2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

Learning Modes of Within-hand Manipulation

Berk Calli, *Member, IEEE*, Krishnan Srinivasan, Andrew Morgan, and Aaron M. Dollar, *Senior Member, IEEE*

***Abstract*— In this work, we investigate methods to detect four phenomena (modes) that occur during prehensile fingertip-based within-hand manipulation without the use of tactile sensors. By using actuator states and visual data, we aim to recognize different modes of operation such as interpreting if the hand is about to drop the object, if the object will begin to slide on the fingers, or if the system is at or near a singularity. For this purpose, we utilize supervised learning techniques, which allow us to detect the modes without the use of a mechanical model of the system. We analyze the individual roles of specific features available through both the actuator and visual data, and identify the ones that have the most significance for detecting the operation modes. Our results show classification performance of 96% (using either Extra Trees, Gradient Boosting, or SVM) when using combined actuator and visual features. Interestingly, we were able to achieve a 94% classification rate using only actuator information, and 93% using only visual information. Overall, the classifiers identified actuator positions, actuator loads, and commanded velocities as the most important features for detecting a mode. These results have implications for enabling the control of within-hand manipulation movements utilizing a minimal amount of sensory information without a model of the hand/object system.**

I. Introduction

The ability to manipulate an object within the hand introduces a great degree of dexterity to a robot as it enables repositioning or reorienting of the object without re-grasping or large whole-arm or whole-body motions [1], [2]. Kinematics and dynamics models of in-hand manipulation have been derived in the literature for various contact models and hand topologies [3]–[5]. Utilizing these models for executing an in-hand manipulation task, however, is typically very challenging, as they generally require an accurate knowledge of the object and hand models. In addition, the contact locations on the object and the fingers need to be known along with the friction coefficients and force magnitudes at these locations. Unfortunately, this information cannot be reasonably known for many robotics scenarios, as precise information about object properties are not often known in advance and sensory information, if available, is generally noisy. Moreover, for compliant/soft hands, deriving accurate models for the mechanical response of the hand/object system may not be feasible at all as very few options exist for estimating complex spatial deformations of soft structures.

In our earlier work [6], we have demonstrated that accurate and efficient within-hand manipulation can be conducted by only using very rough gripper models and without the knowledge of object models, contact locations or applied forces if two key components are combined together: system compliance and vision feedback. The role of system compliance is to ensure contact with the object during manipulation. This can either be achieved mechanically by adaptive/underactuated/soft robotic hands [7]–[9] or by the use of an impedance control framework [10]. With vision

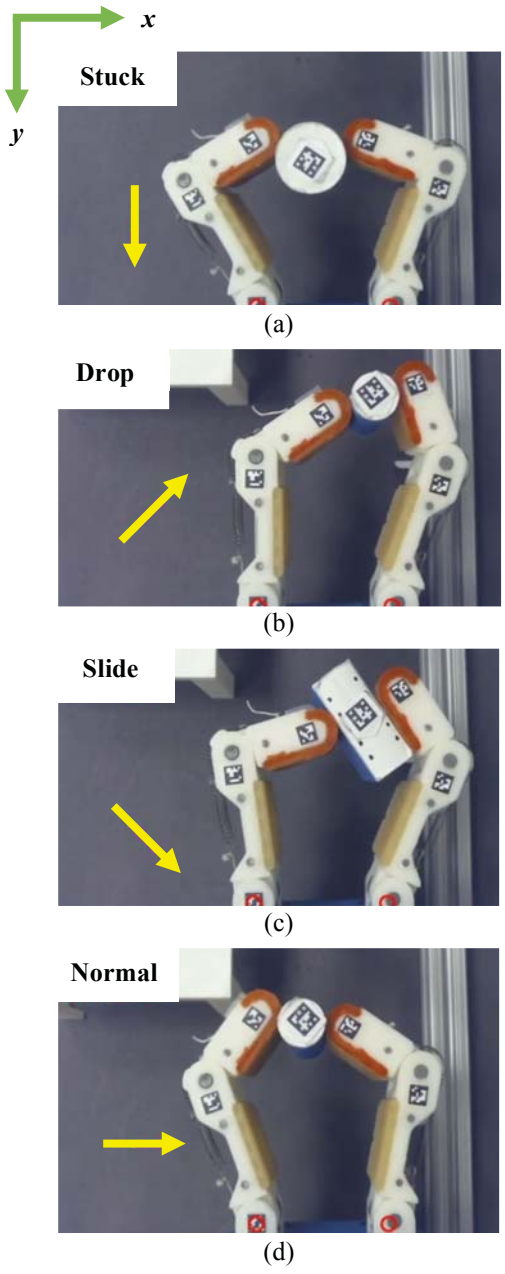


Figure 1. We investigate the ways of detecting in-hand manipulation modes using visual features, actuator data and user/controller commands (yellow arrow signifies commanded velocity). The classifier predicts whether the system will: (a) get stuck, (b) drop the object, (c) slide the object within hand or (d) do normal operation. Classification results generalize for objects with different shapes and sizes.

feedback, we are able to close the loop in the task space, which allows us to maintain convergence even with very rough gripper models since visual servoing algorithms are robust to robot calibration errors. In addition, the performance of the system can be boosted by utilizing advanced control techniques: we have used the model predictive control framework for achieving close-to-optimal object trajectories in the image space, which results in faster convergence and less travel distances while moving between set points [11].

Such a combination of system compliance and vision feedback allows us to avoid sophisticated motion planning strategies that relies on accurate system models. On the other hand, the lack of accurate models prevents us from calculating workspace constraints and singularities of the system. Therefore, it becomes challenging to determine when the in-hand manipulation system would move out of the normal operable workspace and drop the object or get stuck in a singularity. In addition, we lack tools to detect and control manipulation phenomena such as sliding, which might be desirable or undesirable depending on the manipulation goal. Detecting such modes is not only crucial for achieving safe manipulation, but also instrumental for learning control policies since they will help reducing the task failures. The study in [12] shows that "people can learn to predict the consequences of their actions before they can learn to control their actions." This allows them to avoid task failures, while learning more complex manipulation policies.

In this work, we investigate ways of estimating four manipulation modes (or states) of the hand/object system during prehensile fingertip-based within-hand manipulation (summarized in Fig. 1): "drop" (where the object is on the verge of being dropped), "stuck" (where a singularity has been reached and motion cannot freely happen in all directions), "sliding" (where the object is on the verge of sliding), and "normal" (where the object can be freely moved in all directions). We seek answers to the following questions:

1) How accurately can these manipulation modes be estimated using actuator information and visual data? Here it is important to note that we utilize an underactuated hand in which the actuator positions and loads do not supply the full hand state.

2) Which features/information have higher importance for estimating the manipulation modes?

3) Can the results be generalized for objects with different sizes and geometries?

4) How can we utilize the mode detection scheme in an online in-hand manipulation control loop?

For answering these questions, we adopted supervised learning techniques and used a Model T-42 underactuated hand [7] as our test bed (Fig. 2). We collect data for each of the manipulation modes and train our system with a range of classifiers. With a feature significance analysis, we examine key features for detecting the manipulation modes, including scenarios where only actuator or only visual information is available. Finally, the classifier is used to conduct safe in-hand manipulation, which avoids the regions with high risk of singularity or object drop.

This paper is organized as follows. The next section presents a review of the related work. In Section III, our method to estimate manipulation modes is explained in detail. In Section IV, classification results are presented together with a discussion on feature importance. Section V concludes the paper.

II. Related Work

Robotic manipulation is challenging to model due to the complexity of the interaction between the robot, the objects and the environment. Physical parameters of these interactions (e.g. object shape and friction coefficients) are also generally not available in unstructured environments. In order to avoid the use of explicit models, learning algorithms are utilized to generate manipulation policies for various tasks. In robotic grasping, these algorithms are used to obtain a mapping between tactile readings and the grasp stability [13]–[15]. In [16], reinforcement learning is used for learning how to manipulate articulated objects. Similarly, a probabilistic framework is proposed in [17] for learning kinematic models of the articulated objects.

During manipulation, gripper compliance brings a great degree of robustness for establishing and maintaining the contact with the object and the environment. Even though, compliance introduces further complexity to the contact modeling, it provides a safe operable workspace for the learning algorithms to explore. For instance, in [18], reinforcement learning methods are utilized for learning compliant manipulation strategies for opening a door and picking up a pen. In this way, the advantage of compliance is exploited without the need for explicit task and interaction models.

The learning by demonstration framework provides another strategy for obtaining policies for manipulation. In [19], models of various manipulation tasks are learnt by human demonstrations and fusing various visual and tactile data. In this work, similar to ours, they also use the acquired data to differentiate patterns of different stages of manipulation: they distinguish between the sensor patterns that corresponds to simple object displacements (without manipulation) and within-hand object positioning.

Specifically for within-hand manipulation, object positioning skills are learnt for a compliant hand by utilizing tactile sensing via reinforcement learning [20]. In [21], deep learning is used to learn in-hand affordances directly from raw images. Learning by demonstration is also utilized for learning fine in-hand manipulation skills in [22], [23].

In this work, we do not learn control policies as the above-mentioned work, but employ actuator and visual data for detecting manipulation modes; we believe that such a high-level supervision coupled with robust control techniques in image space (e.g. [11]) can result in accurate, generalizable and stable manipulation strategies. Nevertheless, the strategy proposed in this paper is very suitable to be combined with the above-mentioned approaches, which do not use an explicit system or task model.

In the literature, several approaches are reported for detecting stick-slip conditions during manipulation [24]–[27]. In all these works, tactile data is used for training the classifiers. Nonetheless, integrating tactile sensors complicates the mechanical and electrical design, and could result in a bulky system. This is especially undesired for a hand like Model T-42, in which the goal is to achieve high functionality with simple and inexpensive design. In our work, we do not use tactile sensing for mode estimation, but investigate the role of actuator and visual data for detecting the manipulation modes. Moreover, we train a single classifier for detecting the conditions for dropping the object, getting into singularities and sliding the object within hand.

III. Data Collection For Mode Estimation

We aim to estimate four modes of in-hand manipulation, which are explained below:

Drop: This mode signifies that the object will be dropped if the system moves towards the commanded direction.

Stuck: The system will get into a singularity and won't be able to move further, if the commanded direction is followed.

Slide: The object will slide within hand if the commanded direction is followed.

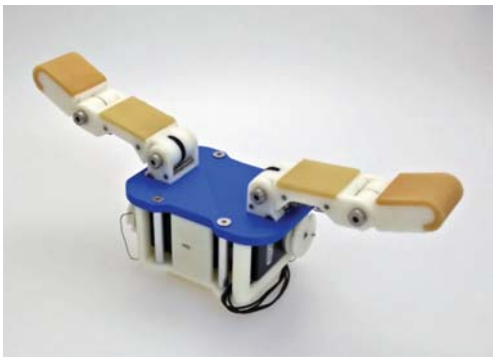
Normal: The system is neither in 'stuck', 'drop' or 'slide' mode.

Detecting these modes gives us the advantage of avoiding failures or undesired object motions during in-hand manipulation; they are useful to manipulate the target object along a desired trajectory without dropping, slipping, or getting into singularities. In some cases, the user may want to trigger these modes for various purposes; sliding the object within-hand may be advantageous to shift the contact points on the object. The "stuck" mode can be useful for squeezing the object with compliant grippers since reconfiguration is jammed in singularities. Detecting the "drop" mode may be used to signal the robot that the object will be released soon.

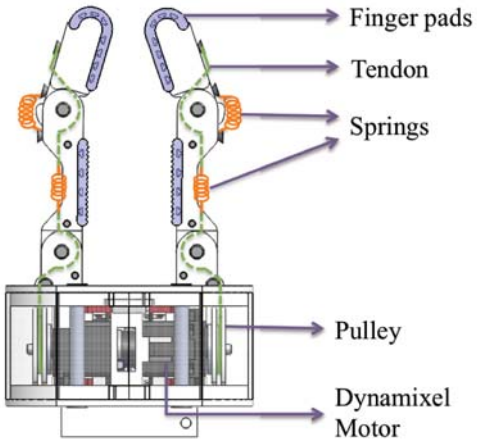
We utilize supervised learning methods for training a classifier using actuator data and visual features, without using object or gripper models. This model-free strategy also allows to integrate the detection method in vision-based manipulation frameworks that do not rely on accurate system models (e.g. [6], [11]).

We trained our classifiers using Model T-42 underactuated hand (Fig. 2; [7]). This hand has two identical opposing fingers each of which has two joints and one Dynamixel MX type actuator. We collected training data by manipulating six different objects: three cylinders and three rectangular prisms with various sizes, as can be seen in Fig. 3. Our setup is presented in Fig. 4. We firmly mounted our Model T-42 hand and placed a camera to observe the system directly from the top.

In order to generate the training data we recorded actuator and visual data streams synchronously with the following procedure. The object is supported with a stand for the initial grasp. After the grasp, the stand is removed so that there is no support plane during in-hand manipulation. The object is moved in the planar workspace by velocity references supplied manually via a keyboard. We have 9 inputs: north, north-east, east, south-east, south, south-west, west, north-west and a stop command. These Cartesian velocity commands are projected into the actuator space via a Jacobian matrix obtained by utilizing simple manipulation primitives as explained in [6]. Dynamixels' built in controllers are used to realize the actuator commands. Using the manual Cartesian space commands, we steer the system for triggering the four above-mentioned modes and recorded data streams in which the object drops, gets stuck, slides and operates without these modes ("normal" mode).



(a)



(b)

Figure 2. (a) The Model T-42 gripper. (b) a detailed schema showing actuators and spring mechanisms.

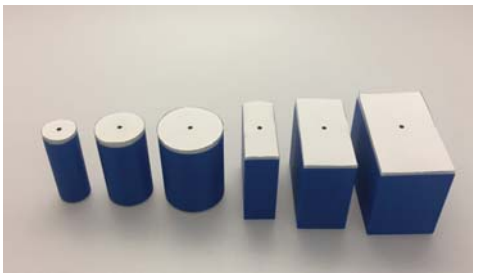


Figure 3. Objects used in the experiments. Cylinders with 2, 3 and 4 cm diameters, and rectangular prisms with dimensions 2x4, 3x5 and 4x6 cm. The objects' weights vary between 12 g and 75 g.

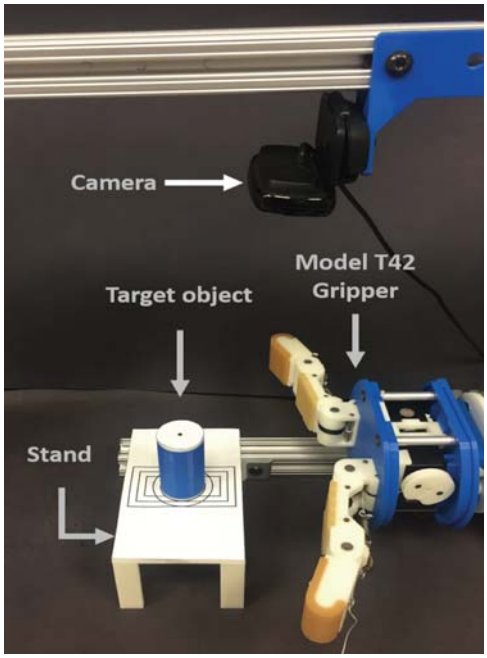


Figure 4. Experimental setup.

The features used for the training are collected using the recorded data streams as follows: we played back each stream to detect the instances when the four modes occur. For each instance, we recorded positions, velocities, loads and load changes of the actuators as actuator data. From the vision sensor we recorded the positions, velocities and orientations of each link and the target object using fiducial markers. The fiducial measurements were taken with respect to a fixed marker positioned at the base of the gripper, so that the measurements are robust to camera repositioning. The user commands are also added to the feature vector which are Cartesian and actuator velocity references. The list of all the features can be seen in Table I. Here, link 1 and 3 are the distal links of the left and right fingers respectively, and link 2 and 4 are the proximal links.

For the 'normal', 'drop' and 'stuck' modes, we collected 50 data points for each object with the procedure explained above. For the 'slide' mode, data is collected only for the rectangular objects, as sliding is not observed while manipulating the cylindrical objects. In total, we collected 1050 data points (50 [data_points] x 3 [cylindrical_objects] x 3 [modes] + 50 [data_points] x 3 [rectangular objects] x 4 [modes]).

IV. Classification Results

A. General Classification Performance

First, we aim for the best classification result that we can obtain using all the available data in Table I. We utilized TPOT [28] and Scikit-learn [29] to investigate the performance of six supervised learning methods: extra trees [30], gradient boosting [31], neural nets [32], random forests [33], ridge classifier and support vector machine (SVM) [34]. For each algorithm, a single classifier was trained using 75% of the collected data, using stratified sampling by label. For testing the classification performance, we ran the classifiers on a test set of the remaining 25% of the data. The

TABLE I. Features Recorded to be Used for Training the Classifiers. Yellow Cells: Actuator Features; Blue Cells: Visual Features; Green Cells: User Commands

	Features		Features
1	Act. 1 position	20	Link 4 orien.
2	Act. 2 position	21	Link 1 vel. x
3	Act. 1 velocity	22	Link 1 vel. y
4	Act. 2 velocity	23	Link 2 vel. x
5	Act. 1 load	24	Link 2 vel. y
6	Act. 2 load	25	Link 3 vel. x
7	Act. 1 load ch.	26	Link 3 vel. y
8	Act. 1 load ch.	27	Link 4 vel. x
9	Link 1 pos x	28	Link 4 vel. y
10	Link 1 pos y	29	Object pos. x
11	Link 2 pos x	30	Object pos. y
12	Link 2 pos y	31	Object orien.
13	Link 3 pos x	32	Object orien. ch.
14	Link 3 pos y	33	Object vel. x
15	Link 4 pos x	34	Object vel. y
16	Link 4 pos y	35	Cartesian vel. ref. x
17	Link 1 orien.	36	Cartesian vel. ref. y
18	Link 2 orien.	37	Act. 1 vel. ref.
19	Link 3 orien.	38	Act. 2 vel. ref.

TABLE II. Classification Results Using All The Features with Six Different Learning Methods

Classifier	Score
Extra trees	96.1%
Gradient boosting	95.9%
SVM	95.8%
Random forests	94.9%
Ridge	85.2%
Neural nets	52.4%

classification scores are given in Table II. It can be seen that extra trees, gradient boosting and SVM performed very close to each other with around 96% classification rate. The random forests classifier gave slightly worse results comparing to the prior algorithms. It can be said that these classifiers can successfully generalize the modes for objects with various shapes and sizes in the experiment set. Ridge and neural nets considerably underperformed for our data; the former due to its limitations as a linear model, and the latter due to the limited size of the training data, leading to overfitting and poor generalization.

The confusion matrix for the best-performed classifier, extra trees, is given in Figure 5. This matrix shows that the classifier mostly has difficulty differentiating between sliding and normal cases. The 'normal' mode is misclassified as 'slide' for 3% of the cases, and 'slide' mode is misclassified as normal for 5% of the cases. The 'normal' mode gets the most false positives as it neighbors all of the other states (this will be demonstrated in the Fig. 7 and 8 shortly). One reason for higher misclassification for sliding is that the motions that cause sliding for rectangular objects with flat surfaces do not cause sliding with cylindrical objects. At this point, using tactile sensing and/or knowledge about the contact surface curvature could be useful for improving the classification performance further. Since we do not want to complicate the design of our gripper, we will aim to visually identify the local curvature and utilize it for

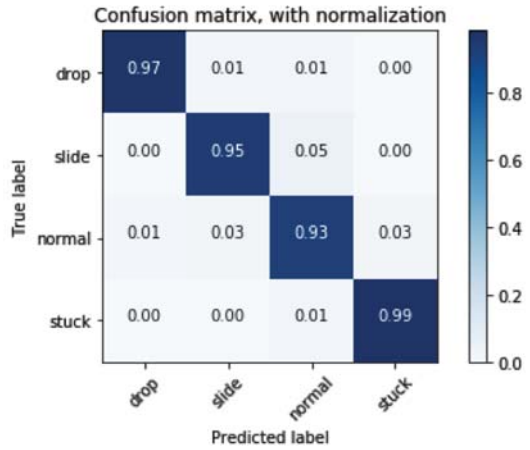


Figure 5. Confusion matrix for Extra Trees classifier using all available features.

TABLE III. CLASSIFICATION RESULTS WITH EXTRA TREES CLASSIFIER FOR THREE DIFFERENT SCENARIOS

Training set	Score %
All Data	96.1 ± 0.014
Only actuator features + act. commands	93.9 ± 0.022
Only visual features + Cart. commands	93.0 ± 0.030

mode recognition in our future work.

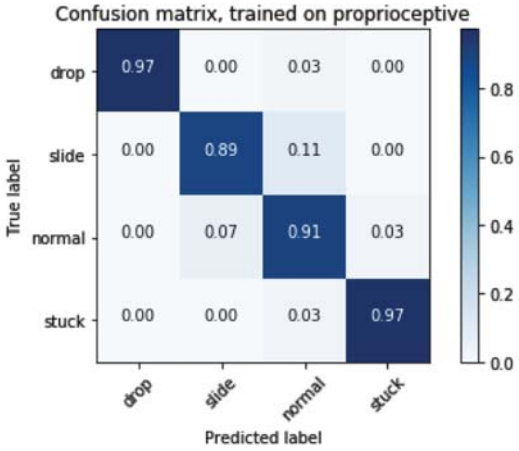
B. Performance of the Sub-components

Next, we investigated and compared the classification performance for three main scenarios. In the first scenario both actuator data and visual information were assumed to be available. In the second scenario, we assumed that the visual features cannot be easily detected (due to occlusions or image processing challenges) so that only actuator data were needed to be used together with actuator space velocity references for detecting the modes (features 1-8 and 37-38). In the third scenario, the classifier did not have access to actuator information, but externally observed the system from the camera, so that only the visual features were available (features 9-36).

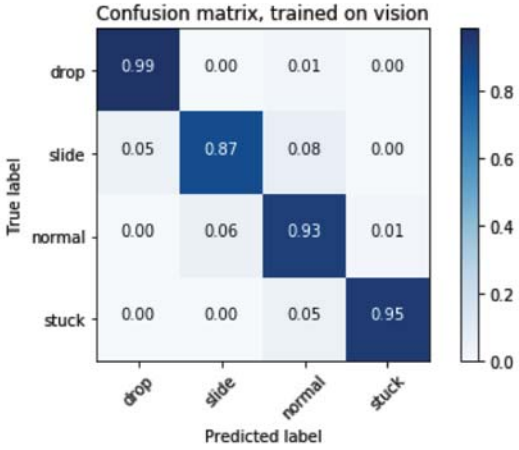
For evaluating these cases, the extra trees classifier was used. The classification results for the three scenarios are given in Table III and confusion matrices are presented in Fig. 6. We see that the classification performance slightly drops using only actuator data or visual sensing. For both of the cases, it becomes harder to differentiate between the "slide" mode and the "normal" mode. For the "drop" and "stuck" modes, classification rates are still quite high for all of the scenarios.

C. Generalization Performance for Different Sizes

Next, we analyzed the ability of the classifier to generalize the mode classification for specific object sizes not included during the training phase. In doing so, we trained the system with only small and medium objects (both cylindrical and rectangular) and tested the classifier on the large objects. The 95% confidence interval of the accuracy can be seen in Table IV after 100 runs. Comparing to the results in Table III, we observe only a marginal drop in the



(a)



(b)

Figure 6. Confusion matrices for (a) using only actuator features and user commands, (b) using only visual data and user commands.

TABLE IV. CLASSIFICATION RESULTS WITH EXTRA TREES TRAINED WITH SMALL AND MEDIUM SIZE OBJECTS AND TESTED ON THE LARGE SIZE OBJECTS

Training set	Score %
All features	94.4 ± 0.016
Only actuator features + act. commands	92.9 ± 0.014
Only visual features + Cart. commands	92.5 ± 0.022

classification performance; the classifier can extrapolate the results to objects outside the training dataset in this case.

D. Feature Importance

Following that, we ran a feature importance analysis with the extra trees classifier for vision only and actuator data only scenarios by averaging the importance values for 100 test runs. The top ten important features for these cases are presented in Table V. Here, we see that user commands get relatively high importance in both of the cases. This is expected, as the direction that the object is heading towards greatly affects which mode the object will fall into. This can also be seen from Fig. 7, which is obtained by projecting classification results into Cartesian space using object

TABLE V. FEATURE IMPORTANCE VALUES FOR VISION-ONLY AND ACTUATOR DATA-ONLY SCENARIOS.

Vision only	
Feature	**Imp.**
Car. vel. ref. y	0.15
Car. vel. ref x	0.09
Link 1 pos. y	0.08
Obj. pos. y	0.08
Link 3 pos. y	0.06
Link 4 pos. x	0.04
Link 2 vel. x	0.03
Obj. pos. x	0.03
Link 1 vel. y	0.03
Link 3 pos. x	0.03

Actuator data only	
Feature	**Imp.**
Act. 2 load	0.18
Act. 1 load	0.17
Act. 2 pos.	0.15
Act. 1 pos.	0.14
Car. vel. ref. y	0.11
Car. vel. ref. x	0.06
Act 1 vel. ref.	0.06
Act. 2 vel. ref.	0.05
Act. 1 vel.	0.04
Act. 2 vel.	0.04

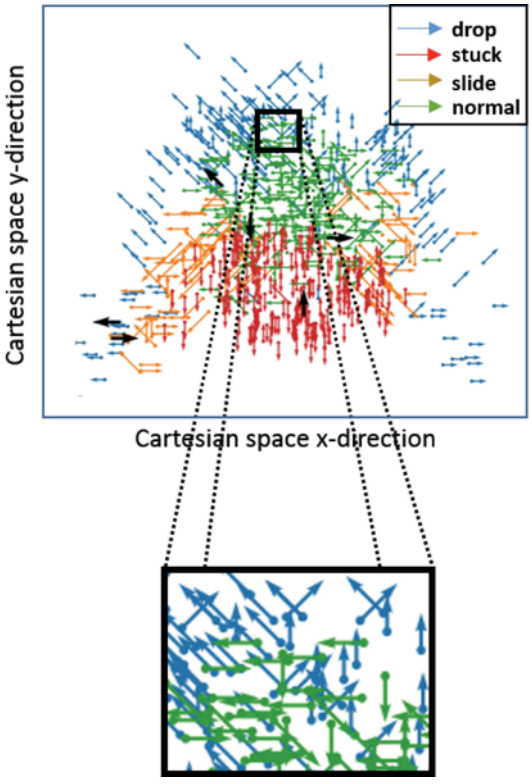


Figure 7. Classification results projected to the Cartesian space using object positions. The arrows indicate the velocity commands. Bold black arrows are the misclassified data points.

positions. The arrows indicate the Cartesian velocity reference given to the system. Here, for similar object positions, the classification results differ for the given velocity commands. For the case that only actuator sensors are used, actuator loads and positions have the highest importance, whereas actuator velocities has the least. For the vision only case, the importance is distributed similarly to positions and velocities of all the links. Object *y*-position also has a high importance as for many of the manipulation modes, the position of the object in the workspace is a strong indicator.

E. Using the Classifiers in the Control Loop

We analyzed the performance of the extra trees classifier by integrating it to an online control scheme (using all the features). For this implementation, we used ROS and utilized the "ml_classifiers" package for interfacing with the classifier. For the implementation of the classifier, we used OpenCV's "Random Trees" functions, and for detecting the fiducial markers we used OpenCV's ArUco Markers library. We refer the reader to our video attachment for the demonstration that the classifier identifies the modes online and stops the controller to avoid dropping the object.

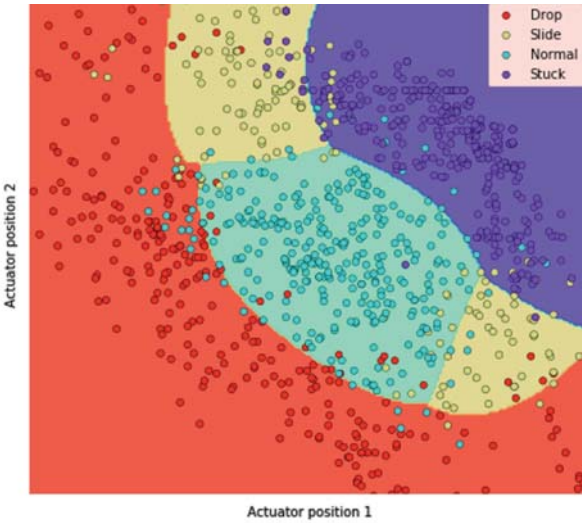


Figure 8. Regions for the four modes in actuator space obtained by SVM classifier.

In addition, considering that the actuator positions are highly important for detecting the manipulation modes, we believe that the results of the classification algorithms can be utilized for in-hand manipulation planning in the actuator space. By using the SVM classifier and using only actuator positions as features, we obtain 89% classification score, and partitioned the actuator space for different modes as presented in Fig. 8. As our future work, we are planning to utilize partitioned maps for acquiring reward functions that can be used for manipulation planning in actuator space.

F. Summary of the Results

- If actuator and visual features are used together, manipulation modes can be detected with 96.1% success rate with a single classifier for objects of various shapes and sizes.
- Using only actuator features or only visual features slightly drops the classification performance comparing to using both.
- With a feature importance analysis, we have concluded that actuator positions, actuator loads, and user commands are the key features for detecting the manipulation modes; up to 89% classification performance can be achieved by utilizing only actuator positions.
- We also obtain higher than 92% classification rate for objects outside the training set.
- Finally, the classifier is used to conduct online in-hand manipulation.

V. Conclusion

In this work, we analyzed the performance of supervised learning algorithms for estimating modes of in-hand manipulation. Our strategy does not require a task model for identifying the "drop", "stuck", "slide" and "normal" modes. Naturally, the specific classifiers obtained in this paper are only valid for our target system. Nevertheless, we believe that the analysis presented in this paper gives very valuable insights for in-hand manipulation with grippers of similar topology (e.g. the OpenHand grippers [7]), and provides a methodology for investigating manipulation modes for other types of grippers.

As a future work, we will concentrate our efforts for utilizing the classifiers in the manipulation planning schemes and vision-based control strategies as well as investigating features and classifiers that will apply more generally to other hands, objects, and tasks.

References

[1] R. R. Ma and A. M. Dollar, "On dexterity and dexterous manipulation," in *IEEE 15th International Conference on Advanced Robotics (ICAR): New Boundaries for Robotics*, 2011, pp. 1–7.

[2] I. M. Bullock, R. R. Ma, and A. M. Dollar, "A hand-centric classification of human and robot dexterous manipulation.," *IEEE Trans. Haptics*, vol. 6, no. 2, pp. 129–44, Jan. 2013.

[3] A. Bicchi, C. Melchiorri, and D. Balluchi, "On the mobility and manipulability of general multiple limb robots," *IEEE Trans. Robot. Autom.*, vol. 11, no. 2, pp. 215–228, 1995.

[4] R. Michalec and A. Micaelli, "Stiffness modeling for multi-fingered grasping with rolling contacts," in *10th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2010, pp. 601–608.

[5] A. M. Okamura, N. Smaby, and M. R. Cutkosky, "An overview of dexterous manipulation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2000, vol. 1, pp. 255–262.

[6] B. Calli and A. M. Dollar, "Vision-based Precision Manipulation with Underactuated Hands: Simple and Effective Solutions for Dexterity," in Proc. of the *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 1012–1018.

[7] R. R. Ma and A. M. Dollar, "Yale OpenHand Project: Optimizing Open-Source Hand Designs for Ease of Fabrication and Adoption," *IEEE Robot. Autom. Mag.*, vol. 24, no. 1, pp. 32–40, 2017.

[8] R. Deimel and O. Brock, "A novel type of compliant and underactuated robotic hand for dexterous grasping," *Int. J. Rob. Res.*, vol. 35, no. 1–3, pp. 161–185, 2016.

[9] M. G. Catalano, G. Grioli, E. Farnioli, a. Serio, C. Piazza, and a. Bicchi, "Adaptive synergies for the design and control of the Pisa/IIT SoftHand," *Int. J. Robot. Res.* , vol. 33, no. 5, pp. 768–782, 2014.

[10] D. Prattichizzo, M. Malvezzi, M. Aggravi, and T. Wimböck, "Object motion-decoupled internal force control for a compliant multifingered hand," *Proc. of IEEE Int. Conf. Robot. Autom.*, pp. 1508–1513, 2012.

[11] B. C. Calli and A. M. Dollar, "Vision-Based Model Predictive Control for Within-Hand Precision Manipulation with Underactuated Grippers," in Proc. of the *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2839–2845.

[12] R. R. Flanagan, P. Vetter, R. S. Johansson, and D. M. Wolpert, "Prediction precedes control in motor learning," *Curr. Biol.*, vol. 13, no. 2, pp. 146–150, 2003.

[13] H. Dang and P. K. Allen, "Learning grasp stability," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 2392–2397, 2012.

[14] Y. Bekiroglu, D. Kragic, and V. Kyrki, "Learning grasp stability based on tactile data and HMMs," in *19th International Symposium in Robot and Human Interactive Communication*, 2010, pp. 132–137.

[15] Y. Bekiroglu, J. Laaksonen, J. A. Jørgensen, V. Kyrki, and D. Kragic, "Assessing grasp stability based on learning and haptic data," *IEEE Trans. Robot.*, vol. 27, no. 3, pp. 616–629, 2011.

[16] D. Katz, Y. Pyuro, and O. Brock, "Learning to Manipulate Articulated Objects in Unstructured Environments Using a Grounded Relational Representation," *Robot. Sci. Syst. IV*, p. 254, 2009.

[17] J. Sturm, C. Stachniss, and W. Burgard, "A probabilistic framework for learning kinematic models of articulated objects," *J. Artif. Intell. Res.*, vol. 41, pp. 477–526, 2011.

[18] M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal, "Learning force control policies for compliant manipulation," *in Proc of the IEEE Int. Conf. Intell. Robot. Syst.*, pp. 4639–4644, 2011.

[19] D. R. Faria, R. Martins, J. Lobo, and J. Dias, "Extracting data from human manipulation of objects towards improving autonomous robotic grasping," *Rob. Auton. Syst.*, vol. 60, no. 3, pp. 396–410, 2012.

[20] H. Van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning Robot In-Hand Manipulation with Tactile Features," in *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, 2015.

[21] K. D. Katyal, E. W. Staley, M. S. Johannes, I.-J. Wang, A. Reiter, and P. Burlina, "In-Hand Robotic Manipulation via Deep Reinforcement Learning," *30th Conference on Neural Information Processing Systems (NIPS), Work. Deep Learn. Action Interact.*, 2016.

[22] R. Zollner, O. Rogalla, R. Dillmann, and M. Zollner, "Understanding users intention: programming fine manipulation tasks by demonstration," *Proc. of the IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 2, no. October, pp. 1114–1119, 2002.

[23] U. Prieur, V. Perdereau, and A. Bernardino, "Modeling and planning high-level in-hand manipulation actions from human knowledge and active learning from demonstration," *Proc. of the IEEE Int. Conf. Intell. Robot. Syst.*, pp. 1330–1336, 2012.

[24] M. A. Armada, A. Sanfeliu, and M. Ferre, "Slip Detection in Robotic Hands with Flexible Parts," *Adv. Intell. Syst. Comput.*, pp. 153–167, 2014.

[25] C. Melchiorri, "Slip detection and control using tactile and force sensors," *IEEE/ASME Trans. Mechatronics*, vol. 5, no. 3, pp. 235–243, 2000.

[26] A. A. S. Al-Shanoon, S. A. Ahmad, and M. K. b. Hassan, "Slip detection with accelerometer and tactile sensors in a robotic hand model," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 99, p. 12001, 2015.

[27] V. A. Ho, T. Nagatani, A. Noda, and S. Hirai, "What can be inferred from a tactile arrayed sensor in autonomous in-hand manipulation?," *Proc. of the IEEE Int. Conf. Autom. Sci. Eng.*, pp. 461–468, 2012.

[28] R. S. Olson and J. H. Moore, "TPOT: A Tree-based Pipeline Optimization Tool for Automating Machine Learning," in *Proceedings of the Workshop on Automatic Machine Learning*, 2016, vol. 64, pp. 66–74.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[30] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, 2006.

[31] J. H. Friedman, "Stochastic gradient boosting," *Comput. Stat. Data Anal.*, vol. 38, no. 4, pp. 367–378, 2002.

[32] R. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, vol. 4, no. 2, pp. 4–22, 1987.

[33] L. Breiman, "Random forests," *Mach. Learn.*, pp. 5–32, 2001.

[34] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, Ny Springer, 2001.