Norwegian University of Life Sciences
Faculty of Science and Technology
Department of Mechanical engineering and technology management

2022

Technical report

ABB RobotStudio and YuMi

Authors: Peder Ørmen Bukaasen, Bård Tollef Pedersen and Lavanyan Rathy – Group 8

Abstract

Peder Ørmen Bukaasen, Bård Tollef Pedersen, Lavanyan Rathy

ABB RobotStudio and YuMi, 10 pages

Norwegian University of Life Sciences

Faculty of Science and Technology

Technical report 2022

Instructors: Alireza David Anisi, Norwegian University of Life Sciences

The purpose of the project was to design an application for ABB IRB 14000 YuMi robot. The thesis work was made for the course Introduction to robotics at Norwegian University of Life Sciences. Computer-aided design was conducted using SolidWorks, ABB rapid programming, and virtual simulation were conducted in ABB RobotStudio.

There were many applications and ideas that came up when trying to find the best use of the truly collaborative robot. But the idea of the tic-tac-toe application was the most fitting.

This thesis describes the whole application process from learning ABB RobotStuido, the theory behind, program, virtual simulation and even a test on the YuMi robot. After completing the application, the end-user could run the application without set-up, pre- knowledge of robotics or programming.

The results of the project can be helpful for further learning of ABB RobotStudio, robotics, and further development of the application.

Keywords: collaborative robot, ABB YuMi, rapid program, ABB RobotStudio

Table of contents

1 Introduction

The YuMi project is a task given in the course TEL200. The project is building on the theory of robot arms. YuMi project is divided into three parts where the first part is to learn the basics of RobotStudio software and get to know the functions. The second part is to make the YuMi robot push down a button, and then spin it open again. The last part of this project is to use the skills learned from the earlier parts and make a creative solution. On the final part we could be creative, so we made the YuMi robot play tic-tac-toe as part of the solution.

2 Theory

When talking about robot arms one needs to understand the basic terminology, when referring to the base of a robot arm, we mean the link of the arm that is connected to the ground. Joints in a robot arm are the parts that allow motion of two links, like the joints in our bodies. A link in a robot arm is all the rigid parts of the arm, like our forearm or upper arm. The end effector of a robot arm is the link of the robot arm that holds the tool (Anisi, 2022).

2.1 Robot pose

In robotics we often need to represent position and orientation, this can be done in several ways, therefore we introduce poses. Poses can describe the correlation between different coordinate frames. Poses consist of both a translational component and a rotational component. If we have a pose consisting of two coordinate frames in 3 dimensions, the translational component is then the position of the second coordinate frame with respect to the first coordinate frame. The rotational component is the rotation of the second coordinate frame with respect to the first coordinate frame. To get the positive direction of rotation for the coordinate frame, you can use the right-hand rule where you point your thumb along the positive direction of the x, y, or z axis, and curl you other fingers around it, the direction your fingers are pointing are curling is the positive direction of rotation. Rotation in robotics can be described in many ways. Rotational matrixes are one of them, the rotational matrix is a 3x3 matrix that describes the rotation of one coordinate frame with respect to another coordinate frame (Corke, 2017).

2.2 Kinematics

Kinematics is a part of mechanics where you look at the movement of a body or system of bodies without considering mass and forces. This is widely used in robotics when studying robot arms, since they consist of several links and joints. All these joints have one degree of freedom, either rotational or translational. Translational joints are also called sliding or prismatic joints and rotating joints are also called revolution joints. Kinematics in robotics is

used to get joint coordinates or the configuration of a robot arm to the position of the end effector. When calculating forward kinematics there is always one solution since you are going from joint coordinates to a pose. Inverse kinematics is mapping a pose to joint coordinates or configurations of a robot arm. When solving a pose with inverse kinematics one gets several solutions since a robot arm can get to the pose with several joint configurations (Corke, 2017).

2.3 Dynamics

Robot dynamics is the relationship between the forces on the robot and the acceleration they create. The purpose of dynamics is to determine joint forces/torques. The robot mechanism is usually modelled as a rigid-body system, which means that robot dynamics is the application of rigid-body dynamics to the robot. Forward dynamics is known as “direct dynamics”, and the main problem here is that the forces are given, but the accelerations need to be worked out. Inverse dynamics is the opposite where the accelerations are given, and the forces need to be worked out (Featherstone, 2007).

2.4 Paths and Trajectories

When robots move, they follow a path or a trajectory. A Trajectory tells us not only how to move from A to B, but how quickly we should move along the path and at what time we should be at what point along the path. The goal is to have a smooth path. The reason is to reduce the peak acceleration of the robot which will reduce the size of motors needed. This can also reduce the vibrations in the robot structure. Two of the most normal trajectory calculations are polynomial and trapezoidal. Polynomial, often quintic polynomial, has six different coefficients which let us specify the first and final position, first and final velocity and first and final acceleration. We can then write this as a simple matrix relationship. This gives us the values as a function of the trajectory time. Trapezoidal, here as the name says, it follows a trapezoid. It has an acceleration phase which follows a coasting or constant velocity phase, which follows a deceleration phase. (Corke, 2017)

2.5 YuMi Robot

In this course we have been working with a YuMi-IRB14000, this robot was introduced in 2015 by ABB as the world's first truly collaborative robot.
The YuMi robot can be described as an over actuated cobot. Cobot means it can work alongside humans without any safety concerns. This is because of the inherent safety. The joints and links have a soft cover as well as low force. It is an over actuated articulated robot because of its 7-axis. An over actuated robot is a robot that has more actuators than degrees of freedom. The articulated part means that the robot has one or more rotary joints (ABB, n.d.).

2.6 Tic-tac-toe

Tic-tac-toe is a simple game played by two players. The board consists of a 3x3 matrix, and you take turns placing pieces. The game ends when either you or your opponent gets 3 of their pieces in a row.

3 Method

3.1 RobotStudio basics

Before being able to accomplish the YuMi application as well as the YuMi challenge the basics of RobotStudio needed to be learned. RobotStudio is a software from ABB designed to make robot programming simpler and faster. Some of the basics consist of: Opening and saving a solution. Importing YuMi robot and tools from ABB Library. Importing and positioning of (CAD) geometries. Creating and modifying WorkObjects and Targets. Changing, creating, and moving along paths. Generating, showing, and changing RAPID code. Play and Record Simulation.

All the basic knowledge was learned by watching and following videos. Some of the videos are from ABB, the maker of the program, and some are from University of Skövde. The two sides of the videos can be found here:

- Tutorials for RobotStudio
- RobotStudio Tutorial Video Library

3.2 YuMi Applications

The first thing needed was to download a Unpack&Go file from Anisi. This file is the backbone of the project as it holds the ABB YuMi IRB400 robot as well as two Virtual SmartGripper tools and an E-Stop button. It also has all the necessary signals to run on the real life YuMi robot.

The next step was to create a mechanism to model an E-Stop button moving up/down. Creating a mechanism was done by pressing “Modeling” followed by “Create Mechanism” and then choosing name, links etc. When trying to add links to the mechanism, the E-stop components did not show up. To fix this we first had to export the parts and then import them back in.

Then it was to create a “Smart Component” from the E-Stop Mechanism. Smart components use signals to move, in this case the button up and down. This will be used in the simulation, so it looks like a robot is pressing the button. Smart component was created by pressing

"Modeling" followed by "Smart Component.". The most important thing is to create an input and output signal. We choose the input signal to be called ButtonUp and the output to be ButtonDown. Now when calling the ButtonUp signal, the button will shift position from up to down. Then we added a signal to the I/O System. This will make it easier to use the button later in the code.

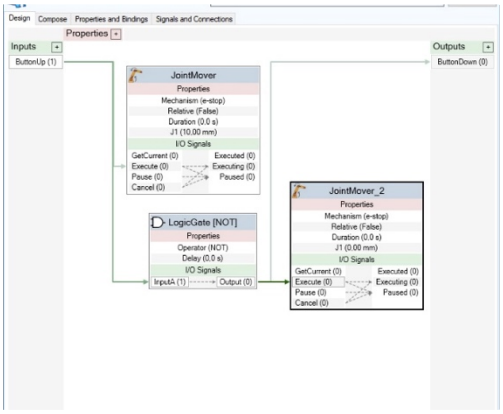

Figure 1 shows Smart Component design of E-stop Button in ABB RobotStudio

The next step was then to add another signal to the I/O System. This will be the signal to call on in the RAPID code. This was done by going to the I/O System editing page. Then right click on the signal name and "New signal." Its name is di_EmergencySituation, this name is important as it is the only name the physical robot will recognize.

The last step is to add the targets and write a script the YuMi robot will recognize. Targets and paths are all part of basic knowledge. When done creating the path, simply right click and synchronize too RAPID. This did most of the coding and we just had to make the final changes. This was done by learning some RAPID code from the ABB manual.

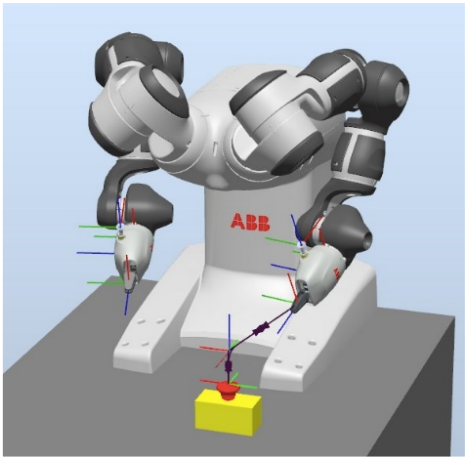

Figure 2 shows YuMi robot with path and targets in ABB RobotStudio

After the code ran without any errors in RobotStudio Simulation, we tried it on the real YuMi robot. We got a lot of help when uploading the code to the physical robot, but luckily to your amusement the code worked.

3.3 Tic-Tac-Toe YuMi Challenge

The first step was to make the parts, this was done using SolidWorks. It takes only three types of parts to make the tic-tac-toe board as well as the game pieces. We needed the "O," the "X," and then a frame, the frame is made from four sticks that assemble like the Full-Scribe building method. After the parts were finished, they were exported as STL files and imported into RobotStudio. After they were imported, we needed to position them. Positioning the components is done by selecting each part and using "set position" to move and orient the object

to the wanted position and orientation. To make these objects pickable we had to set the physics of them to either kinematic or dynamic, we chose kinematic, to minimize the risk of objects bouncing around because of minor collisions.

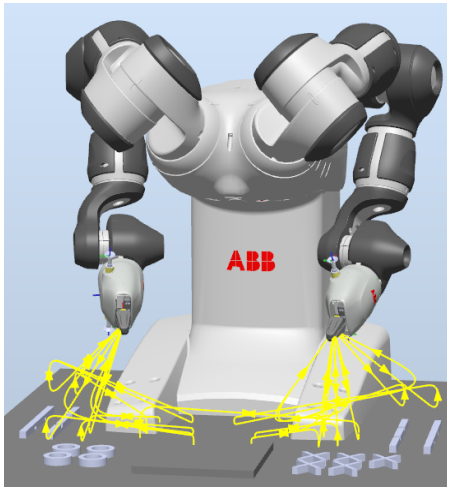

Figure 3 shows YuMi robot with Paths in ABB RobotStudio. Tic-tac-toe application

Then it is just to do the same as the YuMi application part, same procedure. Creating targets on all the pieces and all the places the tic-tac-toe pieces should be placed. Then creating a path that moves to the different targets, the right configurations for the robot arm were found by trial and error, since the auto configuration function did not always work. For dealing with singularities, we used SingArea\wrist, this function slightly changes the orientation of the tool to pass the singular point. Then the paths and targets were synchronized too RAPID. In RAPID the code for the grippers was added, the functions g_GripIn, g_GripOut, and g_MoveTO, were used. The g_MoveTO function was used when the arm was closer to other objects, so the fingers did not collide with the board and other pieces. We also utilized the WaitTime function to make sure that the grippers had time to grip and let go of the object at the respective targets. Since the simulation uses both arms, we created two signals, do_arm_move_L and do_arm_move_R,when the do_arm_move_R was activated, the right arm moved likewise with do_arm _move_L. To set these signals we used the set and reset functions. These signals were both digital output signals, with default value 0. The access level of this signal was set to all, so that we could set and reset it in RAPID.

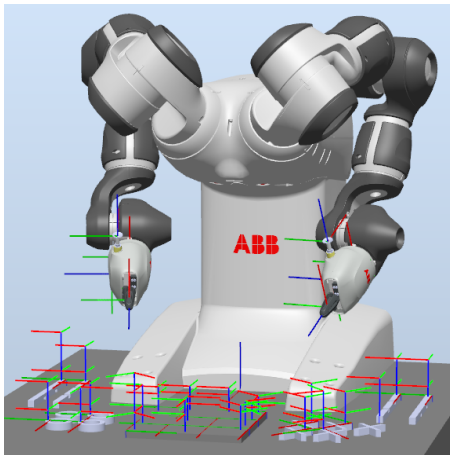

Figure 4 shows YuMi robot with targets in ABB RobotStudio. Tic-tac-toe application

```
24  PROC Main()
25
26      WaitDo do_arm_move_R,1;
27      Path_10;
28      Set do_arm_move_L;
29      Reset do_arm_move_R;
30      WaitDO do_arm_move_R,1;
31      Path_20;
32      Reset do_arm_move_R;
33      Set do_arm_move_L;
34      WaitDO do_arm_move_R,1;
35      Path_30;
36      Reset do_arm_move_R;
37      Set do_arm_move_L;
38      WaitDO do_arm_move_R,1;
39      Path_40;
40      Reset do_arm_move_R;
41      Set do_arm_move_L;
42      WaitDO do_arm_move_R,1;
43      Path_50;
44      Reset do_arm_move_R;
45      Set do_arm_move_L;
46
47      ENDPROC
```

```
PROC Path_50()
    g_MoveTo(5.5);
    MoveL Target_40,v1000,z100,Servo\WObj:=wobj0;
    SingArea\Wrist;
    MoveL Target_240,v1000,z100,Servo\WObj:=o_3;
    WaitTime(1);
    g_GripIn;
    WaitTime(1);
    MoveL Target_250,v1000,z100,Servo\WObj:=o_3;
    MoveL Target_270,v1000,z100,Servo\WObj:=o_3_base;
    MoveL Target_260,v1000,z100,Servo\WObj:=o_3_base;
    WaitTime(1);
    g_MoveTo(5.5);
    WaitTime(1);
    MoveL Target_270,v1000,z100,Servo\WObj:=o_3_base;
    MoveL Target_250,v1000,z100,Servo\WObj:=o_3;
    MoveL Target_40,v1000,z100,Servo\WObj:=wobj0;
    g_GripIn;
ENDPROC
```

Figure 5 and Figure 6 shows Rapid code to Tic-tac-toe application for Right arm and Path_50 in ABB RobotStudio.

4 Results

As results to the RobotStudio basic we learned the basics and can manage RobotStudio on an Intermediate level. In this part we learned the basics of robot arm movement, and their limitations. This knowledge was particularly important to learn before trying part 2 and part 3 of this project.

On the YuMi application we managed to make the robot follow and move along a path as well as open and close the gripper. The robot did as we wanted and managed to press the button down as well as grip and rotate the button, making the button pop up.

On the Tic-Tac-Toe YuMi Challenge the robot managed all the things as in the application part, as well as place the board correctly, then play a simple game. The simple game is predefined and will have the same outcome every time. In this part we learned how to pick and place objects in RobotStudio.

5 Discussion

5.1 Artificial intelligence

After making the simulation, we saw the potential for further development. What makes a robot more powerful is an ability to think and make decisions on its own. The robot is currently using a pre-programmed simulation, but the idea of using artificial intelligence (AI) can really boost the efficiency of the YuMi-robot. In general, if a system has a feedback controller, the efficiency increases highly. In our case, the robot is going to play tic-tac-toe, and there are many algorithms we can choose from. Our idea is to have the two arms play with each other with different algorithms. The two algorithms we chose were Minimax and Alpha Beta search.

The Minimax algorithm is based of two players playing, and the AI will always take the outcome with “Max” score. Using this method, the robot will either draw or win the game. With the Alpha Beta search algorithm, the AI will evaluate all the possible consequences of each move it makes and choose the outcome that will lead to a win or a draw. Using YuMi we can see what happens when two perfect AI clash against each other.

5.2 YuMi application.

We could improve the simulation by making the button rotate as it gets pressed down. This could be done by adding a rotational joint to the button smart component. Then it would both rotate and slide down into the down position.

We could also use both hands. One to hold the button base and the other to press and lift the button.

5.3 YuMi challenge.

The game is predetermined and as an improvement we would make the robot play using randomness or as mentioned above, artificial intelligence. We could also make the robot play with user interface. This would combine AI, the robot and the robots collaborating function. So, when the human places a piece, the robot uses a camera to see where it is placed. Then use AI to choose its move. Then repeat this process to either the robot or the human win. To make this happen we would need some way to read which squares on the board that is occupied, this could be done by machine vision or with pressure plates on each of the squares. One would also need paths that could move to all the nine squares for both the x's and the o's.

Improvements could also be made to the board since the grippers touched the board parts in some paths. These pieces could have been made longer so that the individual squares would have been bigger. This would make the path creation easier and ensure us that the grippers would not collide with other parts.

6 Conclusion

The conducted project was finished by using CAD model of the parts, the RAPID program language, and the ABB RobotStudio software. Where the goal of the project was to learn the basics of ABB RobotStudio as well as programming the YuMi robot to both press an emergency button and play a game of tic-tac-toe. There were a few challenges, most of which were resolved around ABB RobotStudio. In the end the goal was accomplished as the group intended.

References

Anisi, A. D. (2022). Lecture notes - TEL200.

Corke, P. (2017). Robotics, Vision, and Control. Springer.

ABB. (n.d.). YuMi® - IRB 14000 | Collaborative Robot. Hentet April 2022 fra https://new.abb.com/products/robotics/collaborative-robots/irb-14000-yumi

Featherstone, R. (2007). Robot dynamics. Hentet April 2022 fra http://www.scholarpedia.org/article/Robot_dynamics#:~:text=Robot%20dynamics%20is%20concerned%20with,rigid%2Dbody%20dynamics%20to%20robots.