IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 5 January 2023, accepted 30 January 2023, date of publication 13 February 2023, date of current version 23 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3244552

RESEARCH ARTICLE

In-Hand Pose Estimation Using Hand-Mounted RGB Cameras and Visuotactile Sensors

YUAN GAO[1], SHOGO MATSUOKA[1], WEIWEI WAN[1], (Senior Member, IEEE), TAKUYA KIYOKAWA[1], (Member, IEEE), KEISUKE KOYAMA[1], AND KENSUKE HARADA[1,2], (Fellow, IEEE)

[1]Department of Systems Innovation, Graduate School of Engineering Science, Osaka University, Toyonaka 560-8531, Japan
[2]Automation Research Team, Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology, Tokyo 100-8921, Japan

Corresponding author: Weiwei Wan (wan.weiwei.es@osaka-u.ac.jp)

ABSTRACT This paper proposes a method to estimate the 6D pose of an object grasped by a robot hand using RGB cameras mounted on the palm and visuotactile sensors installed at the fingertips. It can deal with objects made from a wide range of materials thanks to combining the two types of sensors. The method allows a robot to robot to perform in-hand pose estimation while holding the object, eliminating the need for preparatory actions or particular environmental backgrounds. The mechanism at the back of the method includes deep-learning-based background subtraction and denoising auto-encoder-based sensor fusion. With the poses estimated using the proposed method, a robot controller can rectify the grasping uncertainty and adjust the robot motion to move an object toward required goals with satisfying accuracy. We conduct various studies and analyses in the experimental section to understand the proposed method's advantages and disadvantages. The results demonstrate the benefits of the proposed combination and mechanism. They also provide essential knowledge to readers considering using a similar configuration for estimating object poses.

INDEX TERMS In-hand pose estimation, visuotactile sensors, robotic manipulation.

I. INTRODUCTION

A critical issue in robotic manipulation planning is that a robot cannot grasp an object in the same pose assumed in simulation. Uncertainty happens during the physical interactions between an object and the robot hand, making the in-hand object pose after grasping different from expectation. Previously, many researchers in robotics have studied using in-hand pose estimation to solve the problem. The goal was to estimate the pose of an object in the robot hand that grasped it. For example, Bimbo et al. [1] used tactile sensors to collect pressure data and computed the co-variance and eigenbasis of the pressure to search for in-hand object poses. Bauza et al. [2] used a visuotactile sensor to measure the contact surface between a finger pad and an object and used a neural network to predict the in-hand pose considering a dense set of contact shapes. Wen et al. [3] used depth cameras mounted on the body of a dual-arm robot to estimate in-hand poses while considering detecting and removing the point cloud of the holding hand. These previous works showed that estimating an object's pose with a single type of sensor was possible. They also revealed that the estimation performance depended on the availability and limitations of particular sensor types. The tactile sensors are applicable only when the grasped object has significant features at the finger contact surface. The depth sensors have a minimum visible range and must be installed far away from target objects. Their performance is also related to the object's surface materials. Considering these problems, we propose combining RGB and visuotactile sensors to make up for the shortages of each single sensor type in this paper. We develop a deep-learning-based background subtraction and a denoising auto-encoder-based sensor fusion method to fuse respective sensors for in-hand 6D object pose estimation.

In detail, our proposed method uses data collected from two RGB cameras mounted on the palm of a robot hand, and two GelSight [4] sensors installed at the fingertip to estimate object poses. The configurations of the cameras

The associate editor coordinating the review of this manuscript and approving it for publication was Yangmin Li.

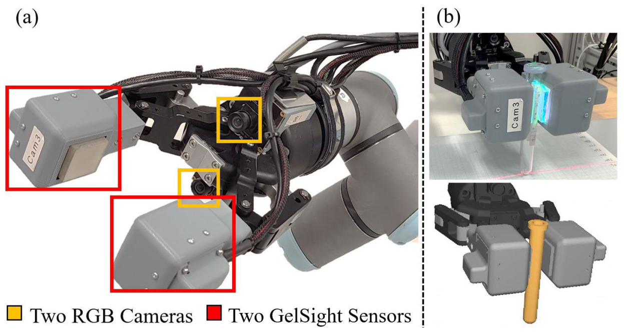


FIGURE 1. Using two RGB cameras mounted on the hand palm and two GelSight sensors installed at the fingertips to estimate in-hand object poses. The method applies to many objects regardless of their texture, optical properties, shapes, and changes in the environmental background. (a) Sensor configuration. (b) Estimating the pose of a crystal test tube.

and GelSight sensors are illustrated in Fig. 1. The RGB cameras and image processing apply to objects made from a wide range of materials but have less flexibility in obtaining depth and rotation information. In contrast, tactile sensors have advantages in recognizing a grasped object's contact penetration and local rotation. By combining them, we can estimate the in-hand pose for various objects. The performance is irrelevant to surface texture, optical properties (transparency, reflectivity), and shapes.

The main contributions of this work are as follows. First, we employ a deep learning-based background subtraction method to separate the hand, hand-held objects, and surrounding environment in the RGB cameras' views. Although the tactile sensors always appear at particular portions in the frames captured by the RGB cameras, an object may be grasped in different poses, and the environment background will be different. Extracting exact object regions and using them for pose estimation is challenging. We propose mixing MiDaS (Mixing Datasets for zero-shot cross-dataset transfer) [5] and RAFT (Recurrent All-Pairs Field Transforms) [6] to get reliable performance. The MiDaS method enables estimating the depth information from a single RGB image. The RAFT method allows the optical flow estimation during the hand motion and differentiates between the stationary and moving image pixels. They jointly permit segmenting the hand, hand-held objects, and surrounding environment in the RGB cameras' views. Second, we send the segmented hand and hand-held object image regions to an auto-encoder trained using simulated RGB and visuotactile data to extract features. We use cosine similarity to compare the extracted features with a dense reference set prepared in the same way (but with simulated grasping configurations) and thus obtain predicted in-hand poses.

We conduct multiple studies and analyses to understand the proposed method's advantages and disadvantages. In the experimental section, we examine the precision and flexibility of the proposed method by using markers and objects with various surface properties. The results demonstrate the benefits of the proposed combination and mechanism. With their support, the proposed method can estimate an object's in-hand pose regardless of its texture, optical properties, shapes, and changes in the environmental background. The experiments and analyses also provide essential knowledge to readers considering using a similar configuration for estimating object poses.

The remaining sections of the paper are organized as follows. First, we present related work in Section II and clarify the novelty of our study. Then, we give an overview of the proposed method in Section III to provide a high-level summary of the workflow. After that, we show data processing, training, and estimation details in Sections IV, V, and VI. Experiments and analyses are carried out in Section VII. Section VIII draws conclusions and discusses future work.

II. RELATED WORK

In this section, we will review the related studies according to the different sensor types used for in-hand pose estimation. Note that our focus is on in-hand pose estimation. Broader pose estimation methods like [7], [8] are out of the scope.

A. VISION-BASED METHODS

The vision-based methods use RGB or depth cameras to detect and estimate in-hand object poses. For example, Mohammed et al. [9] used AR (Augmented Reality) markers attached to the object to simplify estimation and implemented real-time in-hand pose tracking for assembly. Kokic et al. [10] estimated the pose of an object grasped by a human hand using neural networks trained with synthesized RGB image data. Like Kokic et al., Hasson et al. [11] estimated the object pose grasped by a human hand by combining CNN (Convolutional Neural Network) and optical flow. Hasson's method does not require synthesizing much training data thanks to optical flow analyses and photometric consistency-based background subtraction. Wen et al. [3] estimated in-hand poses of a hand-held object by matching the object's model to point clouds obtained from a depth sensor mounted on the robot body. The robot hand was adaptive, and its finger positions cannot be precisely obtained from motor encoder values. For this reason, a hand state estimation method was developed to remove the fingers from the depth sensor's view correctly. Liu et al. [12] developed a similar depth sensor-based method to estimate the pose of a pen held by a robot gripper and then used the estimated pose to update previously planned or optimized drawing trajectories. Although using depth sensors for in-hand pose estimation has high fidelity, it retains a problem that the depth sensors must be installed far away from the objects. Visual obstructions may frequently happen due to the installation. They are challenging to be avoided.

The above vision-based methods were good at estimating large and well-textured objects. However, they had difficulties extending to objects with various surface textures, optical properties, and shapes since the observation suffered from transparency, reflection, and occlusions. In this work, we use a similar optical and photometric method to avoid synthesizing a large amount of data for training pose recognition neural networks [13], [14]. We employ deep-learning-based background subtraction and binarization to avoid influences from objects' surface properties.

B. TACTILE-BASED METHODS

For the tactile sensor-based methods, Bimbo et al. [15] used multi-fingered hands and F/T sensing-based tactile sensors at the fingertips to re-estimate in-hand object poses. The initial object pose was obtained using a vision sensor before grasping, and the objects had spherical, cylindrical, or rectangular shapes. The same group also estimated the in-hand object pose by comparing the geometry pre-annotated based on the object model and the tactile information obtained from pressure sensors embedded on a Barrett robot hand [1]. Li et al. [16], and Cui et al. [17] used a GelSight sensor to estimate and adjust various workpieces for successful insertion. Pirozzi et al. [18] used a photo reflector array to measure the deformation of soft finger pads and estimate the in-hand pose of electric wires. Koyama et al. [19] developed a proximity sensor-based tactile fingertip to detect the tilt and distance of in-hand or near-hand objects. Hogan et al. [20] used an ABB Yumi robot with two GelSlim sensors [21] installed at each arm's tool center point to manipulate cubic objects and estimate their poses simultaneously. They considered mixed prehensile and non-prehensile grasping to accord with the large sizes. Bauza et al. [22] estimated the in-hand pose of an object by matching the geometry information obtained offline using the GelSlim sensor. The same authors also proposed an alternative method to estimate the in-hand object pose by comparing the GelSlim data with simulated images [2], [23]. She et al. [24] used GelSight to estimate the pose of a cable in the grip and the friction forces during cable sliding. They developed methods to adjust the gripping power and pose in real time according to the estimation results. Yamaguchi and Atkeson [25] used Finger Vision to measure tactile forces, and control knife poses for successful cutting. Liang et al. [26] tracked the object pose in hand by using tactile information obtained from BioTac touch sensors installed on the four fingertips of an Allegro robot hand. They compared the tactile information with the values obtained in a continuously updating physical simulator to optimize prediction. Kuppuswamy et al. [27] used a gripper with Soft-Buble fingertips to measure 3D contact point clouds and estimate the in-hand object poses. The method could help estimate the in-hand pose of crystal wine glasses. Lambeta et al. [28] used a multi-finger hand with tactile sensors on each fingertip to detect the in-hand marble positions and perform model predictive control. Tian et al. [29] presented a similar predictive control method for rolling a 20-sided die. The work carried out the learning in an end-to-end style instead of explicit pose estimation.

The above tactile-only methods were reliable, but they assumed that the grasped objects had significant features at the finger contact surface, and the contact information is enough to imply a whole object pose. As discussed in the following subsection, the assumption led to limitations and might be secured by fusing tactile data with other sensor values.

Besides the above tactile-based work, there are also studies that used geometric constraints and a series of contact to estimate in-hand object poses [30], [31], [32], [33], [34], [35]. The methods were clever but required multiple or continuous robotic actuation.

C. METHODS USING MULTIPLE SENSORS

Some studies estimate in-hand object poses by fusing multiple sensor information. For example, Herbert et al. [36] fused multiple data from F/T sensors at the wrist, joint sensors in the fingers, and stereo vision sensors in the environment to estimate the pose of a grasped object. Izatt et al. [37] presented an object-tracking method by fusing the point cloud information from an external RGB-D camera with GelSight. Pfanne et al. [38] fused joint measurements and visual features to estimate the in-hand object pose. Li et al. [39] combined RGB cameras and resistive tactile sensors [40] to estimate the pose and, thus, kinematics of articulated tools. Dikhale et al. [41] combined external RGB-D data and tactile data to estimate 6D object poses in hand. Combining different sensor data is beneficial compared to a single one. Suresh et al. [42] employed an overlooking depth camera to initialize observation and incrementally built an object's 3D shape through multiple touches. Anzai and Takahashi [43] used a hand-mounted RGB camera and a GelSight sensor attached to the robot hand to estimate changes in the grasping posture from an initial position. Gao et al. [44] built a multisensory database for versatile object recognition. Chaudhury et al. [45] presented a similar system to this work. The authors used two cameras and one Gelsight sensor to locate objects. The objects are on a table rather than grasped by a robot hand. Fusing RGB (vision) and GelSight (visuotactile) data using a deep neural network trained with simulated data is quite new, as it was not until recently that the simulation techniques were developed. Especially for the GelSight sensors, Sferrazza et al. [46] proposed a method to express the image from the GelSight sensor in the simulation by combining the Finite Element Methods (FEM) and optical flow. Agarwal et al. [47] used a physical rendering simulator to generate simulated GelSight views. Gomes et al. [48] presented a sim-to-real learning method to simulate GelSight sensors. Lee et al. [49] developed a deep-learning method to interconvert the GelSight sensor's photometric and marker images.

This study uses hand-mounted RGB cameras and finger-equipped GelSight sensors to estimate in-hand object poses. Unlike RGB-D sensors, our sensor setting does not require long sensing distances. We can install cameras and GelSights into a small robot hand while still having competitive estimation performance. Also, we can avoid preparing a lot

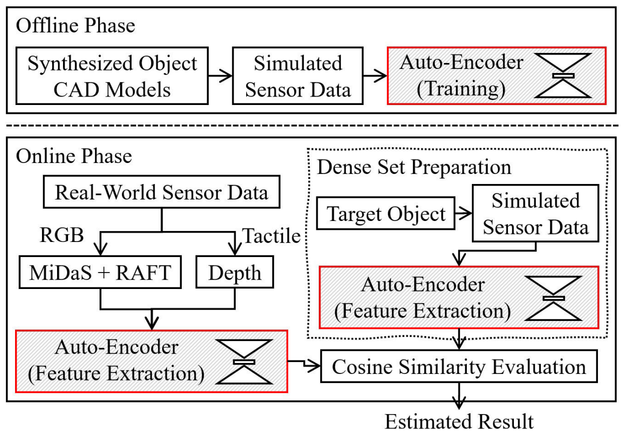


FIGURE 2. An overview of the proposed method.

of real-world data using modern simulation and photometric methods and achieve satisfying estimation performance.

III. AN OVERVIEW OF THE PROPOSED METHOD

Our proposed method can be divided into offline and online phases, as shown by the diagram in Fig. 2.

In the offline phase, we perform grasp planning using a database of synthesized object CAD models and the target robot hand, collect the ground-truth grasping poses of the synthesized object models, and obtain the simulated RGB and visuotactile data using simulation. We use the simulated RGB and visuotactile data to train an auto-encoder [50] for extracting features and performing cosine similarity analyses.

In the online phase, we obtain the real-world sensor data while an object is grasped and estimate the in-hand object pose. Particularly, we collect a series of RGB images using the hand-mounted cameras when the robot hand is performing a task (moving the object). The background of these RGB images is different since the robot hand is in action. Contrarily, the foreground stays the same. We extract the foreground (the object and hand area) of these images by using a combined optical-flow estimation neural network (MiDaS) and depth estimation neural network (RAFT). Along with the RGB images and background subtraction, we also collect depth images using the GelSight sensors at the fingertips. The foreground silhouette and depth images will be sent together to the auto-encoder trained in the offline phase to extract features for cosine similarity evaluation.

The cosine similarity evaluation is carried out against features extracted from sensor data simulated based on a dense set of in-hand poses for the same object. The auto-encoder trained in the offline phase is again used for feature extraction. The extracted features are considered the ground truth for comparison. The cosine similarity evaluation compares the features extracted using the real-world sensor data with the ground truth and finds the most similar one. The in-hand object pose that produces the most similar feature will be counted as the estimated result.

The above two phases and the details of data processing, training, and estimation will be presented in the following Sections IV, V, and VI.

IV. AUTO-ENCODER TRAINING WITH SIMULATED DATA

In this section, we present the details of the offline phase. The content corresponds to the upper "Offline Phase" frame box of Fig. 2. The goal is to train autoencoders using simulated data. The trained autoencoders will be used in the "Online Phase" frame box and "Dense Set Preparation" dashed frame box of Fig. 2 for extracting features. Fig. 3 shows the structure of the auto-encoder designed for feature extraction. There are two autoencoder networks in the structure. The first encodes the RGB camera data and is named the camera encoder. In our design, we convert the original RGB images captured by the cameras to binary images to reduce the influence of textures and illumination. The environmental background is binarized as black, and the foreground (hands and in-hand objects) is binarized as white. The camera encoder extracts features from the binarized RGB images. The second autoencoder network extract features from the GelSight sensors and is named the visuotactile encoder. The GelSight sensors could output depth images about the contact. The depth images are directly used for feature extraction. For both autoencoder networks, we reshape the matrix output of the third Maxpooling 2D layer into vectors as the extracted features. The lengths are $16\times9\times8 = 1152$ for RGB cameras and $12\times12 \times 8 = 1152$ for GelSight sensors. Since there are two RGB cameras and two GelSight sensors in the system, we get a total of $1152 \times 2.1152\times2$ feature vectors for pose evaluation. Fig. 3(a) shows the training process, and Fig. 3(b) shows the prediction process of the two autoencoders respectively. Especially, we use a denoise autoencoder for camera data. As we will see in the next section, the results of MiDaS and RAFT are binary images with noisy boundaries. Using a denoise autoencoder will be more effective in feature extraction in the presence of noises [51].

To train the auto-encoders, we use synthesized objects in a simulation environment to collect a large amount of training data, as shown in Fig. 3(a). The synthesis considers four primitive shapes as the basis: a box, a cylinder, a pyramid, and a cone. These four shapes are resized and laid up randomly to form dummy objects, as shown in Fig. 4(a). We use a grasp planner [52] to plan many grasping poses for the dummy objects and obtain the simulated RGB camera images and visuotactile depth images by rendering the objects and the planned grasped poses in simulation. Fig. 4(b) shows an example of the images obtained in the simulation. Especially for the visuotactile data, we sample the object surfaces to get surface point clouds and extract the points that fall into the region of the GelSight sensor's silicone pad to simulate deformation at the contact. The deformation is normalized against the maximum sensing depth of the GelSight sensor to produce depth images. We do not have special considerations about the elastic deformations of the

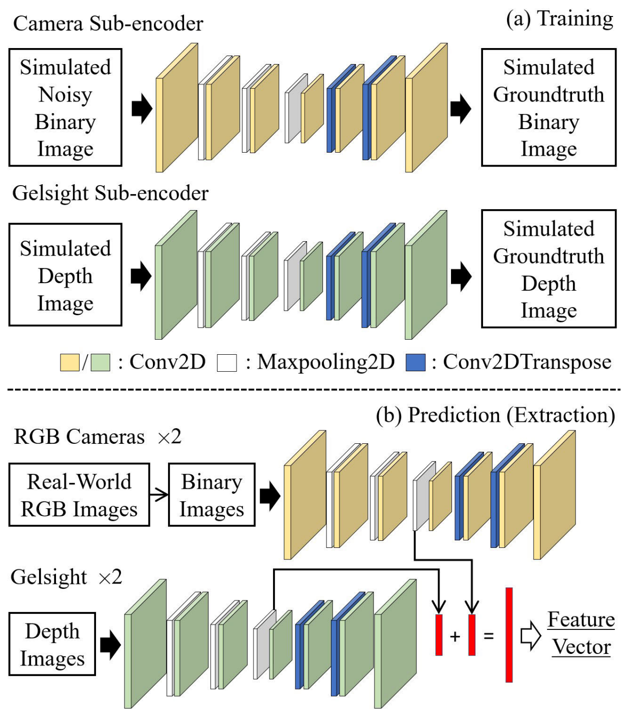


FIGURE 3. Structure and usage of the auto-encoders. (a) Training process. (b) Extracting Features using the auto-encoders. Especially a denoise autoencoder is used for the RGB cameras.

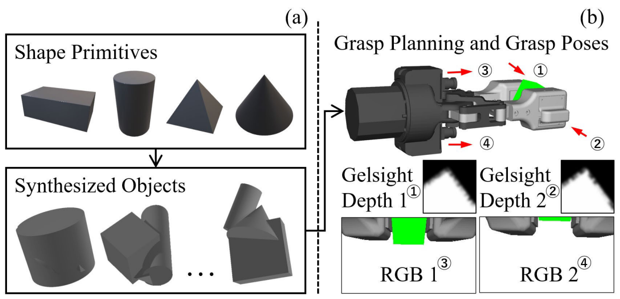


FIGURE 4. (a) Synthesizing objects using primitive shapes. (b) Planning grasping poses by using a synthesized object and the robot hand's CAD model. Simulated RGB camera and visuotactile depth images are obtained by rendering the objects and the planned grasped poses in simulation.

Gelsight's rubber finger pads. The grasping power is fixed to a given value and the penetration depth is always the same at this stage. After obtaining the RGB camera and visuotactile depth images, we process them to prepare for training. First, we recast the RGB images to binary images by considering the background pixels as black values and the hand and in-hand object pixels as white values, and add noises to the binary images to make them similar to the result of a real-world binarization. The considered noises include (1) Extra background removal; (2) Less background removal; (3) Jagged boundaries. We simulate the noises by expanding or contracting binary edges and adding noises to the expanded or contracted area. After that, we apply Mosaic processing by making the foreground area smaller or larger to restore the original size. Second, we add noises to the

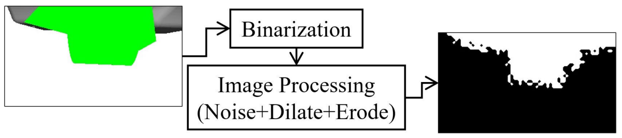


FIGURE 5. Originally simulated RGB image and the result after including the various noises.

visuotactile depth images to make them similar to real-world data. Fig. 5 illustrates an example of the originally obtained images and the results after including the various noises. We train the auto-encoder by using the processed binary and depth images. We use the noisy binary images as the input for the camera encoder and their corresponding noise-free binary images as the output to learn the network parameters. For the visuotactile encoder, we use depth images as the input and output for learning. The trained networks are used in the online phase to extract features from real-world sensor data, as illustrated in Fig. 3(b). The details of the online extraction will be presented in the next section.

It should be noted that the training process did not take into account real-world objects. This was due to the difficulty in replicating the exact grasping state in the simulation with a real-world robot. Furthermore, the simulation employed a variety of grasping poses, making it impractical to prepare real-world Gelsight data for them. However, recent studies have suggested that combining real-world data with simulation data during training can improve the performance of neural networks, as noted in reference [53]. As such, it would be beneficial to develop a set of simple, "standard" objects to enable the collection of corresponding real-world and simulation data, which could then be combined with pure simulation data to enhance training performance.

V. EXTRACTING REAL-WORLD SENSOR FEATURES

This section presents the details of the online phase. It corresponds to the "Online Phase" frame box of Fig. 2. The dashed frame box inside it is irrelevant.

In an online phase, we take advantage of a moving robot hand for the cameras to get a series of RGB images with changing backgrounds. The hand and in-hand objects do not change in the image series since they are relatively stationary concerning the cameras. We thus can segment the background and foreground considering the changes of each image and binarize the RGB data. Conventional methods to perform the separation were model-based methods like building a Gaussian Mixture Model or analyzing the changes in the optical flow of the image series. They exhibited satisfying performance but were not robust to illuminations and environmental changes. Unlike the model-based methods, we use a combined MiDaS and RAFT deep neural network in this paper to carry out the separation and image binarization. The MiDaS [5] network can estimate the depth information from a single RGB image. Since the hand and the in-hand object are relatively close, and the background is relatively

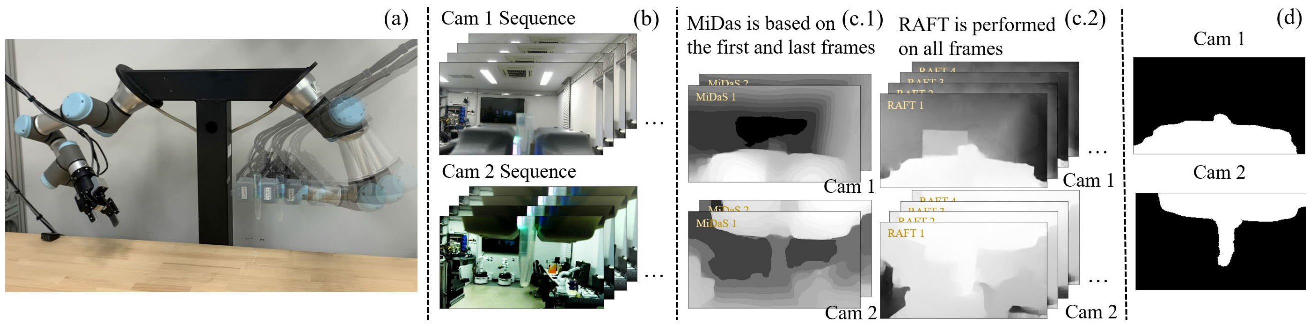


FIGURE 6. Using MiDaS and RAFT to remove background and binarize the RGB images. (a) Capture RGB images while performing a task (transporting a tube held in hand). (b) Captured RGB images. (c.1) Results of MiDaS. It is applied to the first and last frames, and thus there are two images. (c.2) Results of RAFT. It is applied to all frames. (d) Final binary output.

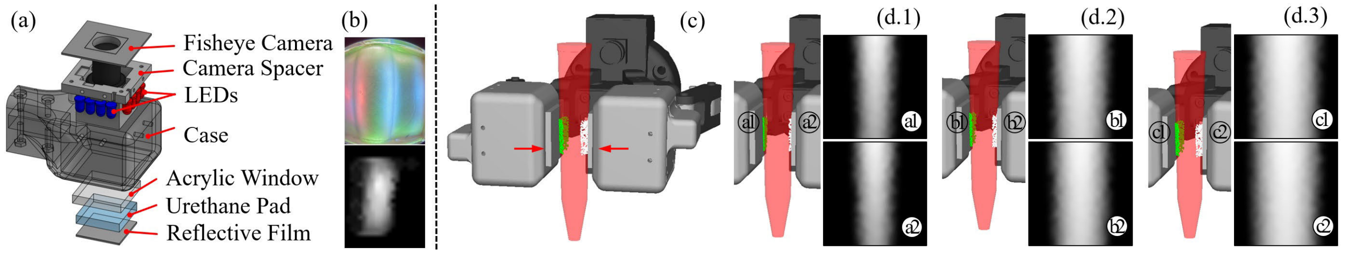


FIGURE 7. (a) Structure of the GelSight sensor used in this paper. (b) Image of the contact surface and corresponding depth map. (c,d) Simulated depth images under different grasping power. (d.1) Small grasping power. (d.2) Medium grasping power. (d.3) Large grasping power. The circled numbers denote the corresponding simulated depth images from the left and right sensors.

far from the cameras, the MiDaS network may help tell the foreground and background from the view of estimated depth values. The RAFT [6] network can estimate the optical flow in a series of images. The moving robot hand leads to changing background and thus changing the optical flow in the images. At the same time, the hand and the in-hand object do not move and do not cause much optical change. Using RAFT, we expect to obtain the optical flow in the image series and tell the background by recognizing the image regions with large optical values and the foreground hand/in-hand objects by recognizing the regions with a low optical response. Note that neither MiDaS nor RAFT can perfectly segment the background and foreground. We apply them to different image frames collected during the robotic manipulation motion to improve integral performance. It is not necessary to prepare specific actions or environments in advance. The MiDaS and RAFT algorithms are capable of compensating for each other's limitations and segmenting the hand and hand-held objects from the surrounding environment based on data collected during task execution with high performance.

In detail, we apply MiDaS to only the first and last frames in the image series and apply RAFT to every adjacent and adjacent frame pair in both forward and backward orders. Fig. 6 illustrates the process. Suppose each frame in the image series has a size of $w \times h$, with a total of N frames. The MiDaS network produces two depth frames. The RAFT network produces $(N-1) \times 2 + (N-2) \times 2$ frames where the value $N-1$ indicates the number of adjacent pairs, the value $N-2$ indicates the number of every other adjacent pair, the number 2 indicates the two orders. We get $4N-4$ frames of $w \times h$ data after applying MiDaS and RAFT. We stack them together into a $w \times h \times (4N-4)$ tensor, normalize the element values, and apply X-means clustering [54] at the last column (cluster the $(4N-4)$ vectors) to determine the pixel clusters on $w \times h$. After that, we assume the top center pixel of the image represents the object with the highest confidence and regard the pixels that belong to the same cluster as the top center pixel to be the foreground.

For the GelSight sensors, the depth images are captured once and are directly used without further processing. The method we used to extract depth information from GelSight sensors follows Yuan's work [4]. It employs a database to estimate the relationship between the reflection intensities and the normal directions of the contact surface. The database is built by pressing a standard 1 cm sphere onto the sensor's surface and collecting the surface images using the embedded RGB camera. The database stores the corresponding relationship between the normals and pixel values. The surface normal map of an online contact is estimated by comparing surface images captured using the embedded RGB camera with the stored database. A dense depth image can be obtained from the surface normal map by solving a Poisson equation. After obtaining the dense depth image, we use an extra sampling process to reduce its density and make it coherent with the training visuotactile depth images obtained in the simulation. Fig. 7(a,b) show details of the GelSight sensor, the dense depth image of the contact surface, and the down-sampled depth image.

After obtaining the binary and depth images from the real-world sensors, they are sent to respective auto-encoders for feature extraction. The extracted features will be compared against sensor data simulated based on a dense set of in-hand poses for the same object for pose estimation. The following section will present the estimation details.

VI. POSE ESTIMATION USING COSINE SIMILARITY

This section zooms into the dashed frame box inside the "Online Phase" of Fig. 2. This dashed frame box represents a preparation step. The prepared features will be used in the online phase for pose estimation.

Particularly, we compare features of the real-world data against features extracted from simulated sensor data to estimate the real-world grasping pose. The simulated data is obtained using a dense set of in-hand grasp poses for the same object. The comparison is divided into two steps to save time. In the first step, we sample a feature set with large granularity and find a rough best match. Then, we re-sample a dense feature set near the rough best match in the second step to refine the estimation result. The following part presents the details of the two-step matching method.

First, given the CAD model of the object to be estimated and the hand model, we plan a set of grasp poses with large granularity, collect the simulated binary and depth images, and encode them into features using the same auto-encoder. The grasp planning and image collection are carried out the same way as the synthesized shapes. Especially for the GelSight Sensor, we additionally consider the robot hand's grasping power. Since different grasping power will lead to different penetration depths, we use a range of maximum penetration depths to produce different depth images and simulate different grasping power. Like the training stage, the depth images are normalized against the maximum sensing depth of the Gelsight sensor. There is no consideration of the Gelsight's rubber pads' elastic deformation. Fig. 7(b,d) shows an example of the various simulated depth images produced using the method.

With sampling and extraction, we can obtain a database of features. Each feature in the database corresponds to a grasping pose with large discretization granularity. We estimate a best-matched grasp pose by computing a cosine similarity between the feature vector from the real-world sensors and the ones in the database. The best estimation result is the output of the first step.

The best estimation may have a large offset from the ground truth value since it was an element in the database sampled with a large granularity. To further refine it, we perform refinement in a second step by re-sampling a dense set of grasps near the rough best match, computing their feature vectors, and re-matching the real-world vector to the dense set. The second step's best-rematched grasping pose will be counted as the estimated in-hand pose.

VII. EXPERIMENTS AND ANALYSES

In this section, we examine and analyze the proposed method using the hardware shown in Fig. 1. The robot gripper is Robotiq-85 2F produced by Robotiq Inc. It has two compliant parallel fingers driven by two five-bar linkages. The two RGB cameras (ELP-SUSB1080P01-LC1100-J) are mounted at the hand palm's upper and lower flat areas. The two GelSight sensors are installed on each of the fingertips. The algorithmic implementations and 3D result visualization are based on the open-source robot planning and control platform developed in our lab.[1] We prepared simulated sensor data using 106 synthesized objects and 60293 associated grasping poses to train the auto-encoders. In addition, we prepared another 10639 grasping poses for the same objects and used the sensor data simulated based on them to validate the trained auto-encoders. The computer used for training and validation was equipped with an i7-11700K CPU, 64 GB of RAM, and an NVIDIA GeForce RTX 3060 GPU, and ran the Windows 11 operating system. The training time for the visual-tactile and camera auto-encoders was 2784 seconds and 3499 seconds respectively. The training processes were stopped when there were no improvements for five epochs. The validation loss for the two autoencoders were 0.1112 and 0.0106 respectively when the training loops were stopped. To carry out the two-step cosine similarity analyses, we collected data of grasping poses by the granularity of 5 [mm] and 5 [deg] in the first step and 1 [mm] and 1 [deg] in the second step. The computer used for the analyses was different from the one used for training and validation since it was a part of the robot's control system. It was equipped with an i9-9900K CPU, 32 GB of RAM, and a GeForce GTX 1080 Ti GPU, and ran the Ubuntu 18.04 operating system. The time cost for one case is approximately 167 seconds, which includes video and image processing, feature extraction, and the two-step analyses and estimation.

We designed two series of experiments to study the performance of the proposed method: The first series studied estimation accuracy using a 3D-printed object. AR markers are attached to the object to obtain ground-truth poses for comparison. The second series study the generalization of the method for daily life objects, with a special focus on the influence of surface textures and physical properties (transparency, reflectivity). Besides the two series, we showcase a scenario where the developed method can help improve the performance of robotic manipulation systems.

A. PREDICTION ACCURACY

First, we study the prediction accuracy of the proposed method by using a 3D-printed object. The object comprised two cubes and a handle that connected them. We attached ten AR markers to the cube faces and used an external camera to estimate the markers' poses. The average value of all observable marker poses will be recorded as the object's pose

[1]https://github.com/wanweiwei07/wrs

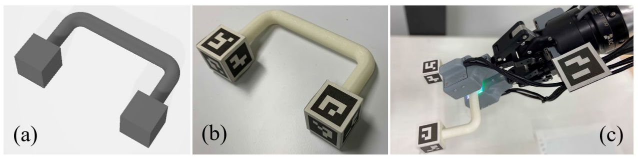


FIGURE 8. (a) CAD model of the 3D printed object. (b) Markers are attached to the end cubes. (c) Experiments in action.

in the external camera's local coordinate system. Meanwhile, we attach a reference AR marker to the wrist of the robot for estimating the hand pose in the camera's frame. The in-hand object pose is computed as the relative pose between the observed object's pose and the observed hand pose. The method does not require calibrating the cameras since there is no need to know the mark poses in the robot's coordinate system.

Fig. 8(a) and (b) show the CAD model of the 3D printed object and the AR markers attached to the end cubes, respectively. Fig. 8(c) shows the view of the external camera. The camera view can catch both the AR markers on the object and the reference marker at the wrist of the robot hand. Suppose the ground-truth pose obtained by the external camera is denoted by (p, R) and the estimated pose by our method is denoted by $(\hat{p}, \hat{R})$, we compute the estimated position error and rotation error using

$$\Delta p = |\hat{p} - p|, \ \Delta\omega = \arccos \frac{tr(\hat{R}R^{-1}) - 1}{2} \quad (1)$$

At the beginning of the experiments, we asked a human to hand over the 3D-printed object to the robot hand with 45 random poses. We used the proposed method to estimate the in-hand positions and rotations. We computed and plotted the Δp and $\Delta\omega$ with collected data to understand the estimation precision. The results are shown as a diagram in Fig. 9(a), with detailed pictures of six representative results listed in Fig. 9(b) and their corresponding sensor data in Fig. 9(c). Four of the results are positive. The other two are negative. The diagram in (a) shows that in 32 out of 45 cases, the proposed method can estimate the pose of the 3D printed object with less than 15 [mm] position error and less than 15 [deg] rotation error. It had good performance when the RGB cameras and GelSight sensors could measure significant features of the object. Fig. 9(b)-①∼④ show four such cases. In the first three of them, the robot hand grasped the middle part of the object handle. The object was in a vertical pose. Its two cubes appeared in the views of the two RGB cameras and exhibited significant features. The GelSight sensors observed vertical handle shapes. The proposed method estimated correct vertical poses. In the fourth case, one camera could not see the cube, but the GelSight sensors understood that the object handle was held horizontally. Thus, the proposed method estimated a lateral pose. Fig. 9(b)-⑤ and ⑥ show two exceptional cases where the proposed method exhibited bad performance. For ⑤, one cube was too near to the camera and was out of the camera's focus. The background subtraction method failed to extract the exact cube shape (see Fig. 9(c)-⑤ for better insight). In the case of ⑥, a large part of the cube was lost during background subtraction (Fig. 9(c)-⑥).

We particularly investigated the reason for the lost cube section by observing the MiDaS and RAFT sequences of ⑥. The sequences are shown besides Fig. 9(c)-⑥ for readers' convenience. We speculated that there might be two reasons for the wrong subtraction: First, there were several large black monitors in the environment. They might impair the optical flow and lead to misjudgment about background and foreground. Second, the robot's motion was short and might fail to provide enough frames to capture optical changes. Thus, we designed additional experiments to examine if our speculations were correct. The additional experiments involved a single in-hand pose, as shown in Fig. 10(a). We examined its estimation result by considering different environments and motion lengths. The result in Fig. 10(b.1) was obtained in an environment with large black objects (monitor screens) and short motion (16 frames of images). The result in Fig. 10(b.2) was obtained in an environment with large black objects (monitor screens) and long motion (30 frames of images). The result in Fig. 10(b.3) was obtained in an environment without large black objects (monitor screens) and short motion (16 frames of images). The result in Fig. 10(b.4) was obtained in an environment without large black objects (monitor screens) and long motion (24 frames of images). By comparing the results, we confirmed that large black objects in the background led to very bad subtraction results. For example, the result in (b.2) was completely wrong. In contrast, results in (b.3) and (b.4) had more acceptable errors. We also understood from the results that longer motion was not necessarily helpful. The result in (b.2) and (b.4) were based on long motion with 30 and 24 frames of images, respectively. However, they exhibited worse performance than (b.1) and (b.3).

In addition to the above experiments and analyses, we also explored the usefulness of combining different numbers and types of sensors for pose estimation. We used the same 3D-printed object to evaluate the accuracy of the estimations since we could easily compare the estimation results with the ground truth observed using the AR markers. The results of these experiments are shown in Fig. 11, which is divided into three parts. Part (a) shows the real-world states, part (b) shows the sensor data, and part (c) shows the estimated poses under different sensor settings. According to the results, the proposed "GS12+Cam12" combination had the best performance. The results of the "Cam12" column, which only used cameras, had lower accuracy in estimating rotations. Particularly the "Cam12"-③ item represents a distinctive case. The "GS12" column, which only used GelSight sensors, had even worse results, as the sensors only provide local contact information and cannot detect axial rotations. The results of the "GS12+Cam1" and "GS12+Cam2" columns, which used a combination of Gel-

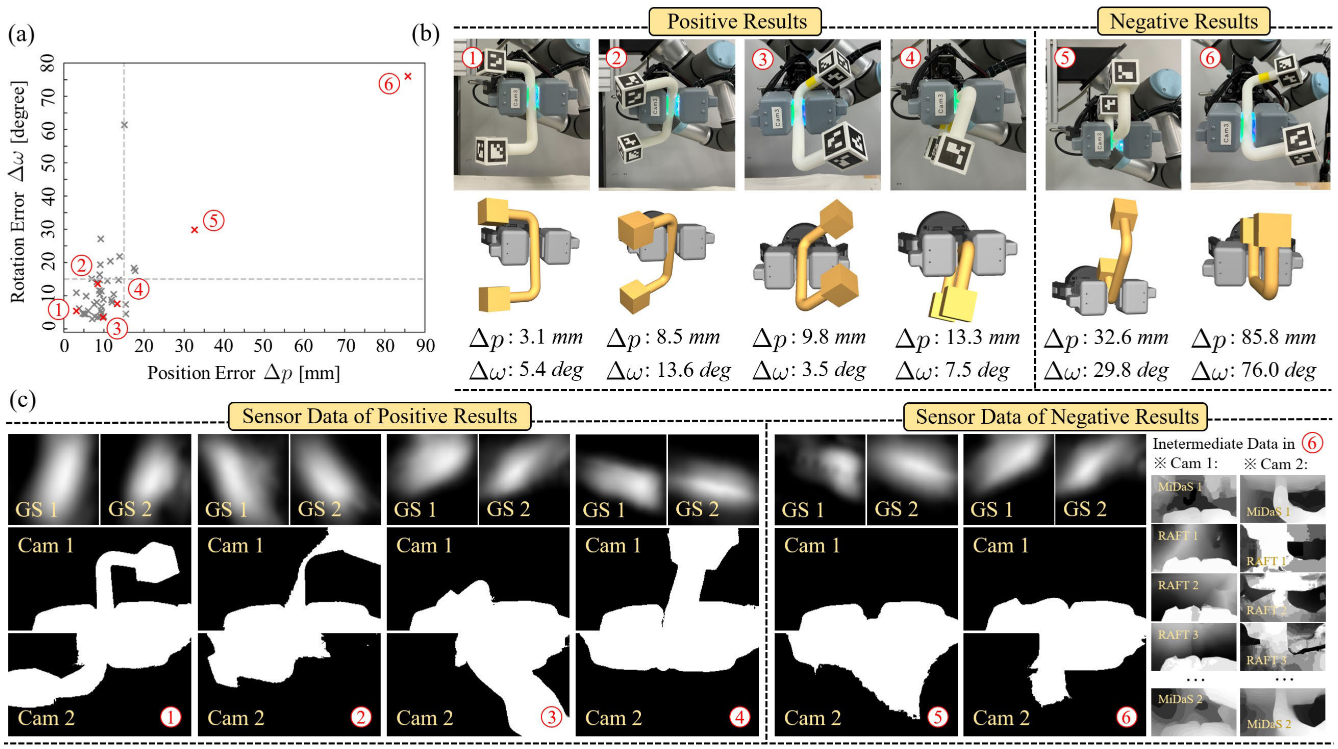
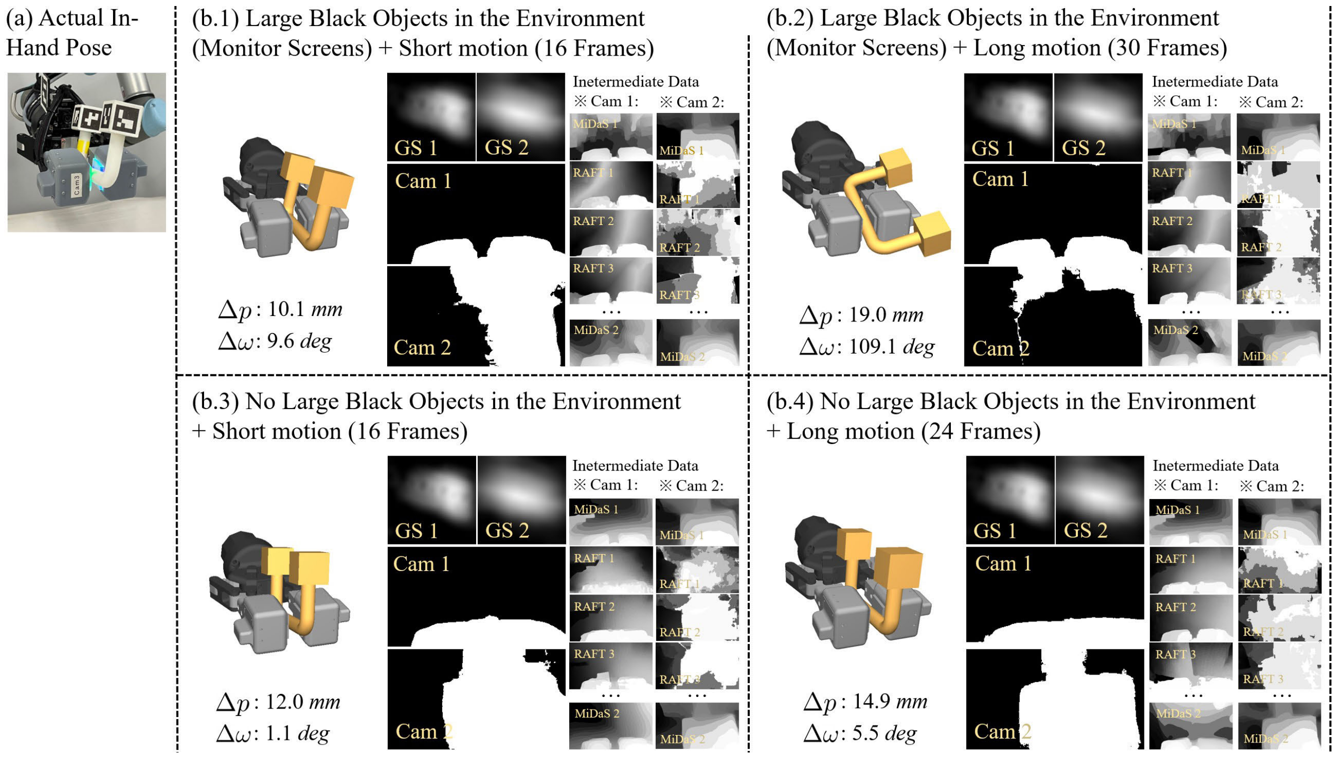


FIGURE 9. **(a) In-hand estimation results (position and rotation errors) of 45 random poses. (b) Detailed pictures of six representative estimation results. The fifth and sixth cases had significant errors and were considered failures. (c) Sensor data corresponding to the results in (b). The small figure sequences at the right-most columns are the details MiDaS and RAFT results of ⑥.**



(a) Actual In-Hand Pose
(b.1) Large Black Objects in the Environment (Monitor Screens) + Short motion (16 Frames)
Δp: 10.1 mm Δω: 9.6 deg
(b.2) Large Black Objects in the Environment (Monitor Screens) + Long motion (30 Frames)
Δp: 19.0 mm Δω: 109.1 deg
(b.3) No Large Black Objects in the Environment + Short motion (16 Frames)
Δp: 12.0 mm Δω: 1.1 deg
(b.4) No Large Black Objects in the Environment + Long motion (24 Frames)
Δp: 14.9 mm Δω: 5.5 deg
GS 1
GS 2
Cam 1
Cam 2
Inetermediate Data
※ Cam 1:
※ Cam 2:
MiDaS 1
RAFT 1
RAFT 2
RAFT 3
MiDaS 2

FIGURE 10. **Investigating the influence of large black backgrounds and motion length. (a) Experiments were carried out on this special pose for better comparison. (b) Results with different background and motion length conditions.**

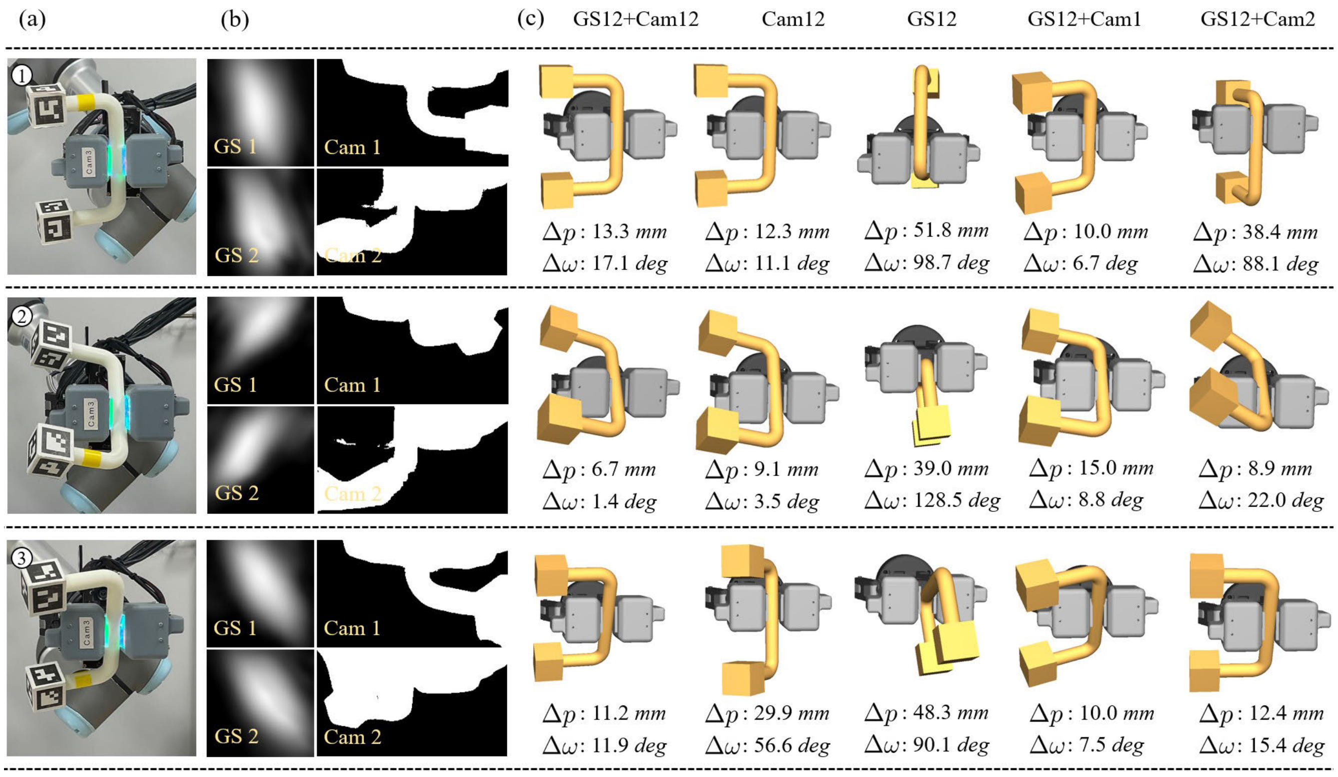


FIGURE 11. **Comparison of estimation accuracy under different sensor numbers and combinations. Part (a) shows the real-world states, part (b) shows the sensor data, and part (c) shows the estimated poses under different sensor settings.**

Sight Sensors with a single camera, had better performance but were still less accurate than “GS12+Cam12”.

B. OBJECTS WITH VARIOUS SURFACE PROPERTIES

1) EXPERIMENTS AND RESULTS

In the second series of experiments, we examine whether our proposed method can be applied to objects with various surface textures and optical properties such as transparency and reflectivity. We selected five representative objects, as shown in Fig. 12(a), which include (1) a translucent test tube, (2) a crystal test tube, (3) a stainless bearing shaft, (4) a plastic bolt, and (5) a shiny metal fixture (Nickel plated). Since it was difficult to attach AR markers to these objects, we examined the estimation precision by requiring the robot to place these objects vertically 10 [mm] above the table. Like the 3D printed object, we asked a human to pass an object to the robot with a random pose in the beginning. The robot estimated the in-hand pose of the object after grasping it and planned a motion to move the object to the desired position with a vertical pose. Fig. 12(b) and (c) exemplify the handover and final vertical pose. We installed a linear laser on one side of the robot and used it to measure the deviation of the vertical pose. When the object is moved to the target pose, we shoot a line laser to the bottom of the object and measure its difference from the expected position in the x, y, z directions illustrated in Fig.12(c) as the estimation accuracy. It is worth noting that the positional displacements obtained using the proposed method are only estimates, as it was challenging to accurately attach AR markers and obtain exact 6D poses. Despite this limitation, the performance of the method can still be evaluated by examining the estimation results. Additionally, the surrounding environment was not specifically modified or controlled in any way; it included elements such as black monitors, glass windows, and fluorescent lights, which were left as they were in order to assess the generalizability of the proposed method. MiDaS plus RAFT is exclusive for segmenting transparent and shiny objects from such random environmental backgrounds. The robot will fail to segment the foreground and background if conventional model-based or Bayesian inference methods were used, let alone fusing with tactile data and estimating the in-hand poses.

We carried out experiments for each object by 15 times to examine the estimation performance. Especially we divided the grasping poses into two main categories. (I) Grasping poses that both RGB cameras can see the object. (II) Grasping poses that only one RGB camera can see the object. We chose the division because the sensors would receive fewer features when one camera cannot capture the object. Also, nearly all negative results of the 3D printed objects involved one camera that could not capture the object. We hope to understand the performance when only one camera is effective. Table 1 shows the results of each object with the divided grasping pose categories. The rows without yellow or green highlights show successfully measured data. They confirmed that the

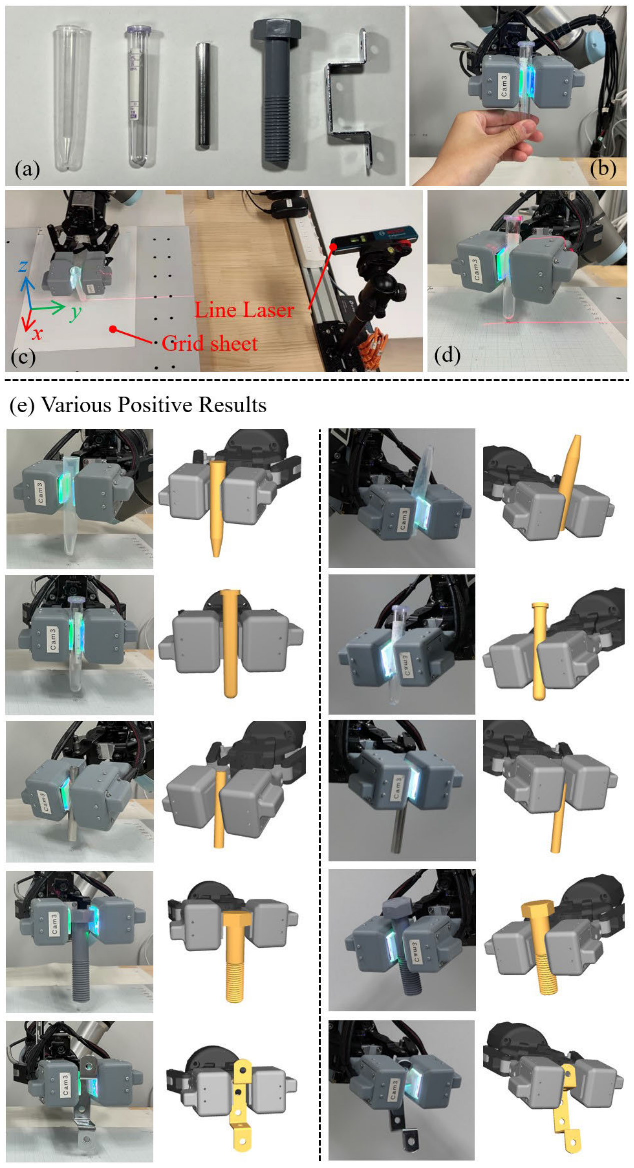


FIGURE 12. (a) Translucent test tube, crystal test tube, stainless bearing shaft, plastic bolt, and shiny metal fixture. (b) Human handover. (c) Placing the object onto a grid sheet vertically following the estimated results. A line laser is used to examine the accuracy. (d) A close-up view of the grid sheet and line laser. (e) Several representative positive results.

proposed method could accurately estimate objects with different surface textures and properties. Fig.12(e) illustrates some representative positive results using simulation. Analyses and discussions about failure cases or the results with larger than 20 [mm] estimation error (elements with gray highlights in the table) will be presented in the following sub-subsection.

2) IN-DEPTH ANALYSES ON EXCEPTIONAL CASES

The experimental results indicate that although the method is applicable to various objects regardless of their surface properties, such as transparency or shininess, there remain exceptional cases where the sensors fail to recognize the object's in-hand pose.

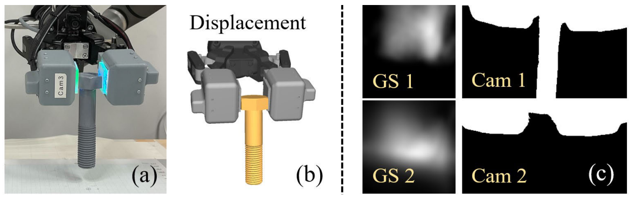


FIGURE 13. A negative result of the plastic bolt. (a) Real object's pose. (b) Estimated object's pose. (c) Sensor data. GS: GelSight; Cam: RGB Camera.

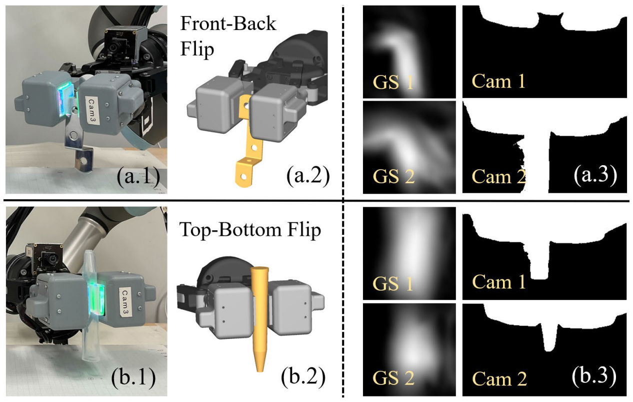


FIGURE 14. (a) A negative result of the shiny metal fixture: Front and back sides flipped. (b) A negative result of the translucent tube: The top and bottom flipped.

The first category of exceptional cases happens when only one RGB camera can see the object or the object is out of the RGB cameras' view. Fig. 13 shows an example. It corresponds to the "Plastic Bolt" (ID 1) results in Table 1 and had a 23 [mm] error in Δx. Here, only the RGB camera 2 can see the object. Also, the same exception can be observed when only a section of the object instead of the whole skeleton is in the camera's view. In this case, the RGB camera images may not differentiate various displacements and rotations.

The second category of exceptional cases happens when an object has symmetric or repeated features. For example, the metal fixture used in our experiments has four right-angle corners. When the gripper holds the fixture like Fig. 14(a.1), the proposed method cannot tell which right-angle corner is in contact. The GelSight and RGB camera data in the case of Fig. 14(a.1) are shown in Fig. 14(a.3). Both the two types of sensors have difficulty in understanding the object's orientation due to noises and the object shape's symmetric geometry. The second category of exceptional cases also explains the failed rows marked with $\times^{1-3}$ (yellow background) in Table 1. Symmetry or repeated features completely reversed the estimation results for these rows. Fig. 14(b) shows an example. The reason for this misestimation was that the transparent tubes were vulnerable to noises. It wasn't easy to extract the tube area from the RGB images. Also, the dilation and erosion added to the simulated images in section IV led to ambiguity at the two ends of the test tubes. The auto-encoder failed to extract useful features from them.

TABLE 1. Estimation results of five representative objects with divided grasp pose categories.

	Translucent Test Tube				Crystal Test Tube				Plastic Bolt				Stainly Bearing Shaft				Shiny Metal Fxiture			
ID	*g.c.*	Δx	Δy	Δz	*g.c.*	Δx	Δy	Δz	*g.c.*	Δx	Δy	Δz	*g.c.*	Δx	Δy	Δz	*g.c.*	Δx	Δy	Δz
1	II	2	4	0	II	4	0	6	II	23	1	6	II	9	5	1	II	6	3	4
2	II	1	1	0	I	2	0	2	II	4	1	2	II	3	5	1	I	12	6	2
3	II	3	0	1	I	8	0	4	II	8	1	3	II	7	2	4	II	11	7	5
4	I	7	6	3	I	3	1	3	I	13	0	4	II	5	5	2	II	3	5	7
5	I	0	3	4	I	1	0	3	I	3	1	1	I	1	0	0	II	14	14	9
6	I	5	1	4	I	6	1	5	I	22	0	4	I	3	1	4	I	1	4	2
7	II	4	18	10	I	5	1	1	I	20	1	3	I	3	2	14	II	1	9	5
8	II	4	6	7	I	2	1	4	I	9	1	5	I	4	1	6	II	17	18	12
9	I	8	2	9	I	1	1	0	II	6	11	2	II	3	0	4	I	4	4	3
10	II	41	11	30	I	14	1	7	I	11	9	1	II	1	2	1	II	11	21	1
11	I	3	1	0	I	1	0	6	I	7	5	1	II	2	2	0	II	14	2	1
12	I$^{\times 1}$	-	-	-	I	6	1	2	II	6	1	3	II	16	13	0	I	3	2	1
13	I	4	0	8	I	5	5	3	I	9	0	0	II	5	1	0	I	0	2	0
14	I	5	1	5	II$^{\times 2}$	-	-	-	I	9	1	2	II	0	1	6	II	6	4	7
15	I	3	0	3	I$^{\times 3}$	-	-	-	I	4	1	0	I	3	3	0	I$^{\times 4}$	-	-	-

* *g.c.* - Grasp pose category; $\Delta x/y/z$ - Errors in x, y, z directions. The values are measured in millimeters.

* $\times^{1-3}$ - Estimation results are completely reversed. $\times^{4}$ - Estimation result goes wrong because of the environment light condition.

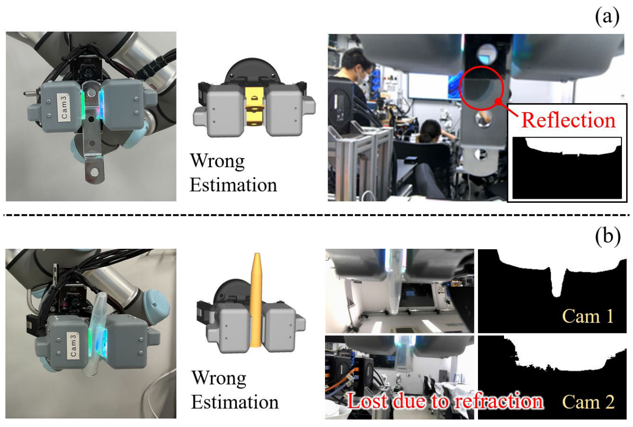


FIGURE 15. (a) Dominating reflection or (b) refraction may cause problems for background subtraction.

The third category of exceptional cases happens to objects with high reflectivity or refraction. The optical-flow method might misrecognize the reflection on an object's surface as the moving background, especially when the object is very close to the camera and the reflection dominates the camera's view. Fig. 15(a) illustrates an example of RGB data under dominant reflection. This corresponds to the results for "Shiny Metal Fixture" (ID 15) in Table 1. Fig. 15(b) illustrates an example of refraction. This corresponds to the results for "Translucent Test Tube" (ID 10) in Table 1. The upper part of the tube, which is close to the camera, is made imperceptible to the camera due to light refraction. As a result, the estimation had an error of more than 40 [mm] in Δx.

Some other factors may also impair the results. For example, the auto-encoder may fail to extract valuable features due to the sim-to-real gap between the simulated and real sensor data. The dilation and erosion of simulated RGB images mentioned in the second category represent one case. For GelSight, the simulated sensor data could be significantly different from the real data because of the sensor's flexible contacting medium, as shown in Fig. 16. The contact area in real sensors could be more significant than in simulated images due to the contact silicon's continuous elastic deformation. It easily leads to ambiguity in pose estimation. Developing a realistic Gelsight simulator [55] would be helpful in solving the problem.

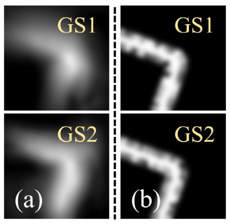


FIGURE 16. (a) Real data from GelSight. (b) Simulated data.

In conclusion, the proposed method might lead to unsatisfying estimation results when one or more sensors provide useless or misleading information. The method can estimate an object's in-hand pose regardless of its texture, optical properties, shapes, and surrounding light changes. However, it may still fail in the presence of dominating symmetry, reflection, and illumination changes.

C. REAL-WORLD TASKS

The proposed sensor configuration and in-hand pose estimation method are used in a practical, real-world task. The goal was to insert a pipette into a transparent tube for liquid aspiration. Fig. 17 shows the experimental setup for this task.

In the task, we assume the pipette is fixed on the robot's right hand with a known pose, and the tube is grasped with a random unknown pose. The mouth of the test tube is small, and the task has high requirements for accurately estimating

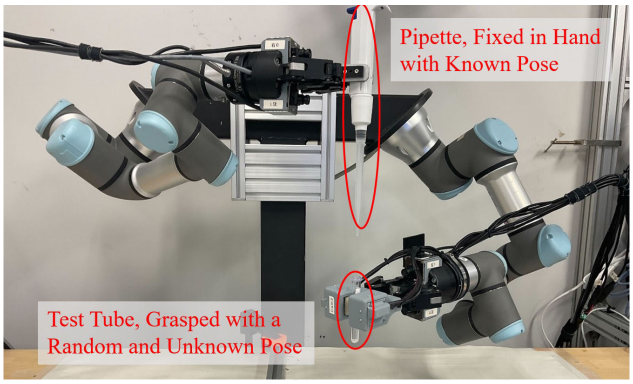

Pipette, Fixed in Hand with Known Pose
Test Tube, Grasped with a Random and Unknown Pose

FIGURE 17. A practical task: Inserting a pipette into a transparent tube for liquid aspiration.

its in-hand pose and, thus, the exact mouth position and orientation. The task is a representative requirement widely seen in industrial scenarios. We used the method mentioned above to estimate the tube pose and successfully carried out the insertion and aspiration action. An execution result can be found in the supplementary video attached to this manuscript. The robot recognized in-hand poses while moving the tube and successfully inserted the pipette.

VIII. CONCLUSION AND FUTURE WORK

In this work, we proposed a method to perform object pose estimation using two RGB cameras mounted on a robot hand and two GelSight sensors installed at the fingertips. The method included an offline part and an online part. In the offline phase, we trained auto-encoders using simulated sensor data and extracted features from simulation data for further comparison using the trained encoders. We used the trained auto-encoder in the online phase to extract features from real sensor data. We estimated the in-hand object pose by comparing the extracted features with the simulation. The proposed method could estimate an in-hand object pose with 15 [mm] accuracy in position and 15 [deg] accuracy in rotation. It could estimate an object's pose without considering its shape, transparency, and shininess. The method was practical and applied to a real-world robotic pipetting task.

Despite the positive conclusions, there remains room for improvement. The method's performance worsens when objects are large or in a pose where one or more sensors cannot provide helpful information. In the future, we will consider using extended sensor design and configurations to promote performance. We are also interested in developing robust optical flow algorithms and realistic simulations by considering the elastic rubber deformation of the Gelsight sensor in order to improve the performance of individual supporting modules.

REFERENCES

[1] J. Bimbo, S. Luo, K. Althoefer, and H. Liu, "In-hand object pose estimation using covariance-based tactile to geometry matching," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 570–577, Jan. 2016.

[2] M. Bauza, A. Rodriguez, B. Lim, E. Valls, and T. Sechopoulos, "Tactile object pose estimation from the first touch with geometric contact rendering," in *Proc. Conf. Robot Learn.*, 2021, pp. 1015–1029.

[3] B. Wen, C. Mitash, S. Soorian, A. Kimmel, A. Sintov, and K. E. Bekris, "Robust, occlusion-aware pose estimation for objects grasped by adaptive hands," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 6210–6217.

[4] W. Yuan, S. Dong, and E. H. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, vol. 17, no. 12, p. 2762, 2017.

[5] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 3, pp. 1623–1637, Mar. 2022.

[6] Z. Teed and J. Deng, "RAFT: Recurrent all-pairs field transforms for optical flow," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 402–419.

[7] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, Y. Taguchi, T. K. Marks, and R. Chellappa, "Fast object localization and pose estimation in heavy clutter for robotic bin picking," *Int. J. Robot. Res.*, vol. 31, no. 8, pp. 951–973, Jul. 2012.

[8] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes," 2018, *arXiv:1711.00199*.

[9] M. Raessa, D. Petit, W. Wan, and K. Harada, "Visually guided extrinsic manipulation for assembly tasks," in *Proc. IEEE 4th Int. Conf. Adv. Robot. Mechatronics (ICARM)*, Jul. 2019, pp. 202–207.

[10] M. Kokic, D. Kragic, and J. Bohg, "Learning to estimate pose and shape of hand-held objects from RGB images," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 3980–3987.

[11] Y. Hasson, B. Tekin, F. Bogo, I. Laptev, M. Pollefeys, and C. Schmid, "Leveraging photometric consistency over time for sparsely supervised hand-object reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 571–580.

[12] R. Liu, W. Wan, K. Koyama, and K. Harada, "Robust robotic 3-D drawing using closed-loop planning and online picked pens," *IEEE Trans. Robot.*, vol. 38, no. 3, pp. 1773–1792, Jun. 2022.

[13] Y. Konishi, K. Hattori, and M. Hashimoto, "Real-time 6D object pose estimation on CPU," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 3451–3458.

[14] S. Stevsic, S. Christen, and O. Hilliges, "Learning to assemble: Estimating 6D poses for robotic object-object manipulation," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1159–1166, Apr. 2020.

[15] J. Bimbo, L. D. Seneviratne, K. Althoefer, and H. Liu, "Combining touch and vision for the estimation of an object's pose during manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 4021–4026.

[16] R. Li, R. Platt, W. Yuan, A. ten Pas, N. Roscup, M. A. Srinivasan, and E. Adelson, "Localization and manipulation of small parts using GelSight tactile sensing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 3988–3993.

[17] S. Cui, R. Wang, J. Hu, J. Wei, S. Wang, and Z. Lou, "In-hand object localization using a novel high-resolution visuotactile sensor," *IEEE Trans. Ind. Electron.*, vol. 69, no. 6, pp. 6015–6025, Jun. 2022.

[18] S. Pirozzi and C. Natale, "Tactile-based manipulation of wires for switchgear assembly," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 6, pp. 2650–2661, Dec. 2018.

[19] K. Koyama, M. Shimojo, T. Senoo, and M. Ishikawa, "High-speed high-precision proximity sensor for detection of tilt, distance, and contact," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3224–3231, Oct. 2018.

[20] F. R. Hogan, J. Ballester, S. Dong, and A. Rodriguez, "Tactile dexterity: Manipulation primitives with tactile feedback," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 8863–8869.

[21] E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez, "GelSlim: A high-resolution, compact, robust, and calibrated tactile-sensing finger," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1927–1934.

[22] M. Bauza, O. Canal, and A. Rodriguez, "Tactile mapping and localization from high-resolution tactile imprints," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 3811–3817.

[23] M. Bauza, A. Bronars, and A. Rodriguez, "Tac2Pose: Tactile object pose estimation from the first touch," 2022, *arXiv:2204.11701*.

[24] Y. She, S. Wang, S. Dong, N. Sunil, A. Rodriguez, and E. Adelson, "Cable manipulation with a tactile-reactive gripper," *Int. J. Robot. Res.*, vol. 40, nos. 12–14, pp. 1385–1401, Dec. 2021.

[25] A. Yamaguchi and C. G. Atkeson, "Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots (Humanoids)*, Nov. 2016, pp. 1045–1051.

[26] J. Liang, A. Handa, K. V. Wyk, V. Makoviychuk, O. Kroemer, and D. Fox, "In-hand object pose tracking via contact feedback and GPU-accelerated robotic simulation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 6203–6209.

[27] N. Kuppuswamy, A. Alspach, A. Uttamchandani, S. Creasey, T. Ikeda, and R. Tedrake, "Soft-bubble grippers for robust and perceptive manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 9917–9924.

[28] M. Lambeta, P. W. Chou, S. Tian, and B. Yang, "DIGIT: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation," *IEEE Robot. Automat. Lett.*, vol. 5, no. 3, pp. 3838–3845, Jul. 2020.

[29] S. Tian, F. Ebert, D. Jayaraman, M. Mudigonda, C. Finn, R. Calandra, and S. Levine, "Manipulation by feel: Touch-based control with deep predictive models," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 818–824.

[30] Q. Li, R. Haschke, and H. Ritter, "Learning a tool's homogeneous transformation by tactile-based interaction," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots (Humanoids)*, Nov. 2016, pp. 416–421.

[31] F. von Drigalski, S. Taniguchi, R. Lee, T. Matsubara, M. Hamaya, K. Tanaka, and Y. Ijiri, "Contact-based in-hand pose estimation using Bayesian state estimation and particle filtering," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 7294–7299.

[32] H. Mao and J. Xiao, "Reducing uncertainty in pose estimation under complex contacts via force forecast," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 2661–2667.

[33] H. Mao and J. Xiao, "Object shape estimation through touch-based continuum manipulation," in *Robotics Research*. 2020, pp. 573–588.

[34] B. Liang, W. Liang, and Y. Wu, "Parameterized particle filtering for tactile-based simultaneous pose and shape estimation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 1270–1277, Apr. 2022.

[35] S. Suresh, Z. Si, S. Anderson, M. Kaess, and M. Mukadam, "MidasTouch: Monte-Carlo inference over distributions across sliding touch," in *Proc. Conf. Robot Learn.*, 2022, pp. 1–21.

[36] P. Hebert, N. Hudson, J. Ma, and J. Burdick, "Fusion of stereo vision, force-torque, and joint sensors for estimation of in-hand object location," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 5935–5941.

[37] G. Izatt, G. Mirano, E. Adelson, and R. Tedrake, "Tracking objects with point clouds from vision and touch," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 4000–4007.

[38] M. Pfanne, M. Chalon, F. Stulp, and A. Albu-Schaffer, "Fusing joint measurements and visual features for in-hand object pose estimation," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3497–3504, Oct. 2018.

[39] Q. Li, A. Ückermann, R. Haschke, and H. Ritter, "Estimating an articulated tool's kinematics via visuo-tactile based robotic interactive manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 6938–6944.

[40] R. Koiva, M. Zenker, C. Schurmann, R. Haschke, and H. J. Ritter, "A highly sensitive 3D-shaped tactile sensor," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, Jul. 2013, pp. 1084–1089.

[41] S. Dikhale, K. Patel, D. Dhingra, I. Naramura, A. Hayashi, S. Iba, and N. Jamali, "VisuoTactile 6D pose estimation of an in-hand object using vision and tactile sensor data," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2148–2155, Apr. 2022.

[42] S. Suresh, Z. Si, J. G. Mangelson, W. Yuan, and M. Kaess, "ShapeMap 3-D: Efficient shape mapping through dense touch and vision," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 7073–7080.

[43] T. Anzai and K. Takahashi, "Deep gated multi-modal learning: In-hand object pose changes estimation using tactile and image data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 9361–9368.

[44] R. Gao, Z. Si, Y.-Y. Chang, S. Clarke, J. Bohg, L. Fei-Fei, W. Yuan, and J. Wu, "ObjectFolder 2.0: A multisensory object dataset for Sim2Real transfer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10598–10608.

[45] A. N. Chaudhury, T. Man, W. Yuan, and C. G. Atkeson, "Using collocated vision and tactile sensors for visual servoing and localization," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3427–3434, Apr. 2022.

[46] C. Sferrazza, T. Bi, and R. D'Andrea, "Learning the sense of touch in simulation: A sim-to-real strategy for vision-based tactile sensing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 4389–4396.

[47] A. Agarwal, T. Man, and W. Yuan, "Simulation of vision-based tactile sensors using physics based rendering," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 1–7.

[48] D. F. Gomes, P. Paoletti, and S. Luo, "Generation of GelSight tactile images for Sim2Real learning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 4177–4184, Apr. 2021.

[49] J.-T. Lee, D. Bollegala, and S. Luo, "'Touching to see' and 'seeing to feel': Robotic cross-modal sensory data generation for visual-tactile perception," in *Proc. Int. Conf. Robot. Automat. (ICRA)*, May 2019, pp. 4276–4282.

[50] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[51] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.

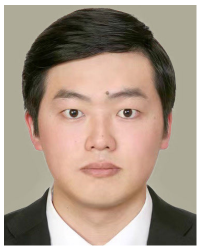
[52] W. Wan, K. Harada, and F. Kanehiro, "Planning grasps with suction cups and parallel grippers using superimposed segmentation of object meshes," *IEEE Trans. Robot.*, vol. 37, no. 1, pp. 166–184, Feb. 2021.

[53] H. Chen, W. Wan, M. Matsushita, T. Kotaka, and K. Harada, "Automatically prepare training data for Yolo using robotic in-hand observation and synthesis," 2023, *arXiv:2301.01441*.

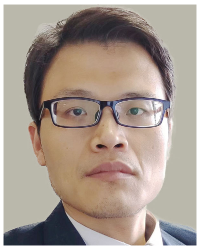
[54] D. Pelleg and A. W. Moore, "X-means: Extending k-means with efficient estimation of the number of clusters," in *Proc. Int. Conf. Mach. Learn.*, vol. 1, 2000, pp. 727–734.

[55] Z. Si and W. Yuan, "Taxim: An example-based simulation model for GelSight tactile sensors," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2361–2368, Apr. 2022.

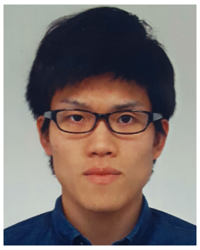
YUAN GAO received the B.S. degree in physics from Zhejiang University, in 2018, and the M.S. degree in engineering science from Osaka University, Osaka, Japan, in 2021, where she is currently pursuing the Ph.D. degree in engineering science. Her research interest includes robotic manipulation with vision-based tactile sensors.



SHOGO MATSUOKA received the bachelor's and master's degrees from the School of Engineering Science, Osaka University, Osaka, Japan. He is currently affiliated with Toyota Motor Corporation, Japan. This work was done when he was pursuing his master's degree at Osaka University. His research interests include robotic manipulation and factory automation.



WEIWEI WAN (Senior Member, IEEE) received the Ph.D. degree in robotics from the Department of Mechano-Informatics, The University of Tokyo, Tokyo, Japan, in 2013. From 2013 to 2015, he was a Postdoctoral Research Fellow with the Japan Society for the Promotion of Science, Tokyo, and a Visiting Researcher with Carnegie Mellon University, Pittsburgh, PA, USA. After that, he became a tenure-track Research Scientist with the National Advanced Institute of Science and Technology (AIST), Tokyo. He is currently an Associate Professor at the School of Engineering Science, Osaka University, Osaka, Japan. His research interests include smart manufacturing and robotic manipulation.



TAKUYA KIYOKAWA (Member, IEEE) received the B.E. degree from the National Institute of Technology, Kumamoto College, Japan, and the M.E. and Ph.D. degrees in engineering from the Nara Institute of Science and Technology, Japan, in 2018 and 2021, respectively. Since 2021, he has been with Osaka University, Japan, as a Specially-Appointed Assistant Professor, and with the Nara Institute of Science and Technology, as a Specially-Appointed Assistant Professor. His current research interests include robot manipulation and robot vision for reconfigurable robotic systems toward agile manufacturing. He is a member of RSJ, JSME, SICE, and JSAI.

KEISUKE KOYAMA received the Ph.D. degree from the Graduate School of Informatics and Engineering, The University of Electro-Communications, Japan, in 2017. He was a Project Assistant Professor at The University of Tokyo, from 2017 to 2019, and was affiliated with the Japan Society for the Promotion of Science (JSPS), from 2015 to 2016. He is currently working as an Assistant Professor at the Graduate School of Engineering Science, Osaka University, Japan. His research interests include high-speed tactile, proximity sensors and actuators, and dexterity robotic manipulation.

KENSUKE HARADA (Fellow, IEEE) received the Ph.D. degree from Kyoto University, Kyoto, Japan, in 1997. From 1997 to 2002, he was a Research Associate with Hiroshima University, Hiroshima, Japan. From 2005 to 2006, he was a Visiting Scholar with the Computer Science Department, Stanford University, Stanford, CA, USA. He is currently working as a Professor with the Graduate School of Engineering Science, Osaka University. Before joining Osaka University, he was a Researcher with the National Institute of AIST, Tsukuba, Japan. His research interests include the mechanics and control of humanoid robots and robotic hands.