

# Lumped-Parameter Modeling and Control for Robotic High-Viscosity Fluid Deposition

William van den Bogert<sup>1</sup>, James Lorenz<sup>1</sup>, Xili Yi<sup>2</sup>, Albert J. Shih<sup>1</sup>, and Nima Fazeli<sup>2</sup>

**Abstract**—Robotic high-viscosity fluid deposition plays a pivotal role in various manufacturing applications including adhesive and sealant dispensing, as well as in the additive manufacturing of deformable materials, such as those employed in soft robotics. Uncompensated high-viscosity fluid deposition can lead to poor part quality and defects due to large transient delays and complex fluid dynamics. In this paper, we propose a lumped-parameter flow model and compensation strategies to address significant transient delays and nonlinearity inherent in high-viscosity fluid deposition using a robotic manipulator. Our computationally efficient model is well-suited to real-time control and can be calibrated in minutes. Our compensation strategies leverage an iterative Linear-Quadratic Regulator to compute compensated deposition paths that can be deployed on robotic dispensing systems. These paths can either be deployed offline or corrected live via feedback from our proposed vision-based flow sensor. To validate the effectiveness of our approach, we conducted experiments extruding high-viscosity liquid silicone using a Kuka IIR iiwa robot. Comparative analysis with several baseline methods demonstrates that our proposed method significantly improves material deposition within desired boundaries.

## I. INTRODUCTION

Precise deposition of high-viscosity fluids is a key enabler for a range of applications including sealant deposition and additive manufacturing (AM), specifically the 3D printing of soft robotic components. Such printing of high-viscosity fluids, often intended to cure into deformable solids, is known as direct ink writing (DIW). The fluid’s high viscosity aids in maintaining the printed shape, yet without proper compensation, it can lead to printing flaws. Traditional material extrusion (MEX) AM follows a naive control approach, assuming linearity and no time delay between an actuator command and the outlet flow rate. However, when this approach is applied to DIW, the viscosity-induced impedance creates transient effects, resulting in deposition imprecision and issues like the stringing in Fig. 1a. Common industry solutions include retraction and priming, linear advancing, and coasting, but these are not informed by any model of the deposition process. As a result, significant defects may still occur, as we demonstrate in this paper. Furthermore, the parameters of these solutions are often hand-tuned which can require many iterations.

In this paper, we introduce a framework for modeling and control that enables DIW systems to approach the precision of MEX. Specifically, we contribute:

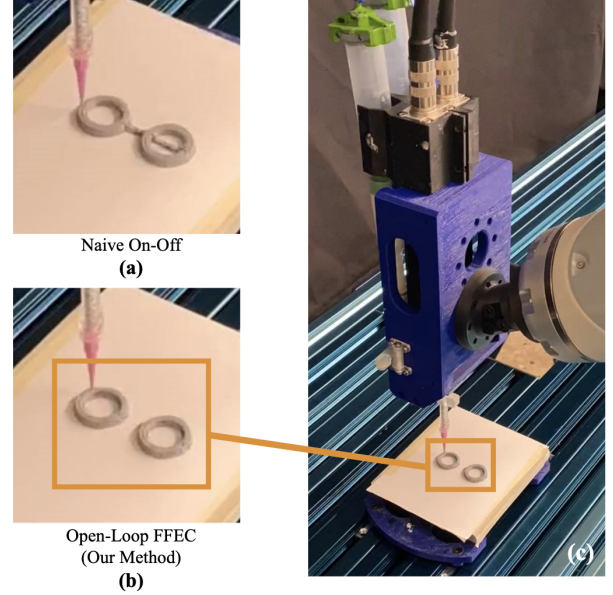


Fig. 1. Robotic 3D printing with model-based flow compensation. A silicone 3D print-head held by a robot arm printing two separated objects. Panel (a) shows the resultant print with naive on-off flow control where we observe significant stringing and part defects due to over-extruded silicone. In comparison, panel (b) shows the result of our open-loop feed forward error compensation (FFEC) where the defect-free objects are separated cleanly. Panel (c) shows the robot 3D printing silicone.

- a data-driven lumped-parameter model that is quickly and easily generalizable for most pump-based high-viscosity fluid deposition systems
- a universally implementable model-based open-loop feed-forward error compensation (FFEC) scheme without the need for additional sensors
- a model-based closed-loop control scheme using real-time data from a novel vision-based flow sensor

The target applications of our framework are industries where robotic manipulators are already widely present. Thus, our methods are directly applicable to existing systems in modern manufacturing plants. This practical implementation potential enhances the relevance and scalability of our approach. Using a Kuka robot, we achieve dynamic extruder movement in 3D space, allowing for dispensing on contoured surfaces, in the same way adhesive is applied at the perimeter of an automotive windshield. This facilitates conformal 3D printing, creating parts with intentional anisotropy or smooth finishes by layering materials on non-planar surfaces. This is crucial for crafting deformable parts to fit complex geometries, such as gaskets or prostheses.

<sup>1</sup> William van den Bogert, James Lorenz, and Albert J. Shih are with the Mechanical Engineering Department at the University of Michigan, MI, USA <willvdb, jplorenz, shiha>@umich.edu

<sup>2</sup> Xili Yi and Nima Fazeli are with the Robotics Department at the University of Michigan, MI, USA <yixili, nfz>@umich.edu

### A. Problem Setup and Statement

Our hardware setup can be seen in Fig. 2a. Two Positive Displacement Pumps (PDPs) force high viscosity uncured silicone through a static mixer and nozzle. Meanwhile, the arm moves this extruder in space to deposit the material in desired locations. Our novel vision-based flow sensor measures the flow rate leaving the nozzle, as feedback for our closed-loop controller. Our PDP system is detailed in Fig. 2b. We assume the fluids in reservoirs A and B behave identically, and that both motors receive the same input. Here, the control input is the motor speed signal as a function of time  $u(t)$ , which is proportional to the intended rate of displacement of the progressing cavities in the PDPs. This input produces the nozzle outlet flow rate as a function of time  $q(t)$ , and both  $u(t)$  and  $q(t)$  are in units of volumetric flow rate. While naive flow control assumes otherwise, we have observed that  $u(t) \neq q(t) \forall t$  (see Section IV for examples). For a given time horizon  $n$ , our goal is to solve for the trajectory  $u(t)$  such that a prescribed  $q(t)$  is achieved. We will discretize the time functions as  $u_k$  and  $q_k$ , defining the trajectories over the horizon  $n$  with time steps  $k = \{1, \dots, n\}$ . Let  $\mathcal{M}$  denote the model mapping the trajectories  $\mathbf{u}_{n \times 1}$  onto  $\mathbf{q}_{n \times 1}$ :

$$\mathbf{q}_{model} = \mathcal{M}(\mathbf{u}) \quad (1)$$

$\mathcal{M}$  is an approximation of the true map  $\mathcal{R}$  representing the complex nonlinear fluid dynamics. Our first contribution is developing a parametric model  $\mathcal{M}^*$  that is i) amenable to real-time control and ii) sufficiently closely approximates  $\mathcal{R}$ :

$$\mathcal{M}^* = \arg \min_{\mathcal{M}} |\mathcal{R}(\mathbf{u}) - \mathcal{M}(\mathbf{u})| \quad (2)$$

For open-loop FFEC, we do not have access to  $\mathcal{R}$ , and instead use the model map  $\mathcal{M}$  to find the optimal input trajectory  $\mathbf{u}^*$  which produces the prescribed output  $\mathbf{q}_{ref}$ :

$$\mathbf{u}^* = \arg \min_{\mathbf{u}} \|\mathcal{M}(\mathbf{u}) - \mathbf{q}_{ref}\|$$

subject to the constraints of the PDP pump and robot end-effector motion. The second contribution of this work is a strategy that leverages our proposed model to compute this optimal open-loop compensation.

For closed-loop control, we have access to a measurement of  $\mathcal{R}(\mathbf{u})$  after it is subjected to the sensor mapping  $\mathcal{S}$ :

$$\mathbf{q}_{meas} = \mathcal{S}(\mathcal{R}(\mathbf{u}))$$

The closed-loop control problem uses the sensor reading to find the optimal input trajectory  $\mathbf{u}^*$  which produces the prescribed output trajectory  $\mathbf{q}_{ref}$ :

$$\mathbf{u}^* = \arg \min_{\mathbf{u}} \|\mathcal{S}(\mathcal{R}(\mathbf{u})) - \mathbf{q}_{ref}\|$$

The third contribution of this work is a strategy that leverages our proposed model alongside real-time measurements to compute this optimal closed-loop control.

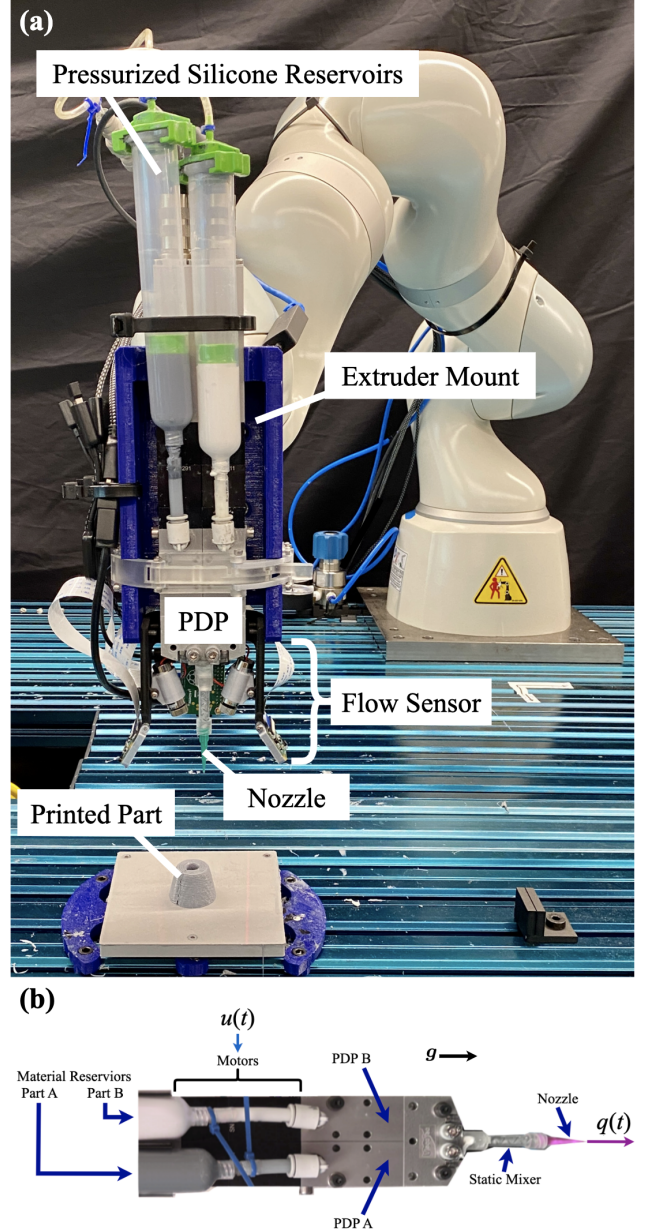


Fig. 2. a) Experiment setup for printing. The whole robotic printing system consists of the robot arm, print-head, and build plate. The print-head includes the extruder, pressurized silicone reservoirs, PDP, flow sensor, and nozzle. The flow sensor is a main contribution of this paper, as it allows use to measure and correct flow rate in real time. b) Positive displacement pump (PDP) system. Two progressive cavity pumps extrude A and B parts of the two-part liquid silicone. A and B parts are mixed via static mixing elements, and mixture is deposited through the nozzle. Here,  $g$  represents the direction of gravity.

## II. RELATED WORKS

Our work fills a gap in high-viscosity fluid deposition research related to transient flow dynamics in PDP systems for DIW. Existing research on fluid flow behavior in various PDP systems includes control schemes based on analytical solutions to physical laws [?] [?], physics-based simulations [?] [?] [?], and data-driven models [?] [?]. However, these approaches are either too computationally expensive for real-time control or lack consideration of transient effects in PDPs.

Baranovskii and Artemov have proposed generalized solu-

tions for optimal control of nonlinear fluid models based on the Navier-Stokes equations [?], though their analytical solutions demand simplifications specific to their system. This is also the case with the work of Josifovic et al, involving computational fluid dynamics (CFD) to model the step response of a valved pump system [?]. Hapanowicz's flow-resistance model [?] is only applicable to quasi-steady cylindrical flows and does not account for transient pump dynamics.

To control the fluid flow of PDP systems, researchers paired simplified models with closed-loop controllers and numerical solving methods in place of analytical physics-based solutions. Froehlich and Kemmetmüller propose a computationally efficient control [?], but it is designed only for injection molding systems. Wang et al combine pressure sensors and PID control to correct flow behavior [?]. Numerical solution methods have also been proposed for flow control of screw [?], helical gear [?], and peristaltic pump systems [?]. Though seldom used in fluid dynamics controllers, a data-based lumped parameter model was used by Fresia and Rundo [?] and Zardin et al [?] to compensate pulsing fluid dynamics generated by variable displacement pumps and vane pumps, respectively. However, because each group derived their models based on their specific PDP system, none are generalizable to the broad spectrum of PDPs.

Models for progressive cavity pumps are relatively unexplored in academic research. Fisch et al [?] and Franchin et al [?] note a strong improvement in precision when a progressive cavity pump is used instead of other PDPs. As a result, research on fluid interaction with progressive cavity PDPs has been limited to basic feedback control [?] and CFD methods to model specific behaviors of the progressive cavity pump system, including wear over time [?], backflow [?], as well as inertial effects and viscous losses [?]. While these models predict unsteady flow in internal regions of the progressive cavity systems, none provide comprehensive predictions of the entire dispensing system, which are required for our robotic dispenser.

To bridge these research gaps, we propose a model of the fluid system dynamics that can predict transient effects of high-viscosity fluid flows in PDP systems, which we then pair with both open-loop and closed-loop control paradigms to compensate for deposition errors during deposition.

### III. METHODS

Our approach is composed of both modeling and control. We will first introduce our modeling framework and then both our open-loop FFEC and closed-loop control strategies.

#### A. Modeling of High-Viscosity Fluid Deposition

Our goal is to design  $\mathcal{M}$  such that it i) is amenable to real-time control and ii) sufficiently closely approximates  $\mathcal{R}$  in Eq. 2, which represents the complex nonlinear fluid dynamics and interaction with both pump and mixer. We model the entire system as a 3 DOF coupled linear dynamic system, presented as the mass-spring-damper network in Fig. 3. Our model is a representation of fluid flow impedance in the form of elasticity  $k_j$ , damping  $c_j$ , and inertia  $m_j$  within both the pump ( $j = 1$ )

and mixer ( $j = 2$ ), as well as inertia  $m_f$  of the fluid that sits between them.

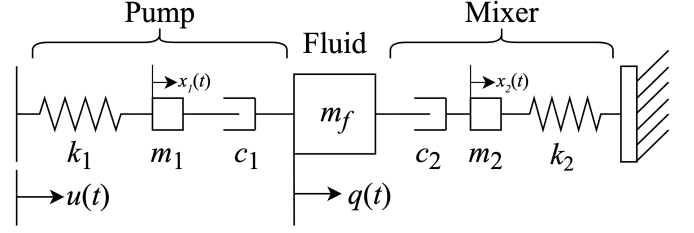


Fig. 3. The high-viscosity fluid, pump and mixer are simplified to a 3 degree of freedom coupled linear dynamical system. Our model is a representation of fluid flow impedance in the pump ( $j = 1$ ) and mixer ( $j = 2$ ) in the form of elasticity  $k_j$ , damping  $c_j$ , and inertia  $m_j$ , as well as inertia  $m_f$  of the fluid. The "positions"  $x_1, x_2, q$  of the "masses"  $m_1, m_2, m_f$  are the degrees of freedom that form the state vector  $\mathbf{x}$  alongside their time derivatives.

When coupled with the initial state of the system, our map  $\mathcal{M}$  introduced in Eq. 1 can take the form of a dynamics function  $f$  that predicts the next state vector  $\mathbf{x}$ , of which  $q$  and its time derivative are elements:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, u_k)$$

The state vector  $\mathbf{x}$  for the dynamics is defined using the 2 internal degrees of freedom  $x_1$  and  $x_2$  and the output  $q$ :

$$\mathbf{x} = [x_1 \quad x_2 \quad q \quad \dot{x}_1 \quad \dot{x}_2 \quad \dot{q}]^T$$

The system can be described with state-space form, with input  $u$  and output  $q$ :

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{A}\mathbf{x}_k + \mathbf{B}u_k \\ q_k &= \mathbf{C}\mathbf{x}_k \end{aligned}$$

where the matrices  $\mathbf{A}$ ,  $\mathbf{B}$  and  $\mathbf{C}$  are as follows:

$$\mathbf{A}_s = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -\frac{k_1}{m_1} & 0 & 0 & -\frac{c_1}{m_1} & 0 & \frac{c_1}{m_1} \\ 0 & -\frac{k_2}{m_2} & 0 & 0 & -\frac{c_2}{m_2} & \frac{c_2}{m_2} \\ 0 & 0 & 0 & \frac{c_1}{m_f} & \frac{c_2}{m_f} & -\frac{c_1+c_2}{m_f} \end{bmatrix}$$

$$\mathbf{A} = \mathbf{I}_{6 \times 6} + \mathbf{A}_s \Delta t$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & \frac{k_1}{m_1} & 0 & 0 \end{bmatrix}^T \Delta t$$

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$$

Given measured data from the real system, the parameters are found as the solution to a least-squares optimization. The cost function  $c$  is a 2-norm loss for a given time horizon  $n$  as a function of the parameters  $\phi$ , where the data is sampled with the same period as the simulation time-step  $\Delta t$ :

$$c(\phi) = \sum_{k=1}^n (q_{k,model}(\phi) - q_{k,meas})^2 \quad (3)$$

$$\phi = [k_1 \quad c_1 \quad m_1 \quad m_f \quad k_2 \quad c_2 \quad m_2]^T$$

Where  $q_{k,meas}$  is the measured data for the time with index  $k$  and  $q_{k,model}$  is the modeled flow rate for the time with index  $k$ . For each iteration of optimization  $i$  the model is solved and

the gradient of the cost computed. Then, the parameters are updated according to that gradient:

$$\phi_{i+1} = \phi_i - h \frac{\partial c(\phi_i)}{\partial \phi_i}$$

where  $h$  is an adjustable increment. We set this optimization to stop once the cost changes by less than 0.1% from one iteration to the next.

### B. Open-Loop Control of High-Viscosity Fluid Deposition

With this model in hand, we desire the input profile  $u$  for a given time horizon such that the nozzle flow rate tracks the desired  $q$ . We construct a quadratic cost function  $b$  for this optimization:

$$b_k = (\mathbf{x}_k - \mathbf{x}_k^{ref})^T \mathbf{Q} (\mathbf{x}_k - \mathbf{x}_k^{ref}) + \mathbf{u}_k^T \mathbf{R}_k \mathbf{u}_k \quad (4)$$

where  $\mathbf{x}_k$  is the current state,  $u_k$  is the current input,  $\mathbf{x}_k^{ref}$  is the desired state at the time with index  $k$ .

In our specific case, where our input only has a single degree of freedom,  $\mathbf{R}_k$  and  $\mathbf{u}_k$  reduce to scalars, so Eq. 4 can be rewritten:

$$b_k = (\mathbf{x}_k - \mathbf{x}_k^{ref})^T \mathbf{Q} (\mathbf{x}_k - \mathbf{x}_k^{ref}) + R_k u_k^2$$

where  $\mathbf{Q}$  is a diagonal matrix containing the weightings for tracking error of the 6 states. Since we only are trying to track the output  $q$ , the third state, all other members of the diagonal can be set to some small value  $\delta$ :

$$\mathbf{Q}_{6 \times 6} = \text{diag}(\delta, \delta, \xi, \delta, \delta, \delta)$$

With the linear dynamics of our model, the cost function defined in Eq. 4 could ordinarily be minimized analytically within the bounds of linear quadratic regulator (LQR) theory. However, we have observed that the reference trajectory is tracked more effectively in reality if the weighting on actuation effort  $R_k$  is significantly reduced ( $r_1 < r_2$ ) once the input  $u_k$  falls below a certain negative value  $u_{th}$ , when the pumps are run backwards in an effort to rapidly shutoff the output flow when commanded as such:

$$R_k = \begin{cases} r_1 & u_k < u_{th} \\ r_2 & u_k \geq u_{th} \end{cases}$$

Since the coefficient of actuation effort within the quadratic cost changes depending on the magnitude of the input, we use an iterative LQR (iLQR) solver [?] for this optimization problem. Each iteration  $i$  occurs in two parts. The first is a backward pass in which the controller gains  $\mathbf{K}_k$  are computed using the value function  $\mathbf{V}_k$  at the next time step, which is then updated for the current time step:

$$\begin{aligned} \mathbf{K}_k &= (R_k + \mathbf{B}^{*T} \mathbf{V}_{k+1} \mathbf{B}^*)^{-1} \mathbf{B}^{*T} \mathbf{V}_{k+1} \mathbf{A}^* \\ \mathbf{P}_k &= \mathbf{A}^* + \mathbf{B}^* \mathbf{K}_k \\ \mathbf{V}_k &= \mathbf{Q}_k^* + \mathbf{K}_k^T R_k \mathbf{K}_k + \mathbf{P}_k^T \mathbf{V}_{k+1} \mathbf{P}_k \end{aligned}$$

where  $\mathbf{A}^*$ ,  $\mathbf{B}^*$ , and  $\mathbf{Q}_k^*$  are the homogenized versions of  $\mathbf{A}$ ,  $\mathbf{B}$ , and  $\mathbf{Q}$  respectively:

$$\mathbf{A}^* = \begin{bmatrix} \mathbf{A} & \mathbf{0}_{6 \times 1} \\ \mathbf{0}_{1 \times 6} & 1 \end{bmatrix}, \quad \mathbf{B}^* = \begin{bmatrix} \mathbf{B} \\ 0 \end{bmatrix}$$

$$\mathbf{Q}_k^* = \begin{bmatrix} \mathbf{Q} & \mathbf{Q}(\mathbf{x}_k - \mathbf{x}_k^{ref}) \\ (\mathbf{x}_k - \mathbf{x}_k^{ref})^T \mathbf{Q} & (\mathbf{x}_k - \mathbf{x}_k^{ref})^T (\mathbf{x}_k - \mathbf{x}_k^{ref}) \end{bmatrix}$$

The second part is a forward pass in which the input  $u_k$  is updated according to the controller gains, and the state  $\mathbf{x}_k$  is updated according to the dynamics with the input:

$$\begin{aligned} u_k^{i+1} &= u_k^i + \mathbf{K}_{k-1}(\mathbf{x}_{k-1}^{i+1} - \mathbf{x}_{k-1}^i) \\ \mathbf{x}_{k+1}^{i+1} &= \mathbf{A} \mathbf{x}_k^{i+1} + \mathbf{B} u_k^{i+1} \end{aligned}$$

We set the controller optimization to stop if the cost changes by less than 0.1% from one iteration to the next.

### C. Closed-Loop Control of High-Viscosity Fluid Deposition

To close the loop on flow rate control, we develop a novel vision-based sensor to measure the just-printed bead. Our sensor projects a laser line from a diode onto the printed bead behind the forward-traveling nozzle, and observing the distortion of the line from a Raspberry PI (RasPI) camera.

At time index  $k$ , the sensor directly measures the cross sectional area in pixels of the bead in the camera frame  $A_{p,k}$ . We assume  $A_{p,k}$  scales linearly to the true cross sectional area in physical units  $A_{b,k}$  by proportionality constant  $K = A_{p,k}/A_{b,k}$ . Through a control volume analysis of the recently printed bead, we represent flow rate  $q_k$  as a function of both area  $A_{p,k}$  and nozzle velocity  $v_k$ :

$$q_k = \frac{1}{K} A_{p,k} v_k \quad (5)$$

With two independent variables, this function can be represented as a surface. We identified  $K$  as the solution to another least-squares optimization, where the surface defined by Eq. 5 is fit to real data. We collected data at various steady state flow rates and velocities, and measured  $A_{p,k}$  with the sensor. This measurement involves two trained segmentation models for the nozzle and laser line to be masked from the RasPI camera feed respectively. The sensing procedure for finding  $A_{p,k}$  is visualized in Fig. 4.

The flow sensor we introduce in this paper provides the unique ability to perform closed-loop control of fluid deposition. A similar controller to the open-loop version is used, but it is continually solved over a horizon  $T_{CL}$  that follows the current time, using an estimation for the current flow rate as an initial condition.

After acquiring the sensor reading, we apply a Kalman filter to generate the flow rate estimation  $q_{init}$  and its time-derivative  $\dot{q}_{init}$ , approximated by finite differences. The open-loop controller described in Section III-B is then initialized with the estimated state-vector:

$$\mathbf{x}(0) = \begin{bmatrix} q_{init} & 0 & q_{init} & 0 & 0 & \dot{q}_{init} \end{bmatrix}^T$$

We give the solver a reference trajectory  $q_{ref}$  for the horizon  $T_{CL}$ :

$$\mathbf{x}_{ref} = \begin{bmatrix} 0 & 0 & q_{ref} & 0 & 0 & 0 \end{bmatrix}^T$$

Once the optimal input  $u_k$  is found, its initial value  $u_0$  is sent to the pumps. This input to the pumps is low-pass filtered



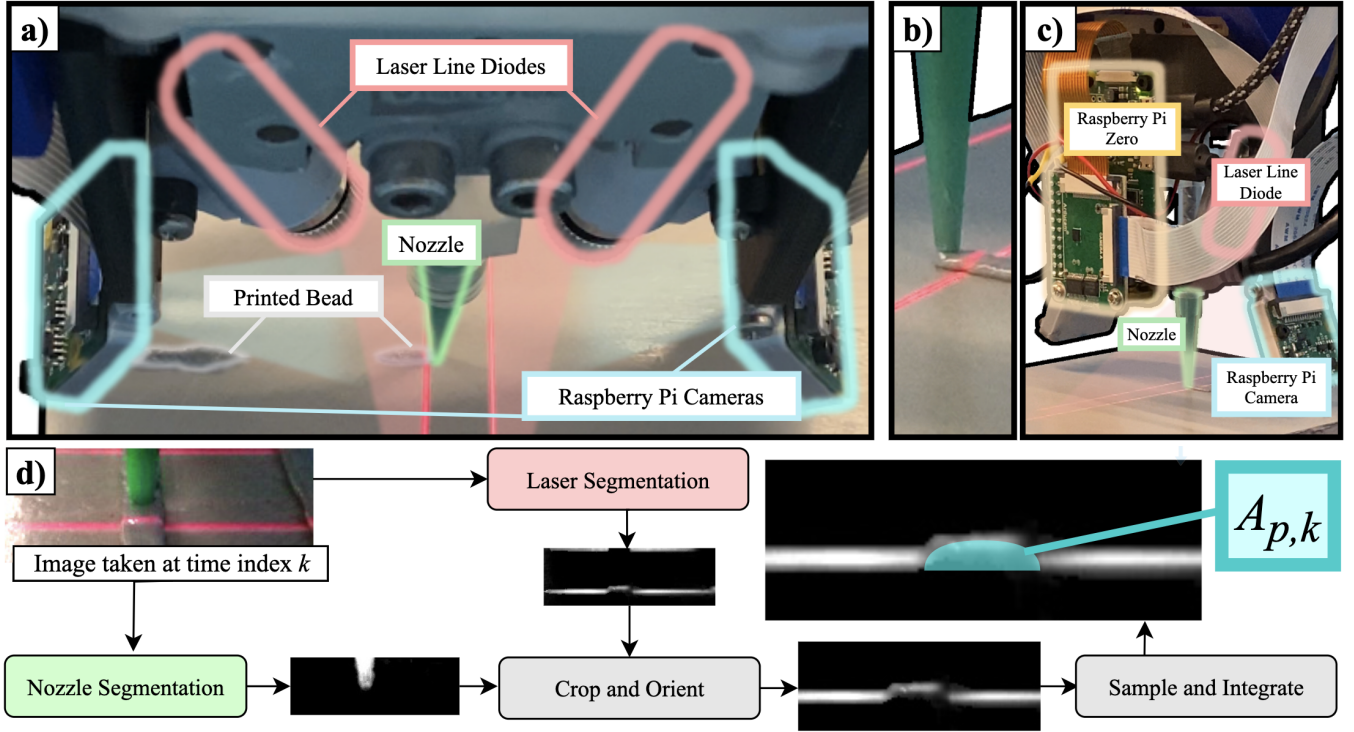


Fig. 4. Novel flow sensor detail and bead area measurement visualization. a) Top view of sensor, showing laser line diodes and cameras underneath the PDPs. b) Close-up of laser line distortion caused by the printed bead. c) Rear view of sensor, showing the Raspberry Pi Zero that collects and sends image data over the ROS network. d) Visualization of bead area measurement. Segmentation of nozzle is used to locate the center of the image (where the printed bead should be) and the laser segmentation is cropped accordingly. The mean location of brightness of each column in the laser segmentation is used to sample the curve of the laser distortion, which is discretely integrated to return the bead area in pixels  $A_{px}$ .

in real-time. The assumption for this procedure is that the duration required for solving the controller is less than  $t_{sens}$ . We wrote the program such that the controller solves every time the sensor has a new measurement, which occurs after each period equal to  $t_{sens}$ .

#### IV. RESULTS

##### A. Experimental Setup and Data Collection

Our experimental setup can be seen in Fig. 2 and detail of the flow sensor can be seen in Fig. 4. We used a 7DOF Kuka LBR 14 R820 robot arm equipped with a customized print-head alongside an adjustable build plate fixed to the base of the world frame. The customized print-head consists of the extruder mount, pressurized reservoirs, PDPs, mixer, nozzle, and flow sensor. The silicone material used for deposition is Dow Corning 121 Structural Glazing Sealant. The stepper motors in the PDPs are controlled by a Raspberry Pi 4 Model B using integration with Python and ROS, while the sensor operates through a Raspberry Pi Zero that communicates with the same ROS network. The silicone reservoirs are connected to a high-pressure air source limited at 100 psi by a pressure regulator.

We collected all data for model training and verification, as well as control verification, with our novel vision-based flow sensor, which is described in Section III-C. The sensor calibration process provided the parameter  $K = 213.7 \text{ px/mm}^2$  for Eq. 5. The data for this calibration and the best fit surface defined by  $K = 213.7 \text{ px/mm}^2$  are shown in Fig. 5.

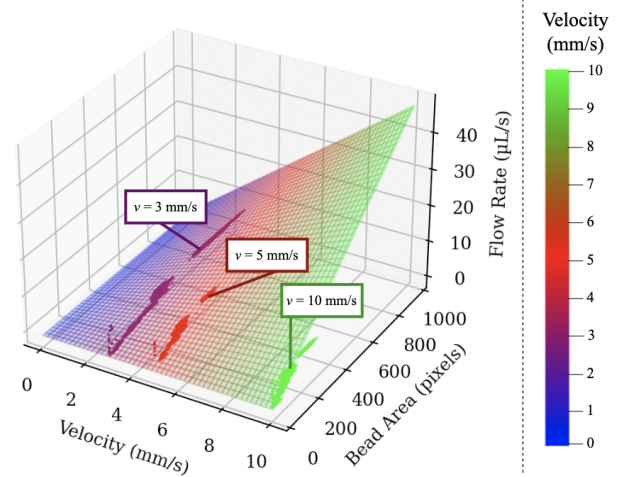


Fig. 5. Sensor calibration surface  $q(A_{px}, v)$  described in Eq. 5, with  $K = 213.7 \text{ px/mm}^2$ , alongside data collected at nozzle speeds 3, 5, and 10 mm/s.

##### B. Model Training

To implement the model training algorithm described in Section III-A, we used values of  $h = 0.1$  for the step increment, and  $\Delta t = 0.01 \text{ s}$  for the simulation time step. The data seen in Fig. 6 was collected by the sensor over 4 trials.

We trained the model on an input command trajectory consisting of 10 square pulses of different magnitudes, both negative and positive, as seen in Fig. 6. This was to anticipate

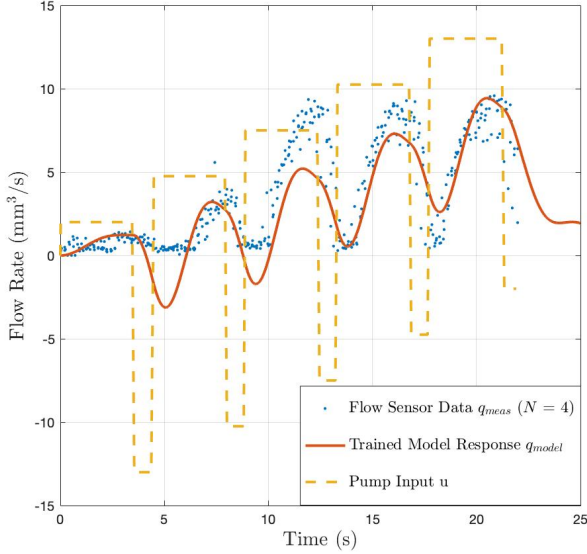


Fig. 6. The lumped-parameter fluid deposition model was trained on 4 sets of data collected live from the flow sensor while the pumps were given the command  $u$  over a horizon of 22 seconds. Due to the real non-linear and specifically non-Newtonian nature of the silicone, a linear model cannot accurately predict across the entirety of a wide range of flow rate magnitudes. Additionally, while negative flow is predicted by the model, the sensor has no means to measure it, though it may truly occur.

the necessary distribution for compensating step-up and step-down flow rate trajectories at the nozzle. The optimization returned the parameters shown in Table I.

TABLE I  
PARAMETERS FOUND AS A RESULT OF MODEL TRAINING

| $k_1$ | $c_1$ | $m_1$ | $m_f$ | $k_2$ | $c_2$ | $m_2$ |
|-------|-------|-------|-------|-------|-------|-------|
| 1.77  | 18.81 | 1.02  | 1.78  | 9.87  | 5.33  | 1.11  |

With these parameters defined, the model exhibits the behavior shown in Fig. 6. The trained model can be used to predict the data used in training. Fig. 7 shows the ability of the trained model to predict unseen data.

### C. Control and Compensation

We tested both our open-loop FFEC and our closed-loop controller against 2 industry standard solutions, for tracking both “pulse” and “dip” set-point flow rate trajectories. The pulse task demonstrates the ability of the flow rate to turn on and off, especially repeatedly, as it may while printing small areas during layered 3D printing. The dip task demonstrates the ability of the flow rate to drop briefly before returning to a nominal value, as it should to avoid bulging during deceleration of the nozzle while dispensing in a tool path that contains sharp turns. All data was collected via the sensor described in Section III-C. For all tests, the desired flow trajectory, pump input trajectory, and cartesian trajectory of the robot were sampled at a period of  $\Delta t_{sens} = 0.2$  s for commanding the arm and PDPs. We chose parameters for industry standard solutions based on typical values for MEX.

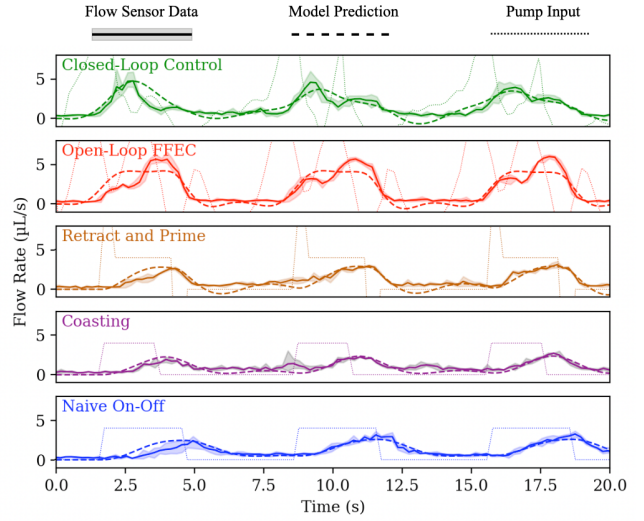


Fig. 7. Prediction of unseen data by the lumped-parameter model. With the parameters from trained model, the model is able to predict unseen data, consisting of the flow rate input to the pumps and output measured by the flow sensor from the pulse validation experiment.

To implement the open-loop FFEC solver described in Section III-B, we used values of  $\xi = 100$  and  $\delta = 0.01$  for the tracking weight, and  $r_1 = 20$ ,  $r_2 = 200$ ,  $u_{th} = -2$  mm<sup>3</sup>/s for the actuation effort weighting. The desired flow trajectory, generated with a time step of  $\Delta t_{sens} = 0.2$  s, was linearly interpolated to  $\Delta t = 0.001$  s for solving.

Our closed-loop controller described in Section III-C was implemented with values of  $\xi = 100$  and  $\delta = 0.01$  for the tracking weight and  $r_1 = r_2 = 80$  for the actuation effort weighting,  $T_{CL} = 5.0$  s for the solving horizon,  $\Delta t = 0.06$  s for the simulation time step, and the stop condition for the optimization was changed to stop once the cost changes by less than 3% from one iteration to the next.

For the pulse task, we commanded a series of 3 square pulses with a magnitude of 4 μL/s. The industry standard solutions applied to this task were retract/prime and coasting. For retract/prime, at each rising edge 10 μL/s is commanded for 0.4 s, and at each falling edge -10 μL/s is commanded for 0.4 s. For coasting, the falling edge is commanded to the pumps 1 s earlier than the desired output. The pulse task was tested 3 times, for a total of 9 pulses sent to each control method. The mean of all 9 output pulses is shown in Fig. 8, for each of the 4 control methods alongside the naive control, in which the set-point is used as  $u(t)$ . The mean-absolute-error (MAE) between the flow trajectories produced by each control method and the set-point can be seen in Table II, showing an improvement of over 40% when using our methods.

For the dip task, we commanded the printer to extrude at 4 μL/s for a period of time before briefly shutting off and returning to 4 μL/s. The industry standard solutions applied to this task were linear advance and no shutoff. We consider no shutoff an industry solution because often no flow compensation is applied for sharp corners in the tool path. For linear advance, the pump motors decelerate to 0 μL/s early before accelerating to 7 μL/s briefly, then returning to

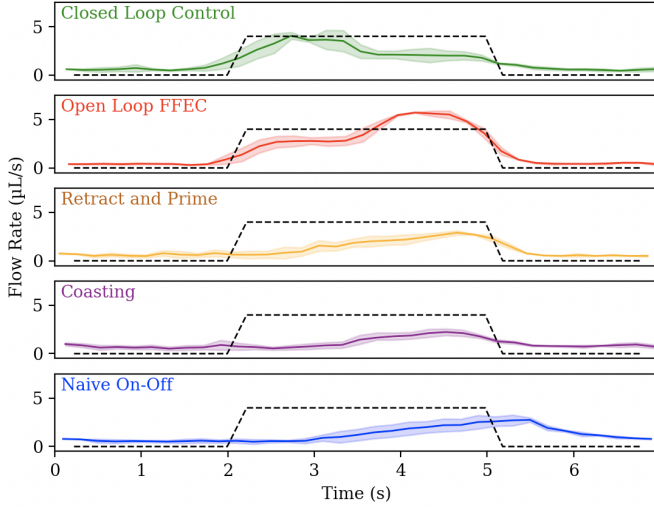


Fig. 8. Results of pulse test. The dotted line represents the set-point.

TABLE II  
MAE RESULTS FOR PULSE TASK

|                     | MAE             | % Improvement |
|---------------------|-----------------|---------------|
| Closed Loop Control | $0.92 \pm 0.28$ | 46            |
| Open Loop FFEC      | $0.81 \pm 0.20$ | 52            |
| Retract and Prime   | $1.26 \pm 0.16$ | 25            |
| Coasting            | $1.52 \pm 0.17$ | 10            |
| Naive On-Off        | $1.69 \pm 0.21$ | 0             |

4  $\mu\text{L/s}$ . No shutoff involves a constant 4  $\mu\text{L/s}$  commanded to the pumps. We tested the dip task 3 times as well, The mean of the 3 controller responses can be seen in Fig. 9, for each of the 4 control methods alongside the naive shutoff control, in which the set-point is used as  $u(t)$ . The MAE between the flow trajectories produced by each control method and the set-point can be seen in Table III, showing an improvement of over 40% using our methods.

We also evaluated the pulse task using observations of material placement using a top view of the print bed after the

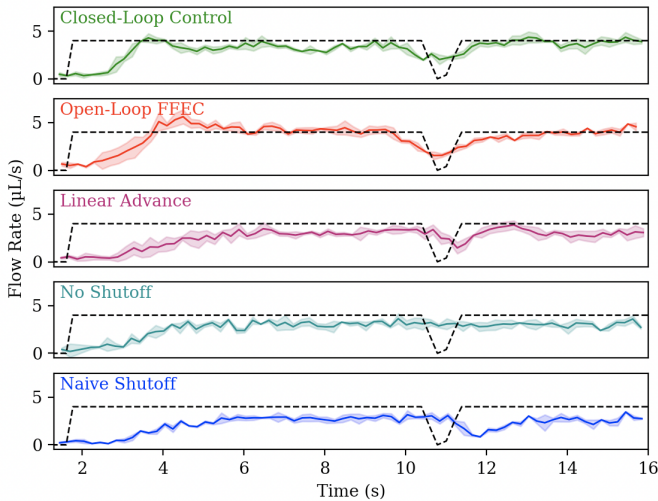


Fig. 9. Results of dip test. The dotted line represents the set-point.

TABLE III  
MAE RESULTS FOR DIP TASK

|                     | MAE             | % Improvement |
|---------------------|-----------------|---------------|
| Closed Loop Control | $0.88 \pm 0.30$ | 49            |
| Open Loop FFEC      | $0.82 \pm 0.34$ | 53            |
| Linear Advance      | $1.40 \pm 0.41$ | 20            |
| No Shutoff          | $1.25 \pm 0.32$ | 28            |
| Naive Shutoff       | $1.74 \pm 0.22$ | 0             |

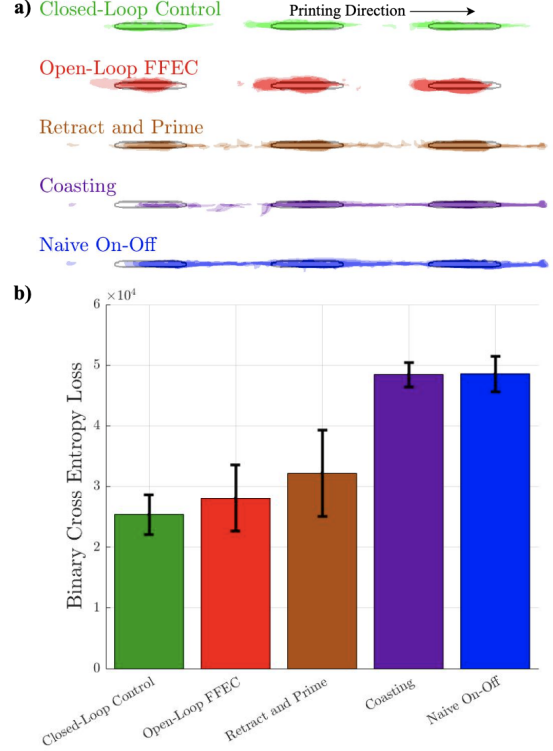


Fig. 10. The printing of 3 pulses analyzed by material placement, using a binary cross entropy loss between the target mask and the result mask. a) The black outlines indicate the boundary of desired material placement, while the colored areas represent resulting material placement. Higher opacity indicates material placement during multiple trials. b) Visualization of BCE results.

deposition test. The 3 trials were done again and the results were photographed, automatically segmented to a binary mask using MATLAB R2020b, and compared with a grayscale mask of the desired material placement using binary-cross entropy (BCE) loss as an evaluation method. This grayscale mask has a small dilation blur to less harshly penalize material that is slightly outside of the intended boundary. The masking comparisons and BCE results can be seen in Fig. 10 and Table IV, showing an improvement of over 40% using our methods.

TABLE IV  
BCE RESULTS FOR PULSE TASK

|                     | BCE/ $10^4$   | % Improvement |
|---------------------|---------------|---------------|
| Closed Loop Control | $2.5 \pm 0.3$ | 49            |
| Open Loop FFEC      | $2.8 \pm 0.5$ | 43            |
| Retract and Prime   | $3.2 \pm 0.7$ | 35            |
| Coasting            | $4.8 \pm 0.2$ | 2             |
| Naive On-Off        | $4.9 \pm 0.3$ | 0             |

## V. DISCUSSION AND LIMITATIONS

In this paper, we demonstrate how after quickly identifying just 7 parameters that define our flow model, we can apply the model to either an open-loop or closed-loop approach of correcting flow errors. We experimentally validated our approaches with deposition tests using our robotic 3D printer. Misplaced material during printing was significantly reduced due to the application of our model-based compensation and control techniques. Fig. 1a shows that the naive on-off pump input leads to unintended deposition amongst two 3D printed rings. In Fig. 1b, the two objects are separated cleanly when open-loop FFEC is applied. Beyond AM, our compensation enhances material placement precision along linear trajectories, as depicted in Fig. 10. This improvement has the potential to reduce error for robotic deposition of sealant and adhesive.

The versatility of our linear model theoretically extends to any material or PDP, provided it is retrained on the system that will be used during the final application. While material properties can change with ambient conditions such as temperature, this work assumes training and testing occur in the same environment; this could be considered a limitation. A physics-based model would be beneficial in adapting the controller to changing ambience. While our model's predictions are satisfactory for this paper's intent, Fig. 6 and Fig. 7 show predictions that are missing some measured features, likely due to sensor and/or model inaccuracy. Our linear model lacks wholly predictive capabilities for non-Newtonian silicone behavior. A more expressive model of the fluid dynamics could mitigate eccentricities of predictions seen in Fig. 7, especially since iLQR is compatible with nonlinear systems [?].

In its current form, the flow sensor is limited to sensing single line beads printed straight. For 3D printing, it must be able to align the laser/camera pairs with the direction of nozzle travel. The sensing scheme must also involve comparison between the front (where material will be soon) and rear (where material has just been placed) camera views. Open-loop FFEC is already applicable to 3D printing (see Fig. 1).

One motivation to keep our model computationally inexpensive is to allow further improvements of closed-loop control, regardless of the type of sensor or control scheme used. The flow sensor and closed-loop control scheme used in this paper both require a significant amount of processing time, which has limited us to a loop time of 0.2 s. This is perhaps why closed-loop control manages to consistently perform just under open-loop FFEC according to data from the flow sensor, as shown in Tables II and III. While the closed-loop control must be conducted in real-time, the FFEC can optimize the input trajectory for any amount of time before it is deployed to the real system. However, according to Table IV and Fig. 10, closed-loop control outperforms FFEC. This discrepancy is likely due to differences in how the data are registered to the desired result. While this may suggest that it makes little difference whether our closed- or open-loop solutions are used, closed-loop control has unique advantages that stem from its reliance on real-time observations, potentially outperforming FFEC on unseen systems. We can expect the FFEC to underperform on such systems, where the mixer, material, or pump may

have changed from training. A higher frequency closed-loop control scheme coupled with a more accurate sensor could better eliminate errors in real-time. Additionally, the nozzle trajectory can be optimized alongside the flow rate so that the transient delay still present in the corrected output (see Fig. 8 and Fig. 9) will not be a permanent barrier to precise material placement. Through this we will achieve even further improvements than those demonstrated in this paper.