# EECS4421: Lab 2

Thu Jan 19 & 26, 2017
Due: Thu Feb 2, 2017

## 1 Preliminaries

This lab teaches you the basics of operating the CRS A150 robotic arm, an industrial robot manufactured in the early 1990s.

### 1.1 Safety

The A150 and A255 robotic arms can seriously injure you if you do not follow reasonable safety precautions:

- Always be ready to press the big red kill switch! Do not let the robot crash into someone, something, or the table!
- Keep the tables the robots are mounted on clear of extraneous objects (coats, books, binders, etc.).
- Do not put your head (especially if you have long hair) within the range of any of the robots.
- Never use a speed of higher than 30% of the maximum (in fact during the debugging use much less that 30%)
- Do not power on a robot unless you are logged in to the workstation connected to the robot. If you are not logged in, it is possible for someone else to remotely log in and control the robot!
- Do not remotely log in to the workstations connected to the robots.
- Avoid working in the lab alone.
- If a robot malfunctions, turn off the power and immediately inform the technical staff (`tech at cse.yorku.ca`) and the instructor (`burton at cse.yorku.ca`) in person during working hours and by email otherwise.

## 2 Instructions to Power the A150 Arm

1. Log in to the workstation connected to the arm.
2. Start a console; in the console start the minicom program by entering the command `minicom`.
3. Power on the robot by turning the clear power switch in the upper-right corner of the control panel clockwise. The switch should turn orange, the control panel should hum, and minicom should display some information.
4. Check the big red kill switch (kills power to the arm motors). If it is pressed in, unlock it by turning it counterclockwise.
5. Manually align the robot. There are 5 sets of markers that you must align by moving the robot by hand (there is no power to the arm motors yet). The robot will be limp, so you will need to support the links of the robot by hand. All 5 sets of markers must remain aligned before you proceed to the next step.
6. Have someone else press the arm power button (located beside the big red kill switch). The button should turn orange and the robot should support itself. There is now power to the robot motors.
7. Tell the robot to home itself by entering the command `HO` in minicom. The message `LOCATED IN ITS HOME BOUNDS?` will appear. Type `y` followed by enter. The system shell has the annoying feature of auto-completing typed commands; the actual command name is `HOME`.
8. The robot will now try to set each of its 5 axes into a pre-calibrated home position. A message will appear for each axis in minicom; the message `TEST PASSED` should appear for each axis (even if you did a poor job in the previous step!); make sure the robot is in the correct home position.
9. If the robot is not correctly homed, support the robot by hand and press the kill switch. Repeat steps 3–7 to re-home the robot.
10. Once the robot is correctly homed it is ready for use. Enter the command `READ` in minicom and quit minicom using `Ctrl-A X`.

Note that the Matlab controller code has a home function; thus, you can avoid the use of minicom. However, minicom is very useful for trying out the many commands supported by the robot (and documented in the large black binder).

## 3 Instructions to Turn Off the A150 Arm

1. Support the robot by hand and turn the main power switch to the off position. Gently allow the arm to come to rest on the foam.
2. Log out of the workstation.

# 4 Robot Geometry

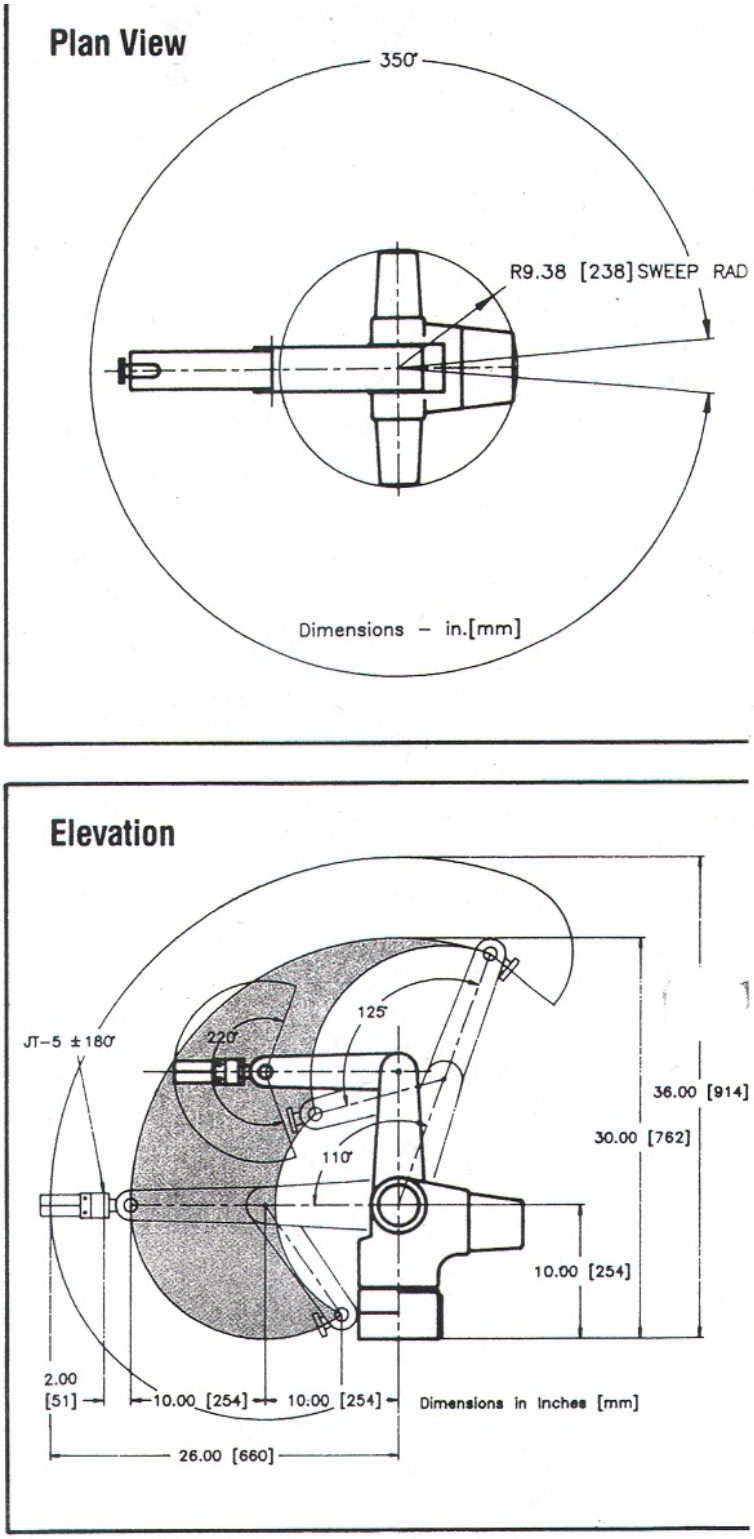



Figure 1: Dimensions of the A150 robotic arm *in inches and [mm]*.

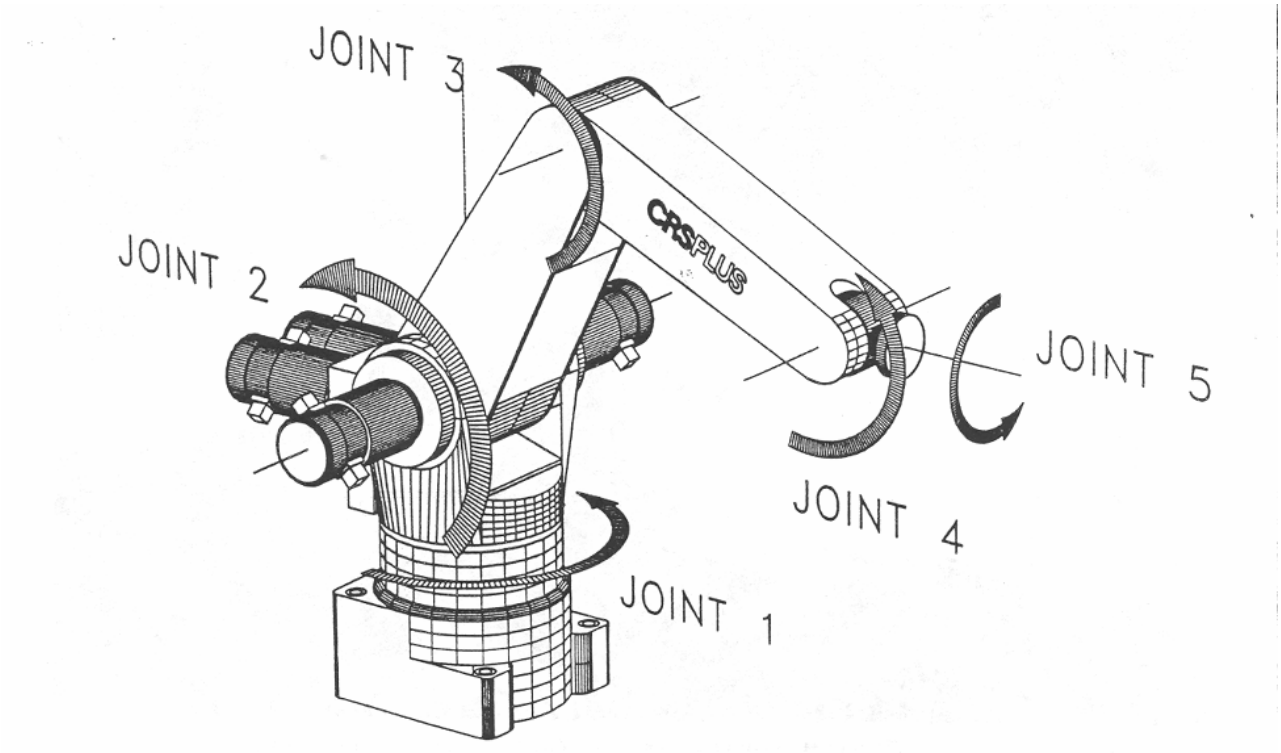



Figure 2: The five revolute joints of the A150 robotic arm.

The range of angles for each joint are shown in the table below.

| joint | joint name | angle range |
|---|---|---|
| 1 | waist | -175–175 |
| 2 | shoulder | 0–110 |
| 3 | elbow | -125–0 |
| 4 | wrist | -110–110 |
| 5 | twist | -180–180 |

The ranges are specified relative the previous link.

When using the robot's native API, the joint angles for joints 2, 3, and 4 are not measured relative the previous link; they are measured relative to the horizon.

# 5 Matlab Source Code

Both a Matlab simulator and Matlab controller are available for the A150 arm. Retrieve the Matlab sources from the "Source Code" section of the course web site.

## 5.1 Using the Simulator Code

Unzip the Matlab source file into a directory somewhere under your EECS account. The Matlab simulator is similar to the controller except that you use the class name `sim150` instead of `a150` to create a simulated instance of the robot. You can run the simulator on any workstation in the labs.
There is a small demonstration program `demosim150.m` that uses the simulator. Read through the demonstration program, paying careful attention to the comments, and run each line of uncommented code one at a time to see what the robot does.

## 5.2 Using the Matlab Controller Code

To use the controller, you must be logged into the workstation attached to the A150 robot (remote login is not sufficient to access the robot). Start Matlab in the directory (or in Matlab navigate to the source directory).
There is a small demonstration program `demo150.m` that uses the controller.

# 6 The Programming Task

First, carefully read this document. Next, study the APIs of the simulator classes and run the `demosim150.m` script; this script uses the Matlab A150 simulator to move the arm through a variety of poses.
Create a new simulator demonstration that moves the simulated arm in the following manner (starting from the home position):

1. Points the arm approximately straight up.
2. Rotates the wrist (joint 4) through its full range of motion ($\pm$90 degrees) and back to 0 degrees.
3. Rotates the waist (joint 1) 90 degrees clockwise (when looking down on the robot).
4. Rotates the elbow (joint 3) so that the end of the arm is level with the horizon.
5. Rotates the shoulder (joint 2) so that arm is pointing down towards the table at 45 degrees.
6. Rotates the wrist (joint 5) 90 degrees.
7. Closes the gripper.
8. Rotates the shoulder (joint 2) so that the end of the arm is level with the horizon.
9. Rotates the waist so that the end of the arm is pointing in the opposite direction from Step 9.
10. Rotates the shoulder (joint 2) so that arm is pointing down towards the table at 45 degrees.
11. Opens the gripper.
12. Returns the robot to the home position.

You can run the example solution script `sol2.p` The solution pauses 2 seconds between movements of the robot. Show me your simulation once you have it working.

Now create a controller demonstration program in Matlab that reproduces your simulation. *Remember to set the speed of the robot to a value around 10* (the maximum speed is 50). When you are ready, test your program using the actual robot.

# 7 Written Questions

1. Consider the RR arm shown in the figure below.

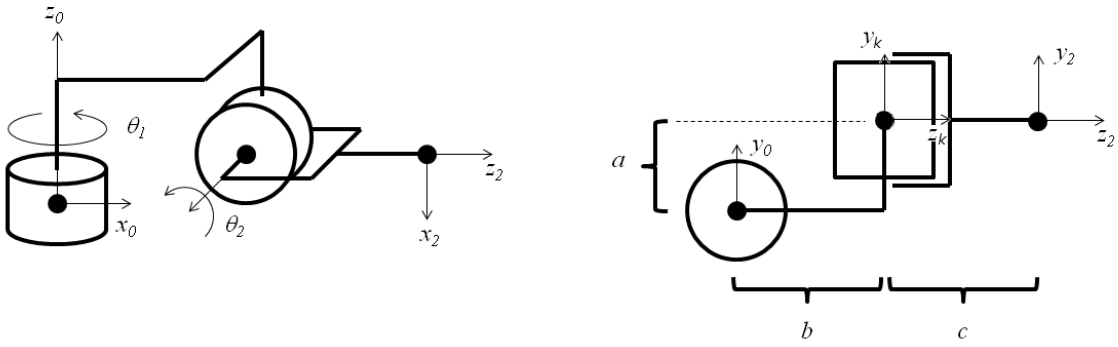


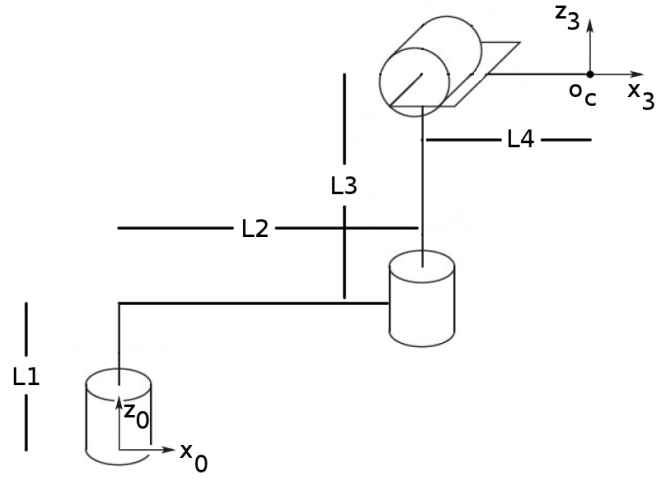

Figure 3: Left: Front view of arm. Right: Top-down view of arm. In this figure, all joint angles are shown at $0°$.

Without using the Denavit-Hartenberg convention, give an expression for the elements of $T_2^0$ when the robot has joint angles $\theta_1$ and $\theta_2$. Please show your work; if you use an intermediary frame (and you should), draw a figure showing the placement of the frame on the robot.

2. Using the Denavit-Hartenberg convention, provide a figure showing the placement of frame 1 for the robot in Question 1. Provide the table of Denavit-Hartenberg parameters and compute the resulting Denavit-Hartenberg transformation matrix $T_2^0$. It should be the same as your answer for Question 1.

CORRECTION: For this question, you should insert an intermediary frame $k$ between frame 1 and frame 2 as shown in the figure above. Frame $k$ moves when joint 2 rotates.

3.




The robot above is shown in a position where all of the joint angles are zero degrees.

Draw coordinate frames $\{1\}$ and $\{2\}$ on the robot that are suitable for use with the Denavit-Hartenberg convention,

and provide a table of Denavit-Hartenberg parameters that would allow you to compute $T_3^0$ (but do not compute this matrix).

# 8 Submit

Submit your Matlab script for the simulation using the command:

```
submit 4421 L2 <your script name>
```

or use websubmit `https://webapp.eecs.yorku.ca/submit`