Improvement through Interaction in Robotic Manipulation: Challenges and Strategies

Erol Şahin

KOVAN Research Lab., Dept of Computer Engineering, Middle East Technical University, Turkey.
Robotics Institute, Carnegie Mellon University, 5000 Forbes Av., 15213, PA, USA.

***Abstract*— This paper proposes a new framework, called *Improvement through Interaction*, towards developing better manipulation skills on robots. Arguing that neither modelling-only nor learning-only approaches can provide the desired level of manipulation skills we propose a framework that can integrate the best of both worlds. Towards this end, we review the existing modelling paradigms describing what they can provide, and discussing their limitations. We then propose a number of strategies on how sensory-motor data acquired from previous interactions can be used to improve the accuracy, robustness and speed of built-in manipulation skills.**

I. Introduction

Robot manipulators operating within factories has long been a key element of industrial manufacturing, and is regarded as a success story for robotics. However, the competency of robots in manipulation tasks to be carried out in unstructured and dynamic human environments is still far below of expectations. Although major advances have taken place on many fronts, ranging from the mechatronic design of manipulators to the development of methods in perception, planning and learning, robotic manipulation still remains as a major challenge. Successful demonstrations in specific tasks such as folding a towel folding[1] or grasping a bottle from fridge[2], has not yielded general and integrated manipulation systems that can operate with to novel objects, in new environments and novel tasks at hand.

II. Motivation

We are interested in how learning can be used to improve and expand the manipulation abilities of a robot. Robots, like all other gadgets, will come out of their box with a certain amount of built-in knowledge that will enable them to perform a set of tasks with a reasonable performance at the push of a button. However, in manipulation, where humans perform remarkably well, expectations are high. We argue that, the *robot-out-of-a-box* should come with both (1) a set of *built-in* capabilities that would enable it to solve manipulation tasks in unstructured human environments, as well as (2) the learning ability to extend and improve, and with guarantees not to hinder, its capabilities through its interactions over the lifetime of the robot with minimal (none wherever possible) user effort. Towards this end, we argue for the development of a framework that can provide *improvement through interaction* to robot's existing manipulation competences.

This study is motivated by the following two observations:

- *Not every aspect of manipulation can be modelled.* Models of manipulation require access to properties of robot, objects and the environment such as friction coefficients between different bodies, inertial parameters, pressure distributions, that are not immediately perceivable. Therefore, these properties remain as open parameters of these models, and need to be estimated during the actual manipulation process.
- *Not every aspect of manipulation needs to learned from scratch.* Learning is a costly process and should be minimized. The interaction of the robot with its environment, despite providing the best source of information, is time-consuming and hence not desirable due to the mechatronic wear-and-tear of the robot as well as the risks it poses to the environment and humans. Therefore, learning from scratch, an approach that has been used for the demonstration of learning capabilities on robots, should be avoided. Instead, the robot should build on the existing models, which were developed through our understanding of kinematics and dynamics, and focus on adapting them.

A. Objectives

Learning has mostly been studied as a problem in the most general case, where the performance of learning in manipulation is often evaluated through its performance on a large set of novel objects and on novel tasks. Although much progress has been made towards this elusive goal, manipulation capabilities of robots still fall short of our expectations.

This study puts forward a less ambitious but more pragmatical objective for learning. *We argue that the robot will be manipulating the very same objects using the very same behaviours over and over*. In a domestic environment, the very same fridge door needs to be opened, the very same mug will be grasped. Therefore instead of trying to learn models that can generalize over multiple objects and over multiple tasks, we seek models that can improve interactions with a specific object for a specific task.

Specifically we aim to focus on improvement strategies that are;

- *Behavior-object specific.* Improvements should be specific for each behavior-object pair, such as the grasping

the same mug, or opening the same drawer. We argue that by indexing the learned improvements with the behavior-object pair, one can achieve better performance for that particular interaction while sacrificing generalization. Although this approach would increase the number of improvements to be learned, we argue that in a practical scenario, it would still remain feasible.

- *Model-based.* Improvement strategies should address the limitations of a modelling paradigm and aim to improve its shortcomings by using the data from previous interactions. For each object and behaviour, we assume to have access to an a priori model of robot-object interaction either in the form of it. In particular, we will assume that all objects have rigid bodies since this would allow us to use mechanical manipulation models.
- *Data-driven.* The sensory-motor data gathered during interactions, possibly coupled with limited user feedback, will be used to improve the competence of the robot.

III. Phases of Manipulation

Manipulation tasks appear in a wide range of domains, ranging from assembly and welding in factory automation to cleaning and organising in household choirs[3]. Within this study, we will restrict ourselves to the manipulation of rigid bodies and focus on quasi-static interactions with the object.

We argue that most manipulation behaviors would consist of four different phases as follows: Consider a pick-and-place behavior operating on a rigid body which consist of the following four phases.

- *Reaching*: This phase covers the duration between the start of the action and the initial contact of the manipulator with the object. Different from subsequent phases, during this phase the manipulator is expected to move in free space and no physical contact with the environment is usually foreseen.
- *Prehensile manipulation*: This phase covers the duration between the initial contact of the manipulator with the object to the point when either the contact with the object is lost or the object is stabilized by the end-effector through force- or form-closure. Grasping a bottle by closing fingers, pushing an object on a table, as well as opening a fridge door fall under this phase since the movement of the object is only constrained partially by the action of the manipulator. Prehensile manipulation actions, can be used to reduce the uncertainty in the object pose due to the mechanics of interaction between the object and other constraints in the environment.
  We would like to point that in literature, if the object is stabilised at the end of the interaction, it is typically referred as *grasping*. If not, the interaction is referred as *prehensile manipulation*. In this study we termed this phase as prehensile manipulation regardless of the outcome.
- *Transportation*: During this phase, the end-effector transports the grasped object to another location. This phase is similar to the *reaching* phase where the manipulator is effectively augmented with the object. Note that if the grasped object is a tool, the *augmented manipulator* can be used for reaching and prehensile manipulation as well.
- *Release*: During this phase, the contact between the manipulator and the grasped object is lost in a controlled manner such that the object ends up in a desired pose. Although this phase resembles to prehensile manipulation, since the object is partially controlled, the end points are different. At the end of this phase, the object will be left to the 'control' of external forces, such as gravity and friction, and the final pose of the object is determined by the static analysis.

We will now analyze the modelling frameworks for developing built-in manipulation skills, discuss their limitations and propose strategies for improving these skills using data gathered from prior interactions.

IV. Modelling Frameworks for Manipulation

There are five modelling frameworks that are relevant to manipulation. Kinematic and dynamic modelling frameworks allow the modelling of the manipulator whereas quasi-static modelling focuses on the interaction between the end-effector and the object. Motion planning framework aims to generate trajectories for the manipulator in free space while avoiding obstacles on its path. Perceptual modelling allows one to use the sensor and manipulator characteristics to predict the perception of the manipulator and/or the object during the interaction.

A. Kinematic modelling

Kinematic modelling links the task space representation of a manipulator with its joint space representation using the forward kinematic model using the joint and link parameters of the manipulator chain. Inverse kinematic models provide a mapping in the opposite direction. The forward kinematic model also relates the joint velocities of the manipulator to the (linear and angular) end-effector velocities in the task space using the Jacobian matrix. Moreover, the transpose of the Jacobian matrix relates the forces of the actuator and the force generated at the end-effector.

The kinematic model of a manipulator can be easily created from its link and joint parameters and constitutes the minimal built-in model that can be assumed to exist in all manipulators. Although the inverse kinematics problem can become ill-posed for redundant manipulators, methods that can solve them analytically or numerically using iterative optimizations are available.

B. Dynamics modelling

Dynamics modelling describes the effect of external forces on the motion of a manipulator. This description is typically derived by writing the Euler-Lagrange equations or the Hamiltonian of the manipulator, to link the joint accelerations to the forces. The dynamic model takes into account the inertial, centrifugal and Coriolis interactions to compute the

joint accelerations which can then be used to compute the joint velocities and positions in time providing an accurate model of the time evolution of the manipulator.

C. Quasi-static modelling

Quasi-static modelling is different from both kinematic and dynamics modelling in two major aspects. First, it focuses on modelling the end-effector-object interaction and aims to predict the change in the state of the object in response to the forces impinging on it. This is different from kinematic and dynamic modelling where the focus is on the manipulator itself and predictions are made about the state of the manipulator. Second, it explicitly considers the frictional force between the object and the end-effector as well as the environment. It takes a middle ground between kinematic and dynamic modelling and uses the force[1] as the control input to compute the velocity or displacements (if a fixed time interval is assumed). Quasi-static modelling is very relevant for the prehensile phase of the manipulation actions under the assumption that accelerations of the object are short-lived.

These methods take a geometric approach and assumes access to the geometry of the object, as well as the point and direction of the end-effector's contact with the object. Moreover, the model's predictions depends on parameters such as friction coefficients and the pressure distribution, that are not directly observable.

D. Motion planning

Motion planning paradigm aims to find collision-free trajectories for the manipulator from a start to a goal pose while avoiding obstacles in the environment. These methods operate in the configuration space[4], dimensions of which are determined by the degrees of freedom the robot has, and in which the robot is transformed into a point. The objective of the motion planning methods is stated as finding a continuous path in the configuration space from initial to final configuration of the robot in a computationally efficient way. The methods take the geometrical description of the robot and the workspace (such as the obstacles to be avoided) as well as the initial and final pose of the manipulator to generate a trajectory that satisfy the constraints of the task (such as the avoidance of other objects, or parts of the environment, legibility of the action in the presence of human[5]). In manipulators, the generalized coordinates of the robot is used as the configuration space and the forward and inverse kinematic models are used for the conversion of task space representations.

Although the piano mover's problem, an early instance of the motion planning problem, is proven to be PSPACE-hard in computational complexity [6], approximate methods that are based on sampling, potential field and combinatorics are proposed in the literature. Among the sampling-based methods, probabilistic roadmap (PRMs) approaches[7] have become popular and were developed into some of the most successful motion planning methods such as Rapidly exploring Random Trees (RRTs)[8] and CHOMP[9]. Recently kinodynamic planners[10] were proposed to take the non-holonomic constraints of the robots into account during planning.

The motion planning methods, with their theoretical guarantees, provide an ideal built-in capability for manipulation. Specifically, trajectories generated provide a good solution for the reaching phase (typically referred to as *transit paths*, and can also be applied to the transportation phase (typically referred to as transfer paths).

However this approach has two major shortcomings. First, these methods assume the existence of a complete geometric model of the environment and the robot. Although the geometric model of the robot can be easily provided, creating a geometric model of the environment through the on-board perception is problematic as will be discussed in the next bullet. Second, the execution of the trajectories generated by these planners are often executed in open-loop[2] and are typically not responsive changes in the environment during the execution. Third, no physical interaction of the manipulator is foreseen during the execution of the trajectories and the interaction of the object with the manipulator is not modelled.

E. Perceptual modelling

Perceptual modelling relates the readings from sensors to the current state of the manipulator and the world. Specifically, perceptual models represent a mapping between the state of the manipulator and/or the world and the sensory readings. For some sensors, such as joint encoders, the mapping from encoder readings to joint state can be straightforward, whereas for other sensors, such as a camera, a mapping from pixels to object pose can be complex and ill-posed. In the case of images, perception models exist as algorithms that detect the existence of an object (from a library of models) in the image and its pose in the environment.

These perceptual models are perfect for building a world model of the environment for motion planning algorithms. However, their use during the manipulation process is hampered by two factors. First, the manipulator often occludes the object making it difficult to the perceptual model to track the pose of the object. Second, the high computational complexity of these models makes them unsuitable for closing the execution loop.

F. Limitations

Although the modelling paradigms provide a good base for developing and deploying built-in manipulation behaviours, they have a number of limitations.

- Errors related to the manipulator models are due to the errors in the kinematic and dynamic parameters of the manipulator (such as link and joint parameters

[1]Quasi-static analysis uses mainly the direction of the force with the assumption that its value is just enough to cancel-out the frictional force without causing any acceleration, and generating constant velocity.

[2]This is not completely true. The manipulators typically use PID type controllers to close the loop through the joint encoders.

for kinematic modelling, and inertial parameters for dynamic modelling), and the errors in estimation of manipulator state (that is, the joint position, velocity and accelerations). Although the parameters can be determined fairly accurate during the design and calibration of the manipulators, these parameters would no longer be valid in cases when the manipulator is holding an object. Errors in the manipulator state estimation typically stem from the errors at the joint encoders. For instance in tendon driven manipulators, encoders (along with motors) are placed away from the actual joint, and the readings from these encoders contain errors due to stretching.

Although system identification methods[11] for computing the joint and link parameters as well as inertial parameters are available, applying them for temporary changes to the manipulators, such as when the manipulator is transporting an object, is not feasible.

- Errors related to the object and the environment are due to the errors in the object parameters (such as the geometry, mass, inertia, and friction characteristics) and the errors in the estimation of the object state (that is, pose, velocity and acceleration). These errors are created by the limitations of the perceptual modelling that operate on exteroceptive sensors (such as cameras). Some parameters, such as the geometry of the object, is mostly inferred from the images by using the $2\frac{1}{2}D$ representation of the object to compute the parameters of a $3D$ model in its repertoire. Others, such as the inertial parameters of the object, are not visible, and is usually assumed conservatively. Estimating object parameters can only be achieved through the interaction of manipulator, and requires information about the contact (type, normal, friction parameters, etcetera) of the manipulator with the object as well as the motion of the object and the forces acting on the object. The object state, usually estimated by fitting object models onto the current view through RANSAC-like algorithms[12] tend to have errors due to occlusions.
- Errors related to the contact between the object, manipulator and the environment stem from errors in determining the contact type (which is usually assumed), the contact position and normals, as well as the frictional coefficients at these contacts. Although the contact position and normal between the manipulator and the object can be estimated from visual sensors, contact type and frictional coefficients cannot be directly observed. Moreover, contacts between the object and the rest of the environments (such as the contacts between a box and the table surface) are usually not observable.
- Computational limitations in the form of representation and computational complexity of algorithms reduces the utility of certain models. Motion planning paradigm is built on the assumption that a world model is created by the perceptual system from which configuration space can be created using the geometrical attributes of the objects in the world and the robot. As a consequence of this, the C-space representation of objects needs to be re-created as the geometry of the manipulator changes, as is the case when the fingers on the end-effector. Moreover, planning in the C-space is computationally costly, and is not suitable for update during manipulation.

  Perceptual models that estimate the parameters and the state of the object tend to be computationally expensive and prone to occlusions that happen during manipulation and hence not suitable for closing the loop during manipulation.

V. Improvement

We define improvement as the generation of control commands that increase the accuracy, robustness and speed of manipulative actions based on the sensory-motor data acquired from previous interactions.

We propose four strategies for improving a built-in manipulation behaviour:

- Improvement over the estimation of parameters. Certain parameters that affect the interaction, such as the mass, inertia of the object, or the friction parameters between the object and the manipulator are simply unobservable by distal perception. Estimation of these parameters based on priors interactions would allow the robot to make better and more robust predictions and plans leading to improvement in its built-in skills. These parameters can be estimated from the interactions in the prehensile, transportation and release phases of manipulation.

  Other parameters such as the link, joint and inertial parameters of the manipulator may also need to be adapted for cases when the manipulator is transporting an object. Although the basic manipulator parameters can be estimated using system identification methods and their accuracy affect the performance in all phases of manipulation, there is little room for improvement. Yet, temporary update of these parameters to include the affect of the object being grasped during the transportation phase would improve the performance.

  This strategy requires minimal changes to built-in manipulation skills, since it can be implemented as an off-line parameter estimation method that uses prior data. By indexing the estimated parameters with the initial view of the object, one can just feed them to the built-in behaviour to improve the performance of the built-in manipulation skill.
- Improvement of the state transition function. At the heart of each built-in behaviour lies a built-in state transition function, that predicts the next state from using the history of previous[3] states and control commands. This function is typically created using one or more of the models. We argue that the discrepancy between the predicted and actual state at the end of the interaction

[3]If the system is Markovian, then the current state and control command would suffice.

can be used to improve the output of built-in state transition function by learning the residual.

Similar to the first strategy, this one also can easily be incorporated into a built-in manipulation skill by adding the predicted residual to the output of the state transition function. However, the change in the state transition function will in turn change the plans made, and may be difficult to provide a robust and reliable improvement in skill since the state transition function is improved only along the previous plans (or trajectories).

Back-propagating the error between the predicted and the actual states to the different built-in state transition functions remains a challenging task.

- Improvement of the control commands by adding compensations for the unmodelled "disturbances". We argue that disturbances that are due to unmodelled aspects of the interaction are likely to be generate repeatable errors on the outcome of the manipulation behaviour. Sensory-motor data collected from prior interactions can be used to predict these errors and be used in generating control commands to compensate for them.

  This approach is different from the previous strategy in the sense that it aims to improve the control signals, rather than the state transition function by adding compensatory terms. Unlike the previous strategy, this would not interfere with the planning part, and is likely to be more stable in terms of performance.

  Iterative Learning Control[13] provides a framework as well as methods to compute compensatory control signals for unpredicted disturbances experienced in prior interactions (see [14] for a survey). The standard framework assumes full access to state information, and it remains a challenge to adapt it into our problem where complete state information is not always available.
- Improvement of control commands by relating them to observable perceptual signals. Open-loop execution of control commands is brittle against the uncertainties and cannot adapt to changes in an online manner. We argue that observable perceptual signals can be used to generate local closed-loop controllers that either replace or subsume open-loop execution.

  This strategy aims to replace the open-loop execution of plans with local controllers that close the loop using perceptual feedback signals. Differential Dynamic Programming[15] using, a local trajectory optimization algorithm, can be used to produce an optimised trajectory, based on a given cost function, and linear feedback policies around the trajectory to make it robust against small perturbations.

  We would like to note that even if the open-loop execution of the behaviour remains intact, relating the evolution of perceptual signals through the course of the interaction paves the way for monitoring the execution of the action, and be used to abort in cases where the perceptual cues diverge from normal.
- Improvement by generating a new policy from the 'demonstrations' of built-in behaviour. One can use the built-in manipulation behaviour as a "demonstrator" to bootstrap a completely new policy as studied in Learning from Demonstration (LfD) paradigm (see [16] for a review). Specifically, LfD methods aim to derive a policy using the perceived state and action tuples stored during demonstrations. There are three main approaches for learning control policies from the demonstrations data. In the *mapping function* approach, the mapping between the observed state and the action is learned using a regression (if the actions are defined over a continuous space) or classification (if the actions are discrete) method . In the *system model* approach the interaction dynamics of the world is used to develop a (probabilistic) system model, which is then used to derive an optimum policy that would map the perceived state to the actions of the robot using reinforcement learning [17] (see [18] for a recent and detailed survey). The *planner approach* uses a planner to sequence pre- and post-action conditions of actions to achieve a desired goal by making multi-step plans. Recent studies aim to extend these approaches by interlacing demonstration and learning process as well as incorporating user corrections. Optimizing a policy requires the use of a cost function, and inverse reinforcement learning methods[19] can be used to infer the implicit cost function that is being used by the built-in model. Similar to the previous one, this strategy aims at developing policies that would replace the existing skills, for a particular object.

In addition to the specific challenges, the use of different modelling paradigms at different phases of the manipulation makes the problem harder. Specifically, the kinematic and dynamics models of the manipulator are used during the reaching and transportation phases, whereas during the prehensile and release phases, a quasi-static model of the object are used in addition to them. Therefore, the different phases of the manipulation should be detected and a separate strategy for each phase should be employed.

VI. CONCLUSIONS

In this paper, we proposed a novel approach, called *Improvement through Interaction*, towards developing better manipulation skills on robots. We argued that neither modelling-only nor learning-only approaches can provide the desired level of manipulation skills and that a middle-ground that integrate the best of both worlds are necessary. Towards this end, we reviewed what the existing modelling paradigms provide, while underlining their limitations, and proposed strategies on how sensory-motor data acquired from previous interactions can be used to improve the accuracy, robustness and speed of manipulation skills.

ACKNOWLEDGMENT

This study is funded by the European Commission within FP7 under the Marie Curie International Outgoing Fellowship under contact FP7-PEOPLE-2013-628854.

REFERENCES

[1] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel, "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2308–2315.

[2] S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. V. Weghe, "Herb: a home exploring robotic butler," *Autonomous Robots*, vol. 28, no. 1, pp. 5–20, 2010.

[3] M. Cakmak and L. Takayama, "Towards a comprehensive chore list for domestic robots," in *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*. IEEE Press, 2013, pp. 93–94.

[4] T. Lozano-Perez, "Spatial planning: A configuration space approach," *Computers, IEEE Transactions on*, vol. 100, no. 2, pp. 108–120, 1983.

[5] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *Human-Robot Interaction (HRI), 2013 8th ACM/IEEE International Conference on*. IEEE, 2013, pp. 301–308.

[6] J. H. Reif, "Complexity of the mover's problem and generalizations extended abstract," in *Proceedings of the 20th Annual IEEE Conference on Foundations of Computer Science*, 1979, pp. 421–427.

[7] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 4, pp. 566–580, 1996.

[8] S. M. LaValle, "Rapidly-exploring random trees a new tool for path planning," Computer Science Dept., Iowa State university, Tech. Rep. TR-98-11, 1998.

[9] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[10] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[11] J. Hollerbach, W. Khalil, and G. Maxime, "Model identification," in *Springer handbook of robotics*, B. Siciliano and O. Khatib, Eds. Springer, 2008.

[12] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[13] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *Journal of Robotic systems*, vol. 1, no. 2, pp. 123–140, 1984.

[14] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *Control Systems, IEEE*, vol. 26, no. 3, pp. 96–114, 2006.

[15] D. H. Jacobson and D. Q. Mayne, "Differential dynamic programming," 1970.

[16] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[17] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*. MIT Press, 1998.

[18] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[19] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning." in *Icml*, 2000, pp. 663–670.