Autonomous Precision Pouring From Unknown Containers

Monroe Kennedy, Karl Schmeckpeper, Dinesh Thakur, Chenfanfu Jiang, Vijay Kumar, and Kostas Daniilidis

***Abstract*—We autonomously pour from unknown symmetric containers found in a typical wet laboratory for the development of a robot-assisted, rapid experiment preparation system. The robot estimates the pouring container symmetric geometry, then leverages simulated pours as priors for a given fluid to pour precisely and quickly in a single attempt. The fluid is detected in the transparent receiving container by combining weight and vision. The change of volume in the receiver is a function of the geometry of the pouring container, the pouring angle, and rate. To determine the volumetric flow rate, the profile for maximum containable volume for a given angle is estimated along with the time delay of the fluid exiting the container. A trapezoidal trajectory generation algorithm prescribes the desired volumetric flow rate as a function of the estimation accuracy. A hybrid control strategy is then used to attenuate volumetric error. Three methods are compared for estimating the volume-angle profile, and it is shown that a combination of online system identification and leveraged model priors results in reliable performance. The major contributions of this work are a system capable of pouring quickly and precisely from varying symmetric containers in a single attempt with limited priors, and a novel fluid detection method. This system is implemented on the Rethink Robotics Sawyer and KUKA LBR iiwa manipulators.**

***Index Terms*—Model learning for control, motion control, manipulation planning, service robots.**

I. Introduction

As THE capabilities of autonomous robots increase, their ability to assist humans in complex environments must also increase. This work is motivated by the pharmaceutical wet-lab industry, where research scientists perform repetitive experiments with relatively small amounts of solution and active ingredients. In this scenario, it is inefficient for scientists to use large batch solution making machines, however considerable time is spent making solutions for experiments. The necessity is for an autonomous robot that is capable of making such small batch solutions while requiring very little environment augmentation as it works alongside the research scientist collaborator. To be effective, the robot must be able to manipulate containers already in use by the wet-lab as well as pour precisely compared to a human counterpart.

The precision pouring problem can be decomposed into the observation of the poured fluid, modeling of the flow dynamics and controlling the pouring container to reach a target volume. Previous work has investigated methods of detecting water in flight [1], [2] as well as fluid in a cup when viewed from above classified into 11 fill percentages [3]. The proposed method measures the volume of fluid in the receiver by combining both mass from scale and vision by detecting water pixels in a transparent receiver.

Once the fluid is detected, the flow dynamics between the containers must be modeled. In [4] and [5] system identification on model parameters is performed to improve the performance of derived models. Motion primitives for the pouring task were learned in [6], [7] and [8]. In [9] and [10] transformations of points clouds from example containers were morphed to observed containers and a corresponding transformation was applied to the task space trajectory for pouring. In [11] simulated pours are used to learn to mitigate spillage. Our approach combines online system identification with model priors leveraging the pouring container geometry, as well as focuses on precise fluid transference assuming no spillage as opposed to just emptying contents.

Once the pouring model has been identified, the system must be controlled to pour the specified volume. In [4], [12], [13] the angular rate of the pouring container is controlled based on the known pouring model. Our approach utilizes a hybrid control strategy that incorporates the process model and estimated time delay in the plant. For this application, our approach improves on [12] with an average pour error and time of 38ml and 20 seconds with an accuracy within 10 ml and average of 3 ml with pour times varying from 20–45 seconds. Our approach improves on [4] and [13] in that we pour precisely in a single attempt and are not confined to a particular geometry given the pouring container is symmetric.

Our letter advances the state of the art in autonomous pouring with an effective wet-lab solution preparation system capable of *a)* Leveraging simulated containers and pours to

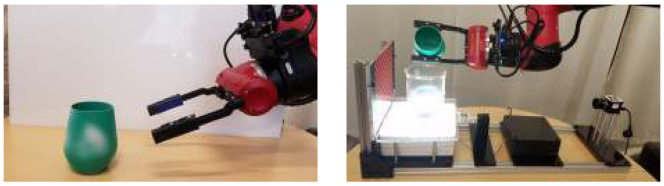
Manuscript received November 27, 2018; accepted February 13, 2019. Date of publication February 27, 2019; date of current version March 15, 2019. This letter was recommended for publication by Associate Editor M. C. Yip and Editor P. Rocco upon evaluation of the reviewers' comments. This work was supported in part by GlaxoSmithKline under Grants NSF-IIP-1439681 (I/UCRC) and NSF-IIS-1426840 and in part by the ARL W911NF-10-2-0016 Robotics Collaborative Technology Alliance. The work of M. Kennedy was supported by the NSF Graduate Fellowship DGE-1845298. *(Corresponding author: Monroe Kennedy.)*

M. Kennedy, K. Schmeckpeper, D. Thakur, V. Kumar, and K. Daniilidis are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: kmonroe@seas.upenn.edu; karls@seas.upenn.edu; tdinesh@seas.upenn.edu; kumar@seas.upenn.edu; kostas@seas.upenn.edu).

C. Jiang is with the SIG Center for Computer Graphics, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: cffjiang@seas.upenn.edu).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors. The Supplemental Materials contain a video showing autonomous precision pouring from unknown containers. This material is 20.9 MB in size.

Digital Object Identifier 10.1109/LRA.2019.2902075



(a) (b)

Fig. 1. The objective is to pour precisely from an unknown container. We model the pouring rate (which depends on the profile of maximum volume of fluid containable at a given tilt angle), by estimating the container geometry (Figure 1a) and then using model priors and online system identification to identify the profile and pour precisely in a single attempt (Figure 1b).

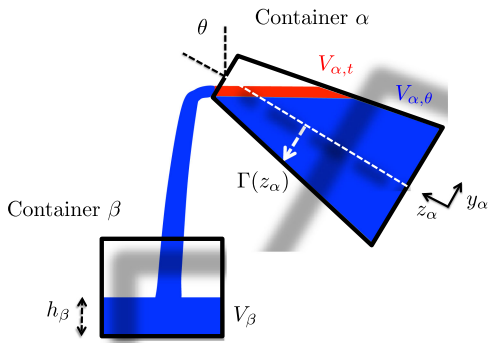


Fig. 2. The goal is to pour a specified amount of a known fluid from container α to container β quickly and precisely. The receiver volume V_β is sensed (through weight and height h_β) and the geometry of the pourer Γ is estimated. $V_{\alpha,t}$ and $V_{\alpha,\theta}$ are the transient and steady state volumes in α for a given tilt angle θ.

obtain pouring dynamics and expected plant time delay for a new target container. *b)* Pouring target volumes quickly and precisely leveraging system identification and model priors. *c)* Combining vision and mass fluid detection methods to obtain the received volume. The rest of the letter is organized as follows: Section II-A presents the problem formulation. Section II-B presents the model estimation techniques. Section II-C discusses the real time volume estimation technique. Section II-D presents trajectory generation and control. Section III discusses implementation details. Finally, results for each method and performance across container geometries is discussed in Section IV.

II. Methodology

A. Problem Formulation

Given a pouring container α, and receiving container β, the goal of this work is to quickly and precisely pour a designated amount of fluid. The general pouring problem is presented in Figure 2 where both containers are open, and during pouring only the volume in the receiver V_β is detected. For symmetric containers Γ specifies the radius of the container as a function of height which is observed before pouring. In container α, the volume of fluid above and below the pouring edge are denoted as $V_{\alpha,t}$ and $V_{\alpha,\theta}$ respectively and are the transient and steady state volumes when α is held at tilt angle θ. We call the function $V_{\alpha,\theta}(\theta)$ the volume angle profile. The height of fluid in the receiver is denoted as h_β.

The generalized pouring problem is to consider the volumetric flow rate between containers α and β and the dynamics of the system are

$$\begin{aligned}
\dot{V}_\beta(t+t_{fall}) &= -\dot{V}_\alpha(t) \\
\dot{V}_\beta(t+t_{fall}) &= -\left(\left.\frac{dV_{\alpha,\theta}}{d\theta}\right|_{\theta(t)} + \left.\frac{dV_{\alpha,t}}{d\theta}\right|_{\theta(t)}\right)\frac{d\theta}{dt} \\
\dot{V}_\beta(t+t_d) &= -\underbrace{\left.\frac{dV_{\alpha,\theta}}{d\theta}\right|_{\theta(t)}}_{V^{(1)}_{\alpha,\theta}}\frac{d\theta}{dt},
\end{aligned} \tag{1}$$

where t_{fall} is the fall time of the fluid from α to β, t_d is the time delay for both the fall time and dissipation of $V_{\alpha,t}$, θ and $\frac{d\theta}{dt}$ are the angular position and velocity of the pouring container. $V^{(1)}_{\alpha,\theta}$ denotes the derivative of $V_{\alpha,\theta}$ with respect to θ. Since we will only observe the receiving container, the volume angle profile must be expressed in terms of container β. At steady state the following holds

$$V^{(1)}_{\alpha,\theta} = -V^{(1)}_{\beta,\theta}, \tag{2}$$

and for bounded velocity pours we will assert this equivalence in (1). Defining the volumetric error as

$$e_V(t) = V_{\beta,des}(t) - V_\beta(t), \tag{3}$$

the goal is to determine the control input $u(t) = \frac{d\theta}{dt}$ that achieves the desired volume

$$\min_{u(t)} \int_t e_V(t)^2 dt. \tag{4}$$

The desired trajectory $V_{\beta,des}(t)$ is specified using a trapezoidal trajectory generation algorithm to minimize pouring time, subject to velocity and acceleration constraints and is discussed further in Section II-D1. The generalized time delay is an unknown function of the pouring container geometry, the tilt angle and control input

$$t_d(t) = f_t\left(\Gamma, \theta(t), u(t-t_u)\right). \tag{5}$$

We define t_u to be the period of the controller, as the current time delay is approximated with the last commanded velocity. The time delay approximates the dissipation of $V_{\alpha,t}$ (which is a function of Γ, θ and u) and t_{fall}. We make the following assumptions *a)* That the pours are slow enough that we can make the substitution of (2) into (1). *b)* There is no spillage during pouring and sloshing is insignificant for a sigmoid desired trajectory and the specified maximum angular rotation rate. *c)* The viscosity of the fluid is known, and at steady state the fluid conforms to the geometry of the container. *d)* The geometry of the receiving container is known. *e)* There is enough fluid in the pouring container required to reach the specified target volume in the receiver. *f)* The fall time can be approximated by a small constant. *g)* That similarity in container geometry Γ correlates to similarity in volume profile $V_{\alpha,\theta}$. *h)* The pouring container has a symmetric edge profile for simplistic implementation in container scanning. But the method extends to any container geometry.

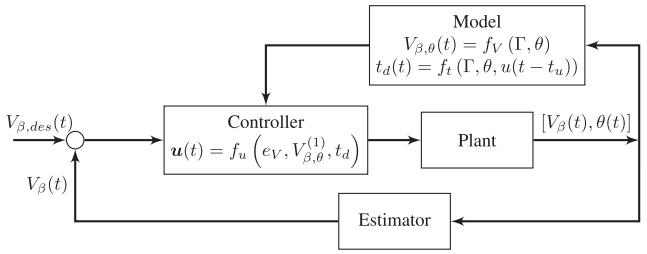


Fig. 3. Pouring Control Diagram. The desired volume is specified by trajectory generator, the controller f_u processes this and the model approximations (f_V and f_t which must be determined) in order to control the angular rate of the pouring container.

The system control diagram is shown in Figure 3. The models for $V_{\beta,\theta}$ and t_d are informed by scanning the pouring container geometry Γ.

B. Model Learning

We define the error (or distance) between two symmetric container geometries Γ_1, Γ_2 as

$$e_\Gamma = \int_0^1 \left(\Gamma_1(H_1 s) - \Gamma_2(H_2 s)\right)^2 ds \tag{6}$$

where H_1 and H_2 are the respective heights of the containers and $\Gamma(Hs)$ defines the radius for $s \in [0,1]$. Our assumption states that

$$\lim_{e_\Gamma \to 0} V_{\beta,\theta,1} = V_{\beta,\theta,2}, \tag{7}$$

and is extended to the time delay in (5) for a given θ, u. The training set of pouring container geometries are artificially generated and used to simulate pouring with known fluid properties as shown in Figure 5. The geometry of the pouring container is scanned before pouring, and the top ten nearest neighbors are then selected based on minimal distance e_Γ. This is then used to generate a mixture probability p_j by

$$p_j = \frac{e_{\Gamma,j}^{-1}}{\sum_i e_{\Gamma,i}^{-1}}. \tag{8}$$

If the new container profile can be interpolated from the training set, then this mixture model will effectively describe the new container profile with adequate resolution in the space of example containers. However in practice this is a strong assertion which is relaxed through adjusting the kernel variance in addition to the mixture probability in semi-parametric model approximation.

1) Combined Parametric and Non-Parametric Approximations: We model the maximum volume profile $V_{\beta,\theta}$ using three methods: parametric, non-parametric and semi-parametric. The parametric method approximates $V_{\beta,\theta}$ using a polynomial of 9th degree

$$V_{\beta,\theta}(\theta) = f_V(\theta) = \sum_{i=0}^{N} c_i \theta^i \tag{9}$$

whose coefficients $\boldsymbol{c}$ minimize the following functional

$$J_\theta = \underbrace{\sum_{j=1}^{C}\left(V_{\beta,\theta,j} - \sum_{i=0}^{N} c_i \theta_j^i\right)^2}_{\text{Residual}} + \underbrace{k_1 \sum_{i=1}^{N} c_i^2}_{\text{Regulator}} + \underbrace{k_2 \exp\left(-\sum_{m=1}^{M}\sum_{i=1}^{N} i c_i \theta_m^{i-1}\right)}_{\text{Soft Constraint}}. \tag{10}$$

The residual term fits the polynomial to C volume observations in the receiver by minimizing the squared error, the regulator term ensures the N coefficients do not diverge with positive gain k_1, and the soft constraint ensures the polynomial derivative is positive at M control points (locations enforcing constraint) evenly spaced over the entire pouring angle domain which respects the physics that volume in the receiver β is strictly increasing with positive gain k_2.

The non-parametric method uses a Gaussian process (GP) with the radial basis kernel function with white noise:

$$\kappa(\theta_i, \theta_j) = \sigma^2 \exp\left(-\frac{(\theta_i - \theta_j)^2}{l^2}\right) + w. \tag{11}$$

The variance is related to the edge profile error through $\sigma_j = k_3 e_{\Gamma,j}^{-1}$ with positive gain k_3, and l is a distance scale factor. The white noise w in the kernel is associated with the volume milliliter measurement error. The covariance matrices for test-train and train-train are defined as

$$K_{i,j}^* = \kappa(\theta_{test,i}, \theta_{trn,j}) \tag{12}$$

$$K_{i,j} = \kappa(\theta_{trn,i}, \theta_{trn,j}) \tag{13}$$

and the information matrix as

$$L = (K + \gamma^2 I)^{-1}. \tag{14}$$

With these terms we define the non-parametric estimate

$$V_{\beta,\theta}(\theta) = f_V(\Gamma,\theta) = \sum_j p_j K_j^* L_j \boldsymbol{V_{\beta,\theta,trn,j}}. \tag{15}$$

The limitation of the parametric method is that it does not leverage prior knowledge of similar containers when available. Likewise, the limitation of the non-parametric method is that it cannot adapt when the new container is drastically different from the training set. By combining these methods, their positive attributes can be leveraged for better performance across a larger range of containers. This is done by making the parametric profile estimation the mean of the GP:

$$V_{\beta,\theta}(\theta) = \sum_{i=0}^{N} c_i \theta^i + \sum_{j=1}^{10} p_j K_j^* L_j \left(\boldsymbol{V_{\beta,\theta,trn,j}} - \sum_{i=0}^{N} c_i \boldsymbol{\theta_{trn,j}^i}\right). \tag{16}$$

The additional component is the last term which uses the current parametric model to evaluate the training pours. The error of the parametric function and true training volume is used to adjust the expected value for $V_{\beta,\theta}$. If the parametric function is a very good approximation, the error between the points in

the vector $\boldsymbol{V_{\beta,\theta,trn}}$ and the function evaluation becomes zero and the parametric mean dominates. If the parametric mean is a poor fit, but the container is interpolated well between example containers, then the GP terms accommodates this error with sufficient sampling of the points in the vector $\boldsymbol{\theta_{trn}}$.

C. Real-Time Volume Estimation

We use a combination of visual feedback and weight measurement to track the volume of the fluid in the receiving container. For visual volume detection, the receiver is first located the container in the scene via a fiducial. Once the container is localized, we use a neural network to find the probability that each pixel is water. As in [13], clustering is used to distinguish between fluid entering the receiver and contained rising fluid. With the known cross section this provides an estimate of the volume of water in the receiving container to be considered with the measured mass. A load cell is used to obtain the weight of fluid in the receiver and the volume estimate from both vision and weight are combined using a Kalman filter whose details are in Section III-A1.

D. Trajectory Planning and Control

1) Trajectory Generation: Trajectories for volume in the receiving container $V_{\beta,des}(t)$ are determined using the user specified target volume, and specified upper and lower maximum velocities dependent on the current residual of the $V_{\beta,\theta}$ approximation. The area under the trapezoid is the target volume. Respecting maximum allowable accelerations, time optimal trajectories are computed in a similar implementation to [14]. A key difference is the calculation of the maximum velocity is a sigmoid function of the mean residual:

$$\bar{r} = \frac{1}{C}\sum_{j}^{C} \left| V_{\beta,\theta,pred,j} - V_{\beta,\theta,meas,j} \right|, \tag{17}$$

where this is evaluated for every new accumulated measurement set C, and new model $V_{\beta,\theta}$ which adjusts $V_{\beta,\theta,pred,j}$. Given this residual, the maximum velocity is calculated using the following function

$$\dot{V}_{\beta,\max}(\bar{r}) = \dot{V}_{\beta,ml} + \frac{\dot{V}_{\beta,mu}}{1 + \frac{\dot{V}_{\beta,ml}}{(\dot{V}_{\beta,mu} - \dot{V}_{\beta,ml})}\left(\exp\left(k_4 \frac{\bar{r}}{r_{\max}}\right)\right)}, \tag{18}$$

where $\dot{V}_{\beta,ml}, \dot{V}_{\beta,mu}$ are the lower and upper bounds on allowable maximum velocities and $\bar{r} \in [0, \infty)$ is the residual for fitting $V_{\beta,\theta}$. The term $r_{\max}$ is a threshold residual ensuring for large residual that $\dot{V}_{\beta,\max} \simeq \dot{V}_{\beta,ml}$.

2) Proposed Controller: Given the trajectory generator specifies $V_{\beta,des}(t)$, the controller then uses the volume error e_V along with model estimates of the volume profile and time delay $V_{\beta,\theta}$, t_d, to calculate the control output which is the angular velocity of the container.

The hybrid controller is dependent on the following conditions (a): $\theta \in [0, \pi]$, (b): $V^{(1)}_{\beta,\theta}(\theta) > 0$, (c): $\dot{V}_\beta > 0$, and (d) $e_V > 0$

$$u(t) = \begin{cases} \left(V^{(1)}_{\beta,\theta}(\theta(t))\right)^{-1}\left(K_p e_V(t + t_d)\right) & \text{if: } (a) \wedge (b) \wedge (c) \\ \delta_\omega & \text{if: } (a) \wedge (d) \wedge \\ & \neg\left((b) \wedge (c)\right) \\ 0 & \text{if: } \neg(a). \end{cases} \tag{19}$$

The third state stops motion if the angle is outside the acceptable regions of operation. The second state (re)initiates the pour. Hence when the container is in the operation domain from condition (a) the first state is obtained.

3) Controller Stability: We present a stability analysis for the first hybrid state, as the second state always results in the first state unless there is not enough fluid in the container to pour the target volume which violates a base assumption. Given the current time T, the future error at $T + t_d$ for time delay t_d is

$$\begin{aligned} e_V(T + t_d) &= V_{\beta,des}(T + t_d) - V_\beta(T + t_d) \\ &= V_{\beta,des}(T + t_d) - \int_0^T V^{(1)}_{\beta,\theta}(\theta(s))u(s)ds. \end{aligned} \tag{20}$$

Consider the Lyapunov function $\mathcal{V} = \frac{1}{2}e_V^2$, the system is asymptotically stable if $\dot{\mathcal{V}} < 0$:

$$\dot{\mathcal{V}} = e_V \dot{e}_V = e_V\left(\dot{V}_{\beta,des}(T + t_d) - V^{(1)}_{\beta,\theta}(\theta(T))u(T)\right). \tag{21}$$

Let

$$u(T) = \left(V^{(1)}_{\beta,\theta}(\theta(T))\right)^{-1} K_p e_V(T + t_d), \tag{22}$$

then if $\dot{V}_{\beta,des}(T + t_d) = 0$, then (21) reduces to

$$-K_p e_V^2 < 0, \tag{23}$$

which is true if $K_p > 0$ and the system is asymptotically stable. If $\dot{V}_{\beta,des}(T + t_d) > 0$ then (21) stability condition becomes

$$e_V(T + t_d) > K_p^{-1}\dot{V}_{\beta,des}(T + t_d). \tag{24}$$

Hence the system will trail until the condition of (24) is true, then when $\dot{V}_{\beta,des} = 0$ the error will attenuate to zero. Note that a larger K_p will reduce the magnitude of e_V required for asymptotic stability in (24).

III. Implementation

A. Volume Measurement

1) Volume Detection: The receiving container is placed on an illuminated stand. A 1280×1040 pixel Point Grey RGB camera is mounted horizontally facing the container, and a checkerboard background shown in Figure 1b is used to leverage distortion and occlusion for fluid detection. We use the network architecture from Holistically-Nested Edge Detection, a network that extracts multi-scale features from VGGNet and uses them for pixel-wise edge detection [15]. The trained network detects pixel masks that show the locations of water (instead of detecting edges as in [15]) and runs at 21Hz on a cropped

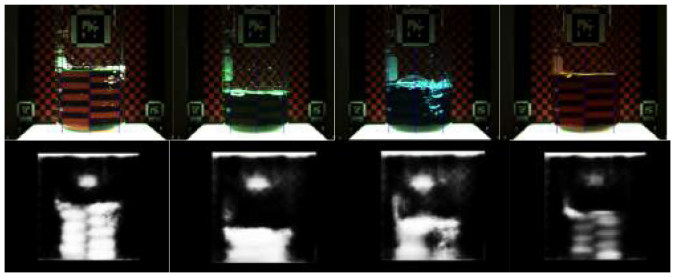


Fig. 4. Volume in the receiver is detected combining vision and weight. The volume is visually estimated by using a fiducial to locate the receiver of known geometry then a network detects fluid pixels. The detection is robust to fluid color and transparency.

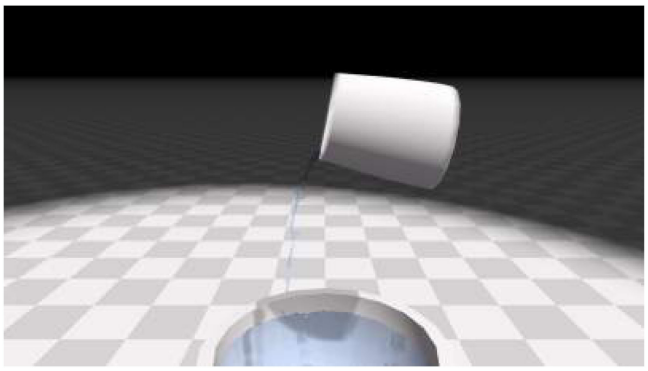
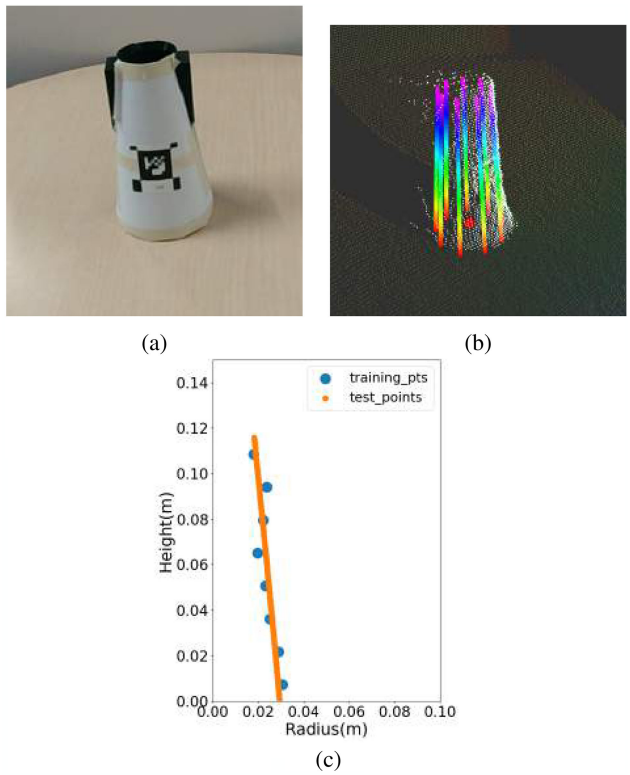


Fig. 5. Artificial containers are generated to provide a simulation of pouring liquid of known properties in NVIDIA FleX [16].



(c)

Fig. 6. Container geometry is scanned, where we assume a surface of revolution and represent the geometry with the edge profile Figure 6c.

390 × 412 pixel image based on the fiducial location. Our visual system is able to calculate the height of the water h_β for many different colors of water, ranging from clear to completely opaque as shown in Figure 4.

The beaker is placed 42 ± 5 cm from the camera, the camera is 6 cm above the platform, and the receiver diameter is known (in this experiment 3.64 cm) as shown in (Figure 1b). This makes the top of the fluid visible if the volume of fluid is below 250 ml, with a maximal error of 1.04 cm translating to 43 ml at the onset of pouring. To mitigate this effect, similar triangles are used to determine the true height of the container as the fiducial provides both the beakers distance from the camera, and pixel-metric scale factor in the fiducial plane. This geometric adjustment is utilized when the height of fluid is below the center pixel of the camera (which for this beaker correlates to below 250 ml). We also use a Uxcell 5 kg load cell with an Arduino Uno micro-controller to detect the weight of the fluid in the receiving container which runs at 12 Hz. The receiver container sits on a suspended platform shown in Figure 1b which is a cantilever with the load cell.

2) Sensor Fusion: Both the volume estimate from vision and scale are combined using a Kalman filter running at 30 Hz to approximate the volume in the receiving container. The associated uncertainties for the scale and vision are 1.5 ml and 15 ml respectively, the process noise was set to be 0.02 ml. For the vision, this is due to an 20 pixel variance in detection of fluid height.

B. Simulated Pouring

To obtain simulated pours we use NVIDIA FleX, a particle based unified graphics solver from [16], to simulate pouring liquids from different containers. We generated 1792 symmetric containers randomly scaled between 5 and 20 cm. Each container is poured once in simulation (Figure 5) and the $V_{\beta,\theta}$ profile is generated by filling the container to the brim with the liquid particles and slowly rotating the container. At each time step, the quantity of liquid inside the container is measured by counting the number of particles that were inside the container's mesh. Parameters in the FleX software were empirically chosen to match behavior for real container geometries for water at room temperature.

C. Volume Estimation and Attenuation

We establish a reference frame at the bottom, center of the pouring container which is assumed to be a surface of revolution (SOR). Given the 128 points along the edge of the container, the edge-profile is defined by the set of vectors consisting of each point height and radius. This edge-profile is then compared with the edge profiles of simulated containers, and the closest top 10 containers are selected. For trajectory generation, we set the parameter $k_4 = 8$ in (18).

D. Volume Profile and Time Delay Estimation

1) Container Edge Extraction: We obtain the geometry of the pouring container to compare to simulated container

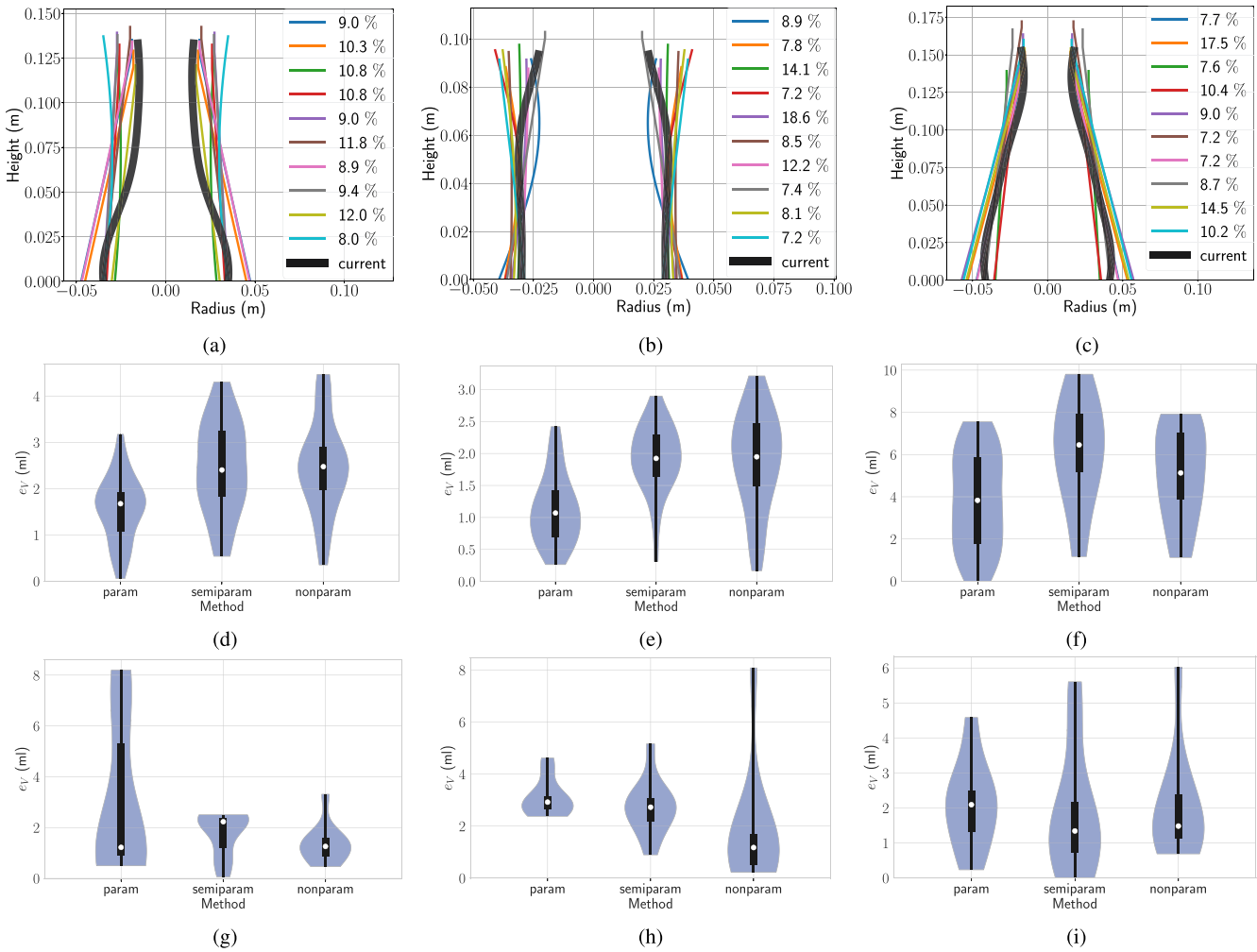


Fig. 7. Container profiles and pour statistics: containers 1, 2 and 3 are 7a, 7b and 7c respectively, with their statistics directly below each container. Percentages are the mixture probabilities defined in (8). Final volume error statistics are shown for a 100 ml target volume in the second row for each method (30 pours each), and for 200 ml target volume in the third row (10 pours each). Each method statistics are shown using violin plots for distribution and inset box plot with mean and quartiles. A total of 360 pours are shown.

geometries to approximate the volume profile. For symmetric containers it is sufficient to scan half of the container to extrapolate the entire geometry. We use the PMD Technologies Pico Flexx time-of-flight depth sensor to extract the point cloud of the container. RANSAC is used to distinguish between the container and table points, then to approximate the container with a set of cylinders every 1.5 cm. The edge profile is then extracted using a Gaussian process. This is shown in Figure 6, where Figure 6a shows the container, Figure 6b shows the extracted points and the fitted surface, and Figure 6c shows the fitted cylinders radii and height and the resulting 128 points from the GP fitting. During data collection, the surface of the containers were augmented with form fitting paper for better performance of the time of flight sensor.

2) Time Delay: We modeled the time delay in simulation by rotating fully filled containers at randomly chosen, constant angular velocities until they reached randomly chosen stop angles. The time delay was defined as time to reach 20% of the initial $V_{\alpha,t}$ from the stopping point. We simulated 3,888 trials for the time delay. A neural network consisting of three convolutional layers for the edge profile and then five fully connected layers combining the convolutional output and the containers height, tilt angle and angular velocity is used to predict the time delay by approximating f_t in (5).

E. Pouring System

The full state machine was implemented on the KUKA LBR iiwa manipulator for the KUKA innovation award and can be found on YouTube: “Finalist Spotlight - Precise Robotic Dispenser System - KUKA Innovation Award 2018”, and “Kuka Innovation Award finalist: Precise Robotic Dispenser System”. In these experiments for brevity we focus on just the container edge-profile extraction and precision pouring. A video demonstrating this method can be found on at https://www.youtube.com/watch?v=C1UTOSfGuXA&feature=youtu.be “Autonomous Precision Pouring from Unknown Containers”.

IV. Results and Discussion

A. Volume Profile Estimation Method Comparison

Three representative containers are shown in Figure 7, with the container classifications along with their final volume error performance for each maximum volume profile estimation method. The statistical data is represented using violin plots showing the data distribution along with box plots consisting of data minimum and maximum as thin black line, first and third quartile as solid black line, and the median as white dot. Figure 7a shows container 1 which is a small flask, in Figure 7b container 2 is a small cylinder, in Figure 7c container 3 is a tall flask. For container 1, Figure 7a shows the training container along with their mixture probabilities. The second row of Figure 7 shows the final volume error for each method while pouring 100 ml of fluid. For container 1 and container 2 in Figures 7d and 7e each method statistic was generated using 30 trials for each method, statistics for container 3 where generated from 30 trials of param, and 10 trials for semi and non-param. For 200 ml pours each method for all three containers were generated from 10 trials each. In Figure 7 the general trend is that for 100 ml pours the parameterized method usually performed better than semi and non-parameterized methods, with these two methods being comparable in performance. However, for the larger volume 200 ml pour the semi and non-parameterized methods were more consistent compared to the 100 ml pour and performed better than the parameterized pour. All of the containers consistently started with 250 ml of fluid before each pour. Another notable aspect is that better performing methods usually had a larger pour time than the lower performing counter parts for the same container and target volume. A negative example is shown in Figure 8 where the priors are used from container 1 in Figure 7a. Here Figure 8b shows each method performance with 30 trials each, and Figure 8c shows each method with 10 trials each. Note that while the trend for 100 ml is similar to that in Figure 7, the spread of the semi and non-parametric methods is much greater. This is an important example as it demonstrates the capacity of the proposed system when a container is far from the training set of containers. In both Figures 8b and 8c, the contribution of the parametric method influences the semi-parametric performance making it comparable to the parametric performance. The parametric method performance varies with the container as it is performing online system identification, this is evident in how parametric performed worse for larger volumes for containers 1 and 2 however improved for container 3. If valid model priors are known then the performance across container volumes can be more consistent as seen in the non-parametric methods with accurate priors. The semi-parametric method usually interpolates these performances with sufficient overlap with the highest performing method.

B. Semi-Parametric Estimation for Diverse Containers

It is shown that the semi-parametric combines the qualities of the other methods by leveraging online system identification and model priors. Due to the strong priors in Figure 7a the semi-parametric method V_θ was very close in prediction to

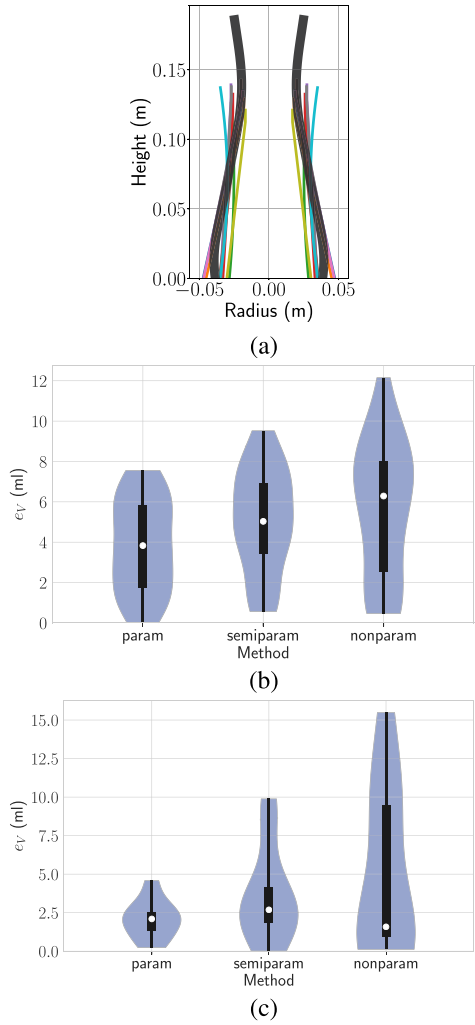


Fig. 8. Container 3 negative example with weak priors along with method statistical data for pours of 100 ml in 8b and 200 ml in 8c.

the non-parametric method for container 1. These profiles vary more for the parametric method and semi-parametric method when the container is sufficiently far from the training set. The semi-parametric method was used on a total of 13 containers pouring target volumes of 100 ml with 10 trials each. The final volumetric error for each container is shown versus each containers height in Figure 9. The mean and variance is shown for each container and a Gaussian process is fit with a radial basis function kernel to demonstrate the increase in error vs height with associated variance. This trend is attributed to the fact that the lower maximum velocity used in the trapezoidal trajectory generator was constant for all containers. This means that the slowest angular rate was constant for the containers which reduces the control authority for larger containers. This can be remedied by making the lower maximum velocity a function of container height.

In Figure 10 the relationship between final error and final pour time is shown for the same pours shown in Figure 9. Over the majority of pours, the performance is largely under 5 ml error and between 20 to 45 seconds for 100 ml pours. The main two outliers to are those with largest height as shown in Figure 9.

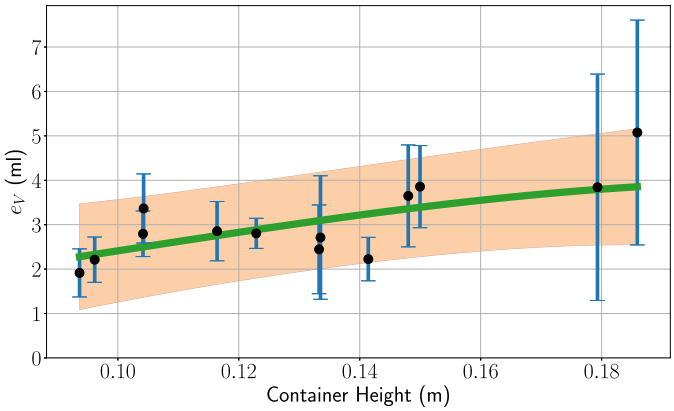


Fig. 9. Semi-parametric method: final error versus container height for 13 containers. Mean and variance are shown for 10 pours of 100ml each. A GP fit with kernel defined in (11) where $\sigma = 5.7$, $l = 0.12$ and $w = 0.0$, shows the increase in error with container height.

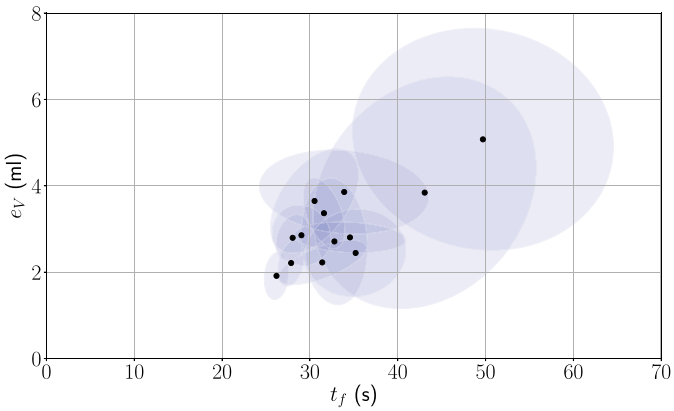


Fig. 10. Semi-parametric method: final error versus pour time for 13 containers pouring 100 ml with 10 trials each. Mean and covariance are shown. Larger error corresponds to larger container height in Figure 9.

Therefore with the remedy of making the lower maximum velocity a function of container height, larger containers would also be expected to perform within 5 ml error. However this remedy while decreasing final error would also increase the pour time, therefore requiring task based optimization to determine appropriate parameters for best performance. In practice, the volume profile for containers observed before can be stored and utilized as an additional prior therefore increasing the performance with continued operation.

V. Conclusions

We present a system capable of pouring fluids from new containers accurately and quickly in a single attempt. This is done by comparing the profile of the new container to simulated containers pouring fluid with known properties. By defining the maximum volume profile as the maximum fluid containable at a given angle, it is asserted that containers with similar geometries have similar maximum volume profiles. This closeness in geometry is used to combine priors in a mixture model to estimate the maximum volume profile required for control. The fluid is detected in the receiving container using both weight and visual detection of the fluid. We propose a hybrid controller that accounts for time delay in the plant and attenuates volumetric error given a specified volume trajectory from an optimal time, trapezoidal trajectory generator that accounts for the residual in the volume profile estimation. We show that by combining online system identification and model priors through a Gaussian process, we can maximize performance with the specified system without tuning parameters for a given container. We show that for constant minimum, maximum desired volume velocity we can achieve performance of under 5ml error and between 20 to 45 second pours for the majority of containers. For larger containers the accuracy can be increased while sacrificing pour time. We demonstrate this system on the Rethink Robotics Sawyer manipulator as well as an implementation on the KUKA LBR iiwa manipulator. Next steps include increasing the complexity of receiving glass detection as well as relaxing the stated assumptions by mitigating spillage, and expanding this implementation to non-symmetric containers.

References

[1] A. Yamaguchi and C. G. Atkeson, "Stereo vision of liquid and particle flow for robot pouring," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots*, Nov. 2016, pp. 1173–1180.
[2] C. Schenck and D. Fox, "Towards learning to perceive and reason about liquids," in *Proc. Int. Symp. Exp. Robot.*, 2017, pp. 488–501.
[3] R. Mottaghi, C. Schenck, D. Fox, and A. Farhadi, "See the glass half full: Reasoning about liquid containers, their volume and content," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct., 2017, pp. 1889–1898, doi: 10.1109/ICCV.2017.207.
[4] T. Tsuji and Y. Noda, "High-precision pouring control using online model parameters identification in automatic pouring robot with cylindrical ladle," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2014, pp. 2563–2568.
[5] Y. Noda and K. Terashima, "Simplified flow rate estimation by decentralization of Kalman filters in automatic pouring robot," in *Proc. SICE Annu. Conf.*, Aug. 2012, pp. 1465–1470.
[6] O. Kroemer, E. Ugur, E. Oztop, and J. Peters, "A kernel-based approach to direct action perception," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 2605–2610.
[7] M. Tamosiunaite, B. Nemec, A. A. A. Ude, and F. Wörgötter, "Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives," *Robot. Auton. Syst.*, vol. 59, no. 11, pp. 910–922, 2011.
[8] L. Rozo, P. Jiménez, and C. Torras, "Force-based robot learning of pouring skills using parametric hidden Markov models," in *Proc. 9th Workshop Robot Motion Control*, 2013, pp. 227–232.
[9] S. Brandl, O. Kroemer, and J. Peters, "Generalizing pouring actions between objects using warped parameters," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Nov. 2014, pp. 616–621.
[10] J. D. Langsfeld, K. N. Kaipa, and S. K. Gupta, "Selection of trajectory parameters for dynamic pouring tasks based on exploitation-driven updates of local metamodels," *Robotica*, vol. 36, pp. 141–166, 2017.
[11] T. Lopez-Guevara, N. K. Taylor, M. U. Gutmann, S. Ramamoorthy, and K. Subr, "Adaptable pouring: Teaching robots not to spill using fast but approximate fluid simulation," in *Proc. 1st Annu. Conf. Robot Learn.*, Nov. 13–15, 2017, vol. 78, pp. 77–86.
[12] C. Schenck and D. Fox, "Visual closed-loop control for pouring liquids," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2017, pp. 2629–2636.
[13] M. Kennedy, K. Queen, D. Thakur, K. Daniilidis, and V. Kumar, "Precise dispensing of liquids using visual feedback," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 1260–1266.
[14] F. Ramos, M. Gajamohan, N. Huebel, and R. D'Andrea, "Time-optimal online trajectory generator for robotic manipulators," *Inst. Dyn. Syst. Control*, ETH Zurich, Zurich, Switzerland, p. 6, 2013.
[15] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1395–1403.
[16] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 153:1–153:12, Jul. 2014.