PourIt!: Weakly-supervised Liquid Perception from a Single Image for Visual Closed-Loop Robotic Pouring

Haitao Lin   Yanwei Fu   Xiangyang Xue
Fudan University
{htlin19, yanweifu, xyxue}@fudan.edu.cn

Abstract

Liquid perception is critical for robotic pouring tasks. It usually requires the robust visual detection of flowing liquid. However, while recent works have shown promising results in liquid perception, they typically require labeled data for model training, a process that is both time-consuming and reliant on human labor. To this end, this paper proposes a simple yet effective framework PourIt!, to serve as a tool for robotic pouring tasks. We design a simple data collection pipeline that only needs image-level labels to reduce the reliance on tedious pixel-wise annotations. Then, a binary classification model is trained to generate Class Activation Map (CAM) that focuses on the visual difference between these two kinds of collected data, i.e., the existence of liquid drop or not. We also devise a feature contrast strategy to improve the quality of the CAM, thus entirely and tightly covering the actual liquid regions. Then, the container pose is further utilized to facilitate the 3D point cloud recovery of the detected liquid region. Finally, the liquid-to-container distance is calculated for visual closed-loop control of the physical robot. To validate the effectiveness of our proposed method, we also contribute a novel dataset for our task and name it PourIt! dataset. Extensive results on this dataset and physical Franka robot have shown the utility and effectiveness of our method in the robotic pouring tasks. Our dataset, code and pre-trained models will be available on the project page [1].

1. Introduction

Recent advances have seen the great progress in the capabilities of grasping and manipulating the rigid objects.

Haitao Lin is with Academy for Engineering and Technology, and Engineering Research Center of AI and Robotics, Fudan University.

Yanwei Fu and Xiangyang Xue are corresponding authors. Yanwei Fu is with the School of Data Science, Fudan University, and Fudan ISTBI—ZJNU Algorithm Centre for Brain-inspired Intelligence, Zhejiang Normal University, Jinhua, China.

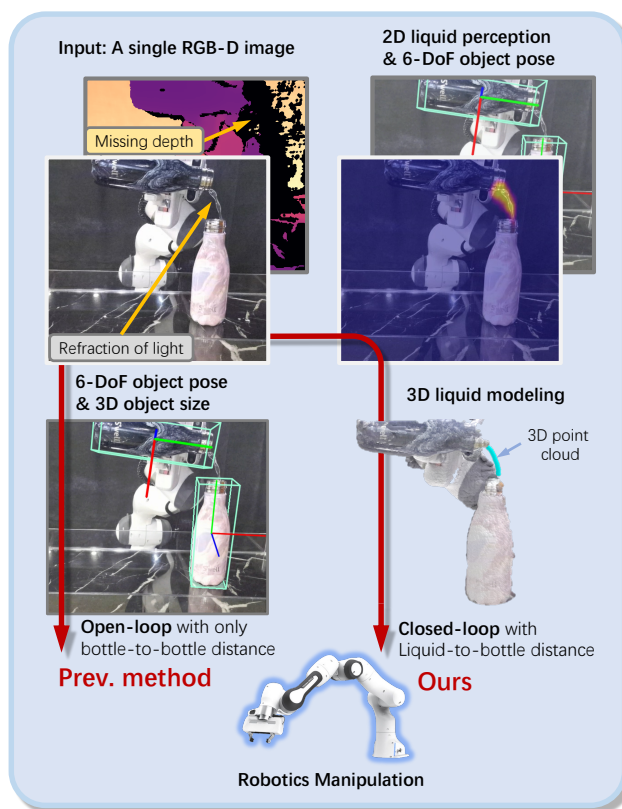[1]Project page. https://hetolin.github.io/PourIt

Figure 1. Our visual closed-loop robotic pouring. Unlike [17], our approach recovers a 3D point cloud of the liquid using the source container's pose and 2D liquid perception data. This allows the robot to pour accurately based on visual feedback, even without the depth measurement of the liquid.

However, the manipulation of non-rigid objects such as liquids, cloth, and rope remains a formidable challenge due to the absence of fixed patterns and geometrical shapes associated with flexible properties. Perceiving images of liquid is particularly challenging due to their reliance on refraction of light as the primary visual cue and the absence of depth measurement as shown in Fig. 1 (top left). Thus, further research to enhance liquid perception would be highly beneficial, especially for robots involved in real-world service

tasks like cooking, drink serving, and plant watering.

Pouring water is a highly relevant task in the field of liquid manipulation, yet it poses a number of significant challenges. These challenges include (1) the need for large amounts of pixel-wise annotated data to facilitate effective training, (2) the lack of salient visual cues within images, and (3) the unavailability of a reliable depth measurement system for accurate liquid pouring. Addressing these challenges is crucial for advancing the state of liquid manipulation technology and enabling practical applications in domains such as robotics and industrial automation.

To address the aforementioned challenges, researchers have explored the use of extra sensors to generate ground-truth labels for robotics pouring. For example, Schenck *et al.* [26] skillfully utilize the thermal camera and heating water to obtain the ground-truth annotations. However, this approach is time-consuming and relies on additional equipment. Other methods [15, 34, 36, 9, 18, 32] rely on the audio but not visual signals to help robotics pouring, which limits their efficacy in noisy environments. Lin *et al.* [17] simply utilize the estimated container's pose and size to calculate the initial pouring point without any liquid perception (Fig. 1), thus cannot guarantee whether the liquid is poured into target container in a closed-loop manner. More recently, a self-supervised method [20] has been proposed to transfer colored liquid into transparent liquid without the need for manual annotations. However, this approach's reliance on colored liquid and a statically placed transparent container limits its applicability in more general environments. Therefore, the challenge of robustly perceiving dynamically out-flowing liquid during pouring operations remains an open question that requires further research.

This paper presents a simple yet effective framework *PourIt!* that solves challenges in robotic pouring tasks, particularly during the pouring stage. Our open-source library is positioned similarly to existing tools such as MoveIt! [7] for motion planning and GraspIt! [19] for grasping planning. We address three specific challenges, including *limited annotated real-world data for training, non-salient visual cues of liquid, and unavailable depth measurement of the liquid.* (1) To address the first challenge, we design an semi-automatic pipeline that will collect images with two-class labels, indicating whether the liquid flows from the bottleneck of the source container. This pipeline enables the robot to collect more data, thereby improving the performance of the network. (2) To tackle the second challenge, we classify the images by distinguishing between the two types of collected data (existence of out-flowing liquid or not). However, such a classification task may not fully focus on the entire liquid region. To overcome this, we use a feature contrast strategy to pull foreground local features close and separate foreground-background local features to identify the liquid region. (3) To resolve the third challenge,

we make the gravitational assumption that the liquid stream aligns along the body of container. This enables us to approximate the liquid's 3D shape with estimated container pose, providing valuable visual feedback for robot control.

Technically, we first collect two types of real-world data by robot or human executing pouring action with using empty or full container, and turning on or off the taps as shown in Fig. 2. The data is divided into positive samples if there is flowing liquid, otherwise negative samples. Then we train a classification network to distinguish the positive and negative samples, which drives the network to focus on the difference of the two types of data, *i.e.*, existence of liquid or not. Thus the derived Class Activation Map (CAM) will coarsely focus on the liquid regions. We further utilize this initial CAM to separate the feature maps into foreground and background local features. Then, we bridge the distance between foreground-foreground pairs while widening the distance between foreground-background pairs. This generates a better CAM which aligns with low-level boundaries and completely covers the target liquid region. Then, we use the off-the-shelf category-level pose estimation, *e.g.*, SAR-Net [17] to recover the pose of the source container for calculating the plane equation aligned with gravity and orientation of container's neck. Finally, the 3D point cloud of liquid is approximated by calculating the line-plane intersections, where the lines are the rays back-projected from estimated 2D liquid region.

In summary, the main contributions of this paper are: (1) We first propose a novel weakly-supervised pipeline to transfer the 2D liquid perception problem into a classification task. We also propose a feature contrast strategy to improve quality of CAM. (2) We first propose the method of approximate 3D shape of liquid by utilizing the 6-DoF pose of source containers and estimated liquid mask for robotic pouring-related task. To the best of our knowledge, we are the first work to perceive and model the 3D liquid from a single image without temporal information. (3) We deploy our real-time framework (10Hz) on the physical Franka robot to serve as visual feedback to pour liquid more accurately into target container. (4) We also propose the *PourIt!* dataset, which could be a test bed benchmark of the self/weakly-supervised liquid segmentation task for computer vision and robotics community. We believe that *PourIt!* framework should endow the ability for robots in continuous self-supervised learning. For example, extending the tasks in Fig. 1 to continuously make the robot mutually pour liquid from two containers to collect data in a self-supervised manner, thus fine-tuning the model using more data for better liquid perception.

2. Related Work

Liquid Perception. Accurately perceiving the liquid is challenging as the lack of fixed patterns of geometry shape.

The simulator [25, 24] that can synthesize the fluid is a good choice to generate an amount of labeled data to train a model. But the real-synthetic image domain gap usually degrades the performance of the synthetically-trained models. Thus our method directly uses the real-world data for training which avoids the so-called reality gap. Recent approaches [26, 27, 24] skillfully use the thermal camera aligned with the color image to generate the accurate ground-truth label for transparent liquid perception. However, the extra calibrated thermal sensor and heating water requirement are usually cost-consuming and human-labor intensive. Another line of works utilize the specialty of liquid motion, such as the optical flow [37] or the sound [15, 34, 36, 9, 18, 32] to perceive the liquid. For example, Yamaguchi *et al.* [37] adopt the optical flow of the liquid motion to perceive the water. However, this method demands no redundant motion when executing the pouring action, as the optical flow is sensitive to the background change, which will easily introduce the outlier pixels. Another works [15, 34] capture and analyze the cue of the audio vibration to estimate the weight or height of liquid in the container. These methods cannot be applied to perceiving the regions of liquid drop. Narasimhan *et al.* [20] first propose the self-supervised method by taking generative model CUT [22] to transform the color liquid into transparent one, further facilitating background subtraction for training the segmentation model. But it is only suitable for liquid detection inside static transparent containers. Our proposed method employs simple image-level labels to weakly supervise the model for perceiving and estimating the 3D shape of the regions of liquid drop. Thus, our proposed method can reduce the cost of either human-labor or equipment.

Weakly-supervised Semantic Segmentation. Weakly-supervised Semantic Segmentation (WSSS) usually uses coarse labels [3, 13, 11, 21, 4, 8, 16] for supervision but produces pixel-level localization maps. The image-level supervised WSSS approaches usually take the CAM technique [42, 28] to train a classification network, and thus generate the initial pseudo pixel-level labels for supervision. However, the primary objective of classification leads the CAM to focus solely on the most discriminative regions, rather than all parts of the objects. Some methods address this problem by using saliency maps [14, 29], erasing strategy [30, 33, 39, 5], or accumulation strategy [10, 12, 38] to help the CAM attend to more complete objects. Recent Transformer architecture has proven superior performance in semantic segmentation tasks [6, 35, 41]. In the field of WSSS, for example, AFA [23] exploits the semantic affinity from multi-head attention in Transformers and pixel-adaptive refinement that incorporates low-level image appearance information to refine the pseudo labels. However, accurately localizing transparent liquid still remains challenging due to non-salient visual cues of boundaries.

3. Methodology

Task overview. This paper focuses on perceiving dynamically flowing liquid when poured into the target container. This task is very challenging, as the target container maybe moving or its neck is narrow. Such a task requires real-time visual and control feedback to accurately control the liquid motion to avoid liquid spilling out of the target container.

Problem Formulation. Given a monocular RGB image $\mathcal{I} \in \mathbb{R}^{H \times W \times 3}$ of the pouring liquid, our goal is to estimate the pixel-wise segmentation $\mathcal{M} \in \mathbb{R}^{H \times W \times 1}$ of a stream of liquid which is poured out from the source container's neck. This paper supposes that the 6-DoF rigid pose transformation $\{\boldsymbol{R}; \boldsymbol{t}\} \in SE(3)$ of the source container is known beforehand with the 3-DoF rotation $\boldsymbol{R} \in SO(3)$ and 3-DoF translation $\boldsymbol{t} \in \mathbb{R}^3$. Due to the gravity, the direction of liquid is always kept consistent with the plane that aligned with the orientation of the container's neck and gravity. Thus, given the camera intrinsic parameters, we finally recover the approximate 3D point cloud $\boldsymbol{P}$ by using the pose of container $\{\boldsymbol{R}; \boldsymbol{t}\}$ and estimated liquid mask $\mathcal{M}$.

3.1. Data Collection

As there are no available mask annotations of liquid in our experiment, thus it is essential to guide the network to focus on the region of liquid drop. Inspired by the idea of weakly-supervised segmentation, we aim to collect data with image-level labels for liquid segmentation by utilizing the CAM technique. Lin *et al.* [17] tackled the task of pouring solid objects (beans) from the source into the target container. Using their method, we can collect much data of pouring actions. Then we design the semi-automatic pipeline to collect two kinds of samples.

Particularly, (1) we will first construct the N positive samples with the stream of liquid by the robot or human executing pouring action by using liquid-fulled container as in Fig. 2 (row 1). During the stage of the source container horizontal with the ground, we record and save the RGB frames where the liquid drop always exists. (2) Then, we replicate

Figure 2. The exemplar of our proposed *PourIt!* dataset. It includes images with liquid and non-liquid, totaling 3354 training images, 374 seen test images, and 336 unseen ones.
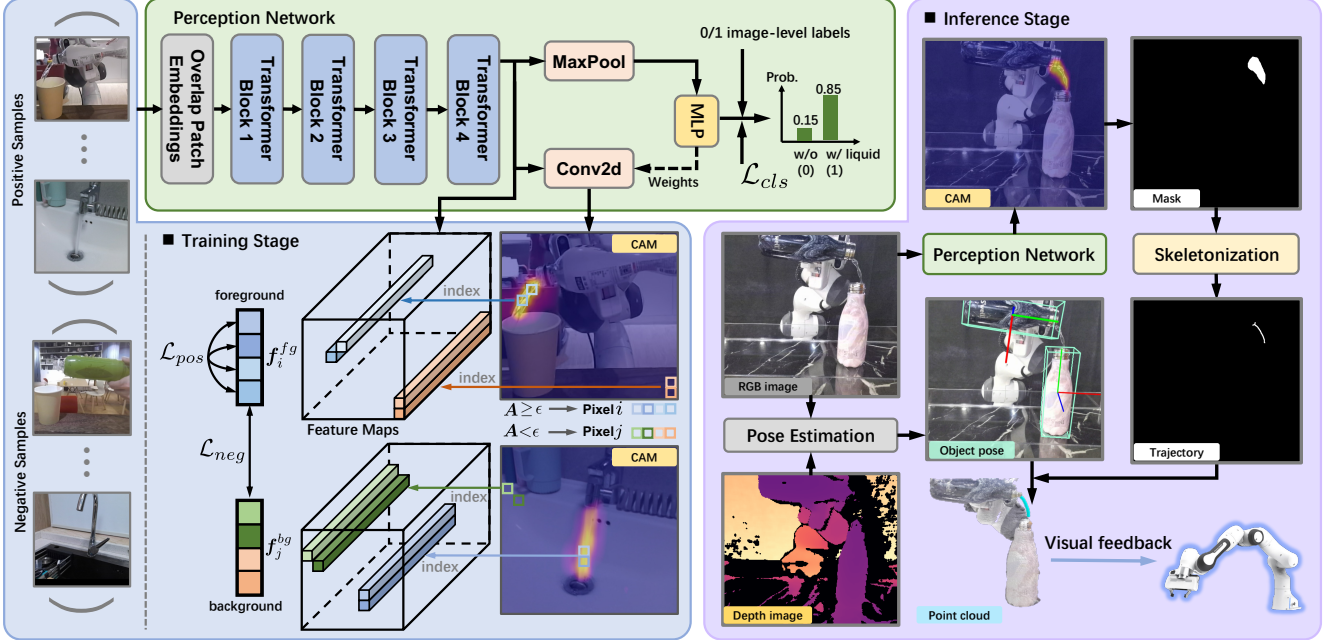
Figure 3. The workflow of our proposed *PourIt!* framework. Given the positive and negative samples of images with simple 0/1 image-level labels, we first use the Transformer backbone to extract features and the MLP layers to predict the image classes. Then, the derived CAM is used to index the localization of foreground and background pixels within the feature maps by using the threshold ϵ. In order to improve the quality of CAM, we force the network to pull close the foreground-foreground features while pulling apart the foreground-background ones. At the inference stage, we use a pose estimation network to recover the 6-DoF poses of source and target containers and a perception network to estimate the potential liquid mask from derived CAM. The trajectory of the liquid is then extracted according to the morphological shape of the mask. Finally, the point cloud of liquid is reconstructed by using the predicted 6-DoF object poses and 2D liquid trajectory, which provides real-time visual feedback of liquid-to-container distance as high-level guidance of robot control.

the same action but use the empty source container as in Fig. 2 (row 2), to collect the negative samples. We also add other images by turning on or off the tap for data augmentation Fig. 2 (column 3). Finally, To balance distribution of positive and negative samples, we individually sample 1677 images for each part.

3.2. Transformer Encoder

Our liquid perception framework uses Transformer as the backbone and an MLP layer to classify the images as in Fig. 3. Given an RGB image $\mathcal{I} \in \mathbb{R}^{H \times W \times 3}$, we first split it into multiple $h \times w$ patches of size 4×4. Then these patches are further flattened and linearly projected into $h \times w$ tokens. The Transformer layer comprises a multi-head self-attention block that outputs n attention maps. For each self-attention head, the patch tokens are projected into queries, $Q_i \in \mathbb{R}^{hw \times d_k}$, keys $K_i \in \mathbb{R}^{hw \times d_k}$, and values $V_i \in \mathbb{R}^{hw \times d_v}$, $i = 1, 2, \cdots, n$. Then the attention map is estimated by the scaled dot-product mechanism as follows,

$$Attention(Q_i, K_i, V_i) = Softmax(\frac{Q_i K_i^{\top}}{\sqrt{d_k}})V_i \quad (1)$$

where $\sqrt{d_k}$ is the scalar factor; d_k and d_v are the feature dimension of the keys and values, respectively. Then we

concatenate all attention maps and feed them into the feed-forward layers to obtain the feature maps. Similar to [23], we use four stacked Transformer blocks in our case, thus outputting the multi-level feature maps $\boldsymbol{F}^{(i)}, i = 1, 2, 3, 4$ at $\{\frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16}\}$ of the original image resolution.

3.3. CAM Generation

After obtaining the feature maps, we apply a pooling operation on features maps $\boldsymbol{F}^{(4)} \in \mathbb{R}^{h'w' \times d}$ that are output from the last transformer block, and send it into Multi-Layer Perception (MLP) layers for binary classification. We use the binary cross-entropy loss $\mathbb{BCE}(\cdot, \cdot)$ as the supervision,

$$\mathcal{L}_{cls} = \mathbb{BCE}(\hat{y}, y) \quad (2)$$

where $\hat{y}$ and y are the predicted and ground-truth image-level label. At the stage of inference, we generate the Class Activation Map (CAM) $\boldsymbol{A}^c$ belonging to the class c according to the weights w_j^c of the feature maps contributing to class c. We simplify the $\boldsymbol{A}^c$ as $\boldsymbol{A}$ in our binary classification task, denoted as,

$$\boldsymbol{A} = ReLU(\sum_j w_j \boldsymbol{F}_j^{(4)}) \quad (3)$$

where w_j is the element of weight matrix in the MLP layers, and $ReLU$ function removes the negative activation.

Finally, we apply the min-max normalization to scale the values of $\boldsymbol{A}$ within $(0, 1)$.

3.4. CAM Adjustment

However, the derived CAM above usually focuses on the part regions of the object, as the single label classification task does not enforce the model to attend to all parts of the object. Thus, our goal is to help the network to capture the regions with the same appearance or textures to tightly cover this object.

Feature Contrast. As CAM is the sum of weighted feature maps, it is intuitive to improve the quality of the feature maps to get a precision CAM. We observe that the derived CAM usually focuses on discriminative regions, which cannot cover the whole region of dropping liquid. Thus, we primarily use the contrastive strategy to supervise feature maps to force the network to gather foreground and background regions, respectively.

Give the activation map $\boldsymbol{A}$, we index the location i which makes $\boldsymbol{A} \geq \epsilon$ and retrieve the equal-sized feature maps $\boldsymbol{F}^{(4)}$ to get the foreground local features $\boldsymbol{f}_i^{fg} \in \mathbb{R}^{1 \times d}$ as in Fig. 3; d is the dimension of the feature maps $\boldsymbol{F}^{(4)}$. By analogy, the background local features are denoted as $\boldsymbol{f}_j^{bg} \in \mathbb{R}^{1 \times d}$, where $\boldsymbol{A} < \epsilon$. Then we aim to narrow the distance between foreground local features, and push apart the distance between foreground-background ones. We use the cosine similarity metric $sim(\cdot, \cdot)$ to measure the distance between each location of feature maps. For the positive contrastive loss, we aim to pull the features from similar regions together, supervised as,

$$\mathcal{L}_{pos} = -\frac{1}{m^2} \sum_{i=1}^{m} \sum_{i=1}^{m} log(sim(\boldsymbol{f}_i^{fg}, \boldsymbol{f}_i^{fg})) \qquad (4)$$

We also encourage the network to push apart the foreground and background features to obtain the more complete regions belonging to the actual flowing liquid.

$$\mathcal{L}_{neg} = -\frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} log(1 - sim(\boldsymbol{f}_i^{fg}, \boldsymbol{f}_j^{bg})) \qquad (5)$$

where m is the number of indexed local features $\boldsymbol{f}_i^{fg}$, and n is the number of $\boldsymbol{f}_j^{bg}$. Finally, the overall loss is the sum of $\mathcal{L}_{cls}$, $\mathcal{L}_{neg}$ and $\mathcal{L}_{pos}$, which is formulated as

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{pos} + \mathcal{L}_{neg} \qquad (6)$$

3.5. 3D Modeling of Flowing Liquid

We extract the final region mask $\mathcal{M}$ of liquid from the CAM using a threshold σ, where $\boldsymbol{A} \geq \sigma$. Due to the transparency of the liquid, the depth camera cannot well calculate the depth of liquid pixels. Thus, we further take the
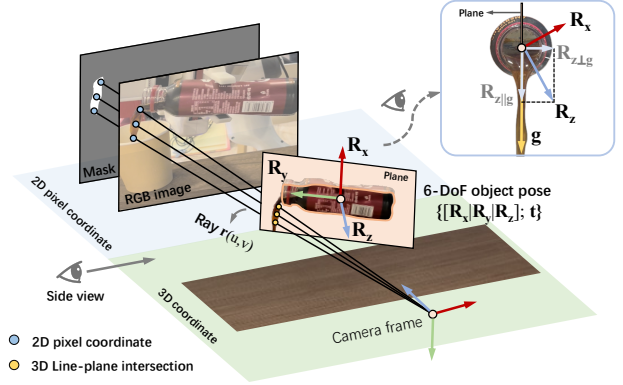

Figure 4. 3D modeling of the liquid. For better illustration, we exchange the position of 3D coordinate and 2d pixel coordinate.

container pose to model the approximate depth of the liquid as in Fig. 4. The pose $\{\boldsymbol{R} = [\boldsymbol{R}_x | \boldsymbol{R}_y | \boldsymbol{R}_z]; \boldsymbol{t}\}$ of source container in the camera frame is acquired by category-level pose estimator, *i.e.*, SAR-Net [17]. Then, the plane aligned with gravity and orientation of container's bottleneck (the green vector $\boldsymbol{R}_y$) can be formulated as,

$$\begin{cases} (\boldsymbol{p} - \boldsymbol{t})^\top \cdot \boldsymbol{R}_{z \perp g} = 0 \\ \boldsymbol{R}_z = \boldsymbol{R}_{z \perp g} + \boldsymbol{R}_{z \| g} \end{cases} \qquad (7)$$

where $\boldsymbol{p} = [x, y, z]^\top$ is the point on the plane, and $\boldsymbol{t}$ is the estimated 3D center of the source container in the camera frame. $\boldsymbol{R}_{z \perp g}$ and $\boldsymbol{R}_{z \| g}$ are the components of $\boldsymbol{R}_z$ that are vertical and parallel to gravity vector $\boldsymbol{g}$ in the camera frame, respectively. Given the camera intrinsic matrix $\boldsymbol{K}$, the discrete ray $\boldsymbol{r}(u, v)$ emitted from the camera is formulated as,

$$\begin{cases} \boldsymbol{r}(u, v) = \boldsymbol{o} + \kappa \boldsymbol{d}, \kappa \in \mathbb{R} \\ \boldsymbol{d} = \boldsymbol{K}^{-1} \cdot [u, v, 1]^\top \end{cases} \qquad (8)$$

where (u, v) is the pixel coordinate. κ is a scalar, which makes it possible to define any position along the ray. $\boldsymbol{o}$ and $\boldsymbol{d}$ indicate the origin and direction of the ray, respectively. Finally, we calculate the 3D intersection position $\boldsymbol{p}$ of the ray and the plane.

$$\boldsymbol{p} = \kappa \boldsymbol{d}, \kappa = \frac{\boldsymbol{t}^\top \cdot \boldsymbol{R}_{z \perp g}}{\boldsymbol{d}^\top \cdot \boldsymbol{R}_{z \perp g}} \qquad (9)$$

With this processing, we back-project 2D pixel (u, v) of liquid region from pixel coordinate into the actual 3D coordinate space. Finally, given the pixel (u, v) inside the mask $\mathcal{M}$, we can get the approximate 3D point cloud of the liquid $\boldsymbol{P} = \{\boldsymbol{p}(u, v) | (u, v) \in \mathcal{M}\}$. Actually, we post-process the mask $\mathcal{M}$ by using the skeletonization algorithm [40]. Although the 3D liquid modeling is based on gravitational assumption, further real-world robotics experiment in Tab. 4 shows that such an assumption is reasonable and robust enough to serve the robot as visual feedback for control.

4. Experiment

Dataset. *(1) PourIt! Dataset.* This dataset has 4064 RGB images recorded by a Kinect Azure camera with different backgrounds (Fig. 2). We partition them into 3354 training images and 710 testing images. The testing set has 374 images of familiar scenes and 336 images of new scenes. We labeled only the testing set for evaluation since annotating every liquid pixel is tough. It includes images of a person pouring liquid, a robot pouring liquid using pose estimation, and water flowing from various taps in a bathroom. It serves as a test bed for self-supervised methods. *(2) UW Liquid Pouring Dataset [27].* It contains 648 pouring trials on the dual-arm Baxter robot. The robot rotates its wrist to pour the liquid from the source container to the target one. Each trial involves controlling six variables: (1) left or right arm grasping the source container; (2) different source containers; (3) different target containers; (4) initial volume of liquid in the source container; (5) pouring trajectory; and (6) proportion of a person appearing in the background. We split the sequences of each scene into liquid and non-liquid images, center-cropped to 300×300 to remove irrelevant background. We finally sample 6387 training images and 1598 testing images. More details refers to our Appendix.

Evaluation Metric. To evaluate the segmentation accuracy of the segmented mask generated by different methods, we calculate the Intersection over Union (IoU) between the ground-truth segmentation labels and predicted ones and the mean IoU(%). For our method and EPS [14], we generate the segmented mask using threshold σ of 0.5 and 0.7, where $A \geq \sigma$. And the mIoU = $\frac{1}{2} \sum_{\sigma=0.5, 0.7}$ mIoU(σ).

Implementation Details. We use six 2080Ti GPUs for training with a batch size of 2 on each GPU. It takes 14000 iterations (nearly $1.5h$) to converge on the *PourIt!* dataset while 18000 iterations on the Liquid Pouring dataset (nearly $2h$). We use the AdamW optimizer with the initial learning rate of 6×10^{-5} for the backbone parameters. The learning rate of other parameters is ten times the parameters of backbone, *i.e.*, 6×10^{-4}. We also adopt the data augmentation with random re-scaling with a range of [1.0, 1.1], random horizontally flipping, and random cropping with a cropping size of 512×512, respectively. Our model is warmed up by only using $\mathcal{L}_{cls}$ for the early 2000 iterations to get an initial CAM. We set $\epsilon = 0.7$ in our experiment.

4.1. Comparison to Baseline

Baselines. Liquid [24] is the first supervised method for liquid segmentation, we implement this method as the upper bound to help compare performance. Then, we compare another three types of baseline methods, including (1) optical flow method (RAFT [31]), (2) image-to-image translation method (CUT [22]) and (3) CAM-based methods (EPS [14] and AFA [23]). Particularly, RAFT is used to detect pixel displacement between frames from the sequences of images

for liquid region detection as in Fig. 5 (column 1). Similar to [20], CUT is utilized to transfer the liquid images into non-liquid images as in Fig. 5 (column 2), and then use background subtraction algorithm to estimate the visual difference between the original and non-liquid translation images. AFA and our method all use the MiT-B1 backbones.

Results on PourIt! Dataset. We compare several baseline methods in our proposed *PourIt!* dataset and report the results in terms of mIoU in Tab. 1. We observe that our method has significantly outperforms the state-of-the-art WSSS method AFA, by a large margin of 10.1% on seen images and 10.8% on unseen images. This also validates the generalize ability of our proposed method. Although AFA uses the pixel-adaption refinement strategy, but the liquid is very different with the solid object which usually have salient boundaries. Such a strategy fails to get a better pseudo labels for segmentation training.

The RAFT, CUT and EPS methods cannot well handle the images with liquid on *PourIt!* dataset, thus we also provide the qualitative results to illustrate the failed cases as in Fig. 5. Concretely, RAFT [31] fails to segment the liquid due to the background movement. Thus, the optical flow method is only suitable for simple static scenes where the containers and background are relatively stationary. Nevertheless, the translation images from CUT method [22] have poor results due to the various contextual information across images. It simultaneously erases the foreground but also unexpectedly modifies the background context, making calculating the mask of the liquid region unavailable. EPS [14] has poor performance as the saliency maps cannot focus on the actual liquid region for supervision.

Results on UW Liquid Pouring Dataset. The comparison results on this dataset are summarized in Tab. 2 and Fig. 5. Our method achieves 54.0% mIoU on the test set, and surpasses AFA method by 4.6%. The RAFT and CUT methods are still failed in this dataset, by analogy to the aforementioned analyse of the failed cases in *PourIt!* dataset. We also adopt the fully-supervised method Liquid [24] as the oracle approach, thus to help us understand the upper bound of the performance of liquid segmentation.

Discussion on limitations of UW Liquid Pouring dataset. The annotations in this dataset contain liquid out-flowed from bottleneck, and liquid stayed in the container. While the original purpose of the dataset [26, 27] was to estimate the volume of liquid to be poured into the container, we thus provide experimental results that serve as a test bed to validate our proposed method. We acknowledge that the model trained on this dataset cannot directly apply to control the real-world robot for accurate pouring, but our approach presents a valuable contribution to the field. Please refer to Appendix for more details.

Qualitative Results. We also compare our CAM results with the baseline on two datasets as shown in Fig. 6. Ob-

Table 1. Segmentation results on *PourIt!* dataset. We report the performance between our method and the baseline methods in metric of mIoU(%). *Sup.* denotes supervision type. $\mathcal{I}$: image-level labels. $\mathcal{S}$: saliency maps. † denotes our implementation.

Method	Sup.	Backbone	seen	unseen
RAFT [31] ECCV'20	–	ResNet	9.4	10.8
CUT† [22] ECCV'20	$\mathcal{I}$	ResNet	38.1	24.3
EPS† [14] CVPR'21	$\mathcal{I}+\mathcal{S}$	ResNet38	47.2	42.0
AFA† [23] CVPR'22	$\mathcal{I}$	MiT-B1	55.8	52.8
Ours	$\mathcal{I}$	MiT-B1	**65.9**	**63.6**

Table 2. Segmentation results on Liquid dataset. We report performance between ours and baseline methods in metric of mIoU(%). *Sup.* denotes supervision type. $\mathcal{F}$: full supervision; $\mathcal{I}$: image-level labels. $\mathcal{S}$: saliency maps. † denotes our implementation.

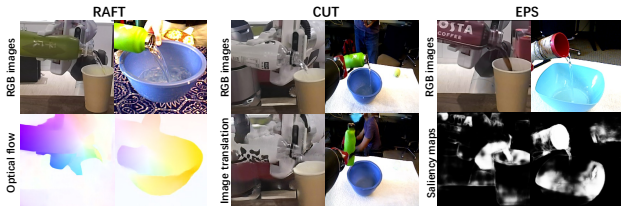| Method	Sup.	Backbone	seen
Liquid† [24] Upper bound	$\mathcal{F}$	FCN	69.0
RAFT [31] ECCV'20	–	ResNet	5.5
CUT† [22] ECCV'20	$\mathcal{I}$	ResNet	38.2
EPS† [14] CVPR'21	$\mathcal{I}+\mathcal{S}$	ResNet38	41.6
AFA† [23] CVPR'22	$\mathcal{I}$	MiT-B1	49.4
Ours	$\mathcal{I}$	MiT-B1	**54.0**

Figure 5. Qualitative intermediate results of CUT, RAFT and EPS methods. For RAFT method, per-pixel displacement vector is colorized in different colors (left). For CUT, we show the image-to-image translation results of transferring the liquid images into non-liquid ones (center). For EPS, we visualize the saliency maps used for supervision (right).

viously, our method outputs more accurate CAM covering the liquid regions, especially for the thin stream of liquid. The results on novel scenes show that our method is robustly generalized to images with different kinds of liquid, containers, and background contexts.

4.2. Ablation Study and Analysis

We perform ablation studies to validate the contribution of each loss employed in our method. We conduct a comparison between the full model and its variants, wherein we disable the losses $\mathcal{L}_{pos}$, $\mathcal{L}_{neg}$, and both of them together $\mathcal{L}_{pos+neg}$. We test these variants on the *PourIt!* dataset. We summarize the results in Tab. 3.

Even without $\mathcal{L}_{pos}$ and $\mathcal{L}_{neg}$, the results outperform the state-of-the-art method AFA [23]. This is because the pixel-adaptation refinement strategy in AFA does not help and instead reduces the performance due to non-salient visual

Table 3. Ablation study for different losses on PourIt! dataset under the metric of mIoU(%).

Variants	w/o $\mathcal{L}_{neg+pos}$	w/o $\mathcal{L}_{neg}$	w/o $\mathcal{L}_{pos}$	Full
seen	61.6	46.7	60.6	**65.9**
unseen	59.2	36.7	63.2	**63.6**

cues of boundaries. It is evident that $\mathcal{L}_{neg}$ is the most significant among them, as it helps the network better distinguish the foreground and background regions. It also helps the network generalize to different liquids, containers and backgrounds, thus maintaining the performance, *i.e.*, 63.2% on unseen images. Interestingly, the single use of $\mathcal{L}_{pos}$ negatively influences the performance of the network. One explanation is that the highly responsive area of the imperfect initial CAM may contains foreground and background pixels; and only using $\mathcal{L}_{pos}$ will guide the network to be improperly optimized to pull together the actual foreground and background features. This can lead to confusion when distinguishing the foreground and background pixels, especially reducing the overall performance, with the results of 46.7% on seen images and 36.7% on unseen images.

Especially, the combination of $\mathcal{L}_{pos}$ and $\mathcal{L}_{neg}$ encourage the network to differentiate clear partition between the foreground and background pixels, even though the initial regions that CAM puts focus on may contain some pixels from background. The final results reveal that our well-designed losses benefit network learning and help boost the final performance.

4.3. Generalization Analysis

For training images in *PourIt!* dataset, we use three types of source and target containers, respectively. The liquid contains *transparent water*, *coffee*, and *tea*. Furthermore, we use another three types of source and target containers for unseen test images. The liquid contains *milk*, *juice of grape*, and *melon*. Surprisingly, our trained model could be naturally generalized well across images of different liquids, containers, and even the background as in Fig. 6 (column 3). This verifies that our model is well-trained to learn the category-level feature representations of liquid with different appearances. Thus, this category-level generalization ability further endows the feasibility of our model in different real-world scenes.

4.4. Robotics Experiments

Hardware Experimental Setup. We use a Franka-Emika Panda 7-DoF robotic arm [2] with a parallel jaw gripper to perform object grasping and manipulation for pouring tasks. The Azure Kinect camera [1] provides RGB-D sensing, which is mounted on the tripod opposite the robot's workspace. This camera is also calibrated to the robot's base frame. In our experiment, three desktop computers were utilized: one computer for the real-time control of
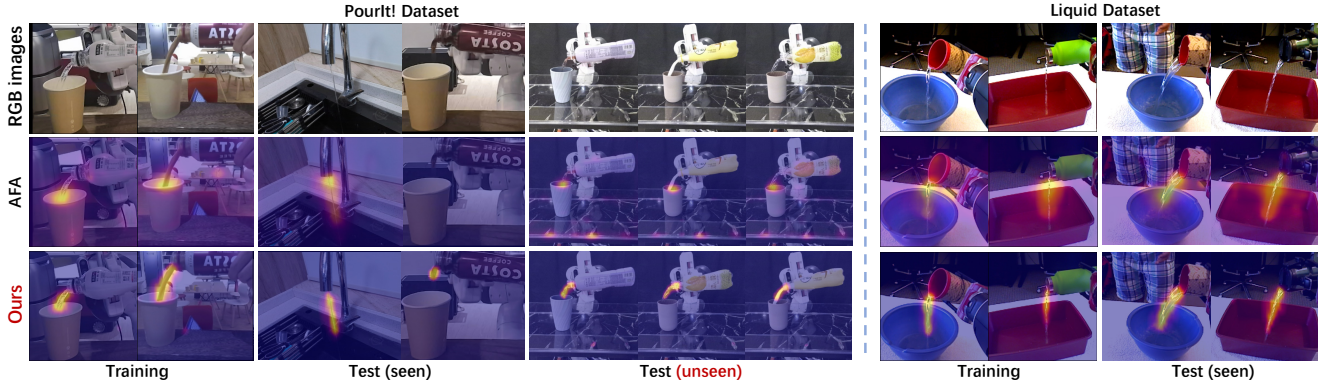
Figure 6. CAM visualization of different methods. Note that the quality of AFA's CAM is similar to its segmented mask results, thus we only visualize CAM but not AFA segmented masks [23] for consistency of visual contrast. The comparison results show that our CAM results focus more tightly on complete liquid region, with well alignment of low-level boundaries and generalization across novel scenes.

Franka, two for real-time visual perception, *i.e.*, the pose estimation network is distributed on a laptop with an NVIDIA RTX 3070 GPU which runs at 20Hz per object, and the liquid perception network is deployed on a desktop with an NVIDIA RTX 2070 GPU which runs at 15Hz.

Task Description. We design two kinds of tasks for robotic pouring, including static scenes of pouring liquid into a stationary positioned target container and dynamic scenes of pouring liquid into a moving target container. For the moving target container, the experimenter will hold the container in hand with linear or random motion. We use six groups of different source and target containers for testing.

Baselines. We primarily compare with two baseline methods. (1) Lin *et al.* [17]. It uses the estimated poses and sizes of source and target containers to calculate the initial pouring point, *i.e.*, the container-to-container distance. (2) AFA [23]. This baseline utilizes Lin *et al.* [17] to determine the initial pouring point and then use the AFA [23] provides 2D perception results, which are then lifted into 3D liquid by using the modeling strategy (discussed in Sec. 3.5) for visual feedback.

Evaluation Metric. We perform 15 attempts for each scene type and tally the overall successful outcomes. Both the baseline methods and our approach adopt the technique from Lin *et al.* [17] to compute initial pouring points. Nonetheless, these initial points can lead to imperfections, leading to liquid spillage as demonstrated in our supplementary video. In such situations, our method and AFA are still considered successful if it manages to rectify the initial imperfection by adjusting the source container, allowing the liquid to be accurately poured into the target container and maintaining this state until the pouring process concludes.

Control Strategy. During pouring, the robot fine-tunes the source container's position on its gripper to precisely regulate liquid outflow into the target container. By measuring the distance error between the target container's center and the liquid endpoint, we feed this error to the PID controller for calculating gripper adjustments. This ensures the liquid's endpoint consistently aligns with the target container's center, maintaining accurate pouring. Further insights are available in our supplemental video.

Results and Analysis. We report the success rate of the pouring tasks in Tab. 4, which proves the efficacy of our method. The method of Lin *et al.* [17] only uses the estimated object pose and size for calculating container-to-container distance as the initial pouring point. Such a strategy sometimes fails due to an imperfect initial pouring point, as the transparent bottleneck results in inaccurate object size estimation. Our method utilizes only object pose information to compute the liquid-to-container distance, providing feedback for adjusting the pouring point, which is less affected by inaccurate size estimation. Additionally, it meets the requirement for the pouring point to adapt dynamically in response to liquid levels within the source container. However, the AFA method performs less effectively than [17] due to the imperfect 2D mask leading to incorrect visual feedback for adjusting the pouring point, resulting in liquid spillage from the target container. Figure 7 demonstrates the visualization of intermediate results from the experiments. Significantly, we recover the 3D point cloud of liquid due to the unavailable depth measurement for transparent liquid. We observe that the depth of milk liquid is still roughly measurable, and our reconstructed point cloud (cyan points in column 3) is aligned well with the shape recovered from the sensor-measured depth. This verifies that our method can accurately reconstruct the shape of a stream of liquid.

Generalization to other applications. Our approach excels in liquid-pouring tasks, as validated in this paper, and demonstrates tremendous potential for seamlessly perceiving 3D liquid from taps when robots are filling containers in both kitchen and plant watering scenarios. With the continuous advancement of pose estimation technology, obtaining the tap's pose has become effortlessly achievable using

Table 4. Success rate of robotics pouring. 'Static' indicates that the target container is placed statically on the desktop. 'Dym.(L)' means the target container is held on experimenter's hand moving by linear motion, and 'Dym.(R)' denotes the random motion.

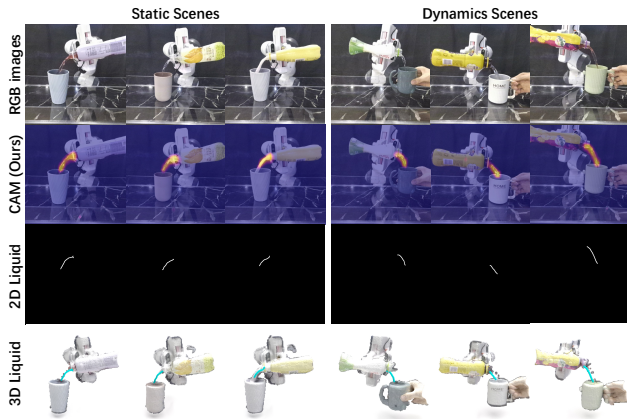| Scene Method	Success Rate (%)		
	$Static$	$Dym.(L)$	$Dym.(R)$
Lin *et al.* [17]	66.7	53.3	40.0
AFA [23]	60.0	46.7	33.3
Ours	93.3	73.3	60.0

Figure 7. Visualization of the output results from *PourIt!* framework. We show the RGB images, predicted CAM, skeletonized 2D liquid trajectory and estimated 3D (cyan) point cloud of liquid.

off-the-shelf solutions. This significant development further enhances the practicality of our approach in a wide array of liquid-filling scenarios. Its versatility proves to be highly valuable for various liquid-filling applications.

Discussion on Uncertainty of Results. Liquid perception provides the feedback signal to the robot. Thus, the uncertainties of the perception results are important for robot controller. Reducing uncertainty in the feedback signal mitigates error triggers during robotic action. In our experiment, the main uncertainties in the final 3D perception stem from 2D segmentation and pose estimation. We minimize the impact of 2D segmentation uncertainty using a high threshold $\sigma = 0.7$ when extracting mask from CAM. The uncertainty arsing from pose estimation is reduced to an acceptable level, thanks to the accurate results provided by SAR-Net [17].

4.5. Failure Mode Analysis

This paper presents a novel approach for accurate liquid perception and 3D shape recovery for robotic pouring tasks. However, the method relies on the pose of source containers to calculate the approximate depth of detected 2D liquid. We demonstrate a scenario in which our method may encounter challenges. When dealing with transparent containers, the depth of the source container's surface becomes immeasurable, as depicted in Fig. 8. Only a fraction of the ob-
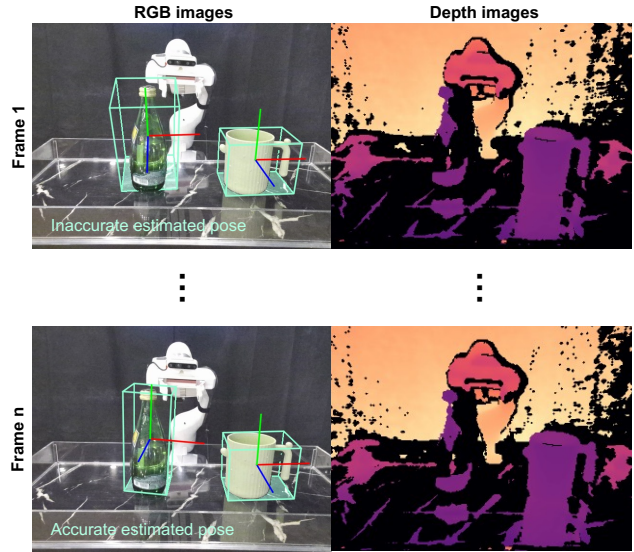

Figure 8. Jitter in the estimated object pose caused by the transparent surface. The object's transparent surface leads to inaccurate depth measurements, resulting in jitter of the estimated pose.

servable surface is accessible, posing a significant challenge for depth-based pose estimators. While SAR-Net exhibits a commendable degree of robustness to this challenge, it does introduce pose jittering in certain frames. Thus, the inaccuracy in pose estimation for transparent objects can adversely affect the modeling of the liquid. Actually, pose estimation for transparent objects remains a highly challenging task, not only in our specific case but also in the broader research field of pose estimation. Additionally, the category-level pose estimator we employed typically yields an estimated error within 1cm during our experiments, making the precision of our system closely tied to the accuracy of object pose estimation. However, compared to previous methods that rely on both object pose and size, our approach primarily leverages object pose, reducing its susceptibility to inaccurate container size.

5. Conclusion

This paper introduces *PourIt!*, a simple and effective framework for visual closed-loop robotic pouring. It utilizes image-level labels and CAM technique for liquid segmentation, enhanced by a feature contrast strategy to better localize potential liquid regions. The container's pose facilitates 3D point cloud recovery of the liquid, and liquid-to-container distance is used as feedback signal for robot control. Future work will explore the application of the recovered 3D liquid shape in various pouring-related tasks.

References

[1] Azure Kinect DK. `https://azure.microsoft.com/en-us/products/kinect-dk`. 7

[2] Franka Emika Panda. `https://www.franka.de`. 7

[3] Jiwoon Ahn and Suha Kwak. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4981–4990, 2018. 3

[4] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*, pages 549–565. Springer, 2016. 3

[5] Zhaozheng Chen, Tan Wang, Xiongwei Wu, Xian-Sheng Hua, Hanwang Zhang, and Qianru Sun. Class re-activation maps for weakly-supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 969–978, 2022. 3

[6] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in Neural Information Processing Systems*, 34:17864–17875, 2021. 3

[7] Sachin Chitta, Ioan Sucan, and Steve Cousins. Moveit![ros topics]. *IEEE Robotics & Automation Magazine*, 19(1):18–19, 2012. 2

[8] Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1635–1643, 2015. 3

[9] Reina Ishikawa, Yuichi Nagao, Ryo Hachiuma, and Hideo Saito. Audio-visual hybrid approach for filling mass estimation. In *International Conference on Pattern Recognition*, pages 437–450. Springer, 2021. 2, 3

[10] Peng-Tao Jiang, Qibin Hou, Yang Cao, Ming-Ming Cheng, Yunchao Wei, and Hong-Kai Xiong. Integral object mining via online attention accumulation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2070–2079, 2019. 3

[11] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 876–885, 2017. 3

[12] Beomyoung Kim, Sangeun Han, and Junmo Kim. Discriminative region suppression for weakly-supervised semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1754–1761, 2021. 3

[13] Alexander Kolesnikov and Christoph H Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 695–711. Springer, 2016. 3

[14] Seungho Lee, Minhyun Lee, Jongwuk Lee, and Hyunjung Shim. Railroad is not a train: Saliency as pseudo-pixel supervision for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5495–5505, 2021. 3, 6, 7

[15] Hongzhuo Liang, Shuang Li, Xiaojian Ma, Norman Hendrich, Timo Gerkmann, Fuchun Sun, and Jianwei Zhang. Making sense of audio vibration for liquid height estimation in robotic pouring. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5333–5339. IEEE, 2019. 2, 3

[16] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3159–3167, 2016. 3

[17] Haitao Lin, Zichang Liu, Chilam Cheang, Yanwei Fu, Guodong Guo, and Xiangyang Xue. Sar-net: Shape alignment and recovery network for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2022. 1, 2, 3, 5, 8, 9

[18] Qi Liu, Fan Feng, Chuanlin Lan, and Rosa HM Chan. Va2mass: Towards the fluid filling mass estimation via integration of vision and audio learning. In *International Conference on Pattern Recognition*, pages 451–463. Springer, 2021. 2, 3

[19] Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004. 2

[20] Gautham Narasimhan, Kai Zhang, Ben Eisner, Xingyu Lin, and David Held. Self-supervised transparent liquid segmentation for robotic pouring. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4555–4561. IEEE, 2022. 2, 3, 6

[21] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1742–1750, 2015. 3

[22] Taesung Park, Alexei A Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 319–345. Springer, 2020. 3, 6, 7

[23] Lixiang Ru, Yibing Zhan, Baosheng Yu, and Bo Du. Learning affinity from attention: end-to-end weakly-supervised semantic segmentation with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16846–16855, 2022. 3, 4, 6, 7, 8, 9

[24] Connor Schenck and Dieter Fox. Towards learning to perceive and reason about liquids. In *International Symposium on Experimental Robotics*, pages 488–501. Springer, 2016. 3, 6, 7

[25] Connor Schenck and Dieter Fox. Reasoning about liquids via closed-loop simulation. *arXiv preprint arXiv:1703.01656*, 2017. 3

[26] Connor Schenck and Dieter Fox. Visual closed-loop control for pouring liquids. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2629–2636. IEEE, 2017. 2, 3, 6

[27] Connor Schenck and Dieter Fox. Perceiving and reasoning about liquids using fully convolutional networks. *The International Journal of Robotics Research*, 37(4-5):452–471, 2018. 3, 6

[28] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 3

[29] Fengdong Sun and Wenhui Li. Saliency guided deep network for weakly-supervised image segmentation. *Pattern Recognition Letters*, 120:62–68, 2019. 3

[30] Kunyang Sun, Haoqing Shi, Zhengming Zhang, and Yongming Huang. Ecs-net: Improving weakly supervised semantic segmentation by using connections between class activation maps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7283–7292, 2021. 3

[31] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 6, 7

[32] Hengyi Wang, Chaoran Zhu, Ziyin Ma, and Changjae Oh. Improving generalization of deep networks for estimating physical properties of containers and fillings. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 9147–9151. IEEE, 2022. 2, 3

[33] Yunchao Wei, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1568–1576, 2017. 3

[34] Justin Wilson, Auston Sterling, and Ming C Lin. Analyzing liquid pouring sequences via audio-visual neural networks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7702–7709. IEEE, 2019. 2, 3

[35] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021. 3

[36] Alessio Xompero, Yik Lung Pang, Timothy Patten, Ahalya Prabhakar, Berk Calli, and Andrea Cavallaro. Audiovisual object classification for human-robot collaboration. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 9137–9141. IEEE, 2022. 2, 3

[37] Akihiko Yamaguchi and Christopher G Atkeson. Stereo vision of liquid and particle flow for robot pouring. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1173–1180. IEEE, 2016. 3

[38] Yazhou Yao, Tao Chen, Guo-Sen Xie, Chuanyi Zhang, Fumin Shen, Qi Wu, Zhenmin Tang, and Jian Zhang. Non-salient region object mining for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2623–2632, 2021. 3

[39] Fei Zhang, Chaochen Gu, Chenyue Zhang, and Yuchao Dai. Complementary patch for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7242–7251, 2021. 3

[40] Tongjie Y Zhang and Ching Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984. 5

[41] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021. 3

[42] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 3