Tracking Objects with Point Clouds from Vision and Touch

Gregory Izatt, Geronimo Mirano, Edward Adelson, and Russ Tedrake
CSAIL, Massachusetts Institute of Technology, Cambridge, MA
Email: $\{gizatt, geronm, adelson, russt\}$@mit.edu

***Abstract*—We present an object-tracking framework that fuses point cloud information from an RGB-D camera with tactile information from a GelSight contact sensor. GelSight can be treated as a source of dense local geometric information, which we incorporate directly into a conventional point-cloud-based articulated object tracker based on signed-distance functions. Our implementation runs at 12 Hz using an online depth reconstruction algorithm for GelSight and a modified second-order update for the tracking algorithm. We present data from hardware experiments demonstrating that the addition of contact-based geometric information significantly improves the pose accuracy during contact, and provides robustness to occlusions of small objects by the robot's end effector.**

I. Introduction

The ability to perceive and control objects as they experience contact is a fundamental skill for any robot interacting with the world. Established approaches to manipulation tasks rely primarily on cameras and optical depth sensors to track object state. However, it is precisely when a robot's manipulator approaches an object that vision sensors are likely to be limited by occlusion. Incorporating tactile sensing into the pose tracker seems natural, but requires continued progress in both the tactile sensors and the algorithms that take advantage of their properties.

Many tactile sensors measure force at a single point or patch of contact, providing potentially rich dynamic information but limited geometric information. In this paper, we investigate the application of a tactile sensor—GelSight—which provides dense geometric information of objects that come in contact with its surface and can thus provide the localization data that is otherwise lost due to occlusion of distant optical depth sensors. Because of the smaller view area, GelSight can provide exceptionally fine geometric information, capturing surface features as fine as 2 microns [8], and it can simultaneously measure shear and slip [24]. We focus on utilizing this precise contact geometry to enable precise object localization for small manipulands.

This material is based upon work supported by NSF Contract IIS-1427050, a National Science Foundation Graduate Research Fellowship under Grant No. 1122374, as well as support from ABB, Draper Laboratory, MIT Lincoln Labs, Shell, and the Toyota Research Institute.

Stereographic, structured light, and LIDAR sensors have spurred fundamentally geometric point-cloud based approaches to robotic perception. Many variants of the Iterative Closest Point (ICP) algorithm have been developed to locate and track objects in point clouds [17]. In conjunction with ICP, signed distance functions (SDF) have proven a valuable tool for reasoning in a continuous and smooth way about the geometries of objects and scenes, and have proven invaluable in object tracking, and simultaneous localization and mapping (SLAM) [2] [20] [15] [23].

In this paper, we show that the contact geometry information from the GelSight contact sensor is compatible with traditional ICP-based tracking techniques. By constructing an object tracker that utilizes both precise, local contact geometry from GelSight, and large-scale point-cloud data, we achieve contact-aware object tracking that utilizes tactile data to output greatly refined pose estimates. We provide experimental results showing quantitative improvement to estimator performance when contact geometry information is added, and demonstrate the use of our system to track the grasping and manipulation of a small tool.

II. Related Work

ICP informs many modern visual tracking methods. These methods have been extended to support articulated object collections with internal joints. Klingensmith, et al. demonstrate closed-loop servoing using articulated ICP for online pose estimation [9], and Hebert, et al. utilize articulated ICP for simultaneous manipulator and manipuland tracking [5]. Following a similar path, the Dense Articulated Real-Time Tracking (DART) framework [20] performs articulated object tracking from dense depth data in real time by leveraging the signed distance function (SDF) to efficiently align an articulated object model to an incoming stream of point cloud data, while balancing a free space term. These online tracking techniques show excellent performance when fed sufficiently rich data. However, they are vulnerable to occlusion during manipulation, when the robot's hand is likely to cover the object being manipulated, and are likely to lose tracking without an additional sensing modality or physically-derived constraint.

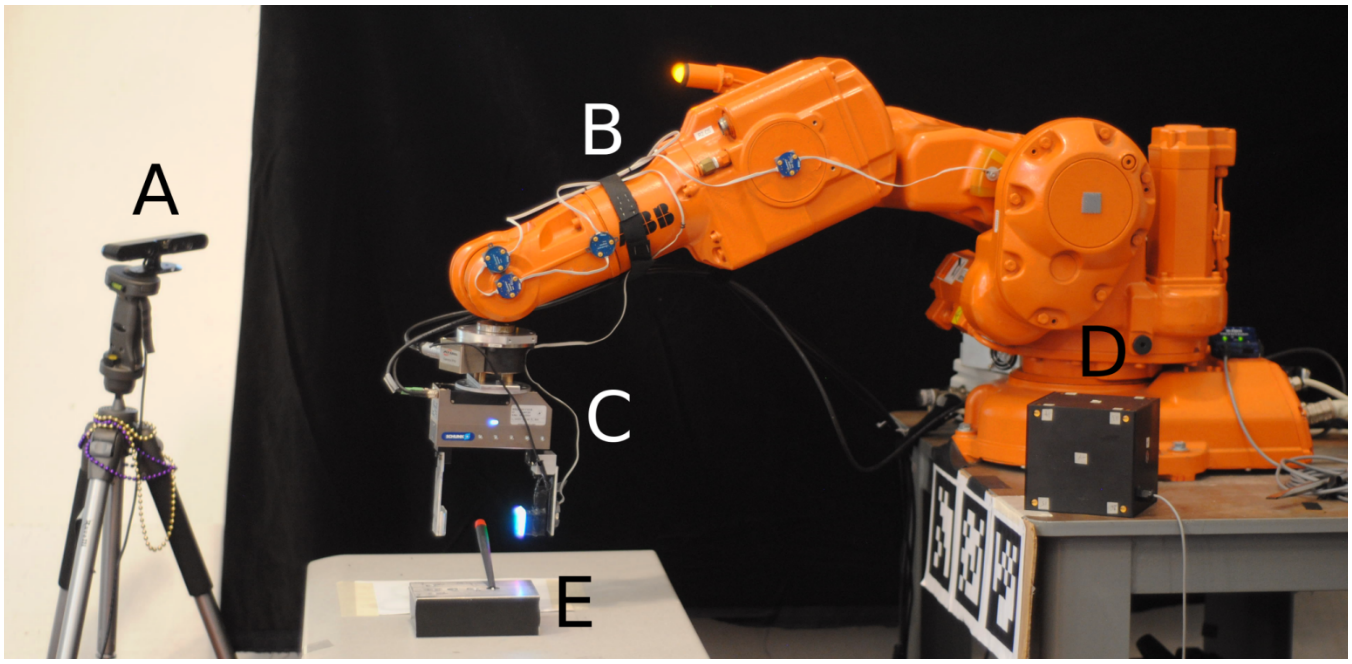

Fig. 1. The setup used in our experiments consists of an Asus Xtion RGB-D camera (**A**) observing a 6-DOF ABB IRB-140 arm (**B**). The end effector is a Schunk WSG-50 parallel gripper with a GelSight-enabled custom set of fingers (**C**). A cube with attached optical markers (**D**), and a small screwdriver and rectangular holster (**E**) are used as manipulands.

To address this issue, DART was extended to include nonpenetration and binary contact constraints, which are made continuous and efficiently enforceable via further application of the SDF [19]. A parallel body of work employs particle filters (PFs) to tackle exactly the ambiguity and nonlinearity often inherent in contact state estimation. Koval, et al. take advantage of the manifold structure of the state space of contact to greatly reduce the critical particle starvation issue facing PFs during contact events by resampling directly from the contact manifold [11]. Zhang and Trinkle tackle the same problem by using a constraint-based physical model to enforce that particle updates stay physically feasible with respect to nonpenetration and contact forces [25]. Li, et al. instead use a PF to track discrete contact modes in the contact graph, while performing continuous state estimation at each particle with a Kalman filter using a process update derived from the particle's contact mode [14]. While particle filters have better theoretical ability to represent the nonlinear and potentially multimodal state distributions that arise through contact events, they face difficult scaling issues even under these optimizations. Klingensmith, et al. make progress on this scaling by leveraging SDFs to avoid expensive explicit parameterization of the contact manifold [10].

Tactile sensors take a wide variety of forms, but few allow for recovery of dense local geometry. Modular sensors designed to be used as fingertips include the SynTouch BioTac [21], which discriminates contact over the entire sensor surface, and the RightHandRobotics Takktile sensor [18]. Both of these sensors output a single pressure signal, though the Takktile sensor can be purchased in an 8mm tiled layout to sense rough contact location. Sensors with greater ability to resolve geometry are under active development. Jamali, et al. discusses the design of sensing skin for the iCub robot's fingertips, which utilize tiled force sensors at approximately 1mm spacing on a flexible PCB [6]. Patel and Correll present an alternate sensor design that combines distance and force elements [16].

We observe that there is a bias towards contact discrimination rather than recovery of dense contact geometry in the majority of these sensors, though cutting-edge sensing skins blur this distinction. The tracking algorithms surveyed above were tailored to these discriminatory sensors. Schmidt, et al. support a binary contact detection signal as an input, and estimates contact locations that explain the binary contact detections [19]. The contact manifold [11] [10] and contact-mode switching [14] approaches are natural when used with a discriminative sensor, but do not extend as naturally to dense geometric contact information.

A key component of our solution to these localization challenges is the GelSight touch sensor. This sensor is capable of producing a rich contact depth map in the vicinity of a contact, which is ideal for small-scale geometric localization of objects [7] [12] [16]. Li, et al. demonstrate a GelSight texture recognition pipeline for localizing objects in-hand, which they use to accomplish a precise peg-in-hole task [13]. By incorporating the dense geometric information from a GelSight sensor with an ICP-based articulated object tracker, we build upon this work by offering a more general estimation pipeline which supports arbitrarily articulated models and non-planar contact surfaces.

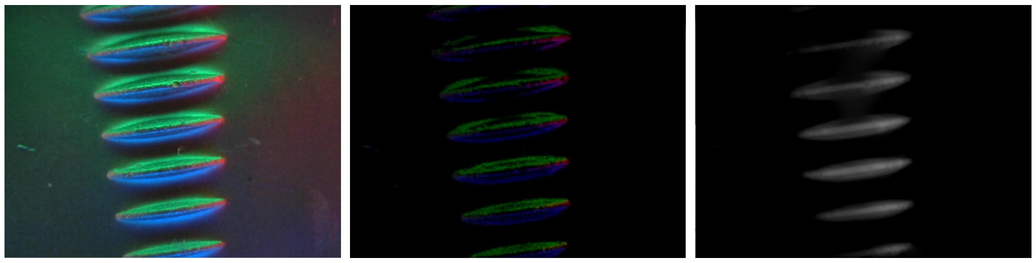

Fig. 2. **Top:** Raw GelSight image of threads on a bolt. **Middle:** Gradient image generated by lookup table after calibration. **Bottom:** Final depth map.

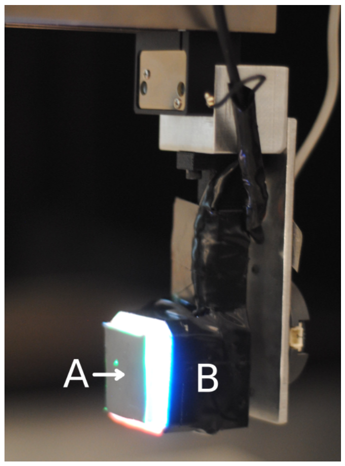

Fig. 3. A GelSight sensor mounted as a finger on a gripper. The elastomer surface (**A**) deforms when pushed against other objects. Deformations of the surface are illuminated from multiple sides and captured with a webcam inside of the sensor housing (**B**).

III. Sensor Overview

Our tracker takes input from two primary sensors: a structured-light dense RGB-D camera, and a GelSight contact geometry sensor.

The GelSight sensor (Figure 3) consists of a thin elastomer observed by a conventional color camera. The camera captures deformations of the elastomer when the elastomer is pressed against an object. The GelSight sensor produces an RGB image which gives geometric information within a $11.5 \times 15 \times 2$ mm volume. The sensor surface is backlit by a different color of light from three sides, such that different slopes of the sensor surface correspond to different colors on the RGB image. By collecting the raw RGB images formed by contacting the sensor with known calibration surfaces, we learn a mapping from RGB points $\in \Re^3$ to depth map gradients $\in \Re^2$.

Following the technique of Li, et. al for producing a depth map from the GelSight images, we use a $16 \times 16 \times 16$ binned lookup table which maps the color of background-subtracted pixels to their corresponding gradient values [13]. For training data, we roll a 2.5mm-diameter ball bearing around the sensor and use machine vision to detect its location automatically in each image. This technique allows us to easily generate enough ground truth data to learn the mapping. At runtime, we use this lookup table to determine the gradients in the full-resolution image, then use a modified Poisson integration on a downscaled gradient image to obtain the depth map.

Poisson integration of the gradient image $\mathbf{g}^{h \times w \times 2}$ into the final depth map $\mathbf{p}^{h \times w \times 1}$ is performed via a large but sparse unconstrained least squares optimization over the $h \times w$ pixels $\mathbf{p}$ of the final depth map. For every horizontally neighboring pair of points $p_{x,y}$ and $p_{x+1,y}$, we add a horizontal gradient violation penalty $||(p_{x+1,y} - p_{x,y}) - g_{x,y,1}||^2$. Similarly, for every vertically neighboring pair of points $p_{x,y}$ and $p_{x,y+1}$, we add a vertical gradient violation penalty $||(p_{x,y+1} - p_{x,y}) - g_{x,y,2}||^2$. An additional set of terms enforces boundary conditions by penalizing $k_e \times ||p_{x,y}||^2$ for all x, y on the image boundary, using a gain k_e to weight this penalty against the integration penalties.

We perform this conversion in real time using OpenCV [1] and Eigen [4]. We have empirically found $k_e = 1.0$ to yield good results. Our pipeline can attain a resolution of 256-by-186 tactels a rate of 12 Hz.

IV. Tracking Algorithm

Our tracking algorithm takes as input a continuous stream of RGB-D images from an off-the-shelf dense depth sensor, and depth images from the GelSight sensor. We take inspiration from the DART tracking system of Schmidt, et al. [20] [19] and construct a single-hypothesis tracker based on an EKF. We use the same formulation, but offer novel optimization strategy. The formulation that follows in this section is repeated from the original presentation of DART in order to motivate the subsequent description of our optimization strategy.

A. *Modified EKF Formulation*

At a time step k, we estimate the state x_k and its variance Σ_k. For us, x_k collects positions and velocities, including floating base translations and rotations and joint angles.

Following a standard EKF formulation, we can use a dynamic model of the scene to generate a *predicted* state and variance $\bar{x}_k, \bar{\Sigma}_k$, using the previous estimated state x_{k-1} and any relevant control inputs u_{k-1}:

$$\bar{x}_k = f(x_{k-1}, u_{k-1})$$
$$\bar{\Sigma}_k = J(x_{k-1}, u_{k-1})\Sigma_{k-1}J(x_{k-1}, u_{k-1})^\top + W$$

Here, $f(x_{k-1}, u_{k-1})$ is the process update, J its Jacobian, and W additive process error. The simplest model would be to assume the state never changes and the variance slowly increases; this corresponds to using $f(x_{k-1}, u_{k-1}) = x_{k-1}$ and W nonzero.

A standard EKF would call for the measurement update to be performed by computing a predicted measurement $h(\bar{x}_k)$ and the measurement residual $\tilde{y}_k = z_k - h(\bar{x}_k)$ using forward measurement models. However, the forward measurement model $h(\bar{x}_k)$ is discontinuous in the case of a camera, and would yield poor gradients and an ineffective approximate Kalman gain. Thus, instead of the standard form, we write the measurement update as a direct optimization of system state over measurement probabilities derived from our sensors. Using θ our decision variable, we write:

$$x_k = \underset{\theta}{\operatorname{argmin}}[-\log(p(z_k|\theta)) + (\theta - \bar{x}_k)^\top \bar{\Sigma}_k^{-1}(\theta - \bar{x}_k)]$$
$$\Sigma_k = H(x_k)^{-1}$$

Here, generalized sensor readings for time step k are written z_k. Since x_k is a maximum likelihood estimate given the negative log-likelihood function above, the variance update takes the form of the inverse of the Hessian $H(x_k)^{-1}$ of that negative log-likelihood function.

The specific optimization problem we will solve depends on the construction of $p(z_k|\theta)$ for our particular set of sensors.

B. Measurement model for point clouds: positive returns

A depth sensor produces a list of pixels $im^{meas} = \{pixel_i^{meas} \in \Re\}$, and from each we can calculate a point in space $pt_i^{meas} \in \Re^3$ using the camera calibration.

Following DART, we will suppose that the likelihood of a point is normally distributed with respect to the signed distance to the closest surface (signed distance function, *SDF*), which depends on the system state θ. We assign a variance of σ reflecting the depth sensor noise characteristics:

$$p(pt_i^{meas}|\theta) = Ke^{-SDF(pt_i^{meas};\theta)^2/\sigma^2}$$
$$K = \frac{1}{\sqrt{2\pi\sigma^2}}$$

The likelihood of the complete image combining all pixels (indexed by i) is

$$p(image|\theta) = \prod_i p(pt_i|\theta)$$

After taking the negative log likelihood, the expression simplifies:

$$-\log(\prod_i p(pt_i^{meas}|\theta)) = \sum_i -\log(p(pt_i^{meas}|\theta)) =$$
$$\sum_i -\log(Ke^{-SDF(pt_i^{meas};\theta)^2/\sigma^2}) =$$
$$\frac{1}{\sigma^2}\sum_i SDF(pt_i^{meas};\theta)^2 + \log(K)$$

We drop the constant term $\log(K)$, as it has no dependence on our optimization variable θ.

C. Measurement model for point clouds: free space

As pointed out by Ganapathi, et al. [3], for each point pt_i in the point cloud, we know that there must be clear line-of-sight between the camera origin and that point. Thus, we know that no surface on the proposed model can lie between that point and the camera. If, for a given proposed model, we produce a simulated depth image as a collection of depths $im^{sim} = \{pixel_i^{sim}\}$, then we want to constrain $pixel_i^{sim} \geq pixel_i^{meas}$. Directly constraining this value yields poor performance, as the simulated depth returns have sharp discontinuities around object edges which hinder optimization. As such, Ganapathi, et al. and Schmidt, et al. instead partition $\Re^3$ into space known to be free, and space out of sight of the camera, and constrain all points on the surface of the proposed model to lie in the second partition [3] [20]. The partitioning surface, S_{obs}, is defined by the points in the measured point cloud; free space lies in front of S_{obs}, and out-of-sight space lies behind it.

We suppose that the probability of simulated depth point $pixel_i^{meas}$ is constant in out-of-sight space, and decreases with distance to the out-of-sight space. To calculate this, we will create another distance function, DF_{obs}, which yields the distance a given point has to move for it to leave free space.

Given DF_{obs}, we specify the probability density to be

$$p(pt_i^{sim}|im^{meas}) \propto Ke^{-DF_{obs}(pt_i^{sim})^2/\sigma^2}$$

Following similar steps as for the positive return case, computing probability over all points in the simulated depth image, taking the negative log likelihood, and dropping the constant term gives

$$-\log(\prod_i p(pt_i^{sim}|im^{meas})) = \frac{1}{\sigma^2}\sum_j DF_{obs}(pt_j^{sim};\theta)^2$$

D. Likelihood model for nonpenetration

During manipulation experiments, we observed a need to further constrain estimates to be physically feasible with respect to penetration. Inspired by DART, we implement this using a very similar distance-function-based penalty to that used in the free space constraint. For each point pt_i^{surf} sampled from the surface of an object, we suppose that the probability density falls off with the penetration distance DF_{pen} into the surfaces of the rest of the robot in configuration θ.

$$p(pt_i^{surf}|\theta) \propto Ke^{-DF_{pen}(pt_i^{surf};\theta)^2/\sigma^2}$$

As before, computing probability over all points sampled from the surfaces of objects of interest, taking the negative log likelihood, and dropping the constant term gives

$$-\log(\prod_i p(pt_i^{surf}|\theta)) = \frac{1}{\sigma^2}\sum_j DF_{pen}(pt_j^{surf};\theta)^2$$

V. Optimization

At each step, we compute a measurement update given the latest depth image from the RGB-D sensor, a depth image from the GelSight sensor, and the last estimated state θ. Written out in full, this update is:

$$\begin{aligned} x_k = \underset{\theta}{\operatorname{argmin}} \\ & \frac{1}{\sigma_{kinect}^2} \sum_{\text{kinect pts}} SDF(pt_i^{meas};\theta)^2 \\ & + \frac{1}{\sigma_{kinect}^2} \sum_{\text{kinect pts}} DF(pt_i^{sim};\theta)^2 \\ & + \frac{1}{\sigma_{gelsight}^2} \sum_{\text{gelsight pts}} SDF(pt_i^{meas};\theta)^2 \\ & + \frac{1}{\sigma_{gelsight}^2} \sum_{\text{gelsight pts}} DF(pt_i^{sim};\theta)^2 \\ & + \frac{1}{\sigma_{nonpen}^2} \sum_{\text{surface pts}} DF_{pen}(pt_i^{surf};\theta)^2 \\ & + (\theta - \bar{x}_k)^\top \bar{\Sigma}_k^{-1} (\theta - \bar{x}_k) \end{aligned}$$

As written, this optimization is nonlinear. It is particularly tough because changing θ changes $SDF(pt_i^{meas};\theta)$, pt_j^{sim}, $DF_{obs}(pt_i^{sim})$, and $DF_{pen}(pt_i^{surf})$ in complex ways depending on the shape of the object surface. We solve this problem by iteratively constructing and solving approximating unconstrained quadratic programs (QPs).

A. Approximate minimization of $SDF(pt_i^{meas};\theta)^2$

In every iteration, we calculate the closest point $\hat{pt}_i$ and corresponding body $body_i$ to pt_i^{meas} in our model in configuration x_{k-1}. These closest point calculations are performed directly via the Bullet collision library, working on convex decompositions of the robot's collision geometry. We observe that locally, $SDF(pt_i^{meas};\theta)^2 \approx ||pt_i^{meas} - \hat{pi}_i||^2$. Using the Jacobians J_{cam} at the camera and J_{body_i} at $body_i$ computed via forward kinematics, we can minimize that term by finding

$$\underset{\theta}{\operatorname{argmin}} ||(pt_i^{meas} - \hat{pt}_i) + (J_{cam} - J_{body_i})(\theta - x_{k-1})||^2$$

The minimizer for this expression corresponds to θ that moves both the camera and $body_i$ to place the measured point pt_i^{meas} on the body's surface.

B. Approximate minimization of $DF_{obs}(pt_i^{sim};\theta)^2$

For pt_i^{sim} on the surface of $body_i$, we calculate the closest point $\hat{pt}_i$ to pt_i^{sim} that is on or behind S_{obs}. Taking inspiration from Ganapathi et al. [3], we compute DF_{obs} and find this closest point efficiently by decomposing DF_{obs} into components perpendicular and parallel to the camera view ray. This decomposition allows us to avoid computing the full 3D distance function to S_{obs}. We calculate a 2D distance function finding the nearest pixel of the image for which $pixel_j^{sim} \geq pixel_j^{meas}$, which indicates how far we would have to move $body_i$ laterally for it to leave free space; and a 1D distance function, which is simply $pixel_j^{meas} - pixel_i^{sim}$, which indicates how far back we would have to push $body_i$ for it to leave free space. We set $\hat{pt}_i$ to the shorter of these two

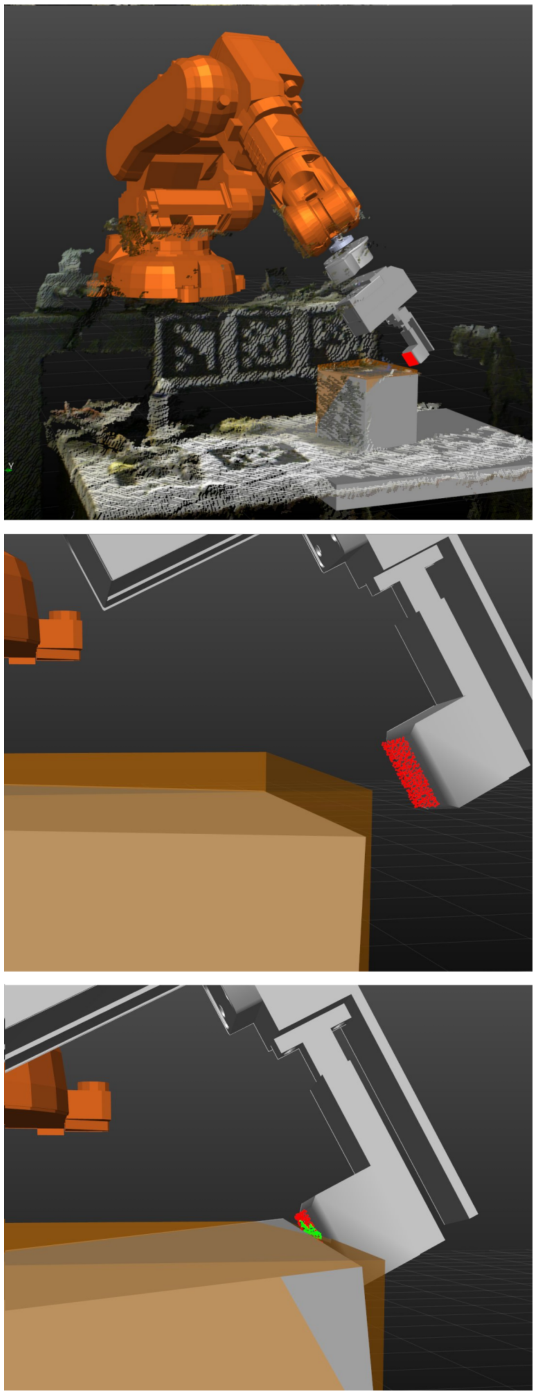
Fig. 4. **Top:** We use a GelSight sensor mounted on the end of a 6-DOF arm to manipulate a simple object. The benchtop is observed by an RGB-D sensor, and ground truth manipulator and object positions are provided by an external motion capture system. **Middle:** The object pose estimate (solid gray) from RGB-D data, alongside the ground truth object pose (transparent orange). A 1-cm vertical bias is injected into the RGB-D data for demonstrative purposes, causing the object pose estimate to be approximately 1cm low. **Bottom:** When the GelSight sensor is brought in contact with the object, the dense contact geometry information is used to improve the object pose estimate, correcting the 1cm bias in the vicinity of the contact.

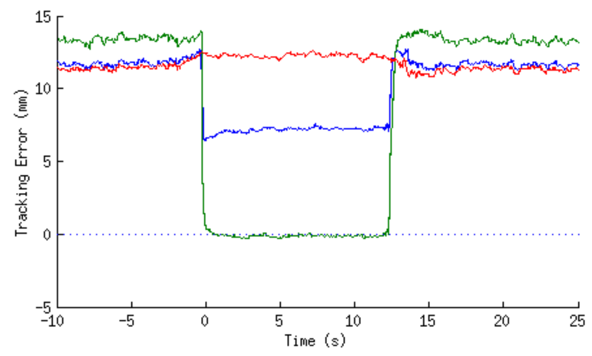

Fig. 5. Tracking error relative to the hand pose, in the vertical axis, as judged against ground truth, of the simple object during manipulation by the GelSight manipulator. **Red**: relative tracking error with no GelSight data used. **Blue**: relative tracking error to the object's centroid when GelSight data is used. **Green**: relative tracking error to the near edge of the object when GelSight data is used. A 1cm vertical bias was injected into the RGB-D point cloud data, causing the object tracking to exhibit a consistent, approximately 1cm bias in the absence of additional information. When the GelSight is brought in contact with the object (at $t = 0$), the contact geometry information counteracts this bias and improves tracking performance significantly. The tracking error returns when the contact is removed (at $t = 13$). The improvement is strongest in the vicinity of the contact sensor.

correspondences. We observe that locally, $DF_{obs}(pt_i^{sim};\theta)^2 \approx ||pt_i^{sim} - \hat{pt}_i||$. Again using the Jacobians J_{cam} and J_{body_i} at the camera and $body_i$, we can minimize that term by finding

$$\underset{\theta}{\operatorname{argmin}}\, ||(pt_i^{sim} - \hat{pt}_i) - (J_{cam} - J_{body_i})(\theta - x_{k-1})||^2$$

The minimizer for this expression corresponds to θ that moves both the camera and $body_i$ to place the simulated depth point pt_i^{sim} out of the measured free space.

C. *Approximate minimization of $DF_{pen}(pt_i^{surf};\theta)^2$*

For pt_i^{surf} on the surface of $body_i$, we calculate the closest point $\hat{pt}_i$ to pt_i^{surf} outside of all other bodies. Locally, $DF_{pen}(pt_i^{surf};\theta)^2 \approx ||pt_i^{surf} - \hat{pt}_i||$. Again using the Jacobians J_{cam} and J_{body_i} at the camera and $body_i$, we can minimize that term by finding

$$\underset{\theta}{\operatorname{argmin}}\, ||(pt_i^{surf} - \hat{pt}_i) - (J_{cam} - J_{body_i})(\theta - x_{k-1})||^2$$

The minimizer for this expression corresponds to θ that moves $body_i$ to move the point pt_i^{surf} to the surface of the object it is penetrating.

D. *Solution*

All of these approximate minimizers are unconstrained QPs in θ. We solve this QP online by solving the necessary and sufficient conditions for optimality as a linear system using QR Factorization with column pivoting in Eigen [4].

VI. Experimental Results

We have implemented this tracking framework, and present experimental results of the tracker running on the testbed documented in Figure 1. We employ a 6-DOF manipulator with the GelSight sensor mounted as finger on a parallel gripper. We observe the scene with an Asus Xtion PRO LIVE RGB-D camera, and use an additional optical motion capture system to recover ground truth for one experiment. The tracker runs as a single thread on a high-end desktop computer, and updates between 10 and 30 Hz depending on the complexity of the scene. To increase the tracking performance, we calibrate the RGB-D sensor position with an AprilTag of known global position, and do not estimate the RGB-D sensor position online. The collective state of the robot arm, attached GelSight sensor, table surface, and manipulated objects are estimated online by our tracker during teleoperation. This state includes multiple pin joint articulations within the arm, and floating base articulations for every object.

We present two experiments. The first demonstrates that contact information from the GelSight sensor can be used to quantitatively improve tracking performance for manipulation. The second employs our tracking framework to track manipulation of a small tool, and demonstrates qualitative improvement of tool pose during the manipulation. Both experiments run the same code, with the only difference being minor parameter changes made between experiments to increase the stability of the tracking of the small tool.

A. *Demonstrating Quantitative Tracking Improvement*

There are many sources of bias when fitting an object to a point cloud, including the use of inaccurate object models and poor camera calibration. However, as argued in Schmidt, et al. [19] it is not global object tracking accuracy that matters, but rather the accuracy of the estimated transform between the robot's hand and the object. A tactile sensor mounted directly on the hand has the potential to dramatically reduce this relative error. To demonstrate this, we use our tracker to estimate the pose of a robot arm and simple object before and after contacting the object with the GelSight sensor. To make the effect of the GelSight data more clear, we induce a 1cm vertical error in the hand-object relative pose by injecting a 1cm vertical bias in the estimated RGB-D camera position. When the GelSight reports contact with the box, the tracker updates the estimated object pose to better explain the detected contact geometry (Figure 4). This correction dramatically reduces the hand-object relative tracking error (Figure 5).

B. *Demonstrating Small Tool Manipulation*

Grasping and using small tools is a difficult task for many robots due to the difficulty of pose estimation of the tool in the robot's hand. When the hand closes on an object, the object is occluded from the external high-resolution vision sensors that could otherwise have been used to estimate its pose. This occlusion poses a serious limitation to the use of tools like screwdrivers, whose precise in-hand pose is critical for effective use. We demonstrate our tracker providing a pose estimate for a small screwdriver while the robot is teleoperated through extraction of the screwdriver from a holster. This maneuver involves significant contact between the screwdriver and holster, which causes the in-hand pose of the screwdriver to shift continuously. Our tracker maintains an accurate pose

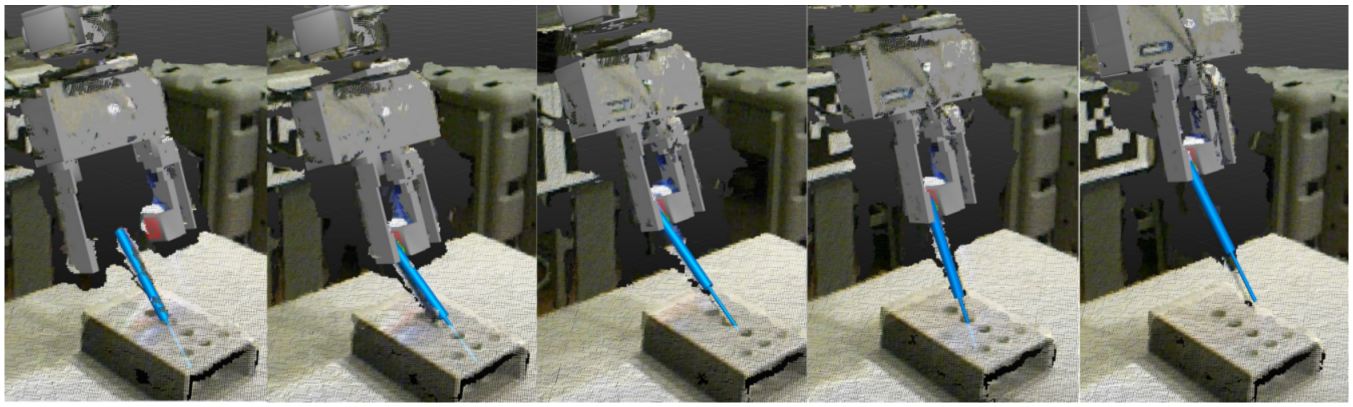

Fig. 6. A GelSight sensor on a parallel gripper on the end of a 6-DOF arm is used to manipulate a small screwdriver via teleoperation of the arm. Rendered robot and object pose estimates are overlayed with the point cloud information from the RGB-D camera. The GelSight sensor surface is shown in red where the depth is less than a threshold (indicating no contact), and green where depth is above the threshold (indicating contact). Our system successfully tracks the position of the screwdriver throughout a sequence of manipulations to remove the screwdriver from a holster. This manipulation involved sigificant contact between the screwdriver and holster, and caused the grasp of the screwdriver to shift significantly.

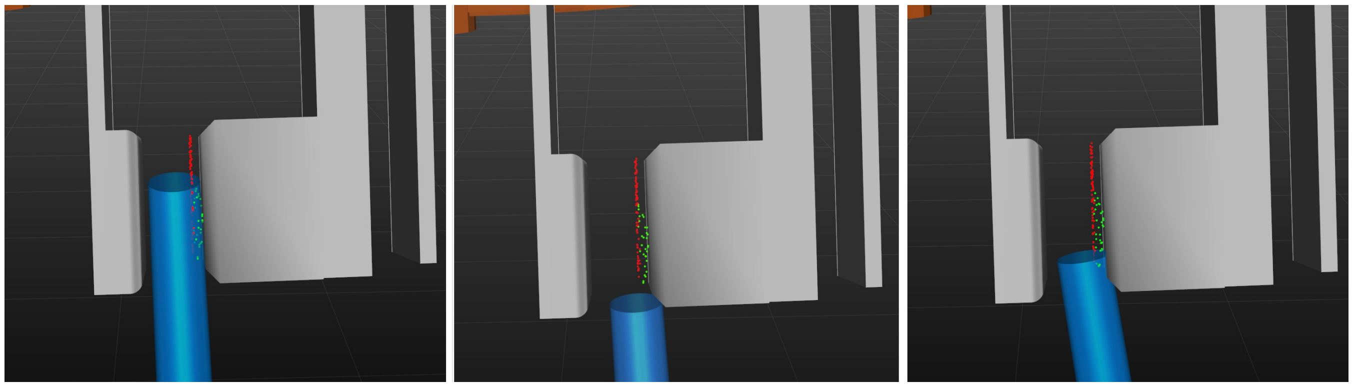

Fig. 7. The use of precise contact geometry information from the GelSight sensor enabled a significantly more precise pose estimation (**Left**) than was achieved by the same tracker with GelSight disabled (**Middle**), or by the same tracker with GelSight data and nonpenetration constraints disabled (**Right**). When the screwdriver was in the gripper, the grasped section was occluded from the RGB-D camera, preventing an RGB-D-only tracker from producing an accurate fit. All three estimates pictured are the stable final pose produced by the tracker after the same manipulation sequence.

estimate of the tool throughout the procedure (Figure 6). This experiment utilizes the same code as the qualitative tracking experiment, with the only parameter tweaks being adjustment of the weighting of the dynamics model and RGB-D camera data. These parameters were reduced in this experiment to improve robustness to transient outliers in the RGB-D data. We provide, for comparison, tracking results acquired by running the same tracker without GelSight data, and by running the same tracker with neither GelSight data nor nonpenetration constraints (Figure 7). Tracking fails in both cases due to a combination of occlusion and the tight fit of the screwdriver in the hand.

Side-by-side tracking performance from this experiment is visualized in the accompanying video.

VII. Discussion

Existing contact-aware object trackers work hard to incorporate sparse tactile data from discriminative contact sensors. Koval, Klingensmith, and others calculate contact manifolds on which poses must lie when a binary contact sensor is active [11] [10], and Schmidt et. al were forced to estimate the contact position as an additional state to incorporate a binary contact detector into DART [19]. Object tracking algorithms that operate on point clouds, in comparison, have proven more scalable and mature. The accurate, dense, and fundamentally geometric output of the GelSight sensor affords an opportunity to apply these point cloud algorithms directly to a tactile sensor, thus circumventing issues associated with sparse contact sensing.

Our quantitative tracking experiment demonstrates that the inclusion of GelSight data into an articulated object tracker can significantly improve relative hand-object pose estimates. Because the tactile sensor sits directly at the interface between end effector and the object being manipulated, it is expected to greatly decrease pose-tracking error during contact. Our experimental data meets this expectation: the GelSight-enabled tracker is able to recover from significantly inaccurate point cloud data once contact is made, with the relative pose error at the contacted edge falling from greater than 1 centimeter to below 1 millimeter. The relative pose error at the object's centroid is also reduced, but unlike the edge being contacted, the error does not fall to zero, because the contact geometry information is local to the contact location and provides too little additional information about the pose of the rest of the object to overpower the biased RGB-D data. The small tool manipulation example demonstrates a practical use of this tracking technique, and highlights the importance of dense geometric tactile sensing as a tool for fighting occlusion. The control cases which ignored the GelSight data were consistently unable to accurately localize the tool in the hand, because the gripper occluded the part of the tool inside of the grasp.

While our system benefits from the simplicity of treating dense geometric tactile data as a point cloud, our approach has limitations that will require further work to resolve. Principal among these limitations is that the contact geometry information available is typically small in volume. During a grasp on an arbitrary object, the contact volume is likely to encompass only a small fraction of the object's total surface. As many small regions on the object's surface are likely to look similar to one another, it is easy for the tracker to fall into local minima. Thus, in the experiments in this paper, we rely on the RGB-D data to provide a strong enough prior to provide adequate initial guesses for our tracker to converge when contact is made. There are many potential solutions to this problem, including the extension to external initialization by a single-shot pose estimator, multi-hypothesis tracking, increasing sensor depth and size, and the introduction of texture as an additional element in the measurement model [13].

VIII. Conclusion

We extend the state-of-the-art object tracker DART to fuse point cloud information from an RGB-D camera with accurate and dense geometric contact data from a GelSight sensor. By focusing on a contact sensor as a source of geometric data, we can leverage dense tactile information identically to conventional point cloud data within the articulated object tracker. The application of our tracking system to fine manipulation tasks shows that the inclusion of dense and accurate tactile information is effective at solving occlusion problems. We believe that geometric sensors like GelSight used in combination with point cloud object tracking techniques will enable the execution of previously unachievable tasks spanning small parts assembly, tactile exploration and search, and grasping of soft and novel objects.

References

[1] Bradski, G. The OpenCV Library, in *Dr. Dobb's Journal of Software Tools*, 2000.

[2] Bylow, E., Sturm, J., Kerl, C., Kahl, F., & Cremers, D. (2013). Real-time camera tracking and 3d reconstruction using signed distance functions. In *Robotics: Science and Systems (RSS) Conference 2013* (Vol. 9). Robotics: Science and Systems.

[3] Ganapathi, V., Plagemann, C., Koller, D., & Thrun, S. (2012, October). Real-time human pose tracking from range data. In *European conference on computer vision* (pp. 738-751). Springer Berlin Heidelberg.

[4] Guennebaul, G., et al. the Eigen Library, http://eigen.tuxfamily.org.

[5] Hebert, P., Hudson, N., Ma, J., Howard, T., Fuchs, T., Bajracharya, M., & Burdick, J. (2012, May). Combined shape, appearance and silhouette for simultaneous manipulator and object tracking. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (pp. 2405-2412). IEEE.

[6] Jamali, N., et al. A New Design of a Fingertip for the iCub Hand, in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[7] Johnson, M. K., & Adelson, E. H. (2009, June). Retrographic sensing for the measurement of surface texture and shape. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[8] Johnson, M. K., Cole, F., Raj, A., & Adelson, E. H. (2011, August). Microgeometry capture using an elastomeric sensor. In *ACM Transactions on Graphics (TOG)* (Vol. 30, No. 4, p. 46). ACM.

[9] Klingensmith, M., Galluzzo, T., Dellin, C. M., Kazemi, M., Bagnell, J. A., & Pollard, N. (2013). Closed-loop servoing using real-time markerless arm tracking. In *Proceedings of the ICRA Humanoids Workshop.*

[10] Klingensmith, M., Koval, M. C., Srinivasa, S. S., Pollard, N. S., & Kaess, M. (2016). The Manifold Particle Filter for State Estimation on High-dimensional Implicit Manifolds. *arXiv preprint arXiv:1604.07224*.

[11] Koval, M. C., Pollard, N. S., & Srinivasa, S. S. (2015). Pose estimation for planar contact manipulation with manifold particle filters. *The International Journal of Robotics Research, 34*(7), 922-945.

[12] Li, R. (2015). Touching is believing: sensing and analyzing touch information with GelSight *(Doctoral dissertation, Massachusetts Institute of Technology)*.

[13] Li, R., Platt, R., Yuan, W., ten Pas, A., Roscup, N., Srinivasan, M. A., & Adelson, E. (2014, September). Localization and manipulation of small parts using gelsight tactile sensing. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3988-3993). IEEE.

[14] Li, S., Lyu, S., & Trinkle, J. (2015, May). State estimation for dynamic systems with intermittent contact. In *2015 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3709-3715). IEEE.

[15] Newcombe, R. A., Fox, D., & Seitz, S. M. (2015). Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 343-352).

[16] Patel, R. and Correll, N. Integrated force and distance sensing using elastomer-embedded commodity proximity sensors, in *2016 Robotics: Science and Systems*, 2016.

[17] Pomerleau, F., Colas, F., & Siegwart, R. (2015). A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends in Robotics (FnTROB)*, 4(1), 1-104.

[18] Right Hand Robotics, TakkTile Products, http://www.takktile.com/product:all

[19] Schmidt, T., Hertkorn, K., Newcombe, R., Marton, Z., Suppa, M., & Fox, D. (2015, May). Depth-based tracking with physical constraints for robot manipulation. In *2015 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 119-126). IEEE.

[20] Schmidt, T., Newcombe, R., & Fox, D. (2014). Dart: Dense articulated real-time tracking. *Proceedings of Robotics: Science and Systems, Berkeley, USA*, 2.

[21] SynTouch, BioTac, http://www.syntouchllc.com/Products/BioTac/

[22] Tagliasacchi, A., Schrder, M., Tkach, A., Bouaziz, S., Botsch, M., & Pauly, M. (2015, August). Robust ArticulatedICP for RealTime Hand Tracking. In *Computer Graphics Forum* (Vol. 34, No. 5, pp. 101-114).

[23] Whelan, T., Kaess, M., Johannsson, H., Fallon, M., Leonard, J. J., & McDonald, J. (2015). Real-time large-scale dense RGB-D SLAM with volumetric fusion. *The International Journal of Robotics Research, 34*(4-5), 598-626.

[24] Yuan, W., Li, R., Srinivasan, M. A., & Adelson, E. H. (2015, May). Measurement of shear and slip with a GelSight tactile sensor. In *2015 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 304-311). IEEE.

[25] Zhang, L., & Trinkle, J. C. (2012, May). The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (pp. 3805-3812). IEEE.