



VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Frederick Adotey Ofei

Issah Musah

DEVELOPMENT OF ROBOT CELL FOR INTERACTIVE CATAPULT

Technology and Communication

2012

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Mechanical and Production Engineering

ABSTRACT

Author	Frederick Adotey Ofei and Issah Musah
Title	Development of Robot Cell for Interactive Catapult
Year	2012
Language	English
Pages	30 + 16 Appendices
Name of Supervisor	Mika Billing

The purpose of the thesis was to create a physical version of the Angry Birds game using an articulated industrial robot. An articulated robot can perform all the functions needed to play it.

This was done by designing and producing a feed and catapulting system for the Angry Birds, fingers for the electric gripper and adapters for the attachment of the electric gripper to the end effector of the robot.

Based on the results from the testing phase and the aims of the thesis stated, it is enough to conclude that the thesis was a success.

Keywords Robotics, Robot cell, Programming, Angry Birds

CONTENTS

ABSTRACT

1	INTRODUCTION	8
1.1	The purpose of the thesis.....	8
1.2	Background to the study.....	9
2	INDUSTRIAL ROBOTS	11
2.1	Types of industrial robots.....	11
2.1.1	Scara robot.....	11
2.1.2	Cylindrical robot	12
2.1.3	Spherical robot	12
2.1.4	Articulated robot	13
2.1.5	Parallel robot.....	13
2.1.6	Cartesian robot	14
2.2	Components of a robot	14
2.3	Characteristics of Industrial Robots	15
2.4	Applications of industrial robots	15
2.5	Advantages and Disadvantages of Industrial Robots.....	17
2.5.1	Advantages of Industrial Robots	17
2.5.2	Disadvantages of Industrial Robots	17
3	METHODOLOGY	19
3.1	The design and production phase	19
3.2	The programming phase	19
3.3	The testing phase.....	19
4	DETAIL DESCRIPTION OF METHODOLOGY	20
4.1	Design phase.....	20
4.1.1	Adapter.....	20
4.1.2	Finger for the electrical gripper	24
4.1.3	Feed and catapulting system.....	28
4.1.4	Feed.....	28
4.1.5	Catapult.....	30
4.2	Programming phase.....	31
4.2.1	Robot programming	31

	5
4.2.2 Creating the Graphic User Interface (GUI)	34
4.3 Testing phase	37
5 CONCLUSION AND LIMITATION	38
5.1 Conclusion	38
5.2 Limitation.....	38
6 REFFERENCES	39
APPENDICES	

LIST OF FIGURES

Figure 1. The ABB IRB 120 robot	10
Figure 2. The Scare robot	12
Figure 3. The cylindrical shaped work envelope (A) and the cylindrical robot (B).	12
Figure 4. The work envelope of the spherical robot.	13
Figure 5. An articulated industrial robot.....	13
Figure 6. The parallel robot	14
Figure 7. The work-shaped envelope of the Cartesian robot (A) and the Cartesian robot (B).....	14
Figure 8. The flow chart of the thesis.	19
Figure 9. The 3D design of the adapters that was attached to the end effector (A) and the electrical gripper (B).	21
Figure 10. The detailed drawing of the first adapter attached to the end effector.	22
Figure 11. The detailed drawing of the second adapter attached to the electrical gripper.	23
Figure 12. The 3D view of the finger for the electric gripper.	24
Figure 13. The detailed drawing of the finger for the electric gripper.....	25
Figure 14. The complete assembly view.....	26
Figure 15. The exploded view of the components after assembly	27
Figure 16. The 3D view of the feed system.	28
Figure 17. The detailed drawing of the feed system.	29
Figure 18. The feed system without the Angry Birds (A) and the feed system with the angry birds (B).....	30
Figure 19. The catapult.	31
Figure 20. The overview of the concept of the GUI	34
Figure 21. The welcome screen of the ABB IRB 120 robot teach pendant.	35
Figure 22. The menu	35
Figure 23. The main screen	36
Figure 24. The game interface	36

LIST OF APPENDICES**APPENDIX 1.** Robot coordinates**APPENDIX 2.** Manual for the Graphic User Interface (GUI)

1 INTRODUCTION

1.1 The purpose of the thesis

Robotics is a general term used to describe the study and the use of robots or the science, technology, study and the application of robots. The use of robots has become common in various sectors such as: industrial, research, entertainment, law enforcement, space, agriculture, medical, nuclear, military, air borne and more [7]. However, more emphasis is laid on industrial robots since this project is based on its application in the physical Angry Bird game.

The Angry Bird game was developed by Rovio, a Finnish computer game developer. Players of the game are able to control a flock of colored wingless birds with a slingshot on their mobile phones and personal computers to launch the birds at pigs stationed in various structures made of different materials such as wood, ice and stone with the sole aim of destroying the pigs and retrieving the eggs that have been taken by the hungry pigs.

One physical Angry Bird game was displayed in the Hunan province of China at the Colorful World Amusement Park where players did not have to use a computer or mobile phone to play the game but experience it in real life. The purpose of the thesis is how effective it will be to use an articulated industrial robot to play the Angry Bird game since it can perform all the functions needed play to it.

The aims of the thesis are as follows:

- Design a program for a physical Angry Birds game using the IRB 120 Industrial Articulated Robot.
- Create a graphic user interface (GUI) with the help of a screen maker on the teach pendant for easy operation and controlling of the system.
- Design and produce an adapter and a finger for the electric gripper.
- Design and produce a feed system for the Angry Birds.
- Design and produce a catapulting or a slingshot system for the Angry Birds.

1.2 Background to the study

In early 2009, Rovio, a Finnish computer game developer, elected a team of designers to view and design a game proposed by a senior game designer Jaakko Lisalo. The proposal was to design a game in the form of a simulated screenshot featuring some angry-looking birds and hence the concept of Angry Birds was developed.

At that time, there was an outbreak of the swine flu pandemic and it was all over the news. The team that was elected to develop the game realized that the angry-looking birds needed an enemy and hence made pigs the enemies to reflect on the pandemic of the swine flu caused by pigs.

In December 2009, the game was released for Apples iPhones Operating system (iOS). Since its release in 2009, over 12 million copies have been purchased and it has been downloaded over 500 million times with paid downloads accounting for over 25% of total download and making it one of the popular and best sold games in the Apple App stores. The popularity of the Angry Birds has led to the creation of its version on personal computers and game consoles. In early 2010, Rovio started developing a variant of Angry Birds for Facebook users with some features that made it easy and interesting to play the game. [4]

This study will go a long way to show that it is possible to use the IRB 120 Industrial Articulated Robot to perform many tasks in our everyday activity in both the industry and the world of entertainment.

The IRB 120 is a multipurpose industrial robot and it is ABB's smallest industrial robot ever to be built. It has a mass of 25kg and can handle a load (payload) of 3kg, 580mm reach and six axis of free movement. Industrial robots will be discussed in more detail in chapter 2. [4]



Figure 1. The ABB IRB 120 robot

2 INDUSTRIAL ROBOTS

Industrial robots are mechanical devices that are used to replace humans to perform various tasks ranging from dangerous or repetitive with a high amount of accuracy. These industrial robots are automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes, and can be attached firmly in place or on a mobile platform. [6] [7]

Modern robots have become an integral and an important device in the automation, and industrial manufacturing environment where different industrial robots are used to execute diverse task.

The standard robot has a number of characteristics that reflect on their nature. These characteristics help identify the type of robot that is needed for the task. These characteristics include:

- The number of axes of motion
- Kinetic structure
- Work envelope
- Maximum payload
- Maximum speed
- Accuracy and
- Drive train

2.1 Types of industrial robots

There are many different industrial robots that are used for industrial purposes and the most commonly used are Scara, Articulated and the Gantry (parallel) robot. The most commonly used industrial robots are discussed in the following.

2.1.1 Scara robot

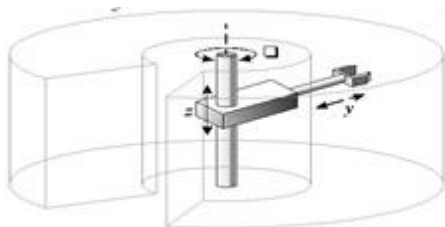
The Scara robot is one of the common robots used for mechanical assembly applications. It has two parallel joints that provide conformity in the selected plane as shown in Figure 2. [1]



Figure 2. The Scare robot

2.1.2 Cylindrical robot

Cylindrical robots consist of at least one rotary joint at the base and prismatic joint to connect the various links of the robot. The rotary joint is responsible for providing a rotational movement along the joint axes and the prismatic joint moves linear motion. The cylindrical robot is characterized by many operations within a cylindrical – shaped working envelope as shown in Figure 3. [1] [2]



(A)



(B)

Figure 3. The cylindrical shaped work envelope (A) and the cylindrical robot (B).

2.1.3 Spherical robot

The arm of the spherical robot is connected to the base with a twisting joint and a combination of one linear and two rotary joints. Its axes form a polar coordinates system and have its working envelope near spherical and spherical as shown in Figure 4. [1]

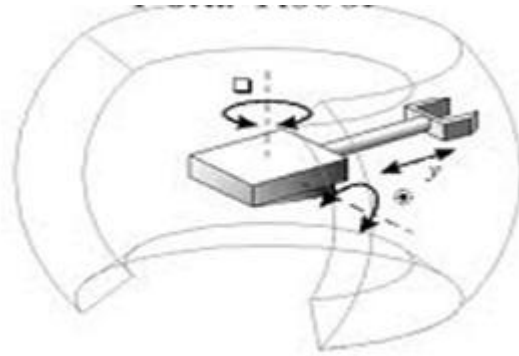


Figure 4. The work envelope of the spherical robot.

2.1.4 Articulated robot

The articulated robot as displayed in Figure 5 is another type of industrial robot with rotary joints, thus using rotational joint to access its work space. It is one of the most common and versatile industrial robots having six axes that permits a high level of freedom in each arm. This robot consists of an upper arm, forearm, shoulder, trunk, and a wrist with a capacity to rotate all joints simultaneously. It can be used to lift small to larger parts with a high amount of precision and accuracy. [1] [2]



Figure 5. An Articulated industrial robot.

2.1.5 Parallel robot

The parallel robot is a machine with a closed loop chain and having a higher speed, high stiffness, accuracy, compactness, time-saving of machine, high load/weight ratio and lower inertia as its advantages. Figure 6 below shows a parallel robot and its working envelope. [1]



Figure 6. The parallel robot

2.1.6 Cartesian robot

This robot has a linear movement which makes it to be more accurate than the rotary style robot such as the Scara robot. Furthermore, the Cartesian robots have offered a trade-off of lower speed for a greater repeatability. Usually, the robot has a rectangular work- shaped envelope as indicated in the Figure 7 [1] [2]

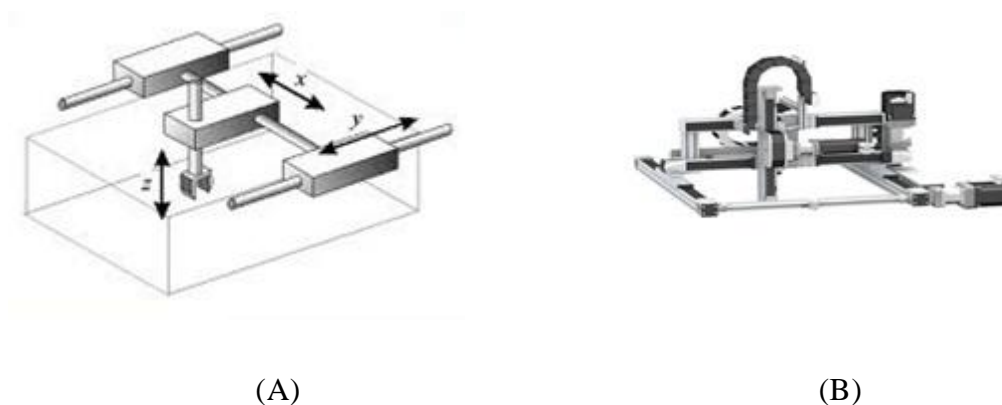


Figure 7. The work-shaped envelope of the Cartesian robot (A) and the Cartesian robot (B).

2.2 Components of a robot

Every industrial robot is unique in several ways and performs several functions based on the design and intended purpose of manufacture. To this effect, every robot could be made of different components, but in general, most industrial robots include the following: [6] [7]

- The controller. This is the brain of every robot and allows the robot to be connected to other systems for easy operation. It runs a set of instruction written codes called a program and helps to execute them in an orderly manner.
- Robot arm. This is the part of the industrial robot that has been designed to operate and handle the object in a similar way the human arm does. The industrial robot arm can vary in size and shape depending on the design and intended purpose of manufacture. Many industrial robots today are six axis operating robots hence make them effective.
- The end effector. This is connected to the robot arm and it functions as the robot hand, this is where the grippers are attached to the robot to enable effective operation.
- The teach pendant. This a device used in controlling a robot. This is used to teach the robot some targets and points and also use to design programs for the robot system.

2.3 Characteristics of Industrial Robots

Some of the characteristics of industrial robot include: [5]

- Payload: the maximum capacity of load the robot can carry without changing its specification.
- Reach: the maximum distance the robot can reach within its working envelope or space.
- Precision: this defines how accurately a robot can reach its defined target.
- Repeatability: this defines how accurate a robot can reach the same target repeatedly.

2.4 Applications of industrial robots

Robots can be used for different purposes and under various conditions. In most industries, the use of six-axis industrial robot has become rampant due to its flexibility and versatility. Industrial robots are rather becoming more prevalent and primarily used in automation applications of mass production industry where re-

peatability and accuracy are major concern. Some industrial applications of robots include: [6] [7]

1. Welding applications include:
 - Arc welding
 - Flux core welding
 - Plasma cutting
 - Resistance welding
 - Spot welding
 - Electric beam
 - Plasma welding
2. Material handling applications include:
 - Packaging
 - Machine loading and unloading
 - Material handling
 - Part transfer
 - Press tending
 - Injection molding
 - Pick and placing
 - Palletizing
3. Other applications include:
 - Bonding and sealing
 - Flame spraying
 - Grinding
 - Milling
 - Polishing
 - Water jet
 - Foundry
 - Material removal
 - Robotic assembling
 - Painting automation
 - Robotic coating

2.5 Advantages and Disadvantages of Industrial Robots

An industrial robot has its own advantages and disadvantages. Some of these are discussed below. [7]

2.5.1 Advantages of Industrial Robots

- Reduction in operation cost. Since robots can replace a lot of workers in a factory, it reduces the labor cost. Furthermore, robots do not necessarily need certain environmental comfort such as lighting, air conditioning and noise protection, which could increase the operational cost of production.
- Improvement of product quality and consistency. Robots have the ability to produce a product repeatedly and accurately without change in any of the products. For example, in the mass production of a component.
- Increase in production output rates. The number of product produced per unit time increases, the reason being that robots work very fast and tirelessly without any break except in the case of repairs or maintenance.
- Reduction in material waste and increase in yield. Robots are more accurate than humans and therefore do not waste materials during their operation therefore, increasing yield.
- Complying with safety rules and improvement of safety and health
- Reduction in labor turnover. Unlike humans, robots will always be available to work except in the case of repairs and maintenance.

2.5.2 Disadvantages of Industrial Robots

- The initial investment cost of industrial robots is high due to cost of equipment and installation, need for peripherals, need for training and programming.
- They have limited duties; they can only execute what they have been programmed to do.
- People can lose jobs in factories due to its introduction.
- It needs a supply of power for its operation.

- It needs maintenance to keep it running and this will mean extra cost for the factory.
- Industrial robots lack the capability to respond to emergencies unless it is predicted and included in the system. The need for safety measures are required to ensure work and operator safety.

3 METHODOLOGY

The thesis is divided into three main phases as shown in Figure 8. These include:

- The design and production phase
- The programming phase
- The testing phase

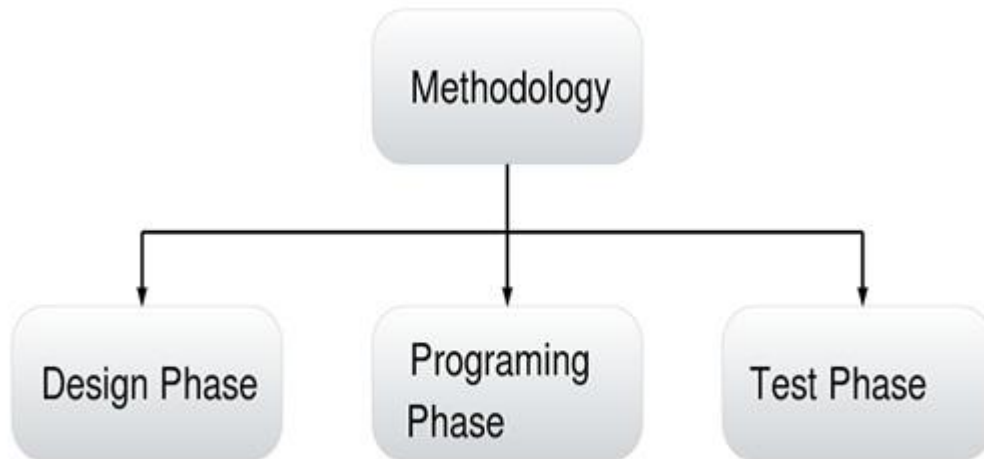


Figure 8. The flow chart of the thesis.

3.1 The design and production phase

This part deals with the design and production of the adapters, feed system and the catapulting or slingshot system.

With the aid of the NX6 3D software, the components were designed for production.

3.2 The programming phase

This part deals with the programming of the robot and also creating the graphic user interface (GUI) on the teach pendant with the aid of robot studio software.

3.3 The testing phase

After the completion of the project, the last phase was to test and see if the final results meet the objective of the thesis.

4 DETAIL DESCRIPTION OF METHODOLOGY

4.1 Design phase

Various components were designed with the aid of the NX6 software. The components are as follow:

- Adapter
- Fingers for the electrical gripper
- Feed and catapulting system

4.1.1 Adapter

The adapter was designed to help to attach the electric gripper to the end effector of the robot. An end effector is the part of the robot that helps to connect the hand of the robot to other devices, such as grippers.

In all, two adapters were designed. One part of adapter was attached to the end effector and the other attach to the electric gripper.

Acrylonitrile Butadiene Styrene (ABS) plastic was the material used in the production of the two adapters after which it was hardened by injecting it with polyester hardener.

ABS plastic was used due to the following reasons

- It is a common amorphous thermoplastic and therefore has a true melting point.
- The material combines the strength and rigidity of the Acrylonitrile and styrene polymer with the toughness of the poly-butadiene rubber.

Figure 9 below shows the 3D design and Figures 10 and 11 the detail drawings of the adapters.

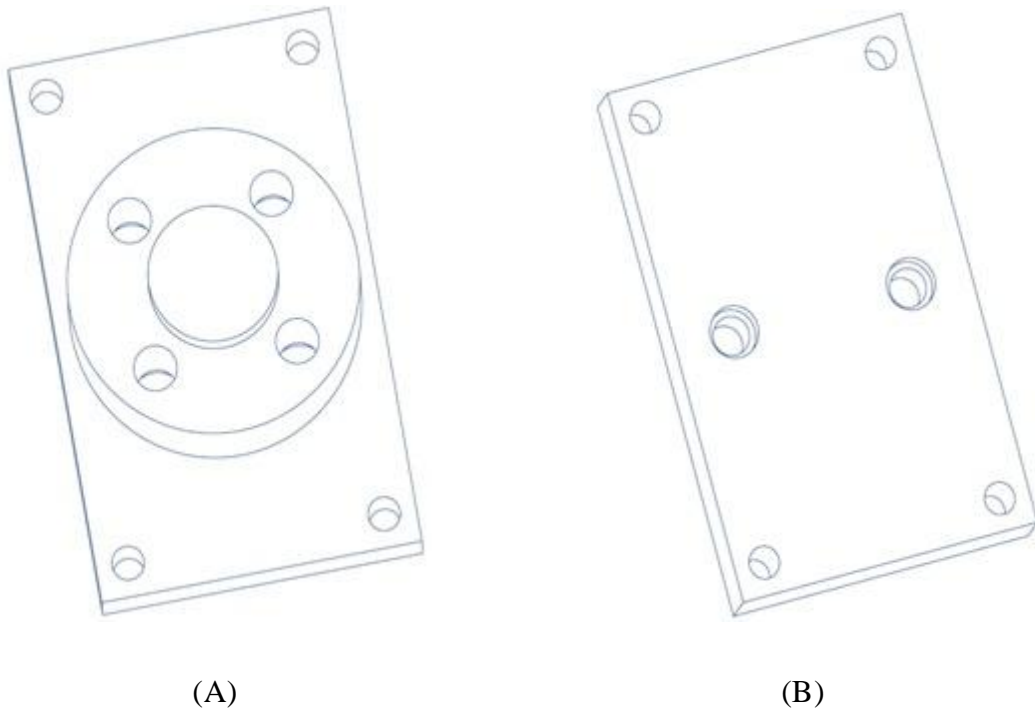


Figure 9. The 3D design of the adapters that was attached to the end effector (A) and the electrical gripper (B).

Below are the detail drawings and all the measurements that were used during the design of the adapter A.

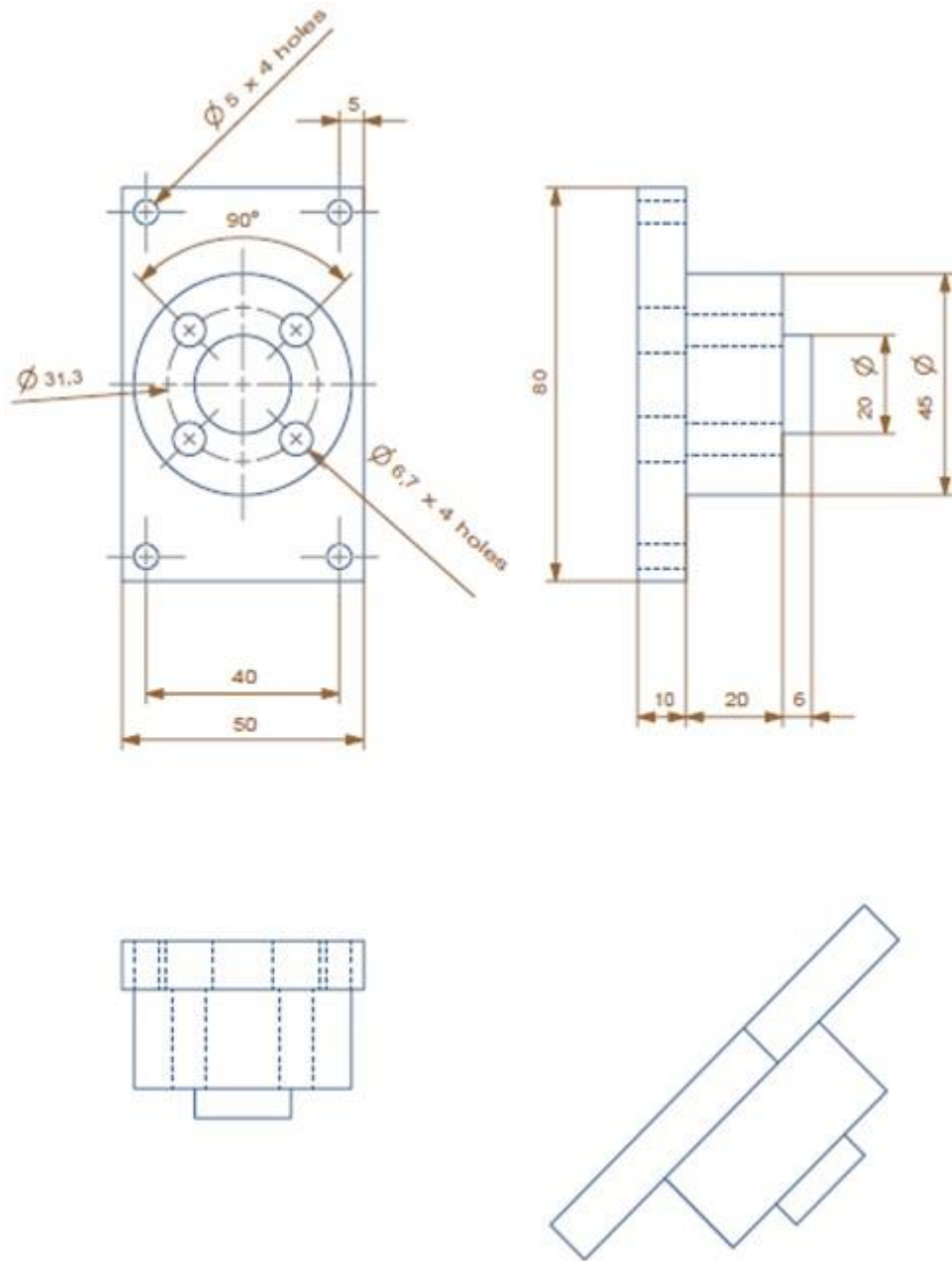


Figure 10. The detail drawing of the first adapter attached to the end effector.

Below is the detail drawings and all the measurements that were used during the design of the adapter B.

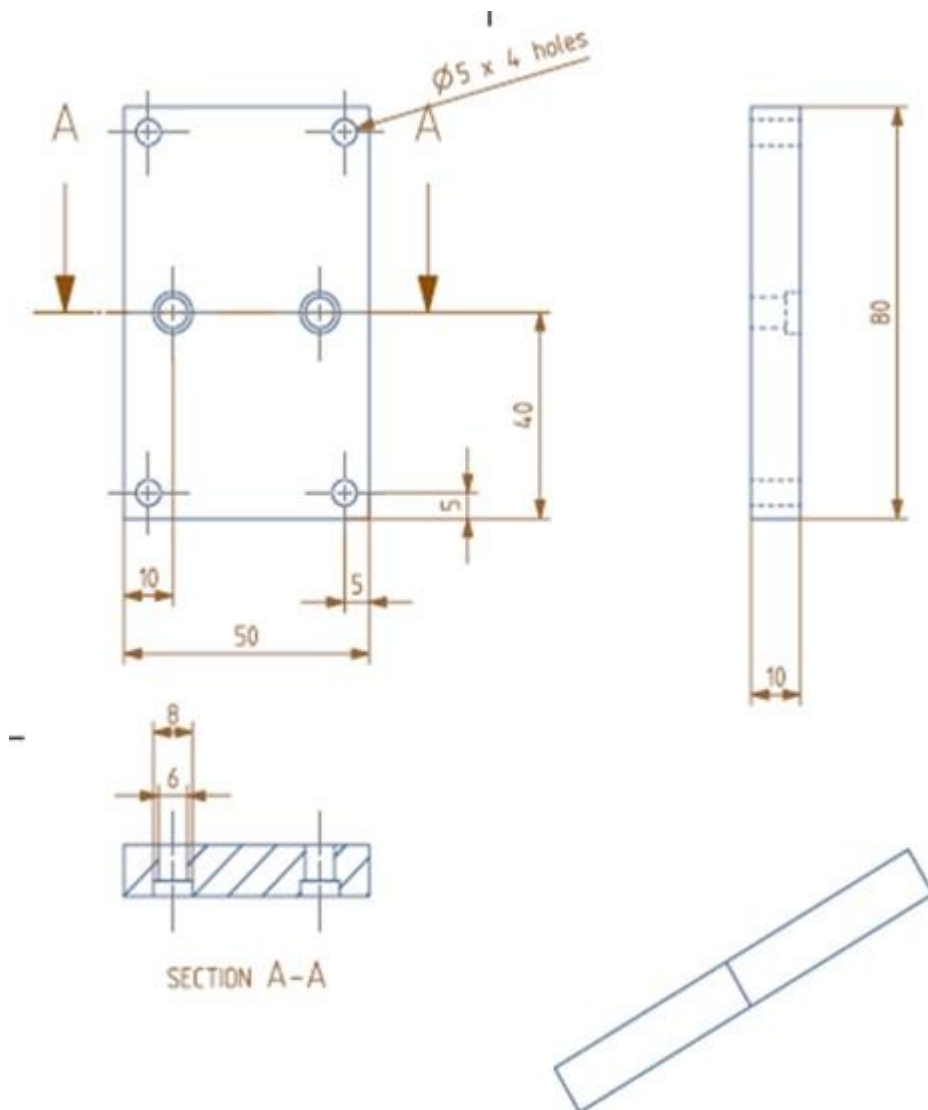


Figure 11. The detail drawing of the second adapter attached to the electrical gripper.

4.1.2 Finger for the electrical gripper

The finger of the electric gripper was designed in such a way that the finger can pick the angry birds from the feed system and place it in the catapulting system. At the same time, it has a grasping side that can hold the catapult, pull and release it. Two of the components were produced though only one was designed as shown in Figure 12 and Figure 13 the detail drawing of the finger.

Acrylonitrile Butadiene Styrene (ABS) plastic was the material used in the production of the finger after which it was hardened by injecting it with polyester hardener. Below are the 3D view and the detail drawing of the component.

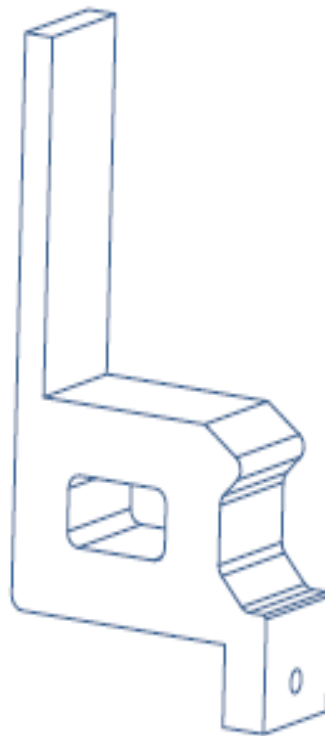


Figure 12. The 3D view of the finger for the electric gripper.

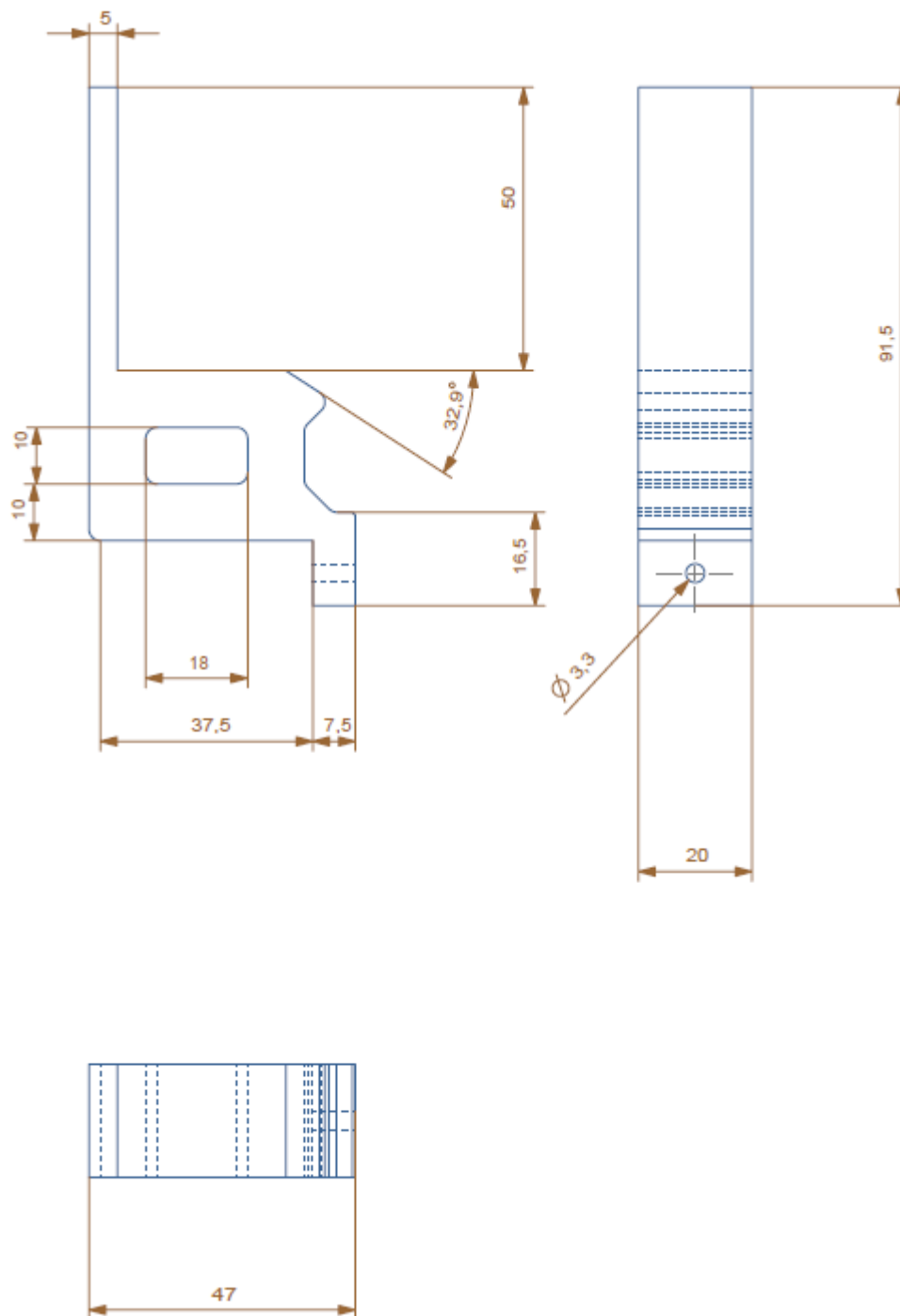


Figure 13. The detail drawing of the finger for the electric gripper.

Figure 14 shows the complete assembly and Figure 15 also shows the exploded view of the adapters, electrical gripper and the finger for the gripper.

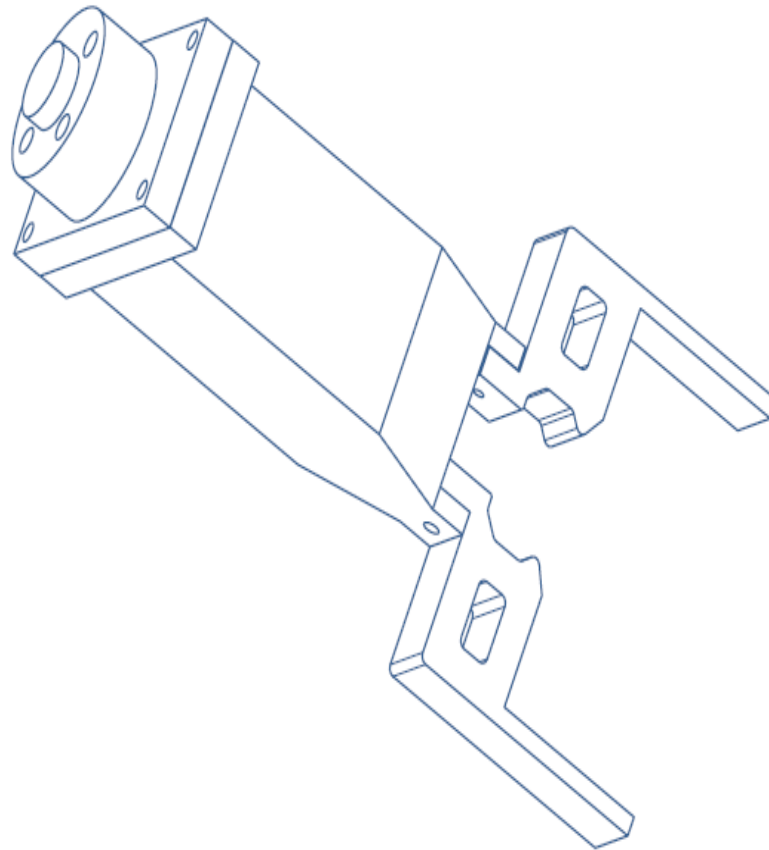
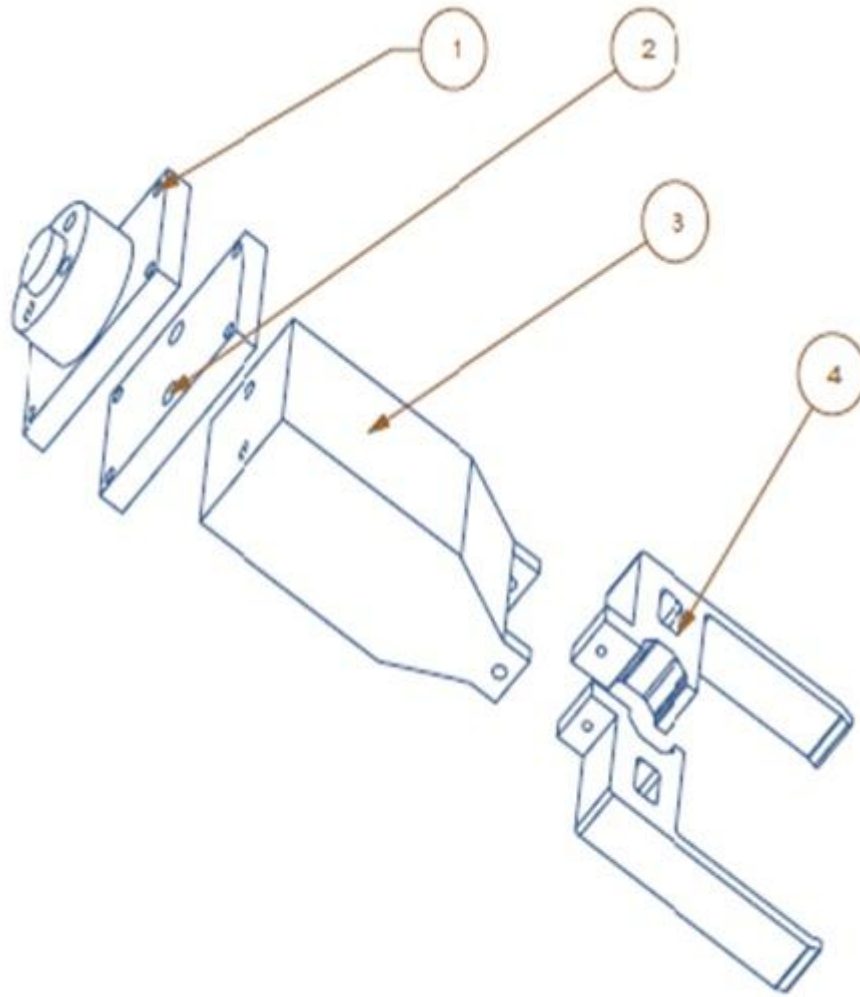


Figure 14. The complete assembly view.



4	FINGER	2
3	GRIPPER1	1
2	MODEL1ADAP2	1
1	MODEL1_ADAP1	1
PC NO	PART NAME	QTY

Figure 15. The exploded view of the components after assembly

4.1.3 Feed and catapulting system

The feed and the catapult are two different components that were assembled to get a whole system.

4.1.4 Feed

This is where the angry birds are placed for the robot to pick. It is designed so that it is inclined at an angle of 60° so that the angry birds will slide to the same point the first bird was picked. In effect, it means that the robot will always have to pick the birds at the same point in the feed system. Figures 16, 17 and 18 below are the 3D view, the detail drawing and the picture of the feed system respectively.

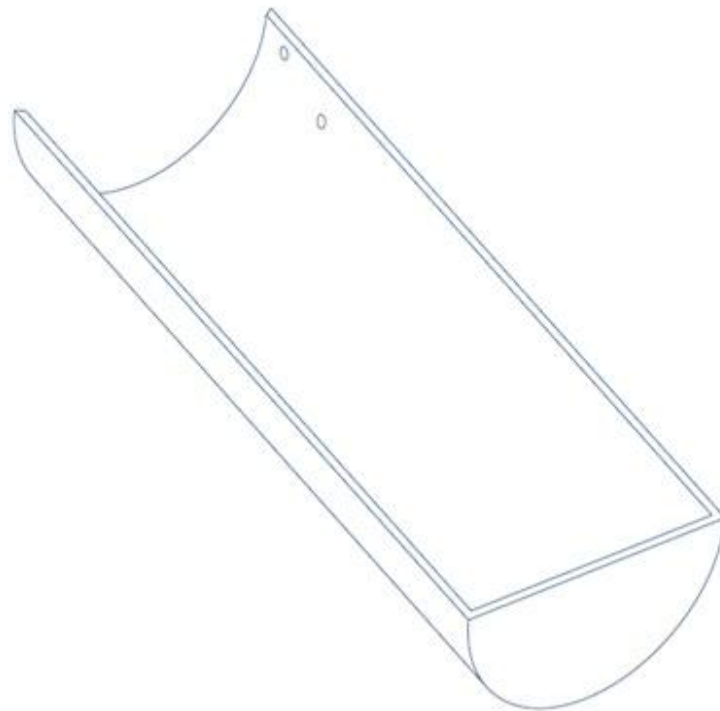


Figure 16. The 3D view of the feed system.

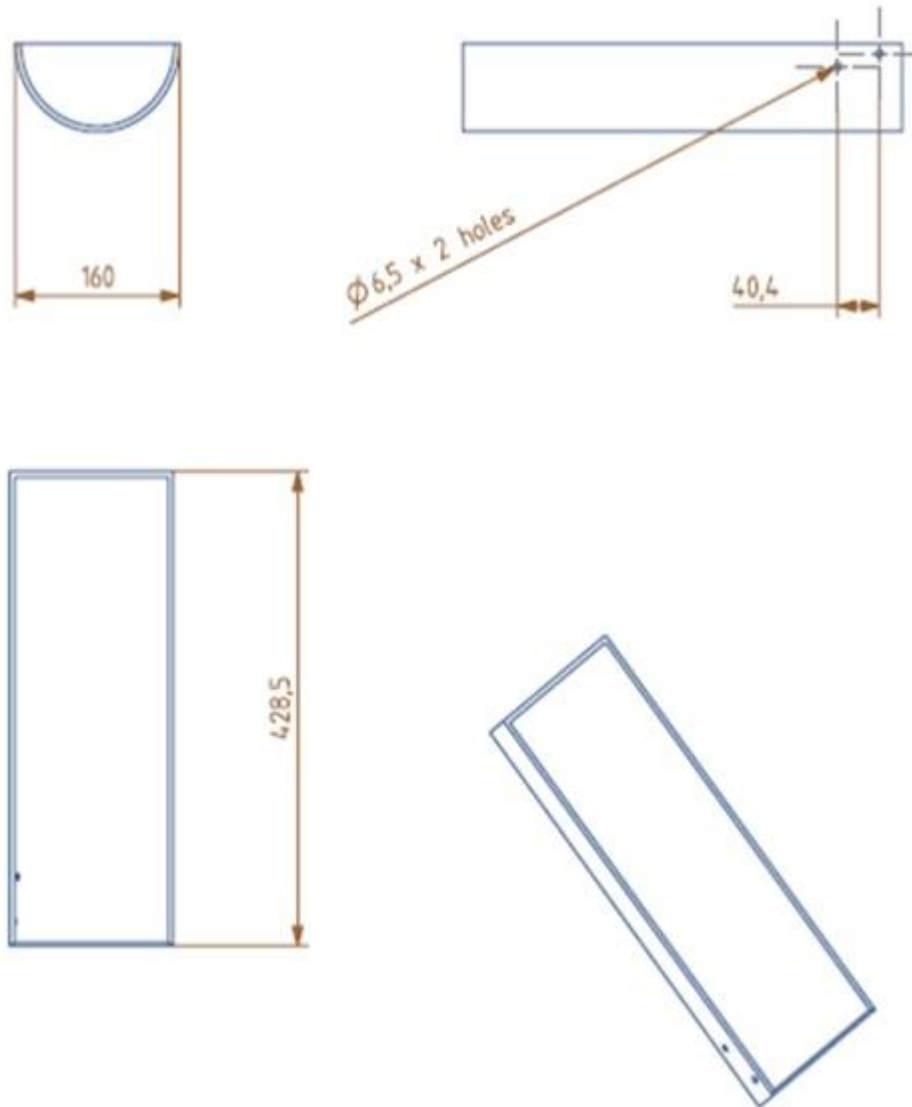


Figure 17. The detailed drawing of the feed system.

The pictures of the feed are shown below.



(A)



(B)

Figure 18. The feed system without the Angry Birds (A) and the feed system with the Angry Birds (B).

4.1.5 Catapult

The catapulting system displayed in Figure 19 is made of three main parts:

- The funnel. This is the part of the catapult in which the robot places the angry birds after picking it from the feed system.
- Elastic rubber. The properties of the elastic rubber enable the funnel to be pulled by the robot in order to shot the angry birds at a certain distance and return the funnel to its original position.
- The frame. This was used to provide rigid support for the feed and catapulting system and also serves a fixing point of the elastic rubber and the funnel.

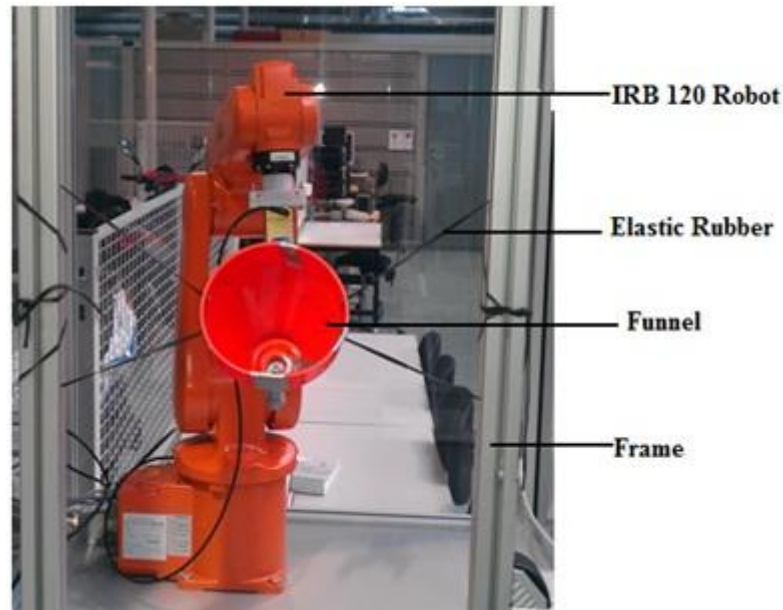


Figure 19. The Catapult.

4.2 Programming phase

The programming phase was categorized into two parts

- Robot programming
- Creating the Graphic User Interface (GUI)

4.2.1 Robot programming

This was done by predefining certain points so that the robot can follow the points in order to

- Pick the angry birds from the feed
- Place it in the funnel
- Pull the funnel to a certain distance and release

Furthermore, the electric gripper also could open and close for picking and releasing respectively.

- Picking the angry bird from the feed: at this stage, the robot was programmed to move from the home position to the feed in order to pick the

angry bird. Beneath are the points that were predefined for the robot to follow in the program.

```
PROC main ()  
  
MoveL home, v1000, z0, electric_gripper;  
  
MoveL p_1, v1000, fine, electric_gripper;  
elgrip_open;  
MoveL p_2, v1000, z0, electric_gripper;  
MoveL p_3, v1000, fine, electric_gripper;  
elgrip_close;  
MoveL p_4, v1000, z0, electric_gripper;
```

PROC main means that the routine was created in the main program. MoveL means the robot should move in a linear direction to home position. The home position is just a name given to the point where the robot moves to during that command, so applies for p_1, p_2, p_3 and p_4.

V1000 is the speed at which the robot moves from one point to another. Z0 or fine shows the closeness of the point at which the robot must reach before moving to the next point. Even though both z0 and fine could be used for the same purpose, fine take the robot closer to the intended point than z0.

The electric gripper in the program shows the type of tool that was used. Elgrip_close and elgrip_open denotes the closing and opening of the electric gripper respectively in the program.

- Placing the angry bird in the funnel: after picking the angry bird from the feed, the next thing was to place the angry bird in the funnel. Below are the points that were predefined for the robot to follow in the program. The explanations of the points are the same as above.

```
MoveL p_5, v1000, z0, electric_gripper;  
MoveL p_6, v1000, z0, electric_gripper;  
MoveL p_7, v1000, z0, electric_gripper;  
MoveL p_8, v1000, z0, electric_gripper;  
MoveL p_9, v1000, z0, electric_gripper;  
MoveL p_10, v1000, fine, electric_gripper;  
elgrip_open;
```


- Pulling the funnel to a certain distance and release: after placing the angry bird in the funnel, the robot moves back to pull the funnel to a certain distance and release. Below are the points that were predefined for the robot to follow in the program. The explanations of the points are the same as above.

```

MoveL p_11, v1000, z0, electric_gripper;
MoveL p_12, v1000, z0, electric_gripper;
MoveL p_13, v1000, z0, electric_gripper;
MoveL p_14b, v1000, z0, electric_gripper;
MoveL p_15b, v1000, fine, electric_gripper;
elgrip_close;
MoveL p_14, v1000, z0, electric_gripper;
MoveL p_15, v1000, fine, electric_gripper;
elgrip_open;

```

The robot was then programmed to move back to the home position for the next cycle. Below are the points that were predefined for the robot to follow in the program. The explanations of the points are the same as above. ENDPROC indicates the end of the program.

```

MoveL home, v1000, z0, electric_gripper;
elgrip_close;
ENDPROC

```

Every industrial robot has an x, y and z values known as the coordinate system. This values are been determined by the robot itself. During the programming, certain coordinate system was created by the robot as shown in Appendix 1. An example of the coordinate is shown below.

```

CONST robtarget home:=[[85.84,-313.23,466.86]

```

The values represents the x, y and z coordinate respectively when the robot was at home position.

4.2.2 Creating the Graphic User Interface (GUI)

The GUI was created with the aid of the ScreenMaker under robot studio. ScreenMaker is a tool in robot studio used in creating or developing customized screen or GUI on a flex or teach pendant.

GUI makes it easier for any person to be able to operate a robot once it is been programmed very well. This goes on to prove that it requires and expect to program a robot but does not necessarily require and expect to operate the robot provided the GUI is created. GUI consists mainly of two parts:

- The view part. Lay out and configuration of controls that appears on the flex or teach pendant.
- The process part. Event handlers that respond to programs that have be predefined.

The whole concept of the GUI is shown below

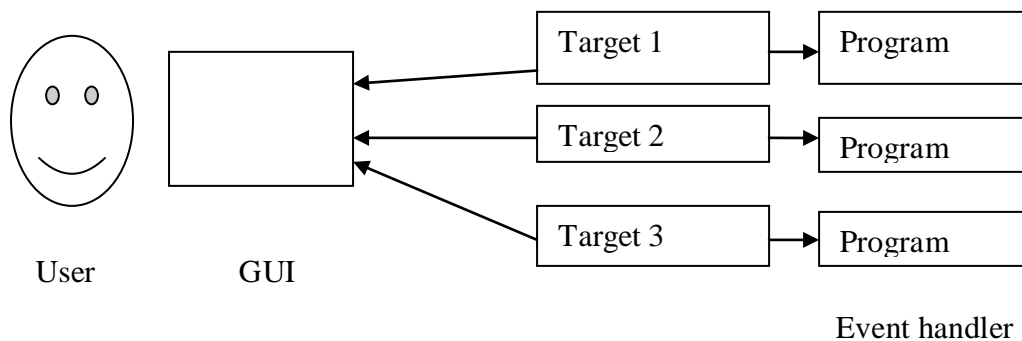


Figure 20. The overview of the concept of the GUI

From Figure 20, target 1, 2 and 3 are all GUI having certain predefined robot programs that the operator does not see. The operator only needs to understand what each interface on the screen means and how it operates without him knowing the program loaded in each interface. The manual for the design of the GUI can be found in Appendix 2 and the figures below give a clear picture of the GUI that was designed for the thesis. [5]

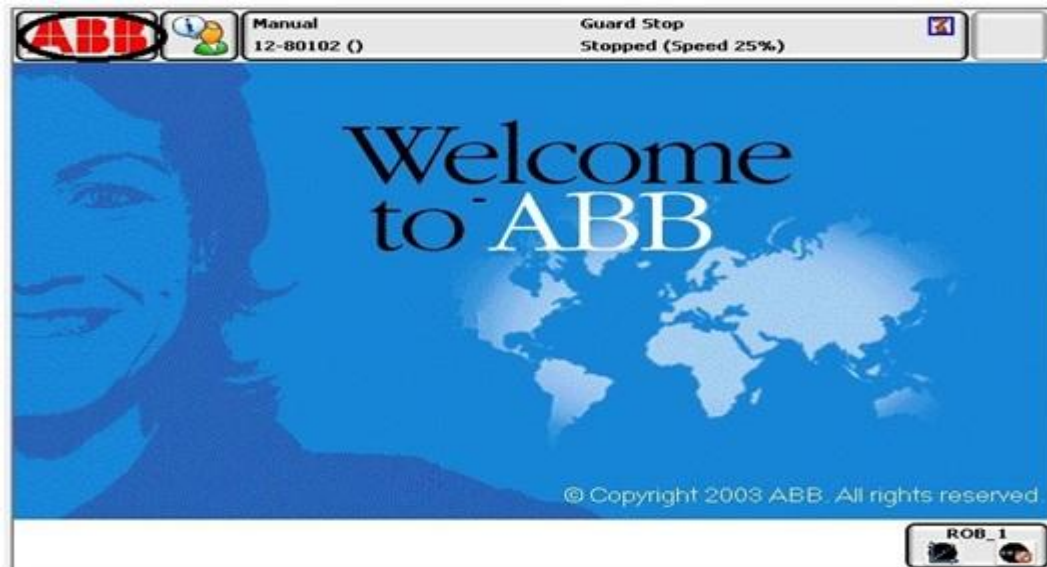


Figure 21. The welcome screen of the ABB IRB 120 robot teach pendant.

This interface in figure 21 is what appears on the teach pendant when the IRB 120 robot is switched on. The user has to click on the ABB logo (marked with a black circle). This will then take the user to the next interface shown below.

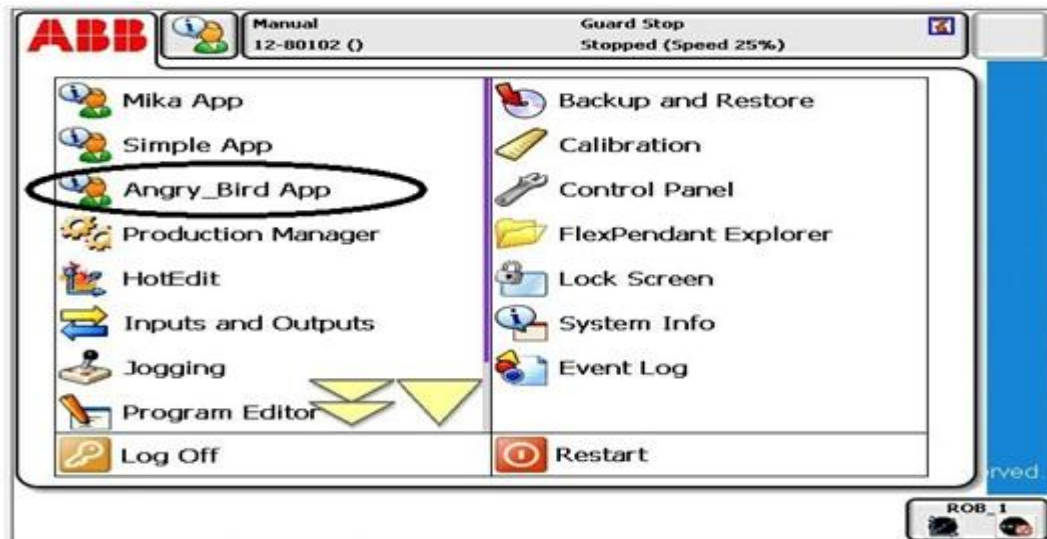


Figure 22. The menu

This interface allows the user to select the type of program to run. The user then has to click on the Angry_Bird App icon to open the main screen as shown in Figure 22.



Figure 23. The main screen

Figure 23, the main screen has two buttons, the welcome and the start game buttons. The welcome button (circled black) displays the picture of the Angry Bird game. This is to serve as a welcome note to the user.

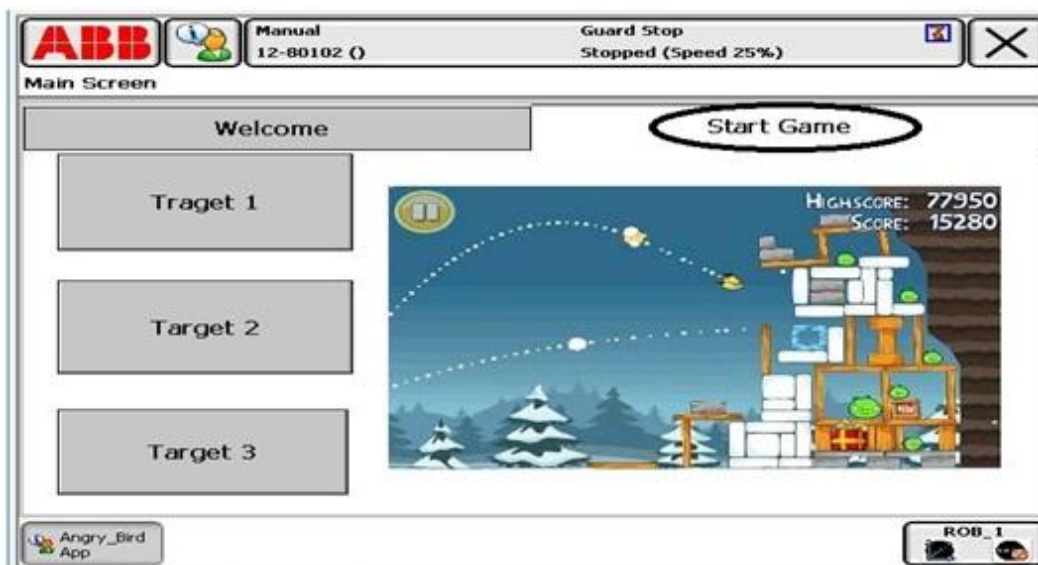


Figure 24. The game interface

The game interface in Figure 24 has three buttons namely Target 1, Target2 and Target 3. These buttons have special programs imbedded in them that send infor-

mation to the robot. These instructions or commands have been previously explained on the programming phase of the thesis.

4.3 Testing phase

The testing phase proved to be very vital during the thesis; it gave a clear picture, understanding, and an opportunity to make some changes and modifications. Some of the changes and modifications were made and they are as follow:

- During the testing phase of the thesis, some problems were encountered with the finger for the electric gripper. The fingers were unable to grip the funnel; this led to the modification of the finger to ensure that it was able to grip the funnel.
- Certain points in the robot program were modified to enhance better performance of the robot. This made it even better than what was done initially.

The testing activities that were carried out include:

- Testing of programmed points
- Testing of the Graphic User Interface (GUI)
- Testing for the strength and durability of the adapters and the finger for the electric gripper
- Testing for the strength of the frame for the feed and catapult system
- The overall functioning of the game.

One major observation made was that the pulling distance of the catapult was small due to the fact that the working area of the robot was limited to prevent damage to the glass cage.

In all, the testing was successful, every design, program and modification made worked as expected.

5 CONCLUSION AND LIMITATION

5.1 Conclusion

Based on the results from the testing phase, it is fair to conclude that the thesis was a success: the aims of the thesis were mostly attained and the question as to whether it is feasible and effective to use the ABB IRB 120 articulated industrial robot to play an Angry Bird was answered.

Nevertheless, one major setback was encountered and it was the fact that the robot could not give enough pulling distance for target 1 and 2. This is because the robot has a limited working area to protect it from breaking the glass cage. For this reason, the end results for target 1 and 2 did not meet the initial targeted point.

Furthermore, the funnel that was used in the catapult for holding the birds before shooting did not give any room for creating enough shooting target. For this reason, only three target points were created for this thesis.

In conclusion, one can say without a single doubt that, it is possible to use the ABB IRB 120 articulated industrial robot to program and create a physical Angry Bird game.

5.2 Limitation

For the above conclusion, it is recommended that:

- When next a similar topic is to be studied, the working area for the ABB IRB 120 articulated industrial robot at the Technobothnia laboratory should be increased if possible so as to help create more moving option and given enough room for the robot to operate.
- Though the feed and the catapulting system in this thesis meet the aim of the study, it is recommended that a new catapulting system should be designed if possible to help attain different target points other than the three points created in this study.

6 REFERENCES

1. **Niku, Saeed B.** *Introduction to Robotics, Analysis, Control, Application. Second Edition.* s.l. : Don Fowler, 2011.
2. **Engelberger, Joseph F.** *Robotics in practice.* London : KOGAN PAGE in association with Avebury Publishing Company, 1980.
3. [Online] [Cited: February 26, 2012.]
<http://www.robotmonkeyboy.com/different-types-of-robots.html>.
4. *Application manual SreenMaker.* Vasteras : ABB AB Robotics Products, 2009.
5. **Wernholt, Erik.** *On Multivariable and Nonlinear Identification of Industrial Robot.* Linkoping : UniTryck, 2004.
6. *ABB web site.* [Online] ABB. [Cited: March 13, 2012.]
<http://www.abb.com/product/seitp327/BE2EEF38406EACA4C125762000319182.aspx>.
7. *Rovio web site.* [Online] Rovio. [Cited: March 15, 2012.]
<http://www.rovio.com/en/about-us/Company>.
8. *robotworx web site.* [Online] RobotWorx Automation. [Cited: March 13, 2012.]
<http://www.robots.com/applications>.

APPENDIX 1. Robot coordinates.

```

MODULE MainModule
  CONST robtarget home:=[[85.84,-313.23,466.86],[0.03609,-
0.0385929,0.995743,-0.0755207],[-
1,0,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_1:=[[429.69,-
207.06,402.20],[0.0321683,0.0328758,0.993547,-0.103676],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_2:=[[465.83,-
262.63,140.37],[0.0347499,0.0244718,0.968498,-0.245369],[-1,-
1,4,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_4:=[[463.09,-
261.72,368.05],[0.0308519,0.0237886,0.968956,-0.244145],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_5:=[[203.79,-
425.95,513.34],[0.0326107,0.240655,-0.94303,0.227412],[-1,-
1,2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_6:=[[91.91,-296.76,710.14],[0.0114538,-
0.155939,0.878731,-0.450981],[-1,-
1,2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_7:=[[63.83,-347.09,798.63],[0.0520183,-
0.0752433,0.77466,-0.625727],[-1,-
1,2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_8:=[[61.28,-524.06,713.70],[0.0559888,-
0.0159603,0.804473,-0.591129],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_9:=[[64.32,-549.61,679.51],[0.0534886,-
0.0163942,0.822828,-0.565531],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_10:=[[81.58,-584.83,559.11],[0.0519497,-
0.018206,0.891905,-0.44886],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_11:=[[55.02,-470.97,786.04],[0.0625474,-
0.0147876,0.750598,-0.657626],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_12:=[[46.15,-397.17,466.01],[0.00436806,-
0.0195631,0.997725,-0.0643725],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_14:=[[93.95,-472.29,355.02],[0.0266411,-
0.0265774,-0.993273,0.109509],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_15:=[[79.78,-263.96,300.19],[0.105738,-
0.00131136,-0.993382,0.0448258],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_16:=[[109.34,-
219.21,277.87],[0.13261,0.0860237,-0.986901,0.0322653],[-1,-
1,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_14b:=[[86.17,-517.95,375.45],[0.0392827,-
0.0372919,0.963581,-0.261873],[-
1,0,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p_15b:=[[86.17,-517.94,375.44],[0.0392957,-
0.0372338,0.963589,-0.261851],[-
1,0,3,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

```


APPENDIX 2. Manual for the Graphic User Interface (GUI)

1 Introduction

1.1. Introduction to ScreenMaker

1 Introduction

1.1. Introduction to ScreenMaker

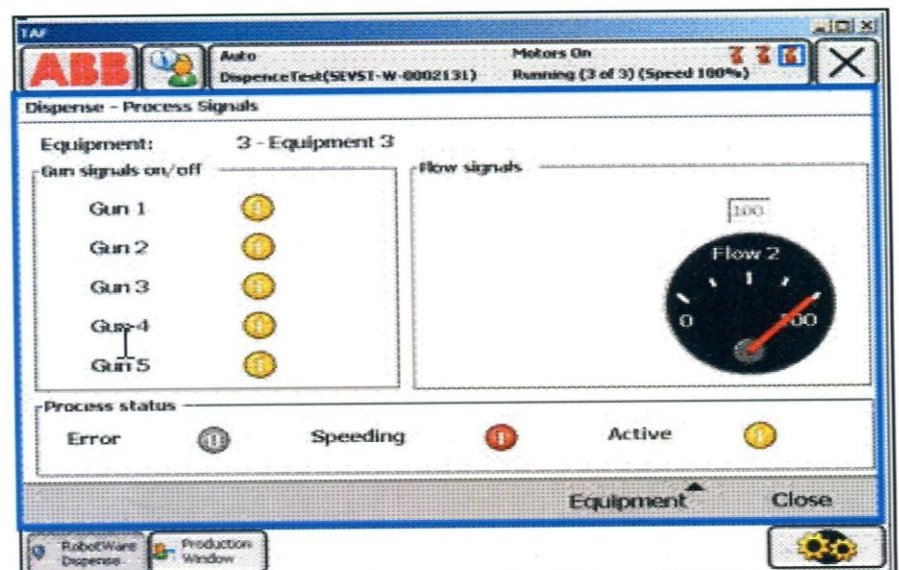
What is ScreenMaker?

ScreenMaker is a tool in RobotStudio for developing custom screens. It is used to create customized FlexPendant GUIs without the need to learn Visual Studio development environment and .NET programming.

Why ScreenMaker?

A customized operator interface on the factory floor is the key to a simple robotic system. A well-designed custom operator interface presents the right amount of information at the right time and in the right format to the user, as such the training time and downtime (due to operating errors) are minimal. However, customized user interfaces are expensive and very time-consuming to develop. Currently, an understanding of some object-oriented programming languages (such as C, C++; VB and C#) and development framework (.NET, Visual Studio) are required to develop screens. Since, this is a requirement for IT professionals and not for the robotics industry whose workforce is generally accustomed to simple programming languages such as BASIC and RAPID; ScreenMaker is used.

GUI concepts



© Copyright 2009 ABB. All rights reserved.

xx080000226

A GUI makes it easier for people to work with industrial robots by presenting a visual front to the internal workings of a robotic system. For FlexPendant GUI applications, the graphical interface consists of a number of screens, each occupying the user window area (the blue box

Continues on next page

1 Introduction

1.1. Introduction to ScreenMaker

Continued

in the figure above) of the FlexPendant touch screen. A FlexPendant screen is then composed of a number of smaller graphical components in a design layout. Typical controls (sometimes referred as widgets or graphic components) include buttons, menus, images, and text fields.

A user interacts with a GUI application by:

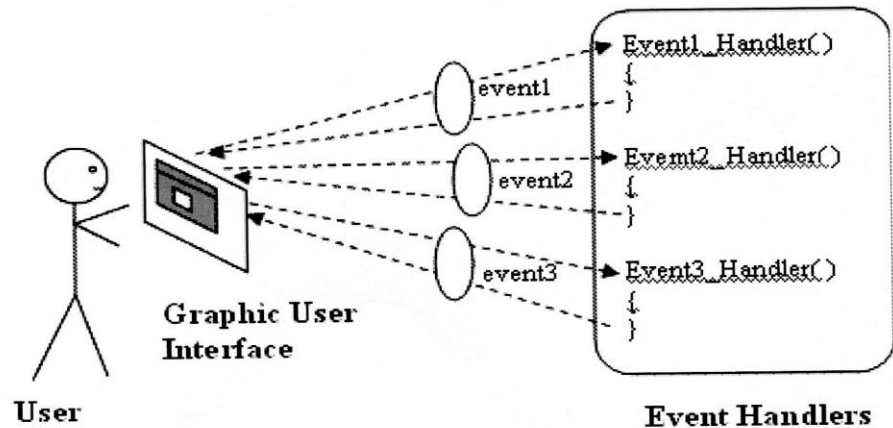
- Clicking a button
- Selecting from a menu
- Typing a text in a text box
- Scrolling

An action such as clicking a button is called an event. Whenever an action is performed, an event is sent to the GUI application. The exact content of an event is solely dependent on the graphic component itself. Different components trigger different types of events. The GUI application responds to the events in the order generated by the user. This is called event-driven programming, since the main flow of a GUI application is dictated by events rather than being sequential from start to finish. Due to the unpredictability of the user's actions, one major task in developing a robust GUI application is to ensure that it works correctly no matter what the user does. Of course, a GUI application can, and actually does, ignore events that are irrelevant.

The event handler holds sets of actions to be executed after an event occurs. Similar to trap routines in the RAPID program, the event handler allows the implementation of application-specific logic, such as running a RAPID program, opening a gripper, processing logic or calculating.

In summary, from a developer's point of view, A GUI consists of at least two parts:

- *the view part*: layout and configuration of controls
- *the process part*: event handlers that respond to events



xx0800000227

Modern GUI development environments often provide a form designer, a WYSIWYG tool to allow the user to select, position and configure the widgets. As for event handlers, typically the developer must use a special programming language recommended by the development environment.

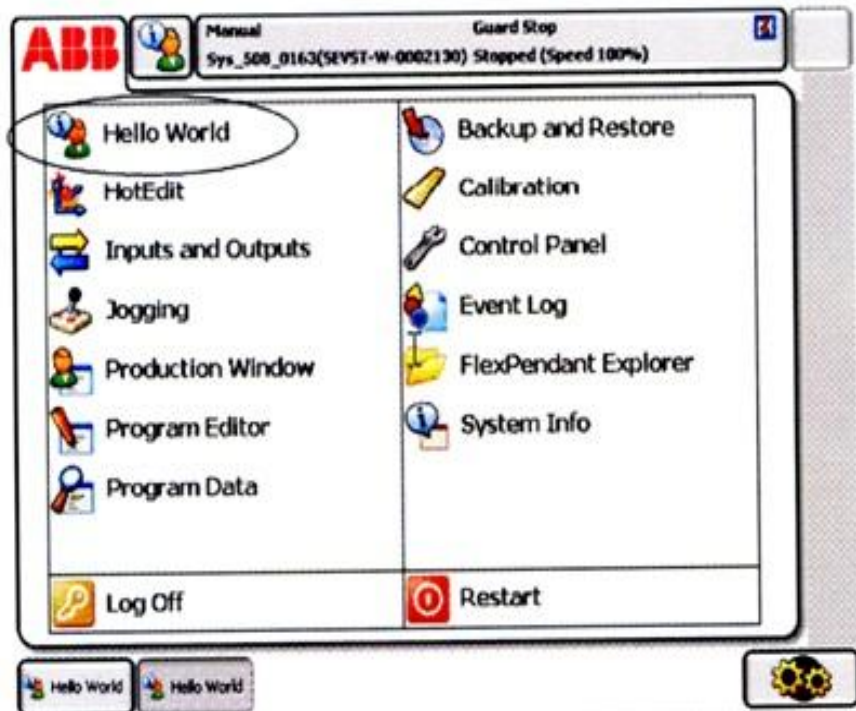
Continues on next page

1 Introduction

1.1. Introduction to ScreenMaker

Continued

FlexPendant concepts



xx0800000228

Running Windows CE, the ABB FlexPendant has limited CPU power and memory compared to a PC. A custom GUI application must therefore be placed in the designated folders on the controller hard drive before being loaded. Once loaded, it can be found in the ABB menu as seen in the figure above. Clicking the menu item will launch the GUI application.

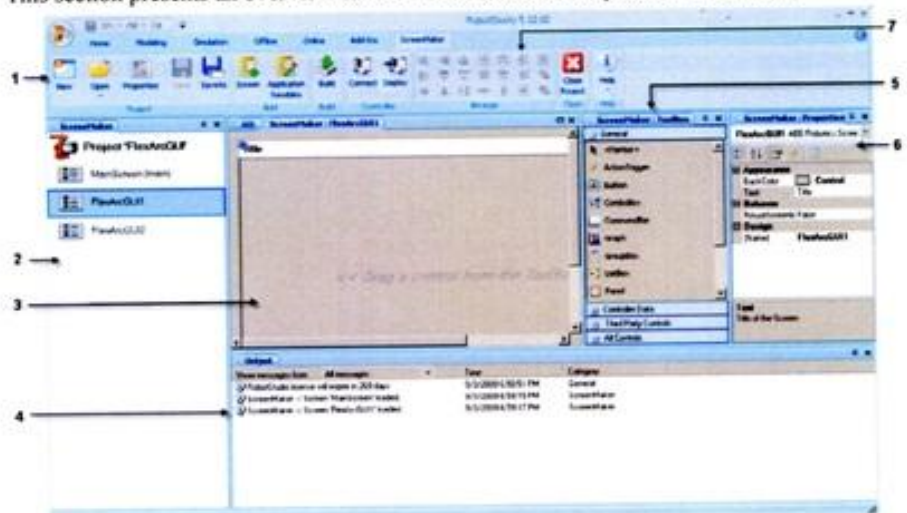
As the robot controller is the one actually controlling the robot and its peripheral equipment by executing a RAPID program, a GUI application needs to communicate with the RAPID program server in order to read and write RAPID variables and set or reset I/O signals.

It is essential for RAPID programmers to understand that there are two different softwares controlling a work cell: an event-driven GUI application running on the FlexPendant, and a sequential RAPID program running in the controller. These reside on different CPUs and use different operating systems, so communication and coordination are important and must be carefully designed.

1.4. Development environment

Overview

This section presents an overview of the ScreenMaker development environment.



en0900000584

Parts	Description
1	Ribbon Displays group of icons organized in a logical sequence of functions. See Ribbon on page 14 .
2	Project explorer Shows the active screen project and lists the screens that are defined in the project. For more information, see Managing ScreenMaker projects on page 26 .
3	Design surface Layout to design the screen with the available controls. For more information, see Form designer on page 35 .
4	Output window Displays information about the events that occur during ScreenMaker development.
5	ToolBox Displays a list of available controls. For more information, see ToolBox on page 15 .
6	Properties window Contains the available properties and events of the selected control(s). The value of the properties can either be a fixed value or a link to an IRC5 data or an Application Variable. For more information, see Properties window on page 17 .
7	Arrange Displays icons for resizing and positioning controls on the design surface. See Arrange on page 14 .

1 Introduction

1.4. Development environment

Continued

Ribbon

The ScreenMaker ribbon tab contains a group of icons organized in a logical sequence of functions that facilitates the user in managing SscreenMaker projects.



en0900000452

The Ribbon is categorized into the following groups:

Group	Description
Project	Facilitates the user to manage ScreenMaker project. See <i>Managing ScreenMaker projects on page 26</i> .
Add	Facilitates the user to add screen and application variables. See <i>Managing screens on page 29</i> and <i>Managing application variables on page 34</i> .
Build	Facilitates the user to build a project. See <i>Building a project on page 33</i> .
Controller	Facilitates the user to connect and deploy to the controller. See <i>Connecting to controller on page 32</i> and <i>Deploying to controller on page 33</i> .
Arrange	Facilitates the user to resize and position the controls on the design surface. See <i>Arrange on page 14</i> .
Close	Facilitates the user to close a project.
Help	Facilitates the user to open the ScreenMaker help.

Arrange

This toolbar displays icons for resizing and positioning controls on the design surface.

NOTE: The icons are enabled once you select a control or group of controls on the design surface.

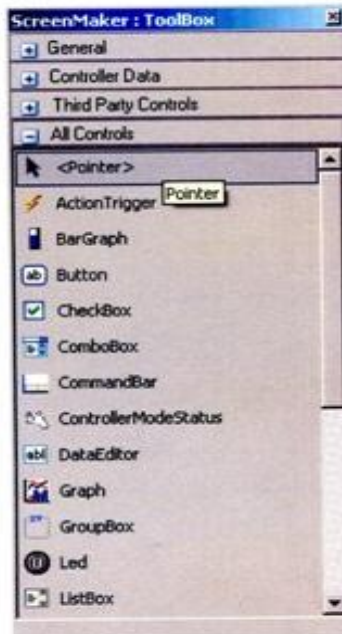


en0900000592

Continues on next page

ToolBox

ToolBox acts a container for holding all the available controls that can be placed on a screen.



en0900000407

The following table displays the GUI controls that can be dragged on to the design surface.

Control	Description
ActionTrigger	Allows to run a list of actions when either a signal or rapid data changes
BarGraph	Represents an analog value in a bar
Button	Represents a control that can be clicked. Provides a simple way to trigger an event, and is commonly used to execute commands. It is labeled either with text or an image.
CheckBox	Allows multiple selections from a number of options. They are displayed as a square box with white space (for unselected) or as a tick mark (for selected).
ComboBox	Represents a control that enables to select items from a list Combination of a drop-down list and a textbox. It allows you to either type a value directly into the control or choose from the list of existing options.
CommandBar	Provides a menu system for a ScreenForm
ControllerModeStatus	Displays the mode of the Controller (Auto - Manual)
DataEditor	Represents a text box control that can be used to edit the data.
Graph	Represents a control that plots data with lines or bars.
GroupBox	Represents a Windows control that displays a frame around a group of controls with an optional caption. Is a container used to group a set of graphic components. It usually has a title at the top.

Continues on next page

1 Introduction

1.4. Development environment

Continued

Control	Description
LED	Displays a two states value, like a Digital Signal.
ListBox	Represents a control to display a list of items. Allows the user to select one or more items from a list contained within a static, multiple line text box.
NumEditor	Represents a text box control that can be used to edit a number. When the user clicks it, a Numpad is opened.
NumericUpDown	Represents a spin box that displays numeric values.
Panel	Used to group collection of controls.
PictureBox	Represents a picture box control that displays images.
RadioButton	Allows to select only one of a predefined set of options.
RapidExecutionStatus	Displays the execution status of the Controller Rapid Domain (Running - Auto)
RunRoutineButton	Represents a Windows button control that calls a RapidRoutine when clicked
Switch	Displays and lets change a two states value, like a Digital Output Signal.
TabControl	Manages a set of tab pages.
TpsLabel	Very commonly used widget that displays text, a label is usually static, that is, it has nointeractivity. A label generally identifies a nearby text box or other graphic component.

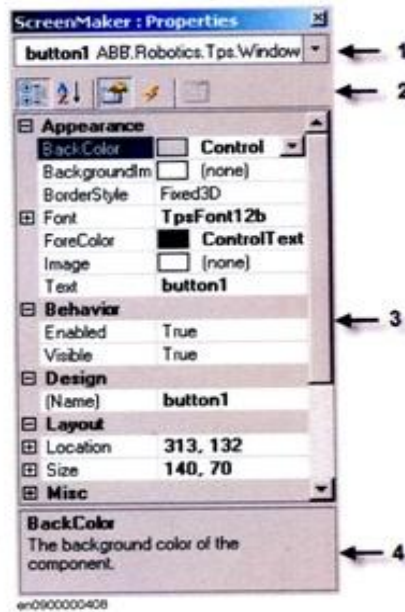


NOTE!


For more information on using these controls and their properties, see the section *Building the user interface on page 18* and the chapter *Using the FlexPendant SDK of the Application manual - Robot Application Builder*.

Properties window

A control is characterized by its properties and events. Properties describe the appearance and behavior of the component, while events describe the ways in which a control notifies its internal state change to others. By changing the value of a property, the controls have a different look and feel, or exhibit different behavior.



en0900000408

Element	Description
1	Graphical component name panel Displays the selected component, and lists the available components of the on the active design screen.
2	Properties window toolbar  <p>en0900000409</p> <ol style="list-style-type: none"> 1. Organizes table panel in categories 2. Organizes table panel alphabetically 3. Displays Properties in table panel 4. Displays Events in table panel
3	Table panel Displays all the properties or events in two-columns. The first column shows the property or event name, the second shows the value of the property or name of the event handler.
4	Information panel Display information about a property or event.

© Copyright 2009 ABB. All rights reserved.

Continues on next page

1 Introduction

1.4. Development environment

Continued

Editing the property value

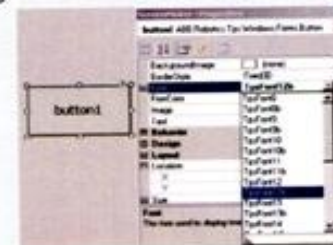
You can edit the property value of a control from the **Properties** window in three ways:

- 1 By typing the numerics, strings and text.
For example, Location, Size, Name etc.



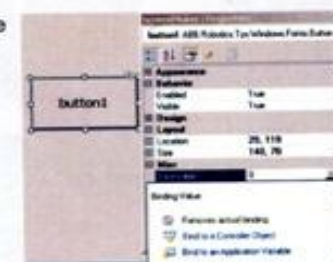
en0900000631

- 2 By selecting the predefined values from the list. For example, BackColor, Font etc.



en0900000633

- 3 By entering the values in the dialog box. For example, Enabled, States, BaseValue etc.



en0900000632

Building the user interface

This section describes building the GUIs using the following controls from the **ToolBox**.

ActionTrigger

An action trigger initiates an event, such as making a hidden object visible when an action is performed using a control. It allows to run a list of actions when the property value changes. The property value can be bound to a signal, rapid data, or application variable.

ActionTrigger control can also be used to invoke the application from RAPID.

Use this procedure to add an ActionTrigger control::

Action

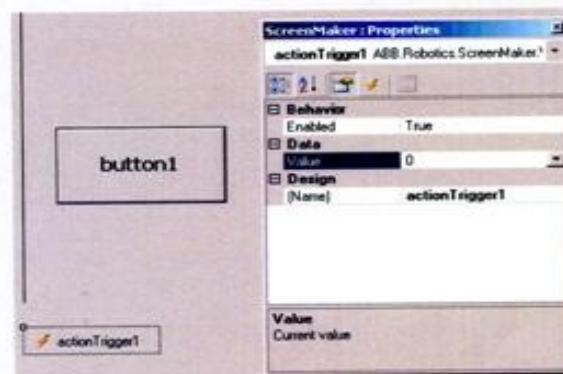
- 1 Drag an **ActionTrigger** control from the **ToolBox** on to the design surface.

Continues on next page

Action

2 You can modify the name, set the default value and configure data binding value for a ActionTrigger control.

- To set the values of a property, see [Properties window on page 17](#).
- You can set the trigger event for an ActionTrigger to any of the event handler created either from a control or from an Events Manager option. To set up the events, see [Setup Events on page 35](#).
- To configure the data binding values, see [Configuring data binding on page 38](#).
- To set the application variables, see [Managing application variables on page 34](#).



en0900000629



en0900000630

NOTE: An action is not triggered when the screen is launched for the first time, but is triggered when there is a difference in the binded value at any point of time. This functionality is supported only in RobotWare 5.12.02.

Example: Consider a signal being binded to the value property. The value of the signal changes at runtime on performing a specific action. The event handler configured for ActionTrigger control gets triggered based on this value change.

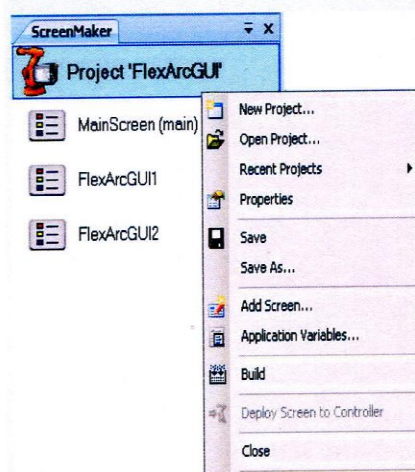
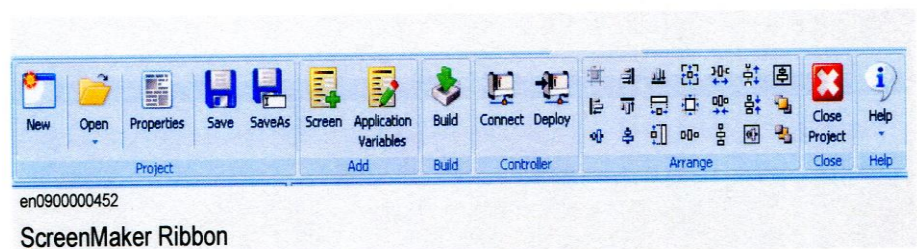
2 Managing ScreenMaker projects

2.2. Managing ScreenMaker projects

2.2. Managing ScreenMaker projects

Overview

You can manage a project (create, delete, load, or save) either from ScreenMaker ribbon or Context menu.



2 Managing ScreenMaker projects

2.2. Managing ScreenMaker projects

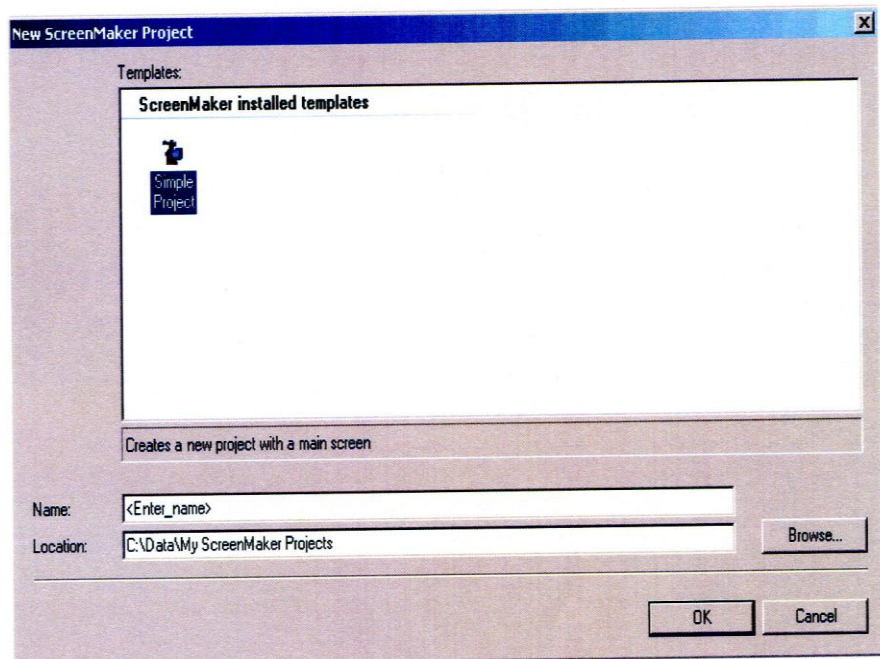
Continued

Creating a new project

Use this procedure to create a new project:

1. Click **New** from the ScreenMaker ribbon or right-click **Project** context menu and select **New Project**.

The **New ScreenMaker Project** dialog box appears.



2. Enter a new project name and specify a location for the new project.

A default screen *MainScreen (main)* is added in the tree view.

By default, the new project is saved on *C:\My Documents\My ScreenMaker Projects*.

3. Click **OK**.

2 Managing ScreenMaker projects

2.2. Managing ScreenMaker projects

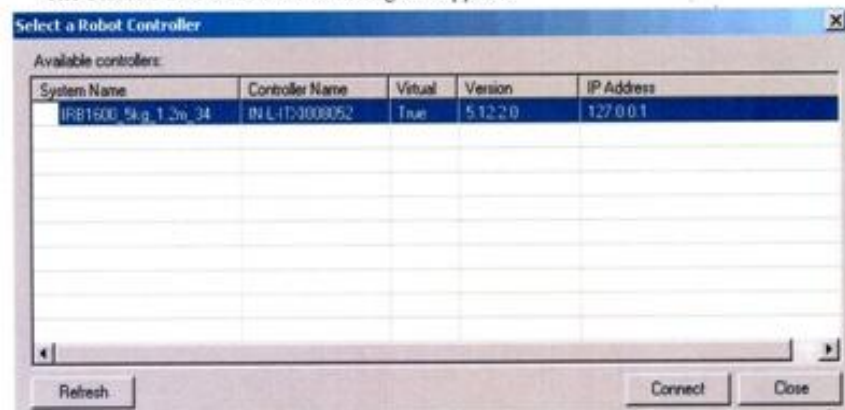
Continued

Connecting to controller

Use this procedure to establish the connection with a controller:

1. From the ScreenMaker ribbon, click **Connect**.

The **Select a Robot Controller** dialog box appears.



2. Click **Refresh** to find a list of all the available controllers.

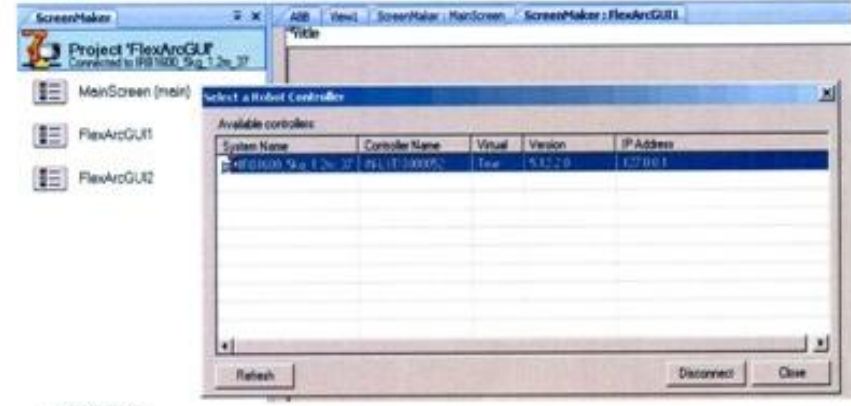


NOTE!

- By default, the currently connected controller is highlighted and has a small icon before the row as an indicator.

3. Select the controller to be connected from the list and click **Connect**.

The connection status is displayed in the Project tree view.



2 Managing ScreenMaker projects

2.2. Managing ScreenMaker projects

Continued

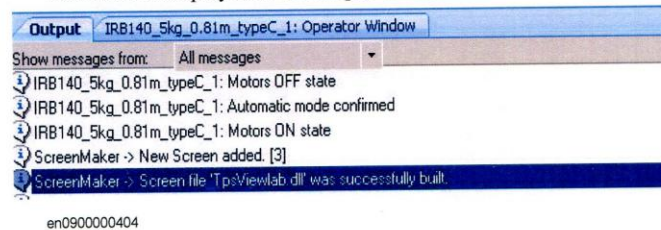
Building a project

The result from building the ScreenMaker project is a set of files including DLL file and images. The ScreenMaker project can be compiled into binary format (.dll) that can be deployed on a FlexPendant.

Use this procedure to build a project:

1. From the ScreenMaker ribbon, click **Build** and select **Build**.

The result is displayed in the output window.



Deploying to controller

Use this procedure to deploy a ScreenMaker project on a real controller or virtual controller:

1. Connect to the controller you want to deploy. See [Connecting to controller on page 32](#).
2. From the ScreenMaker ribbon, click **Deploy**.

The Download dialog box appears displaying the progress of download. It disappears once the download is successful.

The **TpsViewxxxxxx.dll** file is downloaded.

3. Restart the controller.



NOTE!

- If a real controller is used, you can reboot the FlexPendant by moving its joystick three times to the right, once to the left, and once towards you.
- If a virtual controller is used, you can reboot the FlexPendant by closing the virtual FlexPendant window.

3 Tutorial

3.4. Building and deploying the project

3.4. Building and deploying the project

Procedure

1. From the **ScreenMaker** ribbon, click **Build**.

For more information on building the project, see *Building a project on page 33*.

2. From the **ScreenMaker** ribbon, click **Deploy**.

For more information on deploying the project, see *Deploying to controller on page 33*.

3. In RobotStudio, press **Ctrl+F5** to launch the Virtual Flexpendant and click FlexArc Operator Panel to open the GUI.



NOTE!

Ensure that you start the RAPID execution and switch the controller into Auto mode.