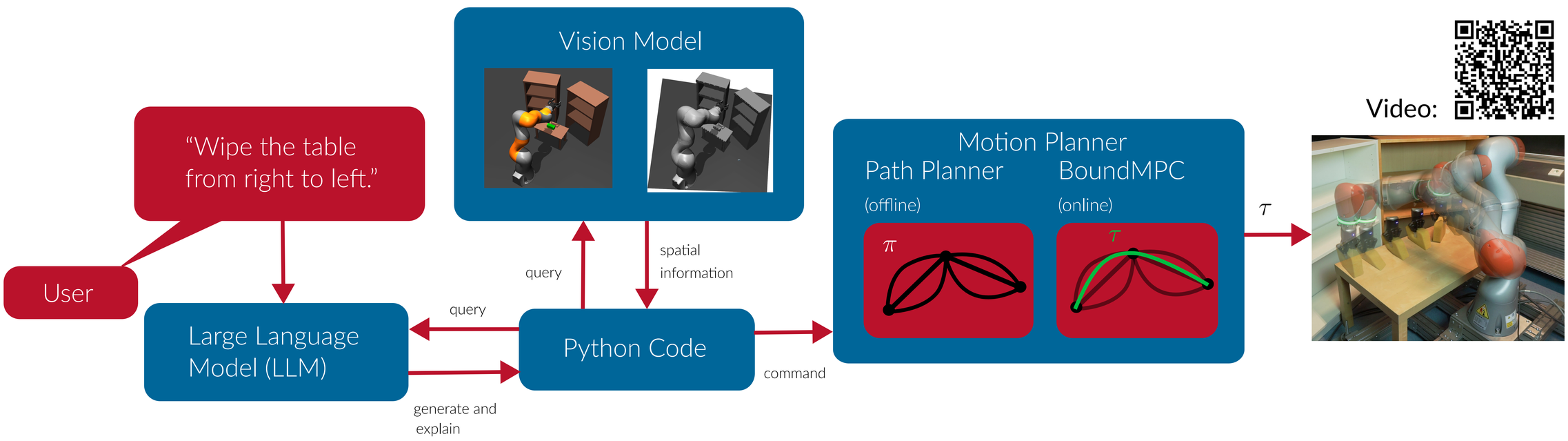
Language-guided Manipulator Motion Planning with Bounded Task Space

Thies Oelerich Christian Hartl-Nesic Andreas Kugi

ACIN AUTOMATION & CONTROL INSTITUTE INSTITUT FÜR AUTOMATISIERUNGS- & REGELUNGSTECHNIK

TU WIEN TECHNISCHE UNIVERSITÄT WIEN



Problem Description

Find a robot manipulator trajectory τ given a language instruction $\mathcal{L}$ by a user, e.g.,

$$\begin{aligned} &\underset{\tau \in \tau_{\text{safe}}}{\text{find}} \quad \tau \\ &\text{s.t. } I_{\text{success}}(\mathcal{L}) = 1 \,, \end{aligned} \tag{1}$$

with

- I_{success}: indicator function for a successful task
- τ_{safe}: set of safe joint trajectories

Example

Consider the instruction $\mathcal{L}$ ="Wipe the table from right to left.":

- τ_{safe}: Joint trajectories that do not collide with the table
- I_{success}: Indicating whether the table was successfully wiped, which requires
  - approaching the right side of the table with the sponge,
  - wiping over the table from right to left.

Framework Overview

The motion planning framework consists of the following components:

1. An LLM for breaking down the instructions $\mathcal{L}$ into actions $\mathcal{A}$ defined in Python code,
2. a vision model to align the actions $\mathcal{A}$ of the LLM with the RGB image of the scene and retrieve spatial locations from a 3D point cloud,
3. a novel path planner based on convex sets to plan a collision-free reference path π,
4. and the model predictive controller BoundMPC to create a trajectory $\tau \in \tau_{\text{safe}}$ that traverses the reference path π.

Motivation for Global Reference Paths

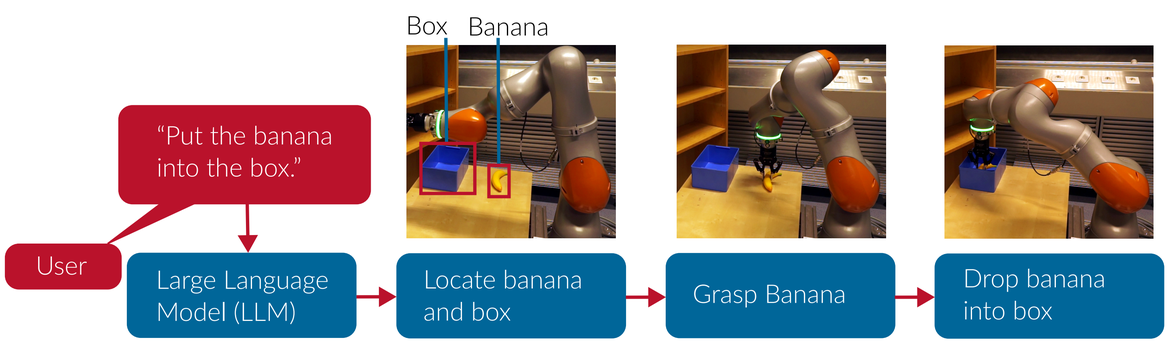
The splitting of Cartesian reference path planning combined with joint-space trajectory optimization is motived by the following considerations:

- **MPC**-based trajectory planning is real-time capable.
- **Global reference paths** for the MPC increase task success and avoid local minima.
- **Cartesian reference paths** are easier to specify and allow task-specific constraints such as moving in a straight line or keeping a cup upright.
- **Bounded reference path deviations** allow more freedom to find executable trajectories, even in kinematically challenging situations.

Vision Model and Large Language Model

The vision model and the LLM are leveraged in the following way:

1. The LLM outputs Python code to perform the necessary actions $\mathcal{A}$ to solve the task.
2. This work uses a combination of language model programs (LMPs) with Chat-GPT4.
3. The Python code interfaces the LLM and the vision model to retrieve information of objects in the scene.
4. The object detection is zero-shot and able to use an open vocabulary.
5. Semantic segmentation helps to locate the object in the 3D scene.



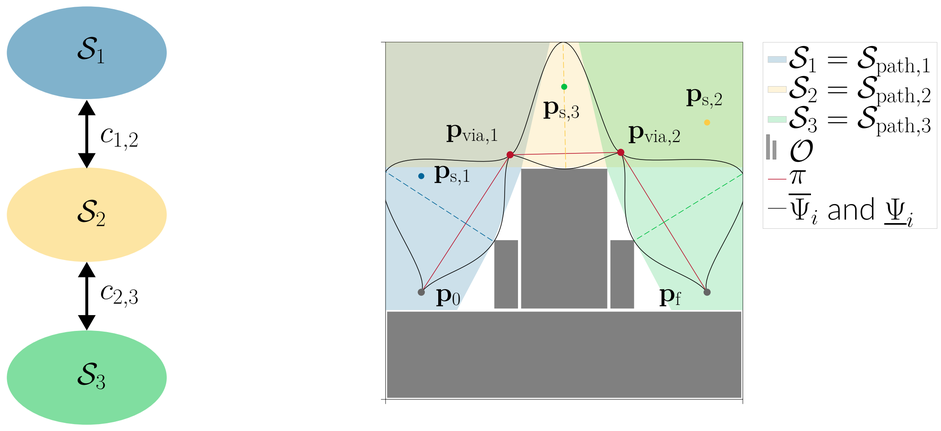
Path Planner based on Convex Sets

Goal

Find a collision-free Cartesian path π from $\mathbf{p}_0$ to $\mathbf{p}_\text{f}$ through the environment. The path deviation bounding functions $\overline{\Psi}$ and $\underline{\Psi}$ specify allowed deviations from the path π.

Procedure

1. Initialize a graph of convex sets $\mathcal{G}$.
2. Grow convex sets $\mathcal{S}_k$ around K given sample points $\mathcal{P} = \{\mathbf{p}_{\text{s},k} : k = 1, \dots, K\}$.
3. Add the convex sets $\mathcal{S}_k$ to the graph $\mathcal{G}$.
4. Compute intersections between sets $\mathcal{S}_k$ in the graph $\mathcal{G}$ and their costs $c_{j,k}$, $j \neq k$.
5. Find optimal path of L sets $\mathcal{S}_{\text{path},l}$, $l = 1, \dots, L$, through graph $\mathcal{G}$.
6. Find piecewise linear reference path through $\mathcal{S}_{\text{path},l}$ defined by the via-points $\mathbf{p}_{\text{via},l}$. Each linear segment is contained within one convex set.
7. Compute the bounding functions $\overline{\Psi}_i$ and $\underline{\Psi}_i$ based on the convex sets $\mathcal{S}_{\text{path},l}$ for each linear segment in basis directions $i = 1, 2$.



(a) Graph $\mathcal{G}$

(b) Example environment

Experimental Results

The following tasks were performed to evaluate our method:

1. **Simple Move:** "Move the [obj] [into/onto] the [left/right] shelf."
2. **Swapping:** "Swap the [obj] with [obj] by using the [table/right shelf]."
3. **Reposition:** "Put the [obj][slowly/quickly] onto the [left/right] side of the table."
4. **Arrangement:** "Arrange the three blocks as a [line/triangle] on the table."
5. **Tea Cup:** "Place the tea cup onto the [left/right] shelf without spilling it."
6. **Wiping:** "Wipe the table from [left/right] to [right/left]."

Table 1. Comparison of our method with VoxPoser (VP)

Task	Success		Collision-free		Avg. path length		Stops	
	VP	Ours	VP	Ours	VP	Ours	VP	Ours
Simple Move	33 %	100 %	83 %	100 %	1.44 m	1.48 m	3	2
Swapping	0 %	100 %	50 %	100 %	-	4.43 m	11	6
Reposition	100 %	100 %	100 %	100 %	1.41 m	0.82 m	3	2
Arrangement	0 %	100 %	0 %	100 %	4.33 m	3.72 m	11	6
Tea Cup	0 %	50 %	100 %	100 %	2.11 m	2.02 m	3	2
Wiping	100 %	100 %	100 %	100 %	1.84 m	1.22 m	4	1

Conclusion

Our proposed method

1. outperforms existing state-of-the-art methods,
2. provides a modular approach to language-guided motion planning,
3. systematically considers the robot's kinematic,
4. creates safe and performant joint-space trajectories.