HW 9

Team members : Jacob Moore & Hermena Mikhael

jhmoore3@mail.lipscomb.edu
hlmikael@mail.lipscomb.edu

Overall team contributions:

Pseudo code: Jacob Moore

Debugging and testing: Hermena Mikhael

Youtube link: https://youtu.be/ySa_QnMSGzI

Github link: https://github.com/HermenaMikhael/matlab_ros_support/tree/2b5f10c83dbb3beb377e7526d5ab4044e5c87fc0/hw/hw09_pick_and_place/hw09_pick_and_place

File 1: convert2ROSPointVec.m

Contributions:
Coding- Hermena Mikhael

Pseudo-code

1. Get robot handle from options
2. Calculate time step = duration / steps
3. Filter joint names
4. Set trajectory joint names
5. For each waypoint:
   - Create new trajectory point message
   - Set positions from trajectory matrix row
   - Set velocities, accelerations, effort to zero
   - Set time from start
   - Store point in array
6. Assign all points to trajectory goal
7. Return trajectory goal

File 2: convertPoseTraj2JointTraj.m

Contributions:
Coding- Hermena Mikhael

Pseudo-code

1. Get robot handle from options
2. Determine number of trajectory points
3. Initialize joint trajectory matrix (n x 6)
4. Get current joint states for initial guess
5. Check if joint angles are valid
6. For each waypoint:
   - Call IK solver with desired pose and initial guess
   - Check if solution is valid
   - Print IK solution
   - Store joint angles in trajectory matrix
   - Update initial guess with current joint states
7. Return joint trajectory and joint names

File 3: doGrip.m

Contributions:
Coding- Jacob Moore

Pseudo-code

1. Get robot handle from options
2. Create gripper goal message
3. Clear feedback and result functions
4. Set grip position
5. If type is 'place', set position to 0
6. Pack grip goal using packGripGoal_struct
7. Print status message
8. Try to connect to gripper server
9. Send goal and wait for result
10. If fails, retry once
11. Return result and state

File 4: packGripGoal_struct.m

Contributions:
Coding- Jacob Moore

Pseudo-code

1. Get robot handle from options
2. Set waypoint position to input position
3. Set joint name to 'robotiq_85_left_knuckle_joint'
4. Set time from start to 1 second
5. Set trajectory point positions to desired position
6. Set velocities to zero
7. Set accelerations to zero
8. Copy trajectory point to goal message
9. Return populated grip goal

File 5: pick.m

Contributions:
Coding- Hermena Mikhael

Pseudo-code

1. If objectData is a 4x4 matrix
        mat_R_T_M is equal to objectData
        Set label equal to can
   Else
        Extract label and pose from objectData
2. Assign zOffset and doGripValue based on the label type
3. Create a new matrix called over_R_T_M1 using lift to add 10 cm and the zoffset to the extracted matrix
4. Move to over_R_T_M1 and display it.
5. Create a new matrix called over_R_T_M using lift to add the zoffset to the extracted matrix
6. Move to over_R_T_M and display it
7. Get the grip_result and grip_state from calling doGrip to pick the object
8. Return the grip result error code

File 6: mat2rosJoints.m

Contributions:
Coding- Dr. Rojas

Pseudo-code

1. Reorders the joint input to match ros joints