Department of Computer Engieneering

TECHNISCHE UNIVERSITÄT CHEMNITZ

Informatik

Prof. Dr. Wolfram Hardt, Dipl.-Inf. Michael Nagler

Hardware/Software Codesign II
Summer Term 2015

Lab 4

Collision Detection

1 Introduction

In the previous Lab a control unit for the FPGA-board that converts control commands into concrete moving instructions was implemented. The robot that is used in the labs is able to move in a way that it collides with itself. Since this can damage the robot, a collision detection for the robot has to be implemented on the FPGA-board that inhibits such behaviour.

2 Explanation

One possibility to prevent damages to the robot is to avoid collisions. Therefore it has to be checked whether the robot would collide with itself in the next desired position. If the collision detection foresees a collision, the robot control will not send that position as target position. An additional difficulty is that not only the target position must be collision free but also the movement to that position itself. We can solve this problem by only allowing a single step at only one motor at a time.

The basic model for the calculations is depicted in Figure 1.

The collision detection distinguishes static and dynamic conditions. A static condition describes the range that a motor value can have. It does not include any calculations that are dependent on other motor values. Pratically, these are the end positions of the motors themself.

Table 1 contains the valid intervals for the motor values.

Motor	rad		deg	
	Min	Max	Min	Max
1	$-2,618$	$2,618$	-150	150
2	$-0,3665191$	$2,094395$	-21	120
3	$-\pi$	π	-180	180
4	$-2 \cdot \pi$	$2 \cdot \pi$	-360	360
5	$-\pi$	π	-180	180

Table 1: Static Intervals for the Robot

The value for **motor 6** are measured in *mm* instead of degrees. The allowed interval for that motor value is ***[0mm, 50mm]***.

If these static conditions are not kept, the new target position is not valid and therefore must not be sent to the computer that controls the robot.

The dynamic conditions comprise e.g. the dependencies between the angles α_3 and α_2 (see Figure 1). These conditions are dynamic because the valid interval will change dynamically according to the current values of other angles. The calculations of these conditions is a bit more complicated. The vertical axis (α_0) is irrelevant for the detection of a self-collision, which simplifies matters. Also the rotation of the grabber (α_4) and its opening width (α_5) can be neclected. This can be derived from the model depicted in Figure 1.

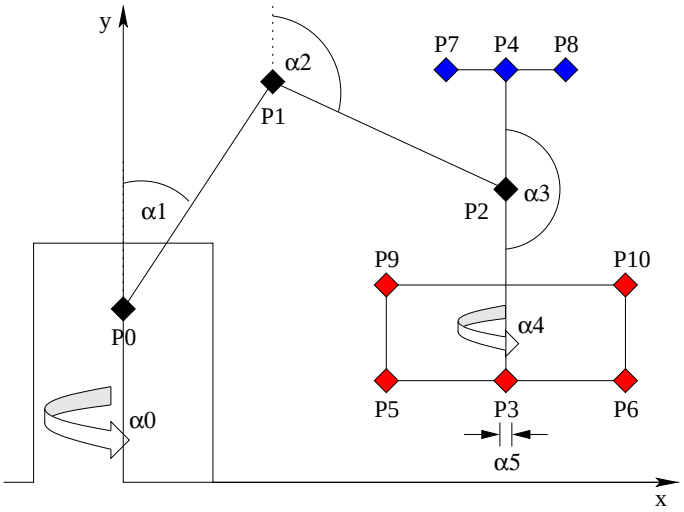

Figure 1: Simplified Point Model of the Robot

The next step of the collision detection is to calculate the locations of the points P_0 to P_{10}. At this point it is important to know that the angles are relative to the vertical axis. If the motor that corresponds with angle α_1 moves, the angle α_3 will not change. ***Hint**: beware that the angles in the data structure are given in degree but most implementations of trigonometric functions take their parameters as radiant.*

$$
\begin{aligned}
P_0 &= (0, l_0)^T \\
P_1 &= P_0 + l_1 \cdot (\sin\alpha_1, \cos\alpha_1)^T \\
P_2 &= P_1 + l_2 \cdot (\sin\alpha_2, \cos\alpha_2)^T \\
P_3 &= P_2 + l_3 \cdot (\sin\alpha_3, \cos\alpha_3)^T \\
P_4 &= P_2 - l_4 \cdot (\sin\alpha_3, \cos\alpha_3)^T \\
P_5 &= P_3 + l_5 \cdot (\cos\alpha_3, -\sin\alpha_3)^T \\
P_6 &= P_3 - l_5 \cdot (\cos\alpha_3, -\sin\alpha_3)^T \\
P_7 &= P_4 + l_6 \cdot (\cos\alpha_3, -\sin\alpha_3)^T \\
P_8 &= P_4 - l_6 \cdot (\cos\alpha_3, -\sin\alpha_3)^T \\
P_9 &= P_5 - l_7 \cdot (\sin\alpha_3, \cos\alpha_3)^T \\
P_{10} &= P_6 - l_7 \cdot (\sin\alpha_3, \cos\alpha_3)^T
\end{aligned}
$$

The values l_0 to l_7 represent the length of the several segments as summarised in Table 2.

Value	Length	Segment
l_0	340	floor-P_0
l_1	220	P_0-P_1
l_2	220	P_1-P_2
l_3	150	P_2-P_3
l_4	145	P_2-P_4
l_5	60	P_3-P_6
l_6	22, 5	P_4-P_8
l_7	82	P_5-P_9

Table 2: Segment Lengths in *mm*

Having calculated the locations of the points these relativly simple checks have to be done:

- to avoid the collision of P_1 to P_{10} with the floor and the socket, it must be that:

$$f_{surface}(x_i) < y_i$$

where:

$$f_{surface}(x) = \begin{cases} 360: & -110 \leq x \leq 110 \\ 0: & \text{else} \end{cases}$$

The surface consists of the panel where the robot is mounted and the socket (radius about $11cm$) of the robot.

- to avoid that the grabber rotates into Segment 2 ($\overline{P_1P_2}$), the enclosed angle between $\overline{P_1P_2}$ and $\overline{P_3P_4}$ must not become to small:

$$min(|\pi - (\alpha_2 - \alpha_3)|, |\pi - (\alpha_3 - \alpha_2)|) \geq 1,1694$$

There is a collision if one of these conditions is not fulfilled.

A more complex condition is the one that describes the relation between the grabber and the Segment 1 ($\overline{P_0P_1}$):

$$a = \frac{x_1 \cdot y_0 - x_0 \cdot y_1}{y_0 - y_1}$$
$$b = \frac{x_1 \cdot y_0 - x_0 \cdot y_1}{x_1 - x_0}$$
$$\alpha_1' = \arctan \frac{b}{a}$$
$$\varphi_1 = \frac{\pi}{2} - \alpha_1'$$
$$p = a \cdot \sin \alpha_1'$$

We calculate the distance between each of the points P_2 to P_{10} and the straight line extended over Segment 1:

$$d_{1i} = x_i \cdot \cos \varphi_1 + y_i \cdot \sin \varphi_1 - p$$

Now we need to check which points $P_i = (x_i, y_i)$ are near the line segment Segment 1. For all those points the term below is true.

$$(x_0 - d_1 < x_i < x_1 + d_1 \vee x_1 - d_1 < x_i < x_0 + d_1) \wedge (y_0 < y_i < y_1 \vee y_1 < y_i < y_0);\ where\ d_1 = 35$$

For these points it must be that:

$$|d_{1i}| \geq 35$$

If one of theses conditions does not hold, there is a collision. The detailed derivation of these conditions can be looked up in [1].

3 Tasks

3.1 Collision Detection

Implement the collision detection on the FPGA-board. Extend the solution from the previous lab with the adequate source code.

Hint: *The trigonometrical functions of the C-library work with radian whereas the angles in the* `RoboStat`*-data structure are given in degree.*

3.2 Test

Test your implementation using the actual robotic arm. There is only little danger of really destroying the robotic arm since the software on the workstation also contains a collision detection. If the board returns illegal target positions, the workstation application will show in the lower right field „Fehler“(Error), but does not transmit the command to the robotic arm.

An appropriate command sequence to test your implementation you will find below.

1. Increase angle of motor 3 until a collision is detected. This is the collision of the grabber with arm segment 2.
   (angles: 0, 0, 113, 0, 0, 0)
2. Set angle of motor 3 to 90°.
   (angles: 0, 0, 90, 0, 0, 0)
3. Set angle of motor 4 to 90°.
   (angles: 0, 0, 90, 90, 0, 0)
4. Set angle of motor 2 to 90°.
   (angles: 0, 0, 90, 90, 90, 0)
5. Move the grabber flat to the mounting panel by incrementing the angle values of motors 2 and 3.
   (angles: 0, 119, 141, 90, 0, 0)
6. Change the tilt of the grabber. This should only be possible in a very limited range.
   (angles: 0, 119, 141, 74..90, 0, 0)
7. Decrease angle of motor 2 to 90°(angles: 0, 90, 141, 90, 0, 0)
8. Increase angle of motor 3 as far as possible (which should be 170°because of a collision with the mounting socket)
   (angles: 0, 90, 170, 90, 0, 0)
9. Change the tilt of the grabber. This should only be possible in a very limited range.
   (angles: 0, 119, 141, 57..117, 0, 0)

Hint: *Because of roundoff errors the might be differences in the possible range limits of about 1-2°.*

4 Questions

1. What had to be considered if the angle would not be relative to the horizontal but relative to the connection arm segments?

References

[1] Twiefel, Jens: *Sichere internetbasierte echtzeitfähige Robotersteuerung.* Studienarbeit, May 2003.