
Compliant Manipulation using Reinforcement Learning guided by Task Specification

Master Thesis

Abhishek Padalkar

December 10, 2019



Fraunhofer

FKIE

Motivation

- Most of the real world robotic manipulation tasks present the need for compliant manipulation.
- Robot needs to respond to the contact forces while executing the task.
- Classical planning and control algorithm fail to perform satisfactorily due to the lack of precise model of contact forces and high computational complexity.
- Reinforcement learning can learn the compliant manipulation task but requires high number of interactions.
- Accuracy of simulations affects the learned behavior.

Problem Statement

Develop a learning based approach for achieving compliant manipulation that allows to exploit partial/incomplete models of the task and the environment provided by the 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 demonstrate our approach by learning demo tasks of

- Opening door, and
- Cutting vegetables.



Figure: Mother teaching child to cut vegetables

Related Work

Task Specification in KUKA Robot Language

Listing 1: KRL code

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

- Heavily used in industrial settings
- Reliable
- Difficult to integrate sensory feedbacks
- Limited expressive power [5]

Task Specification using Task Frame Formalism [6]

Listing 2: 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
```

- Using hybrid control, various control modes are assigned to each axis of the *task frame* or *compliance frame* [7].

Other Task Specification [5]

- Leidner presents symbolic representation of the task in PDDL along with the geometric representation of the task using sequence of low level movement sequences needed to complete the action.
- iTaSC synthesizes control inputs based on provided task space constraints [1, 2, 3].
- It formulates an optimization problem considering provided constraints in the environment.

Reinforcement Learning

- **Model-based** reinforcement learning learns the model of environment as well as policy for performing the task.
- Number of interactions required can be very high for learning model of the environment.
- **Model-free** reinforcement learns policy required for achieving the task without learning the model of the environment.
- Model-free reinforcement learning algorithms require less number of interactions.

Policy Search Methods

- Policy search methods are model-free learning methods which directly learn policy to take action in current observable state without learning value function.
- Policy search methods are used heavily in robot control due to their ability to handle high dimensional state-action space.
- Algorithms:
 - Finite difference method : Learns by comparing returns with baseline after exploration in parameter space.
 - Likelihood ratio method / REINFORCE : Uses policy gradient theorem
 - Natural actor critic algorithm
 - Path integral policy improvement (PI^2)
 - Expectation-maximization method (PoWER) : Learns by exploration in the parameter space and by weighing the exploration by returns

Door opening task

- Motion synthesis using the geometric model of the door : requires accurate model of door
- Adaptive control along with on-line estimation of the door parameters using force-torque sensors : sensor noise affects the parameter estimation
- Learning force policies using reinforcement learning : requires numerous interactions with environment
- Learning the task from human demonstrations using Dynamic Motion Primitives : not generalizable for different doors

Vegetable cutting task

- Deep recurrent neural network with model predictive control : requires large amount of data to learn the model
- Using sequence of Dynamic Motion Primitives : purely motion-based approach

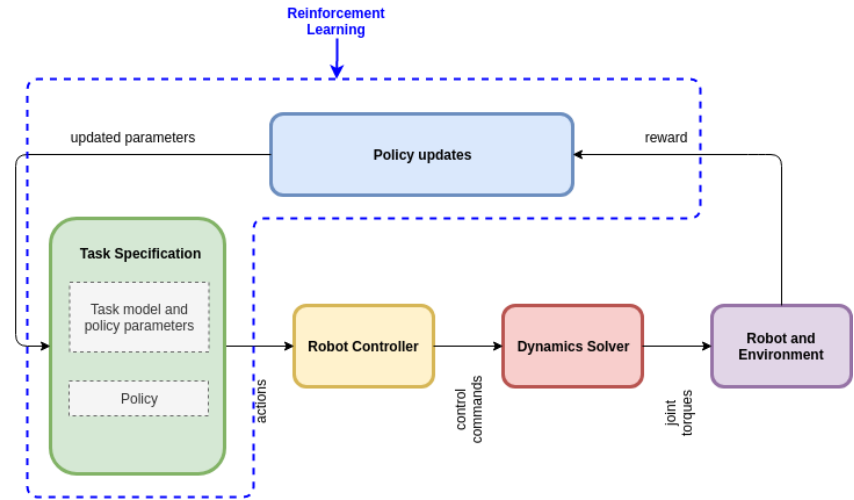
Solution

Approach

- Provide task specification using Task Frame Formalism (TFF), which also contains a policy to be learned by reinforcement learning for generating control commands.
- Design policy for control command generation in the task specification. This policy can be an engineered policy for a well understood problem or can be a generic parameterized policy e.g. neural network or radial basis function network.
- Design a reward function for the given task.
- Use existing reinforcement learning methods to learn parameters of the policy to maximize given reward function by carrying out trials in the environment.

Solution

Components



Task Frame Formalism

Task frame formalism provides guidelines for identifying reciprocal degrees of freedom and task specification using them.

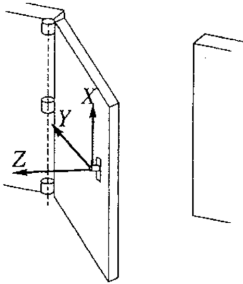


Figure: Task frame selection for door opening task

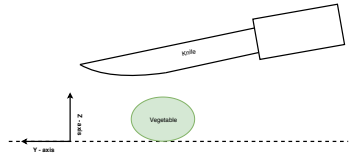


Figure: Task frame selection for vegetable cutting task

Task Specification for opening door

Listing 3: Task Specification using TFF: Open Door

```
move compliantly {  
  with task frame directions  
  xt: force 0 N  
  yt: force 0 N  
  zt: velocity v m/sec  
  axt: force 0 Nm  
  ayt: force 0 Nm  
  azt: force 0 Nm  
} until chord length > d m or force z > f N
```

Task Specification for cutting vegetables

Listing 4: Cut vegetables

```
move compliantly {  
  with task frame directions  
  xt: velocity 0 m/s  
  yt: velocity  $v(t)$  m/s  
  zt: force  $f(\text{environment}, \text{robot}, \text{task})$  N  
  axt: velocity 0 rad/s  
  ayt: velocity 0 rad/s  
  azt: velocity 0 rad/s  
}until distance  $y > d$  m
```

Reinforcement Learning Algorithms

REINFORCE

Input : Policy parameterization θ_h .

Return: gradient estimate $g_{FD} = [g_1, \dots, g_h]$.

1 **repeat**

2 perform a trail and obtain $x_{0:H}, u_{0:H}, r_{0:H}$.

3 **foreach** *gradient element* g_h **do**

4 estimate optimal baseline $b^h = \frac{\langle (\sum_{k=0}^H \nabla_{\theta_h} \log \pi_{\theta}(\vec{u}_k | \vec{x}_k))^2 \sum_{l=0}^H a_l r_l \rangle}{\langle (\sum_{k=0}^H \nabla_{\theta_h} \log \pi_{\theta}(\vec{u}_k | \vec{x}_k)) \rangle}$

5 estimate the gradient element

$g_h = \langle (\sum_{k=0}^H \nabla_{\theta_h} \log \pi_{\theta}(\vec{u}_k | \vec{x}_k)) (\sum_{l=0}^H a_l r_l - b^h) \rangle$

6 **until** *gradient estimate* $g_{FD} = [g_1, \dots, g_h]$ *converged*.

- Uses policy gradient theorem to estimate the gradient of the parameters.
- Adds exploration noise in the actions at every discrete time step.

Policy learning by Weighting Exploration with the Returns (PoWER).

Input : Policy parameterization θ_n .

Return : Updated policy parameters θ_{n+1} .

1 **repeat**

2 sample new policy parameters $\theta_0, \theta_1 \dots \theta_{k-1}$ from $\mathcal{N}(\theta_n, \sigma)$

3 perform k roll-outs and obtain trajectories: $x_{0:H}, u_{0:H}, r_{0:H}$ for each roll out.

4
$$\theta_{n+1} = \theta_n + \frac{\langle (\theta_n - \theta_i) \sum_{t=0}^{T-1} r_{t+1} \rangle}{\langle \sum_{t=0}^{T-1} r_{t+1} \rangle}$$

5 **until** θ_{n+1} is converged.

- Uses expectation-maximization method to estimate the gradient of the parameters.
- Adds exploration in the parameters at the beginning of the episode.

Policy Representation

■ Linear policy

$$f = -(A\dot{y} + B(0.5^2 - (0.5 - \phi_y)^2) + C(1 - \phi_z)) \quad (1)$$

■ Gaussian policy

$$f = -(A\dot{y} + B(0.5^2 - (0.5 - \phi_y^2))) + \sum_{i=0}^{N-1} W\psi_i(\phi_z), \text{ and} \quad (2)$$

$$\psi_i(\phi_z) = \frac{1}{\sqrt{2\pi}\sigma^2} e^{\frac{(c_i - \phi_z)^2}{2\sigma^2}}, \quad (3)$$

where A, B, C are policy parameters,

ϕ_y and ϕ_z are cutting phases in Y and Z direction respectively

W is weight vector,

N is number of Gaussian functions,

c_i is the center of i^{th} Gaussian, and

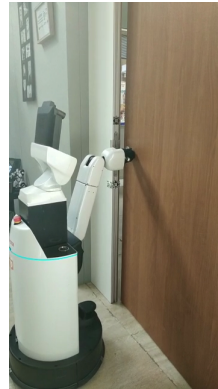
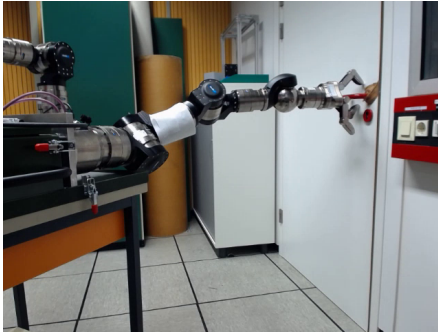
σ is width of Gaussian functions.

Experimental Results

Door opening task

Experiments were performed with two robots:

- Schunk LWA4D equipped with Schunk FTM115 force torque module and Schunk Dexterous Hand
- Toyota Human Service Robot (HSR)



Schunk LWA4D opening unlatched door

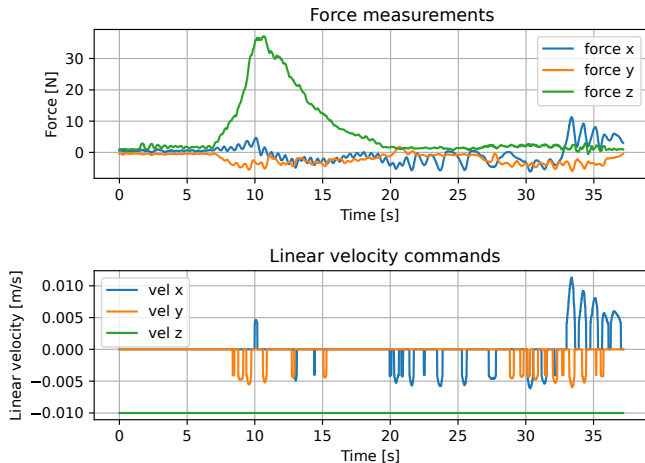


Figure: Force readings at TCP and linear velocity applied to TCP

Schunk LWA4D opening unlatched door

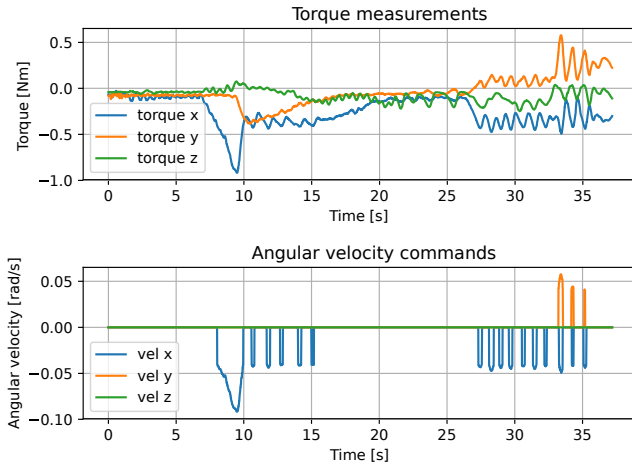


Figure: Torque readings at TCP and velocity angular velocity applied at TCP

Schunk LWA4D opening unlatched door

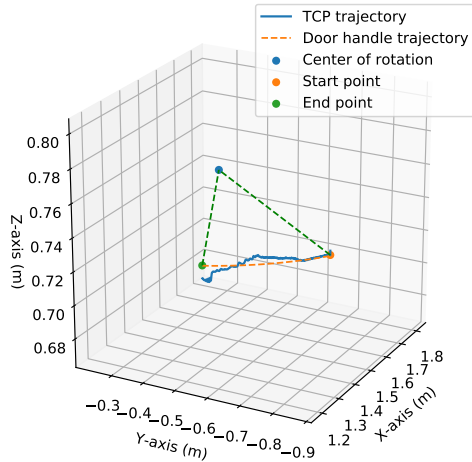


Figure: Geometric analysis of trajectory traced by TCP

Toyota HSR trying to open jammed door

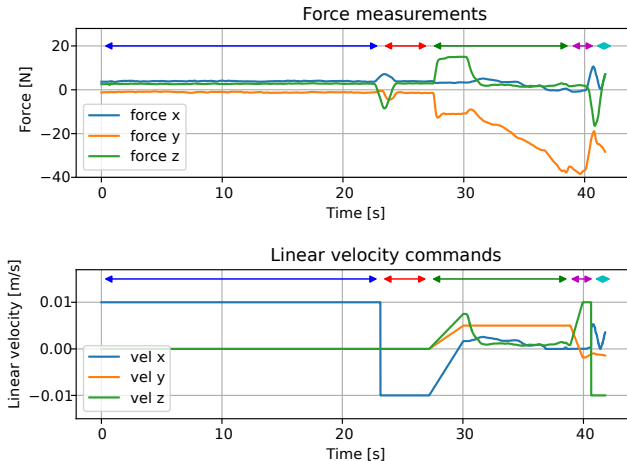


Figure: Force readings at TCP and linear velocity applied to TCP

Toyota HSR trying to open jammed door

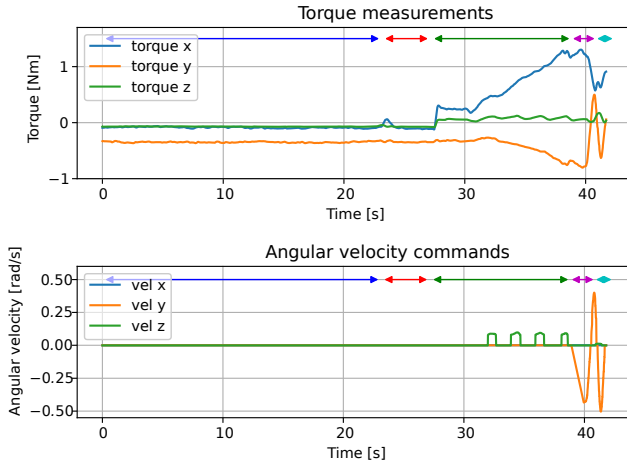


Figure: Torque readings at TCP and velocity angular velocity applied at TCP

Toyota HSR trying to open jammed door

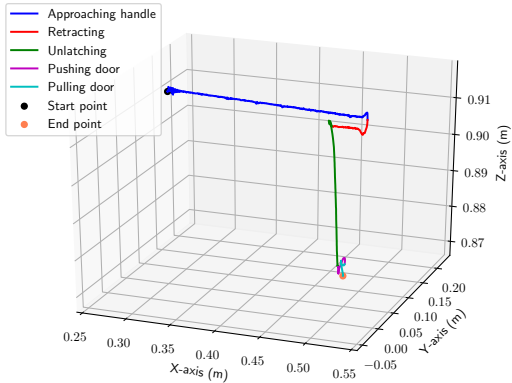


Figure: Geometric analysis of trajectory traced by TCP

Discussion

- Schunk LWA4D was able to open the door every time in 27 trials conducted.
- Robot was able to open the door in the presence of a bump on the floor.
- Toyota HSR robot was successful in detecting the opening direction of door and in determining whether the door is jammed or not.
- The chord length traced while opening door varies because :
 - One or more joint limits reached
 - Operation was stopped to avoid self-collision
- Oscillations were observed in the trajectory because of :
 - Drift in the sensor readings over time
 - Delay in motion command update

Vegetable Cutting Experiment

Testing algorithm in simulation

- Vegetable cutting task was simulated as motion of object through a viscous medium.
- REINFORCE algorithm successfully learned cutting policy.

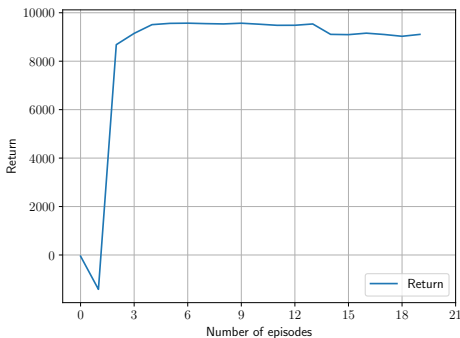


Figure: Performance of REINFORCE algorithm in simulation

Experiments with KUKA LWR 4+

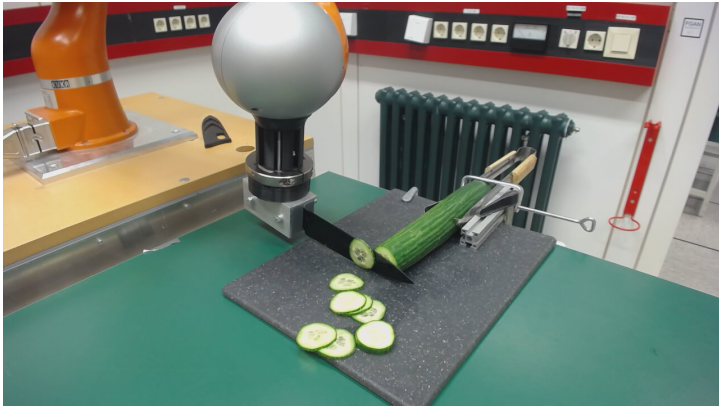


Figure: KUKA LWR 4+ cutting vegetable

Learning Force Policy with REINFORCE Algorithm

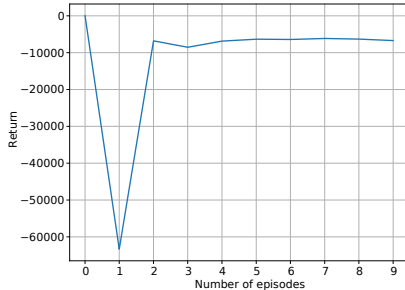
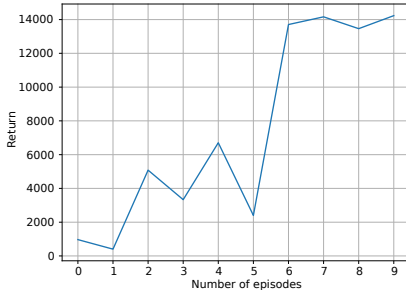


Figure: Performance of REINFORCE algorithm with KUKA LWR 4+

REINFORCE algorithm fails to learn force policy:

- REINFORCE uses high frequency noise for exploration in action space
- The task and robot dynamics acts as a low-pass filter
- Contact dynamics also acts as low pass filter
- Hence noise is damped and REINFORCE does not converge.

Learning Force Policy with PoWER Algorithm



PoWER algorithm successfully learns force policy:

- Policy was learned after 6 episodes
- Power uses low-frequency exploration noise

Figure: Performance of PoWER algorithm with KUKA LWR 4+ using linear policy

Learning Force Policy with PoWER Algorithm

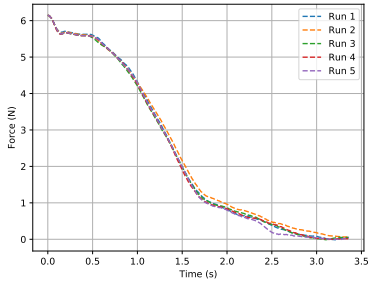


Figure: Force generated by linear policy while cutting vegetable

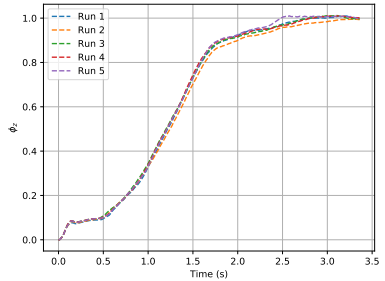
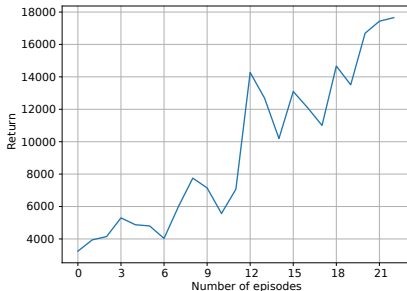


Figure: Cutting phase ϕ_z in Z-direction

Learning Force Policy with PoWER Algorithm



PoWER algorithm successfully learns force policy:

- Policy was learned after 20 episodes
- Increase in number parameters requires more trials to learn the policy.

Figure: Performance of PoWER algorithm with KUKA LWR 4+ using linear policy

Learning Force Policy with PoWER Algorithm

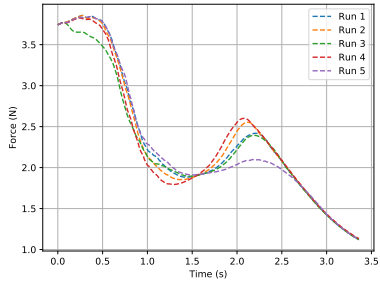


Figure: Force generated by Gaussian policy while cutting vegetable

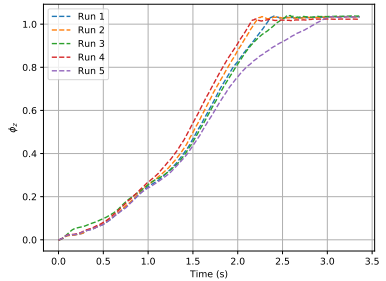


Figure: Cutting phase ϕ_z in Z-direction

Conclusions

- A framework was developed for compliant manipulation using reinforcement learning guided by task specification.
- Task specification approach was successfully tested to solve door opening task using two different robots.
- Evaluation of the task specification approach on two different robots proves the generalization capability of the approach.
- We also extended the door opening solution to detect the opening direction of the door and if the door is jammed.
- Oscillations were observed in the end-effector trajectory while opening door due to sensor noise and low command update rate.
- Parameters of admittance controller were tuned to reduce the oscillations.


Conclusions

- Force policies for cutting vegetable were learned using PoWER algorithm with linear and Gaussian policy representation.
- Policies were learned directly on the robot, without using simulation.
- It was also observed that the cutting depth achieved in previous trial affected the return of the next trial due to more resistant provided by vegetable.
- Task specification framework reduced the dimensions of the problem from six to one, which resulted in faster learning directly on the robot.
- REINFORCE algorithm failed to learn force policy for cutting vegetables because of ineffective exploration with high frequency noise in low-pass filter like environment.
- Increase in policy parameters resulted in increase in number of trials required for learning.
- simple policies can be very effective when one is interested in learning to perform the task rather than learning the complex dynamics of the environment in which the task is being performed.

Future Work

- Auto-tuning for tuning of admittance controller parameters.
- Reducing the delay in ROS communication.
- Extending the approach for tasks such as sweeping the floor, cutting curtains and bags during an investigation of the potentially dangerous environment.
- Extending for tasks involving multi-directional force interactions with the environment such as collaborating with human to transport objects from one place to another.

References I

 J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes, and H. 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.

 W. Decré, , H. Bruyninckx, and J. 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.

References II



W. Decré, R. Smits, H. Bruyninckx, and J. 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.



M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal.

Learning force control policies for compliant manipulation.

In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2011.



D. S. Leidner.

Cognitive Reasoning for Compliant Robot Manipulation.

PhD thesis, Universität Bremen, 2017.



M. T. Mason.

Compliance and force control for computer controlled manipulators.

IEEE Transactions on Systems, Man, and Cybernetics, 11(6), 1981.



References III



F. Nägele, L. Halt, P. Tenbrock, and A. Pott.

A prototype-based skill model for specifying robotic assembly tasks.

In 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018.