Report Title: Development of a Hydraulic Soft Robotic Arm Control System on STM32H723

Project Overview

This project presents a sliding pseudo-joint for soft pneumatic robot arms, inspired by the octopus's ability to dynamically relocate bending points. A lightweight slider (0.095 kg) is mounted on an inflatable nylon-TPU tube and repositioned by creating a pressure differential between two internal chambers. When one chamber is pressurized, the slider propels toward the lower-pressure end (reaching mode); when both are pressurized equally, the slider locks in place and acts as a rigid joint (grasping mode). Experiments confirmed a linear relationship between flow rate and slider speed, enabling straightforward position estimation through flow integration; at 14.4 L/min the slider covered 260 mm in 3.25 seconds (Luo et al., 2025).

The project's position control relies entirely on open-loop flow integration with no direct feedback sensor on the slider, and object localization is left as future work. These open problems are directly relevant to the hydraulic arm developed in this project, which addresses both challenges: a bidirectional flow sensor provides closed-loop position estimation, and the PI controller developed here actively corrects for tracking error during fill and drain operations.

Given Tasks

Design and implement a closed-loop position control system on the STM32H723 microcontroller for a hydraulic soft robotic arm. The system estimates slider position in real time by integrating flow rate measurements from a bidirectional flow sensor. It uses a PI controller to regulate pump operation, stopping early by a calibrated amount so that hydraulic pressure carries the slider to the desired set point without overshoot.

Contribution

Closed-loop System Simulation

The control system was first designed and validated in MATLAB Simulink before implementation on the STM32H723. A PI controller computes an error signal between the desired slider position and the estimated position, splitting it into proportional (K_p) and integral (K_i) paths before summing and saturating the output as a pump command. The plant is modelled using two transfer functions representing the pump dynamics and the flow rate sensor response to simulate a realistic, delayed response.

A key feature of the model is the separation between true position and estimated position: the true position is derived from the actual simulated flow output through the full pump dynamics, while the estimated position mirrors what the embedded firmware computes via flow integration. Both are logged and compared, allowing the accuracy of the position estimate to be validated in simulation before deploying to hardware.

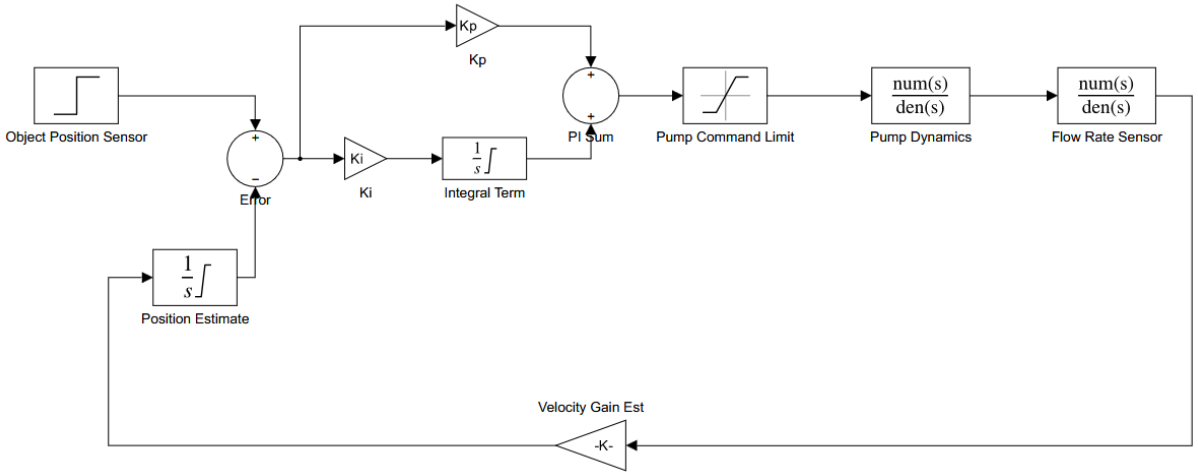

Figure 1: MATLAB Simulink model of the closed-loop PI position control system.

Closed-loop System Embedded Program

The closed-loop control system was implemented across two files: `position_control.h` and `position_control.c`. The header file defines all system parameters — including the effective arm area (`PC_A_EFF` = 1.5625×10^{-3} m^2), the maximum pump flow rate (`PC_Q_MAX`), the PI gains (`PC_KP`, `PC_KI`), and the two compensation margins (`PC_OVERSHOOT_COMP` and `PC_DRAIN_COMP`) that stop the pump early to account for hydraulic inertia. The core logic lives in `PositionControl_Update()` inside `position_control.c`, which runs every control cycle: it reads the flow sensor, integrates the measured flow rate divided by the arm area to update `pos_estimate`, computes the error between the effective target and the current estimate, and drives the pump via a saturated PI output. For the fill direction, the controller applies both proportional and integral terms and scales the result to a PWM duty cycle; for drain, it runs the pump at full speed and cuts it off when `pos_estimate` reaches the early-stop threshold.

A companion function, `PositionControl_TrackFlowOnly()`, was also written for states where the pump is manually driven at a fixed duty rather than PI-controlled, accumulating flow into `pos_estimate` without interfering with the pump command. Both functions are called from the arm state machine in `main.c`, where `arm_state 8` uses the full PI controller and `arm_state 2` uses the fixed-duty tracking approach.

Results

Following iterative calibration of the PI controller and compensation parameters, the system achieved a position tracking error of 5% or less across tested fill targets. Two control modes were evaluated: `arm_state 2`, which pumps water in at a fixed low duty cycle (`PumpIn(250)`) with flow integration for volume tracking, and `arm_state 8`, which uses the full PI controller to fill or drain to a set target.

For `arm_state 2`, early testing with the PI controller at full duty revealed an overshoot of approximately 80 mL, caused by hydraulic inertia continuing to drive flow after the pump was commanded to stop. Switching to a fixed low duty cycle of 250 significantly reduced trailing flow, and after tuning `PC_OVERSHOOT_COMP` from an initial value of 0.042 m

down to 0.0356 m, the system consistently reached target volumes within the acceptable margin. At a 500 mL target (0.32 m), the final measured volume confirmed a successful fill with no significant overshoot.

For `arm_state 8`, the drain direction was calibrated by adjusting `PC_DRAIN_COMP`. An initial value of 0.005 m produced a 20 mL undershoot at a 200 mL target — a 10% error — which was corrected by increasing the compensation to 0.018 m, halving the error to within 5%. On the fill direction, a target of 300 mL (0.192 m) was reached with a 10 mL undershoot, corresponding to a 3.3% error, confirming that the flow integration approach reliably estimates slider position and the compensation margins effectively account for pump inertia in both directions.

References

Luo, et al. (2025). *Toward Octopus-Inspired Whole-Arm Manipulation: A Sliding Pseudo-Joint for Soft Pneumatic Robot Arms.* RoboSoft 2025, Hong Kong University of Science and Technology.