INF3480 - spring 2010

Compulsory exercise 3

Deadline: Monday, May 3rd (before midnight)

Introduction

In this exercise we will keep working on the X2 robot. You should know it well by now, so we refer to the previous exercises for details on the robot.

We have developed a simulation of the robot. This is included with the download of this paper. You will use the model when you are working with this exercise, and after the submission we will try out the matlab scripts on real robots. We will be using the X2 robot (that you have been working with so far) and the industrial IA20 robot that was shown in the lecture on April 22nd.

The exercises in this assignment will mostly involve path generation for the robot. The robot should be programmed to draw certain shapes on a board, and for this to work it needs to receive a stream of positions along a trajectory. This is what you will program in matlab, test out on the simulation and later transfer to the real robots. The animation works with any angle, but the real robot will only accept angles between -180 and 180 degrees, so you should limit your angles to this range.

Simulation

The simulation is developed in Max/MSP/Jitter, and is set up to receive UDP messages from matlab in the same way as the actual robot does.

If you want to run the animation on your own computer, you can download Max/MSP/Jitter from http://www.cycling74.com, (free fully functional 30 day trial version). You need QuickTime and Java Runtime Environment installed, and if you are on a Windows PC, you will maybe need Microsoft .NET framework 3.x which you get from Windows Update.

Matlab and Max/MSP/Jitter is installed in the Robin lab at ifi, and the setup is tested and works fine there.

It is important that you get things working in the simulation before we try it on the X2 robot. This is to prevent the "unintelligent" X2 robot from getting damaged. The robot model may appear to be hovering, this is because the base is not in the model.

MATLAB

Three matlab scripts are provided along with this exercise. You can use these to communicate with the robot (or write your own if you prefer).

The script *activateAnimation.m* sets up three UDP connections to localhost on ports 7701, 7702, 7703. These are the ports where the simulation are listening for commands.

The script *deActivateAnimation.m* closes the three UDP connections.

The function *setRobotAngles.m* takes three angles (in degrees) as input, and sends these to the animation via the UDP connections.

Basic usage

1. Start the animation (the file x2model.maxpat)
2. Click the button next to "load all robot-parts" to load all the robot files.
3. Type "activateAnimation" in the matlab prompt to setup the udp-connection.
4. Type for instance "setRobotAngles(90,90,-90)" to set the robot to another configuration.
5. Try other configurations.
6. Close the UDP-connection by typing "deActivateAnimation" in matlab when you are done.

You should start this exercise by exploring the animation.

1

Write a path generator in matlab that calculates stepwise each new discrete cartesian point on a circular trajectory that the robot is supposed to move along. Combine this path generator with your inverse kinematics function.

These parameters are input to the path generator:

- Circle radius
- Origo position

These variables must be easily controllable in the code:

- The time delay between each position that is sent to the robot (the robot needs to get a position, then move, then get a new postion, etc.). You can use the matlab function *pause(seconds)* to create a pause between each robot step.
- The resolution of the path generator (i.e. the length of each step).
- The position of the centre of the drawing board.
- The orientation of the drawing board.
- The static DH-parameters (in practice meaning only the length of the links). This is to be able to transform your matlab code to work on the IA20 robot which has different dimensions than the X2 course robot.

2

Use the drawing function in the animation, and draw a circle that is parallel to the Y_0Z_0-plane with center at $X = 30cm, Y = 0cm, Z = 20cm$ (in the base coordinate frame of the robot), and a radius of 5cm. In the simulation the white dot marks the center point on the drawing board.

Draw the same circle rotated around the X-axis by 30, 60 og 90 degrees (with center in the same point as the first circle). Take screenshots of the drawings (use printscreen).

3

When we are working on the robot, we will project the tilted circles onto a vertical drawing board as shown in figure 1 on the next page. Make your matlab function able to do this sort of projection (e.g. by locking the y-coordinate.)

4

A regular industrial robot uses both the inverse kinematics and forward kinematics. Why do we not use the forward kinematics in this exercise?

Requirements:
Each student must hand in their own assignment, and you are required to have read the following requirements to student submissions at the department of informatics: http://www.ifi.uio.no/studinf/skjemaer/declaration.pdf

Your submission should be sent as a zip-file. Send it by e-mail to **vokjelse[at]student.matnat.uio.no**.

Your submission must include:

- The matlab-functions for path generation etc. (.m-files).
  Use comments in the code to document the functions!
- Screenshots of the circles drawn by the model

Name the file: "inf3480-ex3-*your_username.zip*".

Deadline: Monday, May 3rd

Don't hesitate to ask us if you have questions.
Contact information is at the course website.

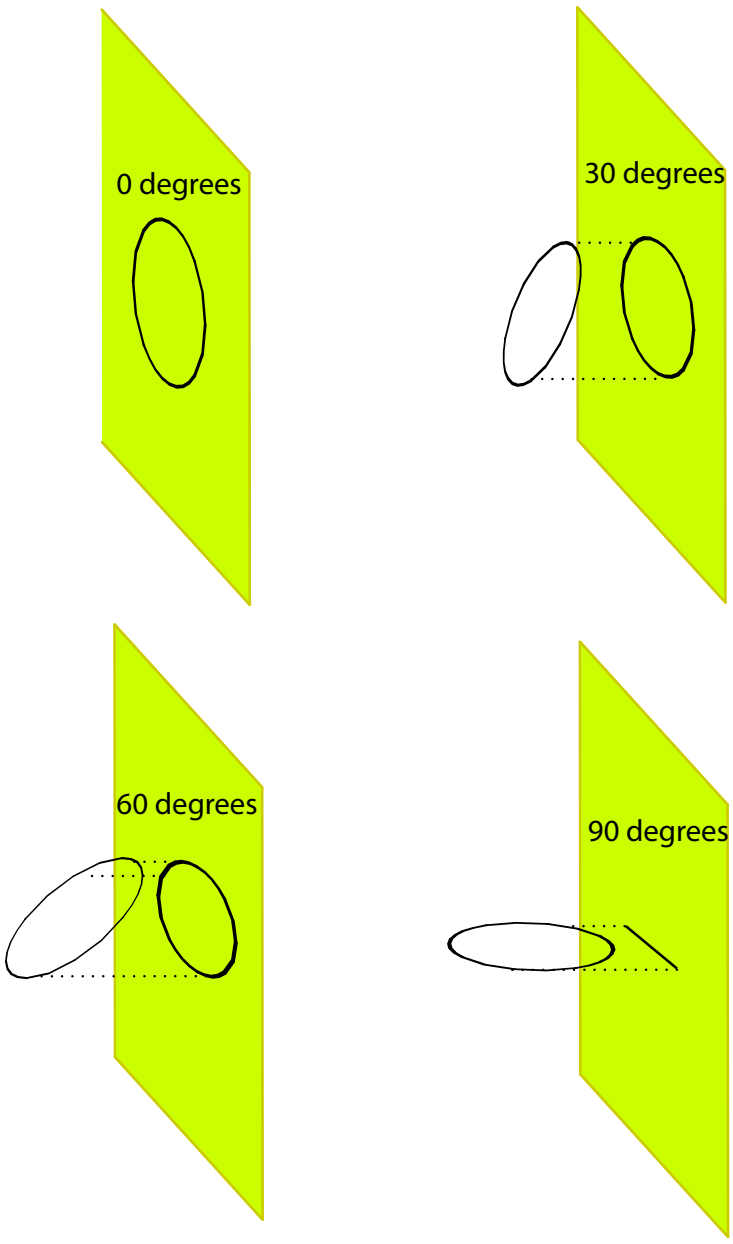

Figure 1: projecting circles on the drawing board