Mobile Manipulation

Satyandra K. Gupta

Director, Center for Advanced Manufacturing
Smith International Professor
Aerospace and Mechanical Engineering Department
Computer Science Department
Viterbi School of Engineering
University of Southern California

https://sites.usc.edu/skgupta/
What is Mobile Manipulation?

Manipulating an object from a moving platform

Natural examples of mobile manipulation
Examples of Mobile Manipulators

Mobile Manipulator

Bimanual Mobile Manipulator
Objectives

To develop computational foundations for the autonomous operation of mobile manipulators

- Automate mobile manipulator motion planning
- Generate smooth motion plans for the robot
- Generate these smooth motion plans quickly
With these planning capabilities, several complex tasks can be performed; Example: Machine tending

- Opening Door
- Picking up part
- Moving the part to new station
Overview

Mobile Manipulation Planning

- Planning for Object Pick-Up
  - Mobile Robot Motion Planning
- Point-to-Point Planning
  - Manipulator Motion Planning
- Planning for Area Coverage
- Task-Assignment & Motion Planning
  - Grasping under Uncertainty
Time-Optimal Trajectory Planning for Pick-and-Transport Operation with a Mobile Manipulator
Motivation & Objective

● Inputs:
  ○ Initial and final configurations of the mobile manipulator
  ○ Static map
  ○ Part pose
  ○ Maximum joint rates
  ○ Grasping strategy

● Output:
  ○ Time-optimal mobile base trajectory for picking up objects from a known location

Example factory setting (generated using V-rep)
Overview of Approach

- Hybrid A* based motion planner for the mobile base in 3D ($x$, $y$, $\phi$)
- Constant-time motion primitives
- Specialized heuristics to guide the search

Constant-time motion primitives
Resulting Trajectories

Mobile Base Travel Time = 17 sec
Manipulator Delay = 5.7 sec
Total Execution Time = 22.7 sec

Mobile Base Travel Time = 18 sec
Manipulator Delay = 0 sec
Total Execution Time = 18 sec
Manipulator Trajectory Planning on a Moving Mobile base for Part Pick-up and Transport Operations
Motivation & Objective

- To determine the manipulator motion for grasping the part while the mobile base moves along a predetermined path
- The manipulator should start moving as late as possible
- The mobile base should slow down only if necessary
Overview of Approach

- Baseline method is to grow multiple trees, one for each grasping strategy.
- Grow a single tree from the initial configuration.
- Connect to a tree at random and that gives a baseline path.
- Three techniques to speed up the motion planning, generate smooth paths, select appropriate goal configuration and the grasping strategy were developed.
Results
Accounting for Part Pose Estimation Uncertainties during Trajectory Generation for Part Pick-Up Using Mobile Manipulators
To reduce time, mobile manipulators can pick-up objects while the mobile base & gripper are moving.

Uncertainty in the pose of the object may result in failure to grasp.
Objectives

- Compute the gripper speed and the gripper closing speed so that there is a high probability of success in grasping.
- Determine the mobile manipulator trajectory so that the gripper moves with that desired speed.
Overview of Approach

- SVM based active learning for determining the meta-model
- Successive refinement based parameter optimization for trajectory generation of the mobile manipulator
Accelerating Bi-Directional Sampling-Based Search for Motion Planning of Non-Holonomic Mobile Manipulators
Motivation

• For transportation tasks with a mobile manipulator, the manipulator can be at a “home” position on the mobile base and move after the mobile base arrives at the target location.

• In cases where large objects are to be transported in narrow passages, there may be a need to simultaneously move the manipulator and the mobile base.
Problem Statement

• Given:
  • Start and goal configurations of the nonholonomic mobile manipulator (3+n DOF) with a single or multiple manipulators mounted on the mobile base
  • The 3D environment with obstacles
• To Find:
  • A feasible smooth path of the mobile manipulator from the start to the goal configuration that satisfies the nonholonomic motion constraints
Overview

- Focused Sampling
- Feasible Connection
- Best Nearest Node
- Tree Extension

RRT-Connect like Motion Planners
The HS-Bi-RRT Algorithm

• The Hybrid Sampling based Bi-directional RRT (HS-Bi-RRT) is the overall algorithm

• $R_s$ is called the Sampling Ratio, i.e. the percentage of the time the sampling is done in the $(3+n)$-D Configuration Space (with mobile base in the workspace disks) vs the 6-D pose in Workspace

• To maintain the completeness property, we sample in the entire $(3+n)$-D Configuration space for a small fraction of time
The HS-Bi-RRT Algorithm

- Sampling is done inside the workspace sphere and the workspace disks from a normal distribution with the mean at their center and a varying standard deviation.
- Resulting in Exploration-Exploitation type sampling to grow the trees in appropriate regions.
• 20 different test cases: Combination of a variety of large objects being carried in varied environments
• Bi-manual mobile manipulator carrying two different objects in two different hands
HS-Bi-RRT has been compared with 3 other competing methods. The connection heuristics are used in all methods:

1. WD-Bi-RRT: Workspace Disks Bi-RRT, sampling for mobile base only in the disks
2. WS-Bi-RRT: Workspace Bi-RRT, sampling only in the workspace spheres
3. Bi-RRT+: Bi-RRT with sampling in the entire configuration space and the connection heuristics

### Results

<table>
<thead>
<tr>
<th>Scene</th>
<th>Average Computation Time (s)</th>
<th>Average Path Cost (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HS-Bi-RRT</td>
<td>WD-Bi-RRT</td>
</tr>
<tr>
<td>S04</td>
<td>37.5</td>
<td>163.4</td>
</tr>
<tr>
<td>S07</td>
<td>32.9</td>
<td>175.8</td>
</tr>
<tr>
<td>S12</td>
<td>20.9</td>
<td>75.7</td>
</tr>
<tr>
<td>S13</td>
<td>19.1</td>
<td>33.0</td>
</tr>
<tr>
<td>S17</td>
<td>77.3</td>
<td>302.4</td>
</tr>
</tbody>
</table>
Results

- Without the connection heuristics, there is high failure rates in all methods:
  1. HS-Bi-RRT: 73%
  2. WD-Bi-RRT: 81%
  3. WS-Bi-RRT: 71%
  4. Bi-RRT+: 90%

- Also, in cases when there is a success, on average the computation time is ~9 times higher as compared to when the connection heuristics are used
Area Coverage Planning for Spray-Based Surface Disinfection with a Mobile Manipulator
Motivation

- Manual disinfection can be a time-consuming, risky, labor-intensive, and mundane, and humans may fail to disinfect critical areas due to fatigue.
- Mobile manipulators mounted with a spray nozzle at the end-effector can be very effective in spraying disinfectant liquid for deep disinfection of objects and surfaces.
Objective

- Given a point cloud, the objective is to:
  - (a) Compute a mobile manipulator trajectory such that the spray nozzle motion results in the entire area of the surface being covered
  - (b) Compute the joint velocities such that each point on the surface receives enough disinfectant to guarantee thorough disinfection
Overview

Determine Multiple possible Spray Paths on the Point Cloud

Choose an appropriate Spray Path

Determine intervals between Waypoints to Guarantee Disinfection of every point on the Point Cloud

Using the Spray Path, determine Nozzle Path & Mobile Manipulator Trajectory
Results: Test Cases
Results (contd.)

- Spray Paths Generated on the polygon
### Spray path generation algorithm performance

<table>
<thead>
<tr>
<th>Test Case</th>
<th># Spray Polygon Edges</th>
<th># Polygon Segments Expanded</th>
<th># Branches Pruned</th>
<th># Spray Paths Generated</th>
<th># Grid Points Not Sprayed on</th>
<th>% Area NOT Sprayed</th>
<th>Spray Path Length (m)</th>
<th>Computation time (s) for generating first spray path</th>
<th>Computation Time (s) for last spray path</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>31</td>
<td>688</td>
<td>266</td>
<td>8</td>
<td>67</td>
<td>0.59%</td>
<td>7.66</td>
<td>0.05</td>
<td>7.6</td>
</tr>
<tr>
<td>b</td>
<td>11</td>
<td>153</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>0%</td>
<td>5.95</td>
<td>0.02</td>
<td>1.0</td>
</tr>
<tr>
<td>c</td>
<td>32</td>
<td>70</td>
<td>53</td>
<td>5</td>
<td>4</td>
<td>0.03%</td>
<td>11.21</td>
<td>0.14</td>
<td>0.98</td>
</tr>
<tr>
<td>d</td>
<td>5</td>
<td>255</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0%</td>
<td>13.65</td>
<td>0.06</td>
<td>1.4</td>
</tr>
<tr>
<td>e</td>
<td>40</td>
<td>463</td>
<td>257</td>
<td>1</td>
<td>115</td>
<td>0.71%</td>
<td>10.79</td>
<td>0.18</td>
<td>5.8</td>
</tr>
</tbody>
</table>
Benchmarking for test case a: (a) The longest convex edge zig-zag path (b) principle component zig-zag Path and (c) Spiral path

Computed using Spray Path Generation Algorithm

(a) The longest convex edge zig-zag path
(b) Principle component zig-zag path
(c) Spiral path
### Spray path generation algorithm performance

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Spray Path Generation Algorithm</th>
<th>Longest Convex Edge Zig-Zag Path</th>
<th>Principle Component Zig-Zag Path</th>
<th>Spiral Path</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Path Length (m)</td>
<td>% Area NOT Sprayed</td>
<td>% of Spray Wasted</td>
<td>Path Length (m)</td>
</tr>
<tr>
<td>a</td>
<td>7.66</td>
<td>0.59%</td>
<td>2.11%</td>
<td>7.37</td>
</tr>
<tr>
<td>b</td>
<td>5.95</td>
<td>0%</td>
<td>1.23%</td>
<td>5.13</td>
</tr>
<tr>
<td>c</td>
<td>11.21</td>
<td>0.03%</td>
<td>1.11%</td>
<td>10.41</td>
</tr>
<tr>
<td>d</td>
<td>13.65</td>
<td>0%</td>
<td>0.96%</td>
<td>12.71</td>
</tr>
<tr>
<td>e</td>
<td>10.79</td>
<td>0.71%</td>
<td>2.15</td>
<td>10.50</td>
</tr>
</tbody>
</table>
• Spray Paths Generated on the point clouds
Results (contd.)

- Results for trajectory execution time before and after time interval determination for thorough disinfection

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Trajectory Execution Time (s) Before Retiming</th>
<th>% Points not sprayed enough before retiming</th>
<th>Trajectory Execution Time (s) After Retiming</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>12.5</td>
<td>23.8%</td>
<td>40.2</td>
</tr>
<tr>
<td>b</td>
<td>14.5</td>
<td>68.1%</td>
<td>38.5</td>
</tr>
<tr>
<td>c</td>
<td>41.4</td>
<td>36.7%</td>
<td>71.2</td>
</tr>
<tr>
<td>d</td>
<td>36.2</td>
<td>13.3%</td>
<td>93.4</td>
</tr>
<tr>
<td>e</td>
<td>24.4</td>
<td>24.7%</td>
<td>48.8</td>
</tr>
</tbody>
</table>
Task Assignment and Motion Planning for Bimanual Mobile Manipulation
• Existing approaches on Task and Motion planning
  – designed for task assignment do not consider motion planning
  – designed for task and motion planning
    o cannot eliminate infeasible task-agent assignment without invoking motion planners
    o are not suitable for operations where agents need to collaborate and coordinate for complex tasks
Objectives

• For complex tasks, have task-agent assignment for a bimanual mobile manipulator such that the query to the motion planners is done only when we’re sure that the task assignment has no conflicts and is physically possible.

• In the case where motion planner fails, re-plan only the failed part of the motion.
• Given a task network (T) for a Bimanual mobile manipulator, our objective is to
  – assign tasks to the agents i.e left arm, right arm and the mobile base,
  – sequence the tasks, and
  – generate continuous configuration space trajectories for each agent
  such that
  – there is no conflict in the assignment, and
  – the time span of the operation is minimized
Basic Idea

- A two-layer architecture for integrating motion planning and task-agent assignment

- Task-Agent Assignment Layer
  - Task Constraint Heuristic
    - based on *immediacy* & *concurrency*
  - Spatial Constraint Heuristic
    - based on *reachability* and existence of *collision free IK*
Overview of Approach

• A search based architecture for task-agent assignment
• Prune the task agent assignment nodes in the search tree based on the task constraints and spatial constraint
• Once the task-agent assignment is complete, move on the motion planning layer
• Cache the motions which succeed and only change those task-agent assignments which fail
Task Assignment Trees

- Divide the task network into time windows
- Each task can be done by one of the six set of agents
Node Pruning and Caching

- Prune nodes based on task constraints heuristic, spatial constraints heuristic and previous assignment
- A caching scheme to move from the motion planning layer to task agent assignment layer
Results

- Spatial constraint heuristics and caching scheme help in reducing computation time by 86% on average

<table>
<thead>
<tr>
<th>Task No.</th>
<th>With spatial constraint checking and caching (sec)</th>
<th>Without spatial constraint checking and caching (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>32.3</td>
<td>479.7</td>
</tr>
<tr>
<td>2</td>
<td>15.3</td>
<td>257.5</td>
</tr>
<tr>
<td>3</td>
<td>71.9</td>
<td>921.3</td>
</tr>
</tbody>
</table>
https://sites.usc.edu/skgupta/publications/
Videos

https://www.youtube.com/channel/UCO82Tsg5Xc5vP_ZWkax4Wpg