Robotic Throwing

Jason Zhao
Perception Robotics Group, University of Maryland
jzhao110@terpmail.umd.edu

Abstract

Robotic throwing traditionally depends on precise calibration of robot kinematics and known metric distances. However, calibration degrades over time and limits robustness in real-world environments. Inspired by human throwing, which does not rely on explicit distance measurements, we investigate whether a robot can learn to throw accurately using only visual feedback. We present a simulation-based approach combining system identification through self-exploration with trajectory optimization. Results demonstrate that accurate throwing can be achieved without calibrated distances by learning an internal model of the robot's dynamics.

1 Introduction

Humans are capable of throwing objects accurately without explicit knowledge of distances or precise calibration of their musculoskeletal systems. In contrast, robotic systems typically rely on carefully calibrated models and metric measurements to execute dynamic tasks such as throwing. Over time, these calibrations degrade due to wear, environmental changes, and mechanical drift, reducing performance and increasing maintenance costs.

This work explores whether robotic throwing can be achieved without calibrated distances, using only visual input. Our goal is to develop a system that can accurately throw a ball to a target using a camera, without reliance on a ruler or explicit metric calibration. Such an approach could reduce calibration costs and improve robustness in real-world deployments.

2 System Overview

We study the throwing task in simulation using a UR10 robot arm, a six-degree-of-freedom collaborative manipulator designed to operate safely alongside humans. The task consists of throwing a ball toward a target under fixed experimental conditions: a ball radius of 3.5 cm, a target radius of 5 cm, and a nominal target distance of 1.5 m.

The system consists of two main components: system identification to learn an accurate internal model of the robot dynamics, and trajectory optimization to generate throwing motions that minimize error to the target.

Figure 1: Real UR10 robot used

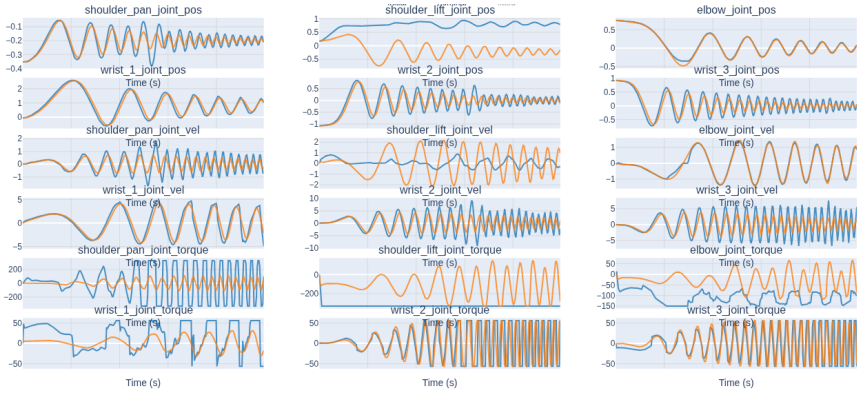
3 System Identification

Throwing without knowing the robot's precise dynamics presents a significant challenge. To address this, we employ a practical system identification procedure based on self-generated motion. The robot executes exploratory "wiggling" motions while recording joint trajectories.

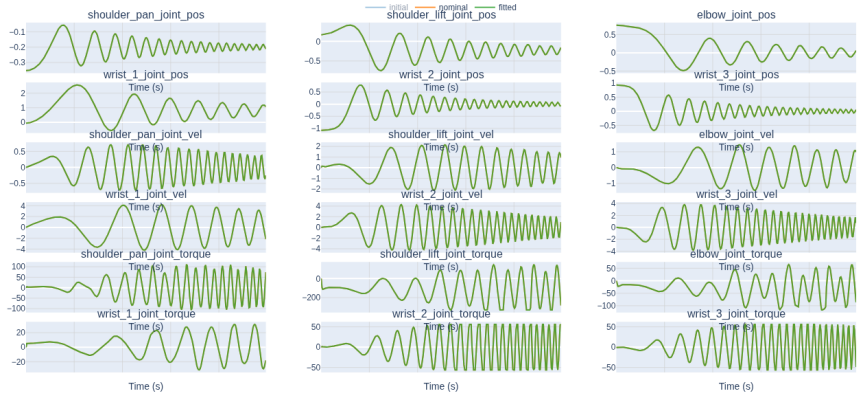
During system identification, the robot:

1. Predicts motion using an initial dynamics model,
2. Observes the actual resulting motion,
3. Updates model parameters to minimize prediction error.

This process is repeated until predicted and observed motions closely align. Figure 2 shows that system identification transforms a poorly fitting initial model into a near-perfect match with observed joint trajectories and torques.



(a) Before system identification



(b) After system identification

Figure 2: Model estimates before and after system identification

4 Learning to Throw with Optimization

Throwing is formulated as a trajectory optimization problem, where a trajectory is a time-parameterized sequence of joint configurations culminating in ball release. Starting from an initial guess, the optimizer iteratively refines the trajectory to reduce throwing error.

Early trajectories often miss the target by a significant margin, while later iterations converge to an optimized trajectory that successfully delivers the ball. Figure 3 illustrates the progression from an initial guess to an optimized throwing motion.

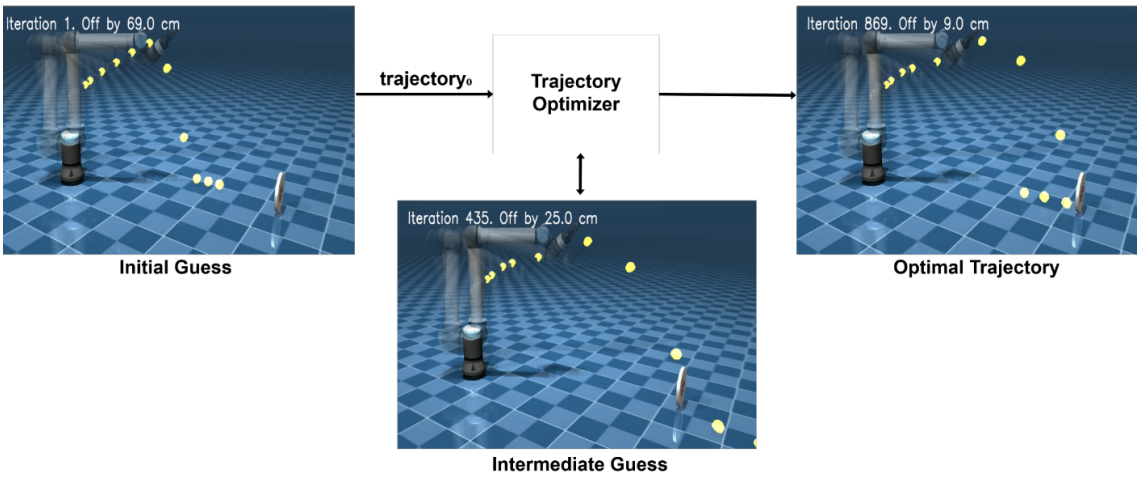


Figure 3: Progression of throwing trajectories during optimization, from initial guess to optimized solution.

5 Results

Simulation and real testing showed that the robot could accurately throw a ball. The next step would have been to move all units into embodied units however, hardware limitations appeared. Due to the limited speed of the UR10, throws beyond 1.5 m were not possible, making an interesting demo unfeasible. As a result, work on throwing was abandoned.

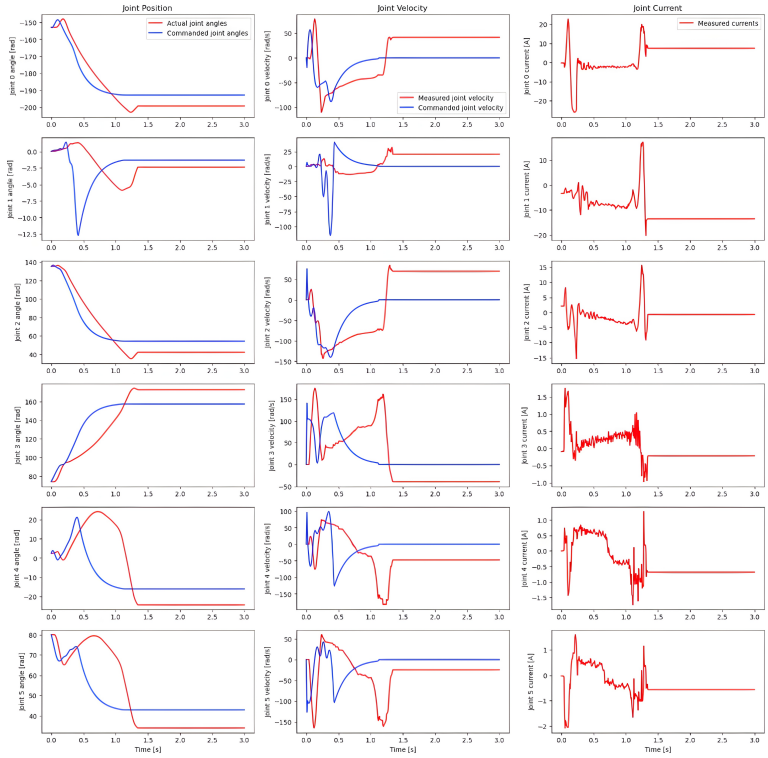


Figure 4: Joint Position, Velocity, and Current after speed violation. Actual joint position (red left column) and commanded joint position (blue right column) before divorced

Table 1: Joint Speed Limits

Joint / Component	Maximum Speed
Base	120°/s
Shoulder	120°/s
Elbow	180°/s
Wrist 1	180°/s
Wrist 2	180°/s
Wrist 3	180°/s
Tool (Typical)	1 m/s (39.4 in/s)

6 Discussion and Future Work

This work demonstrates that robotic throwing with embodied units is an interesting idea, however, due to hardware limitations, other mediums to demonstrate embodiedment are more practical. Future work is currently embodied putting.

References

1. L. Burner, C. Fermüller, and Y. Aloimonos, "Embodied Visuomotor Representation," *arXiv preprint arXiv:2410.00287*, 2024.

2. K. Zakka, L. Burner, et al., "Practical System Identification," in preparation.