



Robotics: Task and Motion Planning

Vladimír Petřík

vladimir.petrik@cvut.cz

13.11.2023

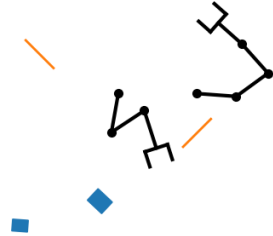
Motivation

- ▶ We know how to plan motion for a robot in robot's configuration space
 - ▶ manually define handle on object
 - ▶ computer grasp and pre-grasp for detected object's pose
 - ▶ plan motion to pre-grasp
 - ▶ interpolate to grasp, grasp
 - ▶ interpolate to pre-grasp
 - ▶ plan motion to pre-place, place, release, pre-place
- ▶ What if we have many handles? Many objects?
- ▶ **Manipulation Task and Motion Planning (TAMP)**
 - ▶ simultaneously plan task and motion solutions
 - ▶ task is the sequence of grasps and placements (discrete space)
 - ▶ motion is the sequence of robot configurations (continuous space)
 - ▶ Humanoid Path Planning (HPP) software approach

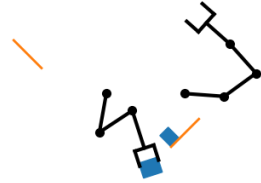


Configuration Space

- ▶ Multiple grippers connected to robots
- ▶ Environment surfaces that can be used for placing an object
- ▶ Multiple objects
 - ▶ multiple handles per object
 - ▶ multiple contact surfaces per object
- ▶ Configuration space is the set of all possible configurations of all objects and robots
 - ▶ $\mathcal{C} = \mathbb{R}^{N_1} \times \mathbb{R}^{N_2} \dots \times SE(3)^M$
 - ▶ N_i DoF of the i -th robot
 - ▶ M number of objects
 - ▶ however, not all configurations are feasible
 - ▶ constraints are used to define feasible configurations



Constraints



- ▶ Object is placed or grasped, i.e. cannot fly
- ▶ Placement constraint
 - ▶ object lies on a surface
 - ▶ numerical constraints
 - ▶ **object surface** is placed on an **environment surface**
- ▶ Grasp constraint
 - ▶ object is grasped by a gripper
 - ▶ numerical constraint
 - ▶ **handle frame** equals **gripper frame**

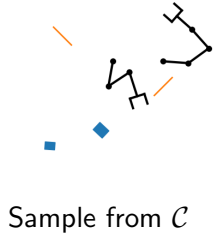


Stav (state)

- ▶ State is a set of constraints
- ▶ Manifold of feasible configurations in the configuration space
- ▶ For example, one state can be defined by constraining both objects
 - ▶ object O_1 is placed on the surface E_1 via object surface S_1
 - ▶ object O_2 is grasped by the gripper G_1 via handle H_1
- ▶ How to sample configuration from a state?
 - ▶ sample from the \mathcal{C}
 - ▶ geometric projection to satisfy all the constraints
 - ▶ numerical optimization (Newton-Raphson) to satisfy all the constraints



Sampling from states



- ▶ Project to state:
 - ▶ O_1 placed on E_1 via S_1
 - ▶ O_2 placed on E_2 via S_1

- ▶ Project to state:
 - ▶ O_1 grasped by G_1 via H_1
 - ▶ O_2 placed on E_2 via S_1



Transitions

- ▶ Transition defines motion between two states
 - ▶ identity transition allows to move robot inside the state
 - ▶ place transition allows to move object from the gripper to the surface
 - ▶ grasp transition allows to move object from the surface to the gripper
- ▶ Sampling on transitions vs sampling on states
 - ▶ transition respect constraints from the given state
 - ▶ for example, identity on place state will not move object (sampling on state can move object)
 - ▶ grasp transition is specified to move via pre-grasp
 - ▶ place transition is specified to move via pre-place



Interpolation on transition

- ▶ Interpolate between two configurations but respect constraints of the states/transition



Grasp

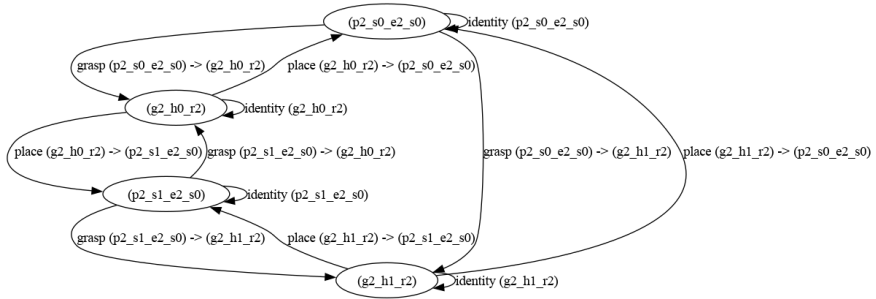


Place



Constraint graph

- ▶ Defines all possible transitions between existing states
- ▶ Example: single arm, one object

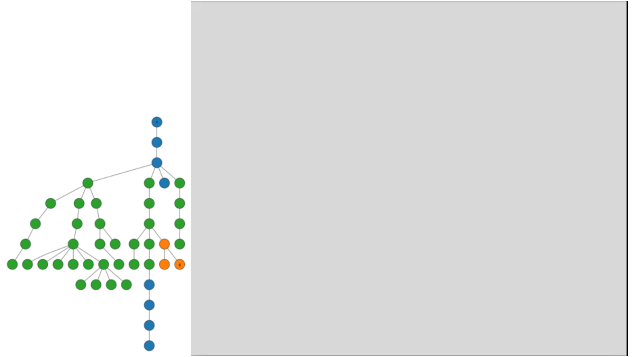
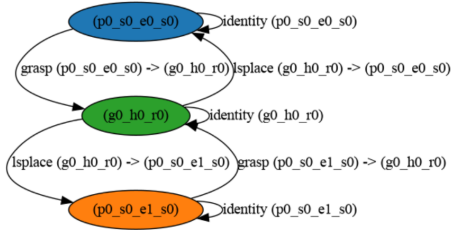


RRT on constraint graph

- ▶ Random sampling q_{rand}
 - ▶ sample random transition
 - ▶ select random existing configuration from the transition source
 - ▶ sample random configuration from the transition target reachable from beginning
- ▶ Nearest neighbor q_{tree}
 - ▶ node that is closest to q_{rand} via interpolation on the transition
- ▶ Local planner uses interpolation on transition



RRT on constraint graph



Conclusion

- ▶ Configuration space for TAMP is complex
 - ▶ discrete set of states
 - ▶ continuous motion
 - ▶ encoded by constraint graph that allow us to use RRT
- ▶ Usually not used in industry
 - ▶ task space sequence is hard-coded by programmers
 - ▶ only motion is found by motion planners (if cannot be hard-coded)
- ▶ How to avoid hard-coding? Video demonstration.



Laboratories

- ▶ Consultation on the final project
- ▶ Final project is now described on the course web page
- ▶ New interface for Bosch robot [optional]
 - ▶ fixed FK, IK
 - ▶ you can install it on your computer, to use FK and IK offline for debugging

