

ABB Robot and its own programming language

LAB University of Applied Sciences
Degree of Bachelor
Mechanical Engineering and Production Technology
2021
Toan Nguyen

Abstract

Author(s) Toan Nguyen	Publication type Thesis, UAS Number of pages 43	Completion year 2021
Title of the thesis ABB Robot and its own programming language		
Degree Mechanical Engineering and Production Technology		
Instructors Jouni Könönen, LAB University of Applied Sciences		
Abstract <p>The goal of the research was to introduce ABB Robot and its own programming language. The objective of this project was to provide a basic knowledge about ABB Robot and its own programming language, specially focus on IRB 2400 – a type of ABB Robot. Besides, through practical exercises, students will initial interact with programming on industrial robots, that would be helpful for working in the future.</p> <p>Thesis work would be divided into theory part and practical part. The theory would be the basic introduction about ABB Robot and its own programming language, then focus on the structural principle, the way of the operation of the robot IRB 2400. The data for this thesis were collected from literature discussing this topic and manufacturers' operating manuals. The information about functions and features of this robot were obtained from manufacturers' operating manuals. The practical part would be some exercises programming on ABB robot studio, or Visual Components Premium. Learning about the robot's equipment and how to utilize the ABB robot studio, or Visual Components Premium, was the first step. After that, some tasks would be created, which is combined with machine vision and product identification.</p> <p>The result of this thesis might be helpful for further research of industrial robot, also helpful for working in the future.</p>		
Keywords ABB Robot, ABB Robot Studio, Visual Components Premium, IRB 2400, RAPID.		

Contents

1	Introduction.....	1
1.1	Goal.....	1
1.2	Objective	1
1.3	Structure.....	2
2	Theory	3
2.1	Basic overview of ABB Robot	3
2.2	Robot IRB 2400	5
2.2.1	Structure.....	5
2.2.2	Various robot models and specifications.....	7
3	Operating principles.....	11
3.1	Overview of operating principles	11
3.2	The controller.....	11
3.3	The manipulator.....	13
3.4	Communication.....	16
4	Programming language – RAPID.....	17
4.1	About RAPID	17
4.2	The data of RAPID.....	17
4.2.1	The Variables	17
4.2.2	RAPID's Persistent variables	18
4.2.3	RAPID Constants	18
4.2.4	Types of operators in RAPID	18
4.3	Move instructions of RAPID robot functionality	19
4.3.1	The MoveL instruction	19
4.3.2	Coordinator systems.....	20
4.3.3	Other move instructions.....	21
5	ABB Robotstudio	22
6	Visual Components Premium.....	23
7	Exercises	24
7.1	Pick and Place.....	24
7.2	Palletizing	32
8	Conclusion.....	38
	References	39

1 Introduction

Robotic automation is now widely used in a variety of fields, and its use is expected to grow in lockstep with technological advancements. There is no exception to industry, robots have been used in industry since they were invented. One of them, ABB Robot is an industrial robot product, provided by ABB Robotics – Switzerland's leading supplier of robotics and automation solutions. ABB automation robots are often used in metal processing, assembly, painting, packaging, shipping and finishing.

In addition to technology robotics, ABB Robotics also provides robotics software, engineering services, and modular manufacturing stations in the aforementioned applications in markets such as automobiles, metalworking, electronics, food, beverage, and so on. Not only providing equipment, ABB Robotics also provides competitive solutions to help business improve productivity, improve product quality, and create a safe working environment for workers.

1.1 Goal

The aim of this study is to get knowledge about ABB robot, along with some hands-on exercises on Robot studio or Visual Components Premium. Because we do not have ABB robot in our campus, so the exercises should be performed successfully in the virtual simulation. The purpose of the first task is to make a new program to pick up boxes and set them into a new position. The goal of this work is to come up with a solution that uses variables for location and requires a few new positions to be added to the position list. The aim of the other task is to design and implement a palletizing program for the robot.

1.2 Objective

The study's first goal is to reveal a fundamental understanding of the ABB robot (IRB 2400). The study includes information on the robot's structure, functions, and features, as well as Robot Studio software and Visual Components for simulation.

The second goal is to design and implement some program for the robot, that were programmed on Robot studio software, or Visual Components Premium.

The last purpose is to have critical thinking ability, problem-solving skill and able to effectively manage time when faced with multiple technical challenges.

1.3 Structure

There are three main sections to this research. The first section is theoretical and provides an overview of the ABB robot, as well as its own programming language and Robot studio software. Especially, focusing on the robot IRB 2400.

The second part is about exercises. As noted above, because we do not have the ABB robot in our campus, so this study just could be performed on Robot studio software, or Visual Components Premium.

The final section, "summary and discussion," will be a conclusion for the study, considering what development might be possible.

2 Theory

2.1 Basic overview of ABB Robot

ABB robot is used as one of the leading advanced automation solutions in the field of mechanical machining today. With ABB automation robot, all stages in the production line are done quickly and easily, reducing time and saving costs. The introduction of the ABB automation robot marked a remarkable development in the field of automation thanks to the outstanding advantages of this product compared to the machines in the traditional production line that are outdated and out of date. Some types of ABB Robot are shown in Figure 1.



Figure 1 shows the ABB automation robotics (Product specification, IRB 2400, 3HAC 5672-1, cover page).

The development of automation technology gradually replaces the outdated and incapable of meeting the modern production needs. Automated robots bring many benefits to the manufacturing industry in general and in the mechanical machining sector in particular. For

example, with ABB welding robot – one of the most used robot in the mechanical machining sector, the biggest advantages are:

- **High speed:** Robot has welding speed many times faster than human manipulation. In addition, the welding cycle of the robot is always maintained, making welding process continuous and uninterrupted. With welding robots, business can safely calculate in advance the performance as well as time to finish the final product. In addition, welding robots also save time because the welds are clean after welding.
- **Improve product quality:** Because the robot moves consistently, the output product will be completely homogeneous and free from the same errors as when done by workers. Pieces in the same location, moved at the same speed, will produce the same welds. There are no risks of deformation, products such as casting the same mold give customers absolute peace of mind about product quality.
- **Material saving:** Because the goods are of consistent quality and have no serious faults, the welds will not be discarded for failing to meet the specifications. This helps to save materials as well as the investor's budget. In addition, the consumable substances in the welding process such as molten gas, welding gas, and filler metal will be reduced due to the faster working speed of the robot.
- **Reduce labor costs:** Human replacement robots help investors reduce labor costs and can run continuously for an hour without human supervision or control. The robot also does not need to take breaks and breaks between hours, but can work continuously, bringing the great productivity to the job.
- **Improving working environment:** The automatic welding robot helps people performing complicated and dangerous jobs, reduces the risk of having to work in a hazardous environment, bringing people to a new role of adjusting and supervising work.
- **Increase the brand's competition:** A brand wants to attract more customers, when producing finished products, it is necessary to meet both productivity and quality of international standards, and at the same time lower the cost of the product to increase competitiveness. With ABB automatic welding robot, the production system in the factory will no longer depend on people, improve the product quality evenly, improve productivity and maintain system stability (WELDCOM, n.d.).

2.2 Robot IRB 2400

2.2.1 Structure

The IRB 2400 robot is a high-performance robot created especially for process applications requiring extremely high accuracy. Inverted mounting is available on all models. The IRB 2400's lightweight nature makes it simple to install.

The robot's design and use of few parts lead to high reliability and long maintenance cycles (ABB Group, n.d.).

The basic structure of the IRB 2400 is outlined below:

- **Robot family:** The IRB 2400 is an industrial robot with six axes that was created for manufacturing companies that rely on Robotic automation with a lot of flexibility. The robot has an open framework that has been specifically adapted for versatile use, and it can interact with external systems in a variety of ways.
- **The operating system:** IRC5 controller and robot control software – RobotWare are included with the robot. Every element of the robot system is supported by RobotWare, including motion control, application program creation and execution, communication, and so on. see IRC5 controller with FlexPendant Product Specification.
- **Safety:** Full robot, manipulator, and controller safety requirements are applicable.
- **Additional functionality:** Optional application support software – such as gluing and jointing, aspects of networking – such as network communication and advanced functionality like as sensor monitoring, multi-tasking, and so on – can be added to the robot for added functionality. See Product specification – Controller software IRC5 for a full list of optional software.
- **Foundry plus:** is intended for use in harsh environments where the robot is subjected to coolant, lubricant, and metal splits, such as those found in die casting and other similar applications. Typical applications include die-casting system, spraying insertion and component extraction, sand casting and gravity casting handling, and so on. For applications in the foundry, as well as other applications fields, special attention must be paid to the operation and maintenance requirements. If you have any questions about the Foundry Plus robot's suitability for a particular application, please contact ABB Robotics Sales. For corrosion safety, over a primer, a two-component epoxy is applied to the Foundry Plus robot. Additional rust preventative is applied to exposed and critical areas to further enhance corrosion resistance, such as the tool flange, which has a special preventive coating. However, Rust can form

on the robot's unpainted sections, joints, and other unprotected surfaces if water or other rust forming fluids are constantly splashed on them. It is suggested that a rust preventative be added to the fluid or that other steps be taken to avoid rust formation on the listed. The robot as a whole is IP67-rated, this indicates that the electrical compartments are protected from water and solid pollutants, from base to wrist (except IRB 2400L, which is IP67 only on connectors and wrist). Both sensitive pieces, for example, are better secured than in the regular bid. Features of Foundry Plus that should know about:

- To achieve IP67, improve sealing to avoid intrusion into cavities.
- Additional cable and electronics security.
- Cavities are covered by special covers.
- Connectors that have been known to work.
- On screws, washers, and unpainted/machined surfaces, use rust preventatives.
- Surfaces and maintenance program have been extended.

Cleaning the Foundry Plus robot is possible with the right washing equipment. “Foundry” (IRB 2400 F/L) or “Foundry Plus” (IRB 2400F/16 and F/10) are the labels on the robot (ABB Group, n.d.).

- **Manipulator axes:** the manipulator axes are shown in Figure 2 and Table 1.

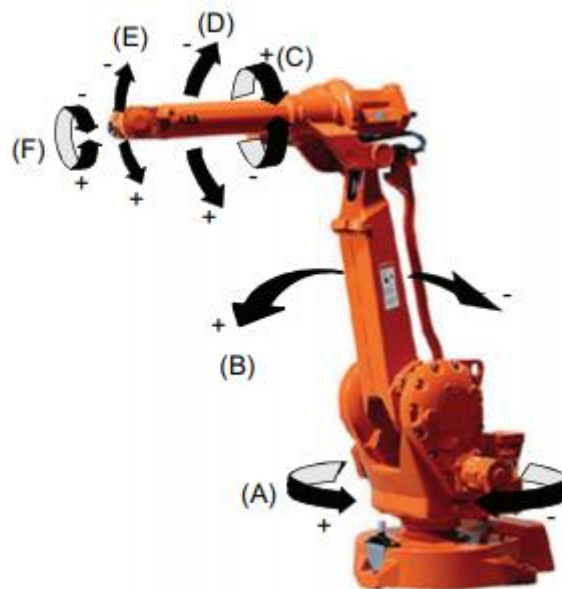


Figure 2 shows the manipulator axes of the IRB 2400 (Product specification, 3HAC042195-001, p. 10).

Position	Description	Position	Description
A	axis 1	B	axis 2
C	axis 3	D	axis 4
E	axis 5	F	axis 6

Table 1: Manipulator axes of the IRB 2400 (Product specification, 3HAC042195-001, p. 10).

- **Global service and support:** ABB also offers Remote-Service, which provides remote access to equipment for monitoring and assistance, allowing for worry-free activity. Customers can also take advantage of ABB's service organization, which has over 35 years of experience in the arc welding sector and offers assistance in over 100 locations across 53 countries.
- **Main applications:**
 - Arc Welding
 - Deburring/Cutting
 - Polishing/Grinding

2.2.2 Various robot models and specifications

The IRB 2400 comes in two versions, both of which may be placed inverted (no titling allowed on the X- and Y-axes). IRB 2400/10 and IRB 2400/16 are the two models. The Table 2 below indicates some specifications of the IRB 2400:

- **Specification:**

Version for Robots	Reach (m)	Payload (kg)
IRB 2400/10	1.55	10
IRB 2400/16	1.55	16
Number of axes	6	
Protection	Foundry Plus IP67, IP54	
Mounting	Both models can be floor or inverted assembled. IRB 2400/10 is wall-mounted.	
Controller	IRC5 Single Cabinet/IRC5 Panel Mounted	

Table 2: the IRB 2400's specification (IRB 2400 – PR10034EN, p. 2).

- **Performance (according to ISO 9283):** are illustrated in the Table 3 below.

	Position repeatability	Path repeatability
IRB 2400/10	0.03 mm	0.11 mm
IRB 2400/16	0.03 mm	0.15 mm

Table 3: the performance of IRB 2400 (IRB 2400 - PR10034EN, p. 2).

- **Technical information:** are depicted in the Figure 3 below.

Electrical Connections	
Supply voltage	200-600 V, 50/60 Hz
Power consumption	ISO-Cube at max speed 0.67 kW
Physical	
Robot base	723 x 600 mm
Robot height	1564 mm
Robot weight	380 kg
Environment	
Ambient temperature for mechanical unit	
During operation	+5°C (41°F) to +45°C (113°F)
Relative humidity	Max. 95 %
Noise level	Max. 70 dB (A)
Emission	EMC/EMI-shielded

Figure 3 indicates the technical information of the IRB 2400 (IRB2400 - PR10034EN, p. 2).

- **Movement:** some axis movement are shown in the Table 4.

Axis movement	Working range	Axis max. speed
Axis 1 rotation	+ 180° to - 180°	150°/s
Axis 2 arm	+ 110° to - 100°	150°/s
Axis 3 arm	+ 65° to - 60°	150°/s
Axis 4 rotation	+ 200° to - 200° Optional: Unlimited	360°/s
Axis 5 bend	+120° to - 120°	360°/s
Axis 6 turn	+ 400° to - 400° Max. rev: + 250 to - 250	450°/s

Table 4: the movement of IRB 2400 (IRB 2400 - PR10034EN, p. 2).

Both the floor and inverted mounting use the same working space. Axis 1 is limited to 30 kg for the wall-mounted 10 kg version.

- **Working range:** are presented in the Figure 4.

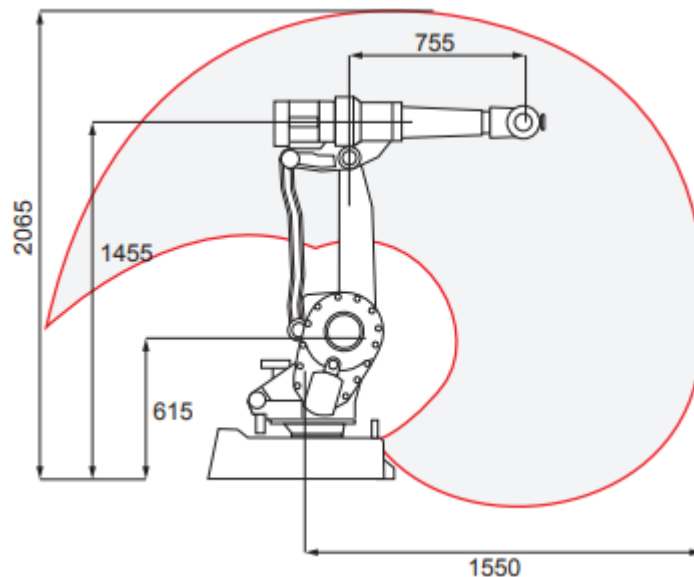


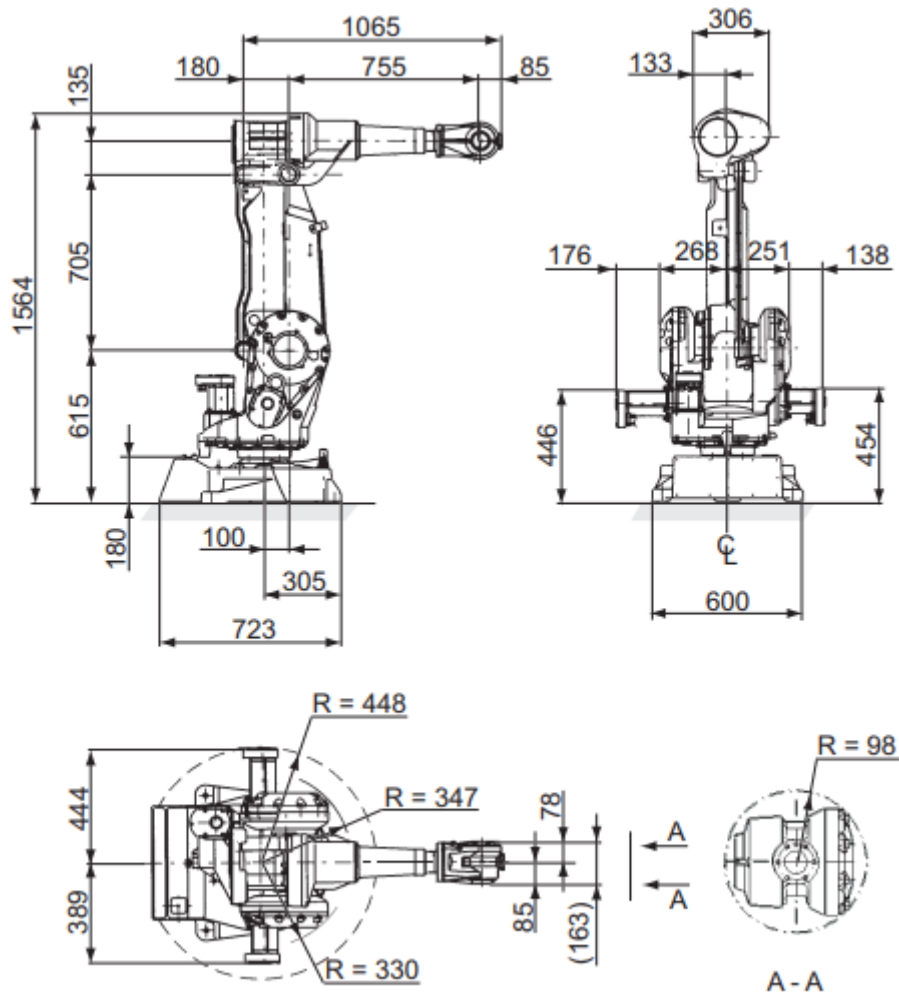
Figure 4 shows the working range of the IRB 2400 (IRB 2400 - PR10034EN, p. 2).

- **Other technical data:** are given in the table 5.

Data	Description	Note
Air born noise level	The sound pressure level outside the working space	< 70 dB (A) Lep (acc. ro Machinery directive 2006/42/EG)

Table 5: Other technical data of the IRB 2400 (Product specification, 3HAC042195-001, p. 11).

- **The dimensions of IRB 2400/10 and IRB 2400/16:** the dimensions of two versions are indicated in Figure 5 below.



xx1100000546

Figure 5 shows the dimensions of two versions of the IRB 2400 (Product specification, 3HAC042195-001, p. 12).

3 Operating principles

3.1 Overview of operating principles

As a 6 degree of freedom robot, IRB 2400 has extremely effective and precise operation, greatly increasing the productivity of the production system export. The system of IRB 2400 consists 2 main parts: Controller and Manipulator, which are presented in Figure 6.

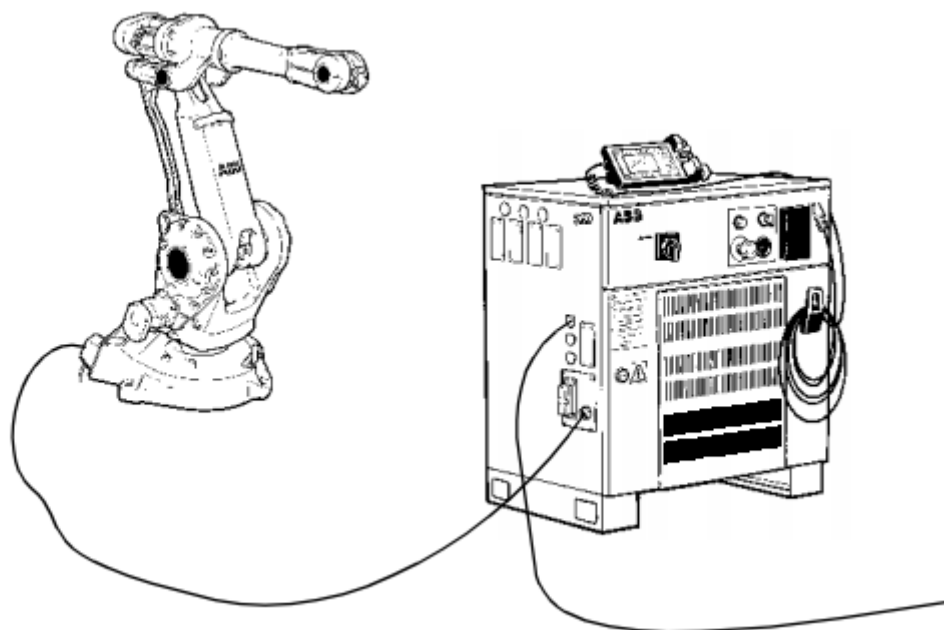


Figure 6 shows the system of IRB 2400 (Product specification, IRB 2400, 3HAC 5672-1, p. 46).

3.2 The controller

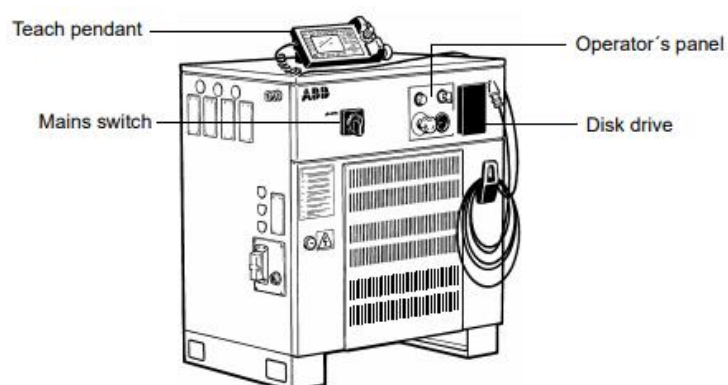


Figure 7 indicates the Controller, which is specifically built to control the robots and ensures maximum efficiency and functionality (Product specification, IRB 2400, 3HAC 5672-1, p. 5).

The Figure 7 above is S4CPLUS-M2000 control cabinet which is the main control for the robot (Manipulator). S4CPLUS contains transformers, power converters, pulse generators, processing computer systems and other function blocks that are responsible for generating control pulses, calculating ... input the voltage power supply for the cabinet 200VAC: 63A or 600VAC: 25A.

The main computer system in the cabinet is divided into 3 parts:

- **Main computer:** This section is responsible for receiving and executing the RAPID program and the controller robot path, responsible for human-robot communication, the controller connecting cooling fans, the generator controller, pulse, and so on.
- **Axis computer:** This part is responsible for adjusting the speed and positions of the shafts (including six main shaft and auxiliary shaft). It directly takes the position value from the main calculator and the actual positions value from the measuring system, using these data for control modify algorithm, calculate the values of torque and position value sent to the drive system to generate control pulses for robot motors.
- **I/O computer:** This is the part that connects the main computer to the system of technology chains and external machines. The cabinet includes 16 inputs and 16 outputs, using a 24VDC-5.5mA source provided by the power supply in the cabinet.

The cabinet offers 2 control modes: manual control with Teach pendant and programmable automatic control. The teach pendant is displayed in the Figure 8.

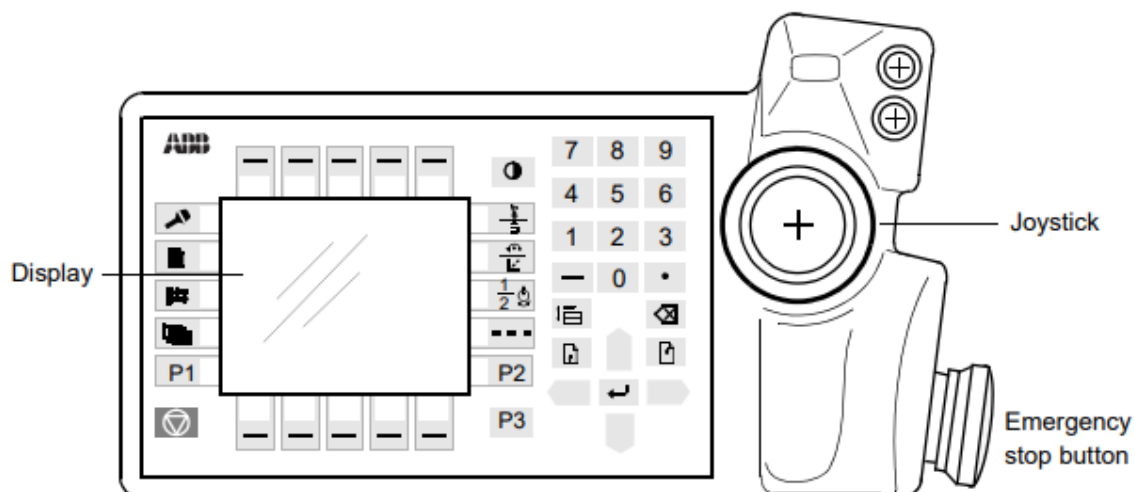


Figure 8 shows a teach pendant with a big monitor that shows prompts, error messages, and other information in plain English (Product specification, IRB 2400, 3HAC 5672-1, p. 7).

3.3 The manipulator

The manipulator, which is shown in Figure 9 below, is the actuator of the system (robot arm). It receives control signals from the control cabinet and moves in the correct manner. It includes 6 main axes and auxiliary axes.

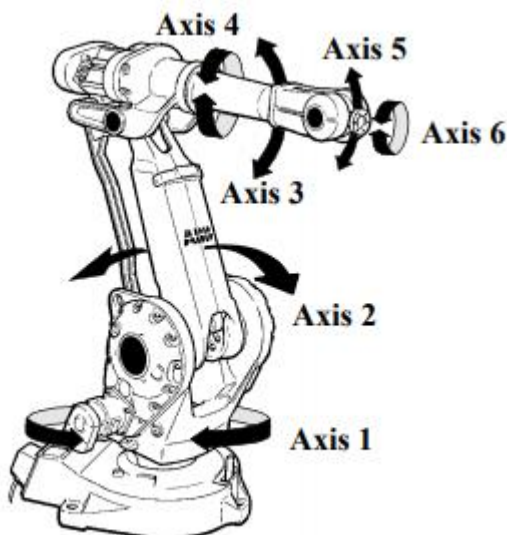


Figure 9 shows the IRB 2400 manipulator with 6 main axes (Product specification, IRB 2400, 3HAC 5672-1, p. 5).

In addition, the manufacturer built the robot to work with auxiliary shafts, the maximum number of auxiliary axes that the IRB 2400 can add is six, and they will be regulated by an S4CPLUS cabinet (the Controller). To simplify the addition and control of auxiliary shafts, ABB has also provided ready-made kits. Figure 10 depicts some popular types of ready-made kits.



Figure 10 indicates the auxiliary shafts and ready-made kits for ABB robot (ROB028EN, p. 1).

Moreover, another important thing to mention is coordinate systems. Numeric values or jogging the robot around a variety of positions may be used to program coordinate systems (the tool does not to be moved). With respect to the tool's location and orientation, each position is defined in object coordinates. This means that if a tool is modified due to injury, the old software can still be used, unchanged, by creating a new tool definition. Only the user or object coordinate scheme must be redefined if a fixture or workpiece is moved. Coordinate systems are shown in Figure 11:

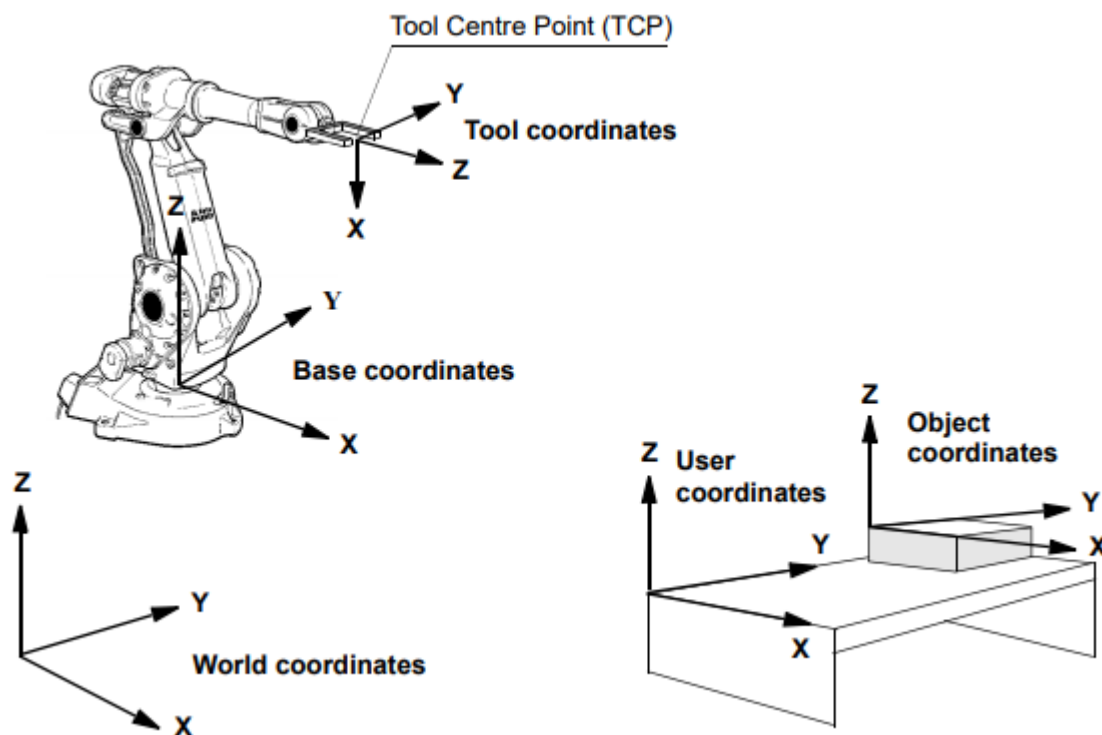


Figure 11 demonstrates the coordinates systems that are used to facilitate jogging and off-line programming (Product specification, IRB 2400, 3HAC 5672-1, p. 14).

- **The world coordinate system:** establishes a reference to the ground, which serves as the starting point for all other coordinate systems. It is possible to link the robot location to a fixed point in the workshop using this coordinate system. When two robots collaborate or when using a robot carrier, the world coordinate system comes in handy.
- **The base coordinate system:** is inserted into the robot's base mounting surface.
- **The tool coordinate system:** indicates the axis of rotation and the central point of the tool.
- **The user coordinate system:** indicates a fixture or a workpiece manipulator's position.
- **The object coordinate system:** describes how a workpiece should be placed in a manipulator or fixture (ABB Group, n.d.).

3.4 Communication

There are two serial channels for IRB 2400, one RS232 and the other RS422 Full duplex, for communicating with printers, terminals, computers, and other devices. The serial point to point is indicated in Figure 12 below.

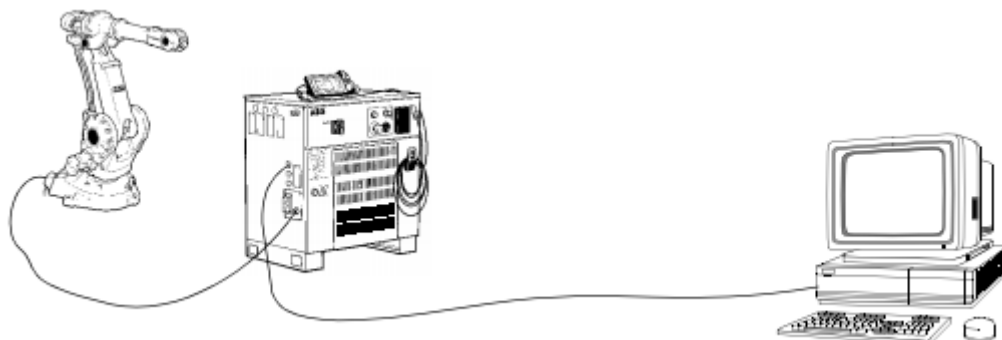


Figure 12 shows the serial point to point communication (Product specification, IRB 2400, 3HAC 5672-1, p. 46).

The serial channels can operate at speeds ranging from 300 to 19200 bit/s (1 channel, 19200 bit/s maximum). High-speed and/or network communication Ethernet interfaces are available. The data transmission rate is 10Mbit/s. To learn how to contact a network in serial mode, look at Figure 13 below.

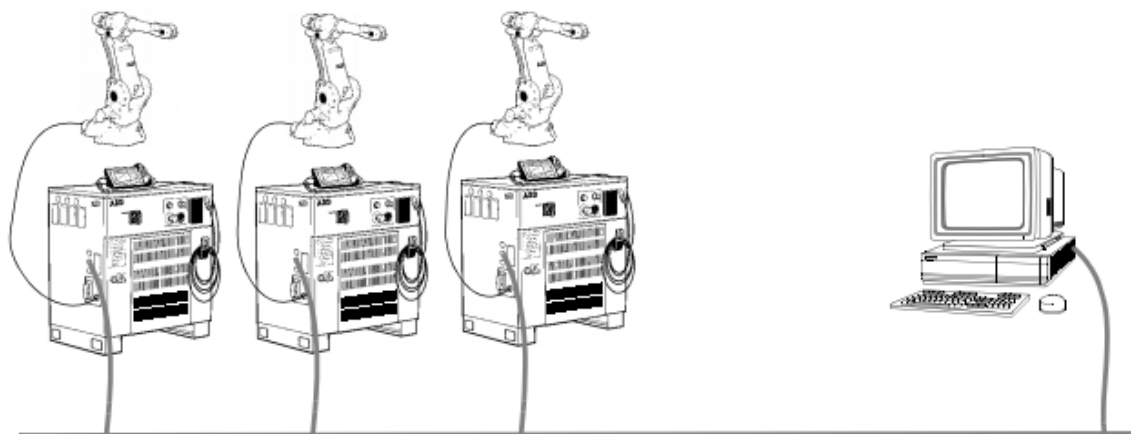


Figure 13 illustrates that network contact in serial mode (Product specification, IRB 2400, 3HAC 5672-1, p. 46).

RAPID instructions may be used to transfer character-based or binary data. This necessitates the Advanced functions alternative (see Product Specification RobotWare). A Robot Application Protocol (RAP) can be used in addition to the Physical channel. This involves FactoryWare Interface or RAP Communication, as described in the Product Specification RobotWare (ABB Group, n.d.).

4 Programming language – RAPID

4.1 About RAPID

RAPID is an application-oriented programming language with a high degree of abstraction. RAPID comes with the following features:

- The structure is hierarchical and modular
- Processes and functions
- Data routines on a global or local scale
- Data types, such as structured and array data types
- Variables, routines, and other items have user-defined names
- Extensive program flow management
- Expressions involving arithmetic and logic
- Handling of interruptions
- Error correction
- Instructions that are specified by the user
- Handler for backward execution (ABB Group, n.d.).

4.2 The data of RAPID

4.2.1 The Variables

RAPID has many different data types, but let's reflect on 3 general data types, which are shown in Table 6 below.

Types of data	Description
Num	Integer and decimal numbers can be used to represent numerical data. E.g. 23 or 23.45.
String	A string of text. There is a limit of 80 characters.
Bool	A logical (Boolean) variable. As a value, only True or False can be used.

Table 6: Three general data types in RAPID (Introduction to RAPID, 3HAC029364-001, p. 12).

These three are the foundation for all other data forms. If you know how to operate on the three basic data types, you can quickly comprehend how they can be joined to build more complicated data kinds.

A data value is contained in a variable. The variable retains its value when the program is stopped and restarted, but when the pointer to the program is transferred to the primary, the data value of the variable has vanished.

Making a variable declaration is the process of naming a variable and specifying its data type. According to the syntax, the keyword "VAR" is used to declare a variable (Introduction to RAPID, 3HAC029364-001, p. 12):

VAR datatype identifier;

4.2.2 RAPID's Persistent variables

A persistent variable is similar to a regular variable, but with one significant difference. Even if the program has been terminated and reinstalled from scratch, the last value assigned to a persistent variable is remembered.

The keyword "PERS" is used to declare a persistent variable, and an initial value must be allocated at the time of declaration (3HAC029364-001, p. 13):

PERS num nbr := 1;

PERS string string1 := Hello;

4.2.3 RAPID Constants

A constant, like a variable, has a value, however the value is always assigned to the item at the time of declaration and cannot be modified after that. The constant functions similarly to a variable in the program, with the exception that it cannot be given a new value.

The keyword "CONST" is used to announce the constant, which is followed by the data type, identifier, and meaning assignment (Introduction to RAPID, 3HAC029364-001, p. 14):

CONST num gravity := 9.81;

CONST string greating := Hello;

4.2.4 Types of operators in RAPID

Numerical operators, relational operators, and string operators are the three types of operators in RAPID. The numerical operators (Figure 14) work with the data type "num" and return the data type "num".

Operator	Description
+	Addition
-	Subtraction Unary minus
*	Multiplication
/	Division

Figure 14 shows the numerical operators (Introduction to RAPID, 3HAC029364-001, p. 15).

The relational operations (Figure 15) return the data type “*bool*”.

Operator	Description
=	equal to
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to
<>	not equal to

Figure 15 shows the relational operators (Introduction to RAPID, 3HAC029364-001, p. 15).

4.3 Move instructions of RAPID robot functionality

4.3.1 The MoveL instruction

RAPID has the advantage of being specifically designed to manage robots, in addition to providing much of the features found in other high-level programming languages. Above everything, there are directions for moving the robot.

This is an example of a basic step instruction (Introduction to RAPID, 3HAC029364-001, p. 23):

MoveL p10, v1000, fine, tool 0;

Where:

- *MoveL* is a command that moves the robot linearly to the destination position from its current position.
- *P10* defines the direction in which the robot should move.
- *V1000* states that the robot's speed must be 1000mm/s.
- *Fine* states that the robot shall follow the path indicated and not take any shortcuts on its journey to the next spot.
- *Tool10* indicates that the mounting flange at the robot's tip should be moved to the specified position (ABB Group, n.d.).

4.3.2 Coordinator systems

A move instruction's target position is defined as coordinates in a co-ordinate system. If no coordinate system is defined, the position is reported according to the robot's base coordinate system. The origin of the base coordinate system is the robot base, which is indicated in Figure 16.

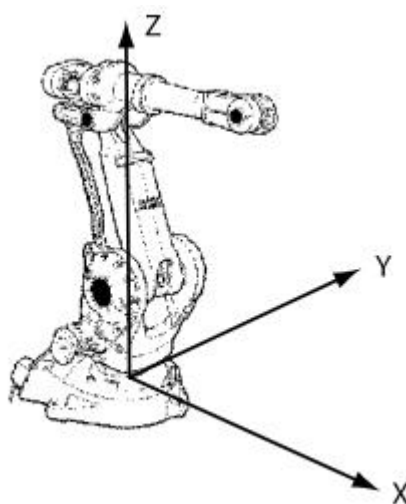


Figure 16 illustrates the base coordinate system (Introduction to RAPID, 3HAC029364-001, p. 25).

Move instructions can use a different coordinate system that has been established. The optional argument “\WObj” specifies the coordinate system that will be used by move instructions (Introduction to RAPID, 3HAC029364-001, p. 25).

```
MoveL p10, v1000, z50, tool0 \WObj := wobj1;
```

4.3.3 Other move instructions

RAPID has multiple move instructions, the most popular of which are *MoveL*, *MoveJ*, and *MoveC*.

- *MoveJ* is used to quickly move the robot from one spot to another when the movement does not need to be in a straight line.
- *MoveC* is used to make the robot move in a circular arc.

5 ABB Robotstudio

ABB's RobotStudio simulation and offline programming software allows robot programming to be done on a PC in the office without interrupting production. It also includes the tools you need to do things like training, programming, and improving without interrupting production.

The ABB Virtual Controller tool is an identical replica of the actual software that runs your robots in production. This enables for very realistic simulations utilizing real robot programming and configuration files that are identical to those used on the shop floor.

RobotStudio comes with a comprehensive set of features and add-ons that enable flawless offline programming, accelerate start-up times, and lower risks, and so on (ABB Group, n.d.). One step in the RobotStudio installation process is depicted in Figure 17 below.



Figure 17 shows the RobotStudio installer issue

6 Visual Components Premium

The Visual Components platform was designed as a developer 3D software for manufacturing simulation applications. Physical behaviour is included in the Visual Components, which is powered by the NVIDIA PhysX engine and allows you to simulate and visualize functional effects such as collisions, gravity, and material properties.

Visual Components were created by the manufacturers to take full advantage of the 64-bit Windows environment, which includes improved graphics, faster loading times, and a much smoother experience.

Layout design, off-line programming, production simulation, and PLC verification are just a few of the applications that Visual Components software is used for. We have Figure 18 below when we first start the software.

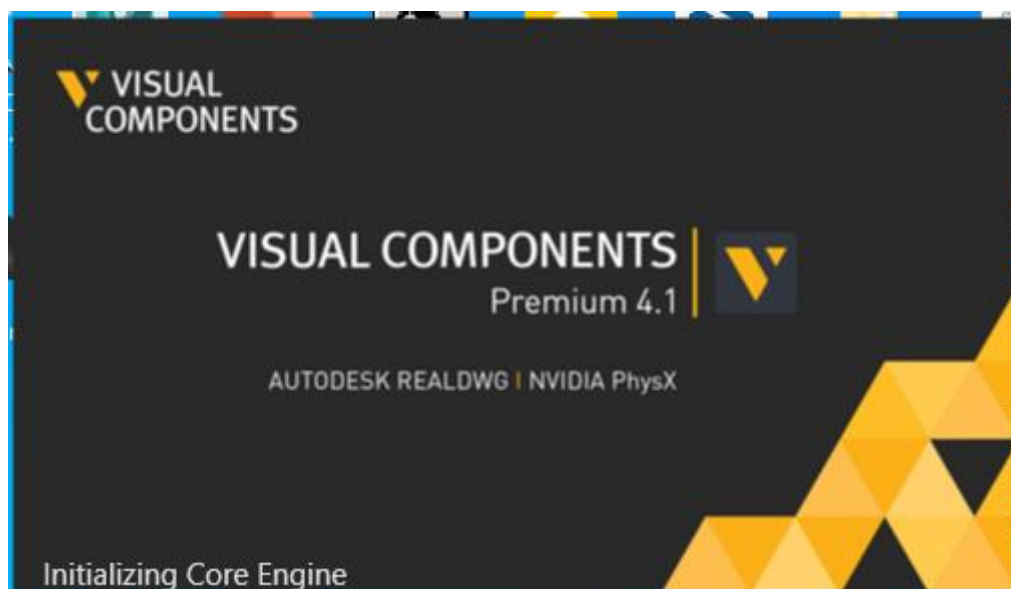


Figure 18 shows the Visual Components display

7 Exercises

7.1 Pick and Place

Based on the advantages of Visual Components Premium: better graphics, faster loading time, and a much smoother user experience, the exercise was recommended to program, run, and test on Visual Components Premium 4.1.

The IRB 2400 was taken in the Visual Components Premium e-Catalog/Collections (Figure 19).

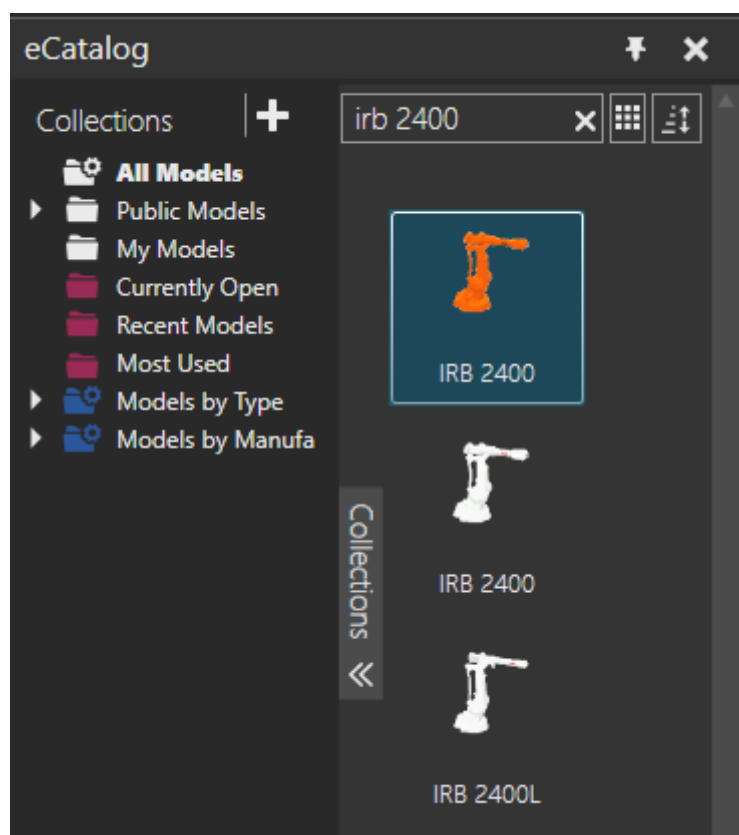


Figure 19 illustrates the Visual Components e-Catalog

4 Conveyors were chosen from the Visual Components e-Catalog/Collections/All Models (figure 20).

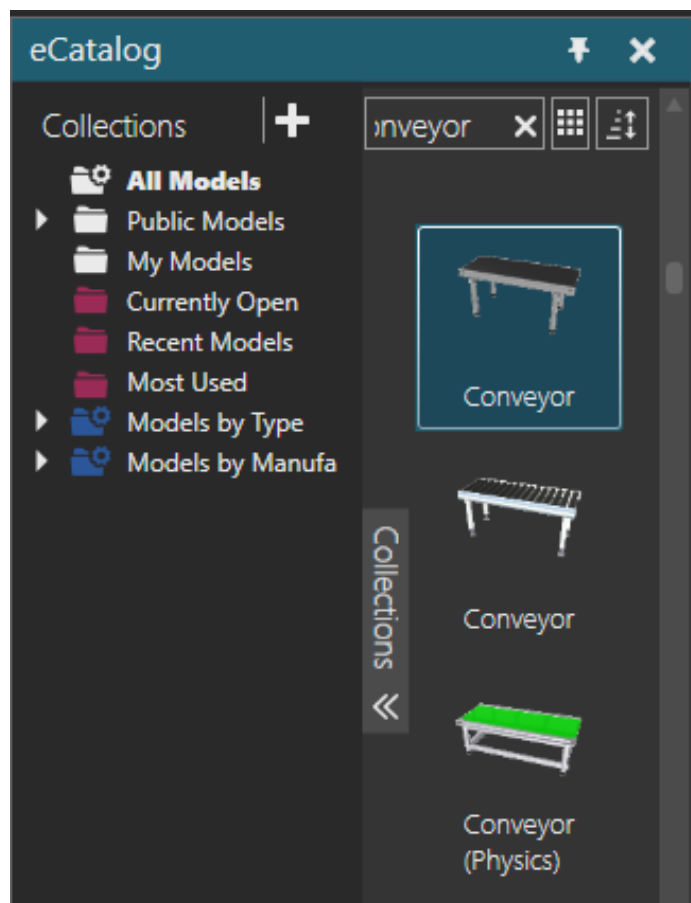


Figure 20 indicates the Conveyor in this exercise.

The Feeder, which creates generic products from the e-Catalog or what is chosen in Component-Creator tab and releases them in the line, was also picked up from the e-Catalog/Collections (figure 21). Connect the Feeder using PnP at the begin of a line to produce products as needed. Shape Feeder can create any existing product in the layout when the product component is attached to the blue node of the feeder.

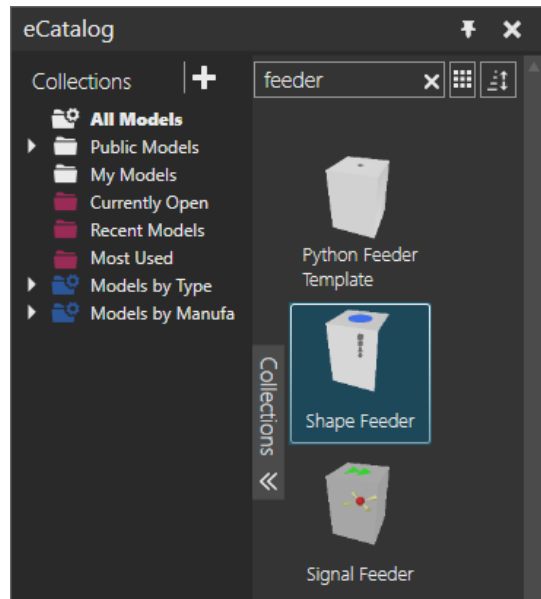


Figure 21 shows the Shape Feeder

Therefore, the simulation includes the IRB 2400, 4 conveyors, 1 shape feeder, 1 end block and 1 pedestal, which are exposed in Figure 22 and Figure 23. The end block, which is used to stop the parts on a line component, should be connected to conveyor using PnP. The parts following in the conveyors stop after reaching the end block.

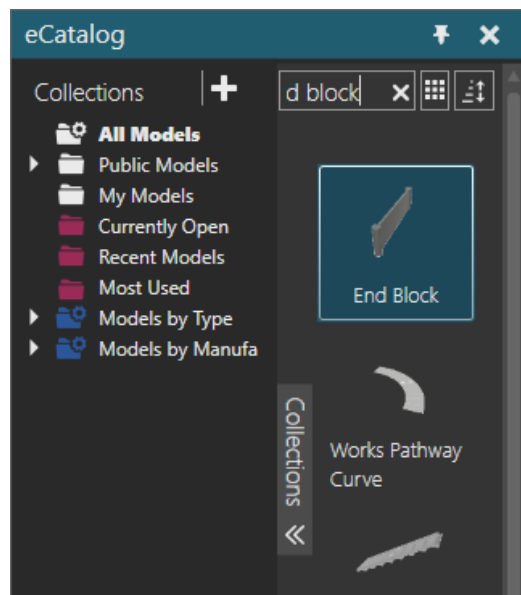


Figure 22 illustrates the End Block.

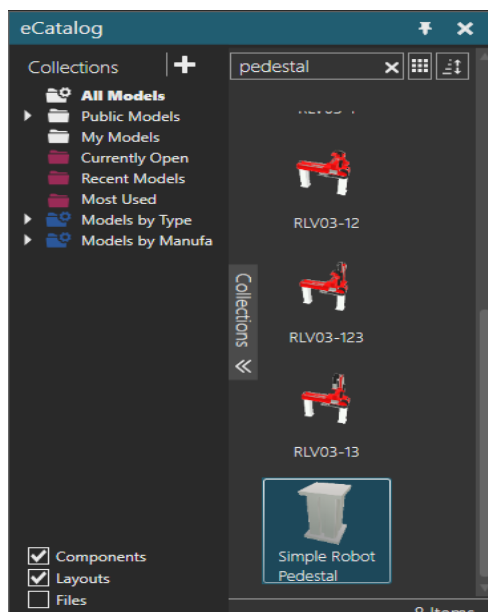


Figure 23 shows the Pedestal, which is a simple robot riser.

To remotely connect the components with wireless interface/ports in the 3D world, from the Home tab click on Signals (Figure 24).

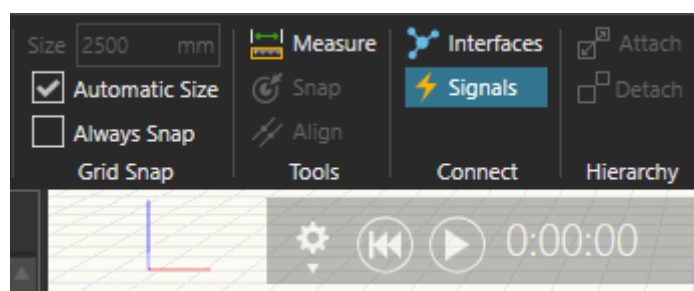


Figure 24 illustrates the Signal on the Home tab.

From the Home tab/Signals, the components with wireless interface/ports should appear in the 3D world. In order to ensure that the program can function properly, the signals should be connected. The input of IRB 2400 should be connected with the Sensor Boolean Signal of the End block. Besides, the output should be connected with the Power on Signal of Conveyors, and Create Signal of Shape Feeder (Figure 25).

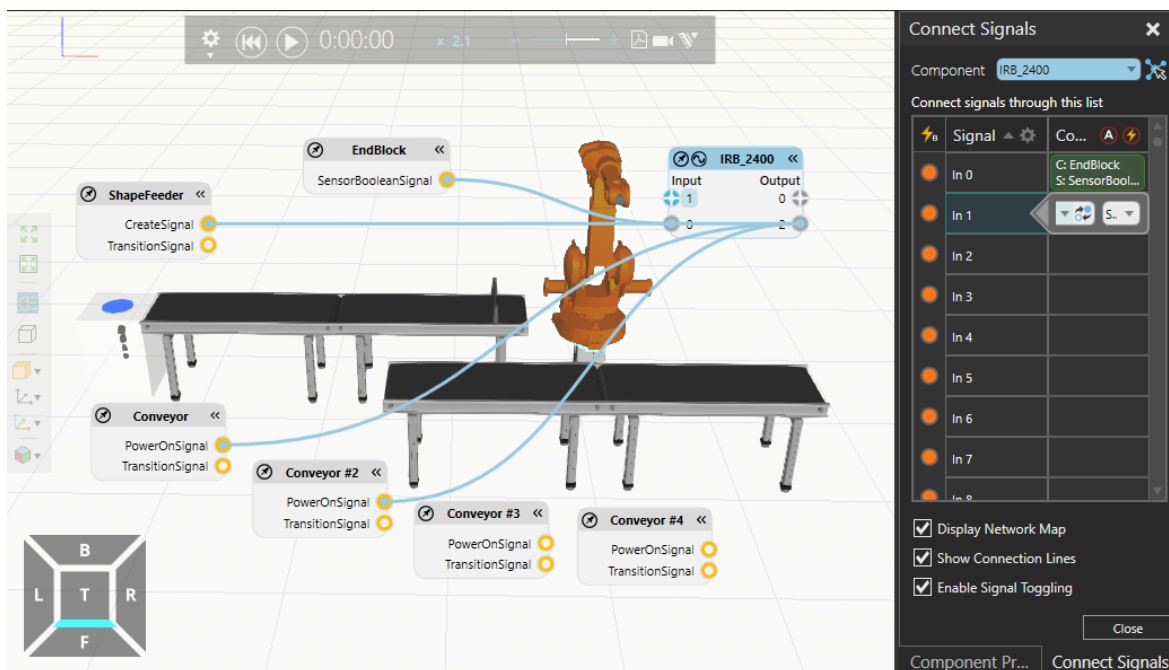


Figure 25 shows the components with the wireless interface/ports.

Once the components are set up, a program should be created to make sure that the robot works as intended. From the Program tab, select the IRB 2400 to start editing the robot program. There were 1 main program and 2 subroutines in this exercise. Pick up and Place are two subroutines that were added by clicking the add sequence button in the Subprograms tab (Figure 26).

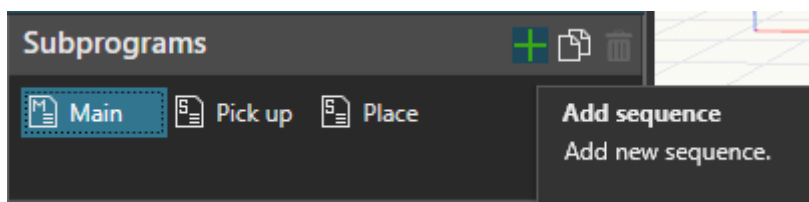


Figure 26 shows the Subprograms tab.

In this exercise, while the statement was used in order to the “Pick and Place” can be repeated until the end. At the main program, create the transshipment point P1, which is not so important but for the robot to function properly is essential. To create a position, move the robot arm to position by using Jog on the taskbar (Figure 27), then click on the Point-to-Point Motion Statement button on the Subprograms tab (Figure 28).

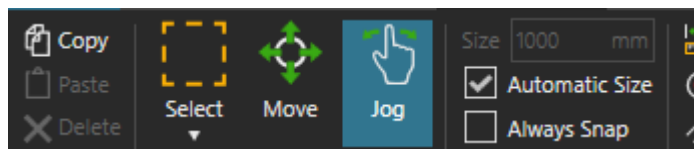


Figure 27 shows the command Jog on the taskbar.

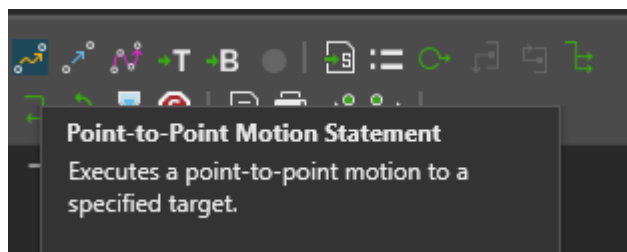


Figure 28 shows the PTP motion statement.

After moving to point P1, using command Wait for Binary Input Statement (Figure 29) to pause the robot's program and listen for a connected signal 's value to match and inputted value. Besides, using command Set Binary Output Statement (Figure 30) to send out a signal and its value to connected behaviours. In simple terms, the robot arm would move to point P1 then waiting for boxes releasing on conveyors until reaching the end block.

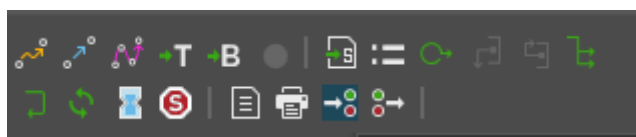


Figure 29 shows command Wait for Binary Input Statement.

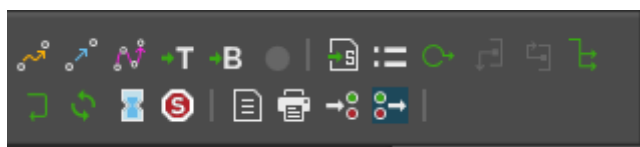


Figure 30 indicates the command Set Binary Output Statement.

When the boxes reaching to the end block, move to 2 subroutines by using command Call Sequence Statement (Figure 31). To finish the main program, using the command Wait for Binary Input Statement again. Figure 32 below illustrated the main program.

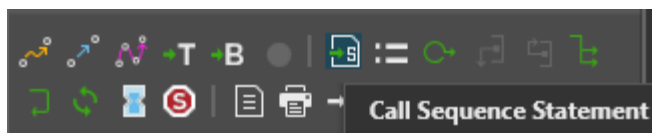


Figure 31 illustrate the command Call Sequence Statement.

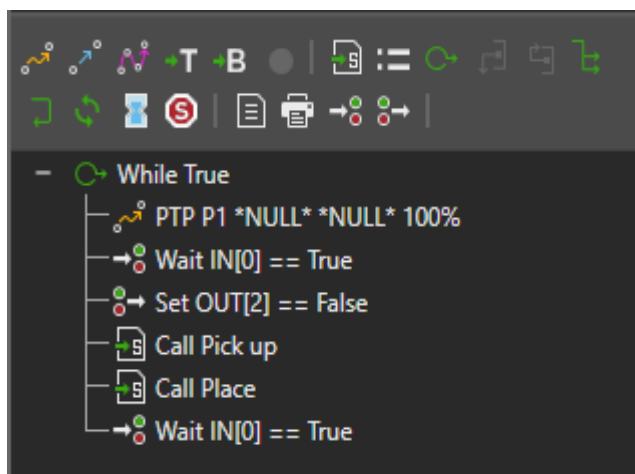


Figure 32 shows the Main program.

In the “Pick up”, create 2 new positions P2 and P3, in while P2 is the point of contact with the box (Figure 33). P3 was created by using command PTP Motion Statement from P1, while P2 was caused by command Linear Motion Statement, that executes a linear motion from P3 to the box. Following step, sending a signal to the robot arm to grasp the box through command Set Binary Output Statement (Output value is True). To grasp the box, the Signal actions, that is in Home tab/Show/Component Properties/Actions Configuration, had to changed according to Figure 34. The robot arm would grasp the box and moved to position P3 (P4) before moving to “Place”. The whole “Pick up” subroutine is shown in Figure 35.

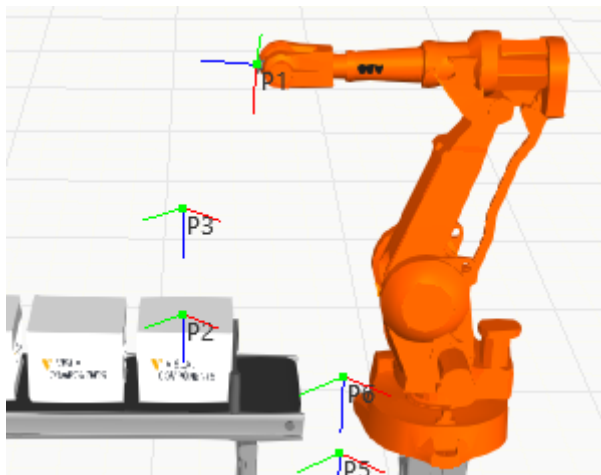


Figure 33 indicates the Positions in "Pick up".

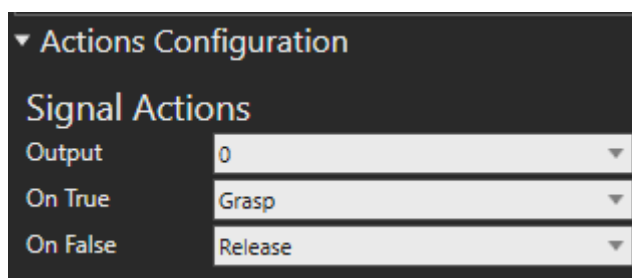


Figure 34 shows the Signal Actions changing.

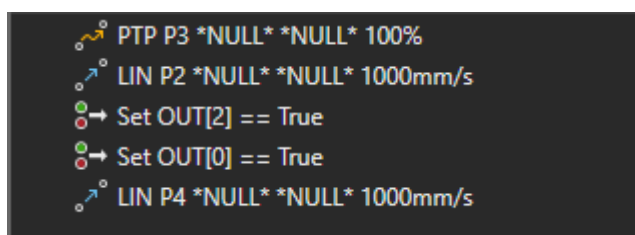


Figure 35 indicates the "Pick up" subroutine.

In the "Place", set up 2 new positions P5 and P6 (Figure 36) in the same way with "Pick up". When the robot arm moved to P5, used command Set Binary Output Statement (Output value is False) to release the box on the conveyor. The robot arm would return to position P6 (P7) before resuming the work. Figure 37 indicated the whole program of "Place" subroutine.

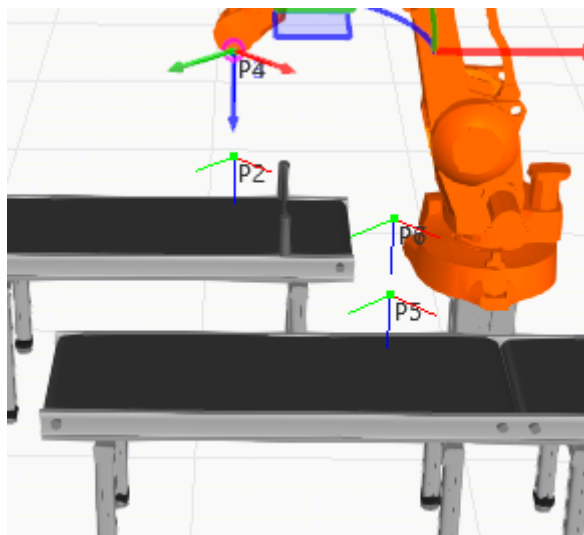


Figure 36 shows the Positions in "Place" subroutine

```

PTP P6 *NULL* *NULL* 100%
LIN P5 *NULL* *NULL* 1000mm/s
Set OUT[2] == False
LIN P7 *NULL* *NULL* 1000mm/s

```

Figure 37 shows the "Place" subroutine.

The last step was to record the simulation of "Pick and Place".

7.2 Palletizing

The purpose of this exercise was to pick up the boxes from the conveyor and arrange them in rows and columns on the pallet. Not much different from the previous exercise, this simulation includes IRB 2400, 2 conveyors, 1 end block, 1 shape feeder and 1 Euro pallet. The basic settings, which are signals connecting, the robot arm's output setting (grasping and releasing), have not changed much either. The difference in this exercise was to define a base statement for the Euro pallet, used in robot programs involving the stacking and unstacking of parts, that changed the properties of a base frame in the pallet during the simulation. This base frame could be used as a base coordinate system for motion statements. Moving this base frame would also move any motion statements referencing this base frame.

There were also 1 main program and 2 subroutines (Pick and Place) in this simulation, in which the "Pick" part (Figure 38) was similar to the previous exercise. The Positions are shown in Figure 39.

```

PTP P1 tool1 *NULL* 100%
LIN P2 tool1 *NULL* 1000mm/s
Set OUT[2] == True
Set OUT[0] == False
LIN P3 tool1 *NULL* 1000mm/s

```

Figure 38 indicates the "Pick" subroutine.

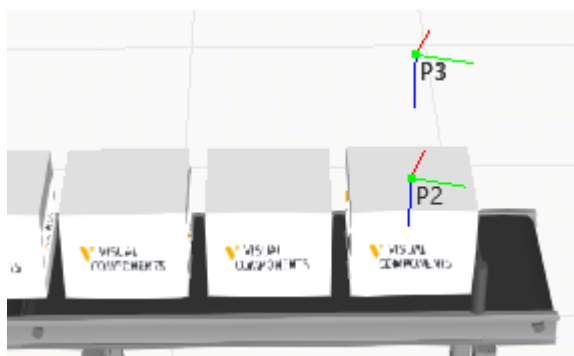


Figure 39 illustrates the positions in "Pick" subroutine.

Return to the main program and define a base frame for the pallet, this step plays an important role in this simulation. First of all, actually attached a base frame in the robot to the node of the pallet. From the program editor tab, clicked on Define Base Statement button to add it to the main routine. In the statement properties panel, chose the base 1, which base frame was needed to edit in the robot. Most robots have their base frames attached to their world frame by default. To see where its role the frame is, clicked on the Move Robot World Frame command. Also went to the three-wheel toolbar and selected the checkbox called robot bases in the Frame types to see the base trains in the 3D world. To attached the base frame 1 in the robot to the node in the pallet, back to the statement properties panel, clicked on the Pick property button and chose the pallet. After the base frame 1 was the origin of the pallet node, changed the coordinates like Figure 40 to move the base frame to the corner of the pallet (Figure 41).

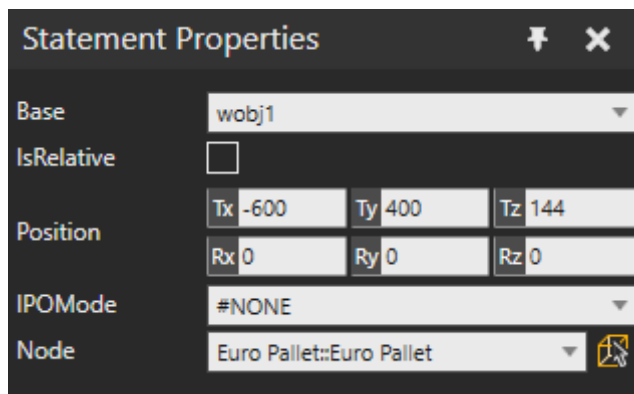


Figure 40 shows the coordinates of the base frame 1.

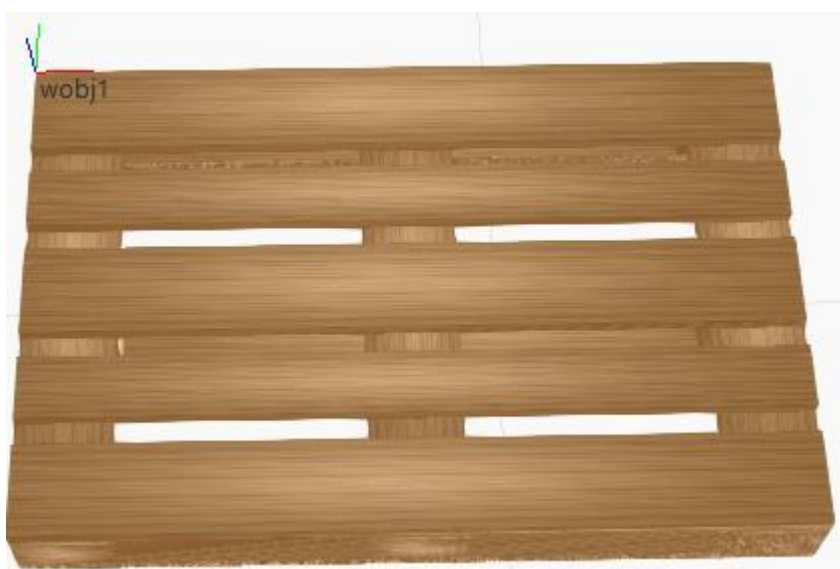


Figure 41 demonstrates that the base frame 1 was in the corner of the pallet.

The purpose of following step was to move the base frame 1 to corner of the pallet permanently. To handle it, just delete the defined based statement then use the Jog command to select the robot in 3D world. In the Jog panel, use the base property to select base frame 1 then click on the gear icon. For base properties, change the node to pallet by using the pick property button, also use the parent system for coordinates, and change the “coordinates” values following the previous step. Now, the base frame 1 moved to the corner of the pallet permanently.

After defining a base frame in the pallet, the main program should be set up. Select the main routine then create 2 variables count X and count Y (Integer variable) in the routine properties panel. These variables were being used to count the items in a row and the

number of rows. There were three while statements in the main routine, in which the first loop's condition was set to true. The second loop indicated that there were 2 rows of items, while the third loop showed there were 3 boxes in one row. The third while statement is where the picking and placing task were doing. After doing those tasks, a command to increment the value of count X variable need to be created using command Assign Variable Statement (Figure 42).

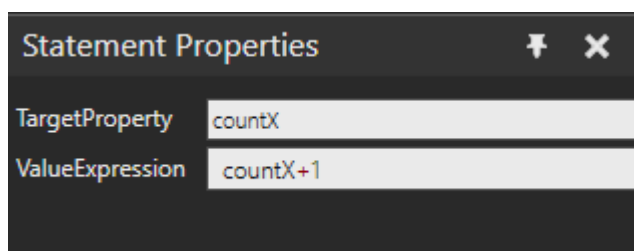


Figure 42 shows the Assign Variable Statement panel.

In the second loop, the value of variable, used to count the number of rows, was being incremented, in while the count X variable was set back to 0. To set the boxes in a row one-by-one, used command Define Base Statement to change the positions and orientation of base frame. Change the value of coordinates as figure 43 for setting boxes in the first row, and figure 44 to set the second row.

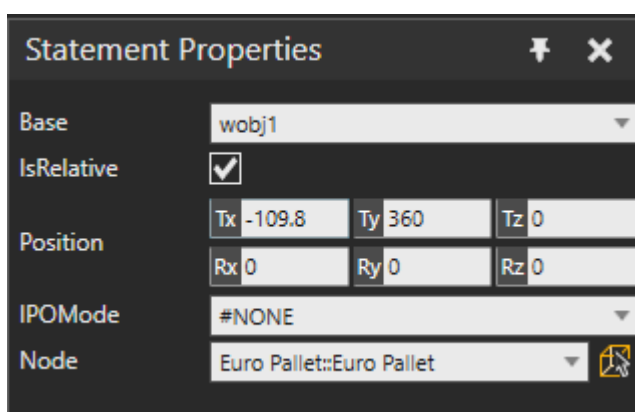


Figure 43 shows the value of coordinates to set the boxes in the first row.

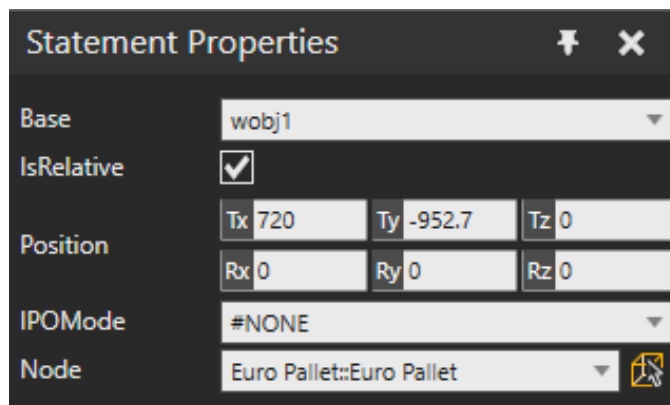


Figure 44 indicates the value of coordinates for setting boxes in second row.

Besides, in the Place routine, the positions, taught for approaching, placing and retracing were using base 1. So when the location of base 1 was changed during the simulation, those positions would automatically update with it. The whole program for “Place” subroutine is shown in Figure 45 below.

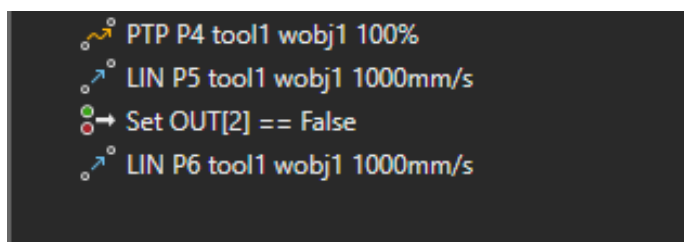


Figure 45 indicates the Place subroutine.

In the first loop, after finishing place 2 rows of boxes, added a Halt Statement to stop the simulation (it can also be used to reset the simulation). The whole main program is displayed in the Figure 46 below.

At the end of this exercise, recording the simulation.

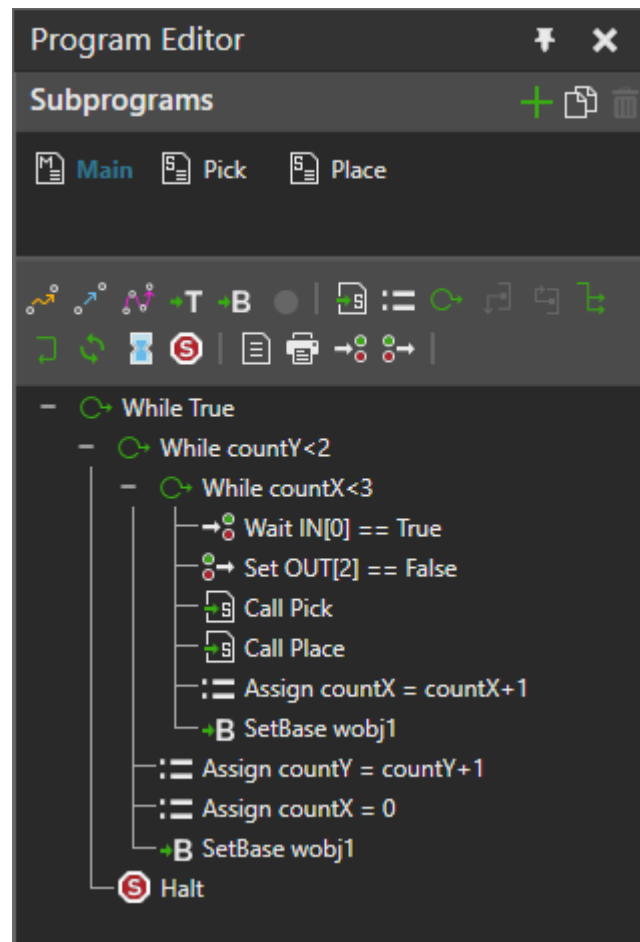


Figure 46 shows the Main routine of "Palletizing".

8 Conclusion

The project was completed by using Visual Components Premium software to create and record simulations. The thesis was divided into several parts to get ideas and knowledge easily. The project has been edited several times to gain a more pleasing result.

Link for the Pick and Place: <https://www.youtube.com/watch?v=P56ZNTK9vF4>

Link for the Palletizing: <https://www.youtube.com/watch?v=V6zL5dORknA>

During the process of the thesis, several difficulties had to be faced and solved. Due to the limited knowledge, information sources on the internet are disturbed, the thesis was completed later than expected. Besides, there is not any ABB robot in our campus, so this study just could be performed on Visual Components Premium. Several mistakes occurred during the simulation, including an incorrect information source, a difficulty with the Visual Components software, and so on.

Through this project, knowledge of industrial robots, their applications and how to simulate a program are expanded even at a basic level. Due to the limitation of duration, the project could not be expanded with complicated programs.

The project has lots of possibilities for improvement and future development such as performing programs that respond to actual needs, etc.

References

1. ABB Group. (n.d.). Technical data for the IRB 2400 industrial robot.
<https://new.abb.com/products/robotics/industrial-robots/irb-2400/irb-2400-data>
Accessed in February 2021.
2. WELDCOM, 2015. ABB robot realize the dream of industrial automation.
<https://m.weldcom.vn/nv374/robot-abb-hien-thuc-hoa-giac-mo-cong-nghiep-tu-dong-hoa.html>
Accessed in February.
3. ABB, Article Huizhong, 2008. Smart robots help propel growth.
<https://search.abb.com/library/Download.aspx?DocumentID=9AKK104295D4235&LanguageCode=en&DocumentPartId=&Action=Launch>
Accessed in February 2021.
4. ABB, Article SEASL, 2008. A better made blade.
<https://search.abb.com/library/Download.aspx?DocumentID=9AKK104295D4213&LanguageCode=en&DocumentPartId=&Action=Launch>
Accessed in February 2021.
5. ABB Group. (n.d.). IRB 2400, data sheet.
https://search.abb.com/library/Download.aspx?DocumentID=PR10034EN_R7&LanguageCode=en&DocumentPartId=&Action=Launch
Accessed in February 2021.
6. ABB Group. (n.d.). Product manual – IRB 2400.
<https://abb.sluzba.cz/Pages/Public/IRC5UserDocumentationRW6/en/3HAC022031%20PM%20IRB%202400-en.pdf>
Accessed in February 2021.
7. ABB Group. (n.d.). Product specification – IRB 2400.
<https://library.e.abb.com/public/fa8324850004a97fc125766d003e863c/Product%20specification%202400%20M98A%20BWOS3.2.pdf>
Accessed in February 2021.
8. ABB Group. (n.d.). Operating manual – Introduction to RAPID.
<http://rovarc.cimr.pub.ro/docs/OpIntroRAPID.pdf>
Accessed in March 2021.

9. ABB Group. (n.d.). Technical reference manual – RAPID instructions, functions and data types.

https://library.e.abb.com/public/b227fcd260204c4dbeb8a58f8002fe64/Rapid_instructions.pdf?x-sign=f79v/883X1nHGc8fqH+WAJ2F30y/M6TZfYUuPuQpP+jeMBy-gouyGg+WSj8A9Otry

Accessed in March 2021.

10. Canonical robots, 2021. Visual Components 4.1 software.

<https://www.canonicalrobots.com/en/software/2031/visual-components-4-1-detail>

Accessed in March 2021.

11. ABB Group. (n.d.). The world's most used offline programming tool for robotics.

<https://new.abb.com/products/robotics/robotstudio>

Accessed in March 2021.

12. ABB Group. (n.d.). External Axis upgrade kits.

https://library.e.abb.com/public/fe6189e54c36ac6bc1257dfe0039a5db/ROB028EN_LR.pdf

Accessed in March 2021.