Realtime State Estimation with Tactile and Visual sensing. Application to Planar Manipulation.

Kuan-Ting Yu[1], Alberto Rodriguez[2]
[1] Computer Science and Artificial Intelligence Laboratory — Massachusetts Institute of Technology
[2] Mechanical Engineering Department — Massachusetts Institute of Technology
peterkty@csail.mit.edu, albertor@mit.edu

***Abstract*— Accurate and robust object state estimation enables successful object manipulation. Visual sensing is widely used to estimate object poses. However, in a cluttered scene or in a tight workspace, the robot's end-effector often occludes the object from the visual sensor. The robot then loses visual feedback and must fall back on open-loop execution.**

In this paper, we integrate both tactile and visual input using a framework for solving the SLAM problem, iSAM (incremental smoothing and mapping), to provide a fast and flexible solution. Visual sensing provides global pose information but is noisy in general, whereas contact sensing is local but the measurement is more accurate relative to the end-effector. By combining them, we aim to exploit their advantages in order to overcome their limitations. We explore the technique in the context of a pusher-slider system. We adapt iSAM's measurement cost and motion cost to the pushing scenario, and use an instrumented setup to evaluate the estimation quality with different object shapes and on different surface materials.

I. Introduction

We are interested in providing robot manipulators with the ability to track the state of a task in realtime with ordinary hardware. Visual object tracking and detection have been widely studied [1, 2, 3] and can provide reliable estimates of object pose, especially in scenarios with no or limited occlusions. However, it is characteristic of robotic manipulation that the robot, the gripper, or the surrounding clutter will "get in the way" and occlude the object from the cameras. In these scenarios, tactile sensing from distal sensors located at the end-effector can help track the state of an object. Unfortunately, there is a lack of algorithms that can make sense of the high frequency but local information they provide. In this work, we describe a flexible state estimation framework that fuses in real time tactile and visual sensing.

In previous work [4], we showed that it is possible to infer the shape and trajectory of a pushed object only from a batch stream of tactile information. However, the performance was slow and not suitable for online tracking. In this paper, we explore the idea of combining *vision sensing* for approximate global estimation with *tactile sensing* for last-inch accurate interaction. We explore the algorithm in the context of a pusher-slider system [5]. The goal is to estimate 2D object poses in real time given intermittent contact measurements and unreliable vision measurements.

This work was supported by NSF award [IIS-1427050] through the National Robotics Initiative.

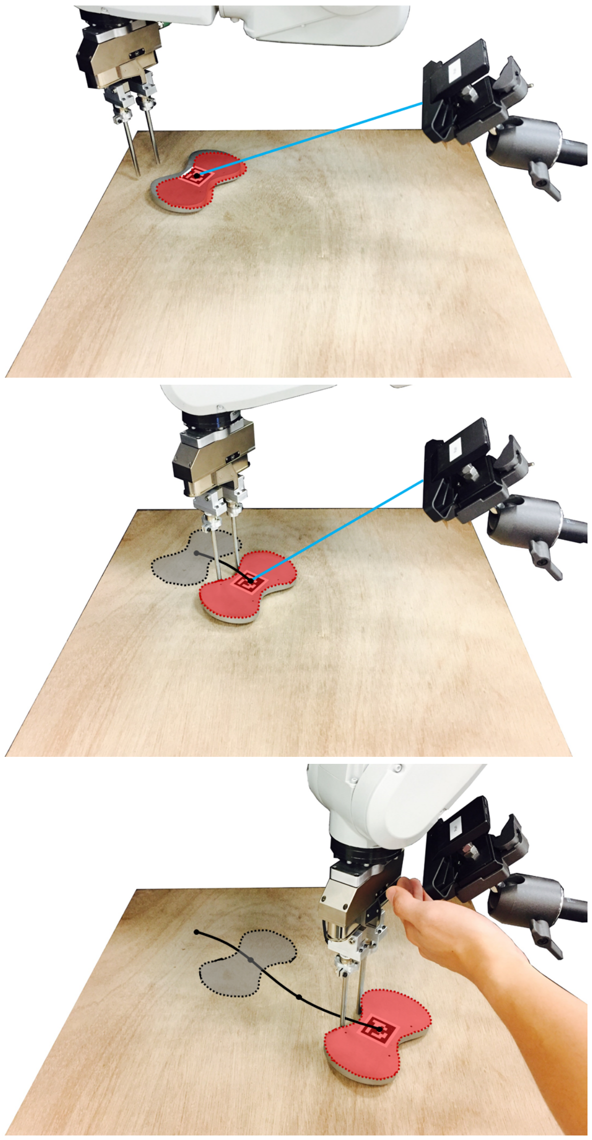


Fig. 1. Concept illustration. (top) Before the robot touches the object, the camera is able to track the object but with noticeable error due to imperfect calibration. (middle) After the robot makes contact with the object, the object pose is corrected based on the contact information. (bottom) During camera occlusion, the estimator still can keep track of the object while being pushed based on contact information. Red shape: current object pose estimate. Grey shape: object real pose in the last image. Black curve: object trajectory. Cyan: camera ray to object.

The robot can push the object in various ways and with different contact modes (single-double contacts and sticking-sliding contacts). Figure 1 illustrates the high-level concept.

We use Incremental Smoothing and Mapping (iSAM) [6] as the underlying optimization framework because of its smoothing structure and the flexibility to fuse heterogeneous measurements and cost functions. The models we use to run the smoothing algorithm assume that we know the shape of the object and that the pressure distribution on the sliding surface between the object and the ground is uniform. In practice, we do not have control over the pressure distribution in the experiment, but the experiments show that the algorithm still works.

Our system features:

- real time estimation at 100 Hz;
- incorporating contact physics in estimation;
- robustness to unreliable sensor input;
- the object can be pushed in various ways, which may involve changing the number of contacting fingers, or involve switching between sticking and sliding;

A key aspect of our system, enabled by the availability of tactile sensing, is that we do not need to consider expensive complementarity programming of the sorts that originates in classical contact problems (contact/no-contact or sticking/sliding). Relying on tactile and force measurements allows us to formulate the problem without complex hybrid dynamics, which speeds up the algorithm.

II. Related Work

In this section, we discuss related literature from three aspects: A. state estimation for object manipulation, B. state estimation frameworks, and C. pushing mechanics.

A. State estimation in object manipulation

Petrovskaya and Khatib [7] tackle the problem of global localization of a known and fixed object by touch. They apply particle filter (PF) to fuse multiple point contact information. PF can handle nonlinear systems and represent multiple modes. In practice, although pure touch-based localization is inspiring, for many real cases, it is often simple to add extra global sensors, e.g. cameras, to quickly trims the search space.

Zhang and Trinkle [8] use PF to track an object during a grasping acquisition with contact sensing patches on static fingers. They find that there is a particle depletion problem, which happens when the contact sensor yields very accurate measurements compared to that from cameras. The inability to fuse information of different accuracy scales is an inherent problem with particle filters.

Koval et al. [9] propose adding manifolds to resolve the problem of particle depletion. They keep track of binary variables representing the contact state. According to the variable, the filter uses different set of dynamical constraints.

Although contacting a surface is usually assumed to eliminate the uncertainty completely in the contact direction, we claim that it is only true if we use the exact contact point as the reference frame. If not, any physical extension from the reference point is not exactly precise and will have different amounts of uncertainty.

Li et al. [10] propose a contact graph that represents the transition of discrete contact states in order to let the contact state evolve according to physics. In terms of computation, adding discrete variables will make the system unscalable due to a combinatorial number of contact modes of participating surfaces.

Schmidt et al. [11] focus on using depth pointclouds and contact constraints for state estimation. Izatt et al. [12] also use both a depth sensor and a high-resolution touch sensor. Hebert et al. [13] fuses both vision and contact sensors. However, they do not consider motion models of the object during frictional contact interaction, which may let noisy measurements introduce unphysical estimates. Their scenario has less dynamical frictional interaction compared to pushing manipulation.

In our previous work [4], we attempt to recover not only pose but also the shape of an unknown object during pushing exploration. We use a batch nonlinear least squares approach to incorporate both contact measurement and motion model constraints. The result was too slow and unreliable to be used in closed-loop interaction. In this paper, we aim to adopt a similar formulation to a easier problem that is useful to enable reactive planning and control. The problem involves tracking the pose of a known object in an online fashion by adding visual input.

B. State Estimation Framework

Extended Kalman Filter (EKF) is a popular framework for online and realtime localization [14, 12, 13]. It linearizes a system so as to apply a Kalman Filter, designed for linear systems. One key drawback with this approach is that the linearization point is chosen as the current estimate of variables. As it is often off from the ground truth, this can result in an inaccurate linearization, followed by an inaccurate estimation.

Kaess et al. [6] propose incremental smoothing and mapping (iSAM) to solve the above issues. iSAM can be viewed as an online nonlinear least-square optimization tool, where cost functions and variables for the optimization can be added during each time step and can update the current estimate of the variables and linearization points. The update is fast because it uses a QR-factorized matrix to represent the linearized cost functions, and only updates very small fraction of the matrix. Also, it exploits the sparsity of the constraints.

Besides being fast, it provides more accurate estimates than filtering based method e.g. extended Kalman Filter and particle filter (PF) because it maintains all the cost functions (soft constraints) of multiple timesteps instead of just the last step, and finds the optimal solution based on all of them. This can avoid noisy measurements to cause jumpy estimates. It also updates the linearization point at a later stage to avoid the inaccurate linearization issue in Kalman Filter.

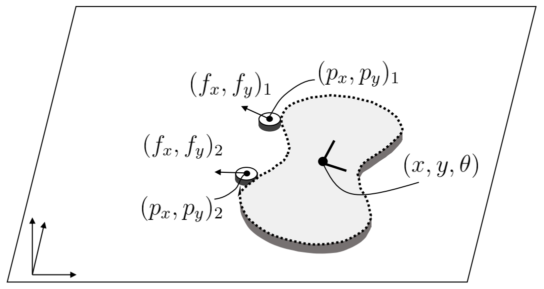


Fig. 2. Diagram for explaining the pushing state estimation problem.

C. Pushing mechanics

Pushing is difficult to model and predict because it involves two or more simultaneous frictional interaction between pushers and the object, and between object and surface. In terms of accuracy, the frictional interactions between real materials are uncertain. This is demonstrated in Yu et al. [5] by analyzing a large experimental pushing dataset. They found that friction properties are difficult to characterize precisely and they change based on many factors. However, based on some simplification, Lynch et al. [15] propose an analytical and probably the most popular physics-based motion model. It is based on Limit Surface (LS) [16] for force-motion mapping, and ellipsoid approximation of LS [17] for fast computation. We use Lynch's motion model as the pushing motion model due to its simplicity. There are more advanced pushing modeling which can be easily plugged in. Zhou et al. [18] use a convex polynomial to represent LS more accurately. Bauza and Rodriguez [19] use a Gaussian process to learn a stochastic model directly from data without physical modeling.

III. Example Problem: Pusher-Slider Object Pose Estimation

We are concerned with the problem of estimating the pose of a rigid 2D object pushed on a table in real time. The interaction between object and pusher is observed with periodicity and we use the subscript $t \in [1...T]$ to indicate the corresponding timestamp along the trajectory.

Object pose. We estimate the object pose denoted by $\mathbf{x}_t = (x, y, \theta)$ with the following inputs.

Visual input. A visual input includes a 2D pose $\mathbf{w}_t$, and a binary variable denoting whether it is available at time t. We need the latter because sometimes the camera is occluded or the frame has not arrived.

Tactile input. A tactile input $\mathbf{z}_t = \mathbf{z}_{t,i}$ includes force experienced $(f_x, f_y)_{t,i}$ and finger position $(p_x, p_y)_{t,i}$ in 2D on finger i. Finger positions are derived from robot joint states. We use $D_{t,i}$ to represent whether finger i at time t is in contact or not by setting a constant threshold τ on the force received. That is, $D_{t,i} = 1[\|(f_x, f_y)_{t,i}\| \geq \tau]$, where $1[\cdot]$ is an indicator function.

We illustrate the above variables in Figure 2.

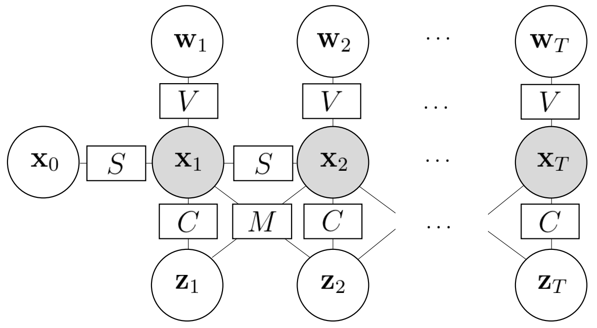


Fig. 3. Factor graph representation of the variables and cost functions.

IV. Method

A. Applying iSAM

Here we describe how we use iSAM in the context of the pushing state estimation problem. Refer to [6] for the details of the iSAM algorithm. Below, we present the problem as solving a least squares problem; i.e., finding variables to minimize a cost function. Note that the variables and cost functions will be added and removed as time proceeds, in contrast to batch optimization techniques. Also, note that iSAM requires the assumption of a Gaussian noise model. We will test normality using real data in Section V.

The overall cost function is a sum of 4 cost functions:

- the pushing motion cost M;
- the tactile measurement cost C;
- the visual measurement cost V;
- stationary prior cost S.

A factor graph in Figure 3 shows the relationship between these cost functions. In summary the overall least squares problem is:

$$\begin{aligned} X^* = \underset{X}{\operatorname{argmin}} \sum_{t=1}^{T} & \|M(\mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{z}_t, \mathbf{z}_{t+1})\|_\Lambda^2 \\ & + \|C(\mathbf{x}_t, \mathbf{z}_t)\|_\Gamma^2 + \|V(\mathbf{x}_t, \mathbf{w}_t)\|_\Upsilon^2 \\ & + \|S(\mathbf{x}_t, \mathbf{x}_{t-1})\|_\Omega^2 , \end{aligned} \tag{1}$$

where X is a long vector formed by concatenating $\mathbf{x}_t$'s, and $|\mathbf{e}|_\Sigma = \mathbf{e}^T \Sigma^{-1} \mathbf{e}$ computes squared Mahalanobis distance with covariance matrix Σ. The matrices Λ, Γ, Υ, and Ω are the covariance matrices for the corresponding noise. We identify them from the measurement input and the ground truth. If some measurement is missing due to physical limitations, we will remove the relevant cost functions, e.g. when the object is not in camera view, we remove the V term.

We always add a stationary prior because, in object manipulation, the object movement is almost zero in the time step of an estimation cycle, about 10 ms. From experiment, this cost helps prevent the program become underdetermined. On the other hand, it stabilizes the estimation result by filtering out jitters due to sensor noise.

Below we will describe the 4 cost functions in detail.

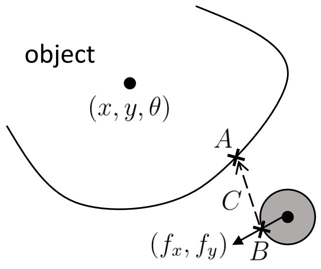


Fig. 4. Illustration of contact measurement cost. Point A is the closest point from the object to the pusher. Object is at the pose described by the variables. Point B is the contact point derived from the finger position and sensed force direction. Vector C represents the distance between the two points that we want to minimize.

B. Physics-based pushing motion

Using Lynch's pushing model [15], we have the following assumptions:

- pushing at quasi-static speed;
- using ellipsoid limit surface approximation;
- uniform friction between object and the surface and between object and pusher;
- uniform pressure distribution between the object and the surface and between object and pusher.

To impose the pushing model, we use the following cost function:

$$M(\mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{z}_t, \mathbf{z}_{t+1}) = \left[\frac{v_x}{\omega} - c^2\frac{F_x}{m}, \frac{v_y}{\omega} - c^2\frac{F_y}{m}\right]^T, \quad (2)$$

where

- v_x and v_y are object velocities in x and y axes, derived from finite differences of $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$;
- ω is angular velocity;
- (F_x, F_y) is the total force i.e., $\sum_i (f_x, f_y)_i$;
- m is the total applied moment relative to current estimate of object center;
- c is a scalar constant derived from the object pressure distribution.

All the variables are in the current object frame.

Note that we do not need to distinguish between sticking or sliding because we sense the force acted on the object directly. By using limit surface representation we can directly map the force acted on the object to the object velocity. Please refer to [4] for detailed derivation.

C. Contact measurement

The measurement cost is defined as the difference between the sensed contact point B and the estimated closest point A on the object with respect to the pusher contour.

$$C(\mathbf{x}_t, \mathbf{z}_t) = A(\mathbf{x}_t, \mathbf{z}_t) - B(\mathbf{z}_t). \quad (3)$$

Figure 4 illustrates the cost function and the two points. Note that this cost imposes not only that the contact point is right on the object boundary but also that the contact direction is correct.

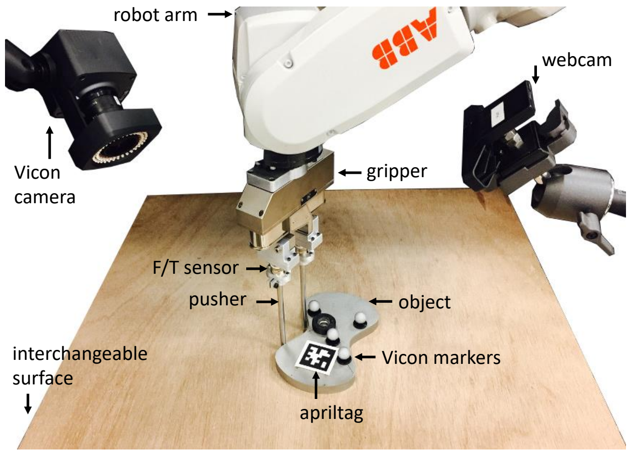


Fig. 5. Experimental hardware setup.

D. Visual measurement and stationary prior

The visual measurement cost forces the pose estimate to be close to the visual input; the stationary prior forces the pose estimate to be close to the one from the last step. Both are implemented with a subtraction of two inputs:

$$V(\mathbf{a}, \mathbf{b}) = S(\mathbf{a}, \mathbf{b}) = \mathbf{a} - \mathbf{b}. \quad (4)$$

The cost functions output a vector of three elements. Since the third element is an angle, we need to wrap it into $[-\pi, \pi)$.

V. Experiments

Our system estimate object pose at 100 Hz. It consumes visual inputs at 30 Hz and tactile inputs at 250 Hz. We want to answer the following questions through our experiments:

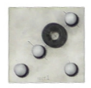
- Is contact measurement noise normally distributed? Since we assume a Gaussian noise model, we test normality of the noise and find the covariance matrices for each cost functions.
- Is iSAM a better parametric estimation framework than EKF in terms of estimation accuracy?
- How does each cost function contribute to form the final estimate? Which cost is able to correct what type of error?
- How well does the estimation work on different shapes? Is there special geometry that is harder than others?
- How well does the estimation work on different surfaces?
- How fast can iSAM compute? Realtime computation is crucial to provide inputs for reactive control or planning.

More details about the experimental software and results are available online [20].

A. Hardware setup

We rigorously evaluate our method, we have an instrumented setup as shown in Figure 5: a 6 DOF industrial robotic manipulator equipped with two stiff cylindrical rods acting as a pusher. The setup is similar to that in our previous work where we collected an extensive pushing dataset [5].

TABLE I
OBJECTS USED IN THE EXPERIMENTS. PHYSICAL PROPERTIES.

Object	`rect1`	`ellip2`	`butter`
Picture			
Mass (g)	837	1110	1197
Width (mm)	90	105	95.3, 54.7
Height (mm)	90	130.9	156

Robot. The system uses an ABB IRB 120 industrial robotic arm with 6 DOF to control precisely the position and velocity of its tool center point (TCP). The TCP moves at 60 mm/s in the experiments.

Force sensing. We use two ATI Nano17 F/T sensors rigidly attached to the gripper to measure the reaction force from the object on the pusher.

Since we only have force sensors but not contact sensors, we assume contact direction and sensed force direction is the same when we compute contact measurement cost. In our experiments, we find them to be very close. Using only the force sensor allows the pusher to be very slim in appearance and strong mechanically.

Pushers. The robot is equipped with two stiff cylindrical steel pushers, mounted on and perpendicular to the measurement plates of the force-torque sensor. The pusher has length 115 mm and diameter 6.25 mm.

Objects. We use 3 objects, all water-jet cut in stainless steel. All objects are 13 mm thick. The friction coefficient between the pusher and the object is approximately 0.25, which was determined using a traditional variable slope experiment. A fiducial marker, Apriltag [21], of 3 cm by 3 cm is stuck on the block to facilitate tracking from webcam. This is to obtain realistic visual object pose estimation input. The objects are also instrumented with reflective markers and tracked with a Vicon motion tracking system for groundtruth. Table I summarizes the objects that we experimented with.

Surface material. We experiment with four surfaces: i) ABS, ii) Delrin, iii) plywood and iv) polyurethane (hardness 80A durometer). We have found different frictional characteristics in [5].

Pushing procedure. We use a pushing procedure to test the system. It covers several kinds of possible ways to push with two fingers: having contact and no contact; having sticking and sliding; having one finger and two fingers in contact; having occlusion or not. The procedure takes around 50 sec and is illustrated in Figure 6.

Default configuration. The default object for pushing is `rect1`, and the default surface is `plywood`. If later we do not specify the configuration then we are using the default.

Computation. All computation was done on a laptop machine with Intel Core i7-3920XM CPU and 16 GB RAM.

Baseline. We use pure visual input without any filtering as our baseline method. When a visual input is not available at a time step, we use the latest available visual input.

B. Noise characterization

Since iSAM assumes a Gaussian noise model, we first want to test normality of the measurement noise models, and then find their covariance matrices. To find the error distributions, we evaluate the cost functions by using sensor measurements and groundtruth object pose from Vicon.

Our results confirmed that all the cost functions can be well approximated with a Gaussian distribution. Due to space constraints, we only show the normality test for contact measurement in Figure 7, which gives error distributions close to normal. While in theory an ideal contact should result in zero distance, in reality, any extension from the contact point will be imperfect. For example, the stiff pusher may deflect slightly when pushing an object such that the contact point given by the robot's kinematics does not math reality.

Having ensuring all the noises can be approximated as Gaussian distribution, we find the error covariance matrices using the groundtruth pose estimated from Vicon. In the following experiments, we use the same covariance matrices found with the default configuration because we find the parameters to be similar across different shapes and surfaces. In doing so, we can demonstrate the robustness of our estimation algorithm to variation.

C. iSAM vs EKF, smoothing vs filtering

In this section, we want to examine whether optimizing over a history of steps performs better than over one stpe like in EKF. Table II the result in terms of root mean squared error (RMSE) in translation and rotation.

We feed the same data with both visual and tactile information into EKF and iSAM using a different history length. The estimation accuracy is shown in Table II. We have two observations from the result:

- EKF does improve the accuracy from pure visual input but is not as good as iSAM with multiple steps of history. Having a very short history makes the system more prone to abrupt sensor noise.
- Keeping longer history in iSAM, in general, increases the accuracy, but the improvement becomes limited as the number of steps increase.

D. Contribution of costs

Visual input has its strengths and weaknesses. It provides global information to guide the local tracking with motion and contact model. In our experiment, if the camera does not provide input for sufficiently long, we will lose track of the object. However, visual input alone is noisy and inaccurate. We see an average of 15 mm translational error as shown in Table II, which is due to calibration error and occlusions. In that situation, contact model can help to refine the estimation.

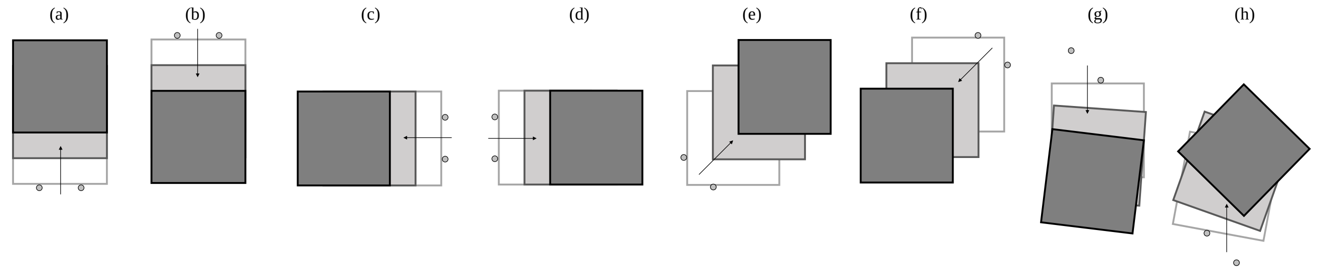


Fig. 6. Illustration of the standard testing pushing procedure. The webcam is located roughly at the right side of the block and looking left. (a)-(d) straight two finger pushes. (e)(f) corner two-finger pushes. (g)(h) one-finger pushes. Although some pushes are symmetric but the robot may occlude the webcam in different ways.

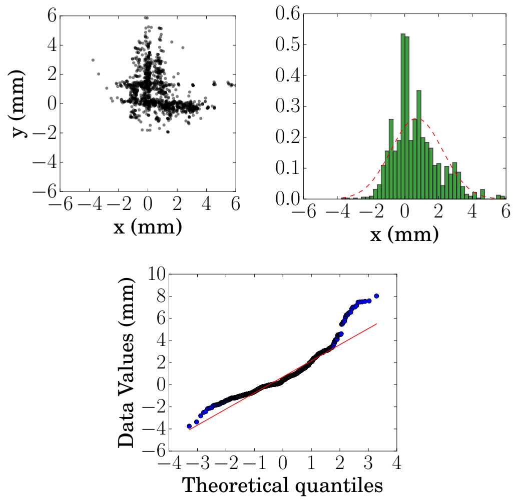


Fig. 7. Normality test of contact measurement error. Top left: scatter plot of the errors in contact frame. Top right: histogram of the data projected to first axis and a fitted Gaussian curve. Bottom: QQ plot in first axis; the closer the data points to the straight line, the better the noise follows a Gaussian distribution.

TABLE II
RMSE WITH DIFFERENT ESTIMATION METHODS.

Method	**Trans. (mm)**	**Rot. (deg)**
Visual input	15.7±11.7	3.4±3.0
EKF	6.4±1.8	6.4±6.4
iSAM 1-step	7.3±3.8	3.7±3.6
iSAM 100-step	6.3±2.2	2.6±2.6
iSAM 200-step	6.0±2.1	2.3±2.3
iSAM 1000-step	6.1±2.1	1.8±1.8

Figure 9 shows some examples where contact helps to refine the visual input.

Moreover, when there is no visual input for a sequence of time due to occlusion, we find that both contact model and motion model are important for accurate estimation as shown in Figure 8, where visual inputs are not available, we show how contact measurement cost and motion cost contribute to good estimation. In (b), we see the estimation works well by using both costs. In (c), we use measurement cost but not contact cost, and the estimation fails because the motion prediction is very sensitive to the current estimate of object pose. On the other hand, in (d), we use contact model but not motion model, the estimated pose drifts perpendicularly to the contact normal.

TABLE III
RMSE WITH DIFFERENT SHAPES.

Shape	**Baseline**		**iSAM**	
	Trans. (mm)	**Rot. (deg)**	**Trans. (mm)**	**Rot. (deg)**
`rect1`	15.7±11.7	3.4±3.0	6.0±2.1	2.3±2.3
`ellip2`	16.7±14.6	4.0±3.3	7.3±4.9	**5.7**±4.7
`butter`	68.4±59.5	10.7±10.7	12.0±8.5	**12.1**±10.4

E. *Varying object shapes*

We first show that the cost functions can be applied to `rect1`, `ellip2`, and `butter` shapes. Figure 9 shows a qualitative result of validating contact measurement cost. The algorithm can be applied if the object can be approximated well as polygons and does not have small cavities where pusher cannot enter.

We want to see if there are relationships between estimation accuracy and object shapes. Table III shows the estimation accuracy with the three objects. Although the standard pushing procedure may result in slightly different pushing interaction with the shape, we can still observe a general tendency. We observe that there are different error characteristic for different shapes. In general, for objects with smooth curves, i.e. `ellip2` and `butter`, the estimation error in rotation will be greater. We can reason it from a simple analysis: the measurement difference of nearby poses have a small gradient. In the extreme case, a circular object is ambiguous in all contact directions, so contact measurement will not be useful to distinguish object orientation.

Note that the baseline error of pushing `butter` is relatively high. That is because Apriltag was occluded more often by the robot and the Vicon markers.

F. *Varying surfaces*

We want to show that our solution works on different surfaces. In our previous work, each surface has different frictional properties [5]. The variations include the variance of dynamic coefficient of friction, anisotropic/isotropic friction, etc. In the experiment, we ensure the initial object pose

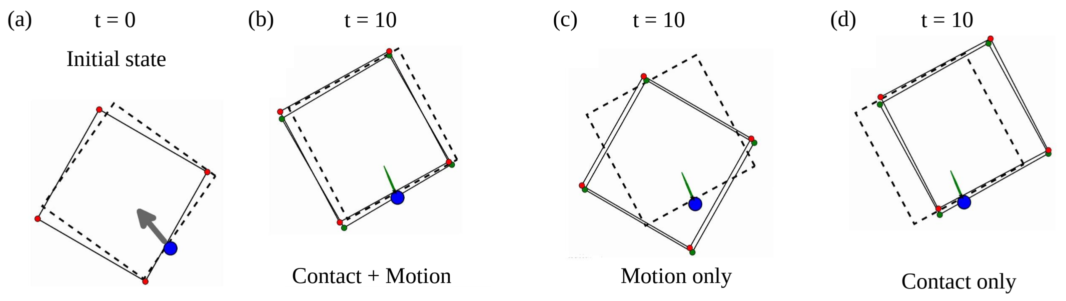


Fig. 8. Estimation results without Apriltag pose estimation due to occlusions and with different contact costs enabled. Red-dotted contour: current estimate of object pose. Green-dotted contour: estimate from the previous step. Dashed contour: groundtruth. (a) Initial condition, the pusher is going to push in the grey arrow direction. (b)(c)(d) are after 10 steps with the same push but with different contact costs enabled. (b) has both contact and motion costs. (c) has motion but not contact costs. (d) has contact but not motion costs.

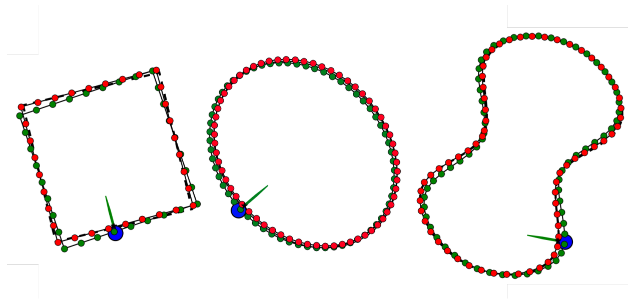


Fig. 9. Qualitative result of estimation. Here we emphasize the contribution of contact measurement cost. Green object contour: noisy pose input from visual pose detection. Red object contour: after iSAM update. Dashed object contour: groundtruth pose from vicon. Blue circle: pusher position. Green arrow from pusher: sensed contact normal. Star: sensed contact point. Notice the penetration between the pusher and the object from noisy input, which is corrected by the contact measurement.

TABLE IV
RMSE WITH DIFFERENT SURFACES USING RECT1.

		Baseline		iSAM	
Surface	**DCoF**	**Trans. (mm)**	**Rot. (deg)**	**Trans. (mm)**	**Rot. (deg)**
`abs`	0.16	14.3±10.9	7.8±7.5	5.6±2.7	3.7±3.6
`delrin`	0.15	13.6±7.0	1.3±1.1	8.7±3.3	**3.0**±2.9
`plywood`	0.28	15.7±11.7	3.4±3.0	6.0±2.1	2.3±2.3
`pu`	0.35	11.0±8.8	3.0±2.9	5.1±2.4	1.9±1.9

is as similar as possible for fair comparison, so the pushing strokes are with respect to the initial pose of the object. Table IV compares estimation accuracy on different surfaces. We also list their dynamic coefficients of friction (DCoF).

In most situations, the system greatly reduces the noise when fusing contact measurement. Only on the `delrin` surface, the rotation estimate is worse than baseline. We hypothesize that surface `delrin` has the smallest DCoF among all the surfaces, so force sensing is not as accurate compared to surfaces with higher DCoF. Therefore, after fusing it, the estimation in rotation become worse. We also see that `abs`, with low DCoF, has poorer accuracy in estimating rotation compared to other surfaces.

Part of the accuracy improvement using contact comes from the fact that we use two fingers to push objects stably, reducing the uncertainties from pushing.

G. Timing

The average time of computation is less than 1 ms for a 200-step history, including periodic reliniearization. The linearization step is more time consuming, averaging 28 ms ($\pm$ 21 ms) to linearize. The maximum linearization time was 70 ms, which corresponds to a pushing distance of 4.2 mm if we are pushing at 60 mm/s.

To ensure constant processing rate, we choose to only maintain a history of 200 steps for all the results in this paper. We remove 100 nodes and related cost functions when the number of nodes reaches 300 steps. We do so because iSAM will relinearize every 100 steps, and removing nodes also requires relinearization. Then we can save time by not relinearizing. The length can be chosen as a trade-off between computation speed and accuracy.

Note that the time reported above does not include visual input processing time. The Apriltag pose is tracked at 30 Hz. So, using tactile sensor also helps with fast motion because contact sensors and robot pose is publishing at the higher rate of 250 Hz.

VI. CONCLUSION

In this paper, we propose and demonstrate the use of iSAM for online estimation of object pose while pushing an object. Through extensive experiments, we understand how well the solution works in different task conditions, including different shapes, materials, and object interactions.

We shows that iSAM can give better accuracy than EKF by keeping a history of observations, while still being fast enough to allow realtime estimation. By design, tactile sensing allows the system to distinguish between contact/no-contact and sticking/sliding without requiring expensive complementarity programming.

Failure modes. In some challenging cases, the estimation loses track of the object. The main reasons are:

- force sensor detects contact when there is no contact;
- the estimation not receiving good visual inputs for a long period of time.

A careful calibration of the tactile/force sensor surely helps with the first issue. A conservative thresholding for when there is contact also helps in reducing damaging false contact positives. When contact is activated, a bad measurement will offset the estimate significantly because the contact measurement has high condidence. The second point can be alleviated if we plan for motions that facilitate visual perception.

Limitations. This work has focused on tracking one object on a clean table top scenario. It may be possible to rely on occasional visual inputs to keep track of object of interest in clutter and reason about the contact situation, but many details need to be addressed. In general, interaction of two or more objects without visual input is very challenging, even for humans.

Future Work. In the future, we would like to test our estimation in the context of a pushing controller [22] for reactive manipulation, which has been tested with accurate ground truth feedback from an external tracking system, and to generalize to 3D tasks such as prehensile manipulation [23, 24, 25]. In both scenarios, state estimation is crucial to enable robot reactiveness and correct for dynamic, motion and sensor noise.

The proposed algorithm relies on the structure of a basic localization problem. There are many concepts and advances from the SLAM community, such as more complex data association schemes, that can be applied in a manipulation scenario. A motivating example is what to do when a high-fidelity contact sensor like Gelsight [26] is available. Texture and salient geometric features can help with associating measurement with the object model.

References

[1] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker Jr, A. Rodriguez, and J. Xiao, "Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge," *ICRA*, 2017.

[2] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *European Conference on Computer Vision*. Springer, 2012.

[3] T. Schmidt, R. Newcombe, and D. Fox, "Dart: dense articulated real-time tracking with consumer depth cameras," *Autonomous Robots*, 2015.

[4] K.-T. Yu, J. Leonard, and A. Rodriguez, "Shape and Pose Recovery from Planar Pushing," in *IROS*, 2015.

[5] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing," in *IROS*, 2016.

[6] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. on Robotics (TRO)*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.

[7] A. Petrovskaya and O. Khatib, "Global localization of objects via touch," *IEEE Trans. on Robotics (TRO)*, vol. 27, no. 3, pp. 569–585, 2011.

[8] L. Zhang and J. C. Trinkle, "The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing," in *ICRA*, 2012.

[9] M. Koval, N. Pollard, and S. Srinivasa, "Pose estimation for planar contact manipulation with manifold particle filters," *IJRR*, vol. 34, no. 7, June 2015.

[10] S. Li, S. Lyu, and J. Trinkle, "State estimation for dynamic systems with intermittent contact," in *ICRA*, 2015.

[11] T. Schmidt, K. Hertkorn, R. Newcombe, Z. Marton, M. Suppa, and D. Fox, "Depth-based tracking with physical constraints for robot manipulation," in *ICRA*, 2015.

[12] G. Izatt, G. Mirano, E. Adelson, and R. Tedrake, "Tracking objects with point clouds from vision and touch," in *ICRA*, 2017.

[13] 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 *ICRA*, 2011.

[14] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[15] K. M. Lynch, H. Maekawa, and K. Tanie, "Manipulation and active sensing by pushing using tactile feedback." in *IROS*, 1992.

[16] S. Goyal, A. Ruina, and J. Papadopoulos, "Planar Sliding with Dry Friction Part 1. Limit Surface and Moment Function," *Wear*, 1991.

[17] S. H. Lee and M. Cutkosky, "Fixture planning with friction," *Journal of Manufacturing Science and Engineering*, vol. 113, no. 3, 1991.

[18] J. Zhou, A. Bagnell, and M. Mason, "A fast stochastic contact model for planar pushing and grasping: Theory and experimental validation," in *RSS*, 2017.

[19] M. Bauza and A. Rodriguez, "A probabilistic data-driven model for planar pushing," in *ICRA*, 2017.

[20] Website for state estimation: data, code, and experimental result. [Online]. Available: http://mcube.mit.edu/push-est

[21] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *ICRA*, 2011.

[22] F. Hogan and A. Rodriguez, "Feedback control of the pusher-slider system: A story of hybrid and underactuated contact dynamics," in *WAFR*, 2016.

[23] N. Chavan-Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. A. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. A. Fuhlbrigge, "Extrinsic Dexterity: In-Hand Manipulation with External Forces," in *ICRA*, 2014.

[24] N. Chavan-Dafle and A. Rodriguez, "Prehensile pushing: In-hand manipulation with push-primitives," in *IROS*, 2015.

[25] N. Chavan-Dafle and A. Rodriguez, "Sampling-based Planning of In-Hand Manipulation with External Pushes," in *ISRR*, 2017. [Online]. Available: http://arxiv.org/abs/1707.00318

[26] M. K. Johnson and E. H. Adelson, "Retrographic sensing for the measurement of surface texture and shape," in *CVPR*, 2009.