Grocery Shopping Robot

Chun Ming Jeffrey Zhang
Gabriela Bravo-Illanes
Max Anders Farr
Motivation

Contactless grocery shopping

- Reduces COVID spread
- Generally convenient option
- Interesting problem to solve with robots
State Machine
High level

Start

Go to shelf

Arrived

Pick object from shelf

More items in shopping list

finished shopping list

Go to checkout

Arrived

Place basket in conveyor belt

End
State Machine
Pick and place

- Start
- Bring arm closer
  - Reached position
- Orient arm
  - Reached desired orientation
- Approach the object
- Lift object
  - Reached position
- Retreat
  - Reached position
- Place Object
  - Reached position
- Touched object
- End
Controllers

1. End effector task controller to reach objects in world
   a. Task controller of the arm and base to reach an end effector position

Goal
End effector position (world frame)

Operational space controller
Controllers

2. End effector task controller to reach objects in base frame (base still)
   a. Compute control of arm in task space to reach point with the end effector in base frame
   b. Compute control of base in joint space to maintain position

Arm goal
End effector position (base frame)

Operational space controller

Base goal
Maintain original (x, y, θ)

Joint controller
Controllers

3. Controller to move only base while keeping robot in home position
   a. Joint controller where arm joint position is fixed

Arm goal
Keep joints home position

Joint controller

Base goal
Go to a new position
\((x,y,\theta)\)
Controllers

4. Gripper finger controller
   a. PD controller to achieve desired position of the fingers
   b. Feedforward of the desired force

Note: This controller works in parallel with the others
Environment
Environment

Products

Top-down view
Environment

Mostly static!
Navigation Path Planning

Insight: Any two points can be reached with at most 2 lattice waypoints! (no need A*)
Case 1: 0 waypoint

Direct line of sight → no waypoint needed
Case 2: 1 waypoint

∃p ⊂ lattice_points()

s.t.

can_see(start, p) and

can_see(goal, p)

i.e. start and goal both see a common lattice point

**There can only be one such point if start and goal don’t see each other!**
Case 3: 2 waypoints (general case)
Challenges
Separate base and arm movement

First attempt: Tweaking Jacobians didn’t work
Challenges
Separate base and arm movement

Solution:

**Navigation controller**

\[
\begin{bmatrix}
F_b \\
F_{arm}
\end{bmatrix} = M(-kv_j \dot{q} - kp_j (q - q_d))
\]

\[
q_d = \begin{bmatrix} q_{B,d} \\
q_{arm,home}
\end{bmatrix}
\]

**Only arm controller**

\[
F_{arm} = J_{0,arm} \Lambda_{0,arm} \begin{bmatrix} -kv (\dot{x} - \nu \dot{x}_d) \\
-k_p \delta \phi - kv \omega
\end{bmatrix} + N_{0,arm} M_{arm} (-kv_j \dot{q}_{arm})
\]

\[
F_b = M_b (-kv_j \dot{q}_b - kp_j (q_b - q_{b,d}))
\]
Challenges

Shelf collision object
Challenges
Shelf collision object

Solution
Challenges

Shadows didn’t show up
- Complex models (the shelf and basket) suffered as a result.
Challenges
Base Frame
Future work

- Move basket with objects
- Picking up individual items from the basket
- Scan object barcodes for self-checkout
- Hands your life-savings to the cashier