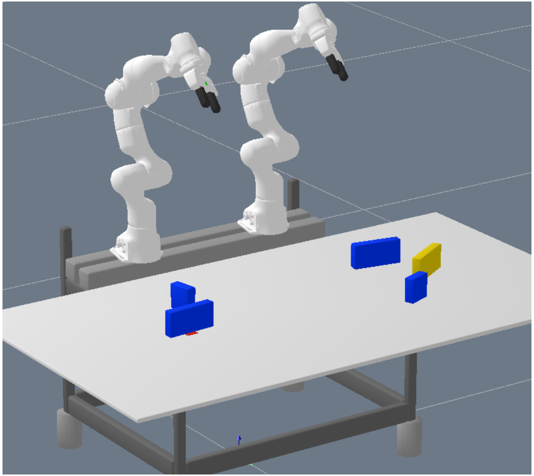
Deep Visual Reasoning: Learning to Predict Action Sequences for Task and Motion Planning from an Initial Scene Image

Danny Driess Jung-Su Ha Marc Toussaint

Machine Learning & Robotics Lab – University of Stuttgart
Max-Planck Institute for Intelligent Systems – Stuttgart
Learning and Intelligent Systems – TU-Berlin

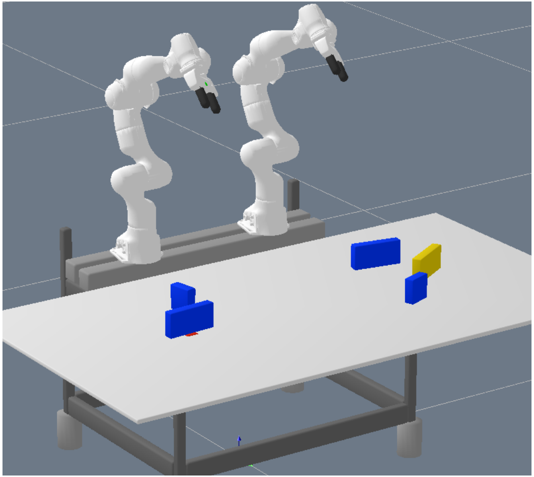
3D Scene Understanding for Vision, Graphics, and Robotics – CVPR 2020 Workshop

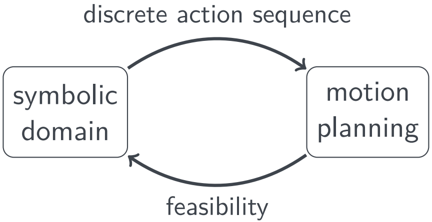
Motivation



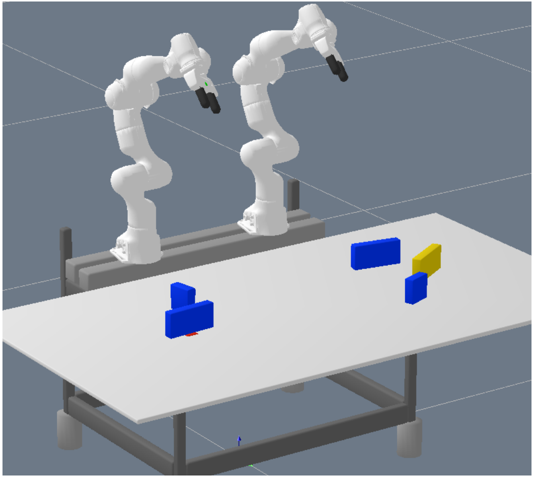
- Goal is to place yellow box on the red spot
- Target location (red spot) might be occupied
- Multiple objects

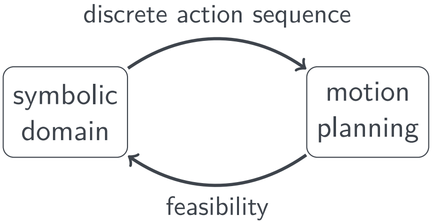
Motivation – Task and Motion Planning





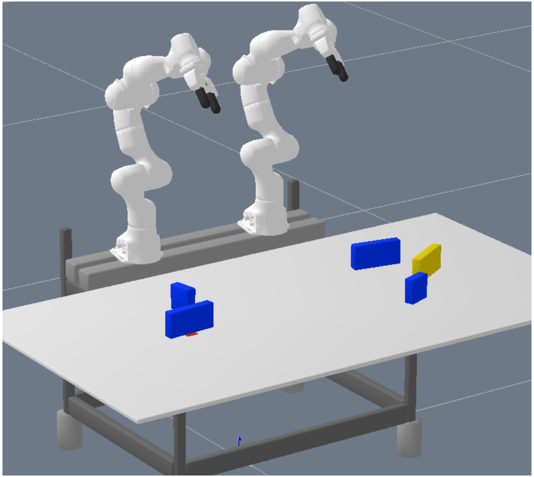
Motivation – Task and Motion Planning

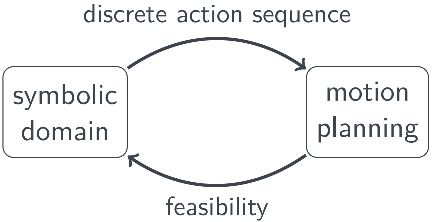




- High-level action sequence $a_{1:K} \in \mathbb{A}$ (plan skeleton)

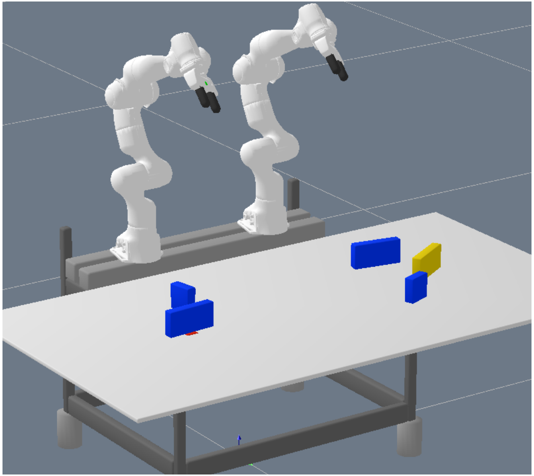
Motivation – Task and Motion Planning





- High-level action sequence $a_{1:K} \in \mathbb{A}$ (plan skeleton)
- Motion planning problem parameterized by $a_{1:K}$, e.g. nonlinear trajectory optimization (NLP)

Motivation – Combinatorial Complexity

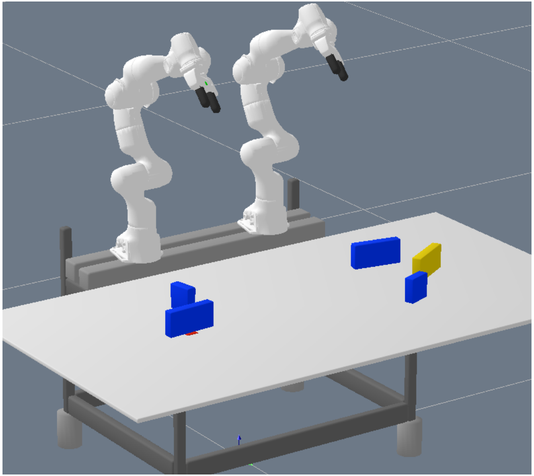


$\approx$ 500,000 discrete action sequences (up to length 6) reach the symbolic goal

- Target location (red spot) might be occupied
- Kinematic constraints (reachability, joint limits)

Majority of discrete action sequences is **infeasible**

Main Idea – Learning to Predict Action Sequences



Given scene and goal as input, predict action sequence that leads to a feasible motion plan.

Contributions

1. **Convolutional, recurrent neural network that predicts action sequences from an initial scene image**

Contributions

1. **Convolutional, recurrent neural network that predicts action sequences from an initial scene image**

2. **Integration of the network into the tree search of TAMP framework**

Contributions

1. **Convolutional, recurrent neural network that predicts action sequences from an initial scene image**
2. **Integration of the network into the tree search of TAMP framework**
3. **Generalization to scenarios with more objects than during training**

Logic Geometric Programming (LGP)

Optimization based approach to TAMP, developed by Marc Toussaint[1]

Logic Geometric Programming (LGP)

Optimization based approach to TAMP, developed by Marc Toussaint[1]

Main idea: Given a scene S

- Find path $x : [0, KT] \to \mathcal{X}$ in the configuration space $\mathcal{X} \subset \mathbb{R}^n \times SE(3)^m$ as the solution of an optimization problem

[1] Toussaint et al., Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning, R:SS 2018

Logic Geometric Programming (LGP)

Optimization based approach to TAMP, developed by Marc Toussaint[1]

Main idea: Given a scene S

- Find path $x : [0, KT] \to \mathcal{X}$ in the configuration space $\mathcal{X} \subset \mathbb{R}^n \times SE(3)^m$ as the solution of an optimization problem
- Path consists of $K \in \mathbb{N}$ phases, each of length $T > 0$

[1] Toussaint et al., Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning, R:SS 2018

Logic Geometric Programming (LGP)

Optimization based approach to TAMP, developed by Marc Toussaint[1]

Main idea: Given a scene S

- Find path $x : [0, KT] \rightarrow \mathcal{X}$ in the configuration space $\mathcal{X} \subset \mathbb{R}^n \times SE(3)^m$ as the solution of an optimization problem
- Path consists of $K \in \mathbb{N}$ phases, each of length $T > 0$
- Costs and constraints are parameterized by a symbolic state variable $s \in \mathcal{S}$

[1] Toussaint et al., Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning, R:SS 2018

Logic Geometric Programming (LGP)

Optimization based approach to TAMP, developed by Marc Toussaint[1]

Main idea: Given a scene S

- Find path $x : [0, KT] \to \mathcal{X}$ in the configuration space $\mathcal{X} \subset \mathbb{R}^n \times SE(3)^m$ as the solution of an optimization problem
- Path consists of $K \in \mathbb{N}$ phases, each of length $T > 0$
- Costs and constraints are parameterized by a symbolic state variable $s \in \mathcal{S}$
- Time-discrete transitions of s_{k-1} to s_k are subject to a first-order logic language through actions $a \in \mathbb{A}(s_{k-1}, S)$ (grounded action operators)

[1] Toussaint et al., Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning, R:SS 2018

Logic Geometric Programming (LGP)

Optimization based approach to TAMP, developed by Marc Toussaint[1]

Main idea: Given a scene S

- Find path $x : [0, KT] \to \mathcal{X}$ in the configuration space $\mathcal{X} \subset \mathbb{R}^n \times SE(3)^m$ as the solution of an optimization problem
- Path consists of $K \in \mathbb{N}$ phases, each of length $T > 0$
- Costs and constraints are parameterized by a symbolic state variable $s \in \mathcal{S}$
- Time-discrete transitions of s_{k-1} to s_k are subject to a first-order logic language through actions $a \in \mathbb{A}(s_{k-1}, S)$ (grounded action operators)
- Symbolic goal $g \in \mathbb{G}$

[1] Toussaint et al., Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning, R:SS 2018

Logic Geometric Programming (LGP)

$$
\begin{aligned}
P(g,S) = \min_{\substack{K\in\mathbb{N} \\ x:[0,KT]\to\mathcal{X} \\ a_{1:K},\ s_{1:K}}} & \int_0^{KT} c\big(x(t),\dot{x}(t),\ddot{x}(t),s_{k(t)},S\big)\,\mathrm{d}t && \text{(1a)}\\
\text{s.t.} & \\
\forall_{t\in[0,KT]}: \quad & h_{\mathrm{eq}}\big(x(t),\dot{x}(t),s_{k(t)},S\big) = 0 && \text{(1b)}\\
\forall_{t\in[0,KT]}: \quad & h_{\mathrm{ineq}}\big(x(t),\dot{x}(t),s_{k(t)},S\big) \le 0 && \text{(1c)}\\
\forall_{k=1,\dots,K}: \quad & h_{\mathrm{sw}}\big(x(kT),\dot{x}(kT),a_k,S\big) = 0 && \text{(1d)}\\
\forall_{k=1,\dots,K}: \quad & a_k \in \mathbb{A}(s_{k-1},S) && \text{(1e)}\\
\forall_{k=1,\dots,K}: \quad & s_k = \mathrm{succ}(s_{k-1},a_k) && \text{(1f)}\\
& x(0) = \tilde{x}_0(S) && \text{(1g)}\\
& s_0 = \tilde{s}_0(S) && \text{(1h)}\\
& s_K \in \mathcal{S}_{\mathrm{goal}}(g) && \text{(1i)}
\end{aligned}
$$

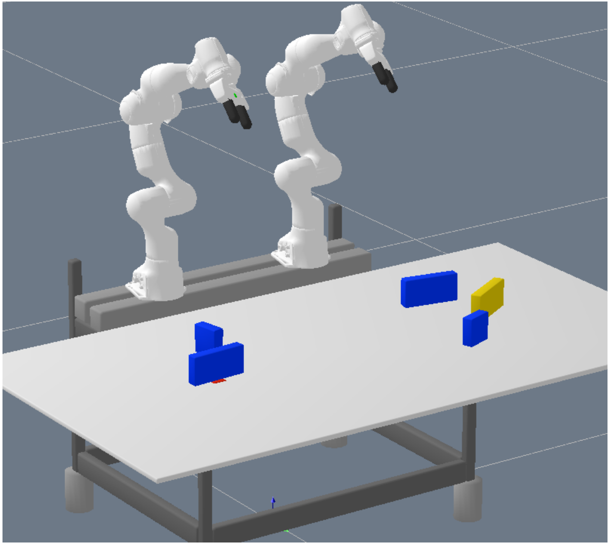
Multi-Bound Tree Search

- Logic defines a decision tree through $a_k \in \mathbb{A}(s_{k-1})$ and $s_k = \mathrm{succ}(s_{k-1}, a_k)$
- Each node corresponds to a nonlinear program (NLP)
- Leaf nodes are candidates for a feasible solution ($s_K \in \mathcal{S}_{\mathrm{goal}}(g)$)

Multi-Bound Tree Search

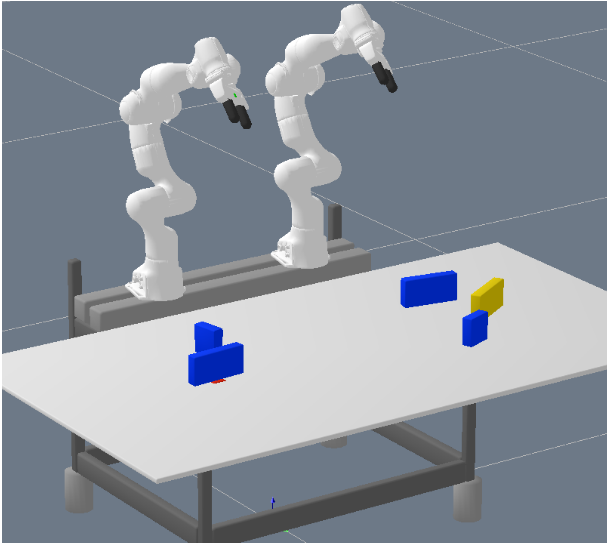
- Logic defines a decision tree through $a_k \in \mathbb{A}(s_{k-1})$ and $s_k = \text{succ}(s_{k-1}, a_k)$
- Each node corresponds to a nonlinear program (NLP)
- Leaf nodes are candidates for a feasible solution ($s_K \in \mathcal{S}_{\text{goal}}(g)$)
- Lower bounds on the full path problem to guide tree search

Multi-Bound Tree Search – Problems



- Approx. 500,000 leaf nodes up to depth 6

Multi-Bound Tree Search – Problems



- Approx. 500,000 leaf nodes up to depth 6
- However, many actions in early phases of the sequence are feasible. Therefore, lower bounds do not help much

Deep Visual Reasoning

Given the scene and goal as input, learn to predict promising action sequences such that (ideally) only one optimization problem would have to be solved.

Deep Visual Reasoning

Given the scene and goal as input, learn to predict promising action sequences such that (ideally) only one optimization problem would have to be solved.

- How to encode scene and goal as input to a learning algorithm?
- How to generalize to changing numbers of objects in the scene?
- How to deal with prediction errors?

Deep Visual Reasoning – Predicting Action Sequences

Set of candidate action sequences

$$\mathcal{T}(g, S) = \{ a_{1:K} : \forall_{i=1}^{K} \; a_i \in \mathbb{A}(s_{i-1}, S), \; s_i = \mathrm{succ}(s_{i-1}, a_i), \; s_0 = \tilde{s}_0(S), \; s_K \in \mathcal{S}_{\mathrm{goal}}(g) \}$$

Deep Visual Reasoning – Predicting Action Sequences

Set of candidate action sequences

$$\mathcal{T}(g, S) = \{ a_{1:K} : \ \forall_{i=1}^{K} \ a_i \in \mathbb{A}(s_{i-1}, S), \ s_i = \mathrm{succ}(s_{i-1}, a_i), \ s_0 = \tilde{s}_0(S), \ s_K \in \mathcal{S}_{\mathrm{goal}}(g) \}$$

Feasibility of an action sequence $a_{1:K} = (a_1, \ldots, a_K)$

$$F_S(a_{1:K}) = \begin{cases} 1 & \exists x : [0, KT] \to \mathcal{X} : (1b) - (1h) \\ 0 & \text{else} \end{cases}$$

Deep Visual Reasoning – Predicting Action Sequences

Set of candidate action sequences

$$\mathcal{T}(g, S) = \{ a_{1:K} : \forall_{i=1}^{K} \; a_i \in \mathbb{A}(s_{i-1}, S), \; s_i = \mathrm{succ}(s_{i-1}, a_i), \; s_0 = \tilde{s}_0(S), \; s_K \in \mathcal{S}_{\mathrm{goal}}(g) \}$$

Feasibility of an action sequence $a_{1:K} = (a_1, \ldots, a_K)$

$$F_S(a_{1:K}) = \begin{cases} 1 & \exists x : [0, KT] \to \mathcal{X} : (1b) - (1h) \\ 0 & \text{else} \end{cases}$$

First idea: Learn $\hat{F}_S(a_{1:K})$ and then

$$\underset{a_{1:K} \in \mathcal{T}(g,S)}{\operatorname{argmax}} \; \hat{F}_S(a_{1:K})$$

Deep Visual Reasoning – Predicting Action Sequences

$$\pi\left(a_k,\ g,\ a_1,\dots,a_{k-1},\ S\right) = p\left(\exists_{K\geq k}\exists_{a_{k+1},\dots,a_K}\ :\ a_{1:K}\in\mathcal{T}\left(g,S\right),\ F_S\left(a_{1:K}\right)=1\ \middle|\ a_k,\ g,\ a_1,\dots,a_{k-1},\ S\right)$$

Deep Visual Reasoning – Training Targets

Sample scenes S^i, goals g^i and goal-reaching action sequences $a^i_{1:K^i} \in \mathcal{T}\left(g^i, S^i\right)$, e.g. with breadth-first search

$$\mathcal{D}_{\text{data}} = \left\{ \left(S^i, a^i_{1:K^i}, g^i, F_{S^i}\left(a^i_{1:K^i}\right) \right) \right\}_{i=1}^{n}$$

Training dataset for π

$$\mathcal{D}_{\text{train}} = \left\{ \left(S^i, a^i_{1:K^i}, g^i, f^i \right) \right\}_{i=1}^{n}$$

where $f^i \in \{0,1\}^{K_i}$ is a sequence of binary labels with components

$$f^i_j = \begin{cases} 1 & F_{S^i}\left(a^i_{1:K^i}\right) = 1 \\ 1 & \exists \left(S^l, a^l_{1:K^l}, g^l, F^l\right) \in \mathcal{D}_{\text{data}} : \\ & \qquad F^l = F_{S^l}\left(a^l_{1:K^l}\right) = 1 \\ & \qquad \wedge\ g^l = g^i\ \wedge\ a^l_{1:j} = a^i_{1:j} \\ 0 & \text{else} \end{cases}$$

Relation to (Universal) Q-Functions

Relation to (Universal) Q-Functions

No clear notion of state!

States in LGP

States in LGP

Input to the Neural Network – Encoding a, g and S

How to encode objects if their number can change?

- Instead of feature space representation with fixed dimension, encode scene in image space (depth image)
- Object masks in image space to encode object identity
- Train on only two objects present in the scene

Input to the Neural Network – Encoding a, g and S

Given an action a, decompose it into

$$a = (\bar{a}, O) \in \mathcal{AO}(s, S) \subset \mathcal{A} \times \mathcal{P}(\mathcal{O}(S)),$$

where $\bar{a} \in \mathcal{A}$ discrete action operator symbol and $O \in \mathcal{P}(\mathcal{O}(S))$ the *tuple* of objects the action operates on.
Goal similarly decomposed into $g = (\bar{g}, O_g)$, $\bar{g} \in \mathcal{G}$, $O_g \in \mathcal{P}(\mathcal{O}(S))$.

Input to the Neural Network – Encoding a, g and S

Given an action a, decompose it into

$$a = (\bar{a}, O) \in \mathcal{AO}(s, S) \subset \mathcal{A} \times \mathcal{P}(\mathcal{O}(S)),$$

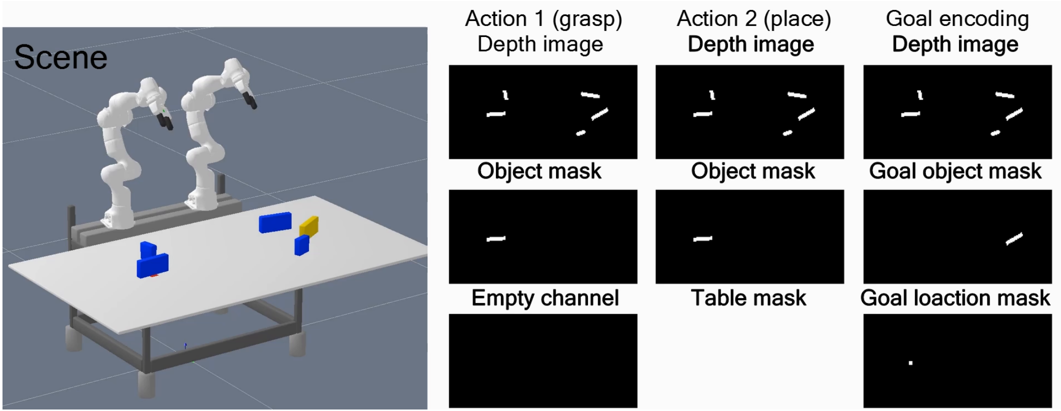
where $\bar{a} \in \mathcal{A}$ discrete action operator symbol and $O \in \mathcal{P}(\mathcal{O}(S))$ the *tuple* of objects the action operates on.
Goal similarly decomposed into $g = (\bar{g}, O_g)$, $\bar{g} \in \mathcal{G}$, $O_g \in \mathcal{P}(\mathcal{O}(S))$.

Cardinality of $\mathcal{A}$ and $\mathcal{G}$ is independent of the scene

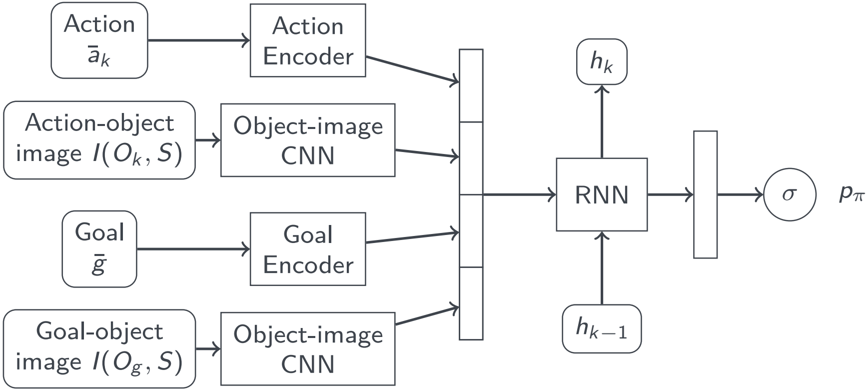
Encoding the objects O and O_g in the image space

Object tuple O is encoded in $n_c + n_O$-channel image

$$I : (O, S) \mapsto \mathbb{R}^{(n_c+n_O)\times w\times h}$$



Network Architecture



$$
\begin{aligned}
(p_\pi, h_k) &= \pi_{\text{NN}}\left(\bar{a}_k, I(O_k, S), \bar{g}, I(O_g, S), h_{k-1}\right) \\
&= \pi\left(a_k, g, a_{1:k-1}, S\right)
\end{aligned}
$$

Experiments

Experiments – Handover

Number of solved NLPs: 1

Total solution time: 1.6 s

Experiments – Geometry Dependence

Number of solved NLPs: 1

Total solution time: 2.1 s

Handover not possible anymore!

Experiments – Placement on Table

Number of solved NLPs: 1

Total solution time: 2.0 s

Experiments – Generalization to Multiple Objects

Number of solved NLPs: 1

Total solution time: 1.5 s

Experiments – Interesting Collaboration of the Two Arms

Number of solved NLPs: 1

Total solution time: 0.9 s

Experiments – Generalization to Multiple Objects

Number of solved NLPs: 1

Total solution time: 1.0 s

Experiments – Object slightly moved – Handover again

Number of solved NLPs: 1

Total solution time: 1.5 s

Experiments – Only one arm needed

Number of solved NLPs: 1

Total solution time: 1.2 s

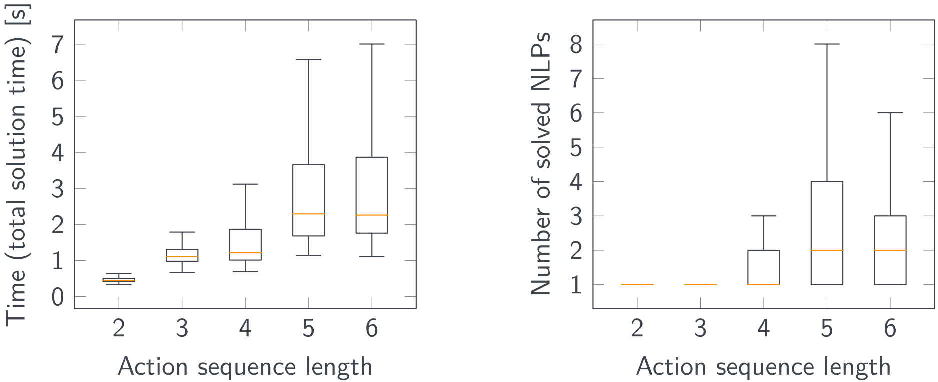
Experiments – Both arms and many objects

Number of solved NLPs: 1

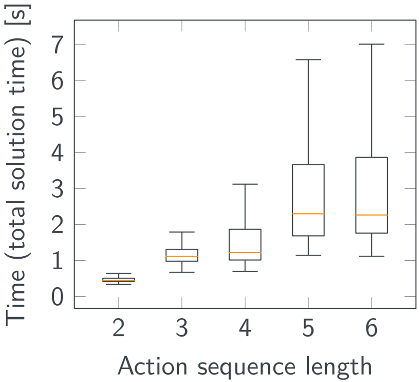
Total solution time: 1.8 s

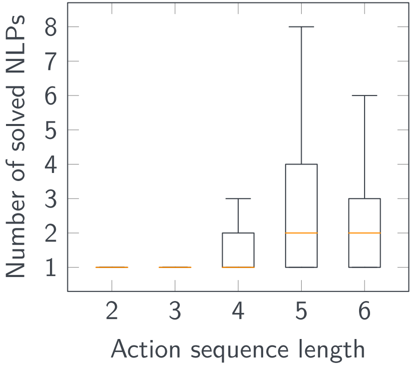
Experiments – Real Robot

Results – Performance on Test Cases with Two Objects



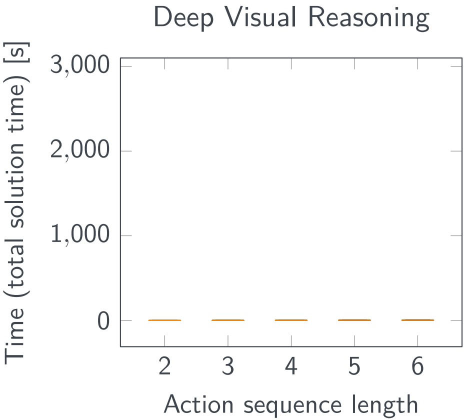
Results – Performance on Test Cases with Two Objects

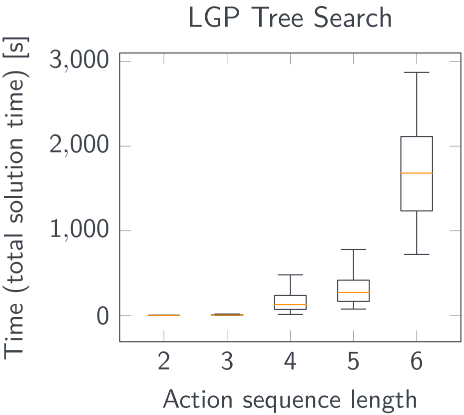




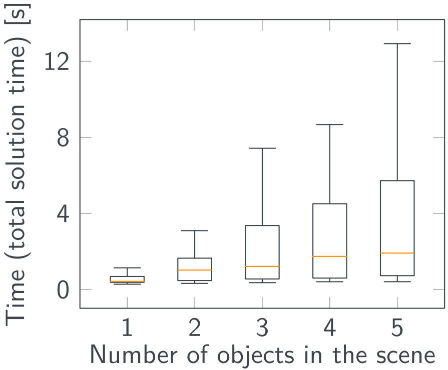
Usually the first proposed action sequence is feasible

Results – Comparison to LGP Tree Search





Results – Generalization to Multiple Objects



Conclusion

- Predict discrete action sequence for Task and Motion Planning from an initial scene image
- High accuracy, i.e. most of the time the first predicted sequence is feasible
- Generalization to multiple objects

Checkout papers

- D. Driess, J. Ha, and M. Toussaint: *Deep Visual Reasoning: Learning to Predict Action Sequences for Task and Motion Planning from an Initial Scene Image.* In Proc. of Robotics: Science and Systems (R:SS), 2020
- D. Driess, O. Oguz, J. Ha, and M. Toussaint: *Deep Visual Heuristics: Learning Feasibility of Mixed-Integer Programs for Manipulation Planning.* In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA), 2020