Trajectory Planning Using Manipulability Maximization and Control Minimization

ALEX LI, McGill University

Code: https://github.com/Matheer0/559-project.git
Video: https://youtu.be/Mqyydrkx49Q

Additional Key Words and Phrases: robot control, optimization

ACM Reference Format:
Alex Li. 2021. Trajectory Planning Using Manipulability Maximization and Control Minimization. 1, 1 (April 2021), 2 pages. https://doi.org/0000001.0000001

1 INTRODUCTION

Trajectory planning is a key task for robots to move from an initial position to the goal, and it is relatively simple for mobile platforms since the manipulation is in 2D and the shortest path and obstacle avoidance can be solved using A* [Nilsson et al. 1968] and RRT (Rapidly-exploring Random Tree) [LaValle and Kuffner 2000]. However, path planning is more challenging for robot arms, because they are usually involved in more sophisticated jobs in 3D space, such as delivering a cup of water without spilling. Normally, the manipulation is abrupt when the robot tip is unable to generate velocities in certain directions caused by singular Jacobian matrices, where the columns become collinear [Park and Lynch 2017].

To avoid such singular configurations, maximizing the manipulability index at each timestep has been widely used in robot arm control to have higher redundancy. Inspired by this idea, I would like to apply it beforehand at the planning level to reduce the controller's work. Meanwhile, since another common objective in robot arm planning is to minimize the control cost for manipulation, I would like to construct an offline energy-efficient planning algorithm that allows the manipulator to perform dexterous tasks.

2 RELATED WORK

Since the manipulability index was first introduced by Yoshikawa [1985], it has been widely used in optimizing robot controllers. For example, Chen et al. [2019] worked on the vision-based mobile manipulator's controller to select and adjust poses by maximizing its manipulability on given manipulation tasks. Recently, Maric et al. [2019] formulated manipulability maximization as a continuous-time optimization problem using Bayesian statistics.

So far, most of the work on manipulability maximization has focused on robot control. In this project, I would like to implement the idea and further extend it to trajectory planning.

3 METHODS

3.1 Naive Trajectory Planning

For quick system setup, we assume there is no obstacle avoidance, and our raw planner simply uses the *transformtraj* function from the Robotics System Toolbox in MATLAB to generate a discretized straight path in 3D space from the end-effector's arbitrary initial position to the goal:

[desired_poses, velocities] = transformtraj(init, goal, travel_time, timestep) (1)

where the inputs and outputs are written in the form of 4-by-4 homogeneous transformations from the Cartesian space.

3.2 Optimize Controller by Maximizing Manipulability

Given the end-effector's planned position at the next timestep, controllers can compute the transformed new configuration (q_new), and calls the inverse kinematics function to calculate the control (u) to move from the current configuration (q_current):

u = inverse_kinematics (q_new, q_current) (2)

Since we would like the robot arm to be in a configuration with relatively high redundancy, we exploit the manipulability maximization to optimize the controller to find the optimal q* that has the highest measure (concerning physical constraints):

$$\max \sqrt{det[J(q)J^T(q)]} \tag{3}$$

Thus, we have our optimized control algorithm:

---

for i = 1 : steps
  desired_pose <- desired_poses(i)
  # fmincon is in MATLAB's Optimization Toolbox for solving (3)
  q* <- fmincon(-manipulability_index, desired_pose, constraints)
  u <- inverse_kinematics (q_current, q*)
  q_current <- update_configuration (q_current, u)
end for

3.3 Minimizing Control Cost

Besides maximizing manipulability, we also would like to reduce the energy for manipulation. Therefore, our additional objective function in trajectory planning is minimizing the total control cost:

$$\min \int u^T(t)u(t)\,dt \tag{4}$$

3.4 Planner Optimization

To plan a balanced trajectory with relatively high redundancy and low control cost, we need to combine (3) and (4) in the planner. To do so, we can introduce a trade-off ratio (α) representing how much we care about the manipulability measure, and our new objective function will be:

$$\min \int u^T(t)u(t) - \alpha \cdot measure\,dt \tag{5}$$

Then, our planning can be solved using *OptimTraj* library constructed by Kelly [2017], with predefined hyperparameters including the number of iterations and the choice of integration method.

Author's address: Alex Li, McGill University, qixuan.li@mail.mcgill.ca.

2021. XXXX-XXXX/2021/4-ART $15.00
https://doi.org/0000001.0000001

Algorithm: Entire Pipeline
Input: (α, q_initial, destination, travel_time, dynamics_constraints)
 # planning level
desired_poses <- Opt_Planner(α, q_initial, destination, travel_time, dynamics_constraints)

 # control level
run optimized controller (Section 3.2)

4 RESULTS

For simulation, I choose to use KUKA iiwa 14, a 7-joint industrial manipulator, from the Robotics System Toolbox, which contains a variety of robot libraries.

4.1 Naive Planning with Optimized Control

To evaluate our optimized controller, we start the experiment with our naive planner. With the same linear trajectory has planned, we run the optimized and non-optimized controllers respectively and store the manipulability index at every time step during the manipulation process.

By comparing the corresponding average, minimum and maximum manipulability values (Table 1), we notice the optimized controller makes a significant improvement in the manipulability because the experiment using the optimized controller always has higher values.

Controller	Average	Min	Max
non-opt	8.224e-02	2.264e-02	1.059e-01
opt	**1.161e-01**	**6.027e-02**	**1.595e-01**

Table 1: Average, minimum and maximum of manipulability measures from linear trajectory planning with different controllers.

4.2 Optimized Planning and Control

We repeat our simulation process as above, but use the optimized planner this time. As illustrated in Table 2, our manipulability across distinct trade-off parameters is also improved with the optimized controller.

alpha	Controller	Average	Min	Max
0	non-opt	3.448e-02	4.624e-03	6.23e-02
	opt	**7.544e-02**	**3.088e-02**	**1.281e-01**
0.1	non-opt	3.821e-02	3.796e-03	9.865e-02
	opt	**7.924e-02**	**3.293e-02**	**1.319e-01**
0.5	non-opt	4.197e-02	1.152e-03	1.015e-01
	opt	**8.752e-02**	**4.003e-02**	**1.465e-01**
0.8	non-opt	3.367e-02	2.007e-03	6.608e-02
	opt	**8.210e-02**	**2.209e-02**	**1.475e-01**
0.9	non-opt	3.383e-02	2.216e-03	6.714e-02
	opt	**8.509e-02**	**2.389e-02**	**1.510e-01**

Table 2: Average, min and max of manipulability measures from optimized planning with different controllers and trade-off parameters.

4.3 Trade-off Parameter Tuning

Since our goal is to provide an offline planning algorithm that generates a sequence of poses with relatively high manipulability and low energy cost, we need to discover a corresponding trade-off parameter for the objective function that balances maximization and minimization objectives.

By analyzing the relationship between the manipulability and α under optimized planning and control (Figure 1), we can get a relatively high measure when α is around 0.5. Thus, 0.5 is a potential value for the optimal trade-off parameter in our optimized planner.

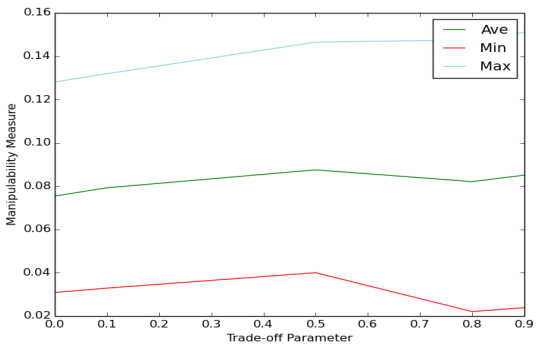

Figure 1: Illustration of the relationships between manipulability measures and distinct α.

5 CONCLUSIONS

In this project, I optimized the robot arm's controller and planner using manipulability maximization. The results showed that generating a sequence of configurations with high redundancy would allow the manipulator to perform dexterous tasks. Moreover, I studied the effect of various trade-off values in the planning algorithm towards balancing the manipulation and control cost and found out that setting it to 0.5 might be a good choice.

Due to lack of time, my hyperparameter selection and parameter tuning were rough, and this optimization process could be replaced by machine learning methodology. Furthermore, we could advance the planner to include more features, such as obstacle avoidance.

REFERENCES

F. Chen, M. Selvaggio, and D. Caldwell. 2019. Dexterous Grasping by Manipulability Selection for Mobile Manipulator With Visual Guidance. *IEEE Transactions on Industrial Informatics, Vol. 15.* 1202-1210 (2019).

M. Kelly. 2017. An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation. *Society for Industrial and Applied Mathematics (SIAM), Vol. 59, No.4.* 849-904 (2017).

S. LaValle and J. Kuffner. 2000. RRT-connect: An efficient approach to single-query path planning. *In Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation, Vol. 2.* 995-1001. https://ieeexplore.ieee.org/document/844730 (2000).

F. Maric, O. Limoyo, L. Petrović, T. Ablett, I. Petrović, and J. Kelly. 2019. Fast Manipulability Maximization Using Continuous-Time Trajectory Optimization. *In IEEE International Conference on Intelligent Robots and Systems (IROS).* 8258-8264 (2019).

N. Nilsson, P. Hart, and B. Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics, Vol. 4.* 100-107 (1968).

F. Park and K. Lynch. 2017. Modern Robotics - Mechanics, Planning, and Control. *ISBN 9781107156302, Cambridge University Press. http://modernrobotics.org* (2017).

T. Yoshikawa. 1985. Manipulability of Robotic Mechanisms. *The International Journal of Robotics Research, Vol. 4.* 3-9 (1985).