

A Learning-based Approach to Modeling and Control of Inkjet 3D Printing

Uduak Inyang-Udoh¹ and Sandipan Mishra²

Abstract—This paper presents a learning-based approach to modeling and control of inkjet 3D printing. First, we propose and experimentally validate a learning-based model for inkjet 3D printing. The proposed model uses a physics-based model paradigm that has been reformulated into a neural-network-like structure. This formulation enables back-propagation and the associated benefits of data-driven model identification while retaining physical interpretation of the learned model itself. Next, we propose and demonstrate a predictive control algorithm that leverages the neural-network-like structure of the model. Back-propagation is used for efficient gradient calculations to determine optimal control inputs, namely droplet patterns for subsequent layer(s), to optimize a quadratic cost function.

I. INTRODUCTION

Inkjet 3D printing is an additive manufacturing technique that builds parts by ejecting liquid material onto a substrate. Due to its ability to form complex patterns especially of polymers, ceramics and metal-based parts, the technique is attractive for manufacture of organic electronics as thin-film transistors and diodes and solar cells; sensors and detectors; and parts for biomedical applications (including building synthetic tissues and scaffolds) [1]–[3]. The printing process involves jetting photo-polymers or suspensions from a nozzle - by thermal or piezoelectric action - onto a substrate, and curing the ejected material as shown in Fig. 1. As droplets are sequentially deposited layer after layer, and cured, the desired 3D part forms. [1], [4]–[6]

The inkjet printing dynamics involves deformation of the liquid droplets on deposition, spreading, and coalescence with previous depositions to form beads or lines. These dynamics occur at the interplay of surface tension, contact line motion, gravity and fluid viscosity [7]–[11]. The inherent complexity of the interplay makes the development of purely physics-based model difficult or computationally expensive, especially since the depositions are swift and the dynamics are non-planar.

For geometry-level (3-dimensional) control of several droplet-based additive manufacturing processes, control-oriented models have been developed that simplify or approximate the complex physics to capture only essential dynamics of the problem. In some models, the height topology is assumed to grow simply by a super-position of droplets [9], [12]–[15]. Others assume interaction between the liquid

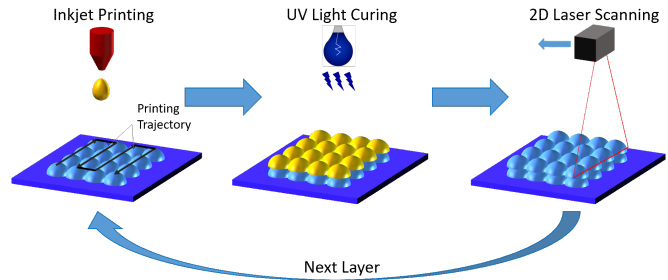


Fig. 1: Steps in printing a layer. First, the nozzles deposit ink on an existing layer following a predetermined trajectory. Then, the printed layer is cured under ultraviolet (UV) light. Finally, a 2D laser sensor scans the layer. Adapted from [18].

material as the topology grows [16]–[18]. In either case, with linear (time-invariant or parameter-varying) behavior assumed, the droplet-to-droplet height evolution may be lifted to give a layer-to-layer height evolution model [12], [15], [17], [18]. In [17] and [18], a fixed droplet shape and linear liquid flow are assumed; then a lifted representation relating the height profiles across layers is derived. The model is valuable for layer-to-layer control but the assumption of constant drop shape and linear flow limits how much printing feature the model can capture. In [15], no prior liquid interaction is assumed; rather the model is estimated for each layer by running a numerical simulation and performing a model fitting on the data obtained from the simulation. Given the layer input, this approach yields greater model accuracy since it can capture actual non-linearities associated with the height evolution because of surface tension. However, this numeric simulation may prove very expensive and requires accurate calibration of physical properties of the ink and substrate. In addition, the approach assumes an estimate of the droplet volume at each point of deposition; but from a control standpoint, this estimate may differ significantly from the optimal volume at the point.

In this paper, we consider the height evolution from one deposition to another and capture the evolution by time-invariant parameters. While we still avoid explicit complex dynamics associated with the morphology of the droplet deposition, a machine learning-based model is used to embody the effect of those dynamics. The model we present lends itself to a recurrent neural network (RNN), which may be trained (and further refined online) to predict subsequent layer height profiles. Though the model parameter identification is data-driven, our insight into the process in formulating

¹ Uduak Inyang-Udoh is with the Mechanical, Aerospace and Nuclear Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180 USA inyanu@rpi.edu

² Sandipan Mishra is with faculty of the Mechanical, Aerospace and Nuclear Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180 USA mishrs2@rpi.edu

the model (i.e., geometric proximity of nodes, ink flowability, and surface tension) results in a specific model structure and thus lowers the need for large amounts of data for training the model. We show the ability of the model to capture essential evolution dynamics by comparing its prediction with that of [18] on experimental data.

We then demonstrate how the model may be employed for feedback control of the printing process in a predictive control framework. To find the controlled input, we parameterize our model by the input pattern, and seek to identify this input ‘parameter’. Hence, both model learning and predictive control are complementary optimization problems: in one case, the model constants are to be learned given the input and output profile; in the other, the input is optimized given the model constants and a reference profile; thus gradient calculations used for one are in fact useful for the other. *In other words, model-prediction error is used to update the model parameters meanwhile the reference-feedback error is used to generate optimal input based on updated parameters.*

The key contributions of this paper are

- the formulation and experimental validation of a learning-based (constrained-flow RNN) model of the inkjet 3D printing process; and
- the development of a predictive control framework for implementing feedback geometry-level control in 3D printing for reference profiles and geometries that vary from layer to layer.

The paper is organized as follows. We describe the form of the model function we intend to develop in Sec II. In Sec III, we present the learning-based model and highlight strategy for learning model parameter. We pose the control problem based on the developed model in Sec IV. The fifth section shows results from learning and validation of model parameters. Simulation result for the model-based feedback control is presented in Sec VI. We conclude and preview further work in Sec VII.

II. PROBLEM DESCRIPTION

The basic scheme of the closed-loop inkjet 3D printing process we develop is shown in Fig. 2. The solid object to be printed is sliced in horizontal layers and each layer is resolved into a n_x by n_y grid space to obtained a discretized height distribution (or profile) for each layer. The reference height profile, input pattern and output are given by $R^{(L)}, U^{(L)}, Y^{(L)} \in \mathbb{R}^{n_x \times n_y}$. In this paper, we aim to obtain a phenomenological model that relates a given pattern $U^{(L)}$ to the new substrate (height) profile $Y^{(L)}$. Also, because the reference height profile $R^{(L)}$ for each layer is unique, we propose to utilize feedback information (that is, the measured profile) to update the model parameters. Then, we find the optimal control input for subsequent layer(s) using the updated model.

Since the droplets are deposited in sequence, suppose that N_L time steps (or sequences) are required to build layer L , then, we may define $U_k \in \mathbb{R}^{n_x \times n_y}$, $k \in [0, N_L - 1]$ as the admissible input space at time step k . Note that U_k is sparse, holding only a non-zero number at (x, y) -position

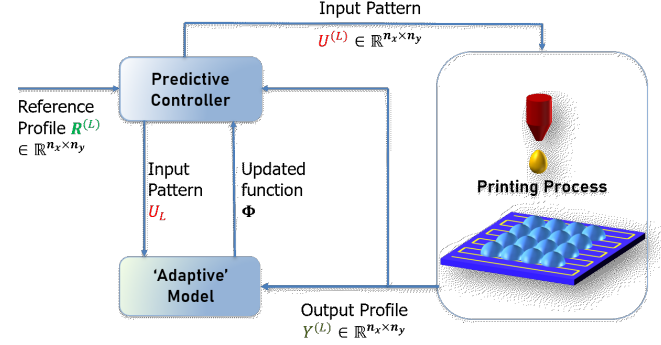


Fig. 2: 3D inkjet printing scheme. For each layer, a reference profile based on the desired part geometry is fed to a controller which generates suitable input sequence for the printing module to execute.

($x \in n_x, y \in n_y$) where deposition occurs at k , that is, $\sum_{k=0}^{N_L-1} U_k = U^{(L)}$. Similarly, let $H_k \in \mathbb{R}^{n_x \times n_y}$ denote the height profile at k^{th} step. Then, we seek the function Φ parameterized on θ such that $H_{k+1} = \Phi(\theta, H_k, U_k)$, $k \in [0, N_L - 1]$ and $\hat{Y}^{(L)} = H_{N_L}$. We find θ^* that minimizes the error $\|Y^{(L)} - \hat{Y}^{(L)}(\theta)\|_2^2$.

For model predictive control, we re-write the form of Φ such that the control input becomes the optimization variable, that is, $H_{k+1} = \Phi(\theta, H_k, U^{(L)})$. Given the reference height profile $R^{(L)}$, the control objective then effectively becomes to find the optimal control input pattern $U^{(L)*}$ that minimizes the error $\|R^{(L)} - \hat{Y}^{(L)}(U^{(L)})\|_2^2$.

In the following sections, the function Φ will be established. We revisit the time-varying graph-based model of [18] and modify the model to incorporate a threshold function to capture unmodeled surface tension effects. We thus eliminate the time dependence of the model parameters, making the model amenable to online learning.

III. GEOMETRY LEVEL DYNAMIC MODELING OF INKJET 3D PRINTING

In this section, we present a constrained-flow model for the printing process that combines physics-based model structure with a neural network modeling strategy. First, we discuss a geometry-level graph-based model of [18] in the context of the problem description. In this model, the flow of the liquid material is driven by gravity. We then present a reformulated structure in which the flow is restricted and the model parameters are now rendered time-invariant.

A. Graph-Based Dynamic Model

The graph-based dynamic model assumes that the height distribution at a given time step is a linear combination of a deposited droplet’s height distribution, and gravity-driven flow within the already-deposited material from the previous time step. Let $n = n_x \times n_y$ and the profile $H_k \in \mathbb{R}^{n_x \times n_y}$ be written as vector $h_k \in \mathbb{R}^n$, the height evolution is given by

$$h_{k+1} = A_k h_k + B_k u_k. \quad (1)$$

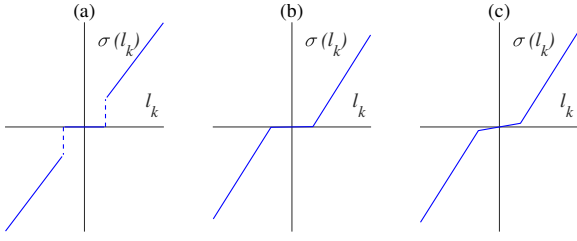


Fig. 3: Potential activation functions to capture surface tension inhibition to flow across links. The *leaky soft-threshold* function (c) is used.

$A_k = (I - DF_k D^T)$ is the state matrix that captures the effect of liquid flow from higher to lower grid points, $D \in \mathbb{R}^{n \times l}$ being the incidence matrix for the grid with l links, and $F \in \mathbb{R}^{l \times l}$, a diagonal positive (semi-) definite matrix containing flowability parameters determined empirically. (The reader is referred to [18] for more details). $B_k \in \mathbb{R}^n$ is the vector containing a unit droplet shape and location at this time step; u_k is the droplet volume.

Remark 1: With the construct of this model, the A matrix is continuously updated from time step to time step to dissociate links active with flow from those with no flow. Also B is updated to place the height distribution values at positions corresponding to current point of deposition. In the following subsection, we reformulate this model structure so that parameters stay time independent.

B. Constrained-Flow RNN Model

In this subsection, we constrain the ubiquitous liquid flow in the graph-based model. We impose the condition that the spread of interacting (contacting) droplets is inhibited by surface tension. To capture this behavior, we introduce an activation function to threshold flow across links. Also, the droplet distribution parameter is kept constant by using a kernel convolution. We thus obtain a convolution-recurrent-neural-network-like model where the spatial dynamics of the process is described by a kernel convolution and a soft-threshold activation function; and the temporal evolution can be constructed upon a recurrent neural network.

The height evolution is re-written as

$$h_{k+1} = h_k - D\sigma(FD^T h_k) + \text{vec}(b * U_k). \quad (2)$$

D and F are as described earlier, however the diagonal positive definite matrix F containing the flowability parameters is now constant. Together, $FD^T h_k$ indicates the effective height differences across links. σ is the activation function that thresholds what effective height difference across a link would cause flow. We suggest this threshold is set by surface tension and may be embodied using an activation function. Fig. 3 shows potential activation functions. The soft threshold of Fig. 3b is preferred to the function of Fig. 3a as it is continuous and holds a threshold value explicitly in its formulation. Yet, (as we highlight in the following section), identifying the model parameters by gradient based methods requires the stability of our activation function's gradient

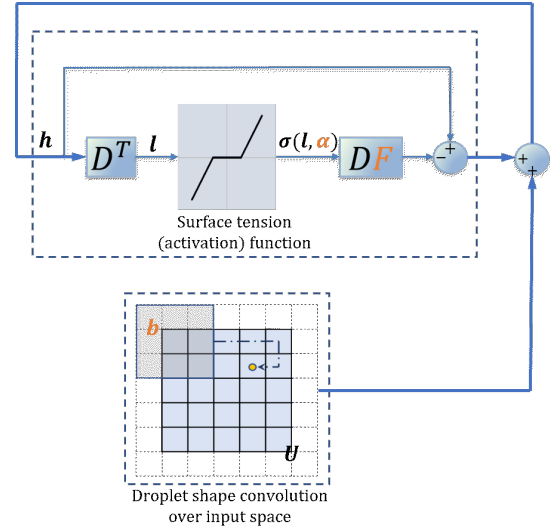


Fig. 4: Height evolution as a recursion. At each timestep, the kernel b is convolved over the current input space which holds entry only at the current deposition spot (yellow spot in the figure), and the result is added to the evolving height distribution

[19], [20]. The function of Fig. 3c, which we term *leaky soft-threshold* allows for gradients stability over long time steps and hence is used in this work. Denoting $FD^T h_k$ by l_k , the leaky soft-threshold function can be written as

$$\sigma(l_k) \triangleq \begin{cases} l_k - (1 - \delta)\alpha & \text{if } l_k > \alpha, \\ \delta l_k & \text{if } -\alpha \leq l_k \leq \alpha, \\ l_k + (1 - \delta)\alpha & \text{if } l_k < -\alpha, \end{cases} \quad (3)$$

where α is the threshold below which height differences cause no flow and $\delta \ll 1$. Instead of the vector B_k used in (1), we convolve the droplet height distribution kernel b over input space U_k . U_k holds only an entry in the position of deposition; hence we implement a sparse convolution, and vectorize the resulting matrix.

Remark 2: Since this model is defined at each time step of the height evolution, the parameters are independent of reference geometry or printing trajectory as well as time. In Fig. 4, we illustrate how the model lends itself to a soft-threshold recursion. As described in following subsection, this formulation is valuable for gradient-based online parameter learning.

C. Parameter Learning for the Constrained-Flow RNN Model

This subsection discusses how the proposed model structure conforms to a RNN. Then, we highlight how the network, and hence, the model parameters may be updated online.

Re-writing (2) as

$$h_{k+1} = \phi(h_k) + \text{vec}(b * U_k), \quad (4)$$

where $\phi(h_k) = h_k - D\sigma(FD^T h_k)$, the recursion in Fig. 4

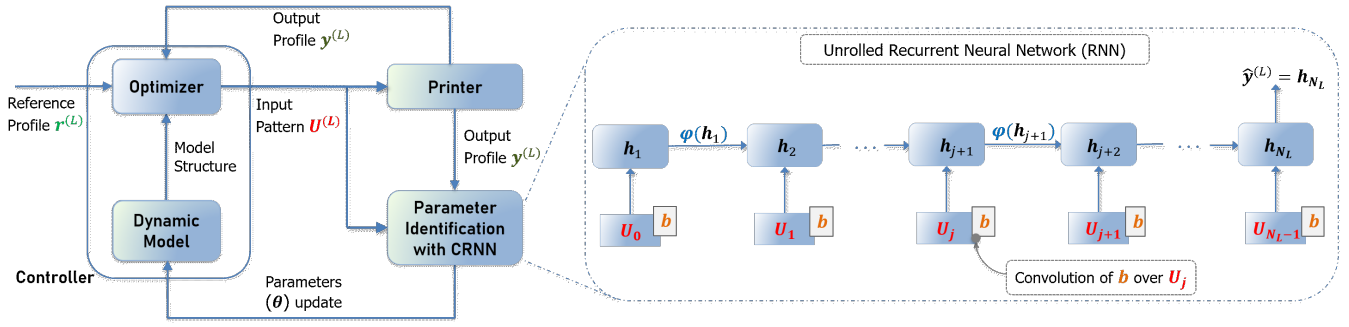


Fig. 5: Strategy for online identification in a closed-loop printing scheme. Note the height evolution unrolled as a Recurrent Neural Network (RNN).

may be unrolled as a RNN with output only at the network's final time step (Fig. 5) since layer measurement becomes available only after the entire layer has been printed. Each measured layer provides additional data for training the RNN. In training, we aim to minimize the 2-norm error with respect to the parameter $\theta \in \{\alpha, b, F\}$ over all measured layers, that is, find

$$\begin{aligned} \theta^* = \underset{\theta}{\operatorname{argmin}} \quad E^{(L)} \triangleq \|y^{(L)} - \hat{y}^{(L)}(\theta)\|_2^2 \quad \forall L, \\ \text{s.t.} \quad \theta_{\min} \leq \theta \leq \theta_{\max} \end{aligned} \quad (5)$$

where $y^{(L)}$ is the vectorized measured height profile of the L^{th} layer, and $\hat{y}^{(L)}(\theta)$ is the corresponding computed height of the RNN's forward pass; $\theta_{\min}$ and $\theta_{\max}$ are respectively the lower and upper bounds for θ . $\{\alpha, b\}$ are constrained to be non-negative. Meanwhile constraint on F is such that $\|F\|_{\max} < 1/\rho(DD^T)$, where $\rho(DD^T)$ is the spectral radius of the Laplacian DD^T . The flow across links may be assumed to be isotropic such that $F = fI$, $f < 1/\rho(DD^T)$. For a given layer L , the desired parameter θ is updated following $\theta^{iter+1} \leftarrow \theta^{iter} - \eta \frac{\partial E}{\partial \theta}$, where η is the descent step size or learning rate, $iter$ is the iteration number and $\frac{\partial E}{\partial \theta}$ is obtained by back-propagation through time. The gradient of the error with respect to the variable $\theta \in \{\alpha, F\}$ is

$$\frac{\partial E}{\partial \theta} = \sum_{1 \leq k \leq N_L} \frac{\partial E}{\partial h_{N_L}} \frac{\partial h_{N_L}}{\partial h_k} \frac{\partial h_k}{\partial \theta} \quad (6)$$

where

$$\begin{aligned} \frac{\partial E}{\partial h_{N_L}} &= 2(y^{(L)} - h_{N_L})^T, \\ \frac{\partial h_{N_L}}{\partial h_k} &= \prod_{N_L-1 > i \geq k} \left(I - D \operatorname{diag}(\sigma'(l_i)) F D^T \right) \end{aligned}$$

and

$$\begin{aligned} \frac{\partial h_k}{\partial F} &= -D \operatorname{diag}(\sigma'(l_{k-1})) D^T h_{k-1}, \\ \frac{\partial h_k}{\partial \alpha} &= -D \operatorname{diag}(\sigma'(l_{k-1})) \sigma'(\alpha), \\ \sigma'(l_k) &= \begin{cases} 1 & \text{if } l_k > \alpha, \\ \delta & \text{if } -\alpha \leq l_k \leq \alpha, \\ 1 & \text{if } l_k < -\alpha, \end{cases} \end{aligned}$$

$$\sigma'(\alpha) = \begin{cases} \delta - 1 & \text{if } l_k > \alpha, \\ 0 & \text{if } -\alpha \leq l_k \leq \alpha, \\ 1 - \delta & \text{if } l_k < -\alpha. \end{cases}$$

We note the effect of the derivative term $\sigma'(l_k)$ on the gradient. Because $\sigma'(l_k) \leq 1$, the gradient is kept from exploding. Also, the network is never 'inactive' [20] as $\sigma'(l_k) > 0 \quad \forall k$. The gradient of the error with respect to the kernel b is

$$\frac{\partial E}{\partial b} = \sum_{1 \leq k \leq N_L} \operatorname{vec}^{-1} \left(\frac{\partial E}{\partial h_{N_L}} \frac{\partial h_{N_L}}{\partial h_k} \right) * \operatorname{rot}_{180}(U_k) \quad (7)$$

where vec^{-1} denotes the vector matricization and rot_{180} denotes a 180 degrees rotation. Given the gradient directions, the optimization may be implemented with a gradient or (approximate) Hessian-based numeric solver [21]. Once θ^* over the measured layers is obtained, the set of parameters is then useful for predicting the height profile of succeeding layers. Hence, this new, or updated parameter may be fed to a model-based controller as illustrated in Fig. 5.

IV. PREDICTIVE CONTROL OF INKJET PRINTING

Instead of using the model presented in (2) for determining the optimal control input, a more convenient form can be used for the optimizer of Fig. 5 by leveraging the same gradients as were used for the model identification. In this section, we re-express the model so that the input is time-invariant and show how the optimal input may be obtained.

A. Constrained-Flow RNN Model Reconstructed for Predictive Control

Since we can express the convolution $b * U_k$ as a matrix multiplication, we now rewrite (2) as

$$h_{k+1} = \phi(h_k) + B_k u^{(L)}, k \in [0, N-1] \quad (8)$$

where $u^{(L)}$ is the input vector for the entire layer; $B_k = B I_k$ where B is the convolution matrix for the learned kernel b , and I_k is a sparse matrix holding only a one at the position corresponding to where a deposition may take place at the time, that is, $I_k u^{(L)} = u_k$ which is the vector for U_k in (2). With this construct, once the model parameters in (2)

are identified, and given a reference profile, we may find a optimal input for layer L , $u^{(L)*}$ by gradient-based means.

B. Predictive Control using the Reconstructed Model

In the Sec III, we had constant parameters whose optimal values were to be determined. We also showed how we could obtain descent direction via back-propagation. Now, we follow a similar pattern to evaluate optimal input. The input vector is independent of time; hence we are to determine the optimal vector for this time-invariant input. Assuming layer L has been printed and we are concerned with output of the next N layers, the optimization problem may be written as

$$\begin{aligned} \mathbf{U}_L^* = \underset{\mathbf{U}_L}{\operatorname{argmin}} J(\mathbf{U}_L) &\triangleq \left\| P(r^{(N|L)} - \hat{y}^{(N|L)}) \right\|_2^2 \\ &+ \sum_{j=1}^{N-1} \left\| Q(r^{(j|L)} - \hat{y}^{(j|L)}) \right\|_2^2 \\ \text{s.t. } h_{k+1} &= \phi(h_k) + B_k u^{(L)}, k \in [0, N_{(j|L)} - 1], \\ \hat{y}^{(j|L)} &= h_{N_{(j|L)}}, \\ u_{\min} &\leq u^{(j|L)} \leq u_{\max}, \forall j \in \{1, \dots, N\}, \end{aligned} \quad (9)$$

where $\mathbf{U}_L^* = [u^{(1|L)} \dots u^{(N|L)}]$, $u^{(j|L)}$ indicates the j^{th} layer control input in the receding horizon; P and Q are positive-definite weighting matrices; and $u_{\min}$ and $u_{\max}$ are the lower and upper bounds to the values in $\mathbf{U}_L^*$. The gradient of the cost function J with respect to the j^{th} layer control input is similar to (7) and is written as

$$\frac{\partial J}{\partial u^{(j|L)}} = \sum_{1 \leq k \leq N_{(j|L)}} \frac{\partial J}{\partial h_{N_{(j|L)}}} \frac{\partial h_{N_{(j|L)}}}{\partial h_k} B_k \quad (10)$$

where

$$\frac{\partial J}{\partial h_{N_{(j|L)}}} = (r^{(j|L)} - \hat{y}^{(j|L)})^T Q, \forall j \in \{1, \dots, N-1\},$$

and

$$\frac{\partial J}{\partial h_{N_{(N|L)}}} = (r^{(N|L)} - \hat{y}^{(N|L)})^T P.$$

Note that since the gradients are expressed analytically rather than by finite difference, the computation to find $\mathbf{U}_L^*$ is expedited.

V. EXPERIMENTAL MODEL VALIDATION

In this section, we first ascertain that the proposed constrained-flow model essentially captures the actual height profile of printed parts by identifying and validating the model parameters on experimental data. (The reader is referred to [16] for the experimental setup). We take measured profile data from a few printed layers of a part, and identify the model based on this data.

We find the identified model parameters to yield profiles that better match the experimental data than the graph-based model parameters does. This is in addition to the fact that the constrained-flow model structure is amenable for update as printing proceed.

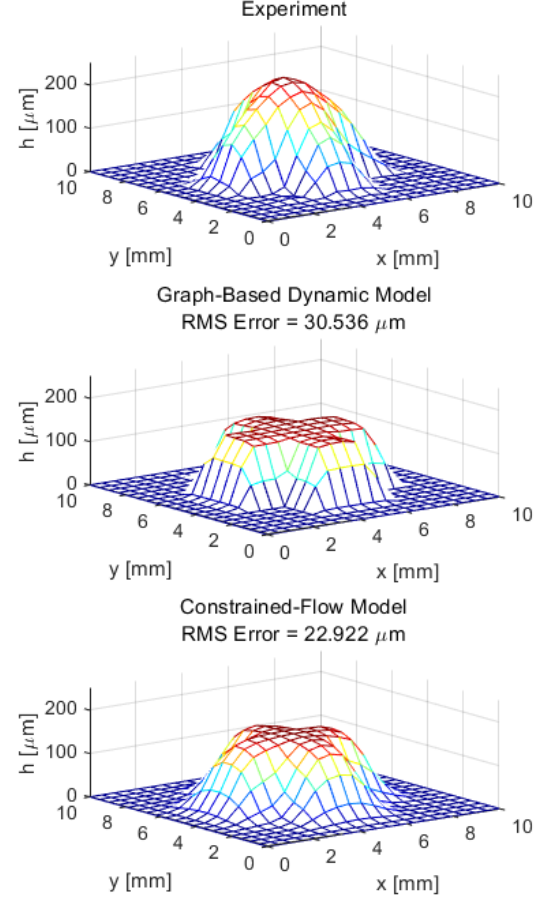


Fig. 6: Comparison of predicted height profiles obtained from models with experiment for a $8 \text{ mm} \times 8 \text{ mm}$ cross-based frustum. 3 layers are printed.

A. Model Parameter Identification

To identify the model parameters, we print a cross-shaped frustum with base dimension $10 \text{ mm} \times 10 \text{ mm}$. Both the input grid resolution and line spacing (or distance between droplets) is 0.125 mm . It is observed that although the droplets are uniformly deposited, the liquid material retreats inwards (Experiment in Fig. 6). We intend to essentially capture this liquid behavior with our model.

The model parameters to be identified $\theta \in \{\alpha, b, F\}$ are obtained by minimizing the 2-norm error of (5). However, we apply ‘standard coarsening’ twice to increase the resolution of both the input and output space by four [22]. In addition to saving computational cost, this helps us learn only the overall liquid material behavior, and ignore details that may only be peculiar to a printing session. The optimization is implemented over 3 layers. Each layer has a smaller surface area than the preceding one. On identification of the parameters, we self-validate the values obtained, and re-mesh the output to yield original resolution.

On self-validation (in-sample), the identified parameters yield a RMS error of 22.9 micrometers on the printed frustum, an improvement in accuracy of 25% over the prediction of the

graph-based dynamic model (Fig. 6). Further, we observe that the identified parameters of the constrained-flow model reflects the elevation of interior observed in the experiment while the graph-based model does not. Note that the graph-based model has indeed identical volume with the printed part, however it fails to capture the inner elevation profile well.

Part Shape	RMS Error(μm)		% Improv.
	Graph-Based	Constrained-Flow	
Cross-shape	30.54	22.92	25.0
T-shape	28.93	23.70	18.0
L-shape	23.59	20.83	11.7

TABLE I: Comparison of RMS error of predicted height profiles obtained from models. The first row displays results from verification of the identified parameters; the next two rows are for validation.

B. Model Validation

Next, we use the identified parameters in the previous subsection to predict the height profiles for other shapes as shown in Table I (for out-of-sample testing). The shapes are printed in the same manner as was the cross-shape part already described. We note, however, that due to the geometry shape, the printing trajectory followed differs from that for the cross-shape. We find the prediction error over the third layer of this build. Accuracy improvements of 18% and 12% over the predictions of the graph-based model are noted.

VI. SIMULATION RESULTS FOR CONTROL IMPLEMENTATION

In this section we implement the closed-loop printing strategy of Fig. 5 in simulation on the cross-shaped part of Sec V-A. To closely conform simulation to the actual print, we fit parameters specifically *for each layer* of the pre-printed cross, that is, the simulation model varies from layer to layer based on identified model parameters from the multi-layer printing experiment. The recurrent network parameters are learned based on accumulated layers hitherto printed (in simulation). In other words, the output obtained for each layer when the layer-specific parameters are used is assumed the ground truth. Our goal is to then find the optimal controlled input that minimizes the reference output error as described in Sec IV-B.

After each layer is simulated, the model parameters are learned, then the controlled input that minimizes the reference-output error as in (9) is computed for one prediction horizon. For comparison, we also simulate the open-loop process based on the layer-specific parameters. The feed-forward input here is based on the superposition model in [18]. The reference, and open-loop and closed-loop outputs for four successive layers are shown in Fig. 7. Again, each layer has a larger surface area than its antecedent. At the first layer, we have no feedback information yet, hence we assume no flow and the kernel b to simply have the spherical shape

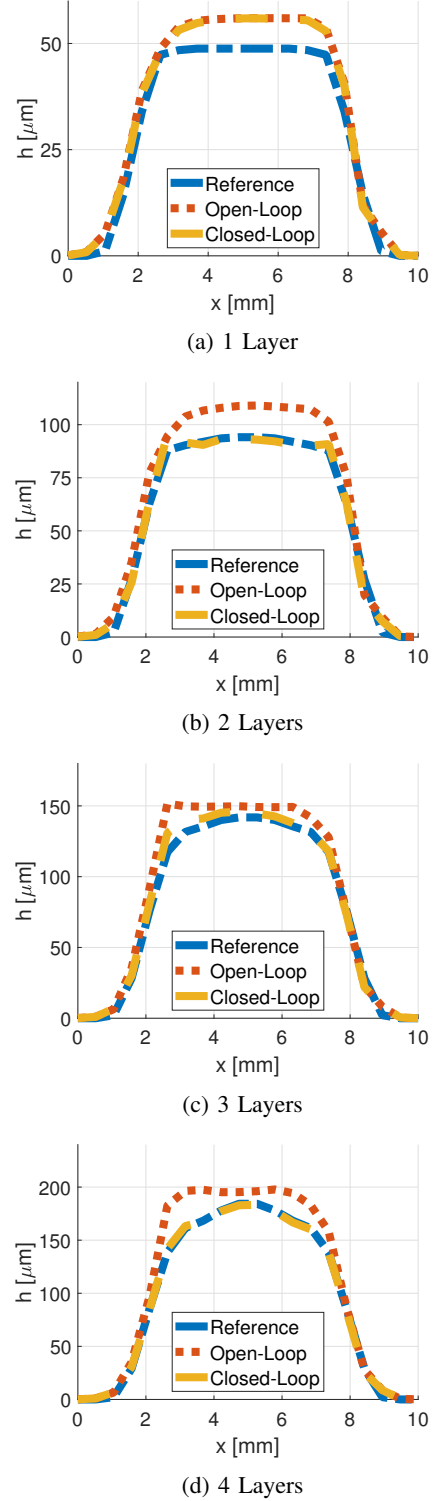


Fig. 7: Comparison between cross-section of open and closed loop profiles after each layer of a cross-shaped part.

of the superposition model. As we begin to obtain output data, the controller performance quickly improves. Though the reference profile for each layer is changing, the controller tracks the profiles closely. At the fourth layer, the RMS error

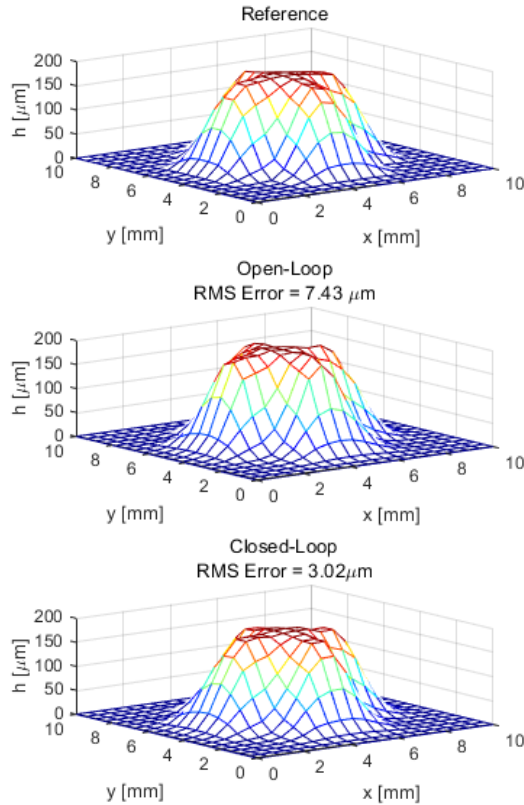


Fig. 8: Comparison of height profiles obtained from closed-loop control with that from open-loop after 4 cross-shaped layers. In this simulation, the controller results in 60% error reduction.

is about $3.0 \mu\text{m}$, a 60% drop in accuracy from the open-loop (Fig. 8). Also, the desired crest in the reference, not captured by the open-loop print, is achieved by the closed-loop control (Fig. 7d).

VII. CONCLUSIONS AND FUTURE WORK

We have developed an adaptive predictive control scheme for inkjet 3D printing. First, we demonstrated that the time-step to time-step evolution of parts' height profile during the inkjet 3D printing may be essentially captured using a dynamic model with a few time-invariant parameters. We also showed that such a model structure fits into a recurrent neural network, and hence makes it amenable for learning during printing. In addition, because the model possesses a geometric and physically interpretable structure, learning may proceed with only small amount of available data. The model has been validated experimentally. Second, we have presented a gradient-based approach to learning a controller for the system in similar fashion as the model identification. The strategy has been demonstrated in simulations with notable results. In future work, we shall experimentally demonstrate the learning process within the adaptive predictive control scheme and evaluated how this in-process model learning effectively enhances control.

ACKNOWLEDGMENT

This work was supported in part by the NSF Data-Driven Cyberphysical Systems Award #1645648 and by the State of New York ESD/NYSTAR program.

REFERENCES

- [1] S. F. S. Shirazi, S. Gharehkhani, M. Mehrali, H. Yarmand, H. S. C. Metselaar, N. Adib Kadri, and N. A. A. Osman, "A review on powder-based additive manufacturing for tissue engineering: Selective laser sintering and inkjet 3D printing," *Science and Technology of Advanced Materials*, vol. 16, no. 3, 2015.
- [2] M. Singh, H. M. Haverinen, P. Dhagat, and G. E. Jabbour, "Inkjet printing process and its applications," *Advanced Materials*, vol. 22, no. 6, pp. 673–685, 2010.
- [3] Y. Guo, H. S. Patanwala, B. Bognet, and A. W. Ma, "Inkjet and inkjet-based 3D printing: Connecting fluid properties and printing performance," *Rapid Prototyping Journal*, vol. 23, no. 3, pp. 562–576, 2017.
- [4] H. Wijshoff, "The dynamics of the piezo inkjet printhead operation," *Physics Reports*, vol. 491, no. 4-5, pp. 77–177, 2010.
- [5] H. Y. Gan, X. Shan, T. Eriksson, B. K. Lok, and Y. C. Lam, "Reduction of droplet volume by controlling actuating waveforms in inkjet printing for micro-pattern formation," *Journal of Micromechanics and Microengineering*, vol. 19, no. 5, p. 055010, 2009.
- [6] S. Rahmati, S. F. Shirazi, and H. Baghayeri, "Piezo-electric head application in a new 3D printing design," *Rapid Prototyping Journal*, vol. 15, no. 3, pp. 187–191, 2009.
- [7] Z. Du, R. Xing, X. Cao, X. Yu, and Y. Han, "Symmetric and uniform coalescence of ink-jetting printed polyfluorene ink drops by controlling the droplet spacing distance and ink surface tension/viscosity ratio," *Polymer (United Kingdom)*, vol. 115, pp. 45–51, 2017.
- [8] L. Luo, X. P. Wang, and X. C. Cai, "An efficient finite element method for simulation of droplet spreading on a topologically rough surface," *Journal of Computational Physics*, vol. 349, pp. 233–252, 2017.
- [9] C. Doumanidis and E. Skordeli, "Distributed-Parameter Modeling for Geometry Control of Manufacturing Processes With Material Deposition," *Journal of Dynamic Systems, Measurement, and Control*, vol. 122, no. 1, pp. 71–77, 07 1998.
- [10] A. B. Thompson, C. R. Tipton, A. Juel, A. L. Hazel, and M. Dowling, "Sequential deposition of overlapping droplets to form a liquid line," *Journal of Fluid Mechanics*, vol. 761, pp. 261–281, 2014.
- [11] W. Zhou, "Interface dynamics in inkjet deposition," Ph.D. dissertation, Georgia Institute of Technology, 2014.
- [12] D. J. Hoelzle and K. L. Barton, "A new spatial iterative learning control approach for improved micro-additive manufacturing," in *2014 American Control Conference*, June 2014, pp. 1805–1810.
- [13] Y. Guo and S. Mishra, "A predictive control algorithm for layer-to-layer ink-jet 3d printing," in *2016 American Control Conference (ACC)*, 2016, pp. 833–838.
- [14] D. L. Cohen and H. Lipson, "Geometric feedback control of discrete-deposition SFF systems," *Rapid Prototyping Journal*, vol. 16, no. 5, pp. 377–393, 2010.
- [15] C. Pannier, M. Wu, D. Hoelzle, and K. Barton, "Lpv models for jet-printed heightmap control," in *2019 American Control Conference (ACC)*, July 2019, pp. 5402–5407.
- [16] L. Lu, J. Zheng, and S. Mishra, "A layer-to-layer model and feedback control of ink-jet 3-d printing," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 3, 2015.
- [17] Y. Guo, J. Peters, T. Oomen, and S. Mishra, "Distributed model predictive control for ink-jet 3d printing," in *2017 IEEE Conference on Advanced Intelligent Mechatronics (AIM)*, July 2017, pp. 436–441.
- [18] Y. Guo, J. Peters, T. Oomen, and S. Mishra, "Control-oriented models for ink-jet 3d printing," *Mechatronics*, vol. 56, 05 2018.
- [19] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.
- [20] Z. Allen-Zhu, Y. Li, and Z. Song, "On the convergence rate of training recurrent neural networks," in *Advances in Neural Information Processing Systems* 32, 2019, pp. 6676–6688.
- [21] M. S. Bazaraa, *Nonlinear Programming: Theory and Algorithms*, 3rd ed. Wiley Publishing, 2013.
- [22] W. Joppich and S. Mijalkovic, *Multigrid methods for process simulation*, ser. Computational microelectronics. Springer-Verlag, 1993.