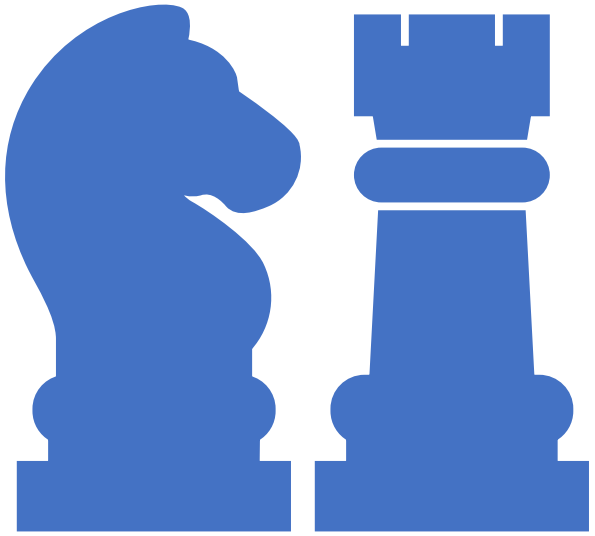
Robot Arm Project: Task Planning

Emir Kocak – Zhiyu Zheng (Jerry)



Problem formulation

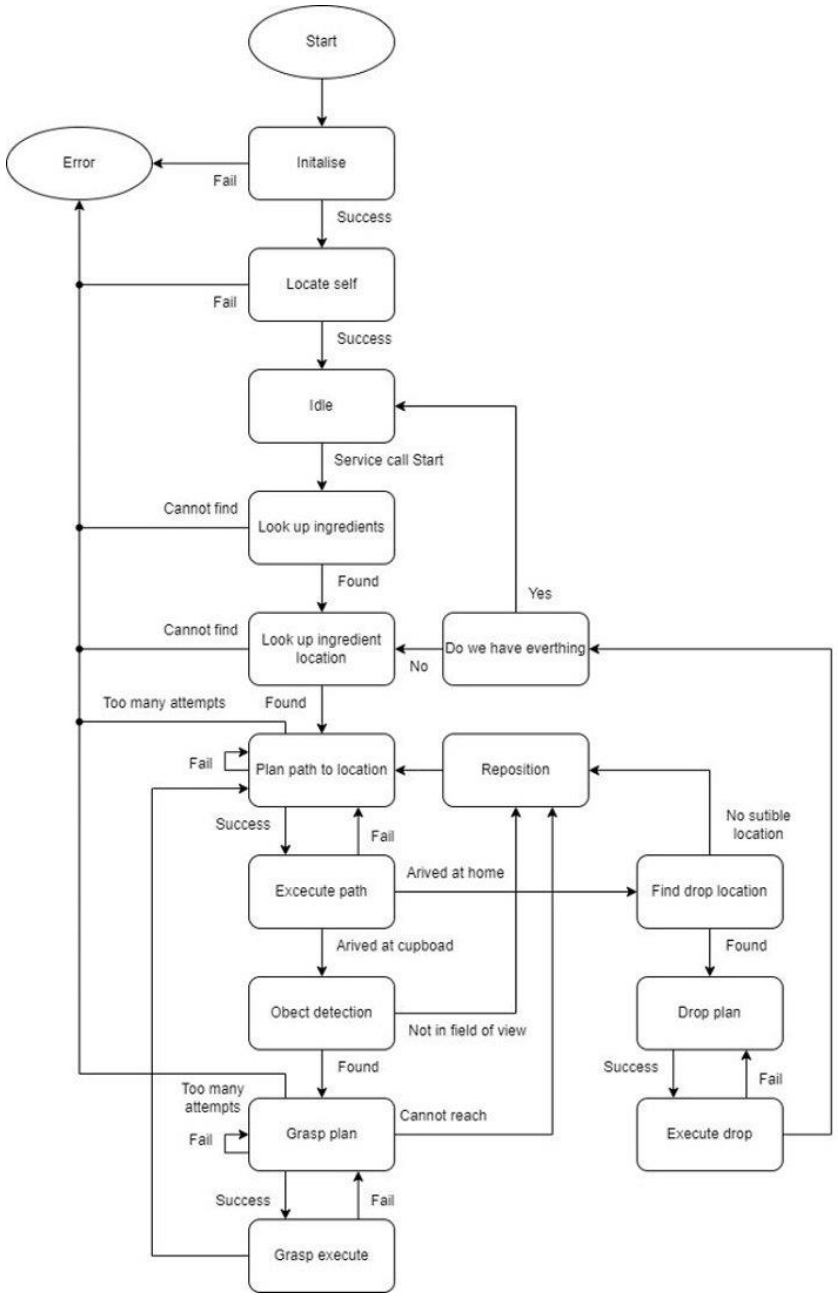
- Defining states and transitions for the robot decision making process
- States represent specific phases in robot operations (e.g. select chess move, grab and move chess pieces)
- Transitions between states happen when certain conditions are met.
- Goal: coordinate all the other robot's functional nodes together and direct the robot’s behaviour in different scenarios.

Strategy

```
if __name__ == '__main__':
    rospy.init_node("State_Machine_1.0")

    sm = smach.StateMachine(outcomes=['success','fail'])

    # Open the container
    with sm:
        # Add states to the container
        # Provide the name of the state, the class from which it is derived,
        # and the next state for each possible outcome
        smach.StateMachine.add('Idle', Idle(),
                               transitions={'Start':'GetPose', 'Stay Idle':'Idle'})
        smach.StateMachine.add('GetPose', GetPose(),
                               transitions={'success':'GetPose'},
                               remapping = {'pose' : 'pose'})
        smach.StateMachine.add('MoveArm',MOVE_ARM(),
                               transitions= {'success' : 'success',
                                             'failure' : 'fail'})

    # Execute SMACH plan
    outcome = sm.execute()
```

- Architecture: Finite state machine (FSM)
- Software packages:
  - SMACH – build the abstract FSM
  - ROS – implement and facilitate the state actions on the robot and handles information flow

Example State Machine



What will we work on?

Design

Task Analysis:

Analysis of the chess-playing task, breaking it down into individual steps.

State Machine Design:

Creation of a state machine diagram that outlines all the possible states and transitions for task completion.

Designing a sequence of states for simple pick and place tasks as a foundation.

Complex Task Design:

Designing error handling and state recovery mechanisms.

Implementation

Framework Development:

Implementing the basic state machine structure in SMACH based on the design.

Conducting initial tests for state transitions and data flow.

Simple Task Implementation:

Implementing the designed sequence for a simple pick and place task.

Testing and establishing communication with perception and motion planning nodes.

Complex Task Implementation:

Incorporating complex task sequences with robust error handling