MAE204 Robotics

Final Report

March 21, 2024

Kaiyu Liu

Summary

This final project is to write a matlab program which enables a 5R-youBot composed of a mobile chassis with four wheels and a 5R arm picking up, carrying, and placing a cube at designed locations in CoppeliaSim software.

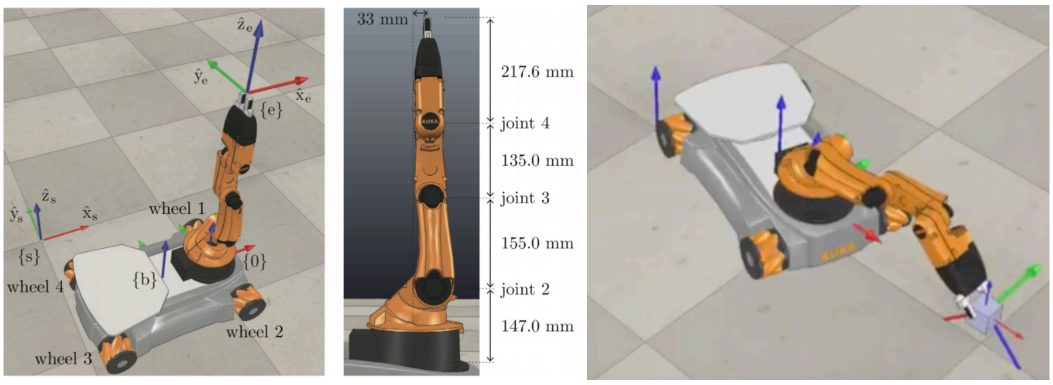


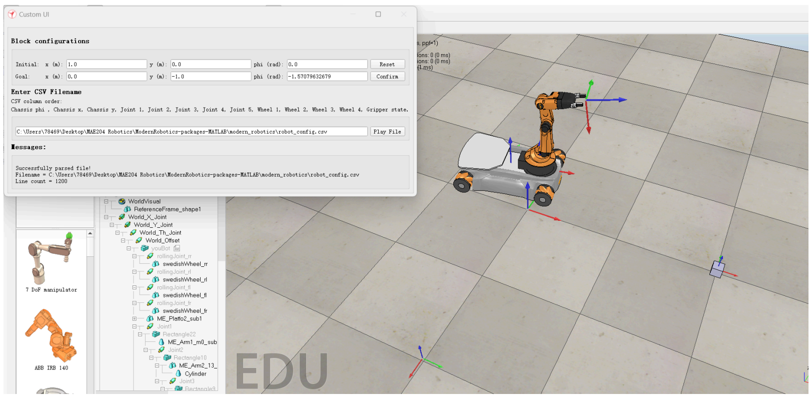
Figure 1. youBot configuration and picking the cube [1]

Procedure of finishing the simulation task:

1. Plan a trajectory for end-efffor including eight segments;
2. Calculate the kinematics of youBot;
3. Use feedforward plus feedback control law to calculate kinematic task-space;

$$V(t) = [Ad_{X^{-1}X_d}]V_d(t) + K_p X_{err}(t) + K_i \int_0^t X_{err}(t)dt$$

4. Save the .csv file and finish the simulation in CoppeliaSim.


Block configurations

Initial: x (m): 1.0 y (m): 0.0 phi (rad): 0.0

Goal: x (m): 0.0 y (m): -1.0 phi (rad): -1.57079632679

Enter CSV Filename

CSV column order:

Chassis phi , Chassis x, Chassis y, Joint 1, Joint 2, Joint 3, Joint 4, Joint 5, Wheel 1, Wheel 2, Wheel 3, Wheel 4, Gripper state.

C:\Users\78469\Desktop\MAE204 Robotics\ModernRobotics-packages-MATLAB\modern_robotics\robot_config.csv

Messages:

```
Successfully parsed file!
Filename = C:\Users\78469\Desktop\MAE204 Robotics\ModernRobotics-packages-MATLAB\modern_robotics\robot_config.csv
Line count = 1200
```

Figure 2. Simulation Screen

During the experiment, it has been found that when the maximum joint & wheel velocity magnitude gets smaller, it will have fluctuations in error once after convergence. Besides, an integral has been added in the FeedforwardControl section to achieve fine control.

Results

Initial robot configuration of three cases is:

robot_config = [0.6, -0.2, 0.2, 0, 0, 0.2, -1.6, 0, 0, 0, 0, 0, 0];

It has been set to at least 30 degrees of orientation error and 0.2m of position error from zero configuration.

1. “Best (kp = 6, ki = 0)”

https://youtu.be/fVMnUM9llgI

This solves the pick & place mission in a minor error at the default position, the simulation is relatively smooth with the feedforward + P control. As shown in figure 3, the error curve converges quickly around 1s when kp=6 and no need to add ki. The final output is nearly zero and there is no obvious fluctuation. In the simulation, the entire motion is continuous, but there is a very small shake that happens in grasping the cube. I think the reason may be the physical model, which is caused by this simulation software.

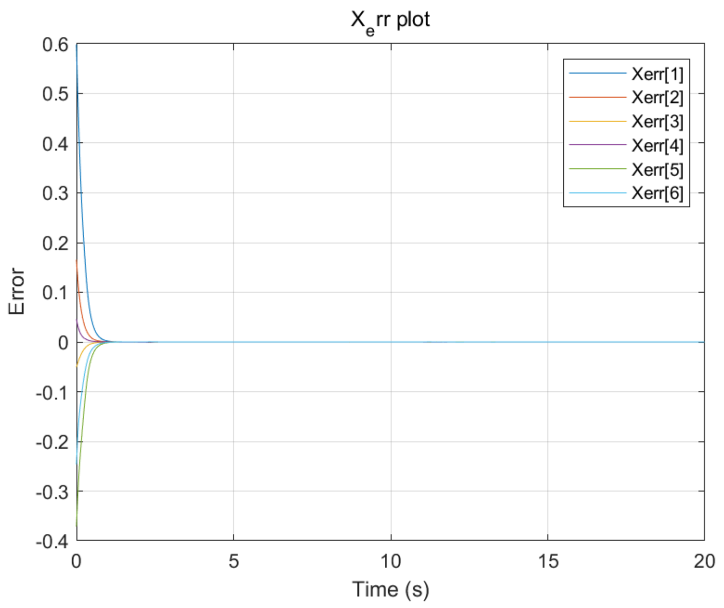


Figure 3. Error plot over time of “Best”

2. “Overshoot (kp=6, ki=12)”

https://youtu.be/itH0SBxjir8

This solves the same task as “Best”, but error gets larger and there is obvious overshoot at the beginning of the task. A less-well-tuned feedforward + PI controller has been used in this situation with kp=6 and ki=12. However, error converges after 4s and the entire simulation becomes smooth. In CoppeliaSim, we can see that the robot will pass the original point and go back to zero in the start, and smoothly finish the rest of the work without error. Overshoot will lead to attitude deviation of the robot arm, which will affect the stability.

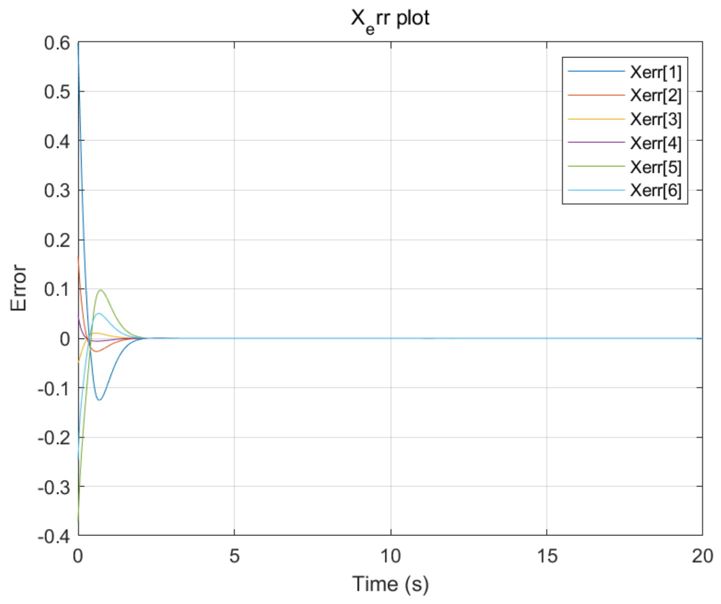


Figure 4. Error plot over time of “Overshoot”

3. “New Task (kp=5, ki=0.05)”
https://youtu.be/zBAOzK0EwS8

Initial and final configuration of the cube are:
Tsc_ini = [0, -1, 0, 1; 1, 0, 0, 1.0; 0, 0, 1, 0.025; 0, 0, 0, 1];
Tsc_fin = [1, 0, 0, 2; 0, 1, 0, 0; 0, 0, 1, 0.025; 0, 0, 0, 1];

This solves a pick-place task for a cube with a different position (1 unit more in y direction for initial position and 1 unit more in x direction for final position compared to default cube initial position). The work is done continuously and without collision, and the feedforward + PI control is used to adjust system outputs. The integrator controller is added to make the new task more smooth, and the error plot converges more quickly.

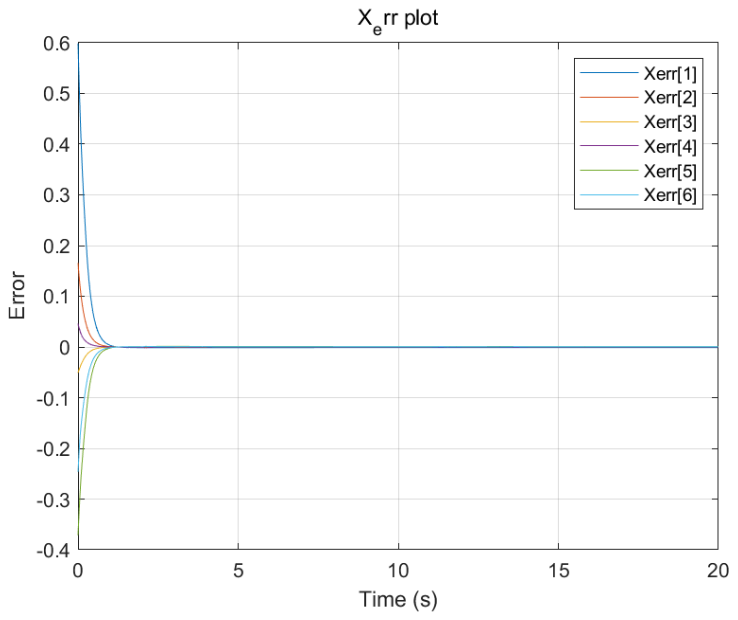


Figure 5. Error plot over time of “New Task”

In the above first two conditions, it can be seen that a very inconspicuous fluctuation around 12s occurs. After checking the matlab code, I found it might be caused by joint and wheel velocity limits (named in max_omg). When this number is set to 10, the fluctuation becomes larger than it is set to 20 (above scenarios are set to 20).

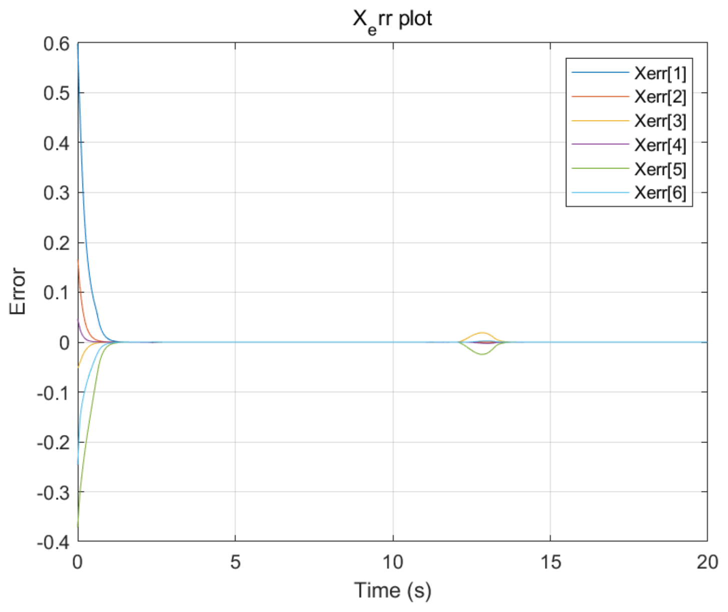


Figure 6. Error plot under a condition of max_omg=10 and kp=6 & ki=0

Discussion

1. Advantages: the integral controller can help to reduce steady-state errors because it could integrate the error over time and achieve fine control which could improve the stability of the system.
Disadvantages: the integrator could lead to oscillations if not tuned properly, and it may cause a longer setting time. Besides, this might result in overshoot effects due to the saturation constraints in the system.
When the error is very small, the addition of the integral controller will produce overshoot.

2. It is easy to have errors when picking up the cube, if I reduce the maximum velocity to a low value, error can't compensate for the difference in speed because large error results in large instantaneous velocity.

3. If the robot has highly accurate actuators that could achieve the desired joint velocities, low friction in each joint and precise control system (sensors, feedback algorithms), the controller directly prescribes the joint velocity could be possible.

4. The SimulateControl, ForwardDynamics, ComputedTorque, EulerStep, JointTrajectory in MR would be used, these functions compute the torque required to achieve desired joint trajectories. The desired joint angles, joint velocities, joint accelerations are inputs, screw axes of joint in a space frame, spatial inertia of links, link frames at initial position, gravity vector, initial joint angle and velocity, and the wrenches applied by end-effector need to be known.

Reference

[1]https://hades.mech.northwestern.edu/index.php/Mobile_Manipulation_Capstone#Milestone_1:_youBot_Kinematics_Simulator_and_csv_Output