2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Collision-aware In-hand 6D Object Pose Estimation using Multiple Vision-based Tactile Sensors

Gabriele M. Caddeo[1,2], Nicola A. Piga[1], Fabrizio Bottarel[1] and Lorenzo Natale[1]

***Abstract*— In this paper, we address the problem of estimating the in-hand 6D pose of an object in contact with multiple vision-based tactile sensors. We reason on the possible spatial configurations of the sensors along the object surface. Specifically, we filter contact hypotheses using geometric reasoning and a Convolutional Neural Network (CNN), trained on simulated object-agnostic images, to promote those that better comply with the actual tactile images from the sensors. We use the selected sensors configurations to optimize over the space of 6D poses using a Gradient Descent-based approach. We finally rank the obtained poses by penalizing those that are in collision with the sensors. We carry out experiments in simulation using the DIGIT vision-based sensor with several objects, from the standard YCB model set. The results demonstrate that our approach estimates object poses that are compatible with actual object-sensor contacts in 87.5% of cases while reaching an average positional error in the order of 2 centimeters. Our analysis also includes qualitative results of experiments with a real DIGIT sensor.**

I. INTRODUCTION

The ability to estimate the 6D pose of objects is of paramount importance for autonomous robotic platforms that interact with the environment and the objects therein. A large number of methods from the computer vision and robotics communities have addressed the object pose estimation [1]–[5] and tracking [6]–[10] problems using RGB-D images of the scene as input.

In robotic contexts, the sense of touch has also been employed to estimate or track the pose of objects, during in-hand manipulation, as the primary source of measurements [11]–[17] or complementary to vision [18]–[22] and sound [23]. Research in this field is supported by the availability of new tactile sensing technologies that provide high resolution tactile images, including the most recent vision-based tactile sensors [24], [25], that we consider in this work.

In the context of tactile-based object pose estimation, the majority of the work focuses on settings where the object is solely in contact with one sensor [11], [12], [18], [19], [23] or is manipulated by a sensorized parallel gripper [15], [16], [21], [22]. Multiple vision-based sensors in contact are considered in [13], however their experiments are limited to objects with size comparable to that of the sensor, hence producing object pose-distinctive features. To the best of our knowledge, no other attempts have been made to incorporate measurements from a generic number of vision-based tactile sensors, especially when considering objects of standard size, as we do.

[1]Humanoid Sensing and Perception, Istituto Italiano di Tecnologia, Genoa, Italy. name.surname@iit.it
[2]DIBRIS, Università di Genova, Via All'Opera Pia, 13, Genoa, Italy.

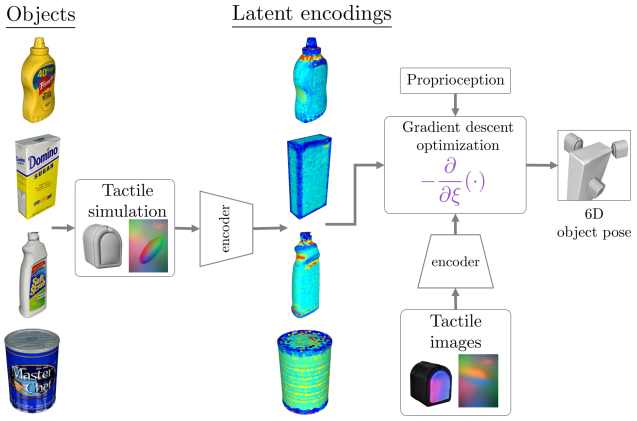

Fig. 1. The proposed pipeline uses an object-agnostic CNN-based encoder to extract contact-related features on the surface of the object from simulated tactile images. We combine these features with tactile images and proprioception in order to estimate the in-hand 6D pose of an object in contact with multiple sensors.

In this work, we propose an optimization-based pipeline that incorporates proprioception and images from several vision-based tactile sensors in order to estimate the pose of an object in contact with them. We reason on the possible configurations of the sensors on the surface of the object 3D model and select the most appropriate according to geometric considerations and the compatibility with the images coming from the sensors. The compatibility is tested using a Convolutional Neural Network (CNN) trained on simulated tactile images. We then use gradient descent (GD) to find a suitable pose for the object by minimizing the distance between the real position of the sensors and the selected configurations.

Differently from works dealing with small objects [11], [13], i.e. having a size comparable to that of the sensor, we instead consider objects of standard size. One of our key insights is that, in such setting, the tactile images of the contact do not depend on the specific instance of the object, as it happens with small objects. Rather, the resulting contact patches can be approximated using simple shapes such as ellipsoidal and rectangular patches with varying aspect ratios. We use this insight to train the CNN with object-agnostic data. Our contributions are the following:

- we propose a GD-based in-hand 6D object pose estimation method which combines several tactile images with proprioception;
- we devise an object-agnostic CNN-based encoding for the tactile images, and training procedure in simulation;
- we provide the results of simulated and real-world trials

using objects from the standard YCB model set [26] and tactile images from a DIGIT tactile sensor [25];
- we compare against a purely geometric baseline showing the advantage of using the information encoded by the tactile images.

II. RELATED WORK

Our work is closely related to the recent literature on tactile-based and visuotactile-based in-hand 6D object pose estimation and tracking.

Visuotactile-based methods In [18] the authors propose to extract point clouds from a single GelSight vision-based sensor and fuse them with point clouds from a depth camera using optimization based on signed-distance functions. More recently, [21] extends this idea by adding also RGB information and by fusing it with point clouds from depth and touch within a multi-stage CNN network. The CNN is trained with synthetic data obtained from the 3D meshes of several YCB [26] objects. A different path is considered in [20] where a prior on the object pose from vision is refined using a GPU-accelerated contact physics simulator that is fed with contact data from a sensorized multifingered hand during in-hand manipulation.

Unlike these works, this paper focuses on solely using tactile data without considering visual inputs.

Tactile-based methods In [16] the authors use Bayesian filtering to fuse proprioception and tactile feedback, in the form of force torque (FT) sensing, and estimate both the location of contacts and the in-hand object pose.

Other works focus specifically on vision-based tactile sensing. In [11] an image translation network is trained, using real and simulated images, to reconstruct local object surface normals from tactile images. Normals are then used within a factor-graph to infer the object pose. Similarly, in [13] the authors train a neural network with real object-agnostic data to reconstruct the local shape of the object at the contact point. The reconstructed images are then compared to a set of simulated images of the same object in order to find the most likely pose of the object. In [15] a CNN, trained on the ShapeNet dataset [27], is integrated within a Bayesian filter in order to map the images from two vision-based tactile sensors, mounted on a gripper, to the position and orientation of the gripper with respect to the object.

Among these approaches, [11], [13] are different from our work as they focus on small objects [11] or objects of comparable size to that of the sensor [13], resulting in *local* tactile features that are distinctive of the object pose. Instead we consider objects of standard size, that produce *local* features that are less dependant of the object pose. Moreover, in [11] the object contacts one sensor only, while we consider multiple sensors.

The authors of [15] make the same hypotheses on the size of objects as ours, however they limit the number of sensors to two, placed in a strict relative configuration, i.e. mounted on a gripper. Conversely, we do not make hypotheses on the sensors configuration. Moreover, they use meshes of real objects for training, while in our work we use simpler ellipsoidal and rectangular patches.

Remarkably, [17] is similar to our work as it uses a PCA-based descriptor to find object poses where the covariances of the local patches of the object 3D model match the principal components of the tactile data, represented as a 3D point cloud. In our work we use a similar concept but we substitute the PCA-based descriptor with CNN-based encoding that is suitable to process the output of vision-based tactile sensors in the form of RGB images.

III. METHODOLOGY

The setting we consider consists of L vision-based tactile sensors that are in contact with an object $\mathcal{O}$, with reference frame O_{xyz}. We assume that a 3D mesh model of the object and of the sensor are available.

Given the L sensors, their associated RGB images $\{I_i\}$ and poses $\{S_i^w \in \mathrm{SE}(3)\}$, in the world frame W_{xyz}, the proposed method estimates the pose of the object $T^w \in \mathrm{SE}(3)$. To this end, we instantiate a set of N initial candidate poses $\{T_{j,0}^w \in \mathrm{SE}(3)\}$ each associated with a L-tuple t_j

$$t_j = \{t_{1,j}^o, \cdots, t_{L,j}^o\}, \tag{1}$$

where $t_{i,j}^o \in \mathbb{R}^3$ is the *candidate* position of the i-th sensor within the j-th candidate pose, expressed in the object reference frame O_{xyz}. These tuples are chosen according to the images $\{I_i\}$. Specifically, we use a CNN-based approach to favour positions $t_{i,j}^o$ that are compatible with the contact patch captured by the i-th image I_i.

The candidate poses are then iteratively refined via GD optimization in order to minimize the distance between the sensors positions $t_{i,j}^w(T_j^w)$, expressed in the world frame according to the pose T_j^w, and the sensor poses S_i^w . Physical reasoning is finally leveraged in order to rank the resulting poses by giving more importance to those that are not in collision with the sensors. The best pose is chosen as the estimated pose T^w.

In the following we detail how we select the tuples $\{t_j\}$ given the images $\{I_i\}$ in Sec. III-A, we describe the GD-based optimization algorithm in Sec. III-B and the physical reasoning-based ranking mechanism in Sec. III-C. An overview of the proposed pipeline is depicted in Fig. 2.

A. Selection of the tuples

We build a database of candidate sensor positions $t^o \in \mathbb{R}^3$ by sampling M points uniformly in space along the surface of the 3D mesh of the object. For each point we collect a simulated tactile image $I(t^o)$ by placing the sensor in position t^o so that it is in contact with the object mesh with the normal to the object surface aligned with the normal out of the sensor surface. We use the Gazebo simulator [28] and the DIGIT simulator TACTO [29] to collect the tactile image readings offline and we save them in a database, for each object of interest.

Tactile-based selection The collected images are compared with the actual sensor images $\{I_i\}$ in order to identify which candidate sensor positions are compatible with the readings

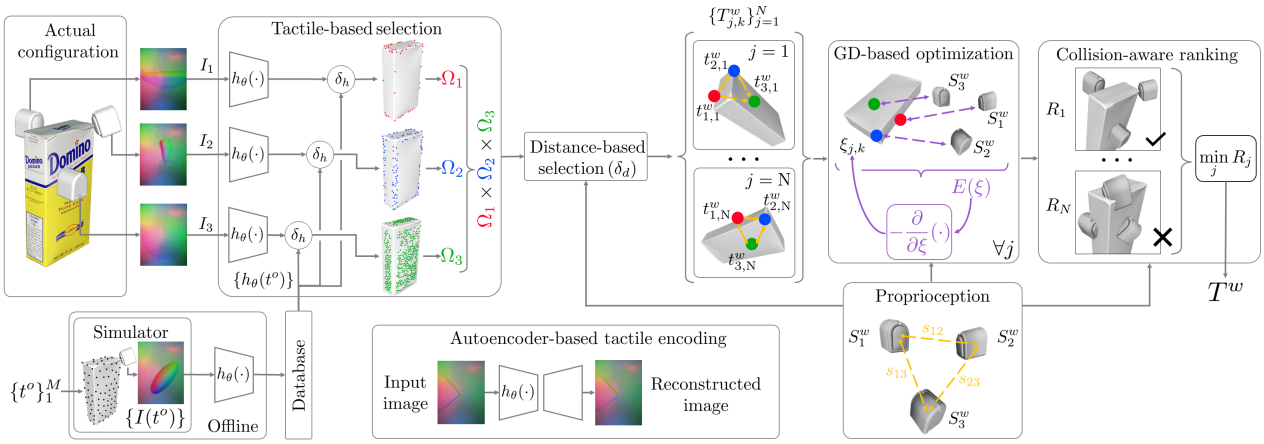

Fig. 2. Overview of our pipeline for in-hand 6D object pose estimation using multiple vision-based tactile sensors. In the figure we consider the case in which 3 sensors are used, i.e. $L = 3$.

from the sensor. To this end, we use a latent encoding $h_\theta(I) \in \mathbb{R}^{128}$ corresponding to the encoder part of a CNN-based autoencoder, later detailed, that is trained to reconstruct tactile images. Here, θ indicates the weight of the autoencoder. Moreover, we define I_{nc} as the simulated tactile image obtained when the sensor is not in contact with the object.

We assume that a given sensor position t^o is compatible with the image I_i if

$$|h_\theta(t^o)^T h_{nc} - h_\theta(I_i)^T h_{nc}| < \delta_h, \tag{2}$$

where $h_\theta(t^o) = h_\theta(I(t^o))$, $h_{nc} = h_\theta(I_{nc})$ are the features representing the absence of contact and δ_h is a threshold. I.e., we favour sensor positions whose tactile images have an inner-product distance (in latent space) from h_{nc} similar to that of the image I_i. We collect all the compatible sensor positions for the i-th sensor in the set Ω_i.

In Fig. 1 we show an example of the latent encoding represented on top of the surface of several objects from the YCB [26] model set. In Fig. 2, in the section "Tactile-based selection", we show an example of the sets Ω_i when the sensor is touching the "Sugar box" object within a corner or along an edge or a flat surface.

Once the candidate sensor positions have been selected for each sensor i, we obtain the set of candidate tuples $\{t_j\}$, with t_j as in Eq. (1), as the Cartesian product of the sets of all the sensors, i.e.:

$$\{t_j\} = \Omega_1 \times \cdots \times \Omega_L. \tag{3}$$

Distance-based selection Before assigning each tuple in Eq. (3) to a candidate pose $T_j^{w,0}$, we exploit the knowledge of the sensors poses S_i^w to filter out all the tuples whose sensor positions do not respect the relative distance between the actual sensor poses S_i^w.

We define $s_i \in \mathbb{R}^3$ as the translational part of the homogeneous transform S_i^w. Hence, we keep a tuple t_j if the following holds:

$$\frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{k=i+1}^{L} \left| \|s_i - s_k\| - \|t_{i,j}^o - t_{k,j}^o\| \right| < \delta_d. \tag{4}$$

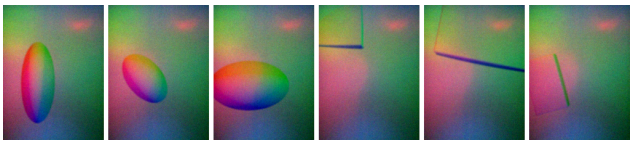

Fig. 3. Sample training images used to train the autoencoder for tactile image reconstruction.

In order to keep only a predetermined number of candidate poses N, we iteratively reduce δ_d until the number of tuples is less or equal to N.

Autoencoder training and details In our setting we assume that the objects of interest occupy a much larger volume than that of the sensor, as it is usual when comparing a fingertip with an object of standard size. As a consequence, we found that the resulting tactile patches do not depend on the specific object instance and are adequately represented by ellipsoidal and rectangular shapes of diverse aspect ratios. Therefore, we train the autoencoder to reconstruct object-agnostic simulated images of this kind, without the necessity to collect training images along the surface of each object of interest.

In practice, we obtained the images by simulating the contact with 3D models of superquadrics [30] with varying parameters, so as to obtain diverse shapes from spheres to ellipsoids to prisms and different aspect ratios. Fig. 3 shows several examples of training images.

Our network is inspired by the autoencoder for implicit 3D object rotation encoding presented in [31]. It comprises 4 convolutional and 4 deconvolutional layers for the encoder and decoder networks, respectively, with ReLU activation functions. We train the network by minimizing the pixel-wise L2 distance between the input image and the decoder output.

B. GD-based 6D object pose optimization

Given an initial candidate pose $T_{j,0}^w$, and the associated tuple t_j, we iteratively refine the object pose $T_{j,k}^w$ using a gradient descent-based approach:

$$\xi_j^{k+1} = \xi_j^k - K \frac{\partial E(\xi)}{\partial \xi}\bigg|_{\xi_j^k}, \tag{5}$$

TABLE I

RESULTS OF THE EXPERIMENTS IN SIMULATION WHEN USING THREE SENSORS: POSITIONAL, ADI-AUC ERRORS AND CONTACT ACCURACY FOR SEVERAL METHODS. #1 AND #1-5 INDICATE RESULTS EVALUATED ON THE BEST POSE AND AVERAGED ON THE BEST FIVE POSES, RESPECTIVELY.

Metric	Positional error (cm) ↓		ADI-AUC$_{2\text{cm}}$ (%) ↑		ADI-AUC$_{2\text{cm}}$ (rotation) (%) ↑		Contact accuracy (%) ↑	
Method	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours
Evaluated poses	#1, #1-5	#1, #1-5	#1, #1-5	#1, #1-5	#1, #1-5	#1, #1-5	#1, #1-5	#1, #1-5
002_master_chef_can	5.02, 4.96	**1.27**, **1.26**	25.00, 23.78	**72.97**, **84.23**	96.31, 93.19	**96.36**, **94.79**	25.00, 10.00	**100.00**, **100.00**
004_sugar_box	4.01, 3.60	**2.05**, **1.87**	25.00, 35.42	**70.69**, **66.95**	69.78, 61.50	**71.81**, **80.28**	0.00, 20.00	**100.00**, **100.00**
006_mustard_bottle	3.15, 3.09	**1.58**, **1.50**	46.06, 48.10	**71.62**, **77.99**	88.90, 77.33	**93.69**, **86.98**	25.00, 35.00	**100.00**, **100.00**
007_tuna_fish_can	**1.48**, **1.47**	2.17, 1.84	94.94, 92.12	**95.05**, **93.52**	95.52, 93.96	**97.44**, **95.45**	50.00, 35.00	**75.00**, **65.00**
008_pudding_box	2.78, 2.47	**1.97**,**2.10**	**92.05**, **88.06**	71.28, 73.27	95.21, **92.86**	**95.92**, 88.11	25.00, 20.00	**100.00**, **90.00**
011_banana	4.07, **3.46**	**3.23**,3.65	46.58, **43.96**	**66.48**, 42.47	**71.86**, **59.08**	69.47, 58.46	25.00, 30.00	**75.00**, **55.00**
019_pitcher_base	9.69, 9.54	**4.81**, **4.63**	0.00, 0.00	**25.00**, **35.14**	0.00, 0.00	**25.00**, **35.09**	0.00, 5.00	**100.00**, **95.00**
021_bleach_cleanser	5.23, 5.65	**2.20**, **2.63**	0.00, 34.72	**91.31**, **66.64**	46.66, 67.33	**94.29**, **91.05**	25.00, 30.00	**75.00**, **95.00**
036_wood_block	9.95, 10.41	**3.27**, **3.49**	0.00, 5.00	**47.09**, **30.34**	67.12, **61.95**	**70.59**, 57.58	25.00, 20.00	**75.00**, **80.00**
040_large_marker	2.26, 2.19	**1.05**, **0.91**	73.01, 75.48	**96.89**, **77.10**	74.01, 78.32	**97.89**, **83.01**	50.00, 50.00	**75.00**, **90.00**
Mean	4.76, 4.68	**2.36**, **2.39**	40.26, 44.66	**70.84**, **64.77**	70.54, 68.55	**81.25**, **77.08**	25.00, 25.50	**87.50**, **87.00**

with $\xi_j^k \in \mathbb{R}^6$ such that

$$T_j^{w,k} = T_j^{w,k}(\xi_j^k) = \begin{bmatrix} \exp((\xi^k_{j\,(3:6)})^\wedge) & \xi^k_{j\,(1:3)} \\ 0^T & 1 \end{bmatrix}, \quad (6)$$

where $(\xi^k_{j\,(3:6)})^\wedge \in \mathfrak{so}(3)$ is an element in the Lie-algebra of $\mathrm{SO}(3)$. Moreover, $K \in \mathbb{R}^{6\times 6}$ is a tuning matrix and $E(\xi)$ is the loss

$$E(\xi) = \frac{1}{L}\sum_{i=1}^{L} \|s_i(S_i^w) - t_i^w(T(\xi))\|^2, \quad (7)$$

where t_i^w is the i-th sensor candidate position at the pose $T(\xi)$, expressed in the world frame. In summary, the loss in Eq. (7) is used to minimize the distance between the candidate sensor positions and the actual ones.

In order to initialize the iterates, we set the rotational part of $T_{j,0}^w$ to a matrix uniformly sampled on the group of the rotations $\mathrm{SO}(3)$. As regards to the translational part, in practice we consider not just one hypothesis but several. Specifically, we sample 14 points at distance l_s from the geometric mean of the sensors positions $s_c = (s_1 + \cdots + s_L)/L \in \mathbb{R}^3$ in the directions of the vertexes and face centers of a cube centered in s_c. We also include the center s_c as one of the hypotheses. This choice is made to better explore the state space at initialization time.

We run the GD optimization routine in parallel for all the N poses for a maximum of k_{max} steps.

C. Physical reasoning-based pose ranking

Once the optimization process is done, we rank the resulting poses $\{T^w_{j,k_{max}}\}_{j=1}^N$ according to their loss and by penalizing poses where the object mesh is in collision with the mesh of the sensors, as they correspond to unfeasible configurations.

To this end, we use a physics engine in order to evaluate the penetration depths $d_{i,j}$ between the object mesh for each pose $T^w_{j,k_{max}}$ and the mesh of the sensor at each pose S_i^w. We then define the ranking score as follows:

$$R_j = E(\xi_j^{k_{max}}) + \max_i d_{i,j}, \quad (8)$$

i.e. we sum the GD final loss with the highest penetration depth between the object and the sensor meshes. We finally select the object pose T^w as the one with the least ranking score:

$$T^w = T^w_{j^*,k_{max}}, \quad j^* = \arg\min_j R_j. \quad (9)$$

IV. EXPERIMENTAL RESULTS

In this section we present the results of several experiments carried out both in simulation and in a real setting using the DIGIT vision-based tactile sensor [25]. We compare both quantitatively and qualitatively against a purely geometric baseline that is obtained by running our algorithm without the tactile-based selection mechanism described in Sec. III-A. In this way, the baseline acts as a point-set registration algorithm that tries to align the candidate sensors positions with the actual sensors positions by using only proprioception. Although our work is similar to [15], we could not compare to it as the software implementation is not available.

We use PyTorch [32] to train the autoencoder for tactile image reconstruction, the JAX [33] Python library to perform the GD-based optimization in parallel and the NVidia PhysX [34] middleware to evaluate the collision depths between object and sensors. Our software implementation is made publicly available for free with an Open Source license online[1].

For all the experiments, we used the following values for the parameters of the algorithm: $N = 5000$, $l_s = 0.2\,\mathrm{m}$, $k_{max} = 700$, $M = 2000$, $K = \mathrm{diag}(I_3 10^{-2}, I_3)$.

A. Experimental setup

Simulated setup We consider 10 objects from the YCB model set [26]. For each object, we select several object-sensor configurations with three sensors. The configurations are selected in order to explore several kind of contacts, e.g. with flat surfaces, edges, corners and rounded parts of object, when available. We remark that the poses of the sensors in the above configurations are different from those used to collect the database described in Sec. III-A. For each configuration, we place the object of interest within the Gazebo simulator at a known fixed pose, used as ground truth $T^{w,gt}$, and

[1]https://github.com/hsp-iit/multi-tactile-6d-estimation

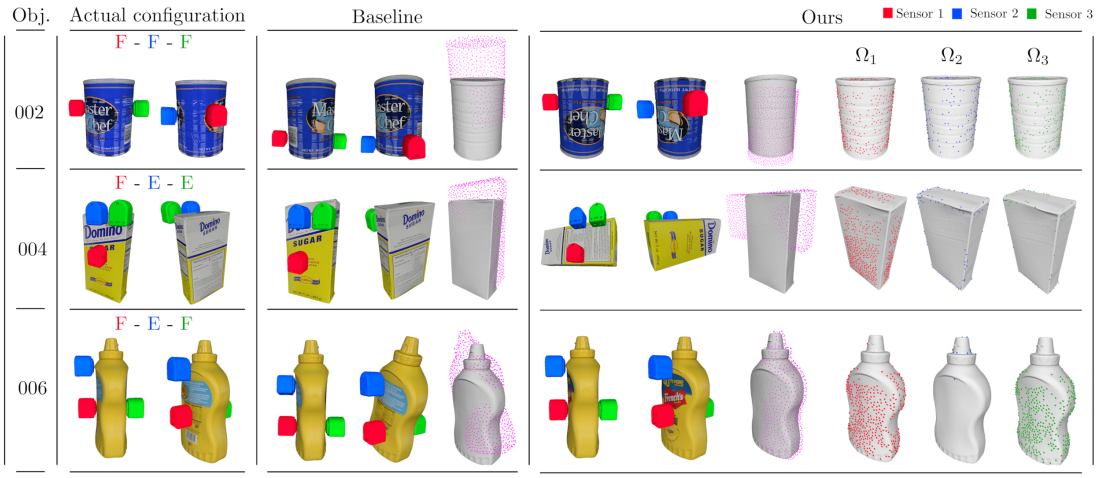

Fig. 4. Qualitative results in simulation for the objects "002_master_chef_can", "004_sugar_box" and "006_mustard_bottle". The letter "F" indicates contact with a flat surface, while "E" indicates contact along an edge of the object In this scenario, we use $L = 3$ tactile sensors.

we collect the L sensors poses S_i^w and images I_i. Sample configurations are shown in Fig. 4.

We evaluate the performance of our method against the baseline by considering the error in position with respect to the ground truth. We do not provide a pure rotational error as it would not be suitable in the presence of objects with symmetries. Instead, we include the standard ADI-AUC metric [1], with threshold set at $2\,\mathrm{cm}$, that jointly evaluates the positional and rotational errors while taking into account the symmetries of the objects. We also provide the ADI-AUC metric evaluated by setting the estimated position equal to the ground truth, in order to report on the rotational error separately. Moreover, we evaluate the ability of our method to estimate object poses that are compatible with the actual contacts. To this end, we report on the percentage of experiments where the object, at the estimated pose, is in contact with all the sensors in parts of the surface that are compatible with the real object-sensors configuration. We remark that, for each object, the results are averaged on all the experiments involving that object.

Real world setup For the experiments in a real scenario, we use a DIGIT sensor mounted on a 7-DoF Franka Emika Panda robot (see Fig. 5). We use the robot to touch the object with the sensor several times in order to reproduce the scenario of simultaneous contacts. The sensor 6D poses are obtained using the robot forward kinematics. In this case we used the Taxim example-based simulator [35] to simulate the tactile images required for training and to build the database. In our experiments, Taxim provided latent features that are better suited for this scenario, thanks to the calibration procedure that make rendered images more similar to those of the real sensor.

B. Results of the experiments in simulation

Pose accuracy Table I reports the positional error and the ADI-AUC metrics for the best pose and the five best poses, in the case $L = 3$. In the following we discuss the results taking into account the best pose.

Fig. 5. Real world experimental setup. On the left: the Franka Emika Panda robot with a DIGIT sensor mounted at the end-effector. On the right: a detail of the DIGIT sensor in contact with the object.

Considering all the objects on average, our method achieves the best performance according to all the metrics and gets an average positional error in the order of few centimeters. The positional error for the baseline is almost doubled on average. Moreover, the maximum error for the baseline almost reaches $10\,\mathrm{cm}$, for the objects "021" and "036", while with our method it stays below $5\,\mathrm{cm}$.

According to the ADI-AUC and the ADI-AUC (rotation) metrics, our method provides an increase in performance of approximately 76% and 15% on average, respectively.

Contact accuracy Table I reports the contact accuracy for both methods. As can be seen, our method outperforms the baseline on average and for each object separately. On average, our method increases the performance by 250%. This suggests that, even though the baseline might achieve a reasonable pose accuracy, most of the times the estimated poses are not compatible with the actual contacts between the object and the sensors. In this respect, we can consider our method as a way to boost the performance of a point-set registration-like algorithm using tactile information.

Qualitative results In Fig. 4, we provide qualitative results for several YCB objects. For each row, we provide two views of the actual object-sensors configuration and similar ones showing the estimate provided by the baseline and our method. We also show a direct comparison between the ground truth pose, represented with the solid gray mesh without texture, and the estimated pose, represented with the magenta point cloud. In the last three columns, the sets of candidate contact

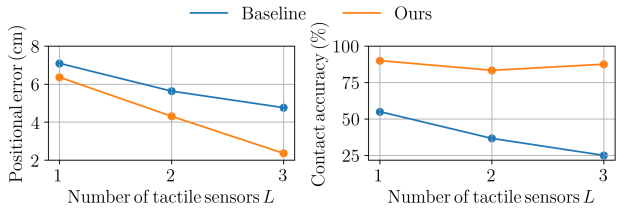

Fig. 6. Positional errors and contact accuracy when the number of tactile sensors varies from one to three.

points Ω_i are reported for each sensor.

Although the baseline can achieve reasonable pose estimates, in all the presented examples at least one sensor contacts the object in a part of the surface that is not compatible with actual object-sensors configuration. E.g., for the object "002", the second sensor should touch a flat surface, while it is in contact with an edge. For the object "004", both the second and third sensors should touch an edge, while they are in contact with a flat surface. A similar reasoning holds for the object "006".

On the other hand, the pose estimates provided by our algorithm are always compatible with the actual contacts. We remark that, for the object "004", the estimated rotation is quite different from the ground truth. This depends on the multimodality of the object poses given the measurements at disposal, i.e. the pose of the sensors and the tactile images.

Role of the number of sensors L In Fig. 6, we report on the performance of the algorithms when the number of tactile sensors varies from one to three. For both methods, the positional error decreases as the number of sensors increases. As regards the contact accuracy, while it is almost constant for our method, it decreases for the baseline when more sensors are used. This fact can be explained by the increase in the number of possible configurations of the sensors. While our method is able to filter them according to the tactile images, the baseline considers all of them, thus reducing the probability to find those that are compatible with the actual contacts between the object and the sensors.

C. Results of the experiments in the real world setup

In Fig. 7, we show qualitative results of the experiments from the real setup. As in Fig. 4, we show several views of the object at the estimated pose alongside the sensors, whose pose is provided by the robot forward kinematics. Instead, we do not compare against the ground truth object pose, as not available in this scenario.

When comparing against the baseline, our method estimates contact positions that are more compatible to the actual case. For example, in the case of object "006", the three sensors are touching an edge, a flat surface and another edge, respectively. In the case of the baseline, on the contrary, the three sensors are estimated to be in contact with a flat surface, an edge and another flat surface.

D. Computation times

The mean time required to execute the full pipeline of our method, averaged on all the experiments that we considered,

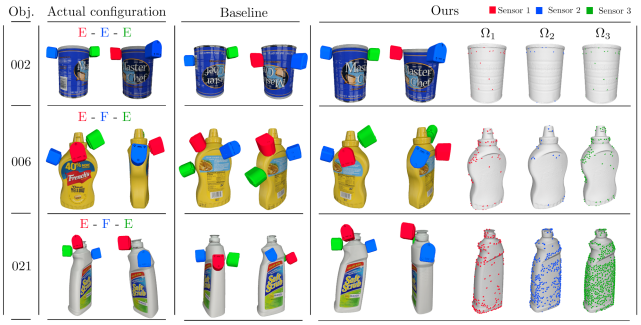

Fig. 7. Qualitative results on the experiments from the real setup for the objects "002_master_chef_can", "006_mustard_bottle", and "021_bleach_cleanser".

corresponds to 10.2s, 29.8s and 35.7s when using one, two or three sensors, respectively. For the baseline, these times increase to 14.5s, 43.6s and 71.9s. This outcome is expected as, for the baseline, the distance-based selection criterion in Eq. (4) needs to be checked for all the possible combinations of candidate sensor positions, i.e. M^L.

E. Limitations

At the moment, we do not completely exploit the information given by the sensor image, such as the portion of the sensor in contact, which could help mitigate the multimodality of the object poses given the measurements.

Secondly, the selection mechanism of the sets Ω_i can result in overly sparse or dense sets depending on the choice of the parameter δ_h, which needs an empirical tuning procedure.

In the real experiments, the sets Ω_i might differ from the expected ones due to differences between real and simulated images. For instance, the set Ω_2 for the object "006" in Fig. 7, corresponding to a flat surface, is much more sparse than the ideal one. Conversely, numerous outliers can be found while touching an edge, as in the set Ω_3 of object "021".

We finally remark that the proposed pose ranking criterion (see Sec. III-C) is based on the Cartesian distance between measured and optimized sensor positions and on the penetration depth. As a consequence, it does not necessarily reward poses corresponding to a lower 6D pose error (as for the object "004" in Fig. 4). This can also be seen in Table I, where the results of the best pose (#1) are not always better than the averaged ones (#1-5), as one might expect.

V. Conclusion

In this work, we presented an in-hand 6D object pose estimation method which combines proprioception with a generic number of vision-based tactile sensors. The core of the method relies on a latent encoding provided by an autoencoder that is trained completely in simulation. Experiments in simulation and in a real scenario demonstrate that the system allows us to estimate the pose of an object with touch alone, improving performance with respect to a geometric baseline.

As future work, we plan to better exploit the sensor information to reduce the multimodality and the estimation error even further. Moreover, we aim to use the pipeline for tracking purposes by using a stream of sensor images while the robot manipulates the object.

REFERENCES

[1] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[2] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects," in *Conference on Robot Learning*, 2018, pp. 306–316.

[3] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-Wise Voting Network for 6DoF Pose Estimation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 4556–4565.

[4] C. Song, J. Song, and Q. Huang, "HybridPose: 6D Object Pose Estimation under Hybrid Representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 431–440.

[5] Y. Lin, J. Tremblay, S. Tyree, P. A. Vela, and S. Birchfield, "Single-Stage Keypoint-Based Category-Level Object Pose Estimation from an RGB image," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 1547–1553.

[6] B. Wen, C. Mitash, B. Ren, and K. E. Bekris, "se(3)-TrackNet: Data-driven 6D Pose Tracking by Calibrating Image Residuals in Synthetic Domains," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 367–10 373.

[7] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "PoseRBPF: A Rao-Blackwellized Particle Filter for 6D Object Pose Tracking," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1328–1342, 2021.

[8] N. A. Piga, Y. Onyshchuk, G. Pasquale, U. Pattacini, and L. Natale, "ROFT: Real-Time Optical Flow-Aided 6D Object Pose and Velocity Tracking," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 159–166, 2022.

[9] M. Stoiber, M. Sundermeyer, and R. Triebel, "Iterative Corresponding Geometry: Fusing Region and Depth for Highly Efficient 3D Tracking of Textureless Objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6855–6865.

[10] Y. Lin, J. Tremblay, S. Tyree, P. A. Vela, and S. Birchfield, "Keypoint-Based Category-Level Object Pose Tracking from an RGB Sequence with Uncertainty Estimation," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 1258–1264.

[11] P. Sodhi, M. Kaess, M. Mukadanr, and S. Anderson, "PatchGraph: In-hand tactile tracking with learned surface normals," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2164–2170.

[12] S. Suresh, M. Bauza, K.-T. Yu, J. G. Mangelson, A. Rodriguez, and M. Kaess, "Tactile SLAM: Real-time inference of shape and pose from planar pushing," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 322–11 328.

[13] M. B. Villalonga, A. Rodriguez, B. Lim, E. Valls, and T. Sechopoulos, "Tactile object pose estimation from the first touch with geometric contact rendering," in *Conference on Robot Learning*. PMLR, 2021, pp. 1015–1029.

[14] M. Costanzo, "Control of robotic object pivoting based on tactile sensing," *Mechatronics*, vol. 76, p. 102545, 2021.

[15] T. Kelestemur, R. Platt, and T. Padir, "Tactile Pose Estimation and Policy Learning for Unknown Object Manipulation," in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 742–750.

[16] A. Sipos and N. Fazeli, "Simultaneous Contact Location and Object Pose Estimation Using Proprioceptive Tactile Feedback," *arXiv preprint arXiv:2206.01245*, 2022.

[17] J. Bimbo, S. Luo, K. Althoefer, and H. Liu, "In-Hand Object Pose Estimation Using Covariance-Based Tactile To Geometry Matching," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 570–577, 2016.

[18] G. Izatt, G. Mirano, E. Adelson, and R. Tedrake, "Tracking Objects with Point Clouds from Vision and Touch," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 4000–4007.

[19] P. K. Murali, A. Dutta, M. Gentner, E. Burdet, R. Dahiya, and M. Kaboli, "Active visuo-tactile interactive robotic perception for accurate object pose estimation in dense clutter," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4686–4693, 2022.

[20] 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 *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6203–6209.

[21] 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 Robotics and Automation Letters*, vol. 7, no. 2, pp. 2148–2155, 2022.

[22] F. von Drigalski, K. Hayashi, Y. Huang, R. Yonetani, M. Hamaya, K. Tanaka, and Y. Ijiri, "Precise Multi-Modal In-Hand Pose Estimation using Low-Precision Sensors for Robotic Assembly," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 968–974.

[23] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 598–10 608.

[24] 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.

[25] M. Lambeta, P.-W. Chou, S. Tian, B. Yang, B. Maloon, V. R. Most, D. Stroud, R. Santos, A. Byagowi, G. Kammerer *et al.*, "DIGIT: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 3838–3845, 2020.

[26] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and model set: Towards common benchmarks for manipulation research," in *2015 International Conference on Advanced Robotics (ICAR)*, July 2015, pp. 510–517.

[27] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "ShapeNet: An Information-Rich 3D Model Repository," *arXiv preprint arXiv:1512.03012*, 2015.

[28] N. Koenig and A. Howard, "Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sep 2004, pp. 2149–2154. [Online]. Available: https://ieeexplore.ieee.org/document/1389727

[29] S. Wang, M. Lambeta, P.-W. Chou, and R. Calandra, "TACTO: A Fast, Flexible, and Open-Source Simulator for High-Resolution Vision-Based Tactile Sensors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3930–3937, 2022.

[30] A. H. Barr, "Superquadrics and Angle-Preserving Transformations," *IEEE Computer graphics and Applications*, vol. 1, no. 1, pp. 11–23, 1981.

[31] S. Martin, M. Zoltan-Csaba, D. Maximilian, B. Manuel, and T. Rudolph, "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018, pp. 699–715.

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[33] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018. [Online]. Available: http://github.com/google/jax

[34] "NVIDIA PhysX for Developer," https://developer.nvidia.com/physx-sdk.

[35] Z. Si and W. Yuan, "Taxim: An Example-Based Simulation Model for Gelsight Tactile Sensors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2361–2368, 2022.