BUDAPEST UNIVERSITY of TECHNOLOGY and ECONOMICS
Faculty of Mechanical Engineering
Department of Mechatronics, Optics and Engineering Informatics

FINAL PROJECT

Development of a universal robot controller: PC based path planning and interpolation, RT-Middleware based control

Budapest University of Technology and Economics

Narvik University College

Balázs Varga

Supervisors:

Péter Korondi, Ph.D., Hungary Bjorn Solvang, Ph.D., Norway

Gábor Sziebig, Dipl.-Ing., Norway

2009

ABSRACT

This final project demonstrates the process of the development of an indutrial robot controller. We made the project with Bence Kovács [1] and Géza Szayer together, the documentation of the robot is complete with their final thesis.

Our main goal was to modernize and Adept SCARA 604-S industrial robot, to make it able to work like it did with the original controller, to make it able to assemlby, actuate with linear and angle interpolation, and make it able to process G code from general CAD/CAM softwares, and to move on these paths. We inserted it in RT-Middleware system, and it can communicate with the PC or other industrial devices via standard interfaces.

In my final thesis I will demonstrate the general infromations and geometry specifications of the robot. The development of the universal I/O card, the interplotator and path planner algorithm and the way how you can join this system into RT-Middleware platform can be seen as well. RT-middleware has a really fast evolution is nowadays, it can make the making of a universal industrial manufacturing chain easier.

Keywords: Industrial robot controller, SCARA, Interpolation, RT-Middleware

KIVONAT

Ez a diplomamunka egy ipari robotvezérlő tervezésének és építésének egy részét mutatja be. A teljes rendszert Kovács Bencével [1] és Szayer Gézával [2] valósítottuk meg, így a vezérlő dokumentációja az ő diplomamunkájukkal teljes.

Célunk volt egy Adept SCARA 604-S típusú ipari robot modernizálása, annak alkalmassá tétele az eredeti feladatainak ellátásán –szerelés, pakolás, lineáris és csukló interpolációs mozgásokkal- túl CAD/CAM szoftver által generált szabványos G kóddal leírt pályák bejárására, RT-Middleware rendszerbe illesztése, valamint univerzálisan több interfészen keresztül PC -vel és más ipari automatizált eszközökkel való kapcsolattartásra.

Diplomamunkámban ismertetem a robot általános leírását, geometriai viszonyait, univerzális I/O kártyájának fejlesztését, interpolátorációs és pályatervező algoritmusait, valamint hogy hogyan illeszthető a rendszereünk a napjainkban kiemelkedően fejlődő univerzális gyártó rendszerek megalkotását lehetővé tevő RT-Middleware rendszerbe.

Kulcsszavak: Ipari robotvezérlő, SCARA, Interpolátor, RT-Middleware

SUMMENDRAG

Denne avhandlingen presenterer en del av planleggingen av en industrirobotkontroller og hans bygning. Det fulle systemet med Kovács Bence [1] og vi fullbrakte med Szayer Géza[2] det, som dette kontrollerens dokumentasjon med sin avhandling fulle.

Enkelt Adept SCARA 604-S var vårt mål maskinskriver moderniseringen av en industrirobot, det egnet man punktet sitt på tilførselen av hans opprinnelige oppgaver - emballasje, linje- og håndledd med interpolasjonsbevegelser også CAD/CAM standard-G nedfelte opp en re-:kondisjonalt på inntreden av omløpsbaner med en kode som programvare, RT-Middleware som knytter ham til et system, og universelt gjennom flere interfacespc - vel og med annen industriell automatisert apparatersandhed opp på kontaktforvaring.

Jeg skisserer av den generelle beskrivelsen av roboten, sine geometrirelasjoner, utviklingen av kortet universelle Meg i min avhandling, interpolator og hans baneplanalgoritmer, og det hvordan kan passes den universelle produsenten som blir bedre i våre dager ualminnelig det å skape systemer mulig en skuespiller RT-Middleware i et system.

Nøkkelord: Industrirobotkontroller, SCARA, Interpolator, RT-Middleware

ACKNOWLEDGEMENTS

I would like to thank my supervisors, Professor Bjorn Solvang, Professor Péter Korondi and Gábor Sziebig for their professional support and guidance throughout the project and for the possibility to see the beautiful city of Narvik in the far northern part of Europe. The warm welcome and all the personal and professional experiences will never be forgotten.

I wish to thank the Norway Program of the Erasmus for their financial support.

Finally I would like to thank my family and all my friends for their endless support, encouragement and most of all their patience.

DECLARATION

I, the undersigned, hereby declare that the Final Project submitted contains the results of my work, assisted by my supervisors and that all other results taken from the technical literature or other sources are clearly identified and referred to.

Budapest, ... month .....day .....year

.....

Balázs Varga

Task sheet

TABLE OF CONTENTS

LIST OF FIGURES

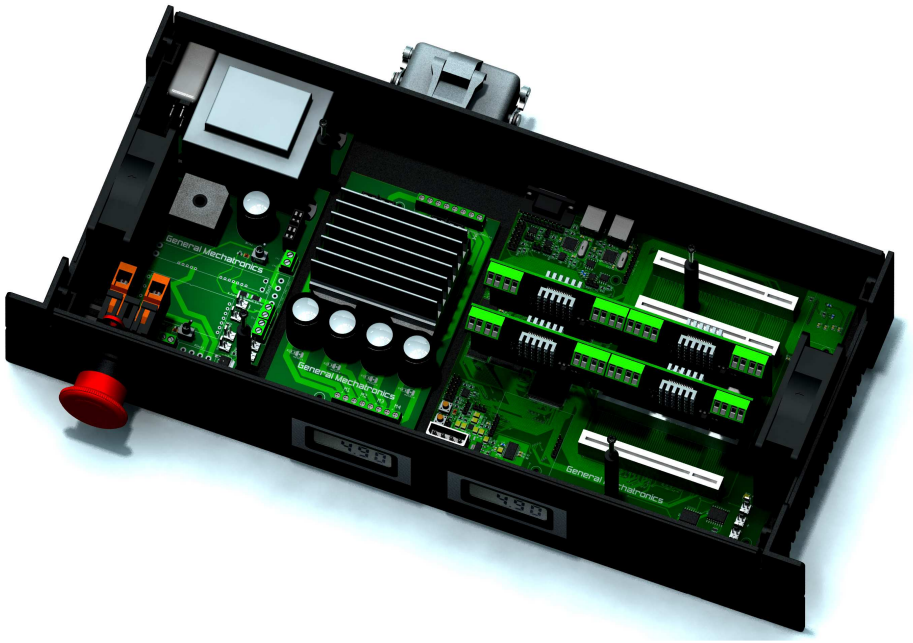
LIST OF TABLES

1 INTRODUCTION

The main goal of the project is to design and manufacture a complete universal robot controller for an Adept SCARA 604-S. Three people are working on the project at the same time, so it is team work. All the people attend to the Budapest University of Technology and Economics, Faculty of Mechanical Engineering, Department of Mechatronics, Optics and Engineering Informatics. Our robot can be found in Norway, in the laboratory of Hogskolen i Narvik, during the project we're going to there.

The main reason of the development is that in our days the small and middle size industrial companies have the possibility of the automatisation to manufacture with industrial robots. The used robot mechanics are not too expensive now. The reason of the low price is the faulty or obsolete controllers. To change them is a too expensive process. That is why it is not economical for the companies.

That is the reason why we design a universal robot controller with relative low price, which can permit the evolution of the industrial infrastructure on high technical level in economical circumstances.



2 MAIN GOALS OF THE PROJECT

2.1 Main goal of the development of universal I/O card

The controller needs a separeted part to handle the sigals of the robot. For the safety of the processors I need to isolate the different voltage levels. First of all I have to analyse the physical and elecrical specifications of the robot. The motors have different type of encoders, three of them have totem-pole, one of them has open-collector output. The gripper of the robot works with pneumatical valves, they need 24V and a minimum of 100mA to switch per each. The quantity of the main processors pins is limited, that is why I need a simple RISC processor, wich can use a sort of compressd communication form to send the end and zero point informations. This has to to use the high power outputs as well.

The card has to accommodate phisically to the PCI slot and controllers box.

2.2 Main goal of the development of simulation

The reason of the simulation was simple. We needed a realtime environment to understand and analyse the specifications of the geometry of the robot arm, to create an interpolator and a path planner. This part of the project is on the higher level of the structure, that is why it needs the lower levels, but during the development of the lower levels we can work with the simulator easily, to develop on the real hardware is a too big and dangerous challange, beacause in case of failure the robot can damage everything in the workspace including itself as well. In a virutal space we can define the border or the special limits of the workspace, testing the orientation constant, check the change of the angles, the movement of the TCP, handle the acceleration process.

With a realtime simulation device we can follow the change of the whole process, compare the results with each other.

Our expectations for the simulation software are: we want to display the workspace with the robot arm, the results of the functions, analyse, test and modify the inputs in realtime, convert the code easily to C code, and it should be prevalent.

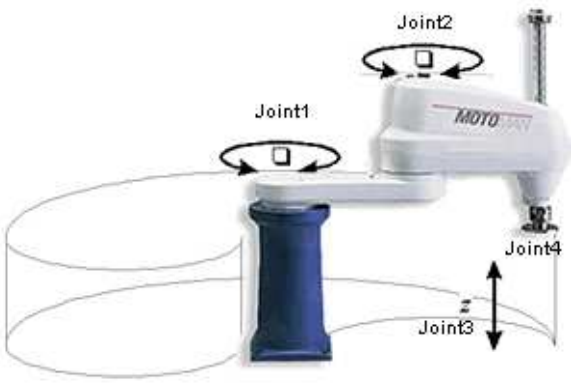
The best choice by these criterions the MATLAB 7.6.0.(R2008a) with Simulink 7.1.(R2008a).

2.3 Main goal of the development of RT-Middleware Components

The robot controller can be really universal if it has a standard interface and everybody can use it anywhere after the development. The users do not need to know the lower levels only most important commands and functions. That is why we need to choice a good open source envoirement where it can connect. In our days the result of the last researches is the RT-Middleware system. During the development I want to build and control our system with the components of RT-M, I will analyse and test the possibilites of the system.

3 THEORETICAL BACKGROUND

3.1 Structure of the robot



3.1. Figure: Structure of the SCARA

The Adept 604-S is a four-axis SCARA (Selective Compliance Assembly Robot Arm) shown in Figure 3.1 Joints 1, 2 and 4 are rotational and joint 3 iss translational.

Joint 1 provides the rotation for the inner link. Limit sensors restrict motion of joint 1 to a range of +100 degrees and -100 degrees. Joint 1 can move 20 degrees past each limit sensor before hitting a physical hardstop. This 20 degrees area is not usable during normal robot operations.

Joint 2 provides rotation for the outer link. Limit switches restrict motion of joint 2 to a range of +140 degrees and -140 degrees. Joint 2 can travel an additional 10 degrees past the limit switches before hitting a physical hardstop. This area is not usable during normal robot operations.

Effectively, motion of joint 2 is similar to an elbow capable of acting in both left and right hand configurations.

Joint 3 provides vertical translation of the quill. Joint 3 will allow the quill to move up and down with a maximum stroke of 150 mm. In addition, Joint 3 can travel an additional 10 mm past each limit switch before hitting a hardstop. This range is not usable during normal robot operations.

Joint 4 provides the rotation of the quill. It does not have hardstops.

The joint motion range, or travel, is limited by both software and hardware. The fixed mechanical limits are called hardstops. The Adept 604-S also uses limit switches to shut

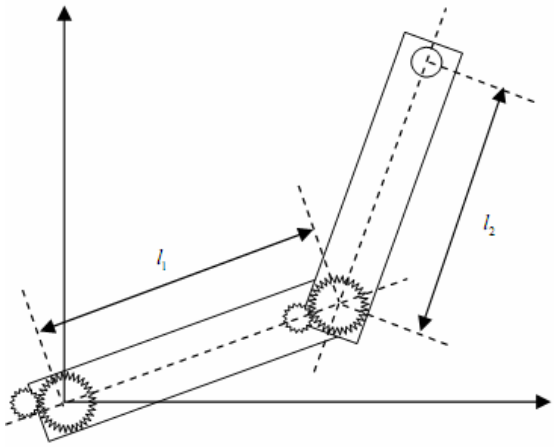
off the robot's power before contacting the hardstops. Softstops are normally used when the normal range of the robot must be limited.

On the other hand, hardstops are placed at the limits of joint travel in Joints 1, 2, and 3. Each hardstop has a limit switch that will close before the joint hits the hardstop. When a limit switch closes, the controller will turn off the robot's power supply. The hardstops are designed to withstand such impacts. Joint 4 does not have hardstops or limit switches.

SCARAs are generally faster and cleaner than comparable Cartesian systems. They require a small footprint and provide an easy, unhindered form of mounting. On the other hand, SCARAs can be more expensive than comparable Cartesian systems and the controlling software requires inverse kinematics for linear interpolated moves. [3]

3.2 Inverse kinematic

In practice we know the coordinates of the tool center point (TCP) in Cartesien coordinate system, where we want to move the robot arm. The motion controller part of the robot controller need angles for the position control, that is why we need to transform the coordinates. This transformation is called inverse kinematic problem. The inputs of the function are X,Y, th outptus are Θ_1, Θ_2.

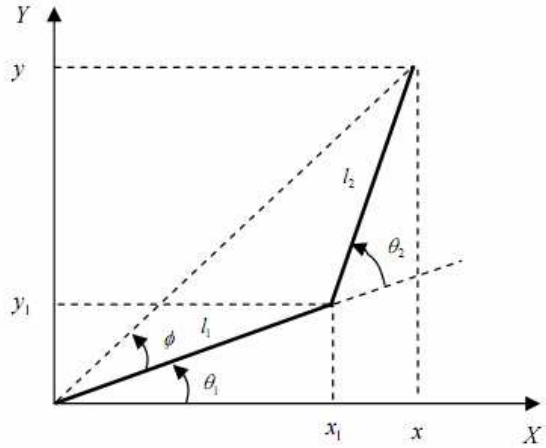


3.1. Figure: Position of the links

The equations between the position and the angles:

$$x = l_1 \cos\Theta_1 + l_2 \cos(\Theta_1 + \Theta_2) \quad (3.1.)$$

$$y = l_1 \sin\Theta_1 + l_2 \sin(\Theta_1 + \Theta_2) \quad (3.2.)$$



3.2. Figure: Position of the TCP

The invers equations:

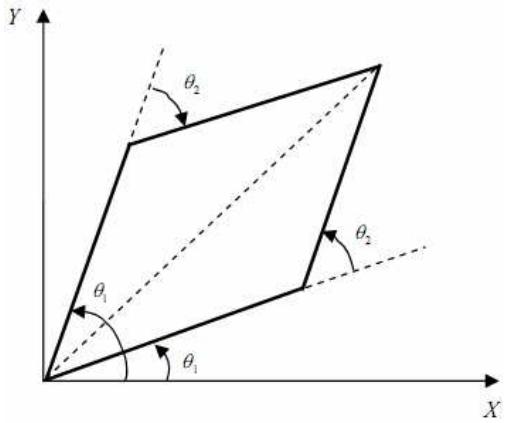
$$R = \sqrt{x^2 + y^2} \quad (3.3.)$$

$$\Phi = \arccos\left(\frac{l_1^2 + R^2 - l_2^2}{2 l_1 R}\right) \quad (3.4.)$$

$$\Theta_1 = \arctan\left(\frac{y}{x}\right) + \Phi \quad (3.5.)$$

$$\Theta_2 = \Pi - \arccos\left(\frac{l_1^2 + l_2^2 - R^2}{2 l_1 l_2}\right) \quad (3.6.)$$

The structure of the arm of the SCARA permits of two different angle compositions for the same TCP position. We must discriminate one pose, it would be faster, -that is why cheaper-, better in the practice. We can do this with a constant value.



3.3. Figure: Orintation problem

$$k = \{1,-1\} \tag{3.7.}$$

$$\Theta_1 = \arctan(\frac{y}{x}) + k\Phi \tag{3.8.}$$

$$\Theta_2 = \Pi - k\arccos(\frac{l_1^2 + l_2^2 - R^2}{2l_1 l_2}) \tag{3.9.}$$

[4]

Where:

x, y Cartesian coordinates of TCP

Θ_1, Θ_2 Angles of the joints

l_1, l_2 Length of the links

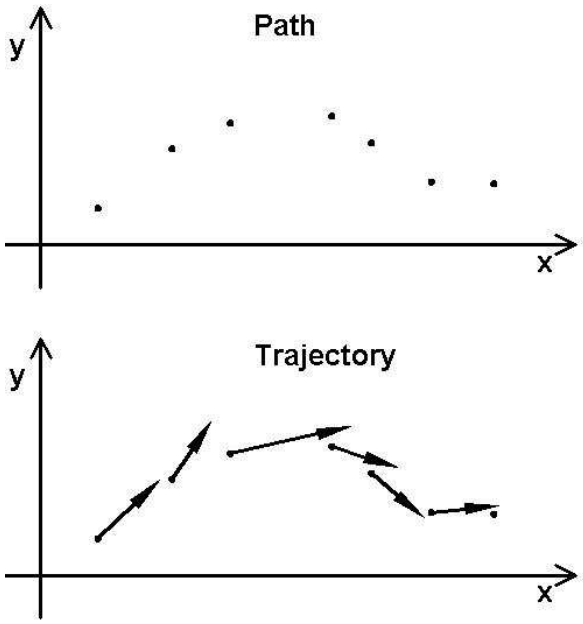
R Distance between TCP and zero point

3.3 Trajectory

The main goal of trajectory making is to generate reference input for the motion controller. We have to make clear two important definitions:

Path: It is a series of points; the robot arm has to follow this way. It is a normal geometry definition.

Trajectory: The points depend on the time, because they have velocity or acceleration information as well.



3.4. Figure: Difference between path and trajectory

There are two different ways of making trajectory:

PTP (Pose To Pose) motion, where the first and the last points of the path are given.

CP (Continuous Path) motion, there is a finite series of points

	CP	PTP
Motion	Consider initial and final pose & position and intermediate motion are important.	Consider initial and final pose & position are important but intermediate motion is not.
Example	Coating, grinding by a robot Precision surface machining	Assembly by a robot Tracking control of HDD head

3.1. Table: CP and PTP trajectory

We have to describe a trajectory as a time-varying function. There are two ways to do it as follows.

Use the position directly (for example, $x(t), y(t)$ in Cartesian coordinate system)

Divide the whole trajectory to some basic trajectories $\eta(t)$ and concatenate them.

We often select a straight line or a circular arc as a basic trajectory.

Straight line: $\eta(t)$ means the length of the line

Circular arc: $\eta(t)$ means the angle of the circle ($\eta(t) = \varphi(t)$)

We can calculate the position $x(t), y(t)$, corresponding the basic trajectory $\eta(t)$, and convert them $\Theta_1(t), \Theta_2(t)$ with inverse kinematics.

In practice the most used velocity profile for the PTP trajectory is the trapeze form. In this case the acceleration is maximal on the beginning of the path, after the speed is consant and the deceleration is maximal as well. Symmetrical trajectory:

$$q_m = \frac{(q_f + q_i)}{2} \tag{3.10.}$$

$$t_m = \frac{t_f}{2} \tag{3.11.}$$

$$\dot{q}_{\max} = \ddot{q}_c t_c = \frac{q_m - q_c}{t_m - t_c} \tag{3.12.}$$

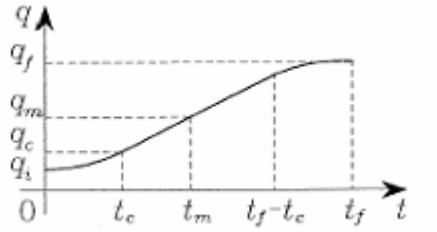
$$q_c = q_i + \frac{1}{2}\ddot{q}_c t_c^2 \tag{3.13.}$$

$$\ddot{q}_c t_c^2 - q_c t_f t_c + \ddot{q}_f - q_i = 0 \tag{3.14.}$$

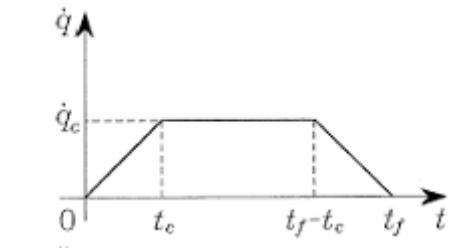
$$t_c = \frac{t_f}{2} - \frac{1}{2}\sqrt{\frac{t_f^2 \ddot{q}_c - 4(q_f - q_i)}{\ddot{q}_c}} \tag{3.15.}$$

$$\left|\ddot{q}_c\right| \geq \frac{4\left|q_f - q_i\right|}{t_f^2} \tag{3.16.}$$

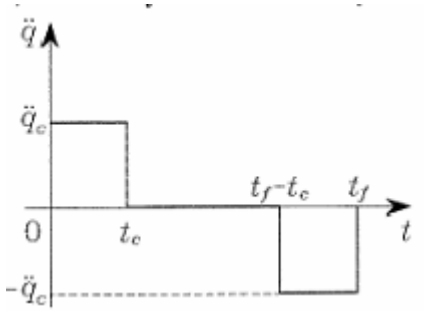
$$q(t) = \begin{cases} q_i + \frac{1}{2}\ddot{q}_c t^2 \bullet 0 \leq t \leq t_c \\ q_i + \ddot{q}_c t_c (t - \frac{t_c}{2} \bullet t_c \leq t \leq t_f - t_c \\ q_f - \frac{1}{2}\ddot{q}_c (t_f - t)^2 \bullet t_f - t_c \leq t \leq t_f \end{cases} \tag{3.17.}$$



3.5. Figure: TCP position



3.6. Figure: TCP velocity



3.7. Figure: TCP acceleration

Where:

q_m Half of the excursion

t_m Half of time of the movement

$q_{\max}$ Maximum of the excursion

$\dot{q}_{\max}$ Maximum of the velocity

q_c End of the acceleration phase

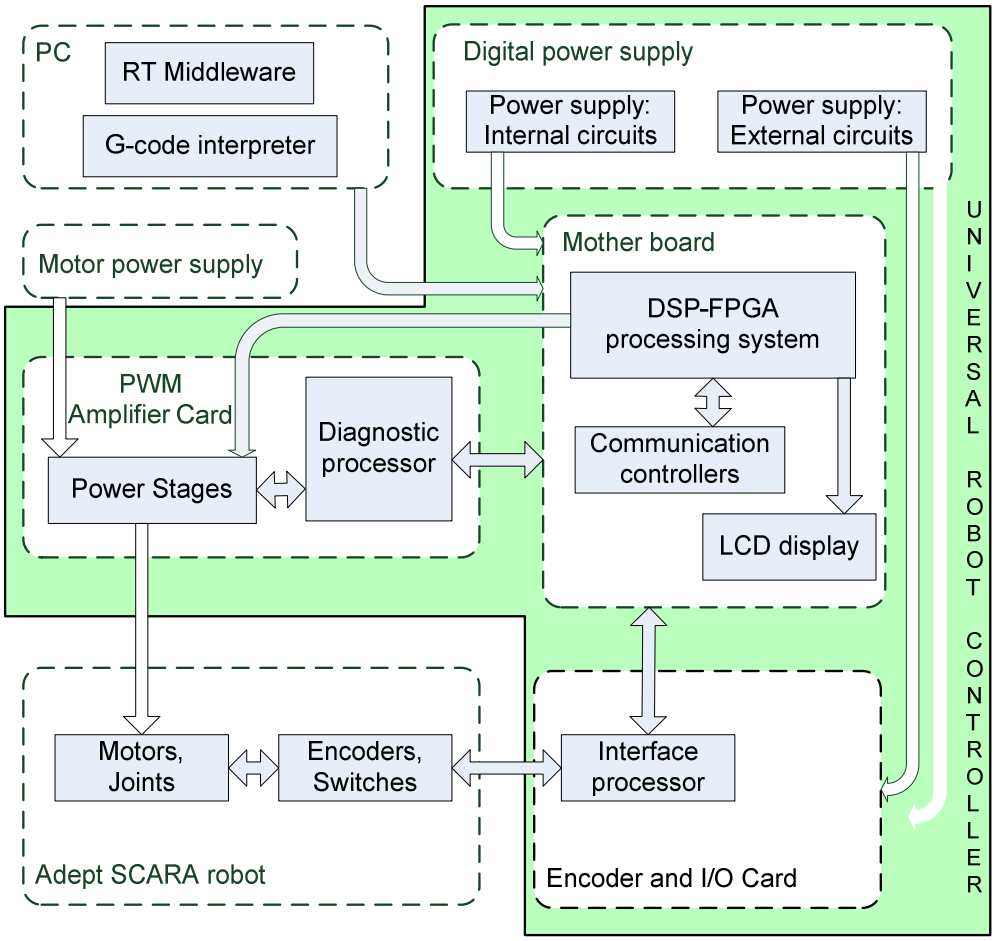
$\ddot{q}_c$ Maximum of the acceleration

t_c End of time of the acceleration phase

4 TECHNICAL SOLUTIONS

4.1 Overview

The whole project was to develop a universal robot controller unit. Our robot controller has several processing units: DSP, FPGA and microcontrollers. The block diagram of the whole system can be seen on figure 4.1. The controller has four main units: The motherboard is the central processing circuit with a high-speed DSP-FPGA system on it. The PWM amplifier card can drive the four DC motors of the SCARA robot, the Universal I/O card connects to the robot's sensors, after that every signal to the motherboard are fully optically isolated, and finally there is a separated main digital power supply board which supplies the internal and external digital circuits too. The motherboard contains five 124-pin PCI sockets for cards, and can handle two PWM amplifier cards, and two encoder and I/O card. With one PWM- and encoder card, the controller can drive a maximum four axis robot. With four cards it can control robots with 8-axis, and one socket remains for later development.



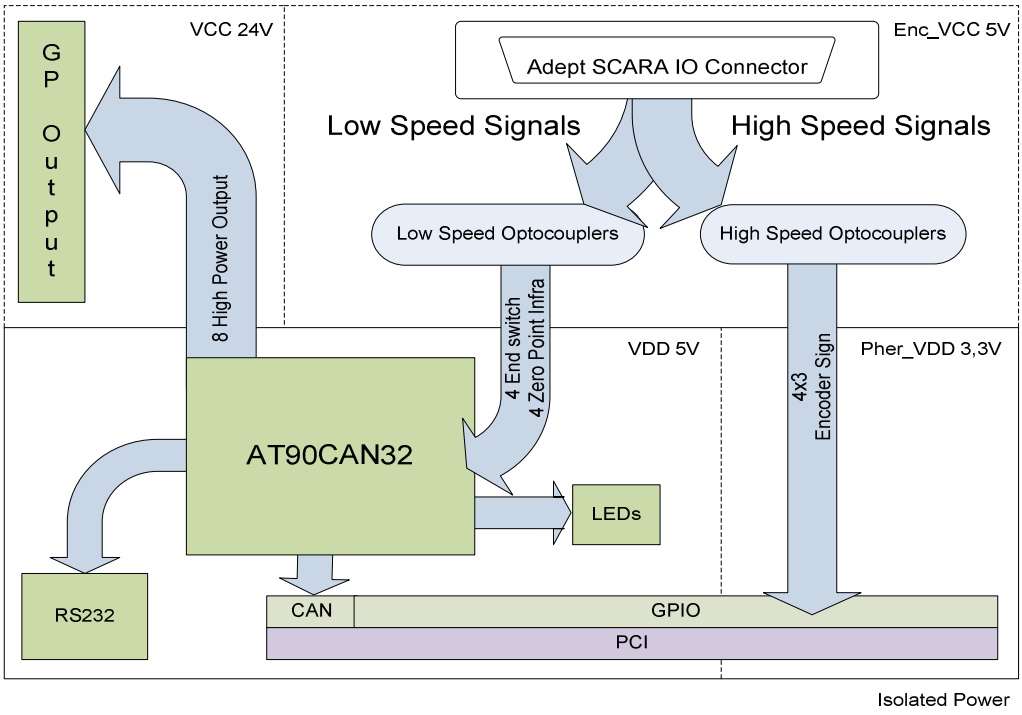
4.1. Figure: The main blockdiagram of the hardware architecture

4.2 Universal Input/Output Card

4.2.1 Requirements

The connector of the Adept SCARA has input, output and power supply pins as well. The card can give isolated 5V for the encoders and other electrical parts.[Apendix 1] The encoder signals are high speed lines, the frquency is around 20 kHz, they need a special version from the opto couplers, and the lower speed lines only need a general version. The encoders of the joints 1-2-3 are totem-pole outputs, the encoder of the fourth joint has open collector output, and there I need to use pull-up resistors. Every encoder has three output wires: channel A, channel B and channel Z. A and B are the quadratic signs, Z is a index sign in the begeining of every revolution.

The fastest and most stabile way to send information for the main processor is the CAN bus. The DSP can handle this, and it is easy to use this if we want to insert other cards into the system.



4.2. Figure: The blockdiagram of the univeral I/O card

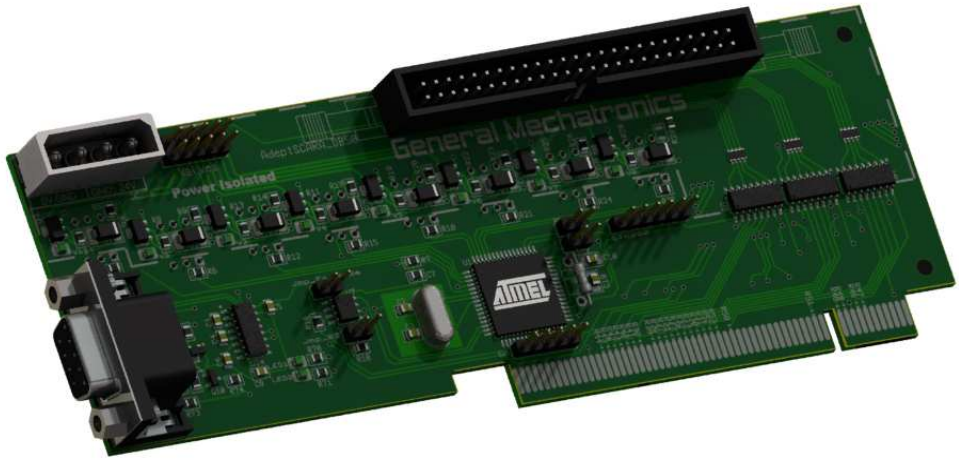
4.2.2 Applied devices

The most important function of the card is isolating the encoder signals. The freqency can be higher than the top freqency of a general optocoupler, so we need a special type, one that is designed for high speed communication. The best choice is the chip of Analog Devices the ACSL-6400. [5]

ACSL-6400 is a truly isolated, multi-channel and bi-directional, high speed optocoupler. Integration of multiple optocouplers is in monolithic form. These devices provide fulduplex and bi-directional isolated data transfer and communication capability in compact surface mount packages. Available in 15Mbd speed option and wide supply voltage range.

These high channel densities make them ideally suited to isolating data conversion devices, parallel buses and peripheral interfaces.

The microcontroller of the card is an AT90CAN32 AVR controller. [6] It works with 16MHz crystal oscillator. The most important features for me are:

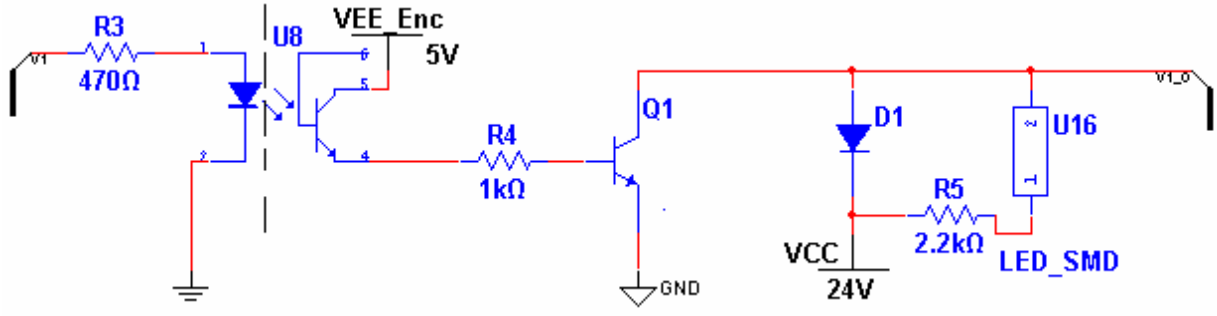


4.3. Figure: 3D design of the universal I/O card

- Up to 16 MIPS Throughput at 16 MHz
- In-System Programming by On-Chip
- 2K/4K/4K Bytes Internal SRAM
- Extensive On-chip Debug Support
- CAN Controller 2.0A & 2.0B - ISO 16845 Certified
- Dual Programmable Serial USART
- 53 Programmable I/O Lines
- Operating Voltages: 2.7 - 5.5V

The AVR collect the end switches and zero point signals. Zero point signals are inverted digital signs. If the robot is in the zero point the controller gets a zero. These signals are important for the calibration of the robot. The end switches can give an E-stop sign. They are after the zero sensors, for the safety of the robot. If the arm is in the end point we have to turn the power of the motors off.

The controller can send information about them via CAN bus or RS232 serial communication and switch the indicator LEDs. It is a good way to indicate errors; development is easier with them.



4.4. Figure: Circuit of the general outputs

The schematic of the general outputs can be seen in the figure 4.4. For the switching of general outputs we need amplifier circuits. The microcontroller switches a simple optocoupler (4N25) with 20mA current. It is for seperating the different voltage levels. The output of the 4N25 is the amplifier transistor. It is an NPN darlington type.

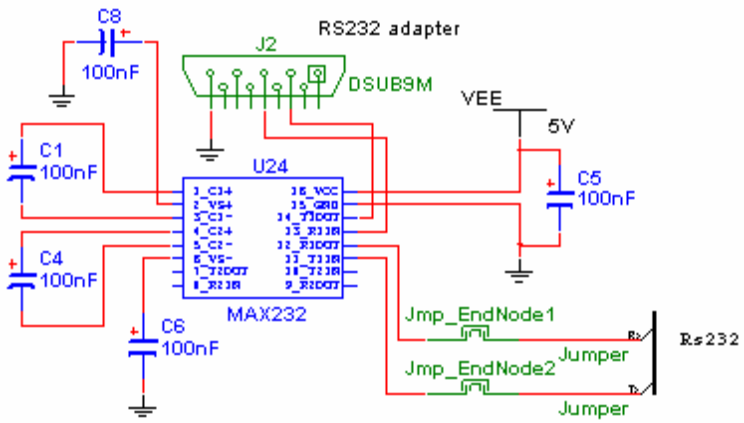
- High current (max. 0.5 A)
- Low voltage (max. 80 V)

Usually these outputs will switch coils, that is why these circuits need some other passive parts. The coils are energy sources after switching off, current want to flow there. The directon would be different like in the normal phase. This current can damage the transistor. To save the transistor there is the (D1) diode in the circuit. It can stop the unwanted current. Every output has an indicator LED in the main circuit. Their purpouse is to make development easier.

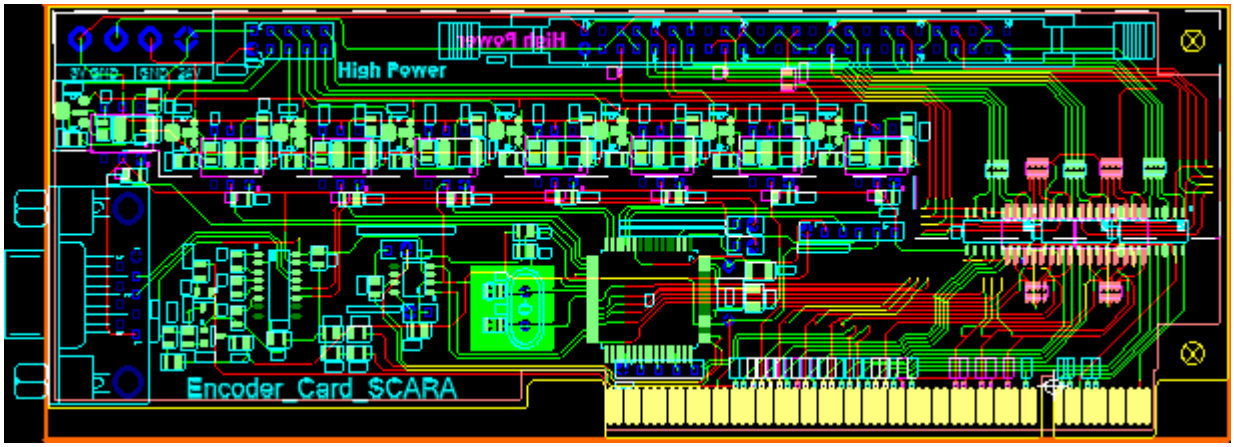
For the serial communication between the microcontroller and the PC an interface is necessary. There is a voltage level distance between the devices. The circuit which can handle this is the MAX232E (figure 4.5.) [7]

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply TIA/EIA-232-F voltage levels from a single 5-V supply. Each receiver converts TIA/EIA-232-F inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V, a typical hysteresis of 0.5 V, and can accept ±30-V inputs.

Each driver converts TTL/CMOS input levels into TIA/EIA-232-F levels. The PCB is a 1,6mm thick PCI bus compatible card with two layers.

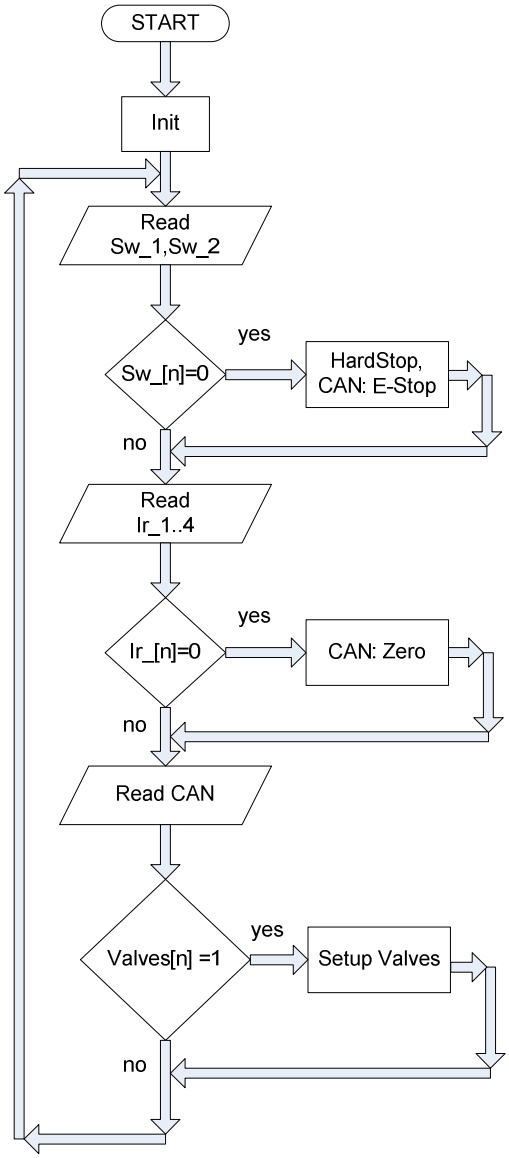


4.5. Figure: Circuit of the serial communication



4.6. Figure: Layout of the universal I/O card

4.2.3 Structure of the software



4.7. Figure: Structure of the AVR's software

The flow chart of the software can be seen in figure 4.7. The first step of the process is the initialisation of the microcontrollers' peripherials. After that we read the value of the end switches. Every signal from the robot is inverted. If one of them is on, it stops the power supply inmadiatelly, after that it sends a message to the DSP via CAN, because the motion controll has to know it, for exapmle the integrator of the PID must not overflow. The prosess is the same with the infra zero point sensors. After that it reads the reciever line of the CAN bus. If it gets a value to switch the general outputs, it will do it. After this iteration the loop will start anew.

4.3 Simulation

4.3.1 Overview of the environment

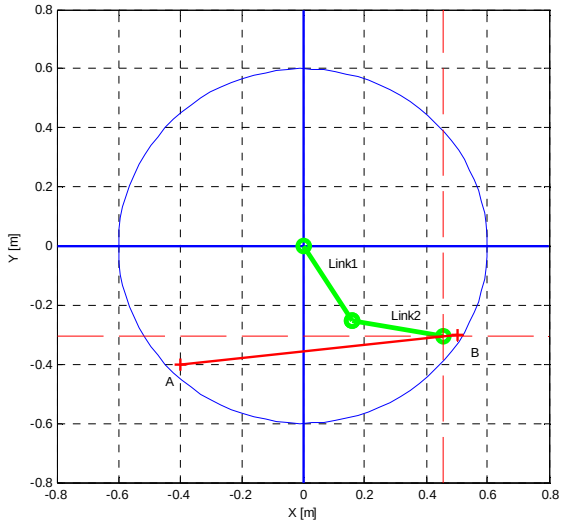
Simulink is an environment for multidomain simulation and model-based design for dynamic and embedded systems. It provides an interactive graphical environment and a

customizable set of block libraries. We can simulate, implement, and test a variety of time-varying systems, including communications, controls, signal processing, video processing and image processing. Add-on products extend Simulink to multiple modeling domains, as well as provide tools for design, implementation, and verification and validation tasks.

Simulink is integrated with MATLAB, providing immediate access to an extensive range of tools. We can develop algorithms, analyse and visualize simulations, create batch processing scripts, customize the modeling environment, and define signal, parameter, and test data. [8]

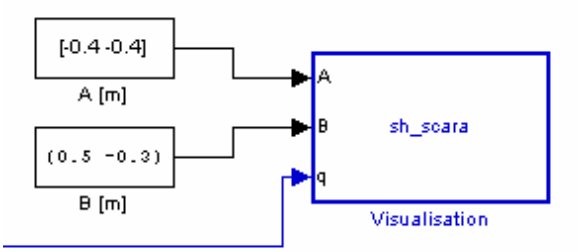
4.3.2 Visualisation

The first step of the work was creating a visualisation modul, where I can visualise the arms of the robot (Joint 1, Joint 2) in the workspace. Ther are two axis (X ,Y) with normal scale, two red broken lines connected to the TCP, they help us to read the value of the component. It shows the path where we want to move the TCP with red countinous line.



4.8. Figure: Visualisation modul

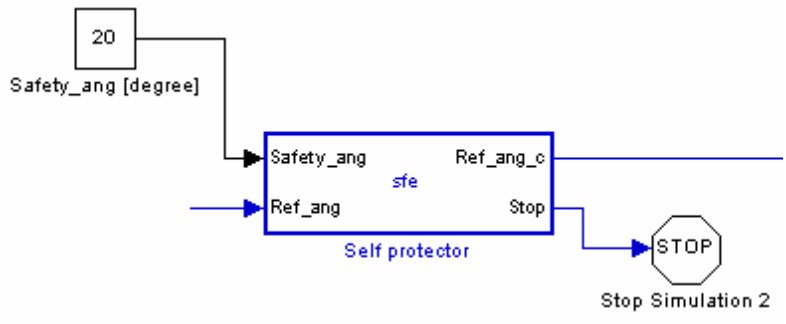
The MATLAB block is on x.Figure. The box has three inputs. The first two are the coordinates of the end points of the path (A,B). The third one is the vector of the joint angles in degree (q). This box has not got output values, just the generated figure.



4.9. Figure: Visualisation block

The box gets the angle vector from the Self prtoector block.

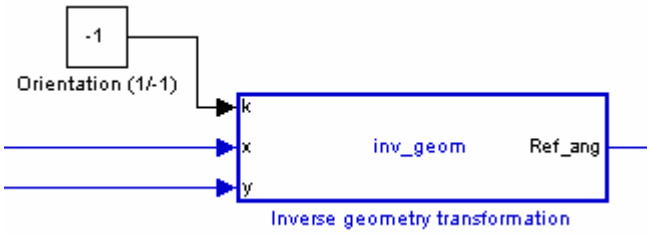
4.3.3 Self protector



4.10. Figure: Self protector block

Every SCARA robot has a bottom limit angle for the second joint. If the robot gets a lower reference value than that, it can damage itself. In the real life the robots have hard stop switches in this phase of the movement, which can turn the power of the robot off, but it is not the correct way to stop the arm, it is just for safety. We need to handle this situation. Inputs of the box are the safety angle (Safety_ang) and the reference (Ref_ang), the outputs are the correct angle value (Ref_ang_c) and a stop bit (Stop). The box gets the referenc angle vector from the Inverse Geometry Transformation block.

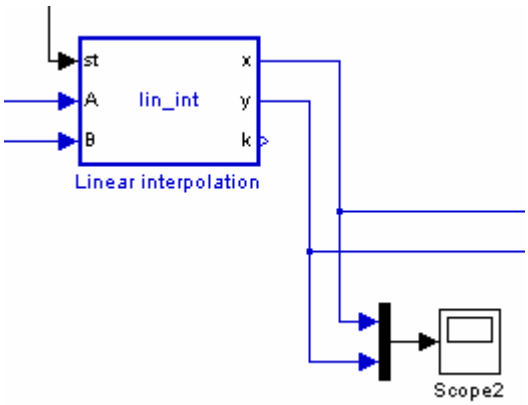
4.3.4 Inverse geometry



4.11. Figure: Inverse geometry block

The exercise of this box is to calculate angles from the end point information of the TCP. It is based on [3] Inputs of the box are the reference coordinates of the TCP (x,y) and orientation constant (k). The output is a reference vector of the angles (Ref_ang). The box gets the reference coordinates from the path planner block.

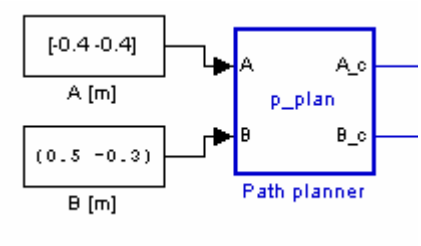
4.3.5 Linear interpolation



4.12. Figure: Linear interpolation block

This box can transform the timed excursion between the end points of the path. Inputs of the box are the timed excursion (st), coordinates of the first and the last points (A,B). The outputs are the coordinates of the TCP (x,y) and the oriantation constant (k). The box gets the reference points from the Path planner block, the excursion from the Acceleration profile block.

4.3.6 Path planner

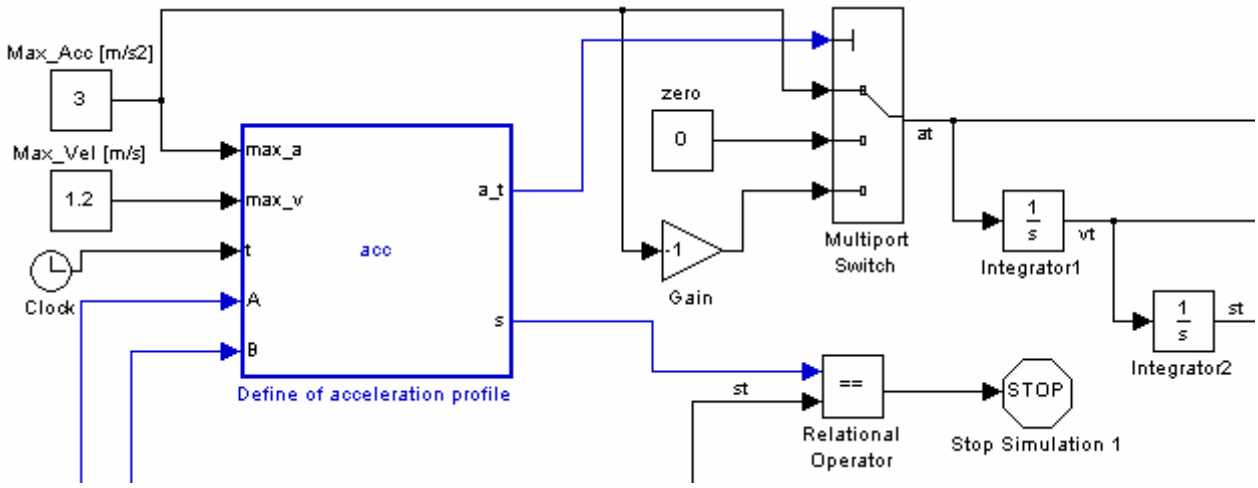


4.13. Figure: Path planner block

The path planner box is place where the limits of the workspace are. Here we can define the edge of the desk or the area of other machines or parts, where the robot must not go. Inputs of the box are the original input of the end points of the path (A,B). The outputs are the correct end points, where the robot can move (A_c,B_c). The box gets the reference from input or constants.

4.3.7 Dynamics

The Acceleration profile creates a timed acceleration function, then a timed position function for the interpolator. The output depends on some constants of the robot, like maximum acceleration and velocity. The ideal acceleration function is a step function, that is why it has discrete output like maximum acceleration, zero or maximum deceleration –it is the inverse of the maximum acceleration-.



4.14. Figure: Dynamics block

The output of the embedded function is a selector sign, which can select the correct output from the inputs of Multiport switch block. After that are two integrators to create the timed velocity and last the timed position sign. If the position reaches the maximum, the simulation will be stopped.

Deduction of the main equations of the block:

$$s = 2\left(\frac{v_{\max} t_c}{2}\right) + (t_f - 2t_c)v_{\max} \qquad (4.1.)$$

$$s = v_{\max}(t_f - t_c) \qquad (4.2.)$$

$$t_c = \frac{v_{\max}}{a_{\max}} \qquad (4.3.)$$

$$t_f = \frac{s}{v_{\max}} + \frac{v_{\max}}{a_{\max}} \qquad (4.4.)$$

If $t_f <= 2t_c$ then the waveform of the velocity will be modified, in this case:

$$t_f = 2t_c \qquad (4.5.)$$

$$v_{\max} = \sqrt{sa_{\max}} \qquad (4.6.)$$

Where

s Maximum of the excursion

$v_{\max}$ Maximum of the velocity

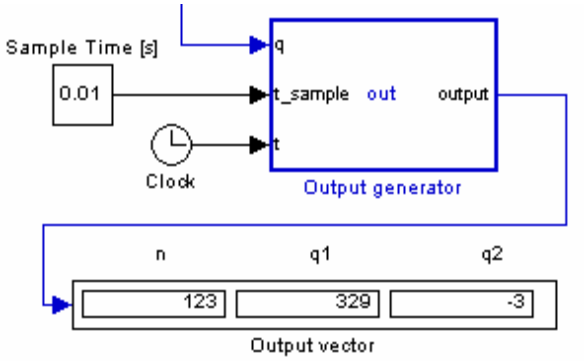
a_c Maximum of the acceleration

t_c End of time of the acceleration phase

t_f Full time of the movement

Inputs of the box are maximum acceleration value (max_a), maximum velocity (max_t), simulation system time (t), endpoints of the path (A,B). The outputs are the discrete selector sign (a_t), value of the whole distance between the endpoints (s). The box gets the reference from input or constans.

4.3.8 Output



4.15. Figure: Output block

This box is generating the output of the prosecces to the motion controller. The output vector is created in every millisecond, this way the motion controller can get timed points. The contoller is able to communicate via Bluetooth wireless protocol. Usually we can handle the bluetooth on PC like a simple serial port. This way the MATLAB sends the output vector via wireless communication for the controller. There is another way to send the information: via a simple text file. It can generate a database, and we can transfer it another way or analyse the bunch of data with other softwares.

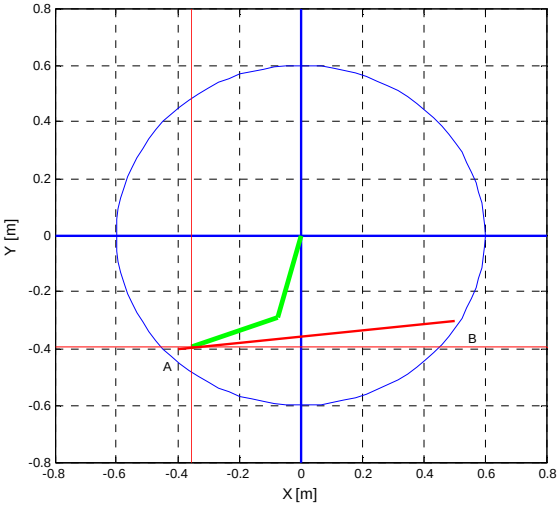
Inputs of the box are reference angle vector (q), sample time (t_sample), simulation system time (t). This box has not got output values, just the generated file or the wireless information.

4.3.9 Results

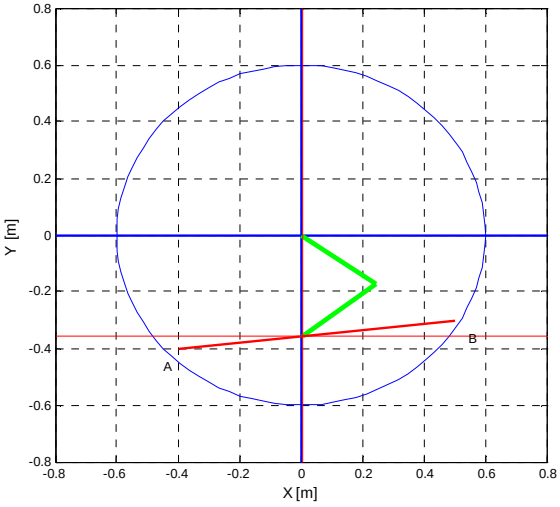
Variable	Value	Unit
A	(–0,4;–0,4)	$[m]$
B	(0,5;–0,3)	$[m]$
Max_acc	3	$\left[\frac{m}{s^2}\right]$
Max_vel	1	$\left[\frac{m}{s}\right]$
$Safety\_ang$	20	$[\circ]$

4.1. Table: The constants of the first test

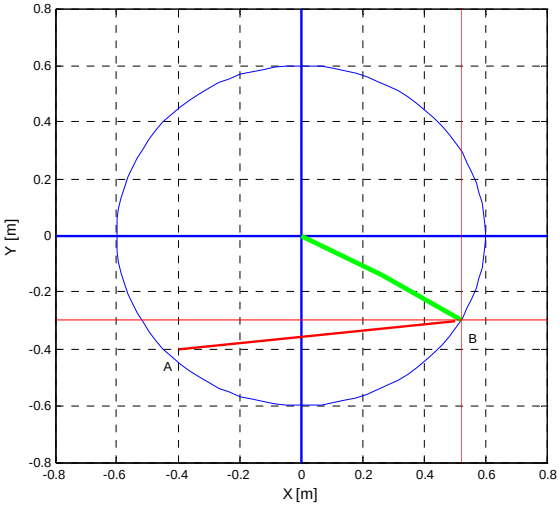
$k = -1$ [–]



4.16. Figure: Movement phase 1.1

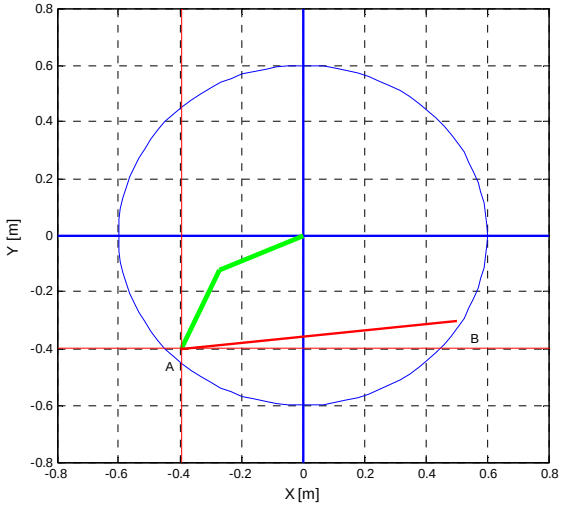


4.17. Figure: Movement phase 1.2

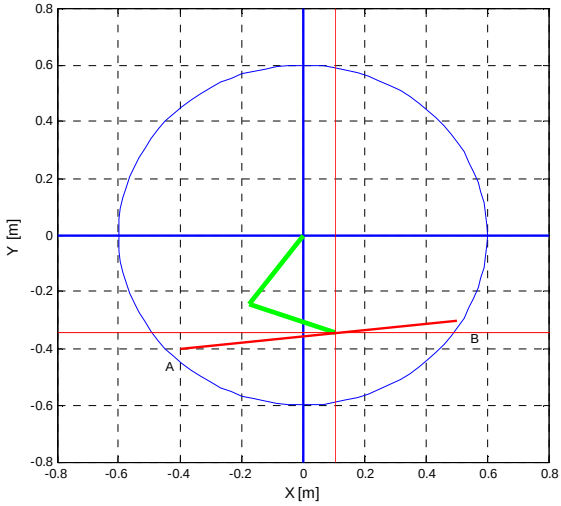


4.18. Figure: Movement phase 1.3

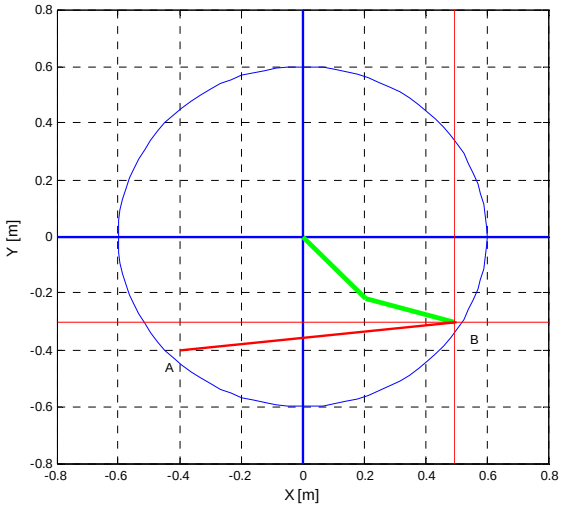
$k = 1$ [–]



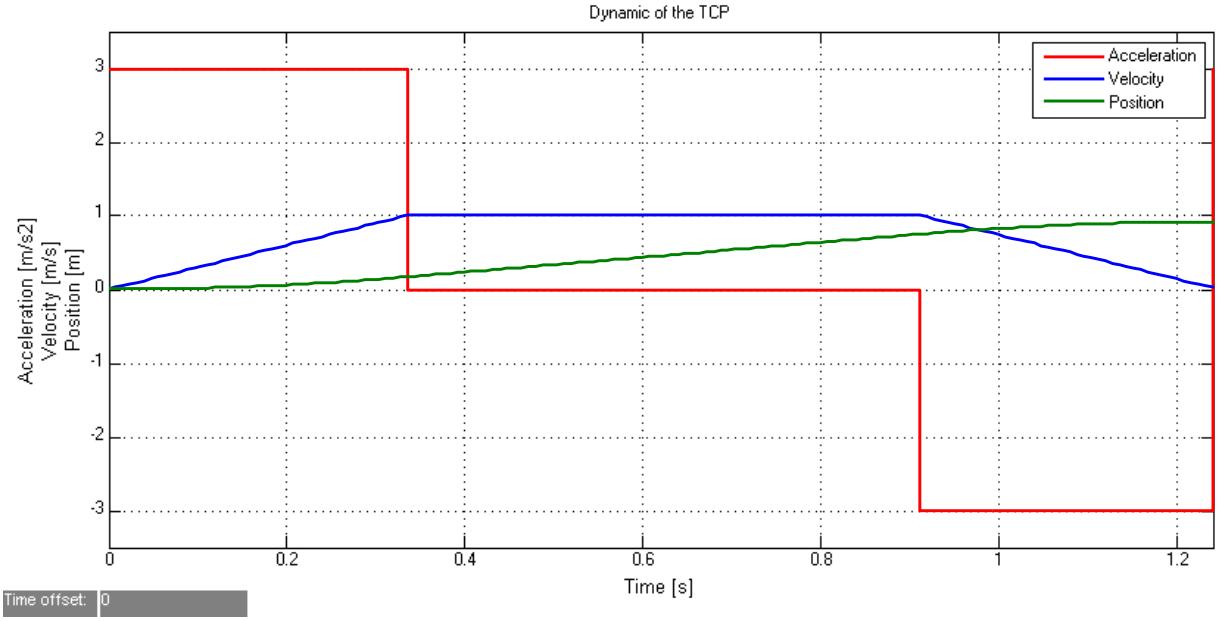
4.19. Figure: Movement phase 2.1



4.20. Figure: Movement phase 2.2

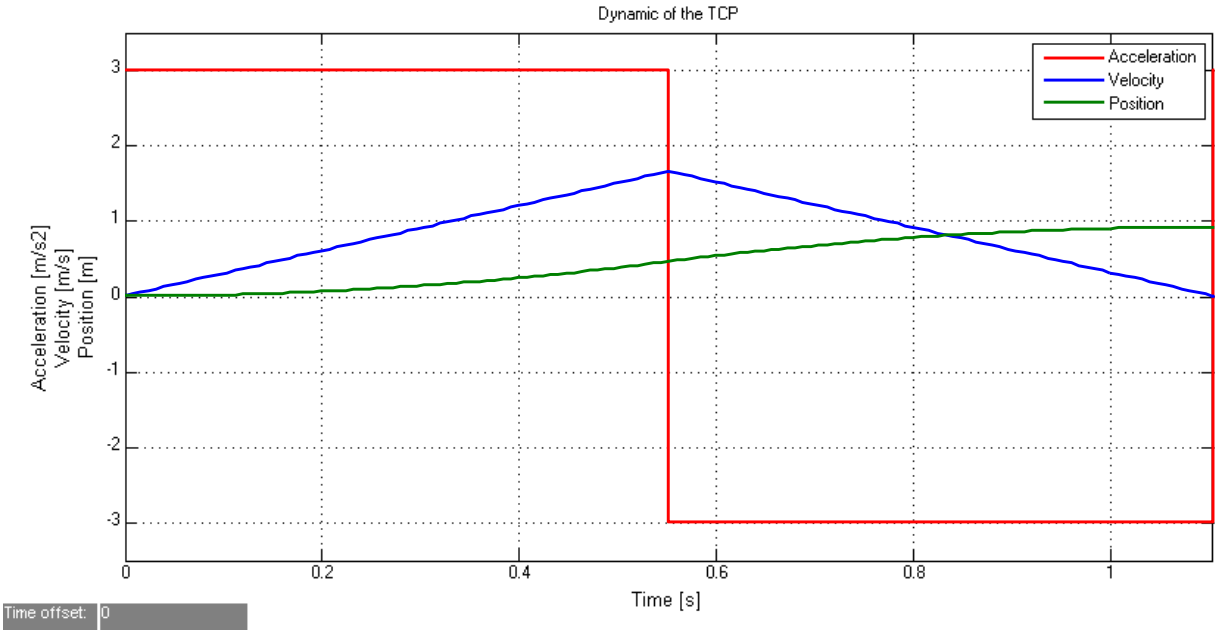


4.21. Figure: Movement phase 2.3

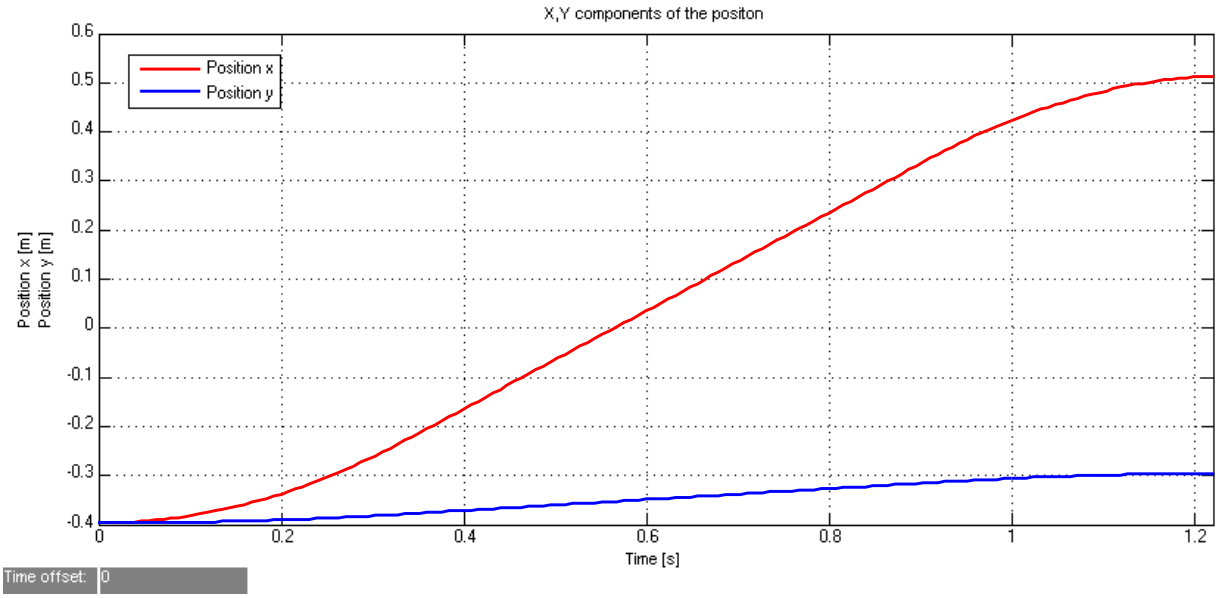


4.22. Figure: Dynamic of the TCP in case of far points

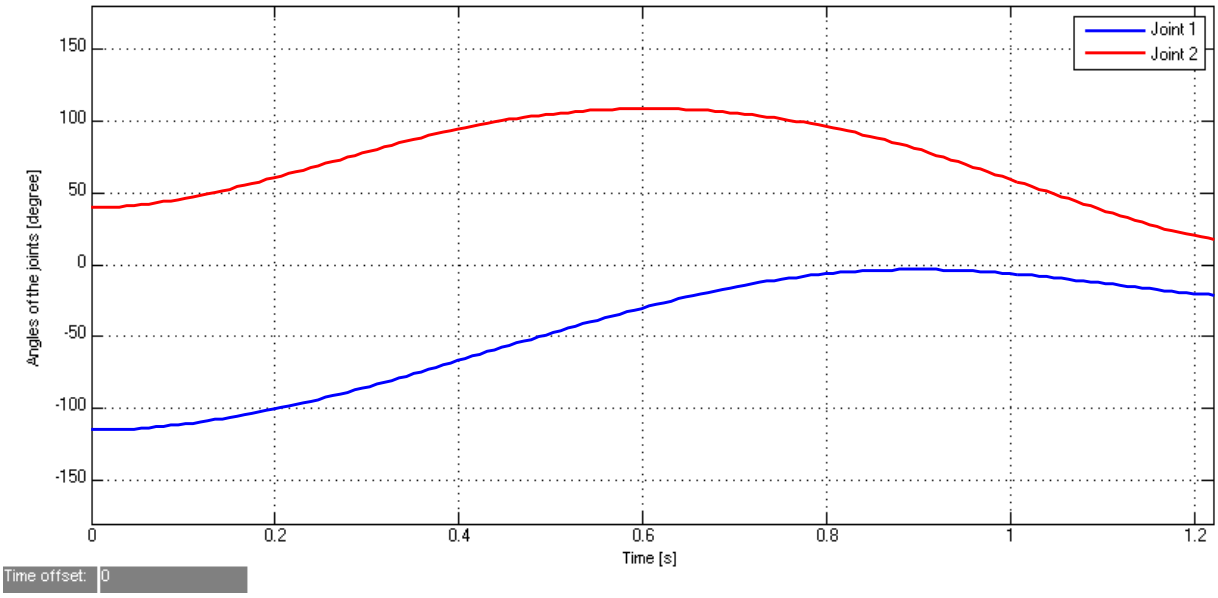
If we increment the maximum velocity, the robot can not hit the target, the waveform wil lbe modified. $Max\_vel = 1{,}8\left[\frac{m}{s}\right]$



4.23. Figure: Dynamic of the TCP in case of nearly points



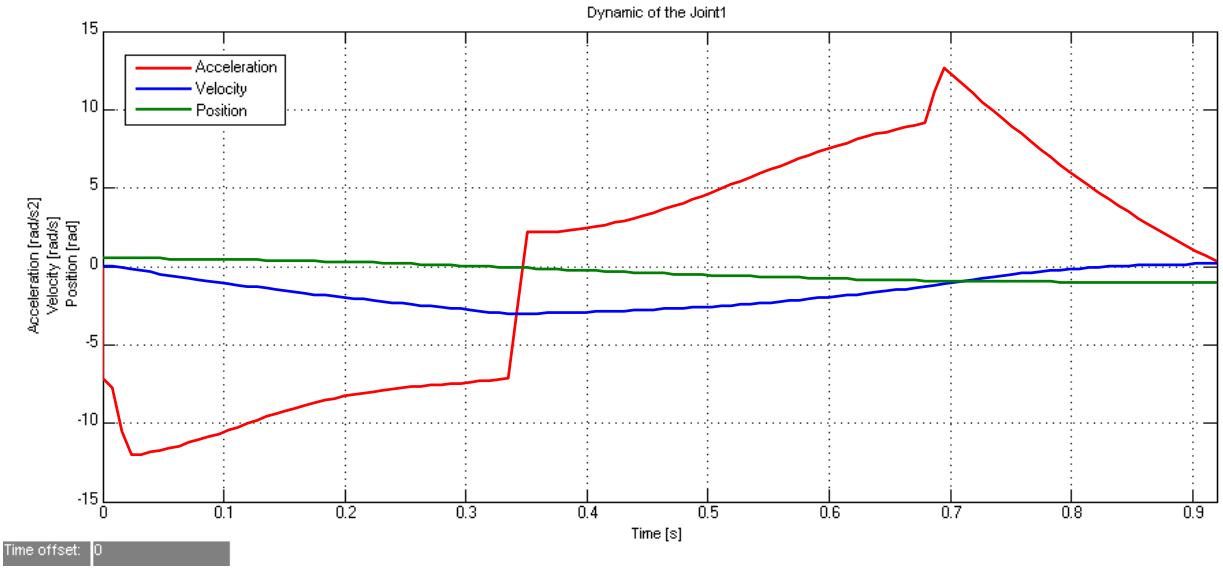
4.24. Figure: X,Y components of the position



4.25. Figure: Angle of the joints

Variable	Value	Unit
A	(0,35;–0,42)	$[m]$
B	(0,45;0,25)	$[m]$
Max_acc	3	$\left[\frac{m}{s^2}\right]$
Max_vel	1	$\left[\frac{m}{s}\right]$
$Safety\_ang$	20	$[\circ]$

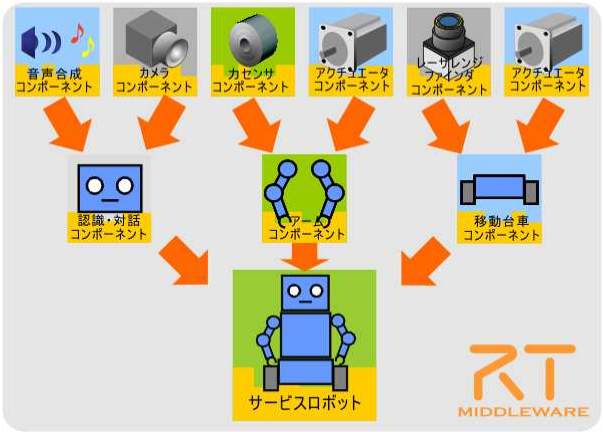
4.2. Table: The constants of the second test



4.26. Figure: Dynamic of the joints

4.4 RT-Middleware components

4.4.1 Overview of RT-Middleware



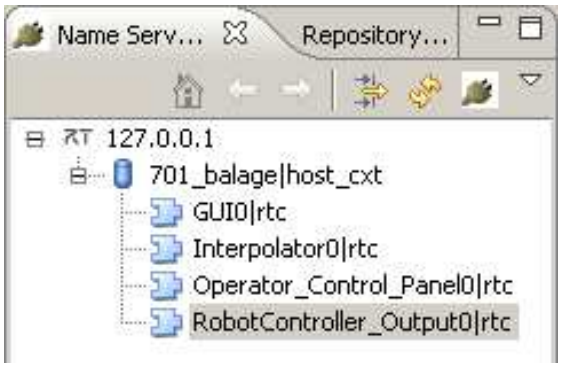
4.27. Figure: Structure of theRT-Middleware

Along with the expansion of the Internet, the number of research and development projects aiming at making robots and robot systems more intelligent by distributing their necessary resources over a network is increasing. Unfortunately, the development of such systems usually requires huge investments as well as a large amount of time. These are the main reasons why up until now we could not witness any success story of a commercial robot reaching a price to functionality ratio high enough to make it widely accepted on the market.

The RT-Middleware aims at establishing a common platform based on the distributed object technology and which would support the construction of various networked robotic systems by the integration of various network enabled robotic elements called RT-Components. [9]

In the RT-Middleware, indepedent software elements are organized into an RT system. These elements are called RT-Components. RT-Components are built using the RT-Component Framework mentioned above. Any system can be designed and realized by combining RT-Components, either by

reusing existing or developing your own RT-Components. All the functionalities necessary to manage systems of RT-Components are provided by a set of services, independently from the component themselves. The RT-Middleware also provides library to assisting the development of RT-Components. The technology supporting all the above mentioned functionality is what we call the RT-Middleware.



4.28. Figure: Name server

The system is based on network technology. That is why the most important benefit of the RT-Middleware system is flexibility. The operator can easily switch the components on or off. On the figures the blue boxes are the activated, the greens are the deactivated components. There is a name server, which can handle the data transfer between the components during the run. This software is based on CORBA. There can be five different languages of the components (C++, Java, Python, C#, Ruby). I used C++ for the development. It is a world-wide language with good support, it is perfect to develop test applications with graphical interface as well.

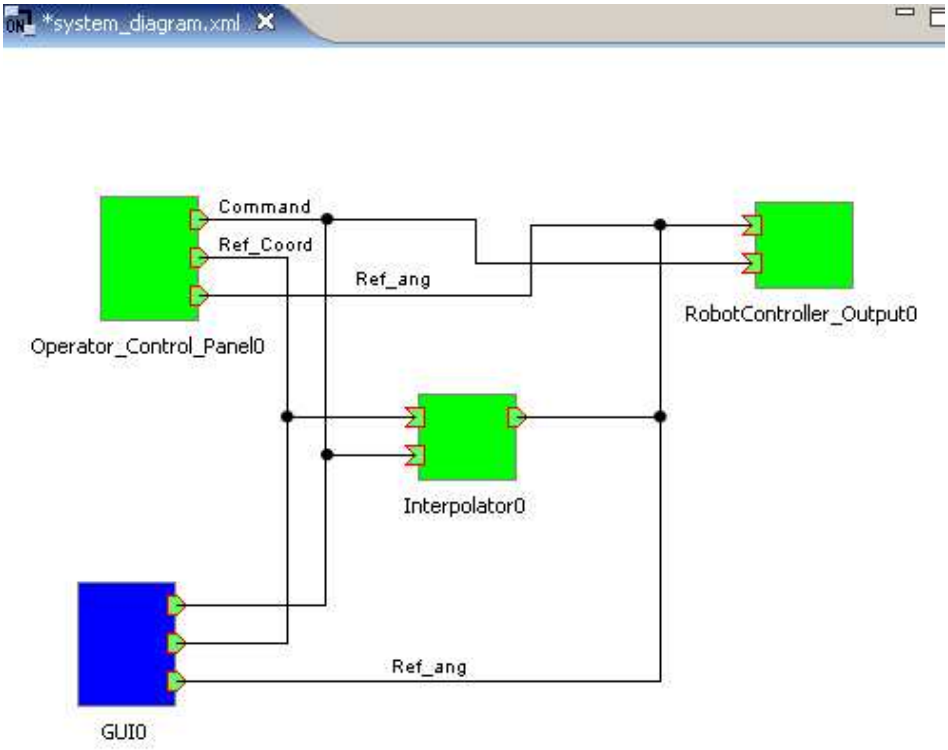
4.4.2 Components of the SCARA

At the first step of the development we have to separate the processes, which will run on the embedded system and which will run under RT-M. The RT-Middleware system is able to coordinate lower and higher levels of the control. It depends on the response time of the mechanical system and the frequency of the communication between the microcontrollers and the server computer, that is why I developed two systems.

Our hardware is now able to communicate via bluetooth and serial ports. This cut down on the number of possibilities. In this case the motion control processes need to be on the microcontroller, we can use the RT-M for the higher levels, like path planning, interpolation.

The RobotController unit is the communication part of the system. It can send basic setup commands and reference angle values to the DSP. The operator has two choices to input informations into the robot. First the Operator_Control_Panel unit is able to handle the signals of a standalone teach pad or joystick. It can create the translated data for the controller. The alternative version of these functions is a graphical user inteface GUI unit. It has a surface which attend to the on- and offline

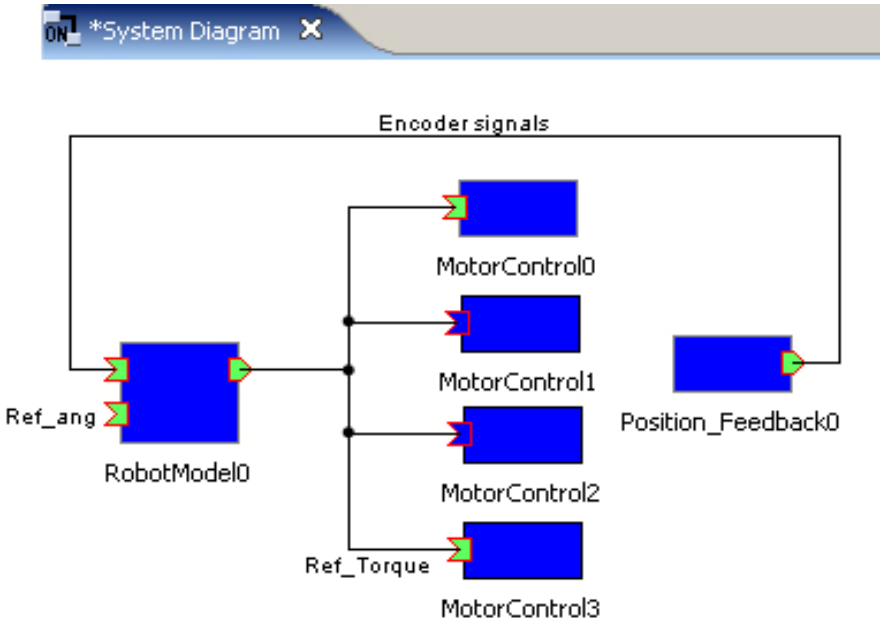
programing as well. Both units can send simple reference angles and coordinates in Certasian system too. To transform the



4.29. Figure: System diagramm 1

Ref_Coord values there is the interpolator. With the command port can we choose the type of the inteprolation (linear, arc, spline) and the orintation of the arm. The outputs are timed angle values.

If the problem of the communication speed with the server computer is not serious, we can create a more flexible system; for example we can connect an assistant motor to the robot or it can easily work with an other robot in the same area. If we use two different robots with two different control software we can not guarantiee perfect and safe results. For this we need an overall system like RT-Middleware In the figure 4.30. we can see the continuation of the previous system. In this case Robot Model unit gets the reference angles and can calculate with the whole dynamic model of the robot, can set the parameters of the external control loops. This box gets the feedback information as well. The outputs are torque reference values for the MotorControl units. They can handle the current control of the motors. The current control is the fastest control loop; it is needed to develop to the embedded system every time, that is why it is a separated unit here.



4.30. Figure: System diagramm 2

5 CONCLUSIONS

The aim of this project was to design and create a universal robot controller for the Adept SCARA 604-S robot of the Hogskole I Narvik to resuscitate it. During the semester we solved every important problem from our main goals. The robot worked with external trajectory source, with G code and with self interpolator module as well. The most important communcation interfaces were tested, the motion controller part gave good and stable answers for the excitations. In this form the controller is capable of continous motion controlling, trajectory making, etc. It can be used for further researches on the motion control of SCARAs. At the end of the project the robot can do everything that the original one was capable of. Videos of the moving robot can be seen on our website: http://www.generalmechatronics.com/Robot Controller_hun.htm

5.1 Universal Input/Output Card

The universal I/O card works correctly, the isolated voltage levels on the card were a huge help during the development, without them the number of the unpredictable problems would be higher. The most important parts, the high speed optocouplers were a good choice; they gave perfect signals back on high frquency as well.

5.2 Simulation

I translated the interpolation software to the language of the DSP, after the MATLAB model, it was tested with the robot. The movement of the arm was similar like in the simulation. It worked perfectly together with the motion control function. To design trajectory making really needed a simulation envoirement, like Simulink-MATLAB, it was much easier to solve coordinate transformation and signum problems with it.

5.3 RT-Middleware components

The main goal of the the RT-Middleware components development was to make our controller able to be inserted into unviversal envoirement, and testing this new technology. I developed two possible systems for the controller, one with motion control loop, and one without it. Via our communcation interfaces (Bluetooth, serial) the RT-M system worked well. In this case the motion control loop was developed on an embedded system, but the bandwidth was not enough for the second version. The RT-M system seems to be a good envoirement for connecting really different robots or NC systems into one overall platform.

6 FUTURE PLANS

6.1 Universal Input/Output Card

During the work with the I/O card were two significant problems, which would be good to solve in the next versions of the controller. First the connection to the robot should be more flexible. This hardware needs a special cable, it is a problem because the number of the wires is over fifty. It is hard to insert new wires; and it is hard to mesure the voltage on the pins. The second problem is that the encoders of the robot could be of different types, like open collector, tote-pole, or sometimes the motors have other type of position feedback, like analog. In the second version of the card it would be important to solve these problems, that way it could be really universal.

6.2 Simulation

The interpolation simulator needs to know other type of interpolations in the future, like arc, spline, etc. By this SCARA, the simulation of the movement did not require 3D simulation, but in case of a humaniod or another more complex structure it is impossible to design trajectory without the third dimension. The MATLAB-Simulink has a module, which can visualise and move 3D parts from general CAD softwares. In case of more complex structures we need to

use it. Now the robot has a safety unit, but this unit can handle only the borders of the robot, it does not know the envoirement, usually there are some other devices too. For the safer movement we need to type in the coordenate of the other parts, or install a camera visualisation system.

6.3 RT-Middleware components

To test the other possibilities of the RT-Middleware, we need faster communication protocols between the controller and the server computer. We can get more results from the system if we have more components, among others hardwares and components from the developers of the RT-Middleware.

7 LIST OF REFERENCES

[1] Bence Kovács “Development of an Universal Robot Controller: Development FPGA Hardware and Software, Modelling of a SCARA Robot” 2009

[2] Géza Szayer “Development of an Universal Robot Controller: Hardware Development, Designing and Implementing Central Motion Control” 2009

[3] NXT SCARA (Two-Link Planar Robot Arm) Controller Design, [Online], Available: http://www.mathworks.com/matlabcentral/fileexchange/22126

[4] Somló J., Lantos B., P.T.Cat: Advanced Robot Control (to be published by the Akadémia Kiadó. Budapest

[5] Datasheet of ACSL-6400 [Online], Available:

http://www.chipcatalog.com/Datasheet/B342FDEA977A0C0843895A35B4D5FAF5.htm

[6] Datasheet of AT90CAN32 [Online], Available:

http://www.atmel.com/dyn/resources/prod_documents/7679s.pdf

[7] Datasheet of MAX232E [Online], Available:

http://www.datasheetcatalog.org/datasheet/maxim/MAX202E-MAX241E.pdf

[8] Simulink - Simulation and Model-Based Design, [Online], Available:

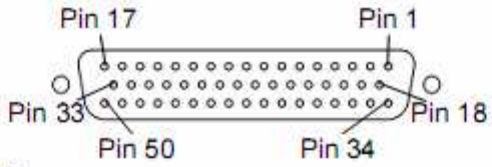
http://www.mathworks.com/products/simulink/

[9] RT-Middleware Homepage, OpenRTM-aist, [Online], Available:

http://www.is.aist.go.jp/rt/OpenRTM-aist/html-en/What20is20RTM.html

8 APPENDIX

1. Pins of the robot's connector



Encoder connector (DD-50):

1	Encoder Joint 1	CON12/9	Ground	Black
2	Encoder Joint 1	CON12/10	A Channel	Red
3	Encoder Joint 1	CON12/7	Ground	Black
4	Encoder Joint 1	CON12/8	B Channel	Blue
5	Encoder Joint 1	CON12/5	Ground	Black
6	Encoder Joint 1	CON12/6	Z Channel	Yellow
7	Encoder Joint 2	CON6/10, CON21	Ground	Black
8	Encoder Joint 2	CON6/11	A Channel	Red
9	Encoder Joint 2	CON6/8	Ground	Black
10	Encoder Joint 2	CON6/9	B Channel	Blue
11	Encoder Joint 2	CON6/6	Ground	Black
12	Encoder Joint 2	CON6/7	Z Channel	Yellow
13	Encoder Joint 3	CON5/11, CON7	Ground	Black
14	Encoder Joint 3	CON5/12	A Channel	Red
15	Encoder Joint 3	CON5/9	Ground	Black
16	Encoder Joint 3	CON5/10	B Channel	Blue
17	Encoder Joint 3	CON5/7	Ground	Black
18	Encoder Joint 3	CON5/8	Z Channel	Yellow
19	Encoder Joint 4	CON4/10, CON22	Ground	Black
20	Encoder Joint 4	CON4/11	A Channel	Red
21	Encoder Joint 4	CON4/8	Ground	Black
22	Encoder Joint 4	CON4/9	B Channel	Blue
23	Encoder Joint 4	CON4/6	Ground	Black
24	Encoder Joint 4	CON4/7	Z Channel	Yellow
25	Infra sensor (Joint 1)	CON13/7		Transparent
28		Connected to 48	Ground	Black
29	Infra sensor (Joint 2)	CON5/1, CON23		Transparent
33	Infra sensor (Joint 3)	CON5/2, CON8		Transparent
36	Infra sensor (Joint 4)	CON4/1, CON24		Transparent
43	End switch	CON13/4 (Joint 1), CON12/2 (Joint 1), CON6/3 (Joint 2), CON5/4 (Joint 3)		White
44	End switch	CON13/1 (Joint 1), CON12/1 (Joint 1), CON6/2 (Joint 2), CON5/3 (Joint 3)		Blue
45	Power supply (Joint 1)	CON13/6, CON12/4	5 volt	Red
46	Power supply	CON6/5 (Joint 2)	5 volt	Red
47	Power supply (Joint 2, Joint 3, Joint 4)	CON5/6, CON7, CON8, CON22, CON23, CON24	5 volt	Red
48	Power supply (Joint 1)	CON13/5, CON12/3	Ground	Black
49	Power supply	CON6/4 (Joint 2)	Ground	Black
50	Power supply (Joint 2, Joint 3, Joint 4)	CON5/5, CON7, CON8, CON22, CON23, CON24	Ground	Black

2. Th whole MATLAB-Simulink model of the interpolator