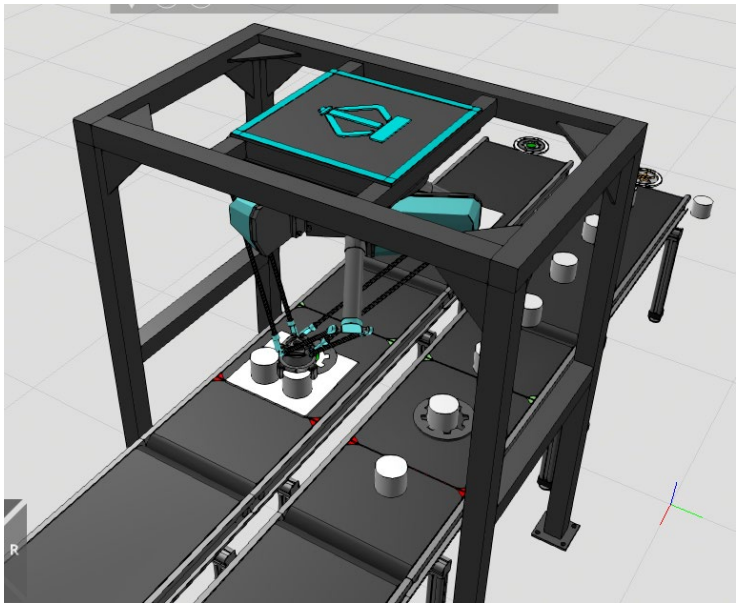
Pick And Place Components - Manual

Visual Components 4.6 | Version: February 15, 2023



This is a manual for *Pick And Place Components* in the Visual Components eCatalog. Components are used for pick and place applications where products are moving on conveyors.

The Library consists of two components:

- *Pick And Place Process* component, is a conveyor which includes the process for either picking or placing.
- *Pick And Place Transport Controller* is the pedestal component where you attach your robot arm, which in many cases is a delta robot.

This manual teaches you how to use *Pick And Place Components* and what the general workflow is.

Contents

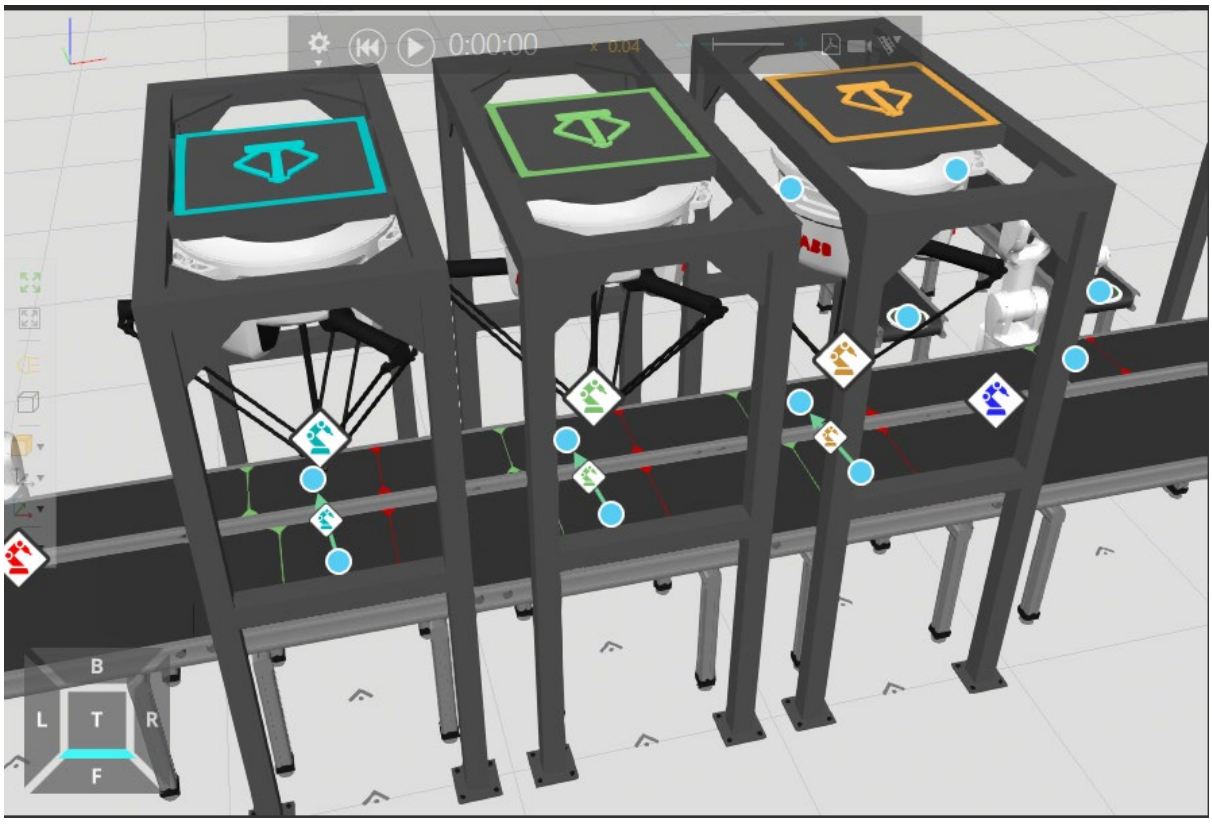
© 2023 Visual Components Oy

Workflow

The general workflow for pick and place, follows the standard workflow in Process Modeling. First you create product types and then you configure the processes. Finally you create the flow for your products and flow groups.

The workflow is different from regular Process Modeling, in that pick and place processes are created using Python, and you don't configure them by creating statements manually. Instead you modify process properties to control the process's behavior. Python is required, because the pick and place process cannot easily be defined as a linear set of statements, but there are parallel processes running at the same time. There can be many products and assemblies at the same time inside the process. The product may get transported to place process, or it might continue to travel along the conveyor, maybe to the next identical pick process.

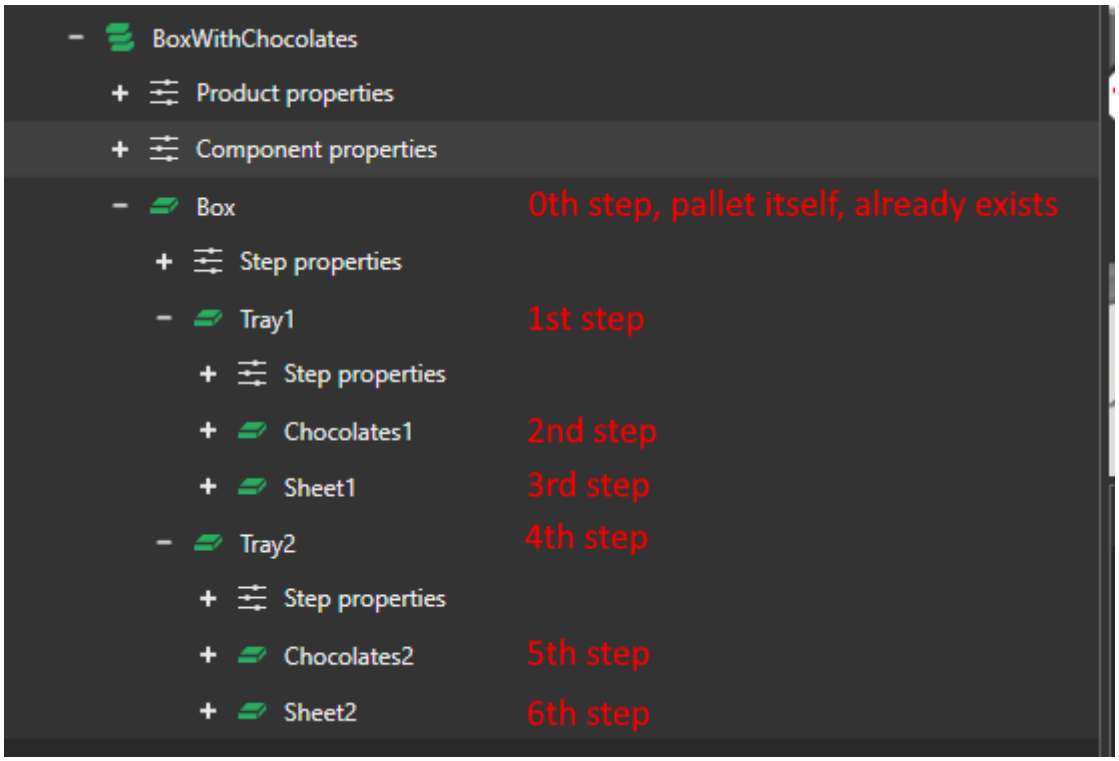
Pick and place process products travel on containers on a conveyor, so they don't require transport links. Transport links are needed for transport from one conveyor line to another. Also before and after the pick and place process, you may use Process Modeling flows as usual. The following image illustrates an example of a pick and place line. Notice that there are many processes in series. They all have the same process names in pick side and place side. The only transport links are from pick process to place process, and otherwise products are transported inside containers (conveyor paths).



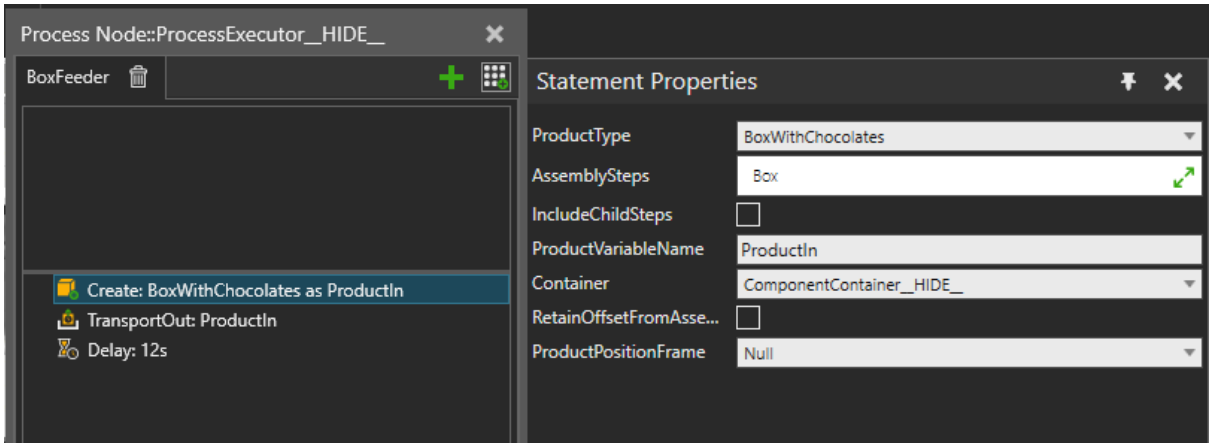
© 2023 Visual Components Oy

Picking And Placing On Assembly

In cases where products are picked from an assembly and/or placed to an assembly instead of a conveyor, you need to use Process Modeling assemblies to define how products are picked or placed on assembly slots. The place process will create transport tasks for assembly steps in *depth first* order. Picking from assembly handles steps in reverse order. Within the step, slots are picked or placed based on their *OrderIndex*, so the smallest index gets processed first unless the order is reversed with *ReverseSlotOrder* property. The following image illustrates an example of an assembly where a *Box* step is prefilled, and other steps are handled one at a time in place processes.

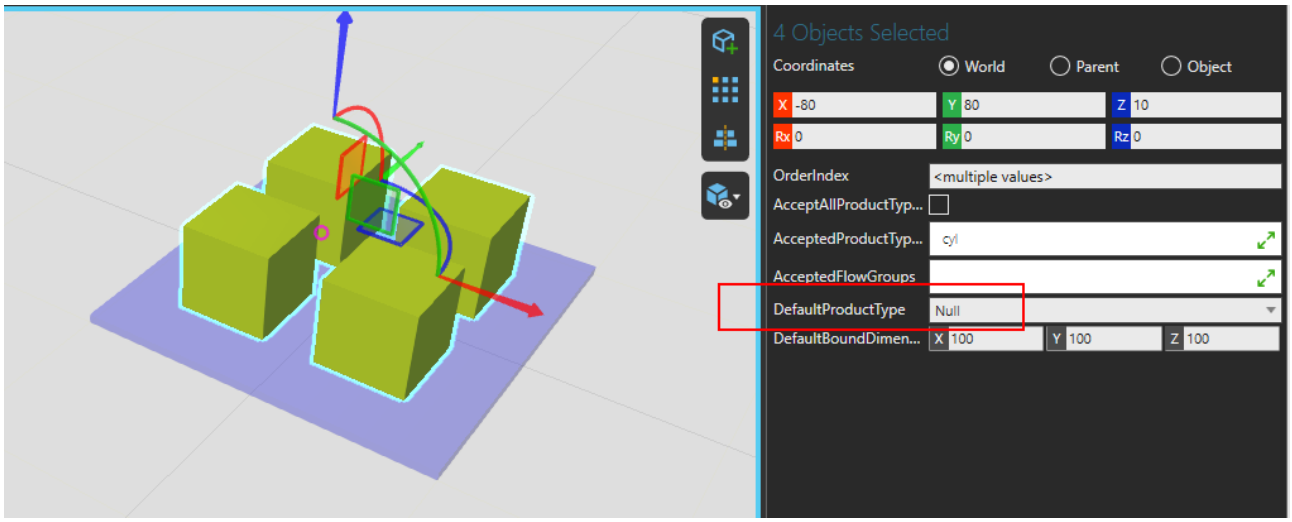


When using assembly for placing, you need to make sure that the incoming tray is already an assembly. So tray as plain product doesn't trigger transport tasks for place process. The following image illustrates an easy way to create an assembly on feeder process with the first step already included.



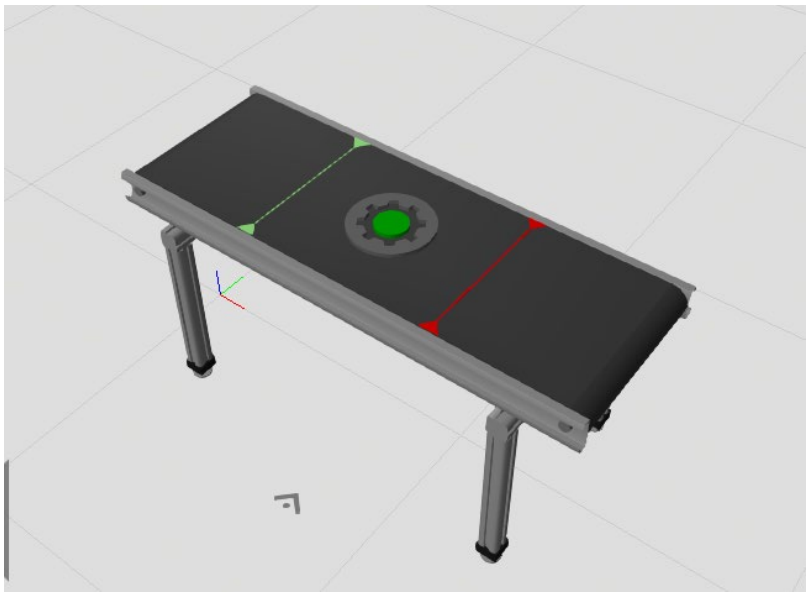
© 2023 Visual Components Oy

Pro Tip! Process feeder components don't allow you to define steps if you create assemblies with them. But there's a way to create an incomplete assembly with it, and that requires you to define assembly so that the slots you want to be empty during creation don't have **DefaultProductType** but it's set to **Null**. The following image illustrates an example of this.



Pick And Place Process Component

Pick And Place Process component can be found under **PM Flow Components** in the **eCatalog**. It is a process where parts can be picked and placed on moving conveyors. The conveyor has two sensors and any part in between those sensors is considered inside the process. When the product is inside a pick process it will be transported out if there's a transport controller free to do that. When an assembly is inside a pick process the products in the assembly slots are transported out. Products are transported to a place process where they can be placed either directly on a conveyor or into an assembly to a free slot. In the following illustration you can see the component and its input and output sensors as green and red lines.



© 2023 Visual Components Oy

Component properties

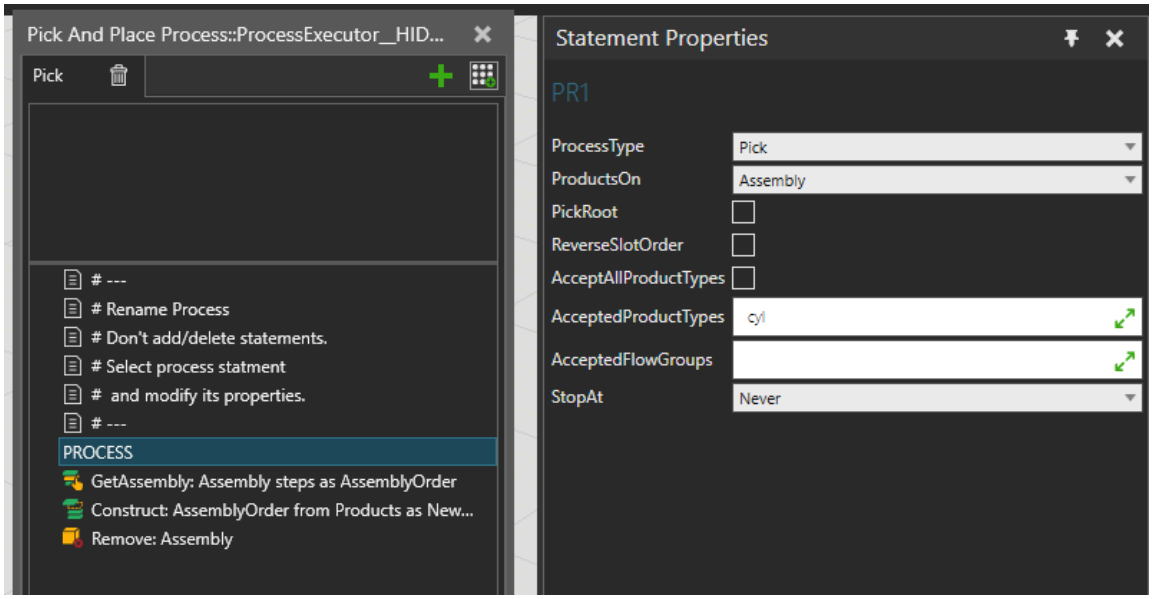
A component has all the same properties for conveyor appearance as a standard **eCatalog** conveyor component. On top of that, it has some other properties that are listed here.

- **InSensorPosition** – Adjusts green sensor position as a percentage value of total length.
- **OutSensorPosition** – Adjusts red sensor position as a percentage value of total length.
- **ShowProcessIcon** – Toggle process icon visibility on and off.
- **ShowSensors** – Toggle sensor visibility on and off.
- **ShowConveyor** – Toggle conveyor geometries on and off while still showing sensors and process icon.

Process

Pick And Place Process component's process consists of a Python process statement and a few assembly statements. You should not add or delete statements, but only change a Python process statement's properties. If you mistakenly delete some statements, you can reset the process back to default by clicking **ResetProcess** in the component properties in the **Process** tab.

One thing you should also do is rename the process. By default the name is *Pick*, but for place process you should rename it, to distinguish it from pick process. If you have multiple pick and place pairs, you can keep the same naming for processes that perform the same tasks. But if there are many different stages in the assembly, then pick and place processes could be named as e.g. *Stage1_Pick, Stage1_Place, Stage2_Pick, Stage2_Place* and so on. The following illustration shows the process Pick Configuration.



© 2023 Visual Components Oy

A Process statement's properties are as follows:

- **ProcessType** – Select either *Pick* or *Place* process.
- **ProductsOn** – Select if products are picked/placed from/to *Conveyor* or *Assembly*.
- **PickRoot (*Pick* from *Assembly*)** – Select if root step is picked from the assembly.
- **ReverseSlotOrder (*Assembly*)** – Pick/place step slots in normal or reverse order.
- Filters for selecting which products or assemblies are handled in this process.
    - **AcceptAllProductTypes**
    - **AcceptedProductTypes,**
    - **AcceptedFlowGroups**
- **StopAt** – Choose if a product or assembly is stopped *Never* or at *InSensor, Process* or *OutSensor*.

Process execution

Pick from conveyor process functions so that if a product arrives at the process (green in-sensor) there's a feed (Transport Out) created for it if the following are true:

- Product's flow group has transport link from the process.
- Product type matches the filter set in the process.

Note: there can be multiple parallel feeds active on the pick process at once. If a product reaches the end of the process (red out-sensor) its feed is cancelled, and it continues to travel along the conveyor. If the product has already been scheduled for transport, then it can be picked, even after crossing the end-sensor. You can force the product to stop by setting the *StopAt* property to some other value than *Never*. Then the product stops to wait for a transport, but only if there's a feed created for it.

Pick from assembly works quite similarly and products in the assembly are fed if the following are true:

- Product's flow group has transport link from the process.
- Product type or assembly type matches the filter set in the process.
- Product is at assembly root step and *PickRoot* is enabled.

Steps in the assembly are disassembled in reverse order. If a step cannot be disassembled completely in this process, then the process passes the assembly forward on the conveyor. Slots within the step can be disassembled in normal or reverse order, by selecting it with *ReverseSlotOrder* property. Assembly can also be stopped with *StopAt* property.

Place to conveyor process works so that it will create a need (Transport In) using the filter in the process. By default the filter is set to accept any type (*AcceptAllProductTypes* is true). When there's a matching feed for the need, then the matching product is scheduled for transport and a new need is created immediately. So there's always only one pending need at a time, but there can be multiple products being transported at the same time.

© 2023 Visual Components Oy

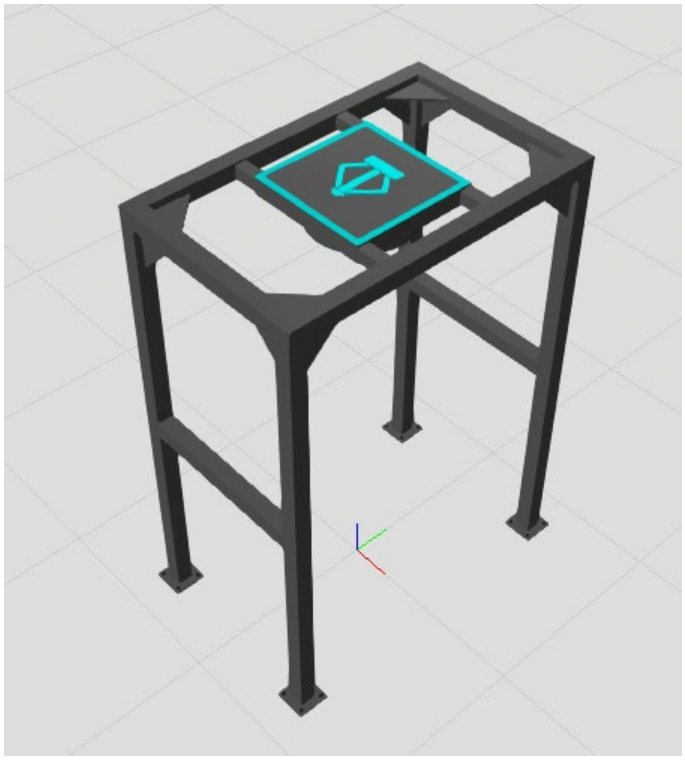
Place to assembly process waits for an incomplete assembly to arrive at the start of the process (green in-sensor). When that happens, there is a need created for the first empty slot in the first incomplete step if the following points are true:

- Needed product types must pass the process filter or the whole assembly passes the process filter.
- Needed product types and their flow groups have transport link to the process.

When a need is matched then it's scheduled for transport and if possible the next feed is created on the assembly. Steps in the assembly are assembled in normal depth-first order. If a step cannot be completed by this process, then the assembly is passed forward on the conveyor. Slot order can be normal or reversed using the *ReverseSlotOrder* property. If an assembly reaches the end of the process all pending needs are cancelled. Needs that have been matched and scheduled for transport can still be completed after crossing the end-sensor. If all slots were filled then the process completes the assembly using the *Construct* statement. You can stop the assembly using the *StopAt* property if you want to make sure all possible slots are filled.

Pick And Place Transport Controller Component

Pick And Place Transport Controller component is found in **PM Transport Controllers** in **eCatalog** and it is similar to a normal *Robot Transport Controller* (RTC) except it has the ability to track products on moving conveyors. Some of the features from standard RTC have been removed to simplify the controller's logic. Another big difference is the appearance. Pick and place is often done with ceiling mounted delta robots, so the default appeared of the component is a ceiling mount frame which you can see in the following illustration.



© 2023 Visual Components Oy

Component Properties

Default Tab

Properties on the default tab are related to the looks of the component. *PedestalHeight* and *FrameHeight* will also affect the robot's mount location.

- **Looks** – Choose looks of the component.
- **PedestalDiameter** – Defines the width / length / diameter of the pedestal.
- **PedestalHeight** – Defines the height of the pedestal.
- **NOTE:** Ceiling Mount Frame only Properties:
  - **FrameLength** – Defines the length of the frame.
  - **FrameWidth** – Defines the width of the frame.
  - **FrameHeight** – Defines the height of the frame.
  - **FrameThickness** – Defines the thickness of the frame.
  - **ShowFrame** – Show or hide the frame.
- **VisualizeStateChanges** – When enabled changes the color of the pedestal according to the current state.

Speeds Tab

- **Enabled** – When disabled uses the maximum speed values of the robot component. When enabled overwrites the robot components speed values.
- **JointForce** – Joint acceleration for PTP motions.
- **JointSpeed** – Joint speed for PTP motions.
- **AngularAcc** – TCP reorientation acceleration for LIN motions.
- **AngularSpeed** – TCP reorientation speed for LIN motions.
- **CartesianAcc** – TCP acceleration for LIN motions.
- **CartesianSpeed** – TCP speed for LIN motions.

AutoHoming Tab

Auto homing can be used to force the robot to automatically move to home position if it has no tasks to perform. The initial position the robot takes when the simulation is reset is considered as the home position, unless there's a sub program *home* available on the program tab and it is not empty. If the *home* subprogram is available, it is executed always when robot needs to go home instead of the initial homing position.

- **Enabled** – Automatic homing is in use when the flag is checked.
- **Delay** – Defines the delay robot is allowed to idle before it automatically goes to home position (or the *home* sub program is called).

LinkDefaults Tab

Link Default tab properties can be utilized to modify the default automatic motions. These properties are also used as default values for the transport link properties that are assigned upon transport link association with this controller. If transport link property *UseCustomParameters* is disabled (i.e. unchecked) these default properties are actively being used.

© 2023 Visual Components Oy

If *UseCustomParameters* is enabled the transport link properties will overwrite values defined in these default properties. Learn more in the Transport Link Properties section of this document.

Advanced Tab

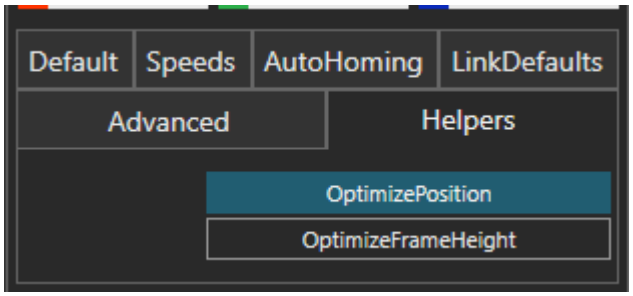
- **UseGripperSignals** - Controls the connected gripper tool with signals by closing it when grasping and opening it when releasing. You can define signal ports in transport link properties. Alternatively, you can set ports to -1 in which case the controller tries to look for signals in the tool component based on naming convention.
    - If there is a signal named “IN_J1_Action” in the tool component, that signal is triggered True to close, and False to open.
    - Or else “IN_J1_Close” is triggered True to close.
    - And “IN_J1_Open” True to open.

Transport Tab

Capacity – For multipick scenarios you can define how many products the robot can transport at once.

Helpers Tab Properties

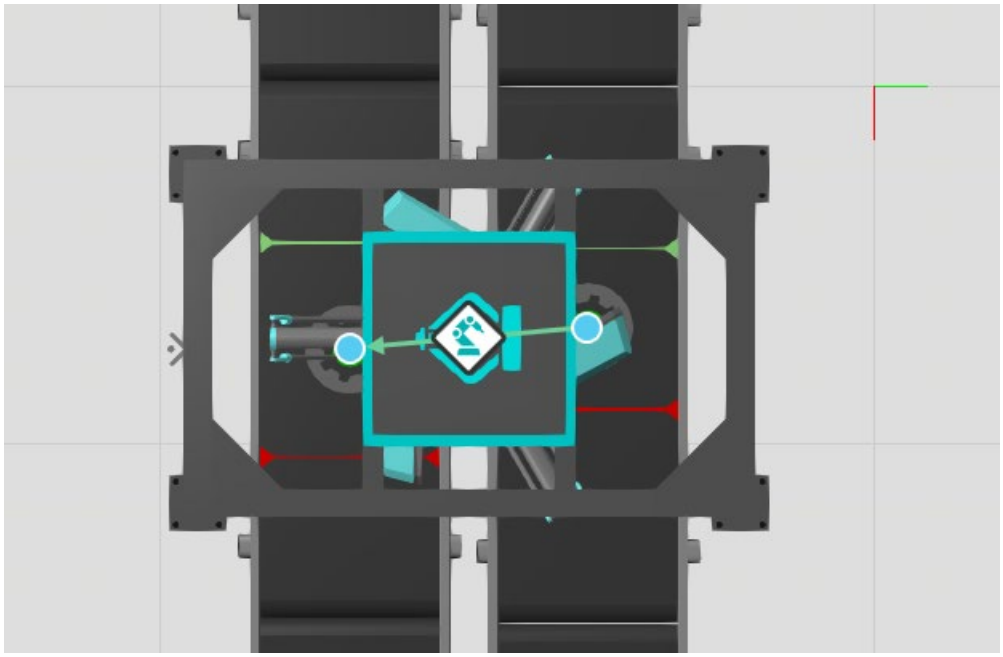
Helpers tab includes few helper tools that mainly automate configuration of the controller component’s position and height in the layout.
NOTE: these helpers are meant for the ceiling mount frame option.



© 2023 Visual Components Oy

OptimizePosition button automatically manipulates the component's X and Y position and rotation around Z axis based on the processes that are associated with this transport controller. Before using this helper make sure to configure your process flow and associate this controller to desired transport links.

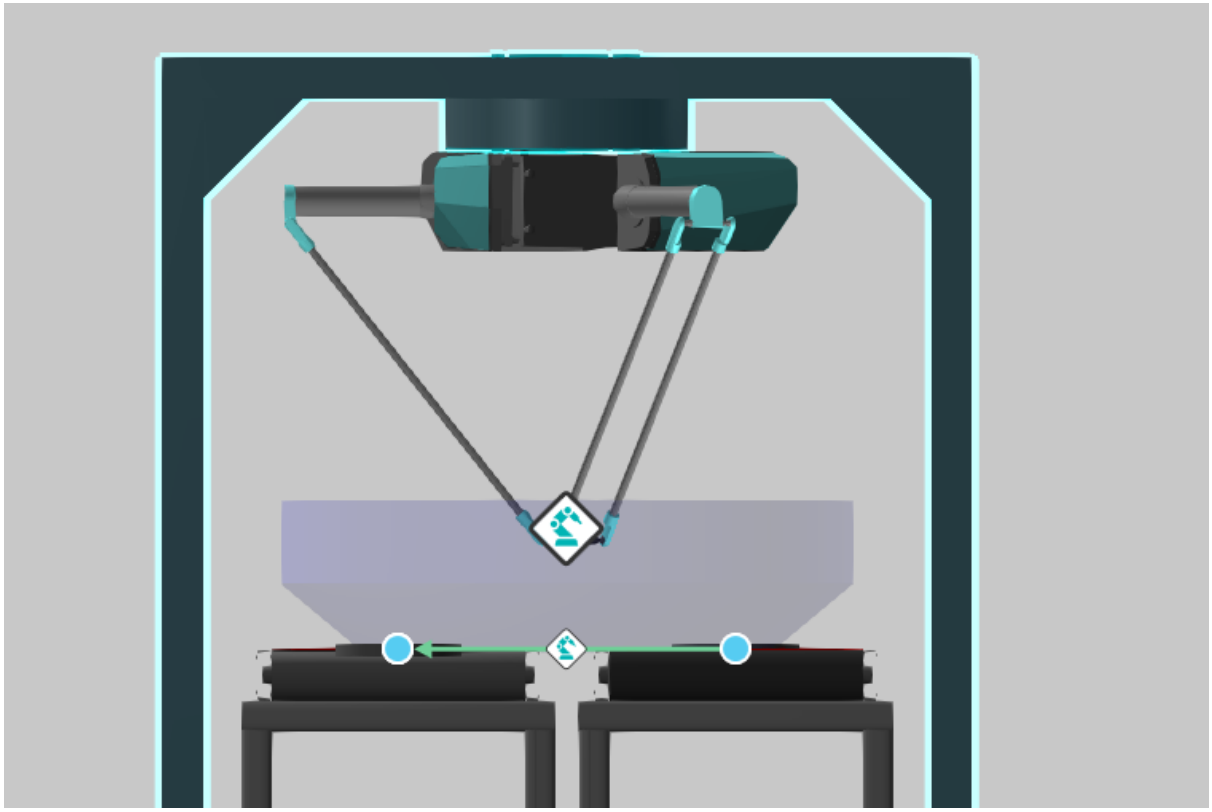
Then click the *OptimizePosition* button and the Python script will position the component at the middle of connected processes. Rotation around Z is set based on the first connected process. Often in pick and place applications all conveyors are parallel so this tool relies on that assumption. The following illustration shows how this helper positions the controller at the middle of two processes and orients it the same way as the conveyors.



© 2023 Visual Components Oy

OptimizeFrameHeight is available when using a ceiling mount frame. Like the previous helper you need to have connected the controller to processes using transport links. When you run the helper a script will check what is the lowest reachable position with the connected robot and selected TCP (LinkDefaults::TcpIndex). Then it will check what is the lowest associated process.

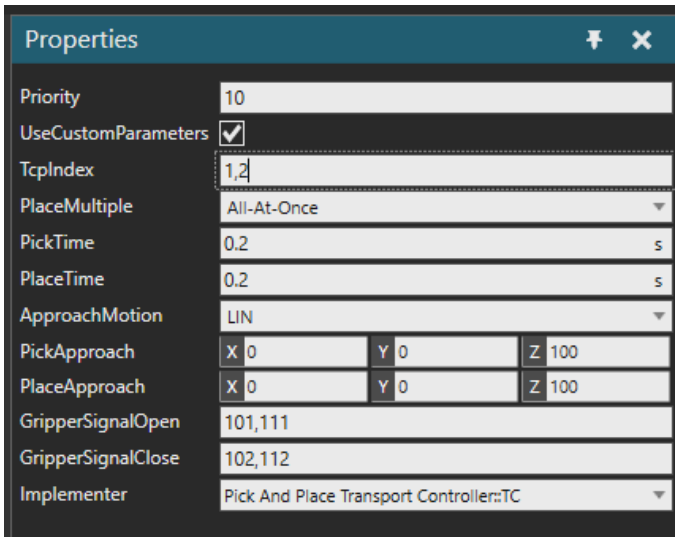
Helper will then adjust FrameHeight so that the lowest process is barely reachable. Note that helper only checks reachability in world Z-axis. In the following illustration you can see how the helper adjusted the frame height so that delta robot barely reaches the processes using the default tool.



© 2023 Visual Components Oy

Transport Link Properties

As with other transport controllers you can control a transport task in pick and place process using properties on the transport links.



- **Priority** – Defines the priority of transportation via the corresponding link among all active transportation tasks. The priority is a number. The lower the value, the higher the priority, e.g. 1 = high priority and 100 = low priority. Tasks with equal priorities are dealt with *FIFO* principle.
- **UseCustomParameters** – Enable this property to overwrite the default values for properties defined in the *LinkDefaults* tab of the controller properties.
- **TcpIndex** – Defines tool frame index used for transportation tasks.
    - All robots in the Visual Components **eCatalog** have 16 tool frames by default.
    - When an end-effector with a predefined tool is connected to a robot, it is automatically added to the list of tool frames with the index of 17. When the value is 17, the tool frame of the connected end-effector is used.
    - If the defined tool frame index is not found (for example, when no end-effector is connected), the first tool frame is used.
    - For multipick scenarios set multiple TCP indices as comma separated list.
- **PlaceMultiple** – In multipick scenario select if products are placed *One-By*-One or All-*At-Once.*
- **PickTime** – Amount of time that the robot spends when grasping the product.
- **PlaceTime** – Amount of time that the robot spends when releasing the product.
- **ApproachMotion** – Sets either PTP or LIN motion for approach / departure / home motions. Exact pick and place motions are always LIN.
- **PickApproach** – Vector value to define the approach offset from the pick location. The vector is applied in the target component's coordinates from the pick target. Default target is on top of the product component. The pick location can be defined with a frame feature in the product component named as "pick".

© 2023 Visual Components Oy

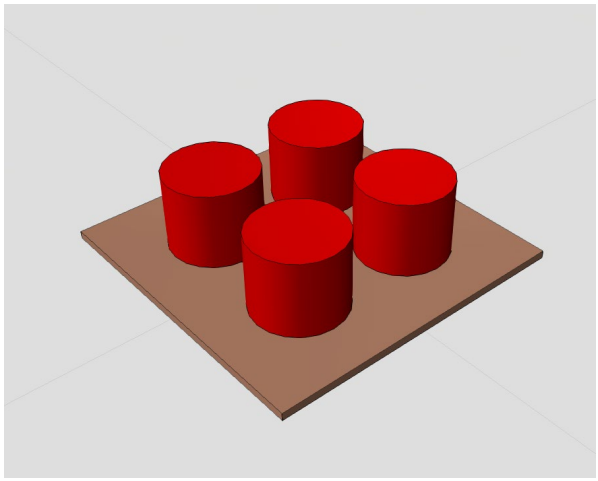
- **PlaceApproach** – Vector value to define the approach offset when placing the part to the target location. In the case of placing on a conveyor the target is the place process (shown with the icon). In the case of placing on a pallet, the assembly slot defines the target. The vector is applied either in process coordinates (place on conveyor) or pallet coordinates.
- **GripperSignalOpen** – Set output port(s) for opening the gripper when placing a product. Value *-1* doesn't use output ports but then signals are automatically searched from the tool component. For multipick scenario you can define multiple signals as a comma separated list.
- **GripperSignalClose** – Set output port(s) for closing the gripper when picking a product. Value *-1* doesn't use output ports but then signals are automatically searched from the tool component. For multipick scenario you can define multiple signals as a comma separated list.
- **Implementer** – Choose the transport controller to implement transport**.**

© 2023 Visual Components Oy

Example Cases

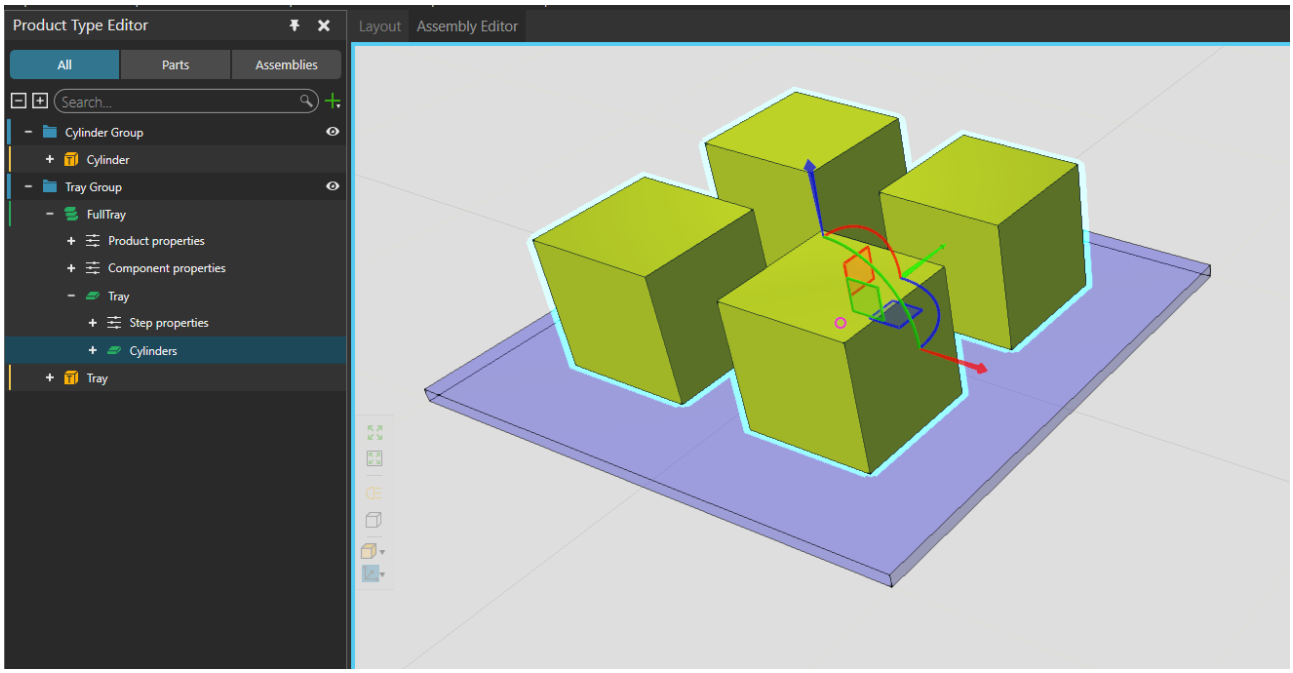
Pick Products From Conveyor To Tray

In this simple example, the goal is to pick cylinders from a conveyor and place them on a tray moving on another conveyor. The final assembly has 4 cylinders on a tray as seen in the following illustration.



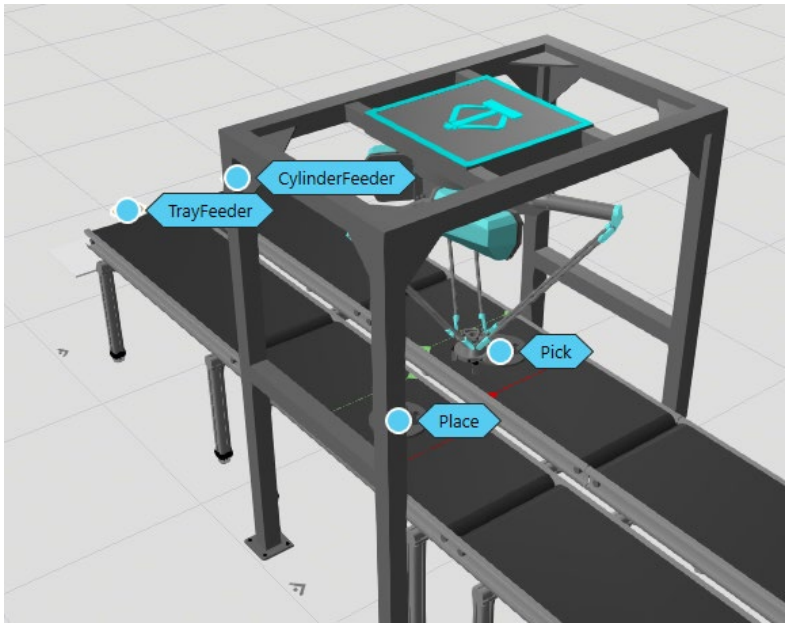
Following the common Process Modeling workflow let's define the **products** for this simulation. There's a product type for both the cylinder and the tray and these types are placed on different flow groups. In the tray flow group there's also assembly for a tray full of cylinders. In the assembly root step is the tray and its child step contains the cylinders as 4 slots.

Slots accept the cylinder product type and their default product type is set to *Null*. This way you can create assembly instances with product feeder so that the cylinder slots are empty. Products for this example are shown in the following illustration.

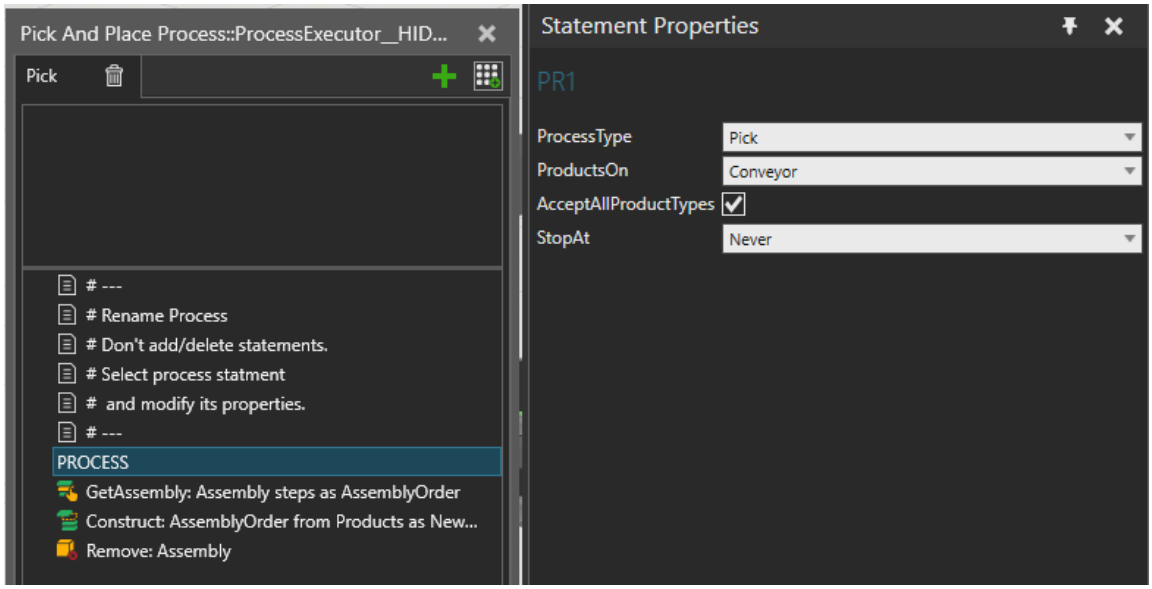


© 2023 Visual Components Oy

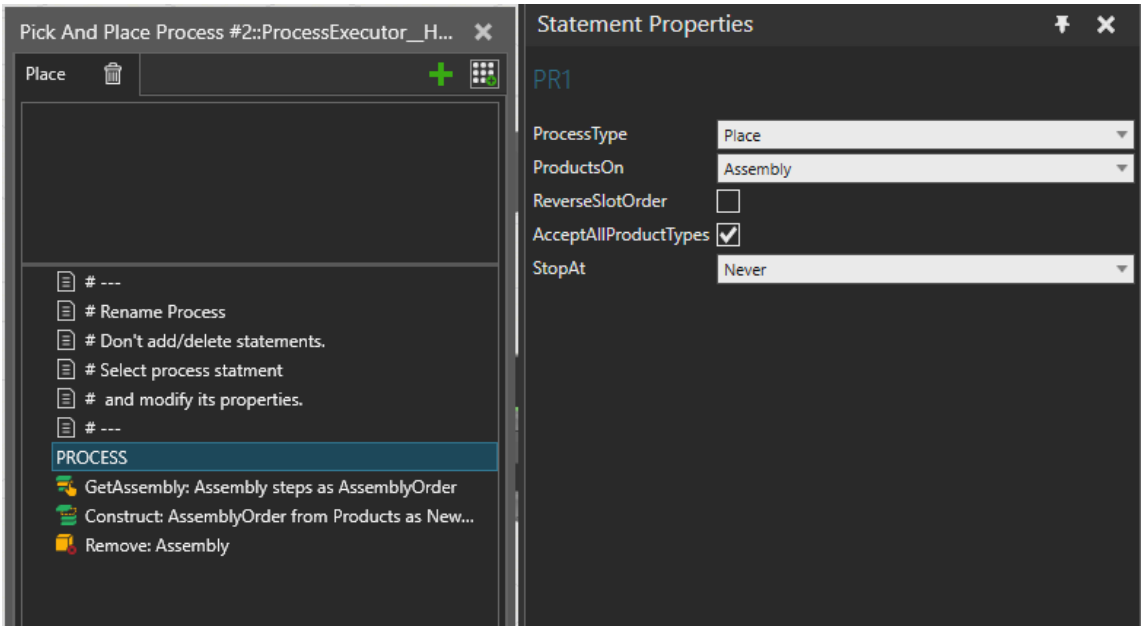
Next let's have a look at the **processes**. In this case we have feeder processes for both the cylinders and the trays. Feeders are connected to conveyors and then to *Pick And Place Process* components. On the placing side, the process is renamed to *Place*. The setup looks like this:



Pick process is configured to have products on conveyor and place process has products on assembly. Process filters accept any product types and processes don't stop the products. Following illustrations show the exact process properties.

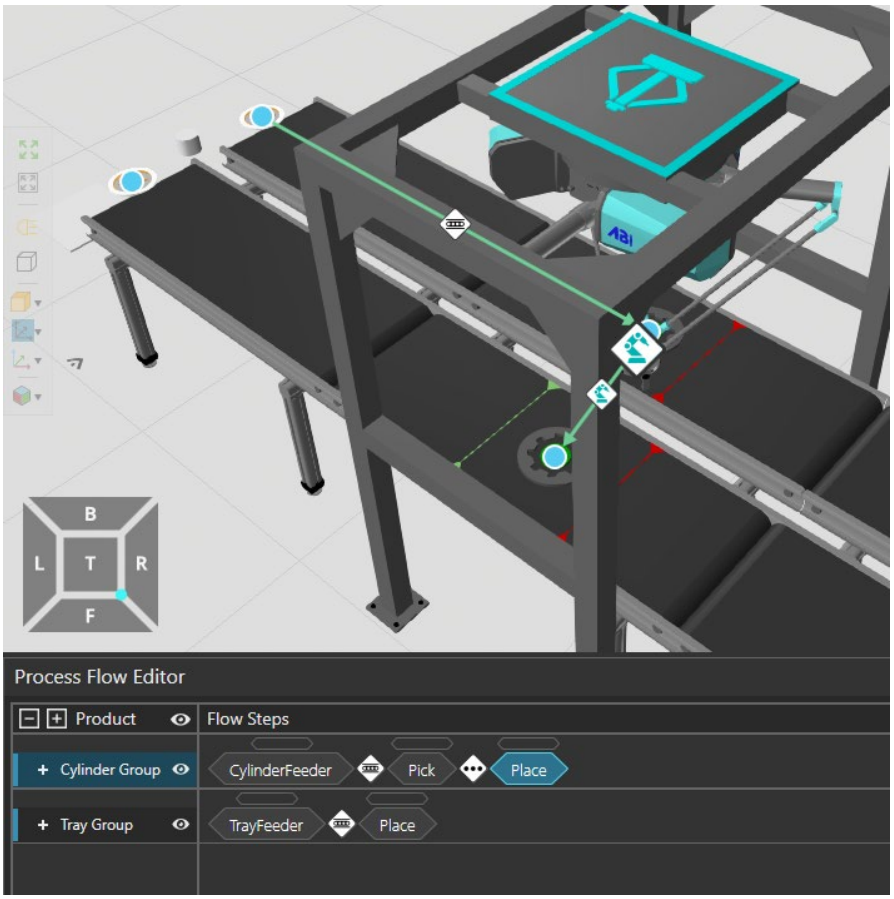


© 2023 Visual Components Oy



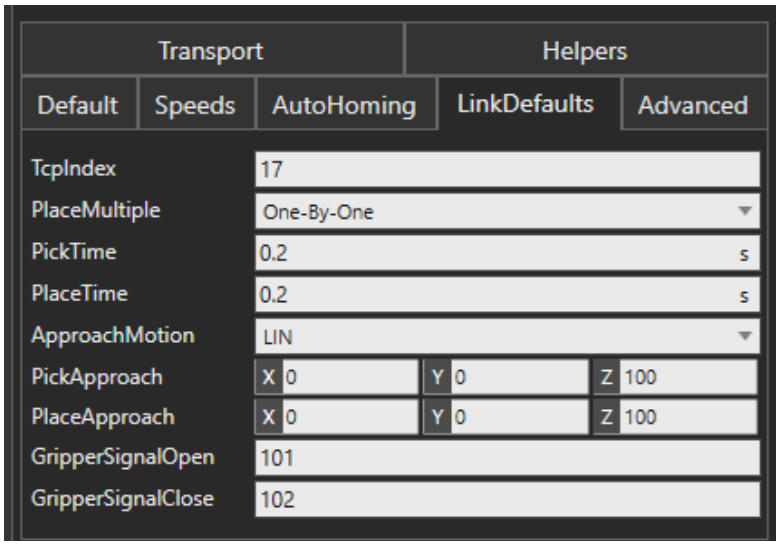
We have placed a *Pick And Place Transport Controller* over the pick and place area and it has a delta robot attached to it. **Flow** for cylinders starts from the feeder, goes to pick process and from there to place process. Pick to place transport is using the *Pick And Place Transport Controller*.

Transport from feeder to pick process is using conveyor transport. Flow for tray is simply the feeder process and place process and transport is done with conveyor. Following illustration shows the flow definitons for this example.

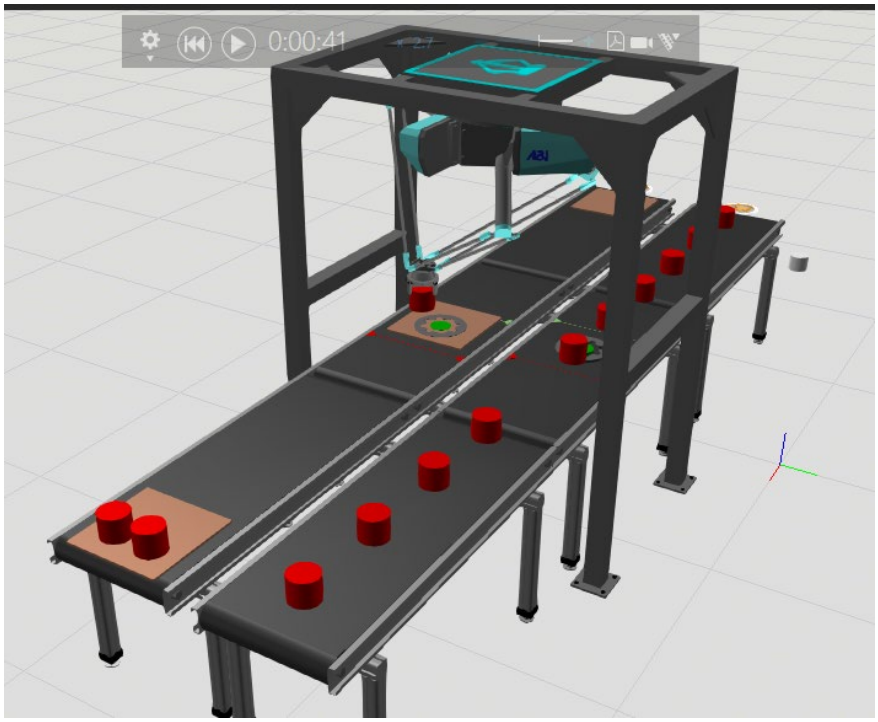


© 2023 Visual Components Oy

Transport link is set to use Transport Controller's default properties which you can see here.



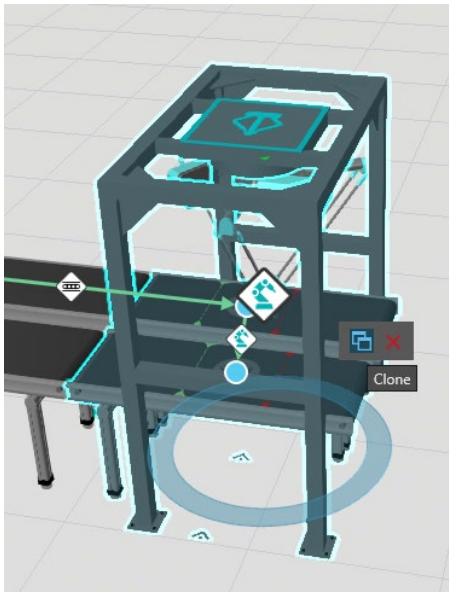
With the products, processes and flows defined the system is ready to be simulated. The result can be seen in the following illustration.

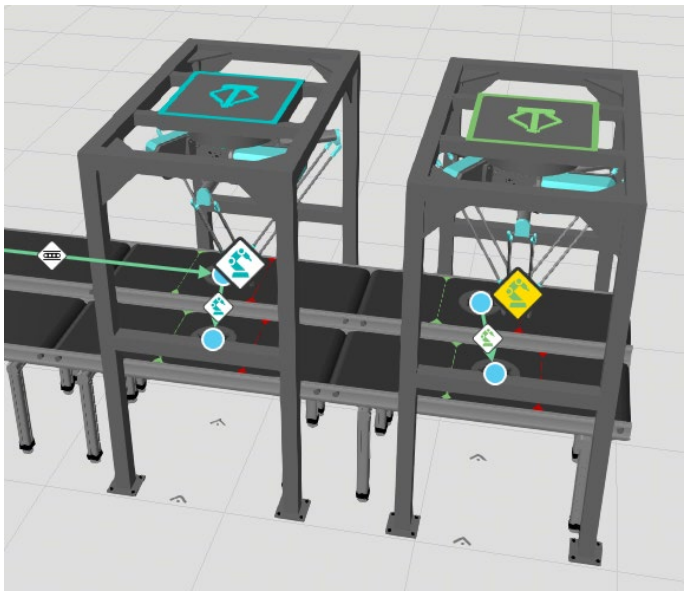


You see that conveyors move too fast for the robot and it can only pick 2 out of the 4 cylinders per tray. In this case you could either slow down the tray conveyor or make it stop. Or if you need high capacity then you can leave the conveyors as is and add a second pick area and robot.

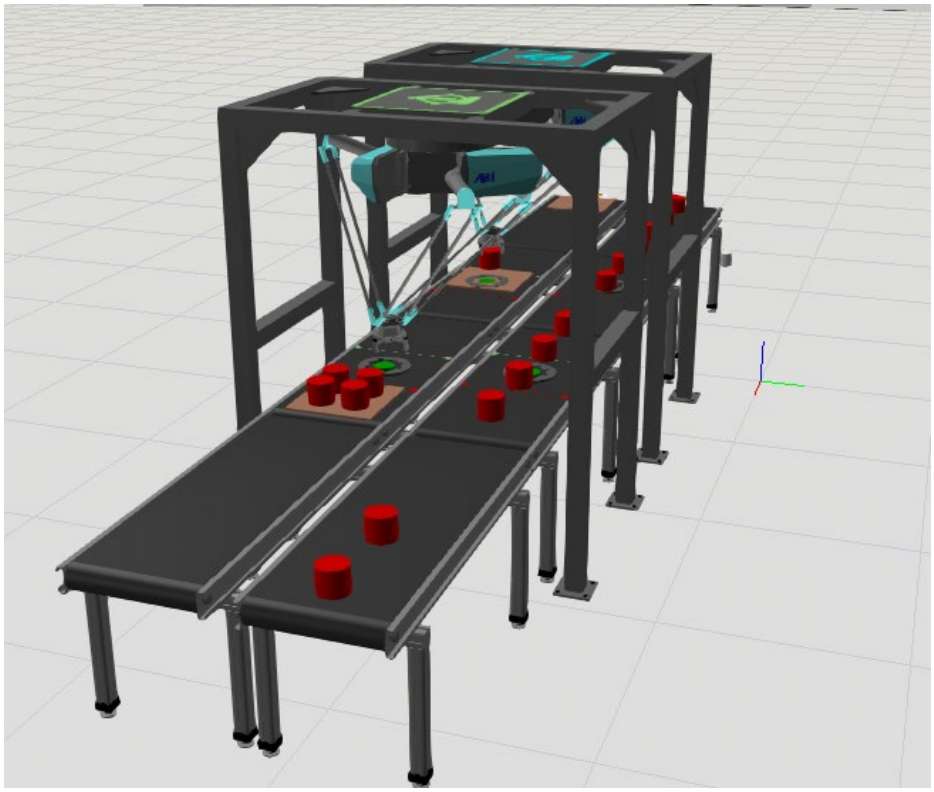
This can be easily made by selecting both the pick and place process components and the transport controller and cloning them. The cloned processes are attached using the Plug And Play (**PnP**) tool and transport link is created using the cloned transport controller as seen in the following illustrations. Note that you don't need to create conveyor transport links from first pick/place process to the cloned process.

© 2023 Visual Components Oy





Now that we have doubled the transport capacity with the second robot, the trays get filled completely as seen in the next illustration. You could continue handling the completed assemblies and missed products downstream using standard Process Modeling features.

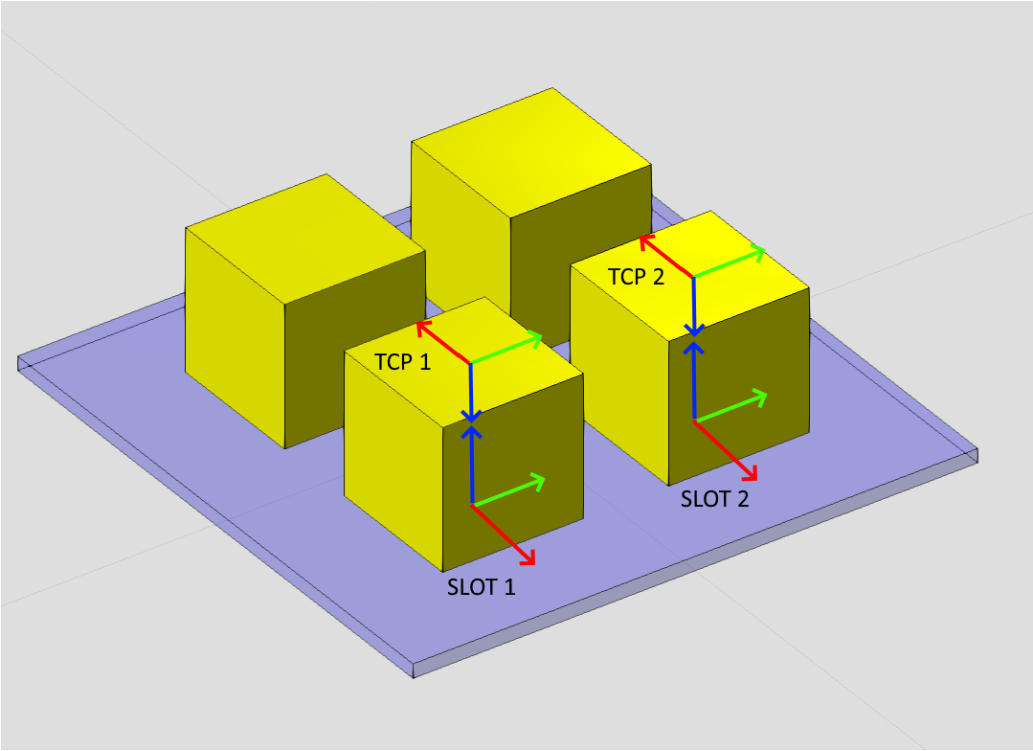


© 2023 Visual Components Oy

Double Gripper

Let's consider a follow-up for the previous example where robot has a double gripper. It picks two cylinders before placing them both at the same time onto the tray. For this, the assembly slot pattern needs to be defined so that the slots have uniform distance between each other. In other words slots 1 and 2 are filled at the same time and so are slots 3 and 4.

The distance between those slots in a pair must be the same for all pairs. That distance also defines how you set up the robot tool frames. Frames must have the same distance between each other as the products in the slot pair. That way it is possible to place two products perfectly at the slots at once. The orientation of the products and the tool frame matters. In the following illustration you can see the orientation of slots 1 and 2 and the correct orientation for tool frames 1 and 2.



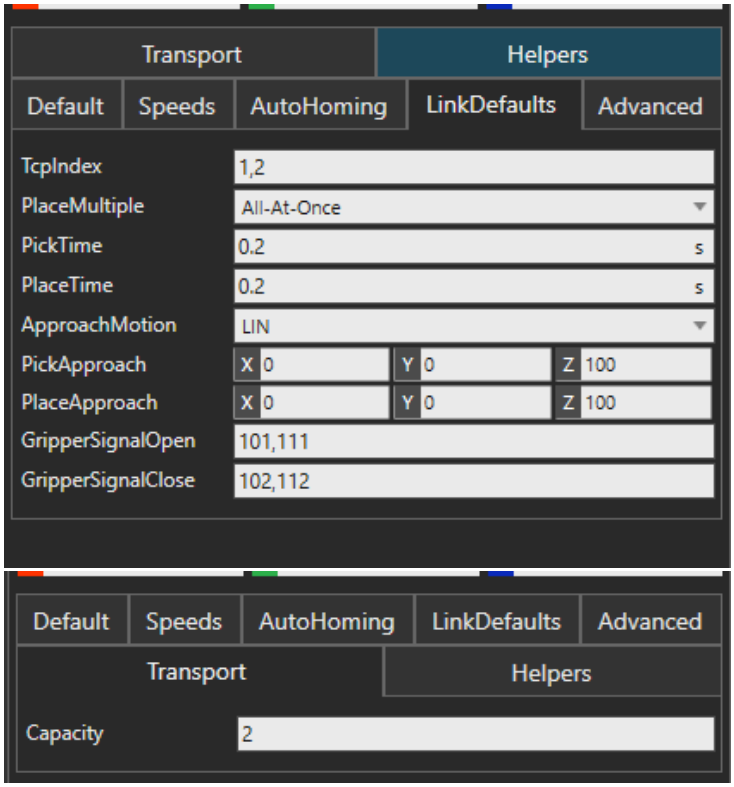
Note that in delta robots, tool frames usually have their Z-axis pointing downwards. So the default pick matrix for a product is at the center of the top surface on the product and the orientation is the product's orientation but rotated 180 degrees around Y-axis.

After setting up the tool frames for the robot, the next elements to be changed are the transport controller and transport link properties.

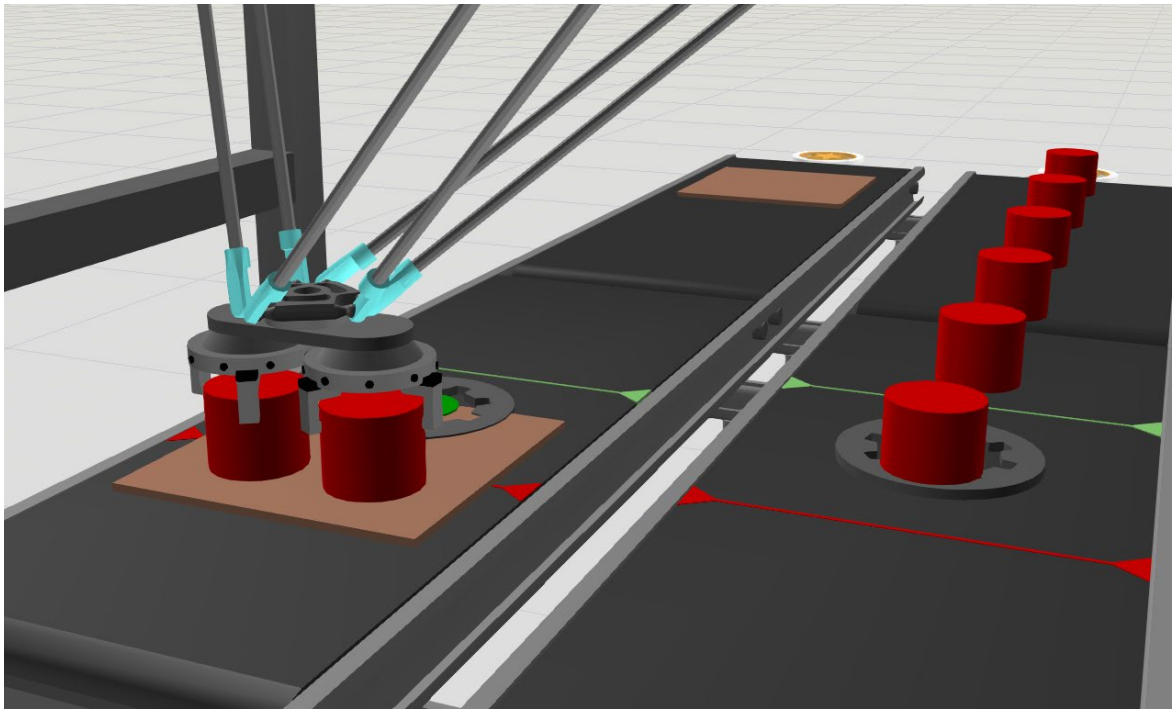
- On the controller's properties under **Transport** tab **Capacity** needs to be set to *2*.
- In link properties *TcpIndex* is now a comma separated list so for example *"1,2"*.
- *PlaceMultiple* is set to *All-At-Once*.
- When using gripper signals *GripperSignalOpen* and *GripperSignalClose* need to have comma separated lists of output ports for both grippers.

© 2023 Visual Components Oy

The following illustration shows an example of Transport Controller's properties in the case of double gripper.



With these changes to the single gripper case, the robot now picks two products and then places them at once on the next two free slots in the assembly.



© 2023 Visual Components Oy