Objectives

- Introduce the TMS320F28335 DSP controller for the CRS robot arm and using Code Composer Studio IDE to program the TMS320F28335 DSP with C.
- Perform forward kinematics analysis for CRS robot arm following the Denavit-Hartenberg (DH) convention
- Practice calculating transformation matrix with Robotica
- Derive inverse kinematic equations for the CRS robot arm
- Use the provided code composer studio project to verify your solution

*NOTE: CRS robot arm comes with five motors, in this lab we are only using the first three.*

**Part 1: forward kinematics**

In this part you will derive solutions to the forward kinematics problem for the CRS robot arm. First find parameters of the CRS robot arm following D-H convention, then verify the theoretical result using a given program in code composer studio.

**Physical Dimension**

Below is the physical dimension of the CRS robot arm used in this lab. Use this when deriving D-H parameters.
Theoretical Solution
Find the forward kinematic equations for the CRS robot. In particular, we are interested only in the position of the end effector. We will use Robotica to find expressions for each of the three components of $d_3$.

Verification
For any provided set of joint angles $\{\theta_1, \theta_2, \theta_3\}$, we want to compare the theoretical position of the end effector given by Robotica, then manually moving the end effector to the given angles, measure the actual position of the end effector.

Procedure
1.1 SVN Check-out
For your lab assignments, all the given code is stored in a SVN repository.
   1. Create a folder under C:\ (using your NetID or whatever), then inside it create two folders named “workspace” and “repo”
   2. Right click on “repo” and select SVN checkout. In the check-out window type in your individual class repository: file:///N:/labs/ME446/ME446Repository and select “OK”. 
1.2 Code Composer Studio (CCS)
In this portion of the lab, you will be modifying a given Code Composer Studio project to read joint angles of your CRS robot arm.

1. Open Code Composer Studio, select the “workspace” folder created in the last step.
2. Click project → Import New CCS Project, click “Browse” then select C:\your\folder\repo\trunk, choose the project named “CRSRobot” and click “finish”.
3. Run the project and power the robot arm according to the procedure for running robot on your desk.
4. After running the CCS project, your TA will show you how use an application called Tera Term to display joint angles from your robot arm (or whatever else you would like to print for debug purposes).

**NOTE:** output of optical encoders will be reset to 0 every time the CCS project is downloaded and run on the DSP (or the code restarted). Make sure your robot arm is at the home position before running your C program.

5. Additionally, we have created some MATLAB functions that can communicate data back and forth between MATLAB and the DSP controller through serial port COM1. If you look at the top of lab.c you will see two global variables, `whattoprint` and `theta1array`. Before these variables there is a #pragma statement that tells the linker to locate these variables in a special memory section that the MATLAB functions can look in and find all the variables MATLAB can either write to or read from. For these functions to work you MUST HAVE THE DSP RUNNING and you MUST CHANGE THE CURRENT DIRECTORY of MATLAB to \c:\<yourcreateddirectory>\<yourrepositorydirectory>\trunk\CRSRobot\matlab. With MATLAB in this directory it can locate the project files it needs to discover the location of these special variables. The three functions you will be using are ME446_serial_ListVars, ME446_serialwrite, ME446_serialread. At MATLAB’s command change directory to this “CRSRobot\matlab” directory in your repository. Check out help on each of these functions by typing “help ME446_serialread” for example. With guidance for your TA, experiment with each of these functions by first playing with the two given variables `whattoprint` and `theta1array`. Then add one more float array to save 100 theta2 values and another float variable that performs something different on the DSP controller when it is changed from 0 to a positive number.

6. For one more initial exercise with the robot arm, determine the positive direction of the robot joint motor’s angle and the positive torque direction of each joint motor. Produce a stick figure picture showing the positive direction of the three joint angles and torques. To do this you will slightly modify the code in the function “lab” in your lab.c file. The function `lab(float thetamotor1, float thetamotor2, float thetamotor3, float *tau1, float *tau2, float *tau3)` is called by the supporting code once every millisecond. It is passed the radian value of each of the three joint angles and references to the three torque commands for the joint motors. Acceptable values for tau1 through tau3 are a real number between -5 and 5. The unit of this -5 to 5 number is directly proportional to torque but as of this writing the scale between this number and N-m has not been found. For this exercise, one at a time change the line of code that assigns an open loop torque to each of the motors and run the robot while watching the theta angles displayed in Tera Term. For example
change the line of code *tau1 = 0.0; to *tau1= 1.0;. Debug your code and perform the
required steps to run the robot arm and you should see joint one turn in its positive
direction until it goes outside of the safety region and then will stop. Repeat for
joint two and three zeroing the other joints so only one joint moves at a time.

1.3 Physical Implementation
1.2.1 Before practicing D-H convention, proper coordinate frames have to be defined. Figure 1.2
gives a sketch of CRS robot arm with all the z axis of joints fixed. Label the rotation
directions of all joint according to the output of your CCS project.

1.2.2 Use right hand rule to determine the direction of all z axis. Then, on figure 1.2, complete
the D-H frame by properly adding x axis.

1.2.3 Finish the D-H table below. Refer to Figure 1.1 for physical dimensions of CRS robot arm.

<table>
<thead>
<tr>
<th>Joint</th>
<th>( a_i )</th>
<th>( \alpha_i )</th>
<th>( d_i )</th>
<th>( \theta_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1.2.4 Show TA your D-H table.

1.4 Theoretical Solution
1.3.1 Work with Robotica in Mathematica to find the forward kinematics of the CRS robot arm.
First, during lab, perform all the steps of the Robotica tutorial to give you an introduction
to using robotica.m. Appendix A has the link to this tutorial along with a link to a more
detailed document that may help you in future labs.

1.3.2 Create a robotica input file for the CRS robot and use Robotica commands that find and
properly displays the forward kinematic equation for the CRS.

1.3.3 These equations will be a function of the thetas defined by the D-H method. The angles of
each of the three motors of the robot link will not necessarily be the same angles as define
by the D-H method but we do know that the D-H thetas can be calculated by a linear
combination of motor angles. The easiest way to find the relationship between theta1, theta2
and theta3 and motor1_theta, motor2_theta, motor3_theta is to move the linkage to different
90 degree joint positions and record in a table the motor theta values and the values of the
D-H thetas. To find the linear combinations for motor thetas create 3 unknowns that can be
solved for using the different 90 degree points. So for example using some constants \( c_1 \), \( c_2 \),
\( c_3 \) set \( \theta_3 = c_1*motor_\text{theta}_2 + c_2*motor_\text{theta}_3 + c_3 \). Create 3 or 4 equations using your
above table values of thetas and motor_thetas and solve for the \( c_1 \), \( c_2 \) and \( c_3 \) constants. You
can do this using Matlab or it may be possible for you to just look at three or four equations
and figure out the values of \( c_1 \), \( c_2 \) and \( c_3 \).
1.3.4 Now that you have equations for the D-H thetas in terms of the motor thetas, substitute into your robotics parameter file these equations for theta1, theta2 and theta3 given motor_theta1 2 and 3. Rerun the robotics commands and then you will have the forward kinematic equations for the robot using the motor angles.

1.3.5 **Show TA your equations.**

### 1.5 Verification

1.4.1 For demonstration your TA will ask you to move your robot arm joints to two sets of joint angles \{\theta_1, \theta_2, \theta_3\}.

1.4.2 Measure then compare the position of the end effector and the output of your Robotica.

![Figure 1.2 CRS robot (DH frames to be assigned)](image)
Part 2: inverse kinematics

Geometric Approach

Given a desired point in space \((x, y, z)\), write three mathematical expressions that yield values for each of the joint variables. For the CRS robot, there are (in general) two solutions to the inverse kinematics problem. We will implement only the elbow-up solution.

Verification

For any provided point in space \((x, y, z)\), we want to compare the joint angles of the robot arm indicated by your inverse kinematics equations, to what your CCS program gives after manually moving the end effector to the specified point.

Procedure

2.1 Theoretical Solution

2.1.1 Establish the world coordinate frame (frame \(w\)) centered at the center of the CRS’s base. The \(x_w\) and \(y_w\) plane should correspond to the surface of the table, with the \(x_w\) axis straight ahead of the robot arm with \(\theta_1\) equal to zero. Axis \(z_w\) should be normal to the table surface, with up being the positive \(z_w\) direction and the surface of the table corresponding to \(z_w = 0\).

2.1.2 Given a set of \((x, y, z)\) coordinate, solve for the corresponding joint variables \(\{\theta_1, \theta_2, \theta_3\}\) in geometric approach. Write down the three mathematical expressions:

\[
\theta_1(x_w, y_w, z_w) = ?
\]

\[
\theta_2(x_w, y_w, z_w) = ?
\]

\[
\theta_3(x_w, y_w, z_w) = ?
\]

2.1.3 You now have inverse kinematic equations to give you your defined D-H \(\theta\)’s. But in the control of the robot arm we will be controlling the individual motors of the robot arm. Use the equations you found in the forward kinematic calculations that equate the D-H \(\theta\)’s to the motor \(\theta\) positions to rewrite the equations to solve for \(\theta_{M1}, \theta_{M2}\) and \(\theta_{M3}\) given \(x,y,z\).

2.2 Verification

For demonstration TA will supply two point with coordinate \((x, y, z)\). Run the code composer program and move the robot arm to those positions, then compare the output of Tera Term to the joint angles calculated from your theoretical solution.

Report:

Produce a lab report that details all the math used to calculate forward and inverse kinematics for the first three joints of the CRS robot arm. Also detail steps taken to find the relationships between motor angles and D-H angles and note the positive torque direction of the joint motors. Pictures and sketches are a very important part in explaining your work and analysis. Hand sketches will be accepted ONLY if they are very neat. Computer aided drawings are preferred. Also add to your report the COMMENTED code you wrote in Lab1.c
(or whatever you call your file). This report should have enough detail in it that a new student in this class would be taught how to perform the different tasks in this lab assignment.

Appendix A:

Small Robotica Manual/Tutorial
Full Robotica Manual