rho$$
$$\frac{dv_x}{dt} = -(a_l \sin(\theta) + a_d \cos(\theta))$$
$$\frac{dv_y}{dt} = a_l \cos(\theta) - a_d \sin(\theta) - a_g$$
$$\theta = \int_0^t \arctan(\frac{v_y}{v_x})dt + \theta_0$$
$$v_x = \int_0^t (\frac{dv_x}{dt})dt + v_0 \cos(\theta_0)$$
$$v_y = \int_0^t (\frac{dv_y}{dt})dt + v_0 \sin(\theta_0)$$
$$x = \int_0^t (v_x)dt$$
$$y = \int_0^t (v_y)dt$$

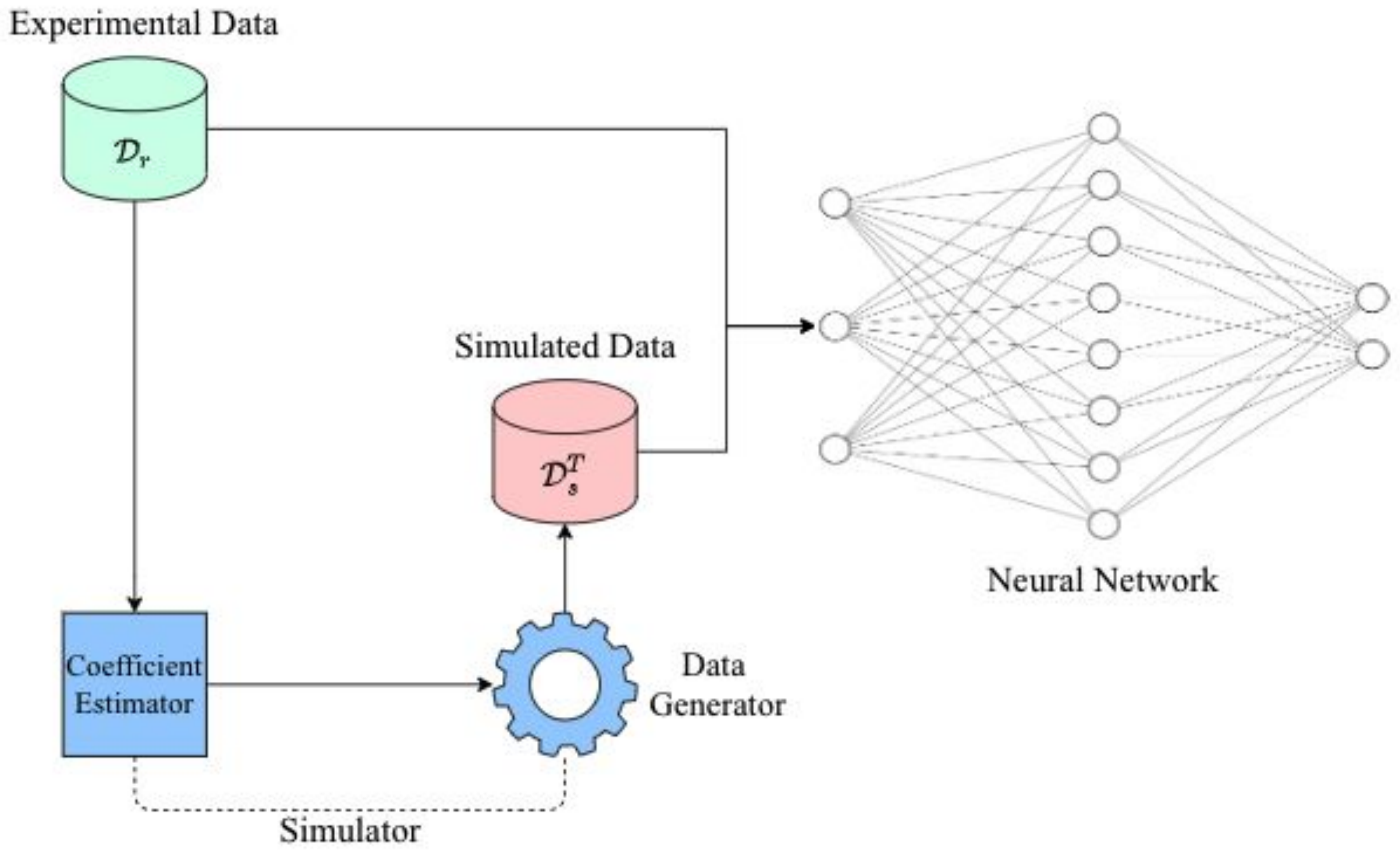
- The ball is given an initial launch speed, angle, and distance.
- Based on these parameters we calculate the position of the ball using known forces and can determine its outcome.

Force Coefficient Estimation



- We iterate through 1000^2 combinations to estimate the coefficients which best model the experimental outcomes.

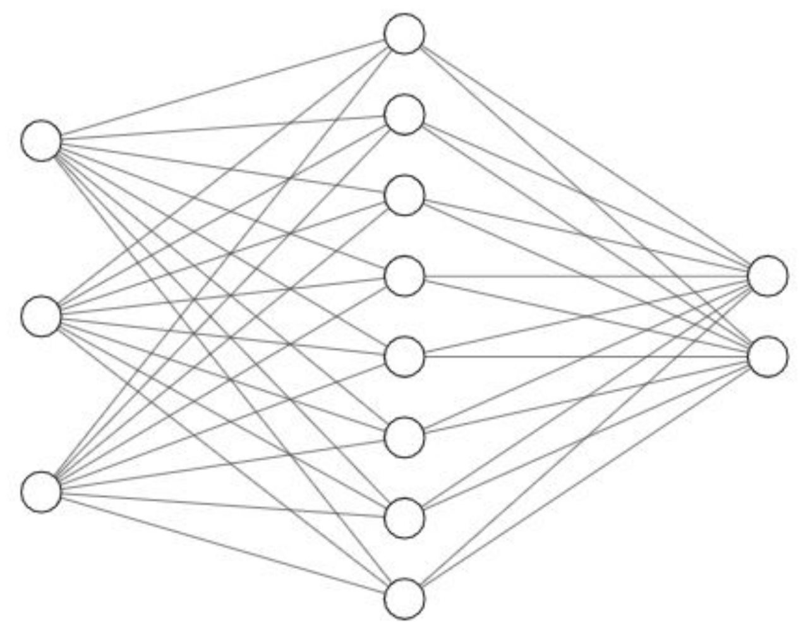
FCE-NN



- Experimental data is collected using the launcher and visibly assigning outcomes
- This data is fed into the coefficient estimator, which tunes the simulator to match the collected data.
- The simulator is then used to generate the simulated data set.
- Both the experimental and simulated datasets are fed into a neural network to generate launch predictions.

Neural Network

- Simple 3-layer forward-feed neural network for classification, 3 input nodes per launcher configuration, 2 output nodes for 2- and 3- point scores.



Robot

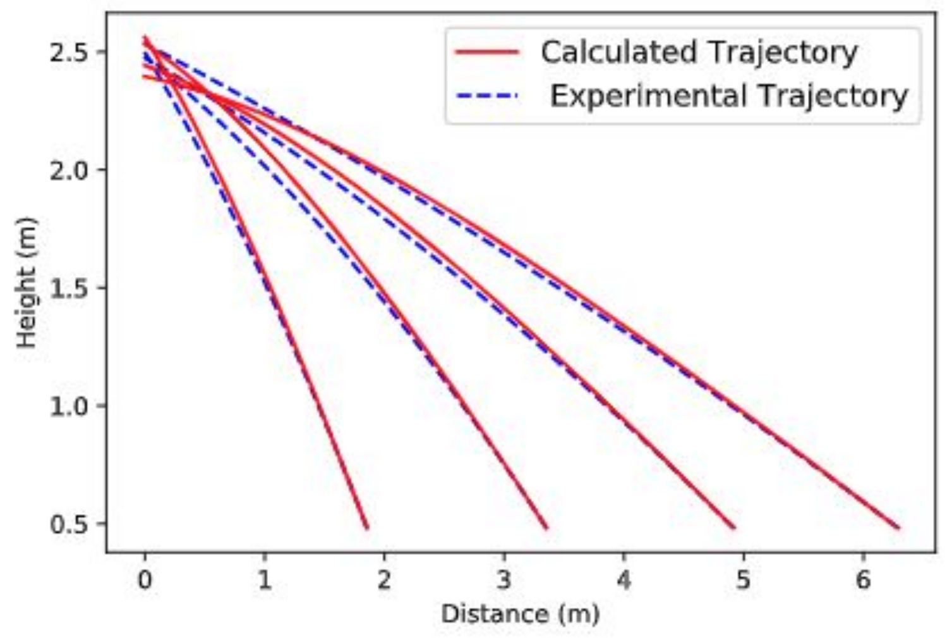
- Our launcher sits on top of a mobile robot.
- We vary launch parameters (angle, speed, and distance) electronically and use sensor data to confirm accuracy.

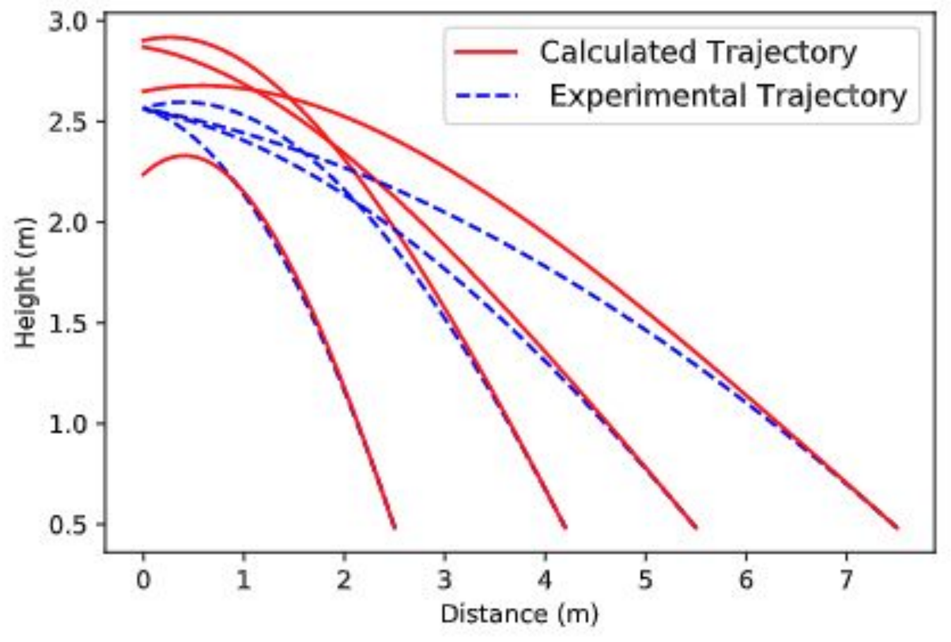
Results

Using Force Coefficient Estimation

- We generate outcomes for both a training and testing dataset using the simulator without the NN
- The accuracy decreased significantly on the testing set, as data outside of the training set is poorly modeled by FCE
- Figure shows accurate trajectories (right) and inaccurate trajectories (left)

$\mathcal{D}_r$ Train/Test	Metrics			
	2pt Acc (%)	3pt Acc (%)	Mean Dev	Median Dev
Training	100	92.31	0.051	0.058
Testing	92.86	**85.71**	0.283	0.124





Using FCE-NN

- We compare outcomes with various sizes of simulated data, and discover that with the optimal amount our overall accuracy increases significantly

$\mathcal{D}_s$ Size	Metrics (Combined evaluation of $\mathcal{D}_r$ and $\mathcal{D}_s^T$)		
	Overall Acc (%)	F1-3pt (%)	F1-2pt (%)
0	44.29	4.17	66.66
500	78.57	50.00	87.50
600	87.14	70.83	91.67
750	91.43	79.16	95.83
900	**98.56**	**95.83**	**100.00**
1000	95.71	91.67	95.83
1250	92.85	87.50	91.67

Conclusion

- We show that the neural network with additional data is able to accurately model and predict the outcome of both experimental and simulated configurations.
- Our future work will investigate advanced NN architectures as well as reverse-engineering NNs to derive force coefficients.