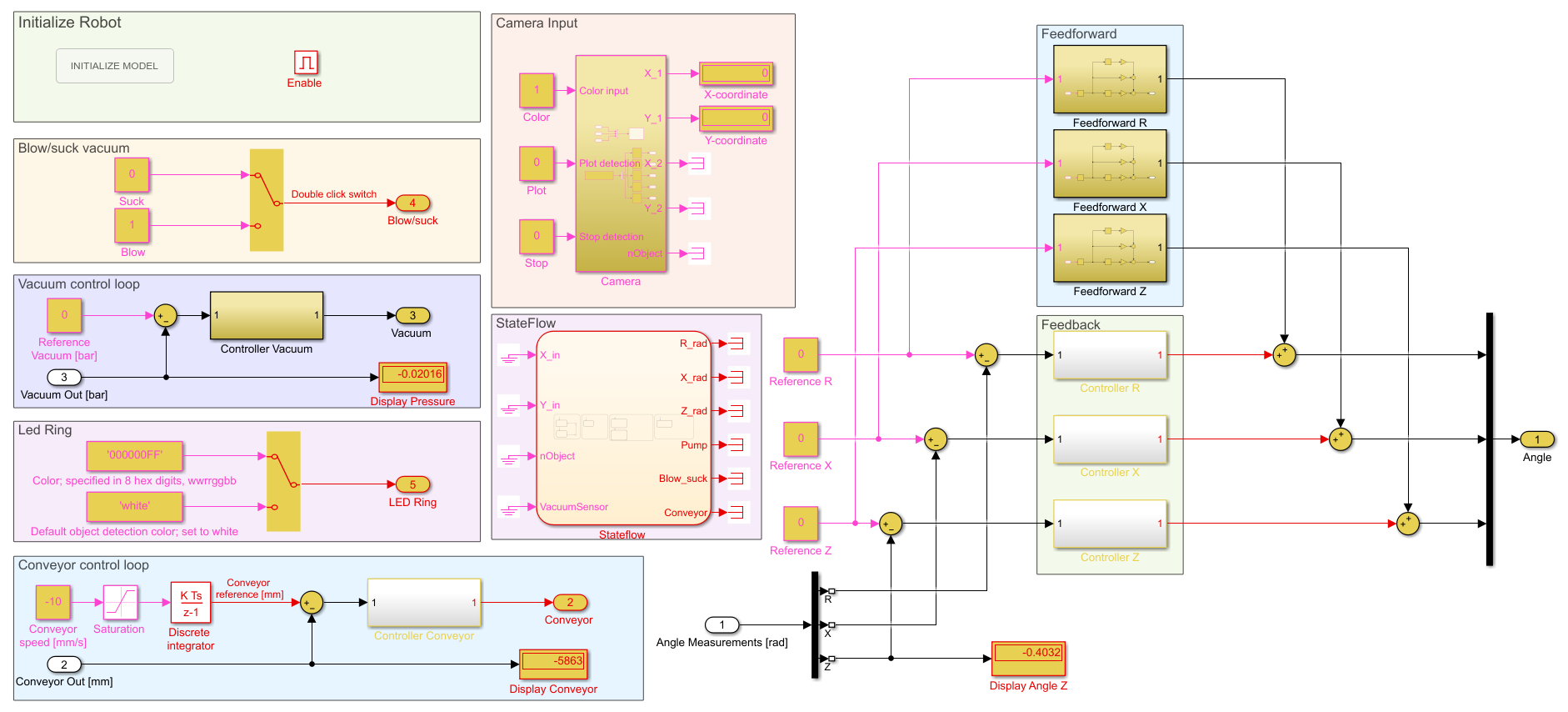
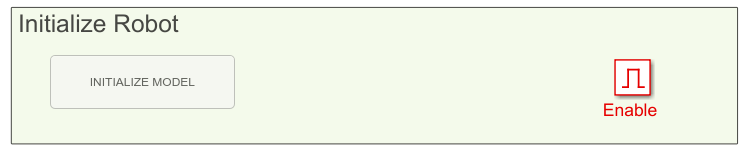
**Simulink student file explanation**



This file contains a short introduction to the Simulink file, which is depicted in the figure above. The following topics will be discussed:

* Initialize robot
* Blow/suck vacuum
* Vacuum control loop
* LED ring
* Conveyor belt control loop
* Camera Input
* Stateflow
* Feedback/feedforward part
* How to control the Robot

**Initialize Robot**

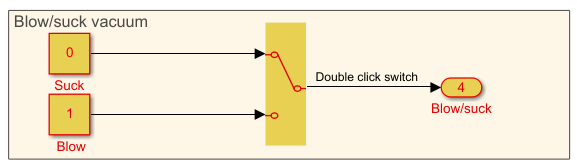


Click the **initialize model** box twice before running the robot, this only should be performed once per setup.

In Matlab the number of your specific robot is asked, which can be found on the setup. 

After initialization the robot always homes on the same position (x = 150 mm, y = 0 mm, and z = 100 mm) and the camera coordinates are translated to the distance from the center of the robot.

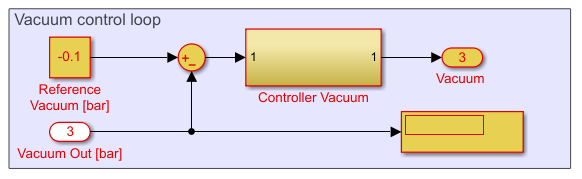
**Blow/Suck vacuum**



The Simulink file has a switch to determine if the vacuum should suck the products **0** or blow products away **1**. The switch can be controlled by double clicking the switch on the run.

The vacuum control loop will be enabled when the vacuum is sucking products, such that the pressure can be controlled. Blowing products away is always performed at maximum blowing power.

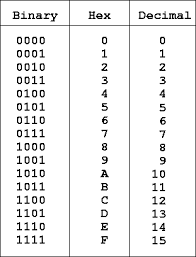
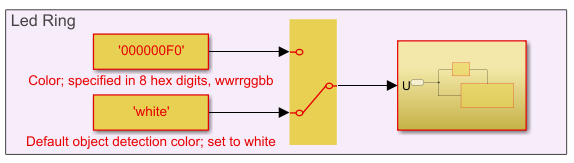
**Vacuum control loop**

****

The force of the vacuum can be determined in this control loop, where the reference vacuum is the proposed pressure. Vacuum out is the measured pressure[bar].

The vacuum sensor is an absolute sensor which means the sensor measured the vacuum relative to the air pressure settings.  
The input for the vacuum is between -1 and 0 bar, where 0 bar is no pressure difference with respect to the air pressure and -1 bar is complete vacuum.

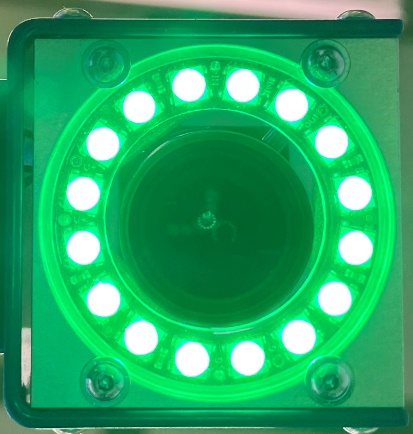
**Led Ring**



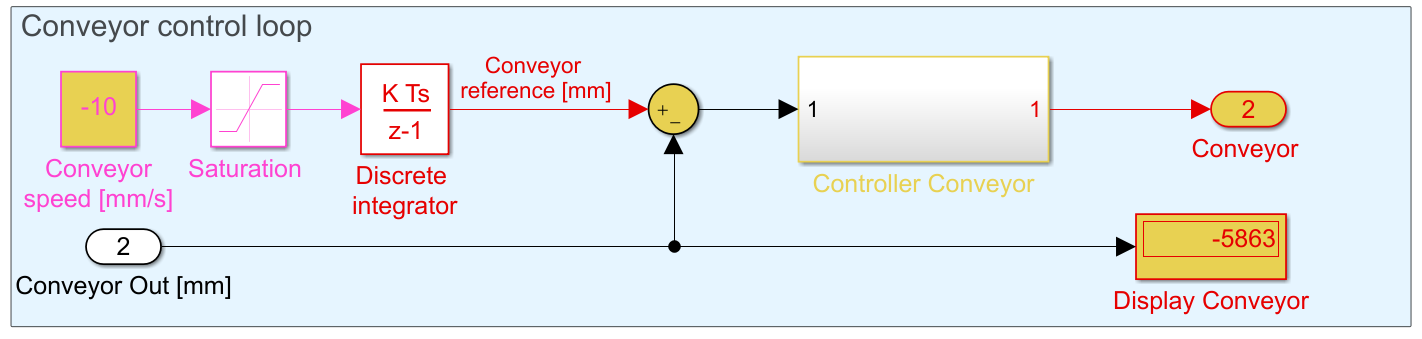
The Simulink file has a switch for a color that can be chosen and white. White is also an option as this is the setting used for the default object detection that is already given. To change the color, you need to specify the color in 8 hex digits, in wwrrggbb. This means that for each color; white red green blue, you need to specify the intensities in hex digits, the first digit is the intensity and the second needs to be zero. The intensity can be chosen between 0 and 15, so 16 options. For example, if you want only red the color is 00F00000. The red intensity is set to F0, which is 15 and thus maximum. The other colors are set to 00, so an intensity of zero. If you want a blue color set 000000F0.

“Color is specified in 6 hexadecimal digits specifying the color in RGB order. Specifying 102030 will set the red component to 0x10, or about 1/16th of full intensity, the green component to 0x20/0xff or about 1/8th of full intensity and blue to about 3/16th.” Note that this is for 24 bits and not for 32 bits, as is for our case.

While running the robot, you will notice the color of the LED ring will change, during homing the led turns Orange, after finishing the homing procedure the led turns green for one second. The led turns red if the robot exceeds the boundaries.



**Conveyor control loop**

****

The speed of the conveyor belt can be controlled in this box. The reference is in mm/s, where the speed is bounded between 100 mm/s and -100 mm/s. This reference is integrated such that the input for the conveyor belt is position instead of velocity. The conveyor belt controller has already been implemented and can be found in the Controller Conveyor subsystem.

**Camera input**

To be able to run the object detection in matlab the following toolboxes are required:

**Required toolboxes in Matlab**

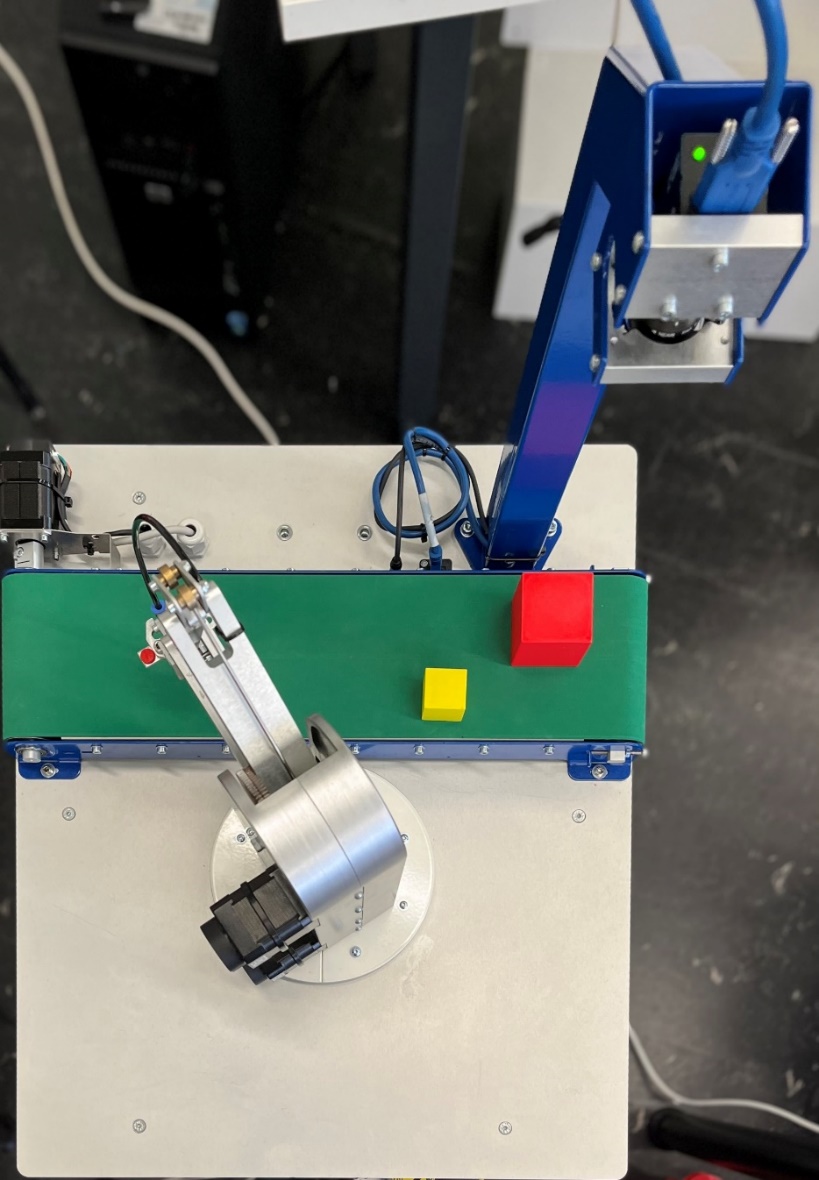
* Image Acquisition Toolbox
* Image Acquisition Toolbox Support Package for OS Generic Video Interface
* Image Processing Toolbox
* Computer Vision Toolbox

**Connect Camera**

Install Daheng Galaxy viewer: **Galaxy\_Windows\_EN\_32bits-64bits\_1.12.2106.9032** from folder Camera\ to be able to connect the camera to your laptop.

**Perform Object Detection**

On Canvas in the folder Camera\_model camera.m must be downladed. **Camera.m** can be executed to detect objects from the robot’s camera. The coordinates of the product are sent to the Simulink file, so a Simulink file of the robot should be opened, whereas it is not necessary to run this file while doing the object detection.

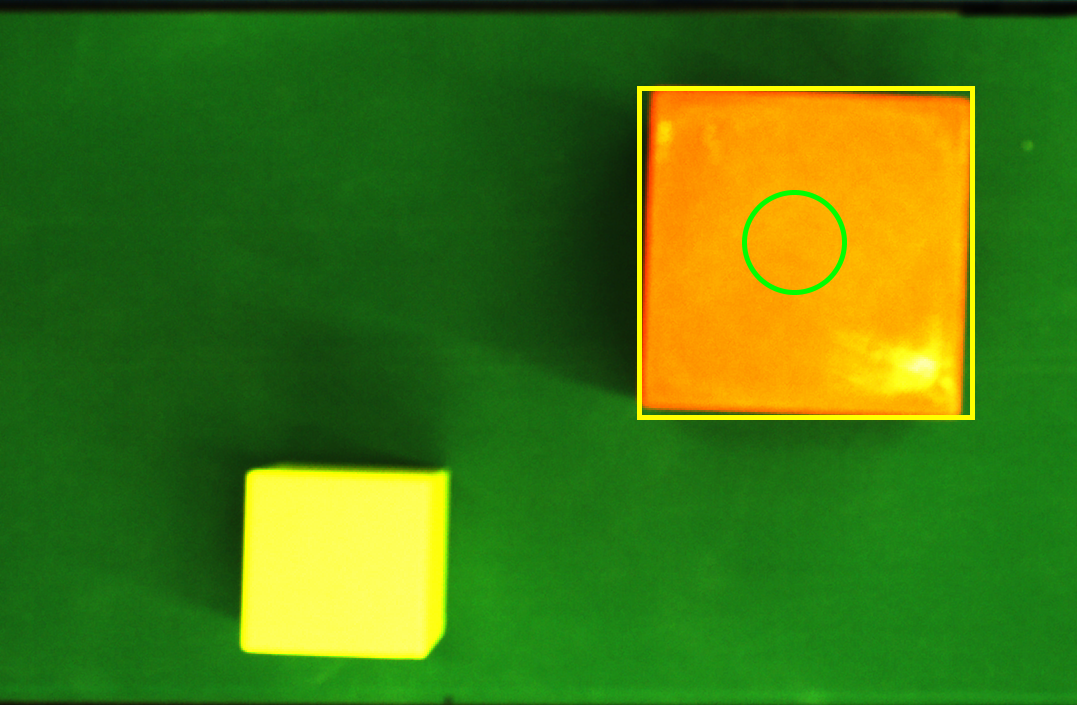


X [mm]

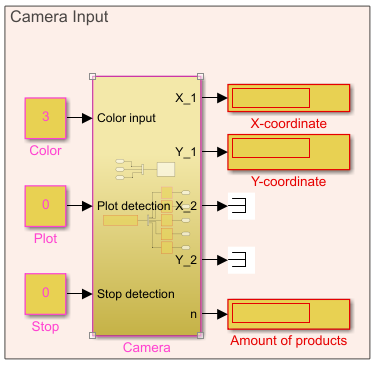
Y [mm]

(120,-110)

(180,-170)



The position of the boxes is determined from the center of the robot in mm, as can be seen in this figure.

****

In this box the centers of the detected boxes are displayed, (X,Y)-coordinates for two products are displayed. The number of detected products is displayed at **n.**

The following settings of the object detection can be changed.

The **Color** parameter determines which product is determined from object detection:

* Red is detected by changing the constant to **1**
* Blue is detected by changing the constant to **2**
* Yellow is detected by changing the constant to **3**

Plotting the camera can be enabled by changing the **Plot** parameter to **1** in the Simulink file, which is useful for testing the camera. Plotting the objects is stopped by closing the figure.

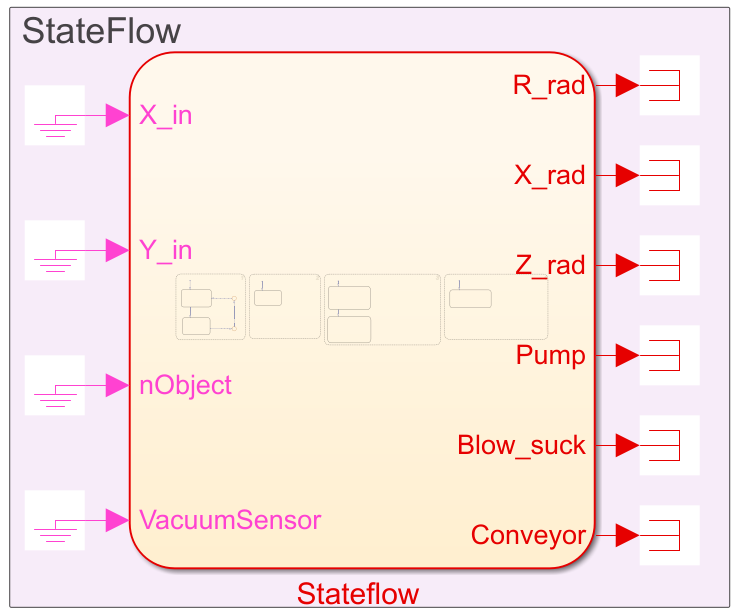
The object detection can be stopped by changing the **Stop** parameter to **1.**

These colors are determined with the difference in hue saturation value (HSV), which are specified in Matlab with lb\_hsv and ub\_hsv. These values can be tuned with **Color Tresholder** application in Matlab.

**StateFlow**

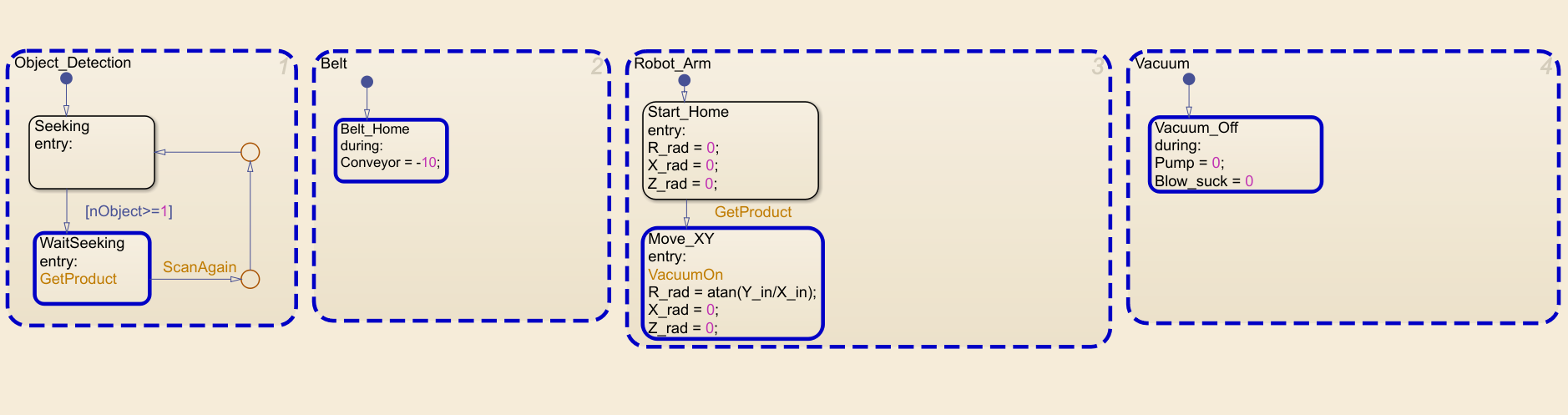
StateFlow is a useful tool in Simulink to make a logic controller of the robot together with the conveyor belt, and vacuum is given here, which should be elaborated/adjusted such that the proposed goal can be reached.

To use StateFlow in Simulink the **StateFlow** toolbox should be downloaded.



Where different blocks run parallel to each other, if an object is detected the robot , vacuum and conveyor belt should react to this. Use this model as starting point and elaborate on this to achieve good results.





**Control Robot**



Finally, the robot itself is controlled in this control loop, where the reference, controller, and feedforward can be adjusted. The angle of each specific axis can be displayed or saved with ToFile block (see Sperte Manual).

The kind off feedback controllers can be chosen by typing DC tools at the matlab prompt.

**Boundaries**

For safety reasons, boundaries of the robot are defined. For each motor, these are defined as follows

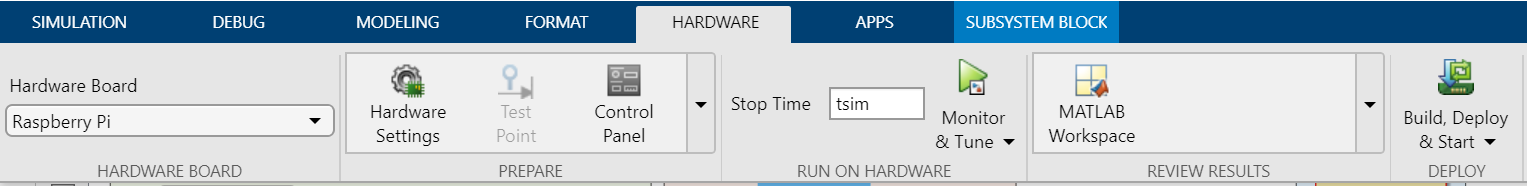
-1< R < 2.5 [rad]

-0.5 < X < 0.5 [rad]

Height of conveyor belt or table < Z < 0.20 [rad]

The robot knows when it is on top of the conveyor belt or on top of the table, which results in different heights.

**How to control the Robot**

1. Open Matlab
2. Plugin camera en raspberry (plug camera in power usb)
3. Connect raspberry pi with p = raspi('10.55.0.1','pi','sperte123').
4. Open Simulink script (Robot\_Student\_2020\_v1.slx) or (Robot\_Student\_2021\_v1)
5. Rename model to own modelname.
6. Initialize model met topleft button.
7. Fill in the robot number.
8. Optional run camera.m in matlab.
9. Make changes to Simulink model before running.
10. In hardware push Monitor and Tune.  
    
11. Wait a minute before robot is homed
12. Parameters in Simulink script could be changed while running the model
13. Stop the model with the stop button in Hardware of simulation  
    