Stereo Vision of Liquid and Particle Flow for Robot Pouring

Akihiko Yamaguchi[1] and Christopher G. Atkeson[1]

***Abstract*— We explore stereo vision for recognizing liquid and particle flow as 3D points (a point cloud). In our pouring research [1], we noticed that we could detect liquid flow using optical flow detection, especially with the Lucas-Kanade method [2]. In this paper we extend this idea so that we can reconstruct 3D liquid flow from a stereo camera in order to learn dynamical models of flow. Such dynamical models would be useful to reason about pouring behaviors. We demonstrate our method in pouring various materials: water, coke, jelly, dish liquid, and creamer powder. The results show that our method could detect the 3D flow as a point cloud, and they captured the actual flow phenomenon. We also show that our method works in a robot pouring scenario.**
Accompanying video: `https://youtu.be/2oFjVJwXhKs`

I. Introduction

Pouring is a fundamental daily life skill. While pouring is a stand-alone task, sometimes it is involved in other tasks such as cooking. Making robots that are capable of pouring is useful, but this is still a difficult problem. Although there are many studies of robot pouring [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], all of them are situation specific pouring. The difficulties of pouring are: (1) Pouring is a deformable object manipulation. (2) Pouring involves different strategies, e.g. tipping a cup of water, shaking a ketchup bottle, and squeezing a shampoo bottle. (3) Modeling liquid and particle behavior is difficult. (4) Perception of flow is difficult.

We are exploring robot pouring as a case study of manipulating deformable objects [14] and in a robot learning context [15], [16], [1], [17]. In [14], we considered many variations of pouring, such as materials, container shapes, initial poses of containers, and target amounts. Through these studies including robot experiments with a PR2 and a Baxter robot as well as dynamic simulation experiments, we found: (A) Adaptation of skill (e.g. adjusting shaking parameters for a new material and container) is not difficult [14]. (B) A model-based reinforcement learning approach gives us a comprehensive solution to planning and learning problems in pouring [16]. (C) A model-based reinforcement learning can generalize (learned pouring behavior works in unexperienced situations) [17]. (D) Learning dynamical system is improved by decomposition of dynamics into subtasks [15]. (E) Stochastic neural networks are useful in learning dynamics [1]. Based on these findings, we conducted robot experiments with the PR2 robot, and obtained positive results as reported in [1]. However the implementation for perceiving material flow did not perform as well as we would like. As we discussed in [15], learning dynamics of pouring needs to perceive (a) the amount of the material in the source container, (b) the amount of the material in the receiving container, and (c) flow properties including flow existence, flow position in 3D coordinates, flow amount, and flow shape (width/variance or a more informative representation). This paper focuses on solving (c); we develop a flow tracking method.

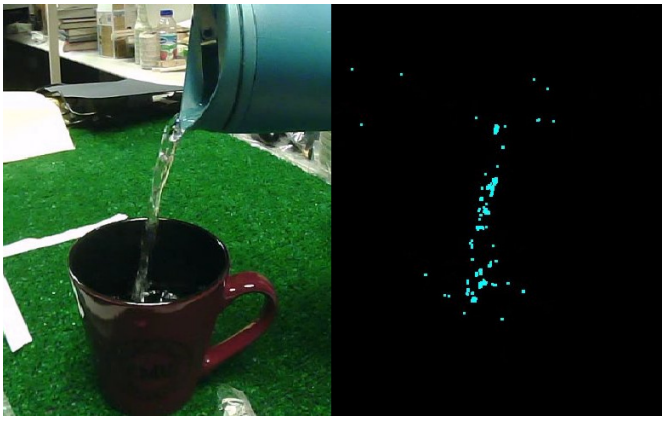

Fig. 1. How the Lucas-Kanade optical flow detection [2] works with water flow. Left: camera view. Right: optical flow. In order to obtain the right image, we applied the optical flow detection method, thresholded at a certain speed, and applied "erode" and "dilate" operators. Refer to the accompanying video.

The (material) flow detection used in [1] is based on the Lucas-Kanade method [2] implemented in OpenCV (`http://opencv.org/`) for obtaining optical flow during pouring. We noticed that we can detect liquid and particle flows including water flow by measuring optical flow with an RGB camera as shown in Fig. 1. In [1], we used two RGB cameras to obtain flow position in 3D. However there were issues: (I) human operators were replacing the cameras for each experiment which was inconsistent over many trials, and (II) it was hard to distinguish the optical flow of the robot movement (especially the shaking motion) and material flow. For the purpose of (II), we used color information of the material, but it was inaccurate due to the difficulty of capturing moving material and it was not applicable to transparent liquids. In this paper, we apply a stereo vision method to reconstruct 3D liquid flow from the optical flow of two cameras. This method provides 3D liquid flow as a point cloud, from which we can extract the information of (c) mentioned above. Since we do not use the color information of material, our new method is applicable to transparent liquids. The point cloud of liquid flow can be transformed

[1]A. Yamaguchi and C. G. Atkeson are with The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213, United States `info@akihikoy.net`

into the robot frame, so it is easy to remove the movements of the robot and the environment.

We test our method by applying it to various materials including liquids and granular materials. The results show that our method can detect 3D flow points, and capture the actual flow phenomenon. We also apply our method to robot pouring where plastic beads are poured. The results show that our method works in the robot pouring scenario. We think our method has wide applications including flow detections of daily-life robotic tasks and industrial tasks.

Related Work

There is much work on robot pouring [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13] including learning to improve pouring skills. However they used a limited perception of materials.

An industrial pouring task (pouring molten metal into a mold) was researched in [3], [4]. An electric resistance-type level sensor was used to measure the liquid levels in the source container [3], [4]. A laser sensor was used to measure the liquid levels in the receiver containers [4].

In [8], [9], a scale was used to weigh the amount poured into a receiving container. In [11], a method to detect important parts of container from a point cloud was developed. It could detect the opening of a container. Some studies directly or indirectly measured an amount in the source or receiving container with a force-torque sensor attached on the wrist of the robot arm [10], [12]. In [11], [5], [6], [7], [13], no amount sensor was used. In these researches, the initial amount of the source container was within the capacity of the receiving container.

An interesting approach to measuring flow was proposed in [18] where a water flow was perceived from audio information. The robot could recognize if water was poured into a container or a non-container object based on the sound. Another study of liquid perception was water detection for unmanned ground vehicles [19] where water in the environment such as a puddle or pond was detected.

Optical flow methods are fundamental computer vision tools in robotics to detect movement of objects. An application is navigation of small UAVs [20], [21]. However it is not popular to apply an optical flow method to detect liquid flow. As stereo vision, our approach is a kind of sparse stereo. While dense stereo methods compute disparities for all corresponding points of two images, sparse stereo methods compute disparities for a limited number of feature points. Schauwecker *et al.* proposed a feature detector for sparse stereo [22]. Witt *et al.* proposed an edge-based search as a sparse stereo method [23], [24], which is close to ours. However using optical flow would be more adequate for detecting liquid flow, rather than edges. Using optical flow in stereo vision is a popular approach [25], [26], [27]. We are focusing on detecting liquids with optical flow. Although this paper does not provide a novel technique of computer vision, the case study of 3D flow perception would be a novel challenge.

Recently deep neural networks have been used to detect and track liquid flow [28]. In order to train the neural networks, they generated large amounts of labeled pouring data using a liquid simulator and a rendering engine. They reported that using multiple frames was better than using a single frame, which is close to our idea used in [1]: optical flow can perceive liquid flow. They also showed that a recurrent network with LSTM (long short-term memory) worked well. However they did not mention if they could distinguish flow from robot movement that we had trouble with in [1]. Our approach is superior to theirs in estimating 3D liquid flow. Additionally our method can detect flows of viscous liquids such as jelly and granular materials such as powder.

II. Stereo Vision of Liquid and Particle Flow Detection

First we apply an optical flow detection using the Lucas-Kanade method [2], implemented in OpenCV, to each image of a stereo camera. Spatial and temporal filters are applied to the optical flow images to compensate for the delay between the left and right cameras. Spatial filters also remove noise. Then we apply a stereo method to the two images to reconstruct 3D flow. Since the images from the optical flow are texture-less, we do not apply a standard stereo method like block matching. Instead, by assuming that flow is a vertical thin line or curve, we compute only a single disparity per each epipolar line, and find matching points around the peak. Consequently we can reconstruct 3D flow robustly. Obviously this method decreases accuracy when the flow has a complicated shape. According to our experience of learning flow dynamics and planning with them [15], [1], we think this inaccuracy will not be a big issue. Another issue will be that the movement of the robot body and environment is detected inaccurately. In this section we also discuss a way to remove such non-flow movements.

A. Optical Flow

We apply the Lucas-Kanade method to detect optical flow, remove noise by applying a spatial filter, and apply a temporal filter.

We use the `cvCalcOpticalFlowLK` function implemented in OpenCV (we use a "legacy" implementation with parameters $(3, 3)$ for the window size). This function takes a grayscale images of the current and the previous frames, and returns horizontal and vertical components of velocity for every pixel. Let $v_x(u, v)$, $v_y(u, v)$ denote the x and y velocity at (u, v) respectively. We calculate a "speed" image by $I_{\mathrm{spd}}(u, v) = \sqrt{(v_x(u, v))^2 + (v_y(u, v))^2}$, and remove noise by thresholding: $I_{\mathrm{flw1}}(u, v) = 1$ if $I_{\mathrm{spd}}(u, v) > i_{\mathrm{th\_spd}}$ otherwise 0. The value of $i_{\mathrm{th\_spd}}$ was 5 in our experiments (with the cameras of 640x480 pixels, 60 FPS).

The issues of using $I_{\mathrm{flw1}}(u, v)$ directly are: (1) $I_{\mathrm{flw1}}(u, v)$ is noisy especially when we use a stereo camera mounted on a robot (the robot is slightly vibrating). (2) The flow is not detected similarly between the two cameras because flow is a subtle phenomenon, and our cameras are not synchronized.

We apply multiple spatial and temporal filters. The parameters below are the values used in our experiments. (A) Apply an "erode" (size 1) and "dilate" (size 1) operator to $I_{\mathrm{flw1}}(u,v)$ for noise reduction, and obtain $I_{\mathrm{flw2}}(u,v)$. (B) Apply a temporal filter; we simply take a pixel-wise OR operation of $I_{\mathrm{flw2}}(u,v)$ over a filter length (5), and obtain $I_{\mathrm{flw3}}(u,v)$. (C) Apply "dilate" (size 6), "erode" (size 8), "dilate" (size 10), and a filter with a vertically-long kernel (1x20, filled with $1/20$) in this order to $I_{\mathrm{flw3}}(u,v)$, and obtain $I_{\mathrm{fmask}}(u,v)$. (D) Mask $I_{\mathrm{flw3}}(u,v)$ with $I_{\mathrm{fmask}}(u,v)$, and obtain $I_{\mathrm{flw4}}(u,v)$. (E) Apply the filter with the vertically-long kernel (the same as above) to $I_{\mathrm{flw4}}(u,v)$, and obtain $I_{\mathrm{flw5}}(u,v)$, which is the final outcome of the flow detection. (A) is for a noise reduction of optical flow. (B) and (E) are for handling the issue (2). (C) and (D) are for stronger noise reduction than (A).

We apply the above calculation for each frame of the left and right images independently. Let $I_{\mathrm{ffL}}(u,v)$, $I_{\mathrm{ffR}}(u,v)$ denote the left and right flow images obtained by the above procedure respectively.

B. Reconstruct 3D Flow

First we find a disparity of a right image from a left image for each epipolar line where the correlation is maximized. Then we search "sparse matching points" between left and right images around the disparity. Finally we reconstruct 3D points from them.

We assume that $I_{\mathrm{ffL}}(u,v)$ and $I_{\mathrm{ffR}}(u,v)$ are already rectified for computing stereo disparity, i.e. each horizontal line is an epipolar line.

The correlation is computed as a sum of a pixel-wise AND operation: for each epipolar line at vertical position v, $I_{\mathrm{corr}}(v,d) = \sum_u (1 \text{ if } I_{\mathrm{ffL}}(u,v) > 0 \text{ and } I_{\mathrm{ffR}}(u-d,v) > 0 \text{ otherwise } 0)$. The disparity at v is given by $d^*(v) = \arg\max_d I_{\mathrm{corr}}(v,d)$. Note that if $\max_d I_{\mathrm{corr}}(v,d)$ is smaller than $i_{\mathrm{th\_match}} = 16$, we assume that there are no matching points at v. The sparse matching points are computed as follows: for each vertical position v, find all points $\{u\}$ on the epipolar line that satisfy $I_{\mathrm{ffL}}(u,v) > i_{\mathrm{th\_exist}}$ and $I_{\mathrm{ffR}}(u-d^*(v),v) > i_{\mathrm{th\_exist}}$. That is, we find points that exist in both images around the disparity. From the point pairs $\{u,v\}$ and $\{u-d^*(v),v\}$, we reconstruct the 3D points. These are the candidates of 3D flow points. Note that they may include moving points such as a robot body. The value of $i_{\mathrm{th\_exist}}$ used in our experiments was 0.08. For a faster computation, we processed the epipolar line every 10 vertical pixels.

C. Inaccuracy of 3D Flow Points

The inaccuracy of 3D flow points would be caused by the flow stereo computation where we find only one disparity per each epipolar line. When the material flow is forming a line or a curve, this assumption would be reasonable. This assumption can be violated by moving objects such as a robot body. This issue is reduced by the region of interest described in the next section.

D. Region of Interest

We consider two types of ROI (Region of Interest). (1) A ROI applied to image planes. We define a ROI in 3D space as a primitive shape such as a cone and a polygon, and map it onto the left and the right image planes. We compute the 3D flow computation only in the ROI. (2) A ROI applied to the 3D flow points. We use the 3D flow points existing in a ROI defined in 3D space as a primitive shape such as a cone and a bounding box. Both (1) and (2) are similar, but (1) improves the efficiency of the stereo computation.

Additionally (1) is useful to remove inaccurate 3D flow points such as the robot movement as we discussed above. Usually we have a robot geometry model and a kinematic state of the robot, thus we can define a 3D region to remove from the 3D flow points. If we map the 3D region to the image planes, we do not compute the 3D flow points in that region. Hence we can omit computing flow points that would be due to robot movement. A drawback is (1) removes more than (2), i.e. we can define finer 3D ROI with (2). Thus a good way is combining the two methods of ROI.

III. Discussion: Understanding Flow Dynamics

Here we discuss using 3D flow points in learning flow dynamics. The elements involved in flow dynamics are a source container, a receiving container, material amounts in containers, and flow. We need to consider 3D information of the containers and the flow to understand the flow dynamics. The material comes from the opening of the source container, becomes flow, and goes into the receiving container through its opening. Thus finding the openings of containers is necessary as well as the 3D flow points.

In our preliminary experiments, we used a depth sensor to obtain the position and orientation of containers for grasping and pouring. Since we had geometry models of the containers, we could obtain the opening information from them. However we noticed that this approach has an issue: since we used different sensors for obtaining opening information (the depth sensor) and 3D flow points (the stereo camera), the calibration accuracy was critical. If the calibration (of the sensor poses) had an error, the dynamical models learned from the opening information and the 3D flow points became unreliable.

In this paper we propose a different approach. We obtain all information that are used in modeling the flow dynamics from the same sensor. Concretely we obtain the opening information as well as the 3D flow points from the stereo camera. Even if the calibration had an error, the learned dynamical models would be more reliable since the position of the opening and the 3D flow points obtained from the stereo camera still have correct relationship.

We implement a simple opening finding algorithm. We define a opening of a container as a polygon. From a current container pose estimated in some way, we render the opening polygon on the stereo image planes. We apply an edge detection filter to the stereo images. We calculate the correlation between the rendered opening polygons and the edges on the images. The opening fitting is done by an

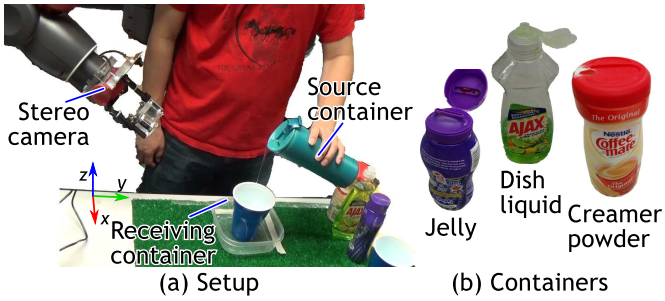

Fig. 2. (a) Setup of the experiments. (b) Containers.

optimization method in terms of the container pose so that the pose maximizes this correlation.

IV. Experiments: Liquid and Particle Flow

We apply the 3D flow point estimation to flows of various materials. The setup of the experiment is shown in Fig. 2(a). As the receiving container, we use a plastic cup of diameter 10 cm on the table. The cup is in a 15 cm x 15 cm tray. We use a stereo camera attached to the right arm of a Baxter robot, thus we obtain 3D observations in the robot frame. The stereo camera is a pair of two USB cameras; each one is ELP Co. USBFHD01M with a wide lens. Each camera can capture images of 640x480 pixels in 60 FPS (frames per second). The left and the right images are not synchronized. Refer to the accompanying video.

We implement the 3D flow detection algorithm as fast as possible with careful thread programing. Each capture and optical flow detection is executed independently. The optical flow detection was actually done in more than 50 FPS. We calculate the flow stereo in every third optical flow detection. Note that since we use a temporal filter with length 5 frames as mentioned in Section II-A, this frame drop would not miss the flow. We use the same parameters of the 3D flow point estimation through the following experiments.

In each experiment, first we apply the opening detection method to find the opening of receiving container. We do not move the receiving container during the experiment. We ask a human to pour liquid or powder from a source container. During pouring, we measure the 3D flow points with our method.

In order to analyze the results quantitatively, we define an average flow position calculated at each frame. The average flow position is an average of the 3D flow points except for the points whose height (z value) are greater than a specific value. The excluded points are considered as a container or an operator movement.

A. Flow of Water

First, we ask a human to pour water from a bottle. During pouring, the human is asked to move the source container in some ways: moving sideways ($-y$ and $+y$ directions), moving forward and backward ($+x$ and $-x$ directions), moving circularly, and moving to pour into the tray (outside the receiving container).

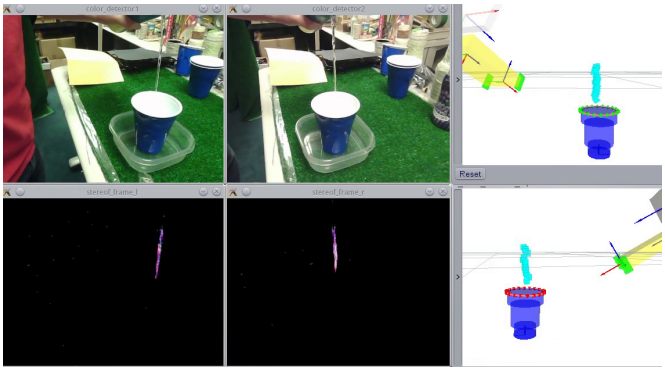

Fig. 3. Snapshot of the flow detection program in pouring water. Top-left two images: left and right camera views. Bottom-left two images: detected flows (colored according to the x position). Right two images: 3D rendering view with a receiving container, a right gripper of the robot, and the stereo camera on it; top: $+y$-directional view, bottom: $+x$-directional view.

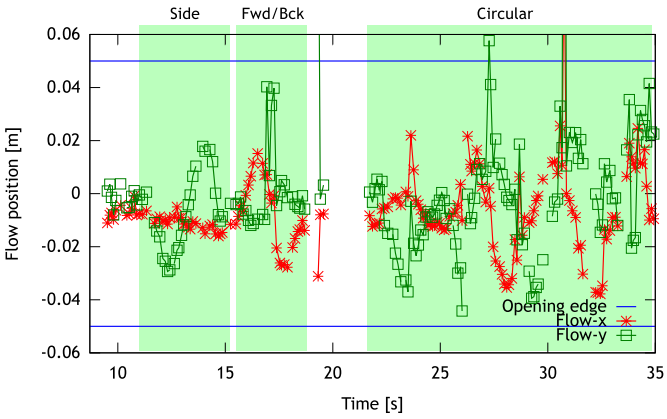

Fig. 4. Trajectories of the average flow position per time during pouring water. The position is relative to the center of the receiver opening.

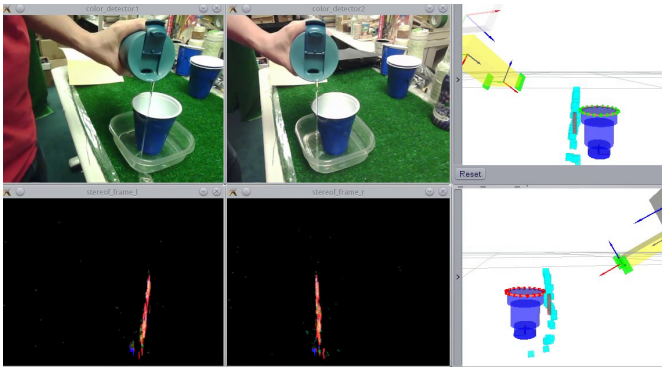

Fig. 5. Snapshot of the flow detection program in pouring water. Refer to the caption of Fig. 3.

Fig. 3 shows a snapshot during pouring. You can see the water flow is detected as optical flow, and the 3D flow points are reconstructed as rendered in the viewers. Fig. 4 shows the x and y trajectories of the average flow position relative to the center of the receiver opening. The disconnections of curves mean that the flow was not

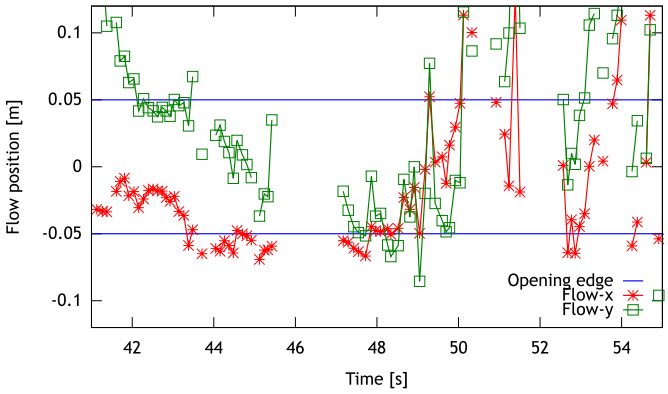

Fig. 6. Trajectories of the average flow position per time during pouring water. The position is relative to the center of the receiver opening.

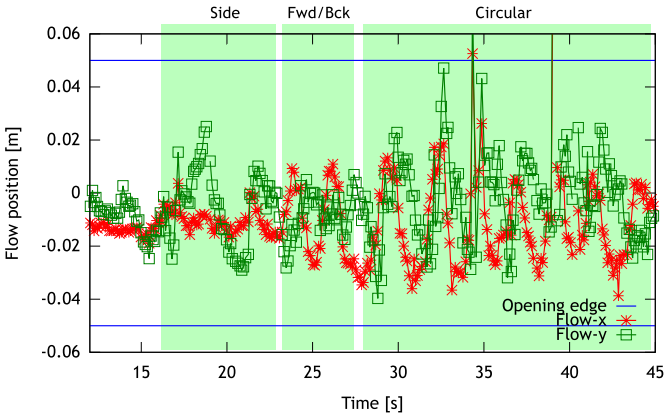

Fig. 7. Trajectories of the average flow position per time during pouring coke. The position is relative to the center of the receiver opening.

observed during these periods. In the first two parts (1 sec to 18 sec), the actual flow was moving via $(x, y) = (0, 0), (0, -r), (0, +r), (0, 0), (+r, 0), (-r, 0), (0, 0)$ (r is a moving radius). The trajectories are close to this movement. There are some jumps of y in the forward and backward movement. They were caused by the container movement (the height threshold to exclude container movement was inaccurate). The big jump around 20 sec was due to the same reason where the human took a backwards motion of the source container. In the last part (after 22 sec), the source container was moving circularly. From the graph, we can see that the average flow position is moving like a circle. Be aware that the flow positions are almost always within the edge of the receiver opening (diameter 5 cm).

Fig. 6 shows the x and y trajectories of the average flow position during pouring into the box. The actual flow was outside the receiving container (diameter 5 cm), which corresponds to the trajectories. Fig. 5 shows a snapshot during pouring. In the rendered images, the 3D flow points are clearly outside the container.

B. Flow of Colored Liquid

We ask a human to pour coke from a can. During pouring, the human is asked to move the source container similarly to the water case: moving sideways ($-y$ and $+y$ directions), moving forward and backward ($+x$ and $-x$ directions), and moving circularly.

Fig. 7 shows the x and y trajectories of the average flow position relative to the center of the receiver opening. We can see the sideways movement, the forward and backward movement, and the circular movement. However even during the forward and backward movement ($+x$ and $-x$ directions), y is changing. This was caused by the instability of the material flow, i.e. this was a real phenomenon. The human poured by rotating the can around the x axis, so the flow was stable in the x direction (x change was small), but there was uncontrolled waving in y. Fig. 8 shows a snapshot during pouring. We can see the 3D point reconstruction similar to that of the water case. Fig. 9 shows another snapshot where the flow was very thin; a part of the flow was droplets. The flow in the camera views is not clear, but the optical flow could detect the flow. The 3D flow points were reconstructed as well.

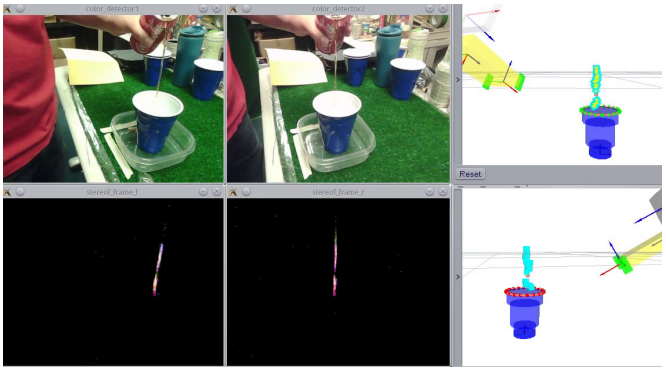

Fig. 8. Snapshot of the flow detection program in pouring coke. Refer to the caption of Fig. 3.

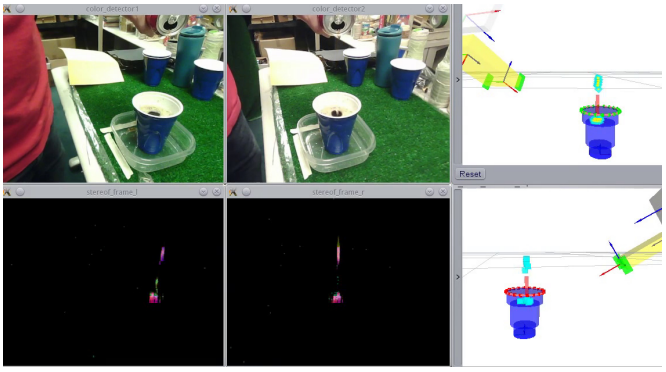

Fig. 9. Snapshot of the flow detection program in pouring coke. Refer to the caption of Fig. 3.

C. Flow of Jelly, Dish Liquid, and Powder

We ask a human to pour three types of materials: blueberry jelly (very viscous, blue color), dish liquid (slightly viscous, transparent green), and creamer powder (white color). These containers are shown in Fig. 2(b).

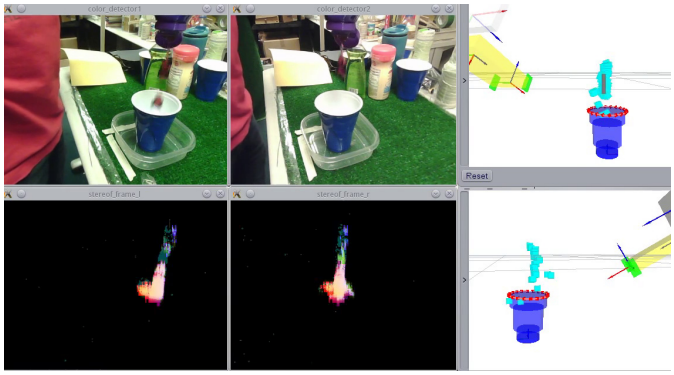

Fig. 10. Snapshot of the flow detection program in pouring jelly. Since the views are not synchronized, this snapshots are composed from two time frames. Refer to the caption of Fig. 3.

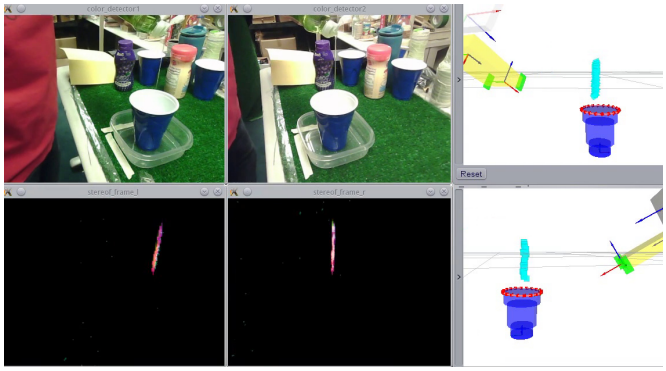

Fig. 11. Snapshot of the flow detection program in pouring dish liquid. Refer to the caption of Fig. 3.

Fig. 10, 11, and 12 show snapshots during pouring jelly, dish liquid, and creamer powder respectively. The jelly flow was a discontinuous sequence of big blocks, and the dish liquid flow was similar to the water and the coke flows but was more thin and smooth. The creamer powder flow was unlike all of them; the flow spread from top to bottom. We could recognize these differences even from the 3D flow points. Since the creamer powder was not smooth, the human waved the container. This motion was detected as the 3D flow points as shown in 12.

Fig. 13 shows the x and y trajectories of the average flow position of jelly, dish liquid, and creamer powder respectively. The human tried to control the flow around the center of the receiving container. From the trajectories, we can see the significant difference of the materials. There are many gaps (disconnections) in jelly. Actually the flow was a discontinuous sequence as mentioned above. The flow of dish liquid is more smooth. The flow of creamer powder has bigger deviation, which also corresponds with the actual phenomenon; the powder flow spread widely. Thus our method could detect the 3D flow points, and they were capturing the actual flow dynamics.

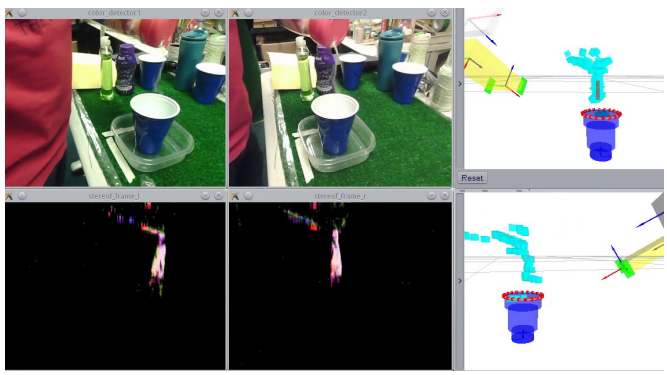

Fig. 12. Snapshot of the flow detection program in pouring creamer powder. Refer to the caption of Fig. 3.

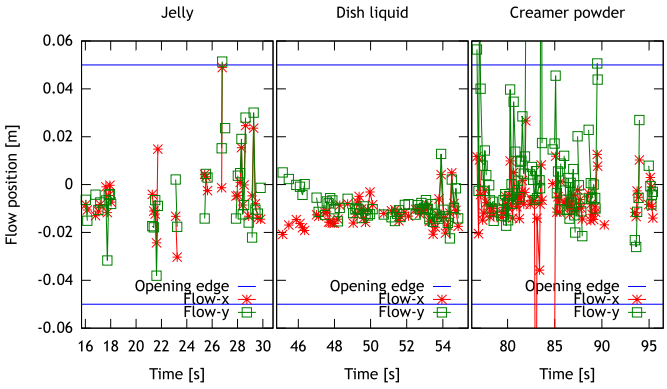

Fig. 13. Trajectories of the average flow position per time during pouring jelly, dish liquid, and creamer powder. The position is relative to the center of the receiver opening.

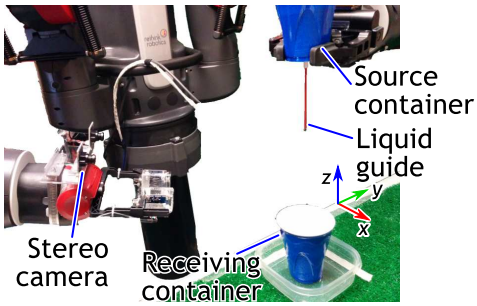

Fig. 14. Setup of the experiments for accuracy evaluation.

V. Experiments: Accuracy Evaluation

We evaluate the accuracy of the flow position estimates informally. The setup of the experiments is shown in Fig. 14. We let the Baxter robot hold a source container with its left gripper. A liquid guide (a thin plastic tube whose internal diameter is 2.5 mm) is attached to the bottom of the source container in order to generate straight flow. In the experiments, we place the left arm so that the liquid guide locates at the center of the receiving container. Then we move the left arm to five different positions: $(d_x, d_y) = (0, 0)$, $(d_l, 0)$, $(-d_l, 0)$, $(0, d_l)$, and $(0, -d_l)$ where (d_x, d_y) denotes the x

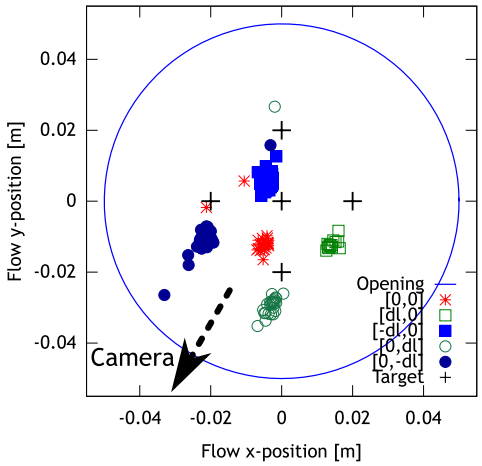

Fig. 15. x and y positions of the average flow relative to the center of the receiver opening. The direction to the stereo camera is illustrated with an arrow.

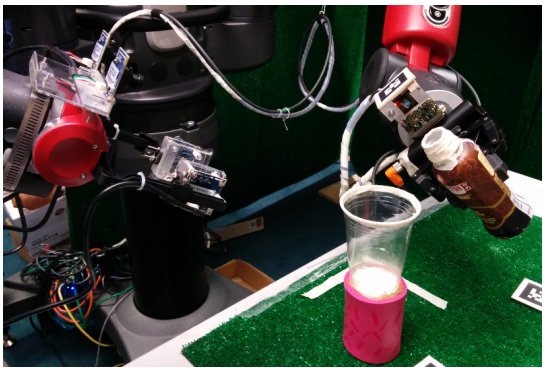

Fig. 16. Setup of the robot pouring experiments. A stereo camera is mounted on the right wrist of the robot.

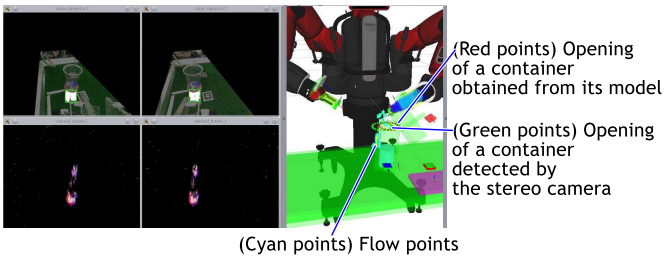

Fig. 17. Snapshot of the flow detection program in robot pouring. Top-left two images: left and right camera views with color detection for measuring the poured amount. Bottom-left two images: detected flows. Right image: 3D rendering view.

and y offset from the initial position, and $d_l = 0.02$ m. At each position, a human pours water into the source container, and observes the flow.

Fig. 15 shows the x and y positions of the average flow relative to the center of the receiver opening. There are three different types of errors: (1) 10 to 15 mm offset along the depth direction of the stereo camera, which seems to be a common value in all targets. (2) Small variance in each target. (3) A few outliers. (1) is the dominant error. The possible error sources are as follows: (A) The inaccuracy of the 3D flow estimation mentioned in Section II-C. (B) The error of the stereo camera calibration. (C) The error of the opening detection. (D) The control error of the Baxter robot. (E) The error of the initial position of the source container. (F) A slight movement of water flow. For distinguishing these error sources, we need to conduct more organized experiments. Since the dominant error (1) seems to be consistent for the different targets, we would be able to reduce this error somehow. We think our method has a good accuracy.

VI. Experiments: Robot Pouring

We explore how our method works in a robot pouring situation. We use the Baxter robot and the stereo camera used in the previous experiments. Fig. 16 shows the setup. Because of the hardware limitation, the stereo camera works at 30 FPS. We use a pouring behavior represented as state machines developed in [14]. The parameters of the actions are planned by their method, including pouring location. Refer to the accompanying video.

We use the two types of ROI as mentioned in Section II-D. ROI-1 is applied to the image planes. We use a prior knowledge that flow goes downwards from the opening of the source container. ROI-1 is a cone whose vertex is at the center of the opening of the source container, the cone's opening angle is 30°, and its orientation is upright in 3D space (does not change). ROI-2 is applied to the flow points in 3D space to remove non-flow points remaining after ROI-1. These are caused by the movement of the source container and the robot body. ROI-2 is also a cone whose vertex is at the center of the opening of the source container, the cone's opening angle is 60°, and its orientation is upside-down of the source container orientation. Unlike ROI-1, the orientation of ROI-2 changes with the orientation of the source container.

We use blue plastic beads as the material to be poured. Fig. 17 shows a snapshot in pouring by the robot. The variance of the flow seems to be large. This was due to the material property; the plastic beads were close to powder, rather than liquids. Fig. 18 shows the x and y trajectories of the average flow position. There are two reasons of the gaps (disconnections): (1) there were actually no flow, and (2) our method failed to detect. The major reason of (2) would be the frame rate of the cameras (30 FPS). For increasing the reliability, we should increase the frame rate. However the flow points captured in 30 FPS still seem to have useful information for learning.

VII. Conclusion

We explored stereo vision for recognizing liquid and particle flow as 3D points. Based on our pouring research [1] where we detected flow using optical flow detection, especially the Lucas-Kanade method [2], we extended the idea so that we can reconstruct the 3D flow from a stereo camera.

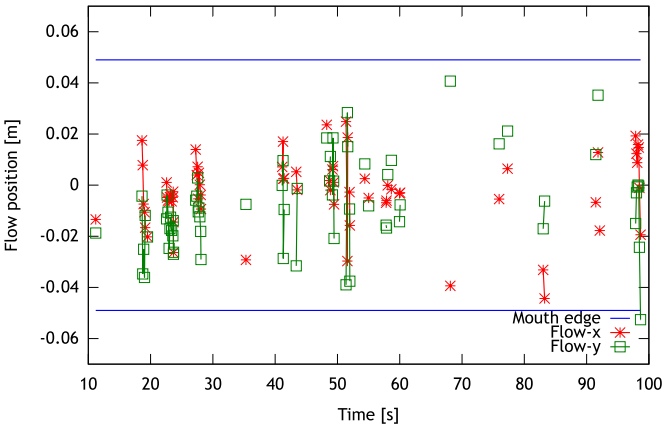

Fig. 18. Trajectories of the average flow position per time during robot pouring. The position is relative to the center of the receiver opening.

The purpose of this work is for learning the dynamical models of flow, which would be useful to reason about pouring behaviors. We demonstrated our method in pouring various materials: water, coke, jelly, dish liquid, and creamer powder. We also applied our method to robot pouring where plastic beads were poured. The results showed that our method could detect 3D flow points, and they were capturing the actual flow phenomenon. Future work includes (1) increasing the robustness of perception by reducing the number of parameters (especially those of spatial and temporal filters), and (2) experiments to evaluate the accuracy of the 3D flow points in a way more organized than that in Section V.

REFERENCES

[1] A. Yamaguchi and C. G. Atkeson, "Neural networks and differential dynamic programming for reinforcement learning problems," in *the IEEE International Conference on Robotics and Automation (ICRA'16)*, 2016.

[2] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *the 7th international joint conference on Artificial intelligence (IJCAI'81)*, 1981, pp. 674–679.

[3] K. Yano, T. Toda, and K. Terashima, "Sloshing suppression control of automatic pouring robot by hybrid shape approach," in *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, vol. 2, 2001, pp. 1328–1333.

[4] Y. Noda, K. Yano, and K. Terashima, "Control of self-transfer-type automatic pouring robot with cylindrical ladle," *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 295–300, 2005.

[5] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *the IEEE International Conference on Robotics and Automation (ICRA'09)*, 2009, pp. 763–768.

[6] M. Mühlig, M. Gienger, S. Hellbach, J. J. Steil, and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *the IEEE International Conference on Robotics and Automation (ICRA'09)*, 2009, pp. 1177–1184.

[7] R. Jäkel, S. Schmidt-Rohr, M. Losch, and R. Dillmann, "Representation and constrained planning of manipulation strategies in the context of programming by demonstration," in *the IEEE International Conference on Robotics and Automation (ICRA'10)*, 2010, pp. 162–169.

[8] M. Tamosiunaite, B. Nemec, A. Ude, and F. Wörgötter, "Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives," *Robotics and Autonomous Systems*, vol. 59, no. 11, pp. 910–922, 2011.

[9] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz, "Keyframe-based learning from demonstration - method and evaluation," *I. J. Social Robotics*, vol. 4, pp. 343–355, 2012.

[10] L. Rozo, P. Jiménez, and C. Torras, "Force-based robot learning of pouring skills using parametric hidden Markov models," in *the IEEE-RAS International Workshop on Robot Motion and Control (RoMoCo)*, 2013.

[11] S. Brandl, O. Kroemer, and J. Peters, "Generalizing pouring actions between objects using warped parameters," in *the 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids'14)*, Madrid, 2014, pp. 616–621.

[12] L. Rozo, J. a. Silvério, S. Calinon, and D. G. Caldwell, "Learning controllers for reactive and proactive behaviors in human-robot collaboration," *Frontiers in Robotics and AI*, vol. 3, no. 30, 2016.

[13] C. Bowen and R. Alterovitz, "Asymptotically optimal motion planning for tasks using learned virtual landmarks," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 1036–1043, 2016.

[14] A. Yamaguchi, C. G. Atkeson, and T. Ogasawara, "Pouring skills with planning and learning modeled from human demonstrations," *International Journal of Humanoid Robotics*, vol. 12, no. 3, p. 1550030, 2015.

[15] A. Yamaguchi and C. G. Atkeson, "Differential dynamic programming with temporally decomposed dynamics," in *the 15th IEEE-RAS International Conference on Humanoid Robots (Humanoids'15)*, 2015.

[16] ——, "A representation for general pouring behavior," in *in the Workshop on SPAR in the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'15)*, 2015.

[17] ——, "Model-based reinforcement learning with neural networks on hierarchical dynamic system," in *the Workshop on Deep Reinforcement Learning: Frontiers and Challenges in the 25th International Joint Conference on Artificial Intelligence (IJCAI2016)*, 2016.

[18] S. Griffith, V. Sukhoy, T. Wegter, and A. Stoytchev, "Object categorization in the sink: Learning behavior-grounded object categories with water," in *the 2012 ICRA Workshop on Semantic Perception, Mapping and Exploration*, 2012.

[19] A. Rankin and L. Matthies, "Daytime water detection based on color variation," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 215–221.

[20] M. V. Srinivasan, "Visual control of navigation in insects and its relevance for robotics," *Current Opinion in Neurobiology*, vol. 21, no. 4, pp. 535 – 543, 2011.

[21] H. Chao, Y. Gu, and M. Napolitano, "A survey of optical flow techniques for robotics navigation applications," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1, pp. 361–372, 2014.

[22] K. Schauwecker, R. Klette, and A. Zell, "A new feature detector and stereo matching method for accurate high-performance sparse stereo matching," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5171–5176.

[23] J. Witt and U. Weltin, "Sparse stereo by edge-based search using dynamic programming," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, 2012, pp. 3631–3635.

[24] ——, "Robust real-time stereo edge matching by confidence-based refinement," in *International Conference on Intelligent Robotics and Applications*. Springer Berlin Heidelberg, 2012, pp. 512–522.

[25] N. Slesareva, A. Bruhn, and J. Weickert, *Optic Flow Goes Stereo: A Variational Method for Estimating Discontinuity-Preserving Dense Disparity Maps*, 2005, pp. 33–40.

[26] C. Unger, E. Wahl, and S. Ilic, *Efficient Stereo and Optical Flow with Robust Similarity Measures*, 2011, pp. 246–255.

[27] M. Hatzitheodorou, E. Karabassi, G. Papaioannou, A. Boehm, and T. Theoharis, "Stereo matching using optic flow," *Real-Time Imaging*, vol. 6, no. 4, pp. 251–266, 2000.

[28] C. Schenck and D. Fox, "Detection and tracking of liquids with fully convolutional networks," *ArXiv e-prints*, no. arXiv:1606.06266, 2016.