Faculty of Engineering of the University of Porto

Visual Components: Programming a manipulator To grasp and release multiple objects

INDUSTRIAL ROBOTICS

Master in Electrical and Computer Engineering

Sandro Magalhães
António Paulo Moreira

2nd Semester of 2019/2020

1. Introduction

The Visual Components is a 3D simulation software for manufacturing. The simulations are usually created using different features from the software. However, the most common are Process Modelling and Works Library. Both can be involved in factory simulation, defining interactions between robots, mobile robots, humans, and the rest of the components.
Besides these capabilities of Visual Components, which allows a quick simulation of the manufacturing and the behaviour between artefacts, the Visual Components also allows the direct programming of robots. The software uses a graphical interface that interacts with the robot and with other components and establishes the controlling signals.
This laboratory work introduces to this programming environment to program manipulators. For this work, it is intended to create a manufacturing cell with two conveyors and an articulated industrial manipulator. The manipulator should pick the objects from one of the conveyors and place them inside the boxes in the other conveyor.
This practical work was performed using Visual Components 4.2 Premium but should work with other versions of Visual Components.

2. Configuring the workspace

Start by placing all the components in the Visual Components workspace. For that, begin with an empty workspace **File** > **Clear All** (Ctrl+N). Now, it is needed two feeders, one to provide the pieces and another one to provide the boxes where we will place the pieces. Therefore, in the left panel, select **eCatalog** > **Models by Type** > **Feeders** and add a **Basic Feeder** and a **Shape Feeder**. It is also needed two conveyors to move the objects when the feeders provide them. So, select now **eCatalog** > **Models by Type** > **Conveyors** and add a **Sensor Conveyor** and a **Product Batcher Conveyor.**

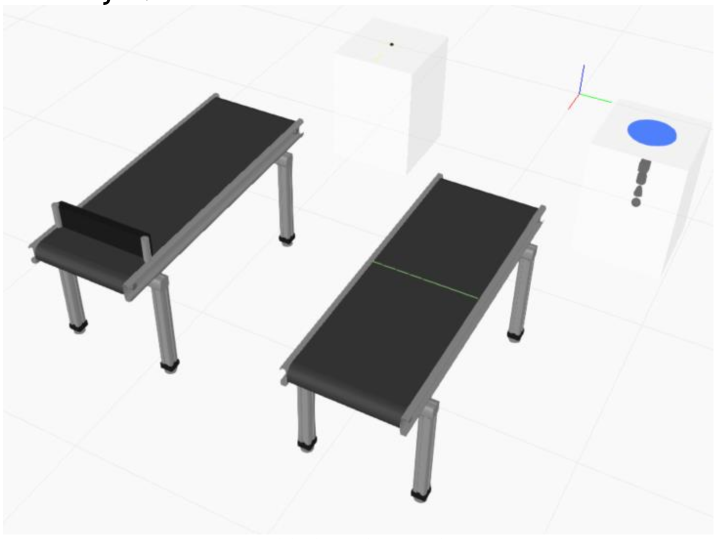
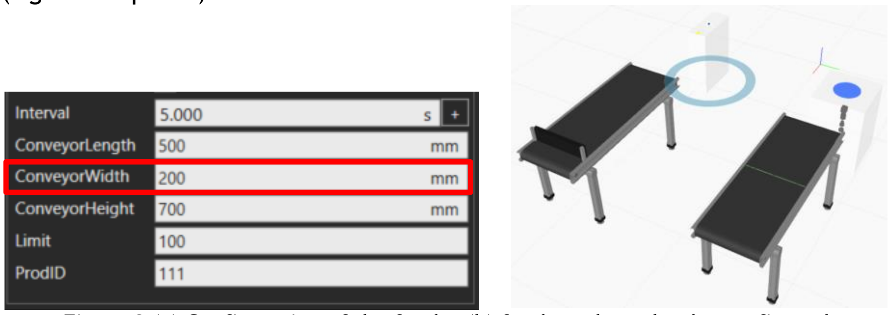


Figure 1 Added feeders and conveyors

Select the **Basic Feeder** and change the **ConveyorWidth** to 200 mm in **Components Properties** > **Default** (right side panel).



Interval	5.000	s
ConveyorLength	500	mm
ConveyorWidth	200	mm
ConveyorHeight	700	mm
Limit	100	
ProdID	111	

Figure 2 (a) Configuration of the feeder (b) feeder selected to be configured

For allowing the products to move from the feeders to the conveyors, attach the conveyors to the feeders. When the conveyor attaches to the feeder, it resizes its width to fit the feeder's **ConveyorWidth**. Select the tab **Home** > **PnP** to select the Plug and Play attachment. Now, Select **Product Batcher Conveyor** in the workspace and attach it to the **Basic Feeder.** Proceed similarly between the **Sensor Conveyor** and the **Shape Feeder.** If the attachment was successful, the yellow arrow should become green. If needed, use the blue circle, which appears when an object is selected, to rotate the objects.

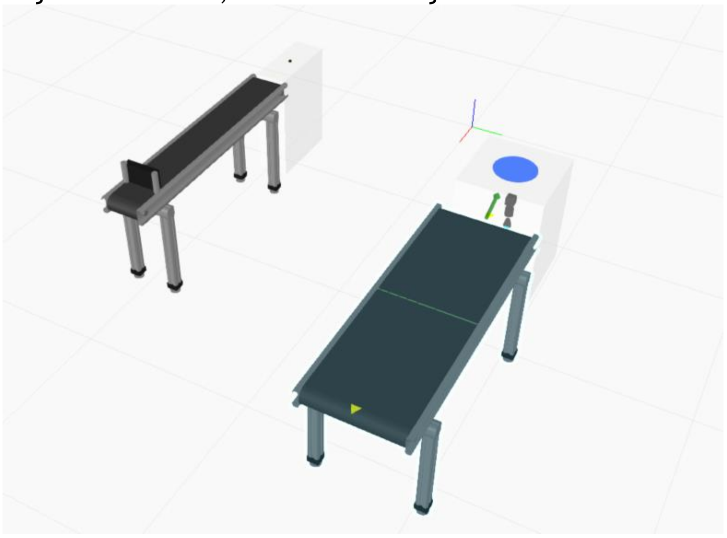
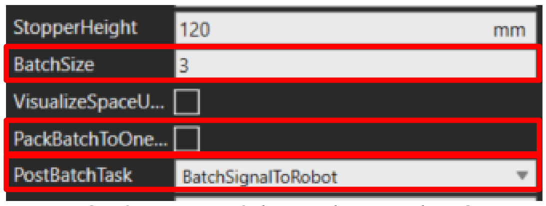


Figure 3 Attaching conveyors to the feeders, using PnP tool

To allow the robot to communicate with the **Product Batcher Conveyor** and grasp the multiple objects, we need to adjust some properties. So, select it, and, in the **Component Properties** > **Default**:

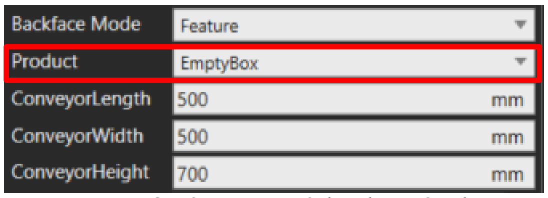
- Change **BatchSize** to 3 (to provide only three objects at a time)
- Unselect **PackBatchtoOneComponent** (to avoid objects aggregation into a single object)
- Change **PostBatchTask** to **BatchSignalToRobot (**to send a trigger to the robot when the conveyor has three objects**)**



StopperHeight	120 mm
BatchSize	3
VisualizeSpaceU...	☐
PackBatchToOne...	☐
PostBatchTask	BatchSignalToRobot

Figure 4 Configuration of the Product Batcher Conveyor

Select, now, the **Shape Feeder** and, in **Component Properties** > **Default** change the **Product** to **EmptyBox.** Now the feeder should provide empty boxes where we will place the objects provided by the **Basic Feeder**.



Backface Mode	Feature
Product	EmptyBox
ConveyorLength	500 mm
ConveyorWidth	500 mm
ConveyorHeight	700 mm

Figure 5 Configuration of the shape feeder

Run the simulation (Figure 6) and check if the objects are moving from the feeders to the conveyors. The **Product Batch Conveyor** should only keep three objects. Reset the simulation.

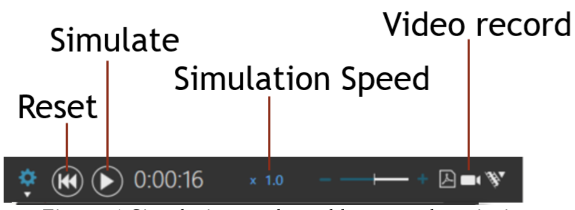


Figure 6 Simulation tools and buttons description

Now, it is only missing the robot which will pick and place the objects. So, add the **Generic Articulated Robot v4** from **eCatalog** > **Models by Type** > **Robots** > **Visual Components.** Place it in the middle of two conveyors. Check whether the conveyors are inside the robot workspace. For that, select the robot and activate the **Envelope** option in **Component Properties** > **Workspace.** If the conveyors are inside the grey bubble, everything is right. Unselect the **Envelope** option.

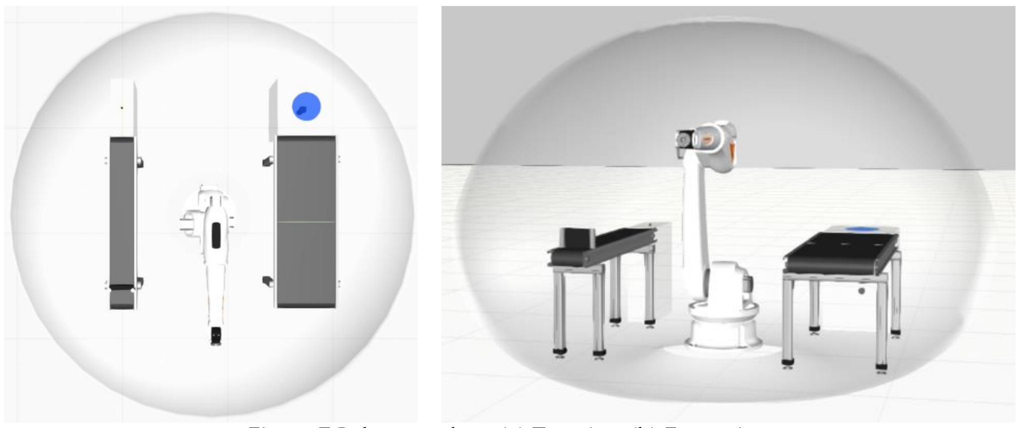


Figure 7 Robot envelope (a) Top view (b) Front view

The robot also needs a gripper to pick the objects. In this case, we will opt by a Suction Gripper. Go to **eCatalog** > **Models by Type** > **Machine Tending Library** > **Visual Components** and add the **Parametric Gripper** to the workspace. Using the **Home** > **PnP** tool, attach the gripper to the robotic manipulator.

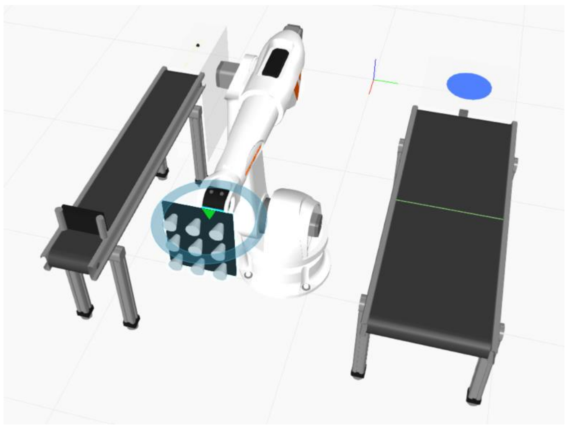
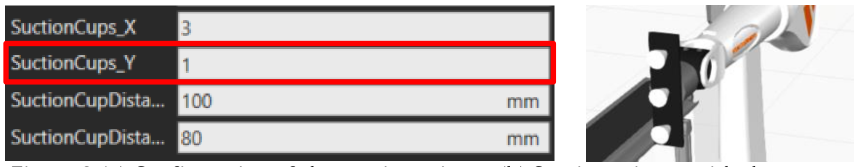


Figure 8 Plugged suction gripper to the robot

As defined previously, the **Product Batch Conveyor** only keeps three products on the top. So, keeping the **Parametric Gripper** selected, change the **SuctionCups_Y** to 1 in the **Component Properties** panel.



SuctionCups_X	3	
SuctionCups_Y	1	
SuctionCupDista...	100	mm
SuctionCupDista...	80	mm

Figure 9 (a) Configuration of the suction gripper (b) Suction gripper with three cups

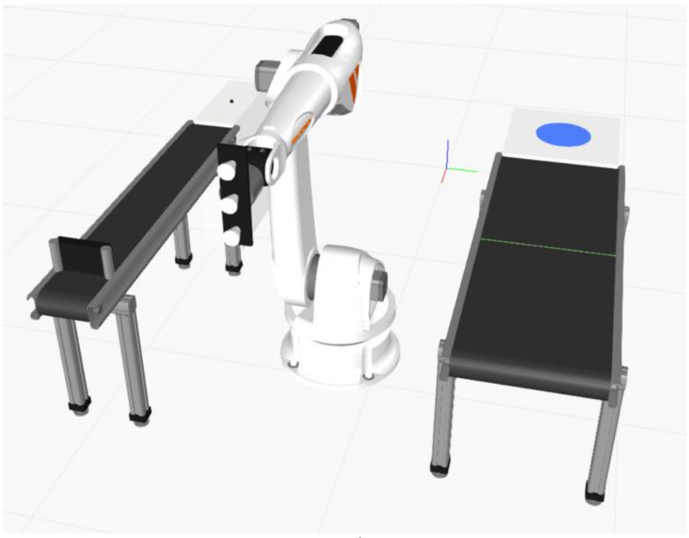
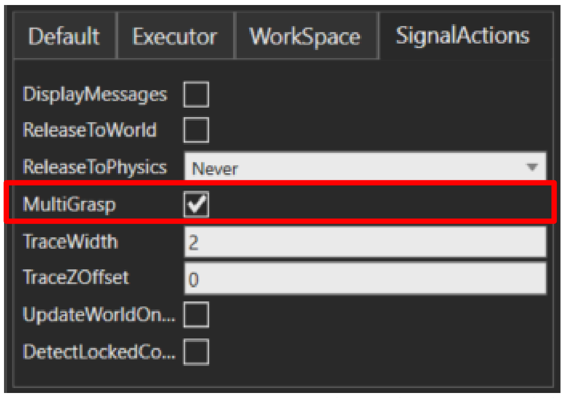


Figure 10 Final scene view

3. Pick and place multiple objects

Still, on the **Home** tab, select the robot and activate the **MultiGrasp** checkbox in **ComponentProperties** > **SignalActions**, to allow the gripper to pick multiple objects.



DisplayMessages	☐
ReleaseToWorld	☐
ReleaseToPhysics	Never
MultiGrasp	✓
TraceWidth	2
TraceZOffset	0
UpdateWorldOn...	☐
DetectLockedCo...	☐

Figure 11 Configuration to activate the multi-grasp capabilities

Now the whole of the simulation environment is configured. So, begin by programming the robot to pick the three objects from the batch conveyor. The programming of the robot is done in the **Program** tab.

Now, it is time to start programming the robot. So, move to the **Program** tab and, using the **Program** > **Jog** tool, select the robot. After, select **Program** > **Signals** (in the **Connect** group). Link the signals from the conveyors to the robot and reassign them. In this case, the signals from the Batch conveyor were registered in 50, and the signals from the sensor conveyor in 51. The StartStop signals are outputs of the robot.

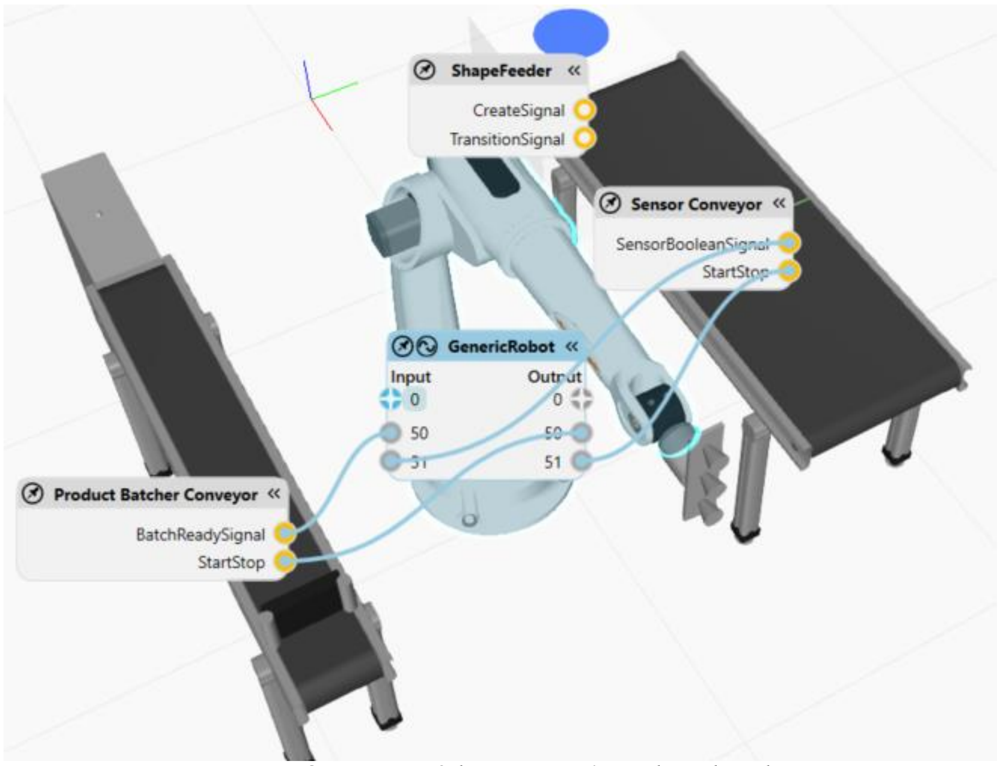
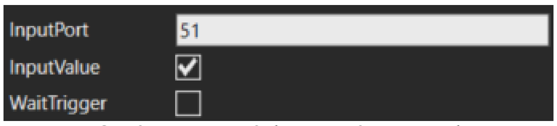


Figure 12 Connection of the conveyors' signals to the robot

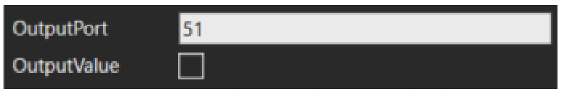
Now, check whether the signals are working right. For that, program the actions for the signals, beginning by instructing the conveyor to stop when it receives a box. In the **Program Editor** panel on the left side, add a **Wait For Binary Input Statement** instruction. In the **Statement Properties** panel (right side), change the **InputPort** to 51 and select the **InputValue.** The **InputValue** flag signalises the program to wait for a True event.



InputPort	51
InputValue	✓
WaitTrigger	☐

Figure 13 Configuration of the wait for a new box sensor

Once the robot receives a True signal from the conveyor, the conveyor should stop. Add a **Set Binary Output Statement** and change the **OutputPort** to the port 51.

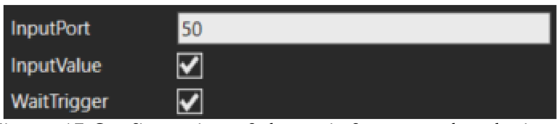


OutputPort	51
OutputValue	☐

Figure 14 Configuration of the signal to stop the boxes' conveyor

Now, run the simulation and check whether the sensor conveyor stops when it receives a box. Reset the simulation.

Now, proceed similarly to the objects in the batch conveyor. Add a **Wait For Binary Input Statement** instruction and change the **InputPort** to 50, select **InputValue**, and select **WaitTrigger**. The **WaitTrigger's** signal allows for detecting whether there was some change in the **inputPort** 50. In this case, checks whether there is a new batch in the conveyor, or the signal True, corresponds to the old batch.

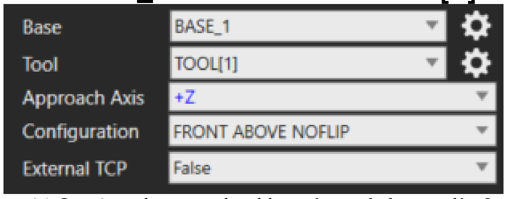


InputPort	50
InputValue	✓
WaitTrigger	✓

Figure 15 Configuration of the wait for a new batch signal

For checking whether it is working or not, add a **Halt** instruction **.** Run the simulation. It should stop when the batch conveyor has three objects ready to pick.

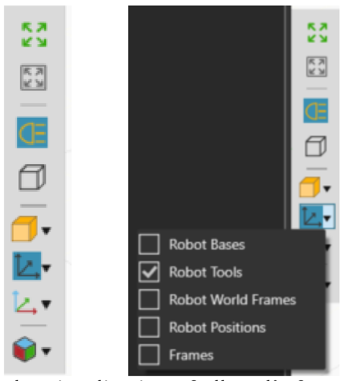
To grasp the objects, we need to configure the frames and the tool of the robot. It is possible to manually configure the **DoubleTool1** (frame of the gripper - **SuctionGripper**’s frame) as a grasp and release frame (attribute to this frame the properties of grasp and release objects). However, the Visual Components has already configured the frame TOOL[1] for this purpose. So, it is more useful to use this frame instead, but first, the **TOOL[1]** should be coincident with the frame DoubleTool1. Using the **Program** > **Jog** function, select the robot. In the **Jog** panel (right side), change the **Base** to **BASE_1** and the **Tool** to **TOOL[1].**



Base	BASE_1
Tool	TOOL[1]
Approach Axis	+Z
Configuration	FRONT ABOVE NOFLIP
External TCP	False

Figure 16 Setting the standard base’s and the tool’s frames

Click now in the Gear icon after the **Tool** option to change the **TOOL[1]** frame position. In the workspace, activate the **Frame Types** and check whether the **Robot Tools** are visible.



Robot Bases
Robot Tools ✓
Robot World Frames
Robot Positions
Frames

Figure 17 Enables the visualisation of all tool’s frames in the 3D world

Now, select the robot again and click on **Program** > **Snap**. In the **TCP Snap** panel, select **Frame** in the **Snap Type** area.

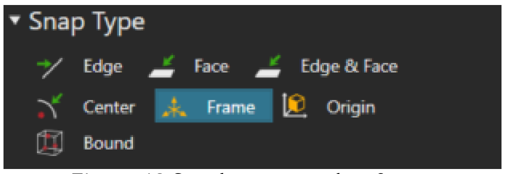


Figure 18 Set the snap tool to frame

Now click on the **DoubleTool1** frame in the 3D World to move the frame **TOOL[1]** to the frame **DoubleTool1**. Thereby, the suction tool gets the properties to grasp objects from **TOOL[1]**.

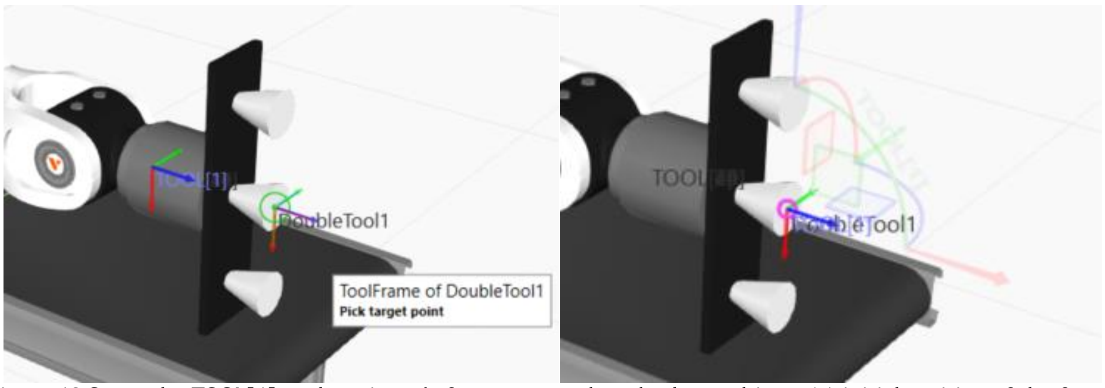


Figure 19 Snaps the TOOL[1] to the gripper's frame to attach and release objects (a) initial position of the frame TOOL[1] and selecting DoubleTool1 (b) TOOL[1] snapped to DoubleTool1

Select the robot again using the **Jog** tool and change the **Tool** to the **DoubleTool1**.

Simulate until it stops or fills the conveyor with three objects. **Do not reset the simulation** this time. Now, teach the robot to grasp the batch of objects. These steps may be supported by the following video, which explains visually the strategy to teach the robot to pick objects: https://educast.fccn.pt/vod/clips/1s28ovdaq5/streaming.html

Select the robot with the **Program > Jog** tool. Using the **Program** > **Snap** tool, select the **Snap Type** to **Edge & Face**.

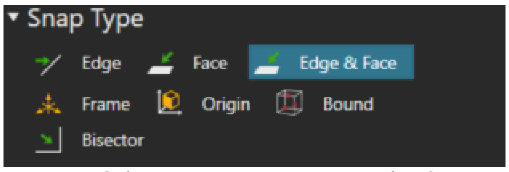
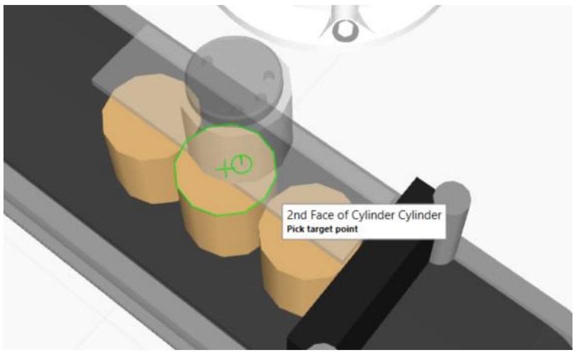


Figure 20 Selecting snapping type to Edge & Frame

Now match the centre of the tool with the centre of the object's batch (Figure 21). Start by approximating the mouse to the top of the cylinder and should appear a green circle around the upper face, and a green cross in the centre of the upper face (Figure 21a). Then, place the mouse in the green cross until it changes to a blue crossed circle (Figure 21b). Click in the left button of the mouse to teach the position (Figure 21c).



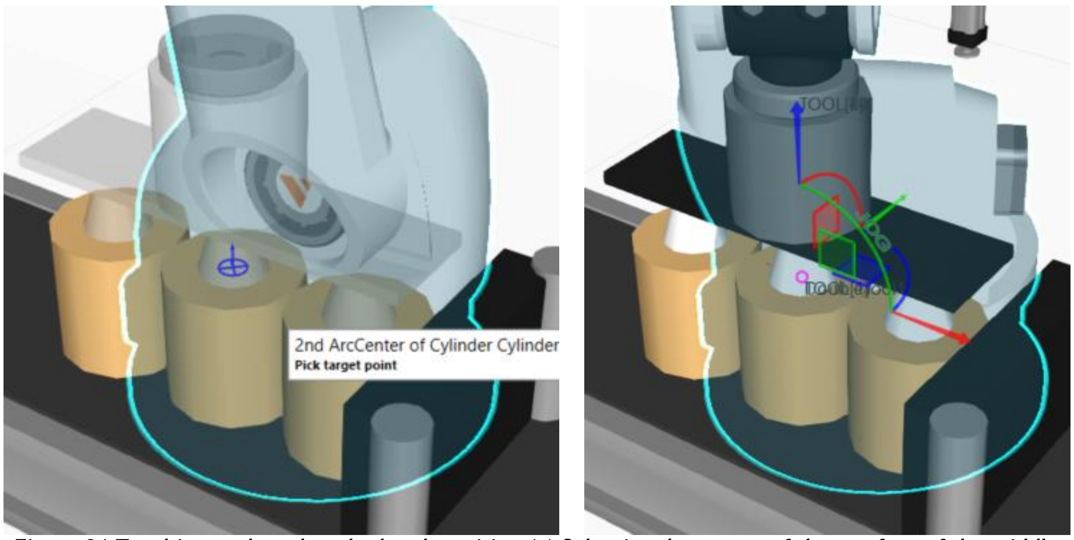
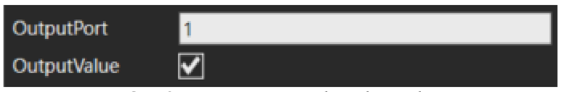


Figure 21 Teaching to the robot the batch position (a) Selecting the center of the top face of the middle cylinder (b) Selected the middle-top of the centre cylinder of the batch - blue crossed circle (c) robot learnt the batch position

To pick the objects, create a new subprogram in the **Program Editor**, and rename it to PickBatch. In this new subprogram, add a **Linear Motion Statement** and **a Set Binary Output Statement** . In **Statement Properties**, change the **OutputPort** to 1 and select **OutputValue.** This will **instruct the robot to move to the objects and grasp them.**



OutputPort 1
OutputValue ✓

Figure 22 Configuration to order the robot to grasp

Select the Linear Motion Instruction **LIN P1 DoubleTool1 Base_1 2500mm/s**, and in the right-side panel, select the **Jog** panel.

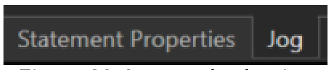
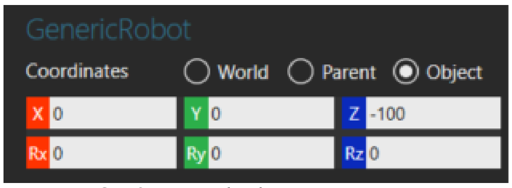


Figure 23 Jog panel selection

It is intended to create a linear approximation area for the objects. So, change the **Coordinates** reference to Object and the Z-axis to -100.



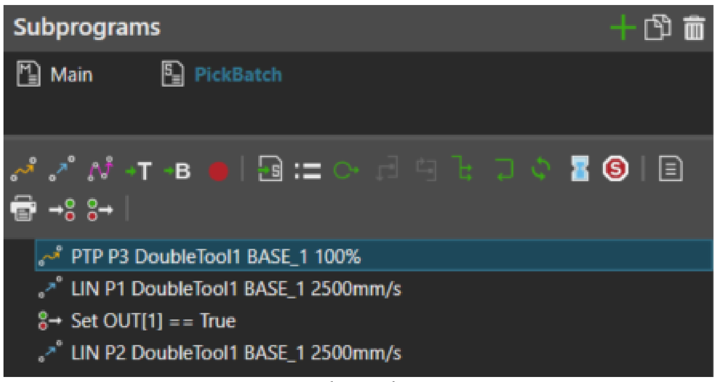
GenericRobot

Coordinates ( ) World ( ) Parent (●) Object

X 0	Y 0	Z -100
Rx 0	Ry 0	Rz 0

Figure 24 Configuring the linear approximation area

Add another **Linear Motion Statement** and a **Point-to-Point Motion Statement** . Move the new **Point-to-Point Motion Statement** as the first statement.


Subprograms
Main
PickBatch

PTP P3 DoubleTool1 BASE_1 100%
LIN P1 DoubleTool1 BASE_1 2500mm/s
Set OUT[1] == True
LIN P2 DoubleTool1 BASE_1 2500mm/s

Figure 25 Pick Batch statements

Now, in the **Main** subprogram, call the **PickBatch** subprogram, before the **Halt** instruction. For that, add a **Call Sequence Statement** and change the **Routine** to **PickBatch** in **Statement Properties.**

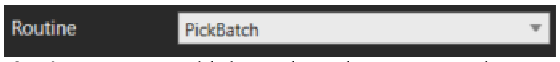
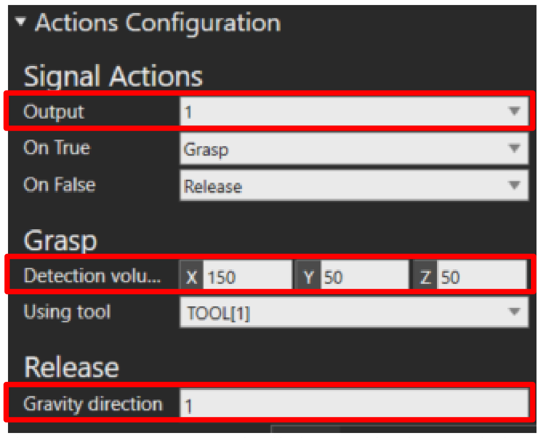

Routine	PickBatch

Figure 26 Configuration to add the PickBatch Routine to the main program

For ensuring that the gripper grasps the three objects, it is needed to expand the grasping area. So, select the robot again and expand **Actions Configurations** in the **Components Properties** panel. In **Signal Action,** change the **Output** to 1 and the **DetectionVolume** to **x** = 150.

One of the particularities of the Visual Components is that it uses the gravity and collisions effect to attach the objects. So, when the manipulator released the batch, it moved in the Z-axis and attached it with the collision that it found. Once the base of the batch was aligned with the box, it is needed to change the **Gravity direction** to 1, in the same panel.


Actions Configuration

Signal Actions

Output	1
On True	Grasp
On False	Release

Grasp

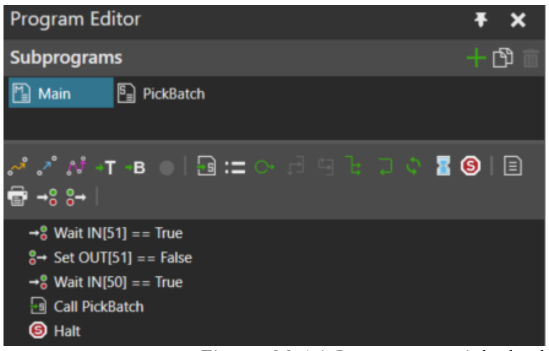
Detection volu...	X 150 Y 50 Z 50
Using tool	TOOL[1]

Release

Gravity direction	1

Figure 27 Expands the grasping area to pick all three cylinders and change the gravity size

Now, reset and run the simulation, and check whether the robot grasps the three objects.


Program Editor
Subprograms
Main PickBatch

```
Wait IN[51] == True
Set OUT[51] == False
Wait IN[50] == True
Call PickBatch
Halt
```

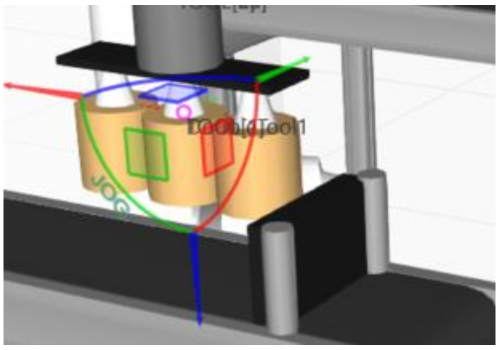


Figure 28 (a) Program to pick the batch (b) Robot picking the batch

Once the robot picks objects from the batch conveyor, now it should place them inside the box. So, create a new subprogram and call it **PlaceBatch.** The procedure to place the batch inside the box may also be followed in the following video:
https://educast.fccn.pt/vod/clips/11646wor1a/streaming.html

To teach the robot place the batch inside the box, we proceed similarly to the example to pick the batch. Select the **Program** > **Align** tool and change the **Snap Type** to **Origin.** Through this strategy, it is intended to align the origin of the central cylinder with the origin of the box.

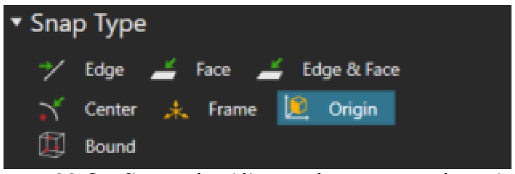


Figure 29 Configure the Align tool to snap to the origin

Therefore, select the batch central cylinder (Figure 30a) and then the empty box aligned with the conveyor's sensor (Figure 30b). Finally, the robot should jog to the empty box (Figure 30c).

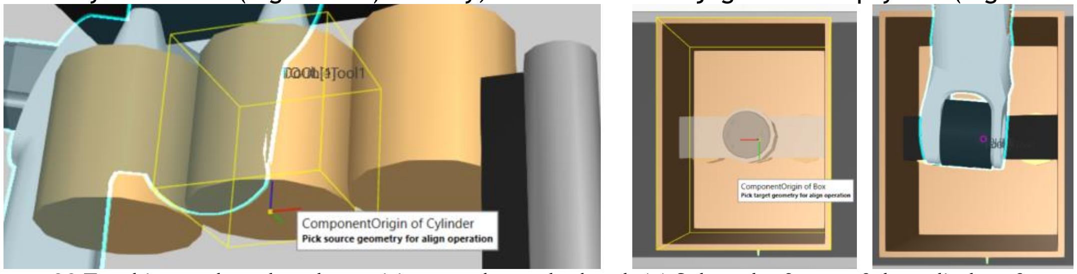

Tool[1]Tool1
ComponentOrigin of Cylinder
Pick source geometry for align operation
ComponentOrigin of Box
Pick target geometry for align operation

Figure 30 Teaching to the robot the position to release the batch (a) Select the frame of the cylinder of centre of the batch (b) Teaching the robot the centre of the box (c) Robot learnt the position where to place the batch

The frame of the box is placed in its base, so we need to fix some collisions. Change to the **Jog** panel, and change the **Coordinates** to **Object**. Here make **Z** $= -\mathbf{10}$ to avoid collisions with the base of the box, and **Rz** $= \mathbf{90}$ to avoid collision between the gripper and the laterals of the box. Then, make **Y** $= -\mathbf{75}$ to free space to add another batch.

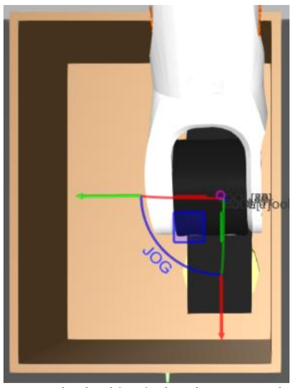
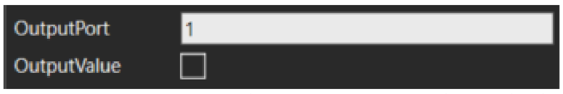


Figure 31 Using only half of the box to place the batch

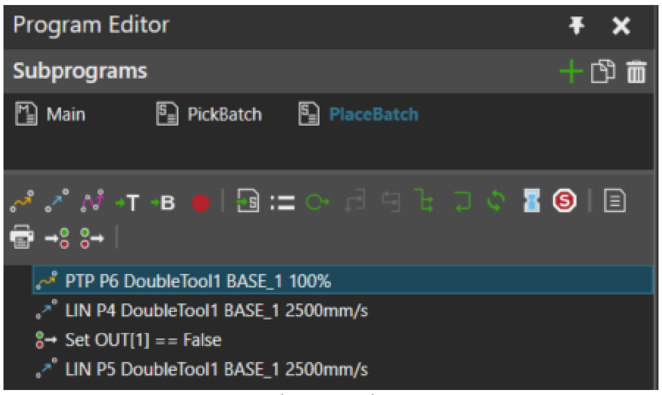
For releasing the objects, implement a routine similar to what we implemented for picking the objects. In the subprogram **PlaceBatch,** add a **Linear Motion Statement** and a **Set Binary Output Statement** . In **Statement Properties,** change the **OutputPort** to 1 and keep the **OutputValue** unselected to release the object's batch.



OutputPort	1
OutputValue	☐

Figure 32 Configuration of the instruction to release the batch

Change to the **jog** panel again and change the **Coordinates to Object.** Move the **Z**-axis −350. Add to the subprogram **PlacBatch** a new **Linear Motion Statement** and a **Point-to-Point Motion Statement** . Set the **Point-to-Point Motion Statement** as the first statement of the subprogram.

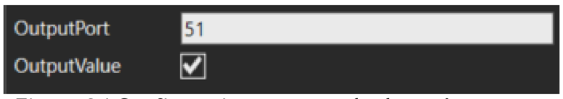


PTP P6 DoubleTool1 BASE_1 100%
LIN P4 DoubleTool1 BASE_1 2500mm/s
Set OUT[1] == False
LIN P5 DoubleTool1 BASE_1 2500mm/s

Figure 33 Place Batch statements

Finally, in the main subprogram, add a **Call Sequence Statement** , after the **Call PickBatch**, and change the **Routine** to **PlaceBatch.**
Run the simulation, and the robot should pick a batch and place it inside the box.

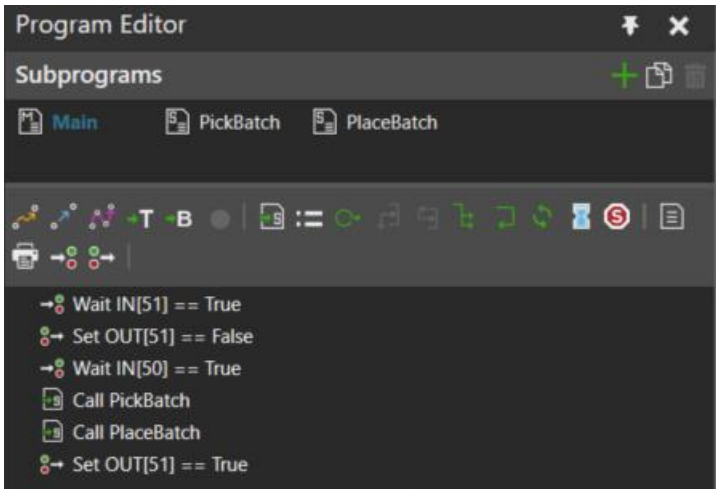
Now it is only needed to order the program to move the conveyor sensor to release the box. For that, after the **Call PlaceBatch,** add a new **Set Binary Output Statement** . In **Statement Properties**, set the **OutputPort** to 51 and select **OutputValue**. Remove the **Halt** instruction.



OutputPort	51
OutputValue	✓

Figure 34 Configuration to move the boxes' conveyor

Run the simulation and check that the robot picks the batch and place it inside the box after the conveyor releases the box. The result similar to this one: https://educast.fccn.pt/vod/clips/yfwa1puvj/streaming.html



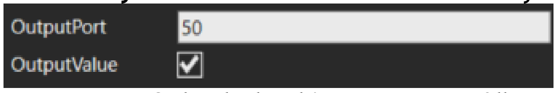
```
Wait IN[51] == True
Set OUT[51] == False
Wait IN[50] == True
Call PickBatch
Call PlaceBatch
Set OUT[51] == True
```

Figure 35 Statements of the main routine

4. Placing multiple batches of objects inside the box

Now, it is intended to use the missing space of the box to place another batch. For that, instead of creating a new subprogram and teach the robot again, reuse the same subprogram and move the reference frame.

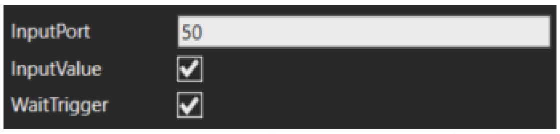
Begin by refilling the **Product Batch Conveyor**, adding a new **Set Binary Output Statement** after the **Call PlaceBatch** instruction. Set the **OutputPort** to 50 and select the **OutputValue** to order the **Product Batcher Conveyor** to move and refill with new cylinders.



OutputPort	50
OutputValue	✓

Figure 36 Order the batch's conveyor to refill

Next, add a **Wait For Binary Input Statement** instruction and change the **InputPort** to 50 and select **InputValue** and **WaitTrigger**. So, when the conveyor had new three pieces, the robot will receive an alert. The **WaitTrigger** signal avoids false positives and only sends a True signal when any changes happen.

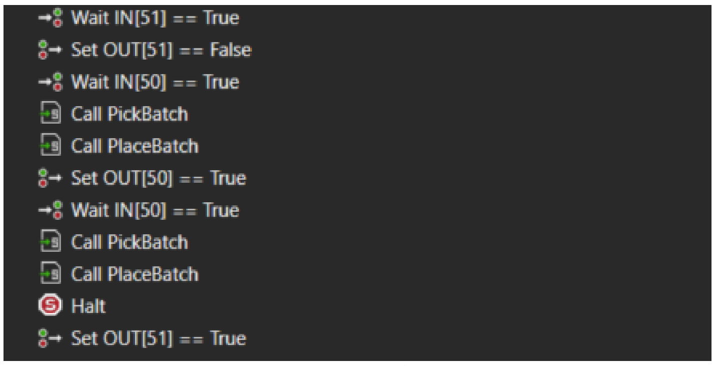


InputPort	50
InputValue	✓
WaitTrigger	✓

Figure 37 Configuration of the wait for a new batch signal

Finally, add two **Call Sequence Statement** , and change the **Routine. The first** one to **Call PickBatch** subprogram and the other to **Call PlaceBatch**.

To check the program, add a **Halt** instruction before moving the **Sensor Conveyor**.



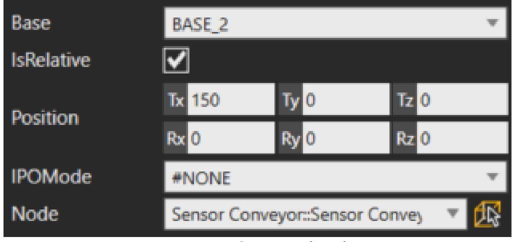
```
Wait IN[51] == True
Set OUT[51] == False
Wait IN[50] == True
Call PickBatch
Call PlaceBatch
Set OUT[50] == True
Wait IN[50] == True
Call PickBatch
Call PlaceBatch
Halt
Set OUT[51] == True
```

Figure 38 Statements of the main program

Run the simulation and check that the robot places the two batches in the same place. So, now, try to move the placing frame for the second time we place the batch.

For that, before the second **Call PlaceBatch**, add a **Define Base Statement** . In the **Statement Properties Panel**:

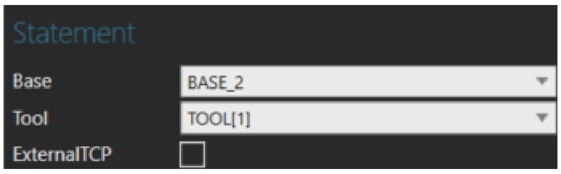
- Change **Base** to **BASE_2**
- Select **IsRelative**
- Set **Tx** to 150 (this may vary according to the orientation of frame BASE_2; move instead in the negative direction)
- Change **Node** to **Sensor Conveyor::Sensor Conveyor** (or use the **Pick property from 3D world** to select the **Sensor Conveyor** in the workspace)



Base	BASE_2		
IsRelative	✓		
Position	Tx 150	Ty 0	Tz 0
	Rx 0	Ry 0	Rz 0
IPOMode	#NONE		
Node	Sensor Conveyor::Sensor Convey		

Figure 39 Base frame displacement

Inside the subprogram **PlaceBatch**, in all move instructions, change the **Base** to **BASE_2.**



Statement

Base	BASE_2
Tool	TOOL[1]
ExternalTCP	☐

Figure 40 Change the reference frame of the PlaceBatch's instructions

Finally, remove the **Halt** instruction and run the simulation. Check that everything runs right. The final result should be similar:

https://educast.fccn.pt/vod/clips/2he8lg0ycc/streaming.html

5. Looping the program

Now, using the knowledge acquired during this tutorial, and exploring the program capabilities, change the main sequence to loop the program for more than one box. Avoid also to duplicate the call to **PickBatch** and **PlaceBatch**. Try to get a similar behaviour of the robot: https://educast.fccn.pt/vod/clips/1uef6nd9uu/streaming.html

Suggestion: Use **Assign Variable Statement** to count the loops. It is possible to define the variables in **Routine Properties** of the subprogram. At the end of the second loop, reset the counter and BASE_2 position.

Produce a small report, explaining the final algorithm. Create also a video with the final result and attach a link to it in the report. Submit the report and the program file.