Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Master Thesis Proposal

Compliant Manipulation with Reinforcement Learning Guided by Task Specification

Abhishek Padalkar

Supervised by

Prof. Dr. Paul G. Plöger
Prof. Dr. Karl Jonas
Sven Schneider
Dr. Matthias Nieuwenhuisen

15 May 2019

Compliant Manipulation with Reinforcement Learning Guided by Task Specification

1. Introduction

Most of the real world robotic manipulation tasks present the need for compliant manipulation, where robot needs to respond to the contact forces while executing the task. Classical planning and control algorithms fail to perform satisfactorily due to the lack of precise models of contact forces and high computational complexity. Various approaches have been proposed to learn compliant manipulation skills with the help of Reinforcement Learning (RL). RL is promising when it comes to learn intelligent behaviors in complex environments with high dimensions. But it requires high number of interactions with environment. In reinforcement learning, an agent learns the skills by exploring the environment and adopting the parameters which governs the trajectory of the agent in the environment. Learning all the parameters of the policy can be computationally very expensive and might require large number of interactions with the environment. Application of RL to robotics is limited by above reason.

On the other hand, task specification approaches, proposed in [1, 8, 11] rely on instantaneous task specifications in task frames where constraint on motion are specified in each direction. Such approaches are more practical because of their deterministic nature which can be used for ensuring the robot's safe operation. But many parameters used in task specification need to be tuned manually which is a tedious task and requires number of human interventions. Moreover, sometimes, the task is too complex to be specified fully using task specification.

Nemec et al. in [14] propose a control algorithm by joining reinforcement learning with intelligent control which learns force policy to open door on top of the specified motion for unlatching and opening door operation. We propose to extend above mentioned work by using task specification approach and using reinforcement learning for searching parameters good enough for executing the task maximizing a given reward function. Our approach combines the features from model based manipulation solutions and reinforcement learning. We propose to bring together the feature of deterministic solution from model based control solutions and self learning capability from reinforcement learning in one solution.

1.1. Problem Statement

- Combining features from reinforcement learning and task specification approach to reduce the time and interactions required in reinforcement learning and reducing need for manual parameter tuning in task specification.
- We plan to solve above problem by dividing the problem into two sub-problems:
  - Model the available knowledge about task and environment using existing task specification framework.
  - Use RL for learning gradually more complete and robust action models for compliant and repetitive manipulation tasks.
- We will demonstrate our approach by learning a demo tasks of
  - Opening door,
  - Cutting vegetables.

2. Related Work

2.1. Task specification

In industrial settings, task are usually specified in the form of sequence of primitive motions[8]. These motion primitives mostly contain point-to-point motion, motion to some default configuration and basic curves. For example, following script taken from KRL interface examples presents the motion primitives as shown in listing 2.1. Though this kind of task specification works in controlled industrial environment, it leaves very little scope for external sensory feedback and the task needs to specified in terms of positional goals.

Listing 2.1: KUKA Robot Language code

```
INI
PTP HOME ; go to HOME joint configuration
LIN P1   ; linear motion to point P1
LIN P2   ; linear motion to point P2
```

In [8], Leidner presented a representation in the form of action templates describing robot action using symbolic representations and geometric process model as shown in listing 2.2. Symbolic representation of the task in PDDL allows classical planners to consider the action in the high level abstract task plan and geometric representation of the task specifies the sequence of low level movement sequences needed to complete the action. Here, geometric representation only specifies high level atomic motion primitives (e.g. *move_hand, move_straight*) and not the parametric representation of the motion primitives. Actual parameterization and control is left to low level control module.

Listing 2.2: Action Template: _object.pick

```
I Symbolic Representation
'''(:action _object.pick:
...
```

```
)'''

II Geometric Representation

def pick(self, manipulator, surface):
''' approach, grasp, and lift an object from a surface '''

graspset = odb.get_property(self.type, 'graspset',
   manipulator)
. . .
return operations
```

In [11], Mason et al. presented the idea behind *Task Frame Formalization - (TFF)* for specifying compliant task. Using hybrid control, various control specifying modes are assigned to each axis of the *task frame* or *compliance frame*[13]. Listing 2.3 shows Open Door task taken from [1]. This framework doesn't consider the specification of task quality or motion quality related parameters like velocity damping or instantaneous sensory inputs.

Listing 2.3: Task Specification using TFF: Open Door

```
move compliantly {
    with task frame directions
    xt: force 0 N
    yt: force 0 N
    zt: velocity v mm/sec
    axt: force 0 Nmm
    ayt: force 0 Nmm
    azt: force 0 Nmm
} until distance > d mm
```

iTaSC developed in [2–4], synthesizes control inputs based on provided task space constraints. It formulates a optimization problem considering provided constraints in the environment. In case of conflicting constraints, constraints are weighted in the optimization problem.

2.2. Reinforcement Learning

Ideally, using reinforcement learning, an agent can find an optimal or near optimal behavior for a given reward function, by interacting with the environment. Instead of providing a detailed engineered solution, a reward function provides the feedback on the task which enables agent to learn a optimal solution.

There are four major components in RL algorithm; policy, model, value function and reward function.

2.2.1. Taxonomy in RL

Policy: A policy is a mapping from perceived states of the environment to actions to be taken when in those states.
Model: Model mimics the behavior of the environment which allows agent to infer how environment will behave in response to a particular action.
Value Function The value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state.
Reward Function Reward function is the feedback sent to the agent by the environment. It defines the goal of reinforcement learning problem.

A RL agent takes actions in the environment and receives feedback from the environment. Based on this feedback it learns a value function which gives the expected future return from that state. This value function is then used for building a policy for taking actions in the environment.

Based on deployment of model of the environment, reinforcement learning approaches are categorized in two categories namely model-based and model-free reinforcement learning.

2.2.2. Model Based Reinforcement Learning

In model based reinforcement learning, the model of state transition dynamics is learned from the data. This transition model is used for learning policy by running internal simulations of robots's dynamics and environment. Value function in its simplest form, can be a table which stores all the state and value pairs and policy can be greedy search to take the action which will take the agent to the next state with highest value. But this approach heavily suffers from the accuracy of the transition dynamics model. This model is often learned in simulated environment, hence accuracy of simulation affects the accuracy of model being learned. So in short, we first model the world and use RL algorithms to learn this model of the world. Instead of this we can provide a incomplete parameterized model of the world and use RL to tune it.

2.2.3. Model Free Reinforcement Learning

In model free reinforcement learning, no model of transition dynamics exist, hence value function and optimal actions are derived by trial and error[16]. In model free RL agent learns value function or policy or both (in actor-critic method) directly without learning model. Most of these trials are conducted in simulated environment. Hence the accuracy of simulation of the environment has significant impact on the value function and policy learned by the RL algorithm. The policy needs to be fine tuned in actual environment to compensate for the inaccurate or partially complete simulation.

2.2.4. Policy search methods

Application of reinforcement learning algorithm in robotics suffers from following problems [5]:

- high-dimensional continuous state and action spaces,

- strong real- time requirements, and
- the high costs of robot interactions with its environment.

Policy search methods in RL bypass the learning of value function for policy generation, instead they learn the policy for action generation directly. Hence reinforcement learning problem involves only learning the policy parameters. Deisenroth et al. in [5] present a comprehensive survey on policy learning in RL. According to this survey, widely used policy representations are: 1. Linear policies, 2. Radial basis function networks, 3. Dynamic motion primitives and 4. Miscellaneous representations. They also present survey on the methods used to learn this policies, which are: 1. policy gradients (PG), 2. expectation–maximization(EM)-based updates, and 3. information-theoretic insights.

Nemec et al. use reinforcement learning for learning force control policy for opening door[14]. They take into account the constraints on the door motion and closed kinematic chain resulting from a firm grasp of robot hand on the door handle. While opening the door in such configuration, high internal forces are generated in the direction where motion is not possible. They use this knowledge to learn a compliant force control policy. Our approach is motivated by this work of Nemec et al.[14]. We try to formalize constraints in terms of already existing task specification methods, which can be generalized to more number of compliant manipulation tasks.

2.3. Door Opening Task

Opening door is one the most important task for service and rescue robots. Number of studies have been conducted to investigate this challenging problem using different techniques. Number of approaches rely on detail geometric modeling of the door and analytical trajectory generation. Nagatani et.al. [12] presented an early approach for opening door by modeling the geometry of the door and then synthesizing trajectory to open door by considering geometrical constraints. To cope with the errors in position control of the early days manipulator used in the experiment, they used a force torque sensor module to achieve compliance with

the environment. But this approach highly relies on correct geometric model and accurate execution of designed trajectories.

Approaches presented in [7, 9, 15] use adaptive control algorithms which synthesize the motion by identifying the environmental constraints. In [7], authors proposed to find the radial direction based on force and torque readings. Door is opened by applying velocity in radial direction. This method is prone to failure in case of uncertain grasp position. Also it considers only the case where door is already unlatched.

Some of the methods for door opening motor primitives like Dynamic motion primitives for generating trajectories for opening door. Disadvantage of these approaches is that we need to know the goal pose to generate trajectory hence it confines the condition of completing the task to achieving a goal pose.

2.4. Vegetables Cutting Task

Vegetable cutting task in an interesting problem for robot manipulation considering the complexity of contact forces. Different vegetables requires different cutting forces. Also force profile changes as we cut through the vegetables, e.g. tomato requires much less force once we cut through the skin. Such a complicated force interactions are very hard to model and specified in the task specification. Reinforcement learning provides the solution by allowing robots to learn force control policy over time or phase of the cutting task. This task was chosen for demonstration because of its complex contact force profile. Moreover the task itself does not need any external sensors other than force torque sensors to for estimating rewards and completion of the task.

In [10], Lioutikov et al. solved the task by learning Dynamic Motion Primitives, which is a trajectory control policy. Authors does not consider the problem as compliant manipulation problem hence contact forces are not taken into account. Task is solved just by performing Cartesian trajectory, and hence requiring multiple attempts to cut through the vegetable.

3. Proposed Solution

We plan to solve a compliant manipulation task by providing a incomplete task specification and using Reinforcement Learning to determine un-modeled components in task specification. Provided task specification will greatly reduce the number of dimensions in reinforcement learning problem. And reinforcement learning will learn the parts in the task specification which is hard to model or tune.

The outline proposed solution:

- Provide task specification using TFF *(Task Frame Formalization)*[11], which also consists of parameters for the policy for generating control input.
- Provide policy for control command generation in the task specification (this policy can be a engineered policy for a custom solution or can be a generic parameterized policy e.g. neural network, radial basis function network, etc).
- Design a reward function for the given task.
- Use existing methods to adjust the parameters of the policy to maximize the reward function by carrying out trials in the environment.

Advantages of proposed solution:

- Task specification models the *prior knowledge* about the task which greatly reduces the dimensions of RL problem.
- An engineered policy for control input generation can be provided whenever possible.
- Whenever it is not possible to engineer a policy, a generic policy can be used and limits on control inputs can be provided for ensuring robot's safe operation.
- Open parameters in policy provided in task specification can be learned using RL algorithm, which reduces human interventions and tedious manual parameter tuning.

3.1. Composition

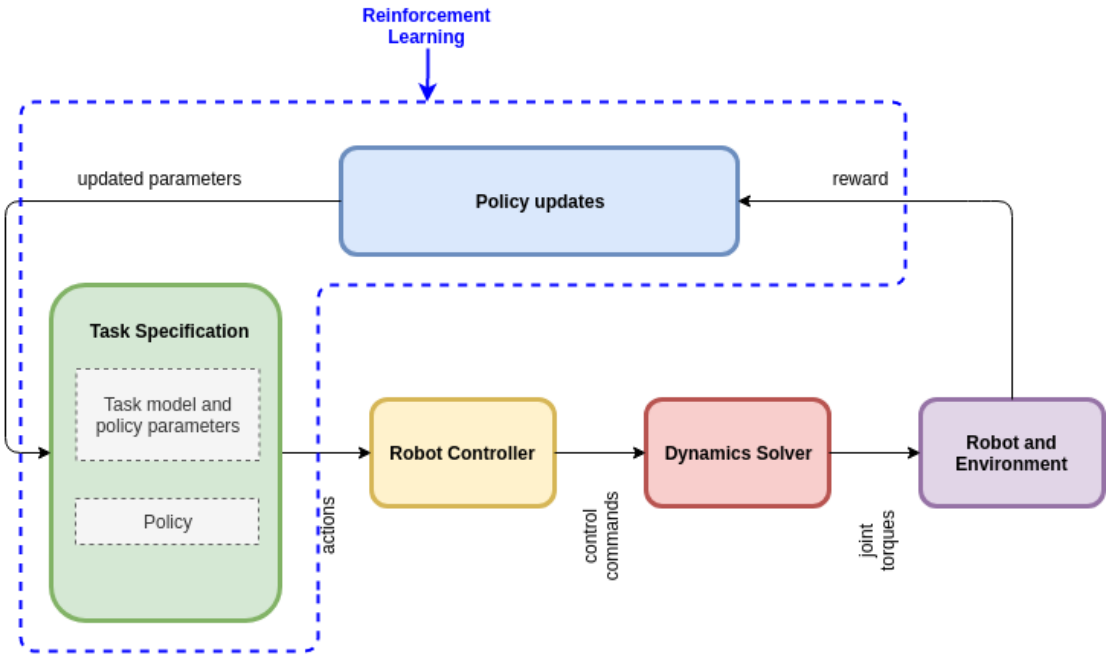

Figure 3.1: Composition

Above figure shows the composition of the proposed solution.

3.2. Task Specification

We propose to use approach given by Mason et al.[11], with further modifications considering limitations on robot and environment. These parameters will also be used for ensuring quality of the task. e.g. smoothness in opening door. Also we introduce force and velocities as functions of robot and environmental parameters and phase of the task.

3.2.1. Examples of task specification:

Opening door

Listing 3.1: Task specification for opening door

```
open_door
{
    with task frame directions
    xt: force 0 N
    yt: force 0 N
    zt: velocity f(environment, robot, task) m/s
    axt: force 0 Nmm
    ayt: force 0 Nmm
    azt: force 0 Nmm
    max_acc = a_max m/ss
    max_dec = d_max m/ss
}until(g(...))
```

To understand the proposed solution, let's have detailed look at task of opening door. Door opening can be solved by applying velocity in tangential direction of door radius until it is opened till certain angle (different metrics of opened door can be brought in, here specified angle is chosen as it is generalizable for different situations). Robot arm needs to be compliant in other dimensions. Task specification given in Listing 3.1, reflects this discussion. Here, task is formalized in door frame where z-axis is perpendicular to the door plane. To open the door smoothly, acceleration and deceleration should be limited. These parameters govern the trapezoidal shape of the velocity applied to door and those are identified as *velocity policy parameters*. These open parameters, maximum acceleration and maximum deceleration, need to be tuned. Policy search methods in reinforcement learning can be deployed to find good enough parameters for maximizing a reward function which is designed to reduce the jerks in the motion and to reduce the time for opening as well.

In vegetable cutting task stated in Listing 3.2, task is represented in chopping board frame such that z-axis is perpendicular to the chopping board and x-axis is in cutting direction. In this task, velocity in x-axis and force in z-axis are not easy to be modeled or parameterized. Here both of these entities can be represented by and generic parameterized policy. Moreover velocity can be learned from demonstrations using methods like Dynamic Motions Primitives[10]. The advantage of the approach is clear here: control policy in only 2 dimensions out of 6 is to be learned, because task specifications defines control inputs in other 4 dimensions.

Listing 3.2: Task specification for cutting vegetables

```
cut_vegetables
{
    with task frame directions
    xt: velocity v(t) m/s
    yt: velocity 0 N
    zt: force f(environment, robot, task, xt) N
    axt: velocity 0 rad/s
    ayt: velocity 0 rad/s
    azt: velocity 0 rad/s
    max_acc = a_max m/ss
    max_dec = a_dec m/ss
}until(g(...))
```

3.3. Reinforcement Learning Algorithm

As already discussed,

1. Task specification approaches are reliable but they need manual tuning of many parameters.

2. Sometimes force interactions in compliant manipulation tasks are so complex that parameterization of task and control policy is not possible.

Both of the above mentioned problems can be tackled by policy search methods in reinforcement learning. In first case, where an engineered control policy can be easily provided, reinforcement learning can be used for tuning policy parameters to maximize the given reward function.

Tasks where model parameterization is not possible, a generic parameterized policy can be used to learn the control policy. It is even possible to initialize this policy using learning from demonstration techniques.

4. Project Plan

Project plan for this master thesis is as follows:

4.1. Work Packages and Milestones

Work Packages		**Targeted date**
WP01	Literature search	25-05-2019
WP02	Analysis of state of the art	
MS01	*Documentation of the state of the art*	
WP03	Modeling demo tasks using task specification approach	30-06-2019
WP04	Design and implementation of the robot controller	
MS02	*Implementing task specification approach for door opening*	
WP05	Integrating RL algorithm in task specification	15-08-2019
WP06	Design and implementation of RL algorithm	
MS03	*Implementing task specification approach with RL for door opening task*	
WP06	Extending implemented solution for vegetable cutting task	30-10-2019
WP07	Experimental evaluation of implemented algorithms	
MS04	*Implementing task specification approach with RL for cutting vegetable task*	

4.2. Deliverables

Deliverables of this project are as follows:

4.2.1. Minimum Viable

- Literature survey and analysis of the state of the art.
- Design and implementation of task specification framework with reinforcement learning.
- Demonstration of task specification framework for door opening task without using reinforcement learning.

4.2.2. Expected

- Demonstration of task specification framework with reinforcement learning for door opening task.

4.2.3. Desired

- Extending proposed approach for cutting vegetable task.
- Demonstration of task specification framework with reinforcement learning for vegetable cutting task.

References

[1] Herman Bruyninckx and Joris De Schutter. Specification of force-controlled actions in the" task frame formalism"-a synthesis. *IEEE Transactions on robotics and Automation*, 12(4), 1996.

[2] Joris De Schutter, Tinne De Laet, Johan Rutgeerts, Wilm Decré, Ruben Smits, Erwin Aertbeliën, Kasper Claes, and Herman Bruyninckx. Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty. *The International Journal of Robotics Research*, 26 (5), 2007.

[3] Wilm Decré, Ruben Smits, Herman Bruyninckx, and Joris De Schutter. Extending iTaSC to support inequality constraints and non-instantaneous task specification. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009.

[4] Wilm Decré, , Herman Bruyninckx, and Joris De Schutter. Extending the Itasc constraint-based robot task specification framework to time- independent trajectories and user-configurable task horizons. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013.

[5] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2, 2013.

[6] Mrinal Kalakrishnan, Ludovic Righetti, Peter Pastor, and Stefan Schaal. Learning force control policies for compliant manipulation. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011.

[7] Yiannis Karayiannidis, Christian Smith, Petter Ögren, and Danica Kragic. Adaptive force/velocity control for opening unknown doors1. *IFAC Proceedings Volumes*, 45(22), 2012.

[8] Daniel Sebastian Leidner. *Cognitive Reasoning for Compliant Robot Manipulation*. PhD thesis, Universität Bremen, 2017.

[9] Martin Levihn and Mike Stilman. Using environment objects as tools: Unconventional door opening. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014.

[10] Rudolf Lioutikov, Oliver Kroemer, Guilherme Maeda, and Jan Peters. Learning manipulation by sequencing motor primitives with a two-armed robot. In *Intelligent Autonomous Systems 13*. Springer, 2016.

[11] Matthew T Mason. Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(6), 1981.

[12] Keiji Nagatani and SI Yuta. An experiment on opening-door-behavior by an autonomous mobile robot with a manipulator. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 2. IEEE, 1995.

[13] Frank Nägele, Lorenz Halt, Philipp Tenbrock, and Andreas Pott. A prototype-based skill model for specifying robotic assembly tasks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018.

[14] Bojan Nemec, Leon Žlajpah, and Aleš Ude. Door opening by joining reinforcement learning and intelligent control. In *2017 18th International Conference on Advanced Robotics (ICAR)*. IEEE, 2017.

[15] Günter Niemeyer and J-JE Slotine. A simple strategy for opening an unknown door. In *Proceedings of International Conference on Robotics and Automation*, volume 2. IEEE, 1997.

[16] Athanasios S Polydoros and Lazaros Nalpantidis. Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent & Robotic Systems*, 86(2), 2017.

[17] Günter Schreiber, Andreas Stemmer, and Rainer Bischoff. The fast research interface for the kuka lightweight robot. In *IEEE Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications How to Modify and Enhance Commercial Controllers (ICRA 2010)*. Citeseer, 2010.

[18] Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. Learning policy improvements with path integrals. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.