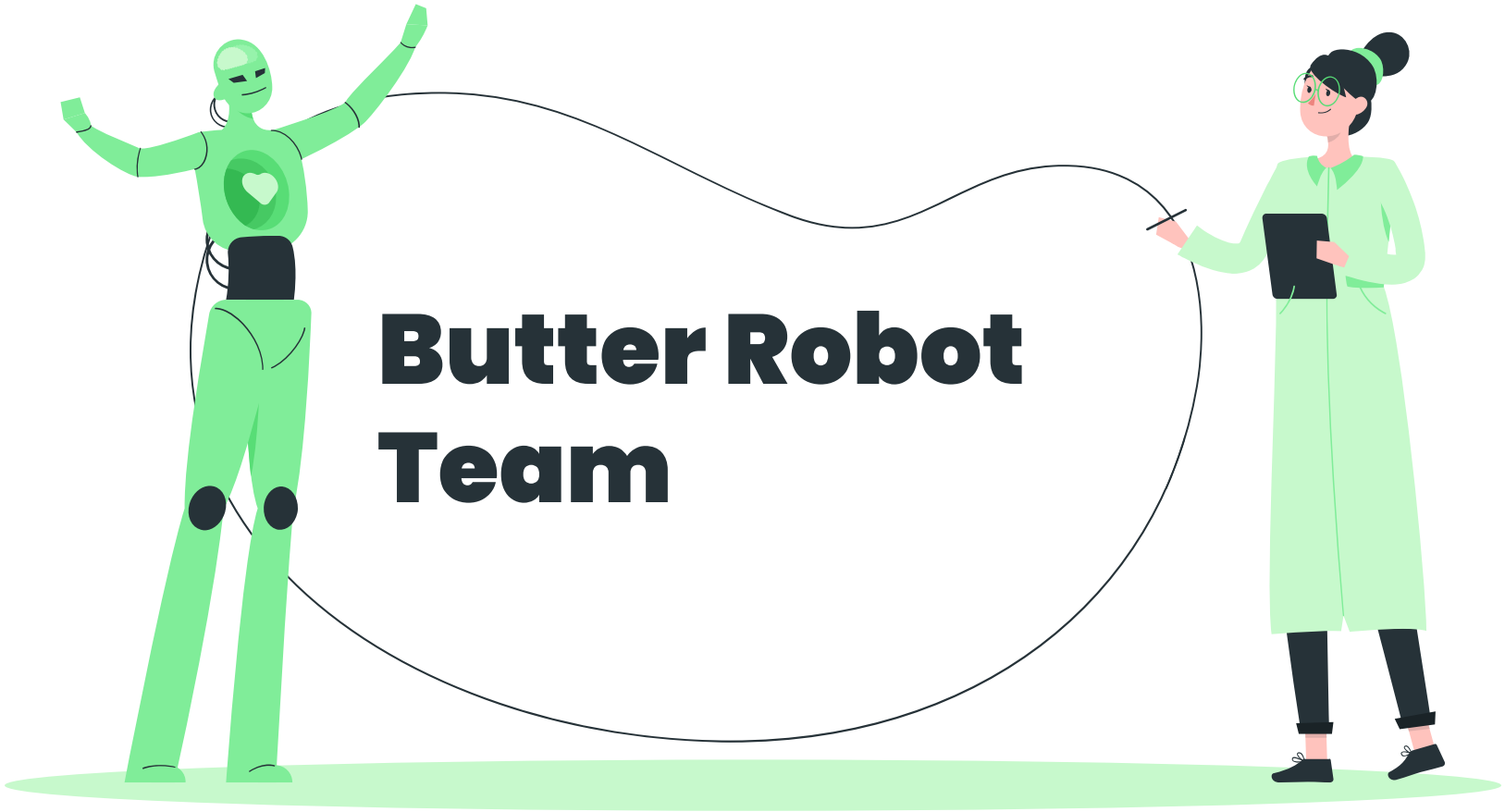
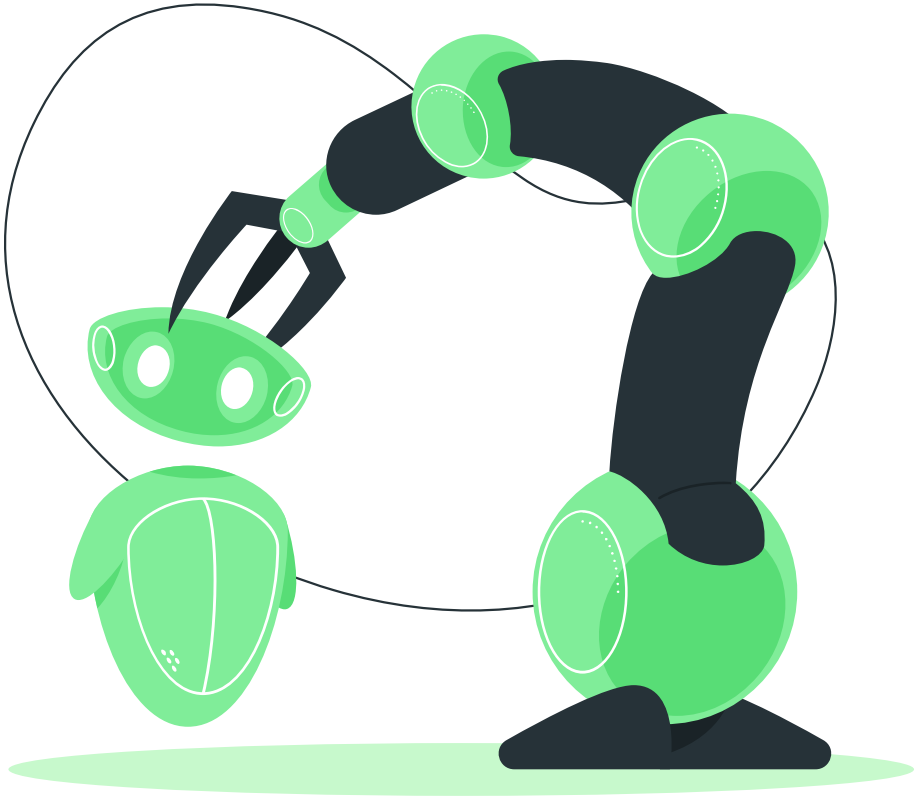


Butter Robot Team

UR Team

Margaret Wang
Sebastien Wah
Luke Chiang
Hunjoo Kim



High Level Plan

1

Detect target bottle

HSV thresholds, Hough Transforms, and Canny Edge Detection

2

Bottle location

Communicate bottle location in UR's coordinate frame

3

Arm Control

Grab and pick up bottle from the top to avoid collisions

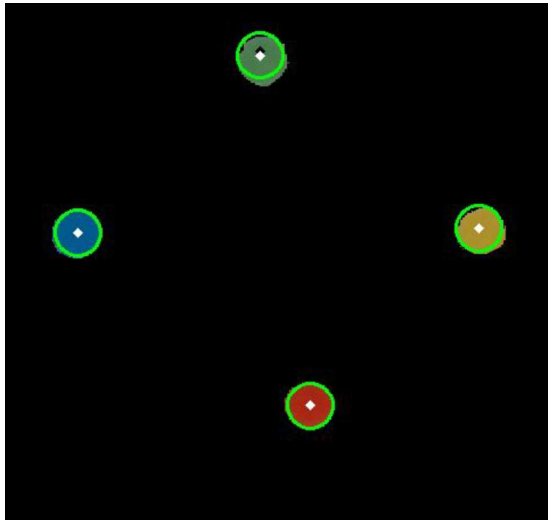
4

Basket Drop off

Drop off bottle at center of desired basket location to maximize target area

Code Execution: Bottle Detection

1. Subscribe to camera feed
2. Convert to HSV and make masks
   a. *Choose the bottle color desired
3. Perform Hough Transform with Canny Edge Detection
4. Draw circles and obtain the center of the target bottle
5. Convert center from pixels in picture coordinate frame to meters in UR coordinate frame
6. Publish target bottle's center coordinates



*Extension: Voice to text feature

Voice Detection

Dependencies: Speech Recognition / PyAudio / pyttsx3

1. Initialize the speech recognizer and begin loop
2. Wait to calibrate to ambient noise level
3. Prompt the user and wait for user input (color)
4. Process audio and convert to text string
5. Check if text is one of the desired colors:
    a. If yes, set booleans for the colored bottles
    b. If no, go back to top of loop and reprompt

Code Execution: Moveit

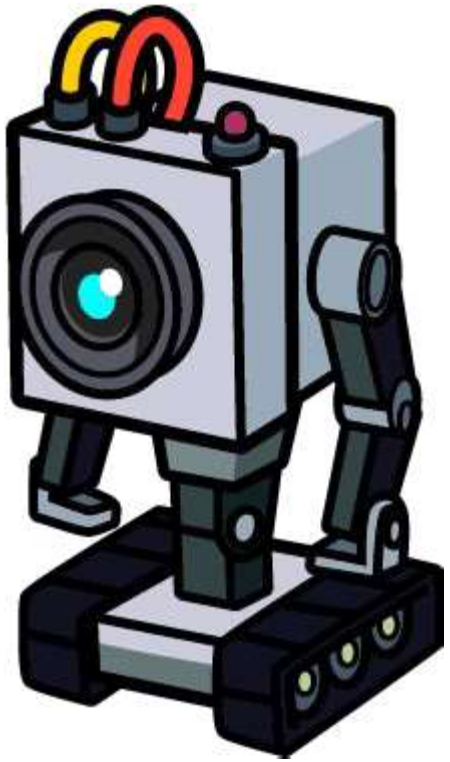
- Subscribe to /ur_coords topic from the bottle detection python code
- Initialize the robot so the end-effector is pointing downwards, and z-height is high enough to clear all bottles
- Plan cartesian path to the x,y, coordinates, such that robot moves one axis at a time
- Move to above the target bottle
- Lower the end-effector straight down in z-axis
- Grab the bottle
- Move up the end-effector straight up in z-axis, high enough to clear all bottles
- Subscribe to /docker_status
- Plan and execute cartesian path to the mobile robot docking location
- Drop the bottle
- Reset to initial position using joint state

Victory Dance

- Spin the end effector around using 4 different joint position for the end-effector rotation actuator
- Move back and forth between two joint states to celebrate success!

MR Team

Ines Pinilla
Sebastian Uribe
Erik Thompson

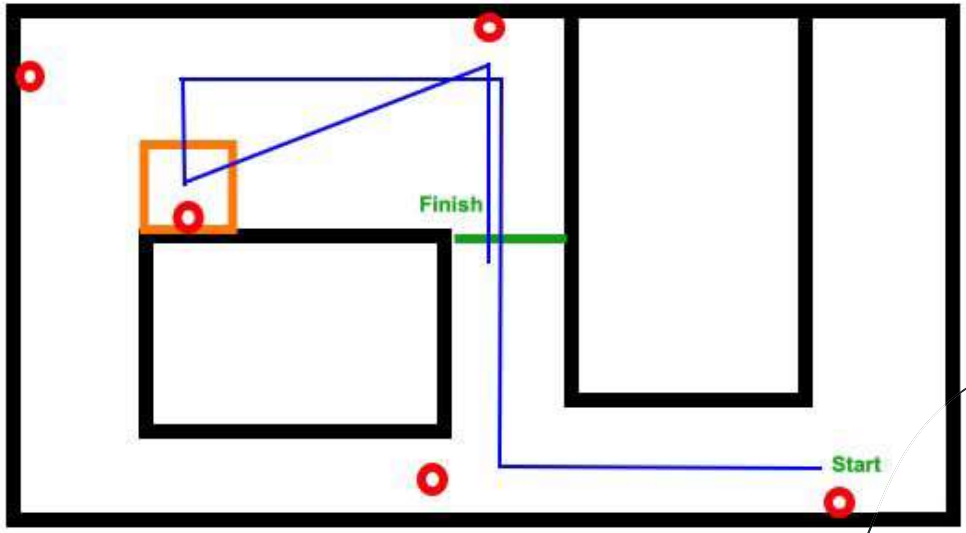


High Level Plan

- Continuously detect April Tags
- Communicate tag location in MR’s coordinate frame
- Approach tag within a certain distance
- Dock at the last tag, send confirmation and position to UR robot
- Communicate with UR to indicate start of end sequence

Hospital Map

red circles = tags
orange box = docking cube
blue line = MR robot path



Coding Challenges

1

Movement

Continually detect tags and move between them

2

Docking

Stop at the correct orientation and position

3

UR Comms.

Communicate end of forward run, receive end of UR process to start second run

Coding Execution: Movement

Using ROS, April Tags, and Joy_Cmd() to publish movement commands

- Detecting April Tags through mounted camera, spinning if not found
- Publish Joy_CMD() commands to move the MR robot
  - Specify pose: x, y, z
  - Specify orientation: rotation on x, y, z axis
- Approach given tag at specified distance
- Look for next tag

Coding Execution: Docking

Using final tag (5) as reference for docking

- Approach tag (4) but only until even with docking cube
- Find and approach tag (5), coming as close as possible
- Send Joy_Cmd() sleep command to wait for UR

Coding Execution: Communication

Using ROS Publisher and Subscriber

- Common topic between UR and MR robots, “/docker_status” topic
- At finish of first MR sequence, publishes message confirming docking
- Waits for the subscriber to signal end of UR process
- Begins final movement sequence to finish line

Thank You!

We’d like to acknowledge all of the support from the 2.12 staff, thank you for your help throughout this process!

Ready to Launch!

End of presentation, prepare for MR and UR demonstration.