Efficient Path Planning for Modular Robots

Background

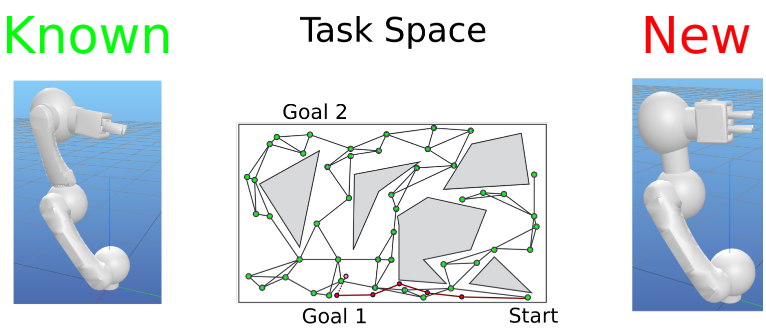
Robots have yet to make an entrance into many industries (outside mass-manufacturing) and the service sector. A major hurdle to their widespread adoption is the complexity of their deployment, requiring costly specialists for every new task one wants to automate. We envision simple tools which help to choose the right robot and do the programming given a task and environment specification.

Description

Path planning for robotics is still an area of active research requiring an ongoing trade-off between run-time and completeness. Sampling based planners, such as RRT(*), offer state of the art performance but consider each planning attempt to be independent of all other, especially if robot types change [2] [4, Ch. 10]. This is often the case when deploying modular reconfigurable robots, which easily offer millions of possible assemblies [1]. We want to test if and how results from planning for one robot configuration could be reused when planning with another (similar) robot.

Tasks

- Literature review on modular robot path planning
- Familiarize yourself with the mcs toolbox / timor [3] and CoBRA benchmark suite [5]
- Extend motion planning pipeline; ideas:
  - Find planning results from similar previously tested robots and use as initialization for next planning
  - Devise strategies to adapt previous planning results to new robot
  - Evaluate effectiveness of reuse and how to find closest previous solution



Idea on how to reuse the previous roadmap planned with the left robot showing explored states in green and connections in black. The new robot is kinematically similar but a bit shorter; it cannot use the old path directly. With some changes to the known plan it can reach goal 1 without having to explore the whole configuration space; its path is shown in red. To also find a path towards goal 2 the new robot tries to explore along the known path from the other robot as shown by the next extended node in lighter red with doted connecting line. Central image based on [4, Fig. 10.20].

Alternatives / Extensions

An additional focus could be laid on a tighter C integration of the robot model created with pinocchio[1] and the motion planner using the OMPL framework[2]. This might also be appropriate as a standalone bachelor's thesis with a strong software engineering background.

[1]https://github.com/stack-of-tasks/pinocchio
[2]https://ompl.kavrakilab.org/

Technische Universität München

Fakultät für Informatik

Lehrstuhl für Echtzeitsysteme und Robotik

Supervisor:
Prof. Dr.-Ing. Matthias Althoff

Advisor:
Matthias Mayer

Research project:
Modular Robotics

Type:
MA/BA

Research area:
Robotics, Path Planning

Programming language:
C++, or Python

Required skills:
Motion Planning, Linux

Language:
English

Date of submission:
20. Oktober 2022

For more information please contact us:

Phone: +49.89.289.18114

E-Mail: matthias.mayer@tum.de

Internet: www6.in.tum.de

References

[1] M. Althoff, A. Giusti, S. B. Liu, and A. Pereira. Effortless creation of safe robots from modules through self-programming and self-verification. *Science Robotics*, 4(31):1–14, 2019.

[2] Mohamed Elbanhawi and Milan Simic. Sampling-based robot motion planning: A review. *IEEE Access*, 2:56–77, 2014.

[3] Jonathan Külz, Matthias Mayer, and Matthias Althoff. Toolbox for Industrial Modular Robotics: Timor Python. arXiv:2209.06758 [cs.RO], 2022.

[4] Kevin M. Lynch and Frank C. Park. *Modern Robotics - Mechanics, Planning and Control*. Cambridge University Press, 3rd printi edition, 2017.

[5] Matthias Mayer, Jonathan Külz, and Matthias Althoff. CoBRA: A Composable Benchmark for Robotics Applications. arXiv:2203.09337 [cs.RO], 2022.

Technische Universität München

Fakultät für Informatik

Lehrstuhl für Echtzeitsysteme und Robotik