Task and Motion Planning using Learning from Demonstration and Reinforcement Learning

Dominik Urbaniak

Master Thesis – Final Presentation
Dr. Alejandro Agostini

Human-centered Assistive Robotics

Technische Universität München

Dirty Dishes

Washing dishes is one common task in our daily life

→ A dishwasher is helpful, but still requires to be loaded

Requirements:

Carefully picking and placing objects of various size and shape in a meaningful order.

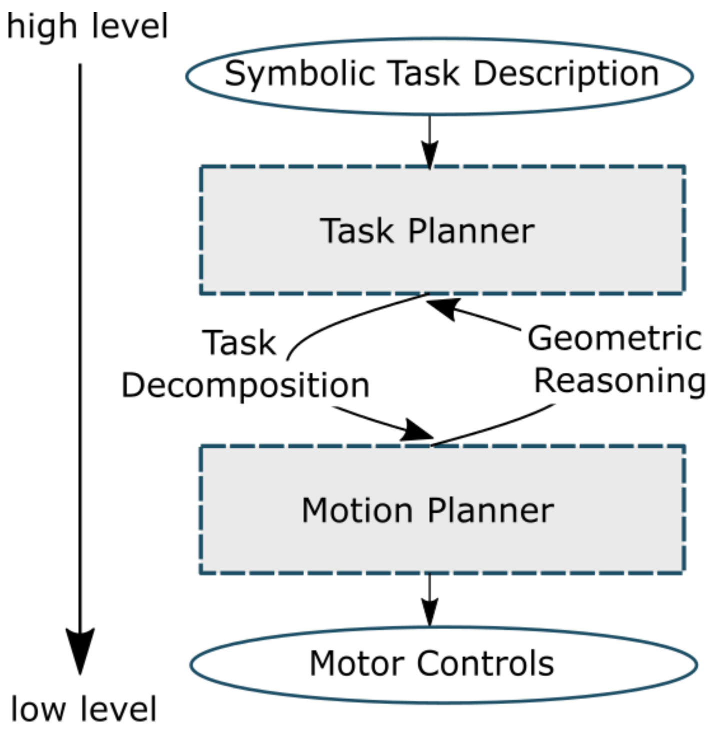
→ Task and motion planning problem

https://home.howstuffworks.com/dishwasher.htm

Why Task and Motion Planning?

Task Planner

- Efficient planning with logic descriptions for long time horizons



Motion Planner

- Considers the detailed geometric specification of the environment

➡ **Challenge:**
Transform symbolic actions into feasible motion in variable scenarios

Related Work

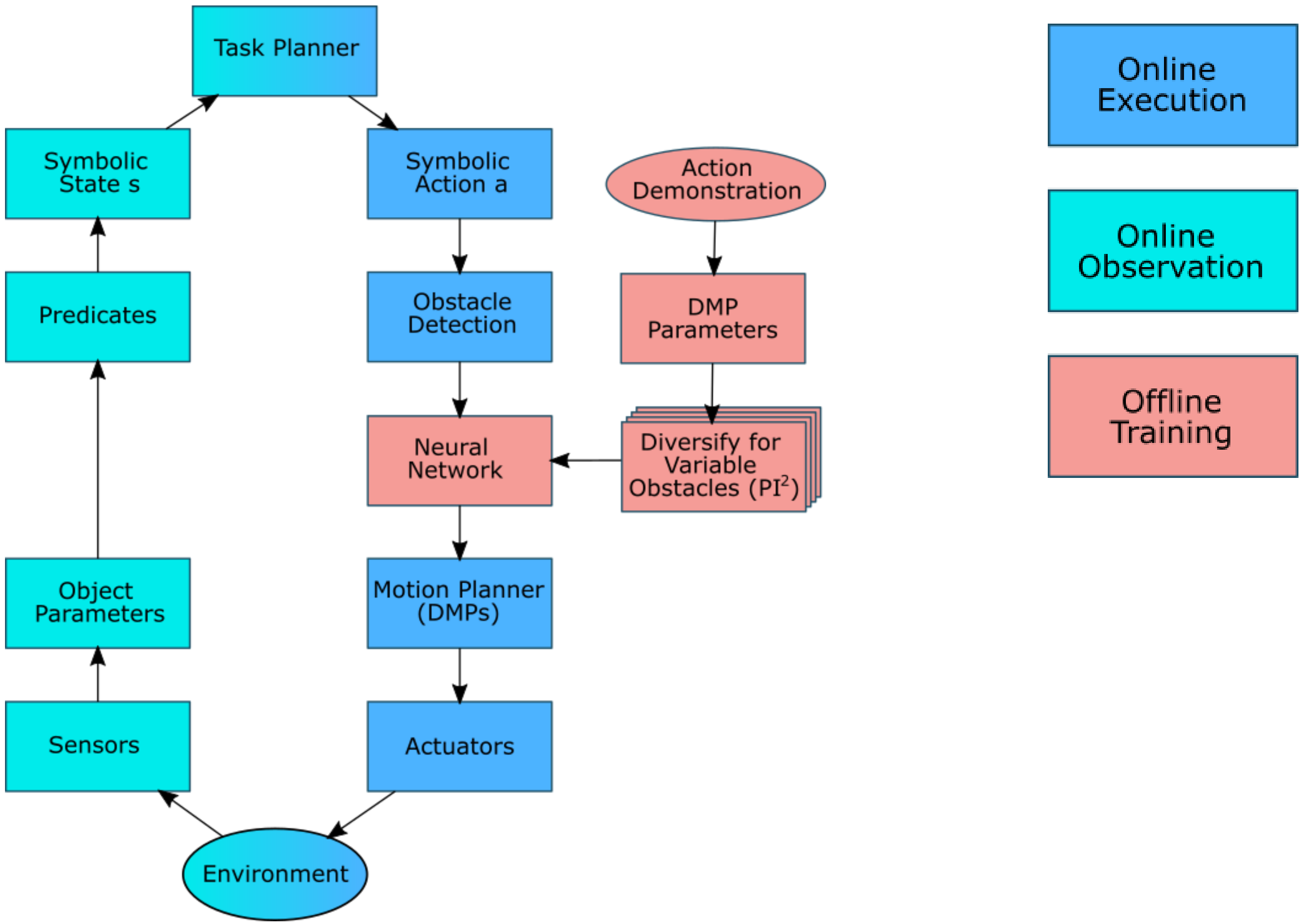
→ Improve state-of-the art methods in terms of **efficiency**

TAMP framework	**Limitation**	**Proposed solution**	**Efficiency gain**
Search-based [Dantam, 2018] [Bidot, 2017]	Deliberation before every execution	Learning-based approach	Faster at execution time
Learning-based using RL [Quack, 2015]	Only performed in low-dimensional spaces	Apply RL on LfD	Efficient motion generation with LfD
Learning-based using LfD [Agostini, 2020]	Many demonstrations required	Generalization with RL	Only one demonstration required

Contributions

- Our TAMP framework permits executing complex tasks comprising long action sequences with obstacle avoidance.
- Each symbolic action is grounded using DMPs that an action policy provides for variable object configurations.
- PI^2 efficiently generates divers sets of optimal collision-free trajectories serving as training samples to learn the action policy encoded in a NN.

The Proposed TAMP Framework



Training Methods

Why DMPs?

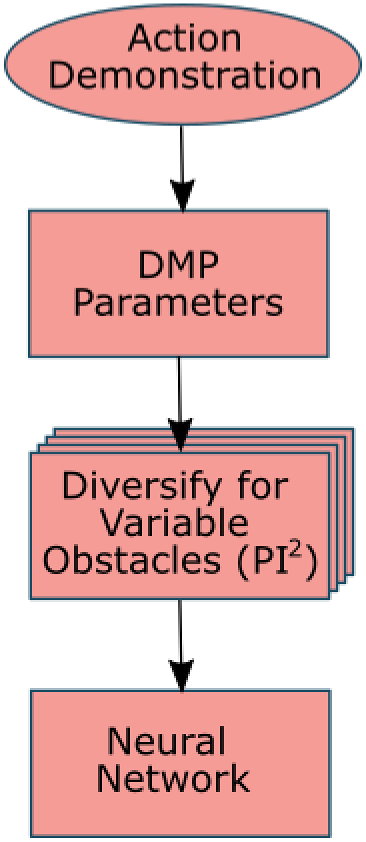
- Efficient motion encoding from one demo
- Translation, dilatation, rotation invariance

Why PI²?

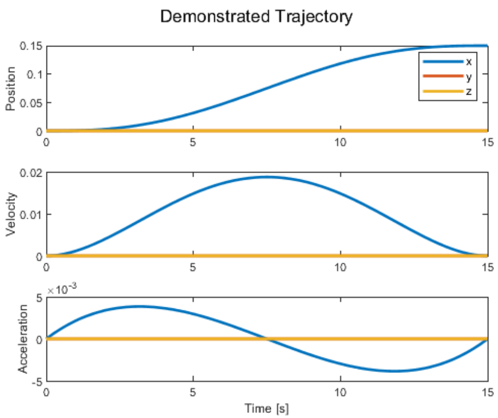
- Tunes parameterized policies like DMPs
- Numerically stable based on SOC

Why NN?

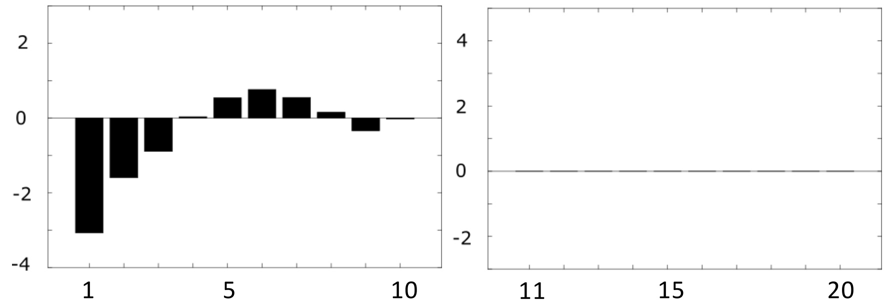
- Encode action policy for action selection



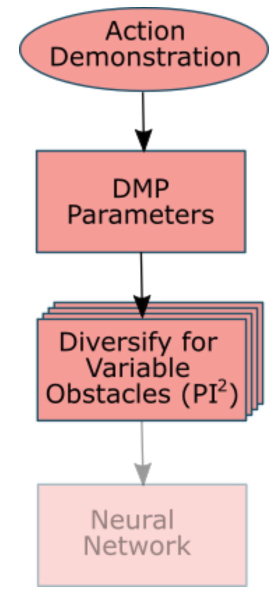
Offline Training – Encode a Demonstration



DMP parameters θ_i^j of the forcing term at $j = 1$



Basis functions i



No ability to avoid obstacles

→ Apply PI² iteratively to tune the DMP parameters θ_i

Offline Training - PI² Optimization Step

The forcing term θ_i^j of the DMPs represents the current policy

1. Generate K random samples from the current policy θ_i^j

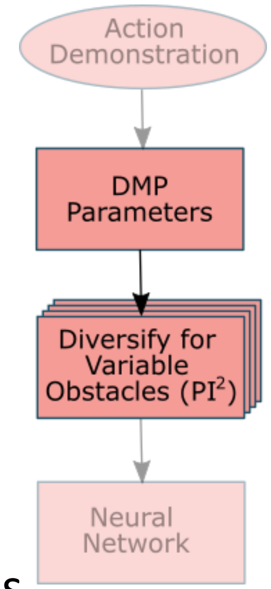
$$\theta_{i,k}^{j+1} = \theta_i^j + \epsilon_{i,k}^j$$

2. Weight each sample k using a cost function S

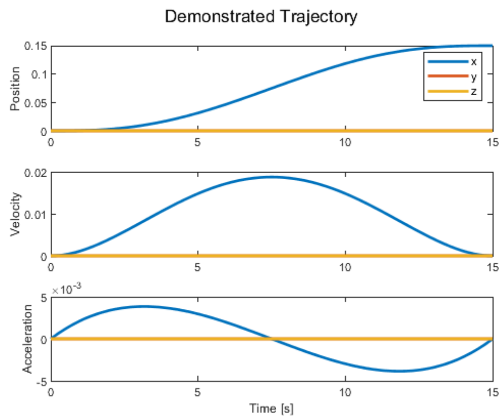
$$W_\theta\left(\theta_{i,k}^j\right) = exp\left(-\gamma \frac{S\left(\theta_{i,k}^j\right) - \min S\left(\boldsymbol{\theta}_i^j\right)}{\max S\left(\boldsymbol{\theta}_i^j\right) - \min S\left(\boldsymbol{\theta}_i^j\right)}\right)$$

3. Compute the new policy θ_i^{j+1} as weighted average of the samples

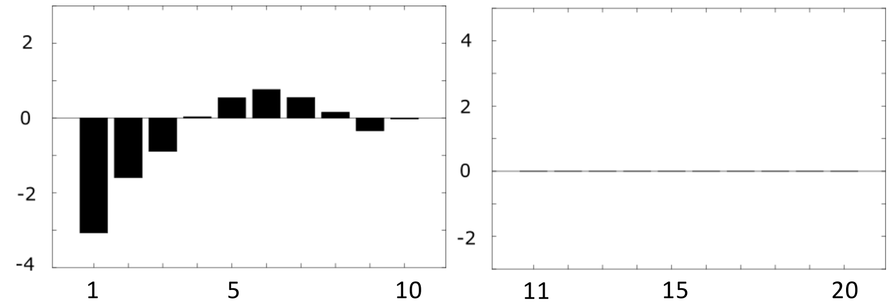
$$\theta_i^{j+1} = \frac{\sum_{k=1}^K W_\theta\left(\theta_{i,k}^j\right)\theta_{i,k}^j}{\sum_{k=1}^K W_\theta\left(\theta_{i,k}^j\right)}$$



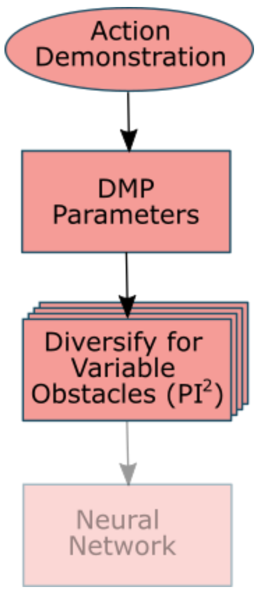
Offline Training – Diversify the Demonstration



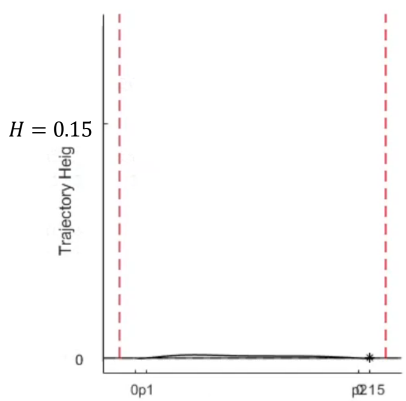
DMP parameters θ_i^j of the forcing term at $j = 1$



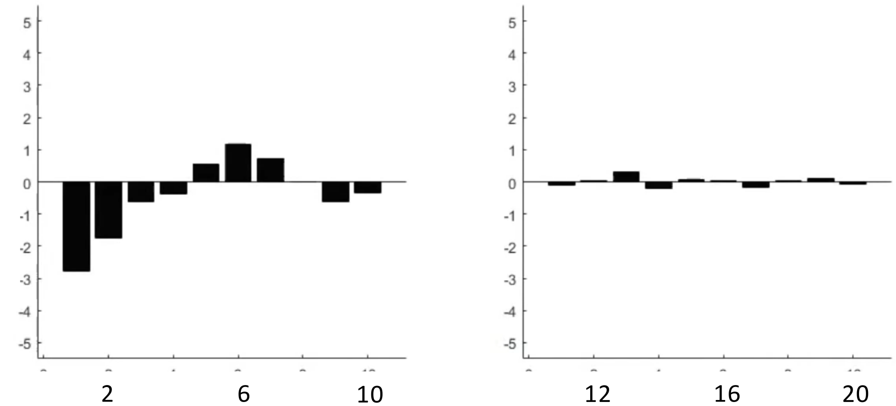
Basis functions i



$S = -\mathrm{H} = \min(h_{p1}, h_{p2})$



Tuned forcing term θ_i^j for $j = \{50, 100, \ldots, J\}$



Basis functions i

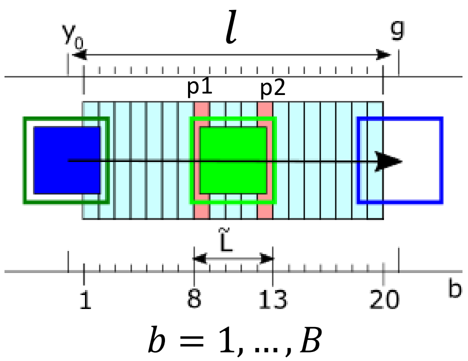
4x

Defintion of the Trajectory Shape

Two variables specify the trajectory shape r_c, r_L

Degree of curvature $r_c = H/l$

Trajectory steepness $r_L = (|\tilde{L}(2) - \tilde{L}(1)| + 1)/B$



→ r_c, r_L are dilatation invariant

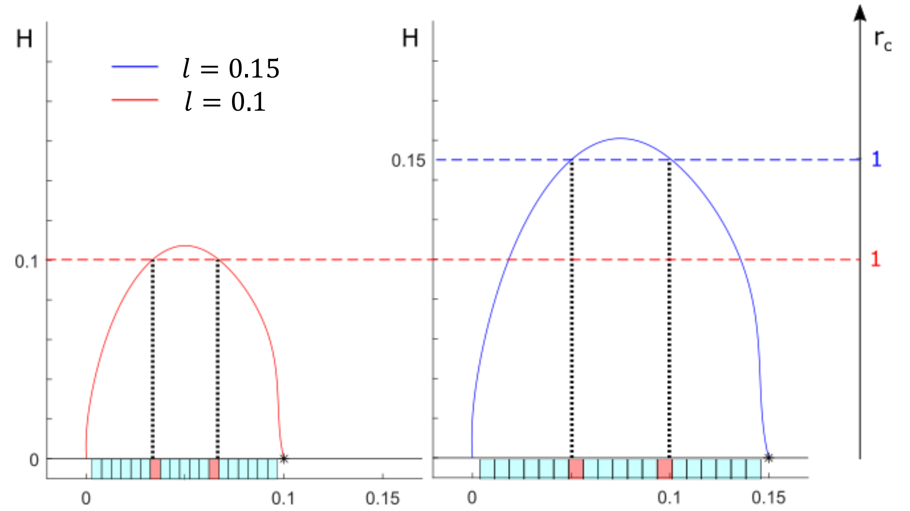
Same set of forcing term parameters θ_i

↕

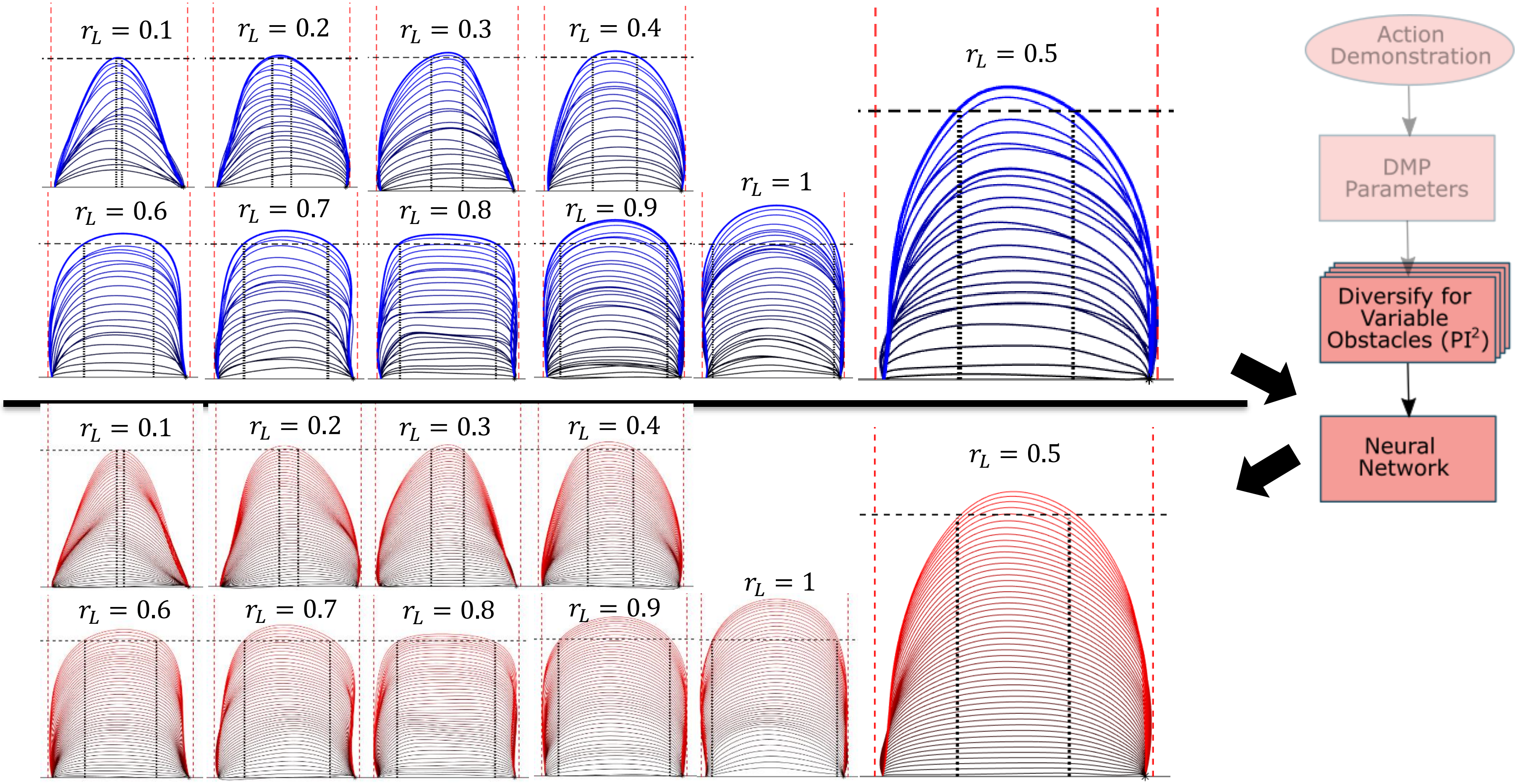
Same trajectory shape

↕

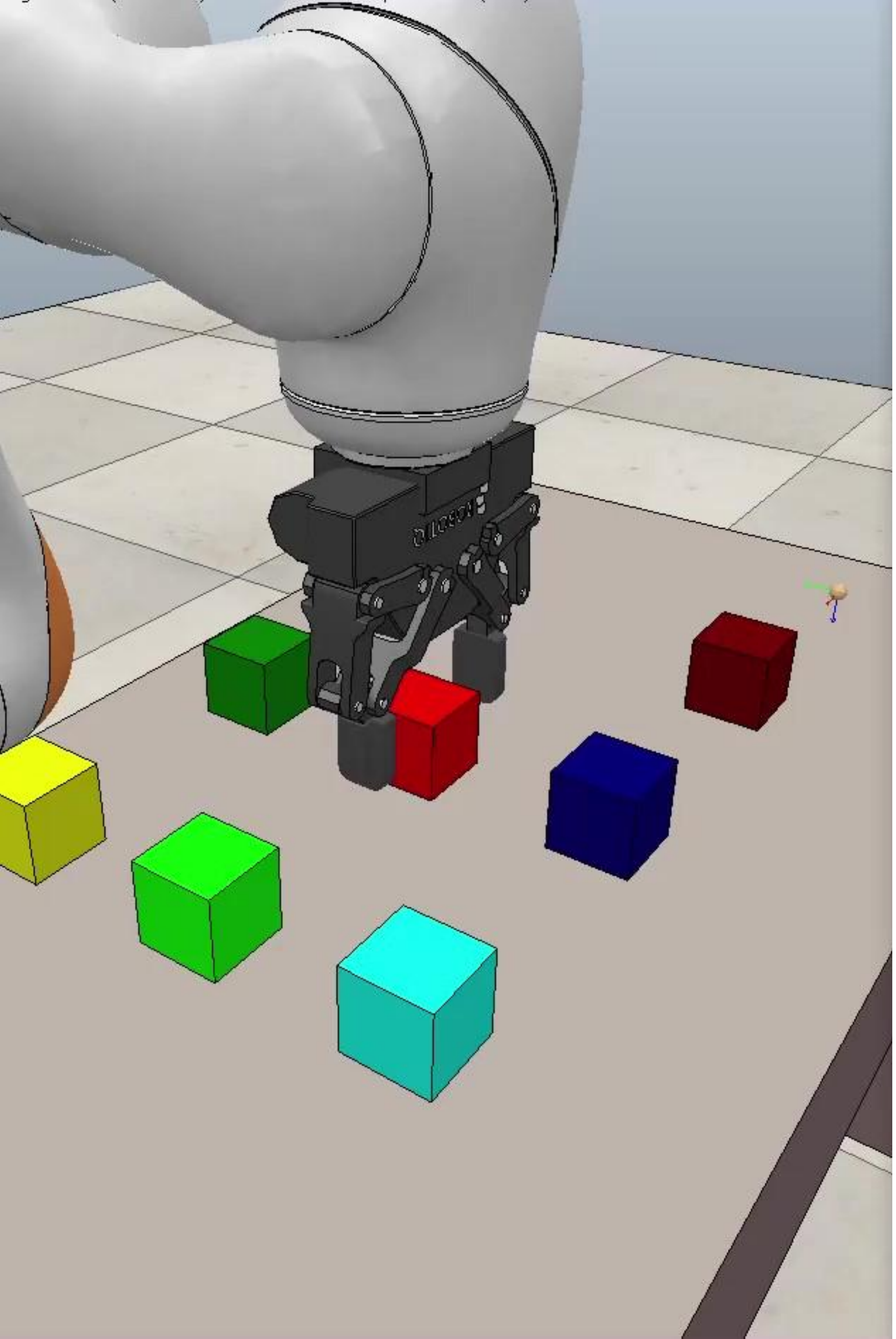
Same pair of $r_c = 1, r_L = 0.4$

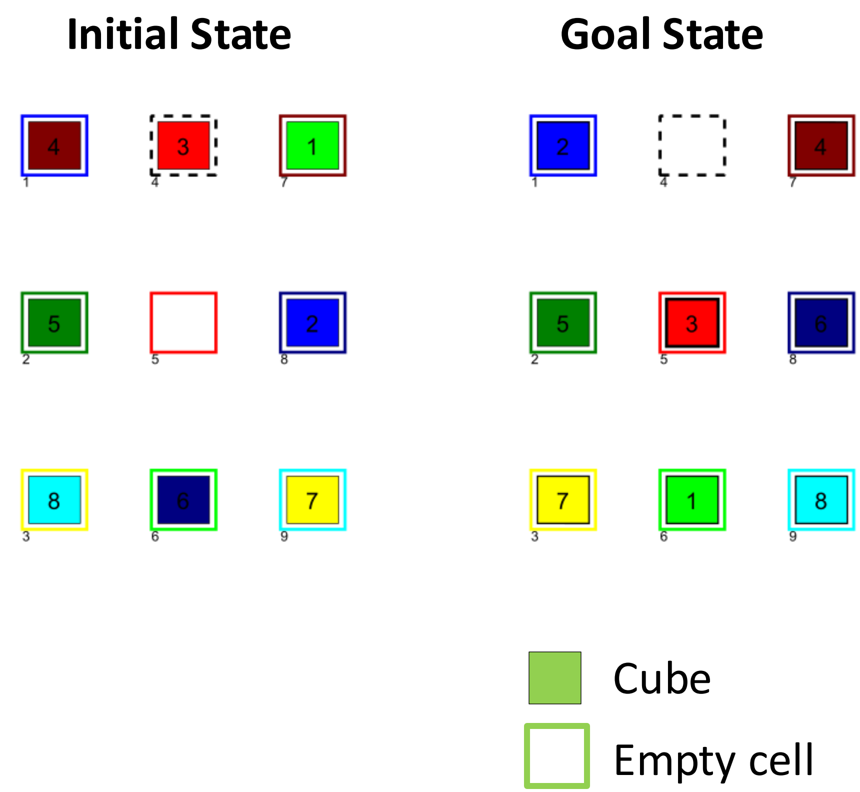


Offline Training – Learning Action Selection



→ Learned optimal parameters to ground symbolic actions in varying scenarios





Initial State
4 3 1
5 2
8 6 7
Goal State
2 4
5 3 6
7 1 8
Cube
Empty cell

Task Planner

Decompose the task into a sequence of *pickplace* actions

The *domain*

constants
`air`

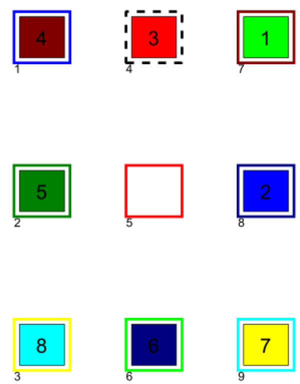
predicates
`(on ?cell ?cube)`

symbolic action
`(pickplace ?from ?to ?c)`

precondition
```
(and (on ?from ?c)
(on ?to air))
```

effect
```
(and (on ?from air)
(on ?to ?c)
(not (on ?from c))
(not (on ?to air))
```

Initial State



Task Plan

```
#1: pickplace cell1 cell5 cube4
#2: pickplace cell3 cell1 cube8
#3: pickplace cell9 cell3 cube7
#4: pickplace cell1 cell9 cube8
#5: pickplace cell8 cell1 cube2
#6: pickplace cell6 cell8 cube6
#7: pickplace cell7 cell6 cube1
#8: pickplace cell5 cell7 cube4
#9: pickplace cell4 cell5 cube3
```

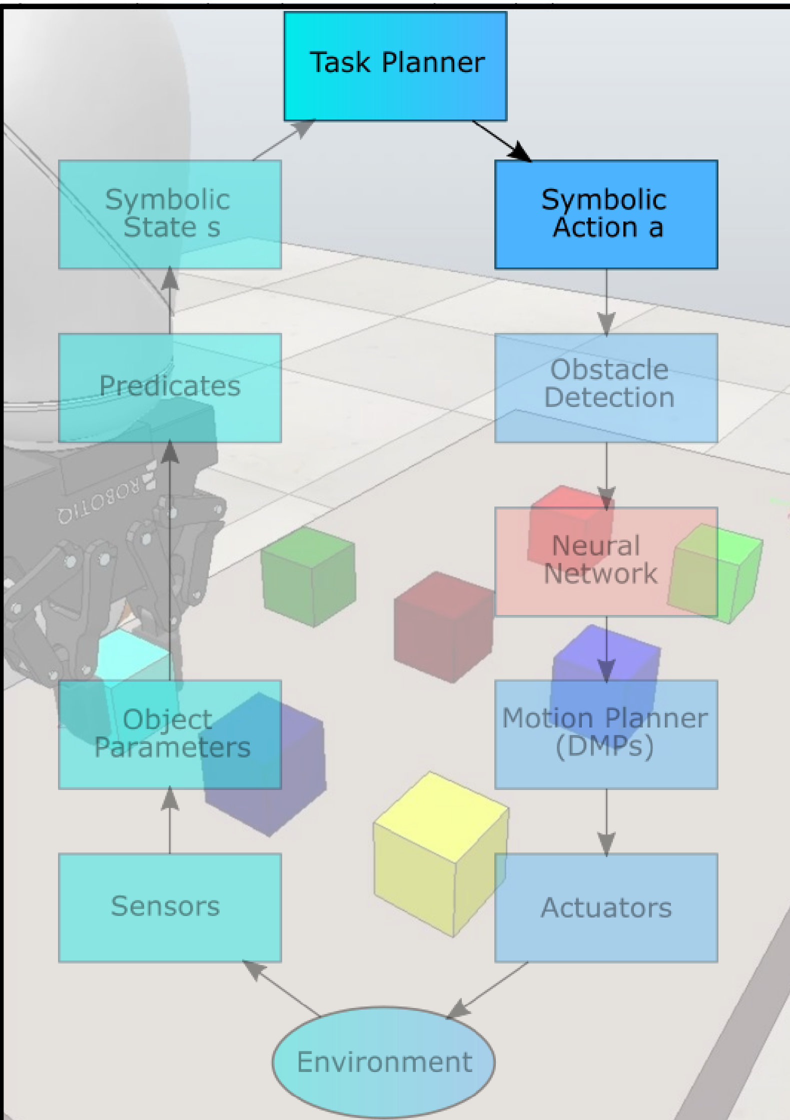
The *task*

objects
```
cube1 cube2 …
cube8
cell1 cell2 …
cell9
```

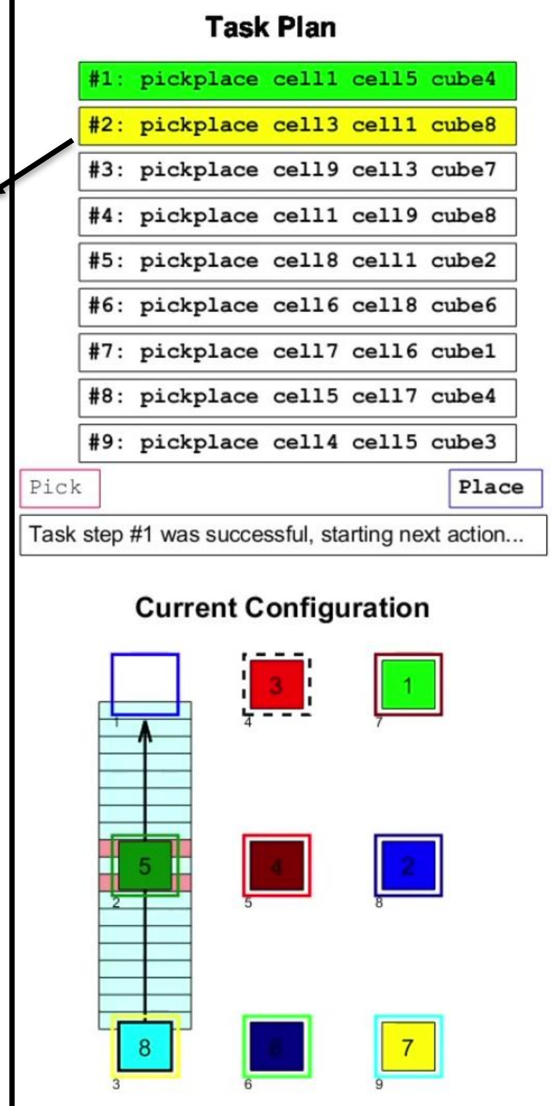
init
```
(on cell5 air)
(on cell1 cube4)
…
```

goal
```
(on cell4 air)
(on cell2 cube5)
…
```

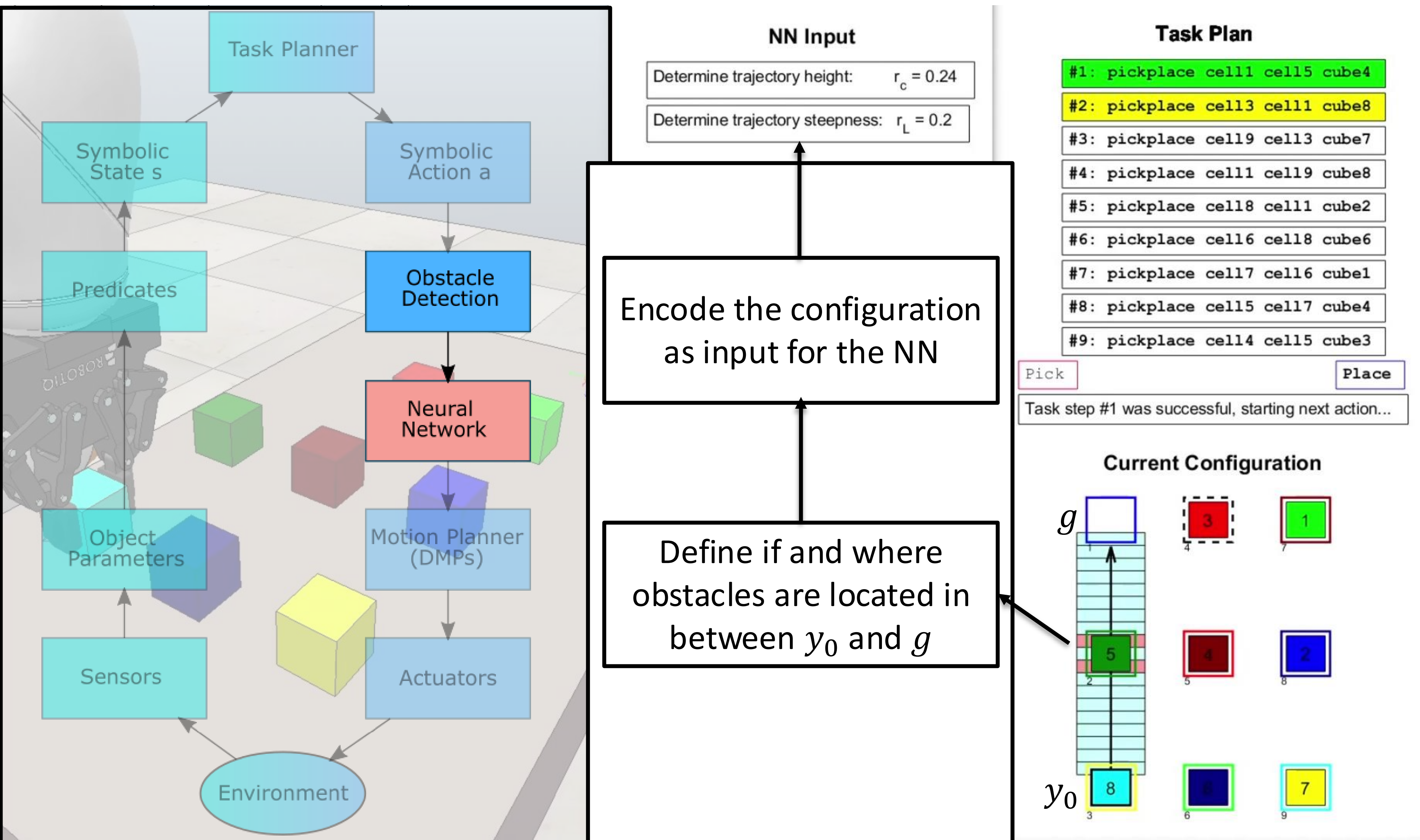
Online Loop – Select Symbolic Action



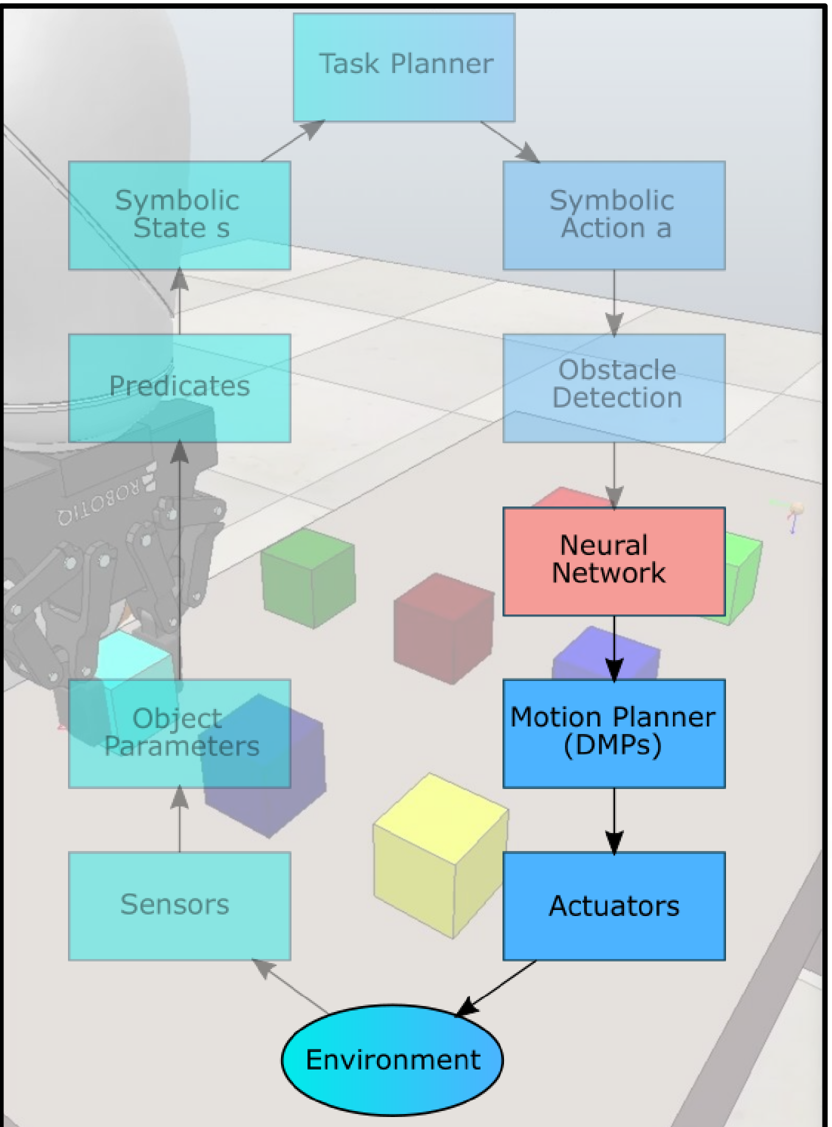
Current symbolic action:
Pick and place *cube8* from *cell3* to *cell1*

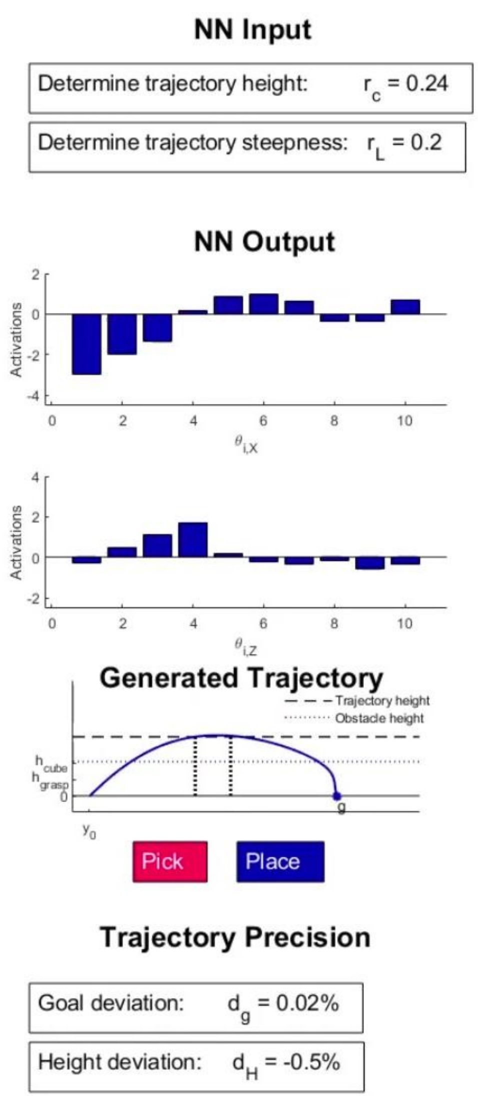


Online Loop – Encode Current Configuration



Online Loop – Execute Action Policy

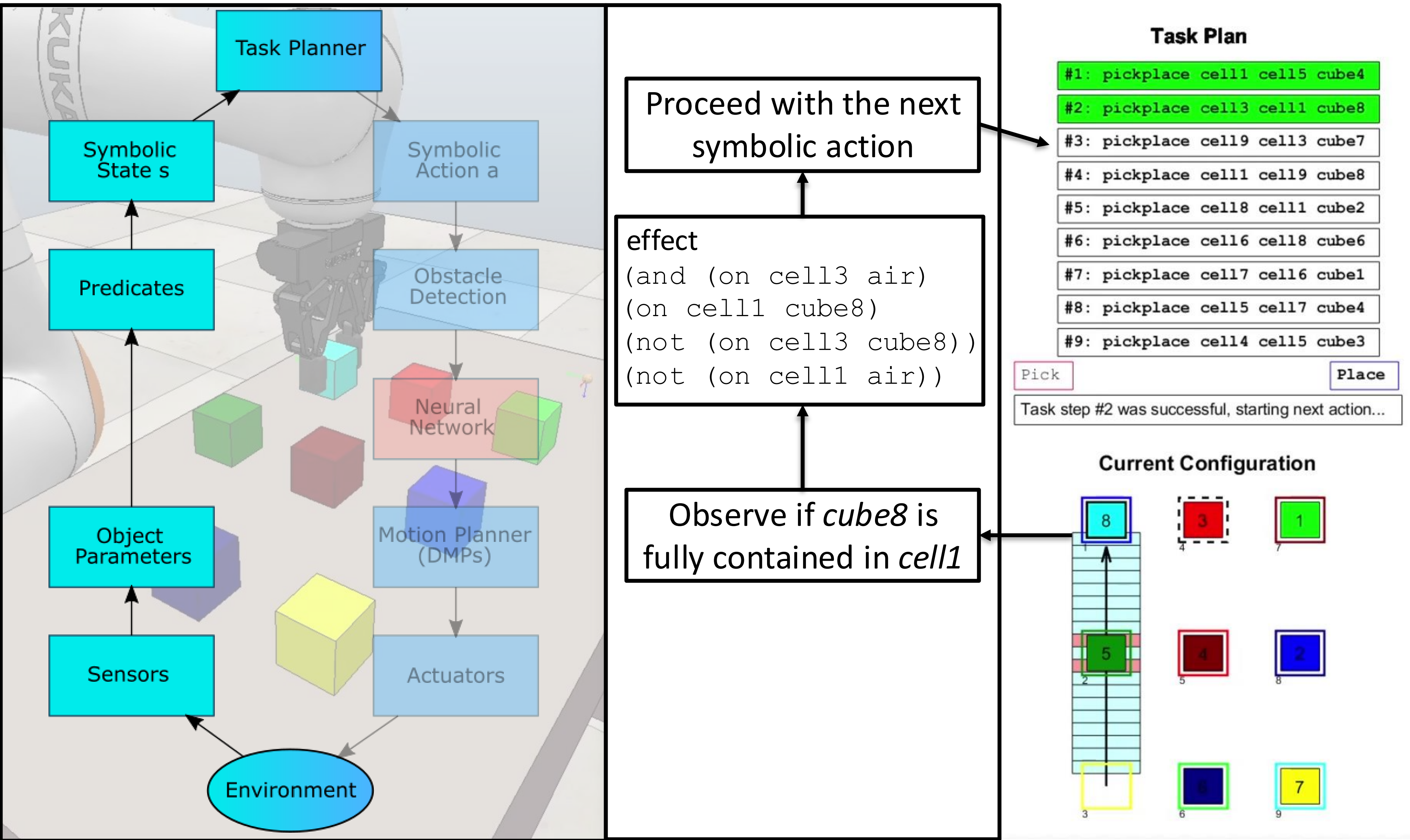




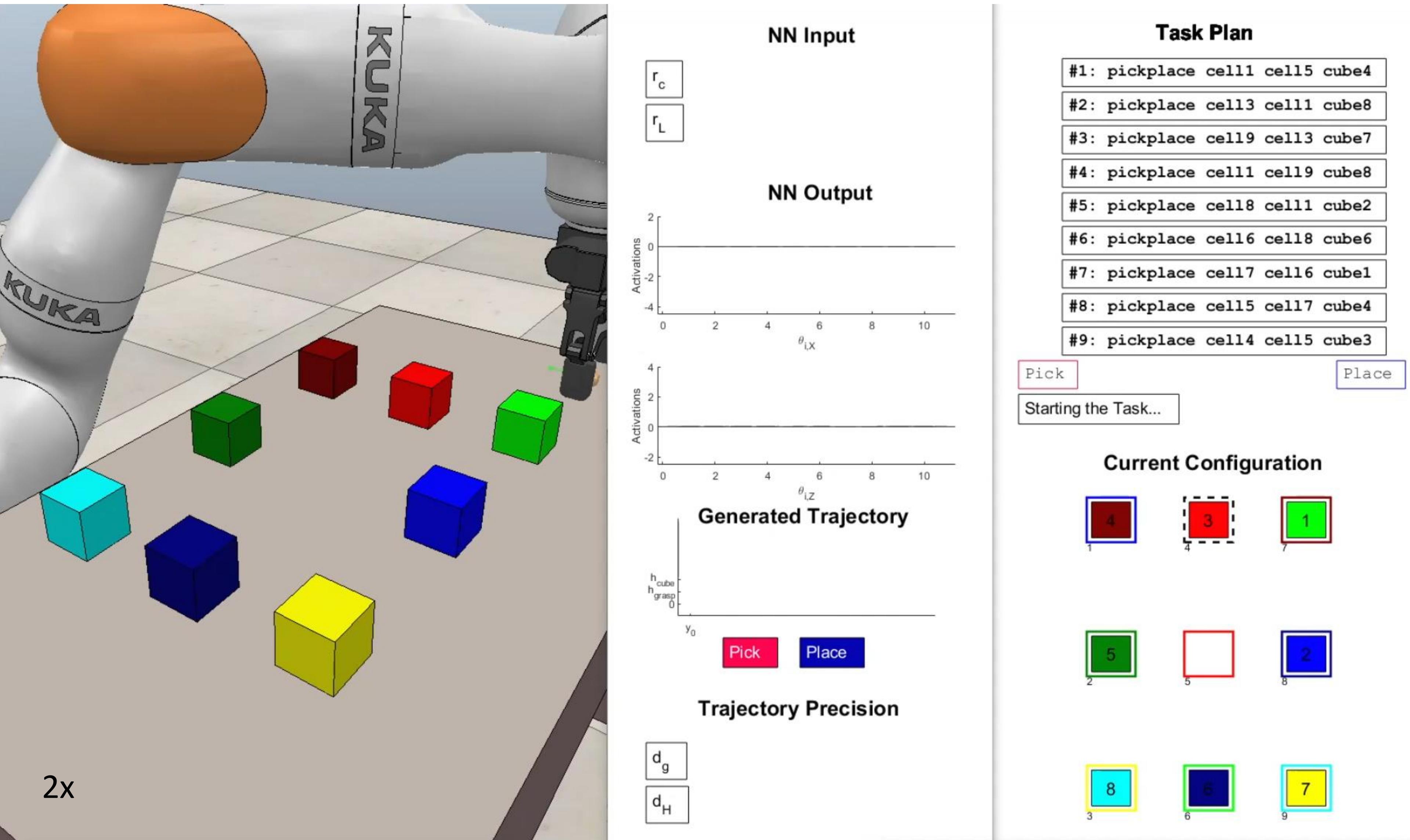
Retrieve and set the forcing term parameters of the DMPs

Generate the trajectory and send it to the robot arm

Online Loop – Observe Changes to the Environment



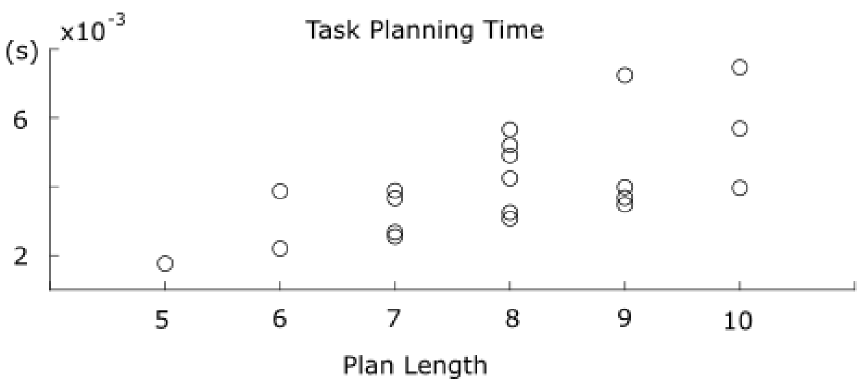
Complete Execution of a Task Example

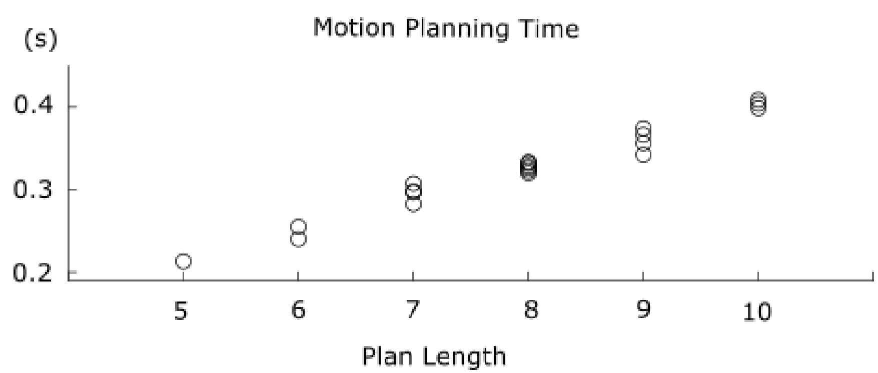


Experimental Evaluation

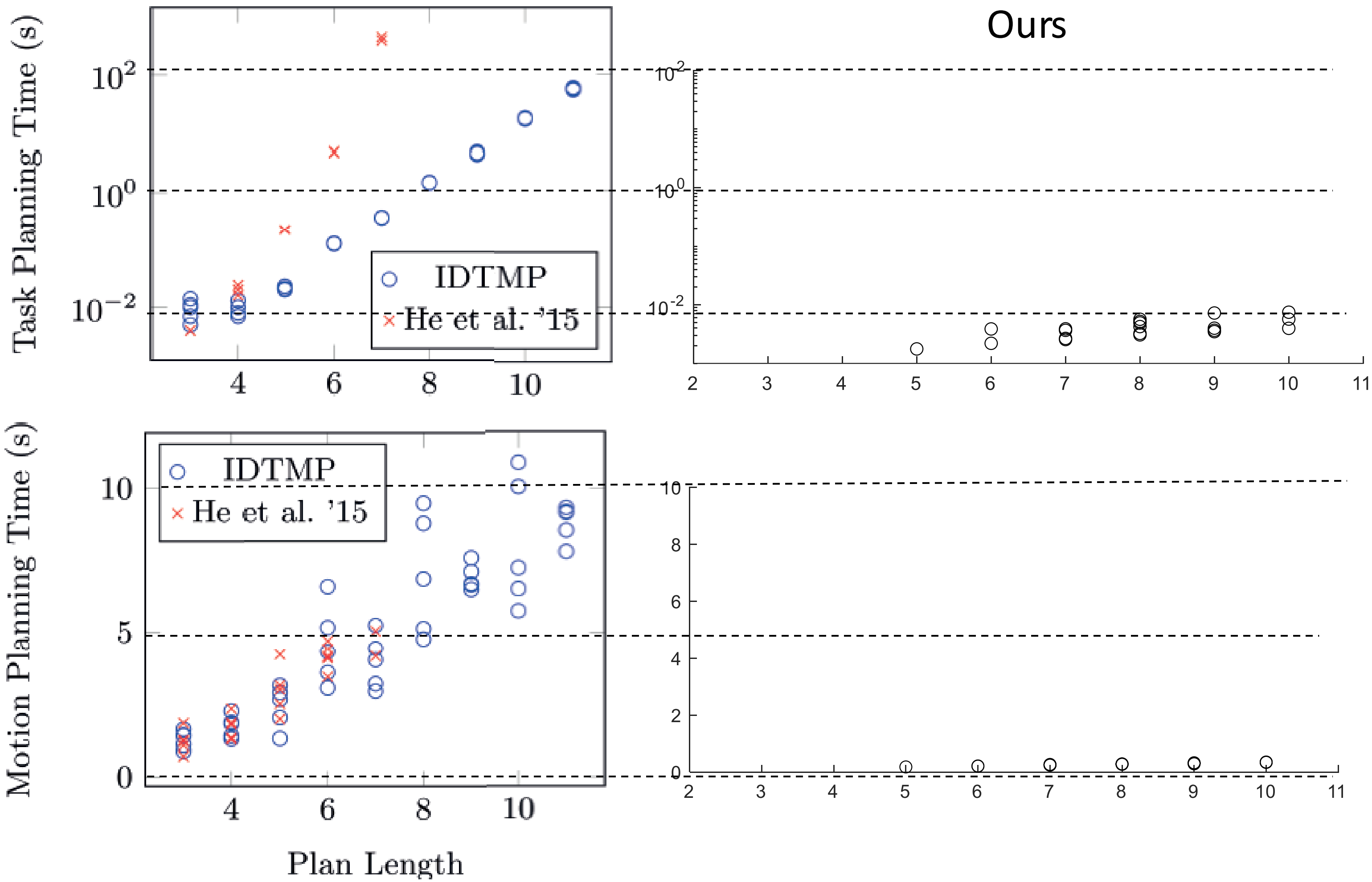
20 consecutive task executions with random initial and goal state
→ All 159 performed *pickplace* actions successful

Training Times	**9 min**
Generating the demonstration	0.02 s
PI^2 optimizations	241 s
Training of the NN	279 s





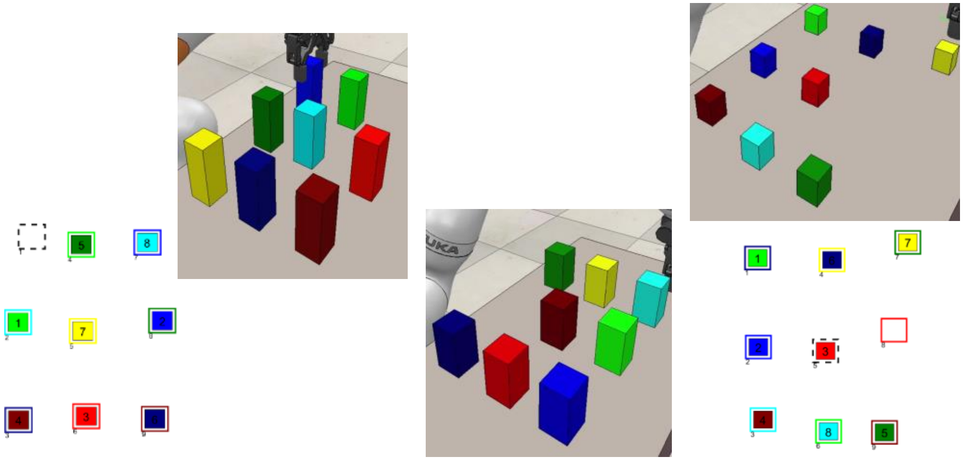
Evaluation – Compared to [Dantam, 2018]



Generalization Ability

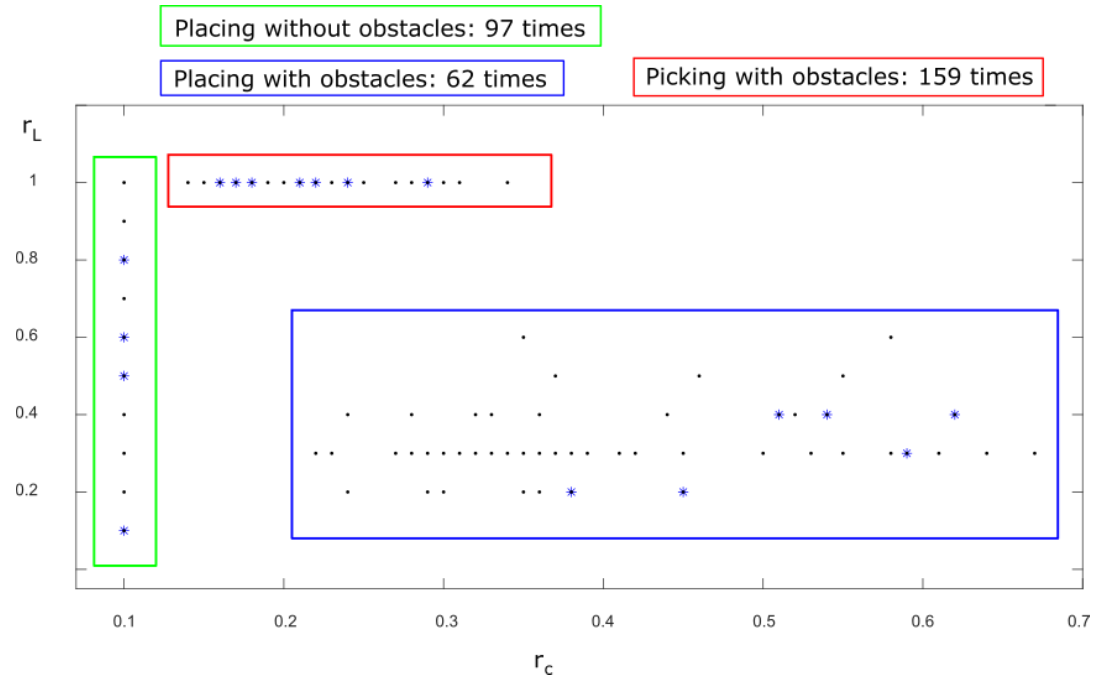
Varying block dimensions

Varying cell positions

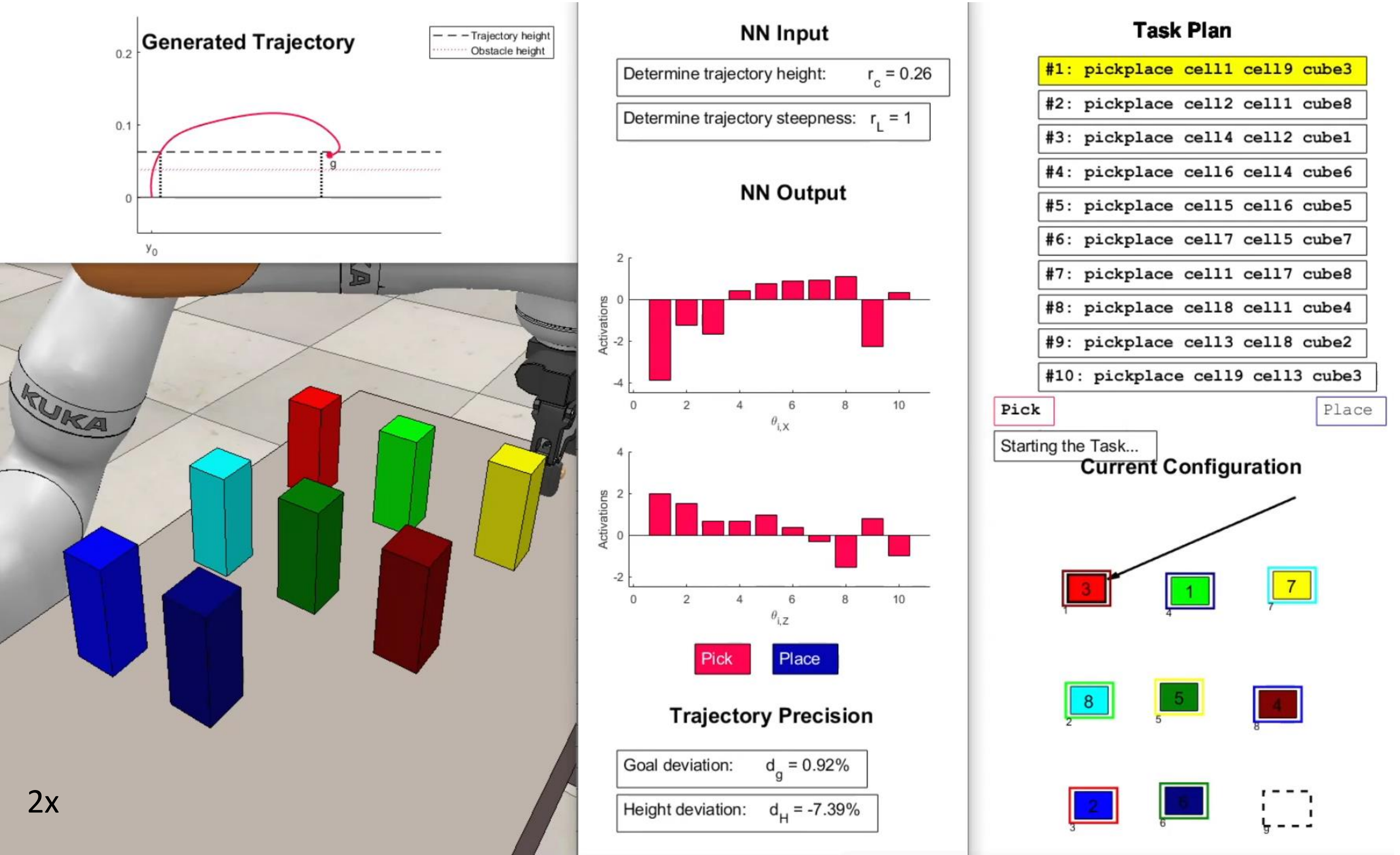


→ Learned action policy finds collision-free trajectories for all scenarios

→ More trajectory shapes required



Complete Execution of a Task Example



Limitation & Future Work

Limitations

1. Restricted to intial and goal positions in the same plane
2. Constant gripper orientation
3. Obstacle avoidance in one dimension only
4. Heuristics required for obstacle definition and optimization costs

Future Work

→ Train NN on independent data to improve generalization

→ Let RL agent learn to select appropriate input parameters and goal poses

Conclusion

- TAMP framework that utilizes LfD to efficiently generate motion and RL to generalize this motion
- PI^2 permits learning DMP parameters from a single demonstration to avoid obstacles of varying size and in varying situations
- After a few minutes of training, the action policy reliably selects collision-free trajectories to ground symbolic actions of a complex task

References

J. Bidot, L. Karlsson, F. Lagriffoul, A. Saffiotti
Geometric backtracking for combined task and motion planning in robotic systems.
In: *Artificial Intelligence*, 2017, pp. 229–265.

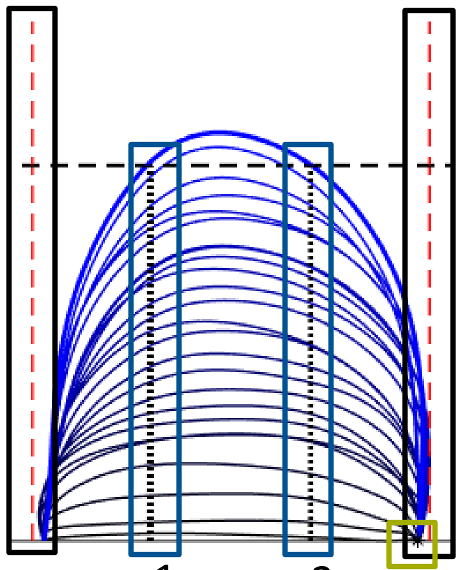
N. T. Dantam, Z. K. Kingston, S. Chaudhuri, L. E. Kavraki
An incremental constraint-based framework for task and motion planning.
In: The International Journal of Robotics Research*, 2018, pp. 1134–1151*

B. Quack, F. Wörgötter, A. Agostini
Simultaneous learning at different levels of abstraction.
In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* , 2015, pp. 4600–4607.

A. Agostini, M. Saveriano, D. Lee, J. Piater
Manipulation planning using object-centered predicates and hierarchical decomposition of contextual actions.
In: *IEEE Robotics and Automation Letters*, 2020, pp. 5629–5636.

Appendix

Cost Function

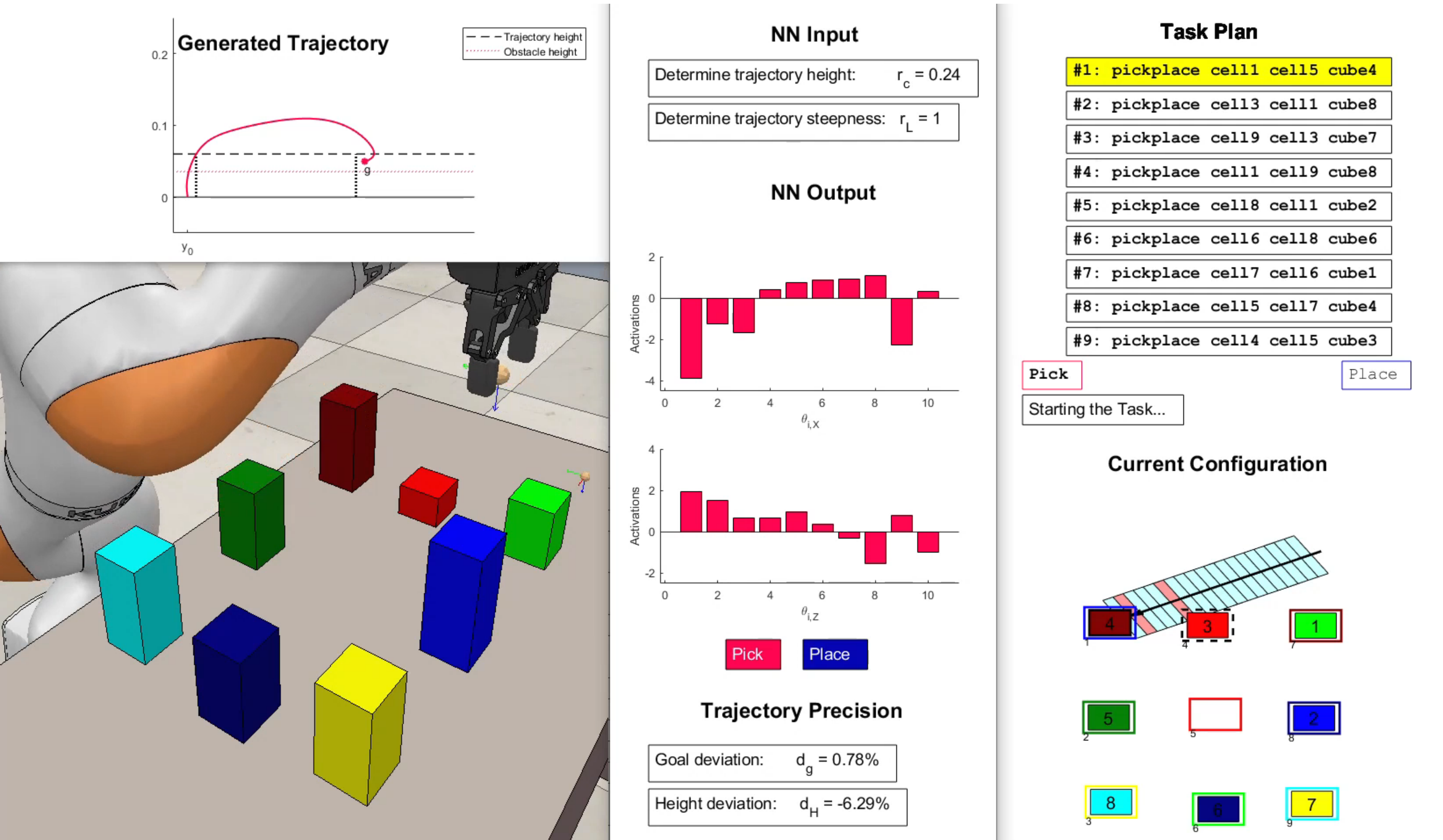


$$S = -H + c_1 \cdot S_{prec} + c_2 \cdot S_{scope}$$

$$\boxed{H = \min(h_{p1}, h_{p2})}$$

$$\boxed{S_{prec} = \|g - y_{end}\|}$$

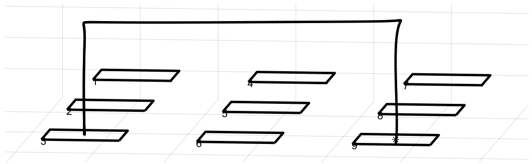
$$\boxed{S_{scope} = -\sum_{t=1}^{T} \min(0, m + y_{X,t} - y_{0,X}) - \sum_{t=1}^{T} \min(0, m + g_X - y_{X,t})}$$

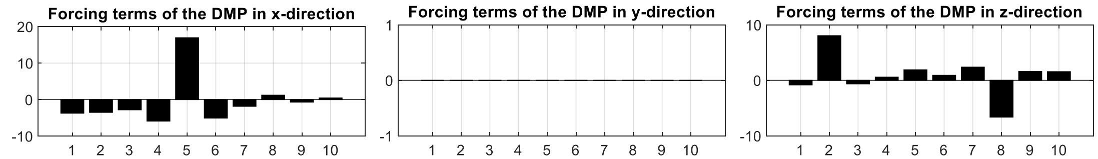


Generated Trajectory
Trajectory height
Obstacle height
0.2
0.1
0
g
y0
NN Input
Determine trajectory height: r_c = 0.24
Determine trajectory steepness: r_L = 1
NN Output
Activations
θi,x
θi,z
Pick
Place
Trajectory Precision
Goal deviation: d_g = 0.78%
Height deviation: d_H = -6.29%
Task Plan
#1: pickplace cell1 cell5 cube4
#2: pickplace cell3 cell1 cube8
#3: pickplace cell9 cell3 cube7
#4: pickplace cell1 cell9 cube8
#5: pickplace cell8 cell1 cube2
#6: pickplace cell6 cell8 cube6
#7: pickplace cell7 cell6 cube1
#8: pickplace cell5 cell7 cube4
#9: pickplace cell4 cell5 cube3
Pick
Place
Starting the Task...
Current Configuration

DMP: Roto-Dilatation Invariance [Ginesi, 2019]

One demonstration encoded in three DMPs



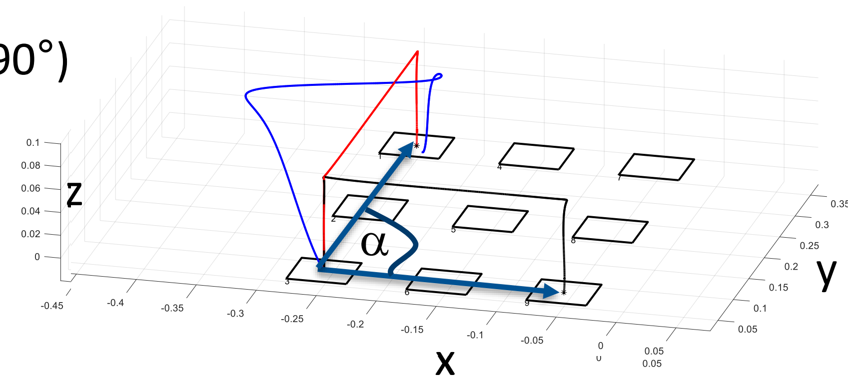


New goal position (rotated around z: α=90°)

Same forcing term parameters

→ Unexpected trajectory shape

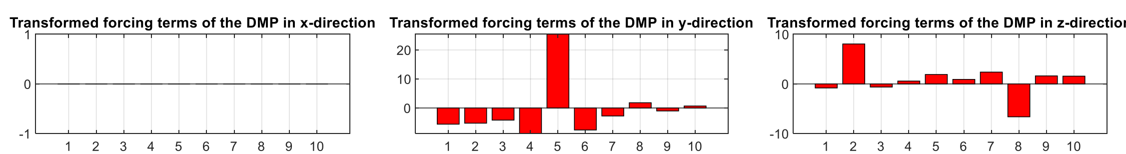
→ Goal position is not reached precisely



Rotate the forcing term parameters

$$f_x^{new} = \cos\alpha * f_x^{demo}$$

$$f_y^{new} = \sin\alpha * f_y^{demo}$$



$\tilde{L} = (7,14)$ $\tilde{L} = (7,14)$