Homework 2 - ECE4560

Brett Erickson

use\_pkg\_here('robotics'); %Helper function written by ChatGPT, for file location resolution only.

import robotics.\*

**1. 4 hours** - primarily spent troubleshooting my package

**2.**

a)

l1 = 4;

l2 = 2;

theta\_1 = pi/4;

theta\_2 = pi/2;

b.x = l1 \* cos(-theta\_1);

b.y = l1 \* -sin(-theta\_1);

b.theta = -theta\_1 + -theta\_2;

e.x = b.x + -l2\*cos(-b.theta);

e.y = b.y + l2\*sin(b.theta);

e.theta = b.theta;

fprintf("Effector configuration is g\_we = ([%.2f, %.2f], R(%.2f rad)",e.x,e.y,e.theta);

Effector configuration is g\_we = ([4.24, 1.41], R(-2.36 rad)

b)

g\_wa = transform2d([0; 0], -theta\_1);

g\_ab = transform2d([0; l1], -theta\_2);

g\_be = transform2d([0; l2], 0);

g\_we = compose(compose(g\_wa,g\_ab),g\_be);

fprintf("Effector configuration is g\_we = ([%.2f, %.2f], R(%.2f rad)",g\_we.displacement(1),g\_we.displacement(2),g\_we.theta);

Effector configuration is g\_we = ([4.24, 1.41], R(-2.36 rad)

3.

function outMat = homogeneousMatrix(x,y,theta)

outMat = [cos(theta) -sin(theta) x; sin(theta) cos(theta) y; 0 0 1];

end

function outMat = homogeneousInverse(outMat)

rotation\_transpose = transpose(outMat(1:2,1:2));

outMat = [rotation\_transpose, -rotation\_transpose\*outMat(1:2,3); 0 0 1];

end

bbprime = homogeneousMatrix(10,1,-pi/4);

h = homogeneousMatrix(0.1,1,-pi/8);

h\_inverse = homogeneousInverse(h);

ccprime = h\_inverse\*bbprime\*h;

fprintf("Transform CC' = ([%.2f, %.2f], R(%.2f rad)",ccprime(1,3),ccprime(2,3),atan2(ccprime(2,1),ccprime(1,1)));

Transform CC' = ([9.62, 4.67], R(-0.79 rad)

4.

g\_wr = homogeneousMatrix(2,3,pi/4);

g\_ro = homogeneousMatrix(3,1,-pi/3);

g\_wo = g\_wr\*g\_ro;

fprintf("Object with respect to worldframe = ([%.2f, %.2f], R(%.2f rad)",g\_wo(1,3),g\_wo(2,3),atan2(g\_wo(2,1),g\_wo(1,1)));

Object with respect to worldframe = ([3.41, 5.83], R(-0.26 rad)

5.

Since I built my own SO-101 arm (named Jack the Gripper, henceforth refered to as Jack), I had to do all of the steps from construction to motor identification.

I started off by attempting to work with conda, but I couldn't get it to work and didn't want to maintain a virtual environment from the command line. So instead I created a github repo and installed uv (the package and environment manager), then began the installation of lerobot and its dependencies. My intention was to make the repo a one-shot solution for other students to pull and code with the SO-101.

It does work, and I hope that in the future my team will work within that repo to collaborate on code, ensuring we all are operating on common dependencies and a standardized virtual envirionment.

<https://github.com/berickon3/ECE-4560_Lab.git>

Once that was done, I did the motor ID, calibrated the arm, calibrated a follower from the lab, then performed the teleoperation. I noted that even a small discrepancy in the calibration (not going to the full extention of a joint) would cause that joint to have a ratio of movement to the leader not 1:1.



### APPENDIX A:

This code was generated by ChatGPT. I got very tired of working to resolve relative and absolute filepaths to find my robotics package, where each homework.mlx has its own folder. The complication is that I want this repo to be available to pull and run anywhere, agnostic of file install location. 'import' no longer suports relative filepaths, so thats out.

I'm tired, boss.

function use\_pkg\_here(pkg)

% Make the parent of the current live script the CWD temporarily so +pkg is visible.

% Robust against Live Editor's temp execution folder.

% Resolve the source file of the active Live Script

src = "";

try

src = string(matlab.desktop.editor.getActiveFilename);

catch

end

if src=="" || ~isfile(src)

st = dbstack('-completenames');

if ~isempty(st), src = string(st(end).file); end

end

assert(src~="" && isfile(src), 'Unable to resolve the source .mlx path.');

% Repo root is the parent of the folder containing this .mlx

% .../Robotics-Code/homework2/homework\_2.mlx

% -> .../Robotics-Code

thisDir = fileparts(src);

repoRoot = fileparts(thisDir);

% Verify +pkg exists at repo root

pkgDir = fullfile(repoRoot, ['+' char(pkg)]);

assert(isfolder(pkgDir), 'Expected package folder not found: %s', pkgDir);

% Temporarily cd into repo root so +pkg is discoverable; auto-restore

orig = pwd;

cleaner = onCleanup(@() cd(orig));

cd(repoRoot);

% Nothing else to do here; the caller can now 'import pkg.\*'

end