UNIVERSIDAD DE OVIEDO

INTERNATIONAL POSTGRADUATE CENTER

MASTER IN MECHATRONICS ENGINEERING (EU4M)

MASTER'S THESIS

VISION GUIDED 6-AXIS ROBOTIC ARM FOR INSPECTION ON A CONVEYOR LINE

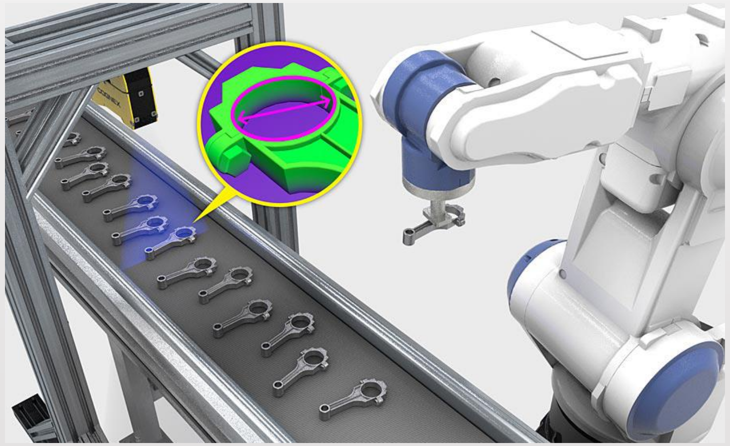
BIBEK GUPTA

JULY 2024

Contents

Background

- Vision-guided robots plays a pivotal role in modern industrial automation . Enhances operational efficiency and accuracy.
- Ability to perform complex tasks such as assembly, quality inspection and material handling.
- A camera mounted on the arm, serving as the "eye of the machine."
- The images are processed to trigger the robots to move and perform specific tasks



Process: Image Capture >> Image Processing >> Response

Objectives

- **Primary:**
  1. Develop and Integrate a Vision-guided robotic system for the purpose of Inspection and Classification.
  2. Design and Integration of a Robotic Cell: Conveyor system with UR3E Robot.
  3. Develop Object-Detection algorithm for classification and pick-place action.

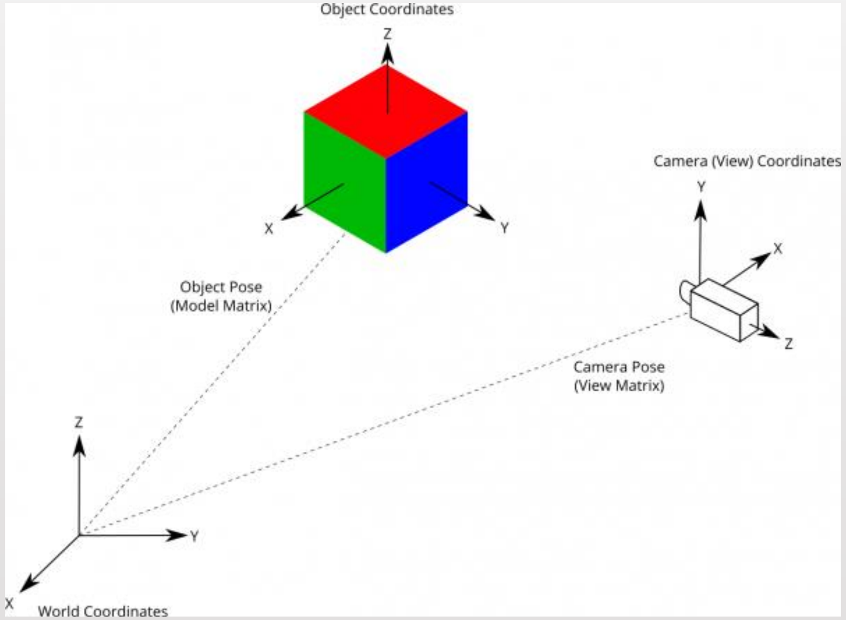
- **Secondary:**
  1. Optimize Camera and Hand-eye calibration for accurate operation.
  2. Improvement of Detection and Inspection algorithm in a dynamic scenario.

Scope of Work

- Conducted in "Grupo de Investigacion SiMuR" laboratory of Universidad de Oviedo.
- Explanation of step-wise calibration method for the Intel camera and robot hand to eye.
- Design, Fabrication and Integration of conveyor with the robot and gripper-camera assembly. (Development of a Digital Twin)
- Three relevant Industrial scenarios are replicated.
- Detection algorithm is run on Conveyor Line with rejections based on:
  1. Object color
  2. Object Size
  3. Deficient Bolts

Coordinate Systems

- Pose of an Object: Describes Position (Tx,Ty,Tz) and Orientation(Rx,Ry,Rz)
- Done by defining a reference frame: Generally called a World Frame
- World Frame defines global coordinate system for the whole scene

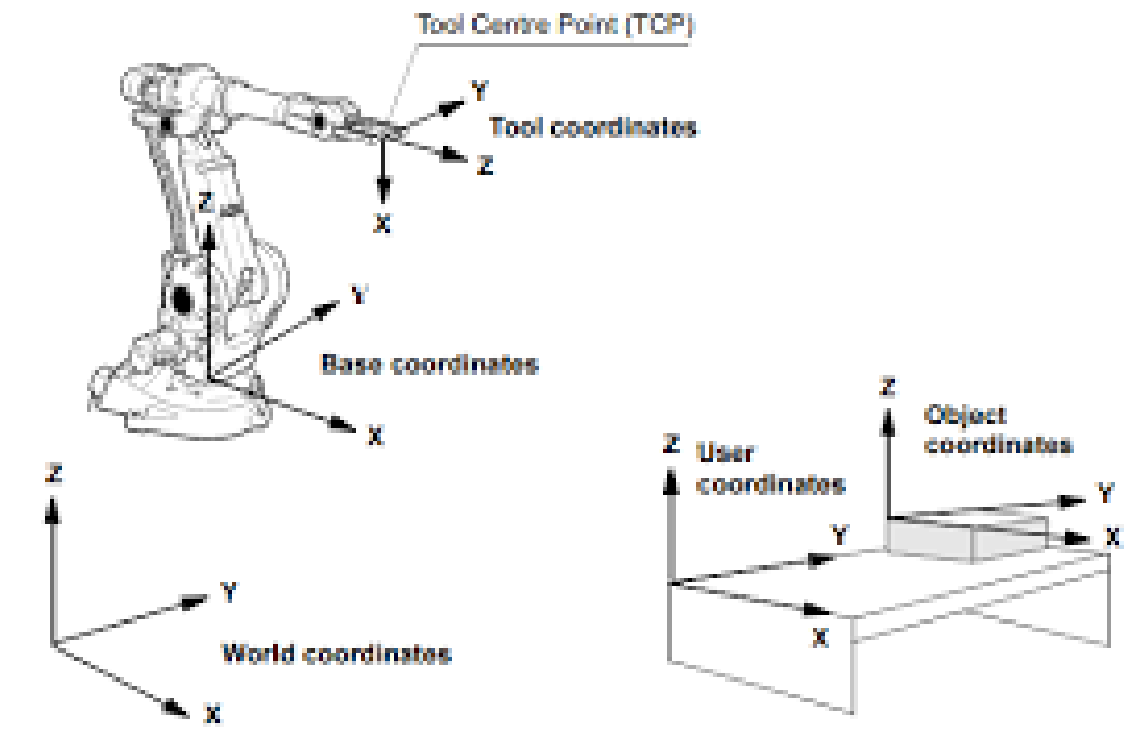


Coordinate Systems

- World Coordinate System (3D)
- Base Coordinate System (3D)
- Tool Frame Coordinate System (3D)
- Camera Coordinate System (3D)

  (Camera Extrinsics)
- Object Coordiante System (3D)
- Normalized Image Coordinate System (3D)

  (Camera Intrinsics)
- Pixel Coordinate System (2D)



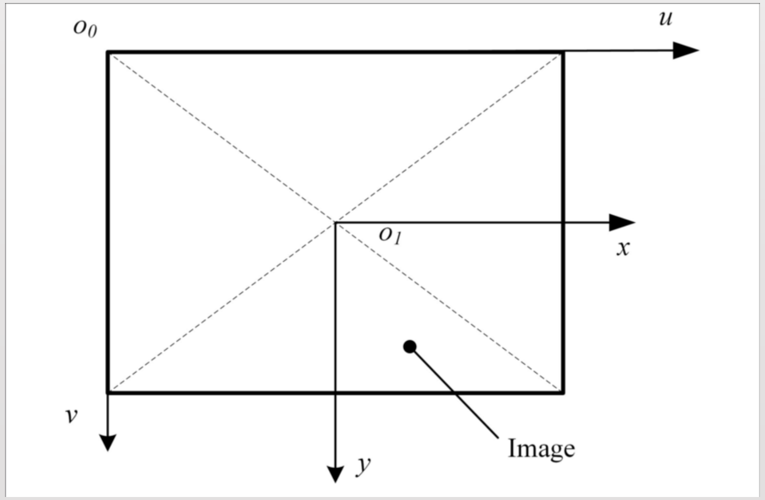


Image Origin: (0,0)

Image Point: (u,v)

Coordinate Transformation

$$X = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Fundamentals in Robotics: Rotation and Translation Operation.
- Homogeneous Matrix:

  A geometric object X can be represented in Homogeneous matrix if X and λX represent the same object for λ≠0.

In 2D : Cartesian coordinate: (x,y) >> Homegeneous Coordinate: (x,y,1)

In 3D: Cartesian coordinate: (x,y,z) >> Homegeneous Coordinate: (x,y,z,1)

Why Convert to Homogeneous Coordinates:

Rotation and Scaling operation require only 3 columns. But Translation requires at least 4 columns.

A 4 column matrix cannot be multiplied with a 3D vector. (Rules of matrix multiplication)

So we use 4D homogenous vectors.

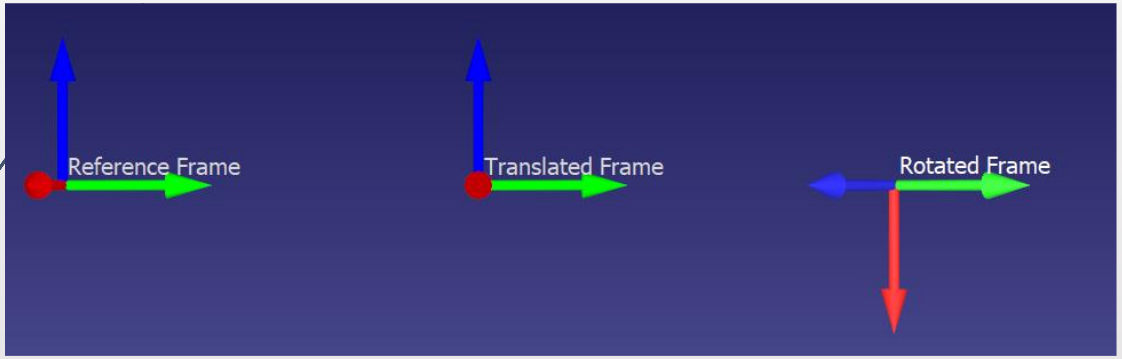
Coordinate Translation and Rotation

- Translation of (X,Y,Z) by a constant vector (tx,ty,tz)

$$\begin{bmatrix} X+t_x \\ Y+t_y \\ Z+t_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Rotation Matrices go to the upper left 3X3 corner of the Translation matrix:

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Translation: (0,500,0)
Translation and Rotation: (0,1000,0) and (90,90,45)

$$R = R_x(\alpha)R_y(\beta)R_z(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & -\cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.4)$$

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

Camera Intrinsics

- The process of estimating the parameters of the camera model
- Distance between the Image plane and the Pinhole : Focal Length
- Using homogenous coordinates for Point P (X,Y,Z)

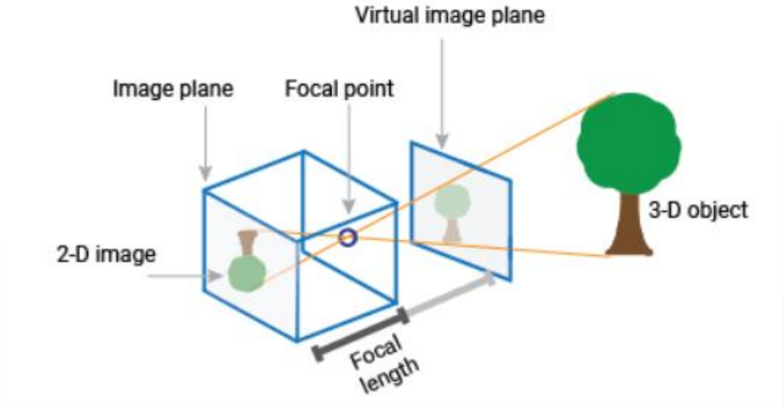
$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & t_u \\ 0 & f & t_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

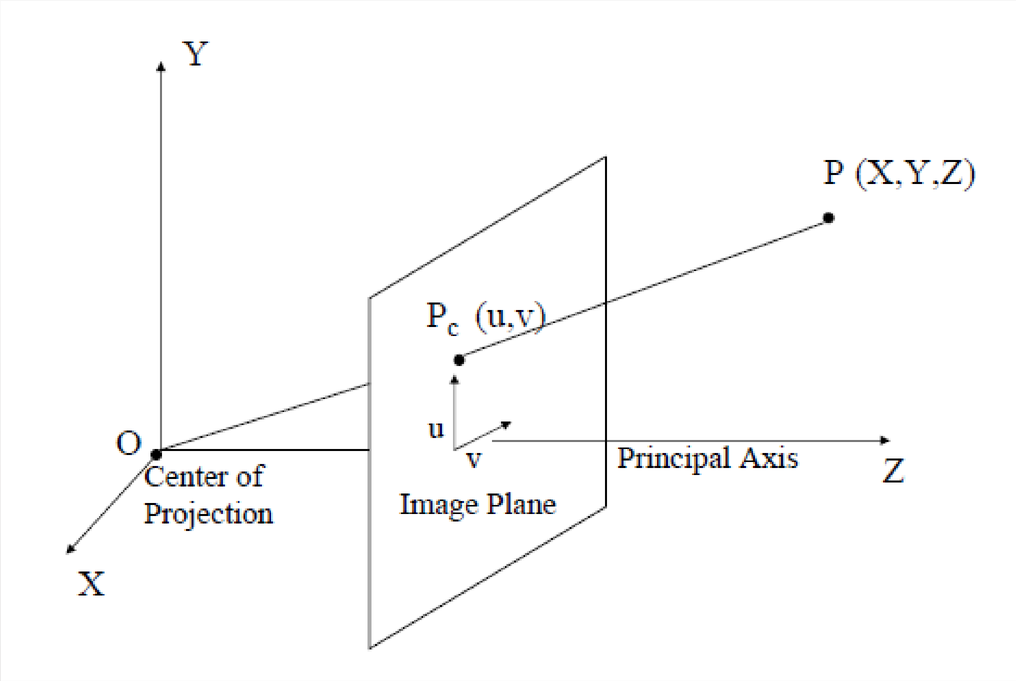
- The Camera Extrinsic Matrix is given as:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

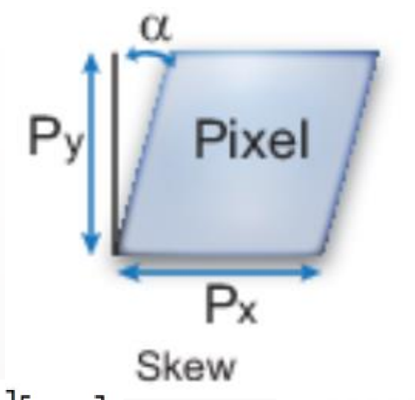
(fx,fy) represent the focal length coordiantes

(Cx,Cy) represents principal or optical axis offset





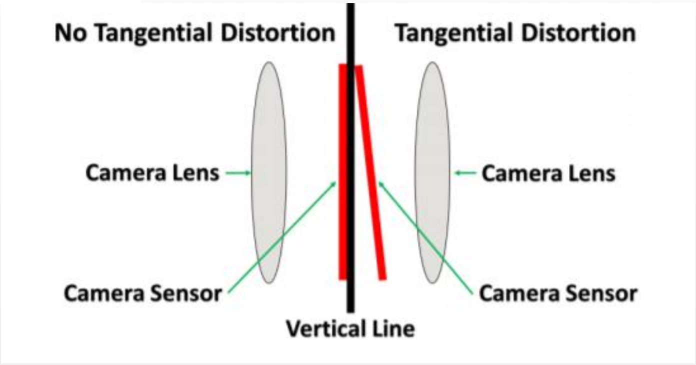
Camera Intrinsics



- Skewness: Camera Coordinate Frame is skewed

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \alpha & -\alpha\cot\theta & c_x & 0 \\ 0 & \dfrac{\beta}{\sin\theta} & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



- Distortion: 2 Types

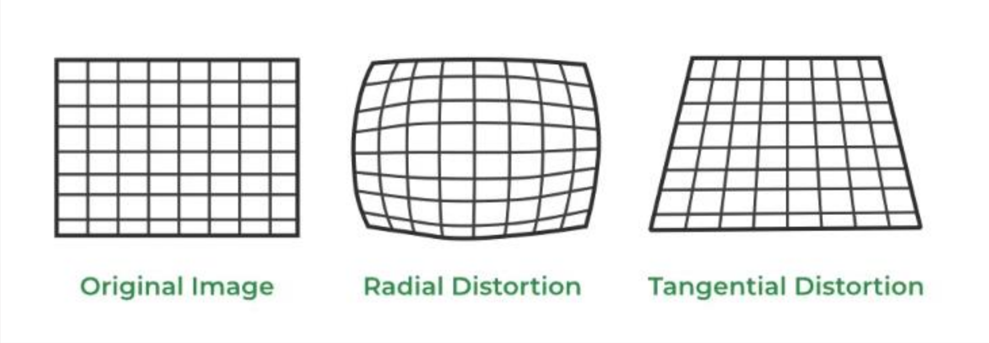
  Radial : Straight lines appears curved. Increases as farther the points are from the center of the image.

  Tangential : Image-taking lens is not aligned perfectly parallel to the imaging plane

  Distortion Coefficients=**[k1 k2 p1 p1 k3]**

  k1,k2,k3 are Radial Distortion Coeff.

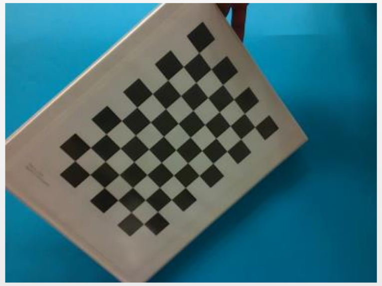
  p1,p2 are Tangential Distortion Coeff.

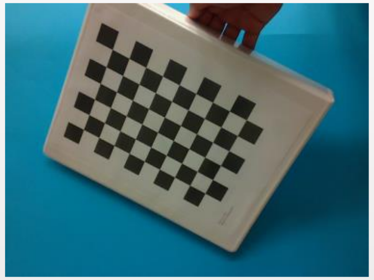


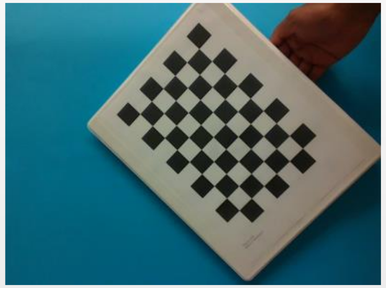
Camera Intrinsics Calibration

- **Camera Mounting**: Fixed on the gripper
- **Chessboard Pattern:** 9X6 Chessboard pattern printed on A4 , 24mm Square size
- **Image Capture:** Total 10 images processed
- **Loading Images and Corner Detection:**
- **Compute Camera Intrinsics Matrix:**

  calibrateCamera(): OpenCV Library function





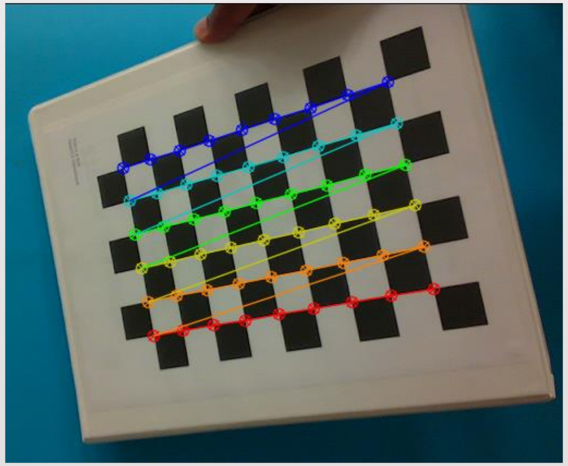


- **Evaluation of calibration Accuracy:**

  Reprojection error:

  - Use of estimated camera parameters to project 3D points into the image plane to get the predicted 2D points

  - Lower Reprojection error indicates the accuracy



Camera Intrinsics Calibration (Results)

- **Camera Intrinsic Matrix:**

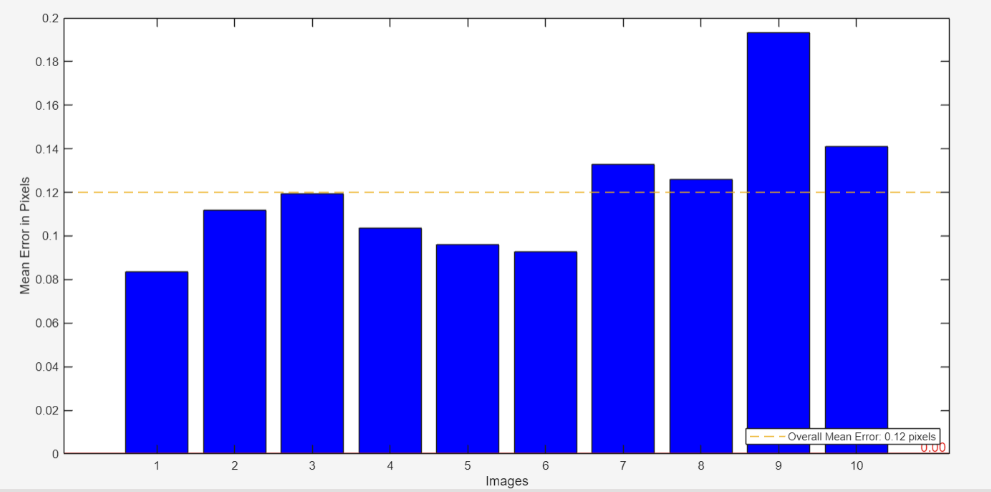
$$K = \begin{bmatrix} 599.51 & 0 & 328.63 \\ 0 & 600.228 & 247.71 \\ 0 & 0 & 1 \end{bmatrix}$$

- **Distortion Coeff. :** [0.04, 0, 0, 0, 0.24]

- **Corrected Intrinsic Matrix:**

$$K = \begin{bmatrix} 599.53 & 0 & 331.04 \\ 0 & 600.37 & 246.21 \\ 0 & 0 & 1 \end{bmatrix}$$

- **Overall RMS Reprojection Error:** 0.12



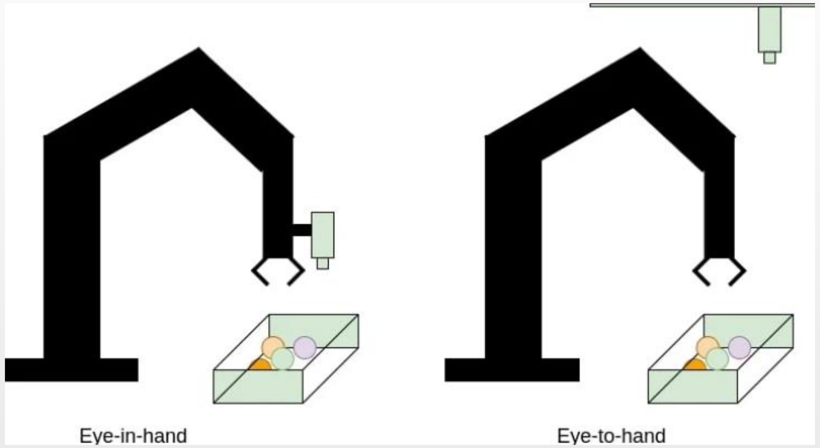
Camera Extrinsics

- Represents the pose of the Camera Eye from the Robot Tool Flange.
- Eye in Hand and Eye to Hand Configuration.
- **Eye in Hand**: Camera rigidly attached to the robot.
- **A 4x4 Transformation matrix:**

  - Relative camera eye rotation and translation matrices

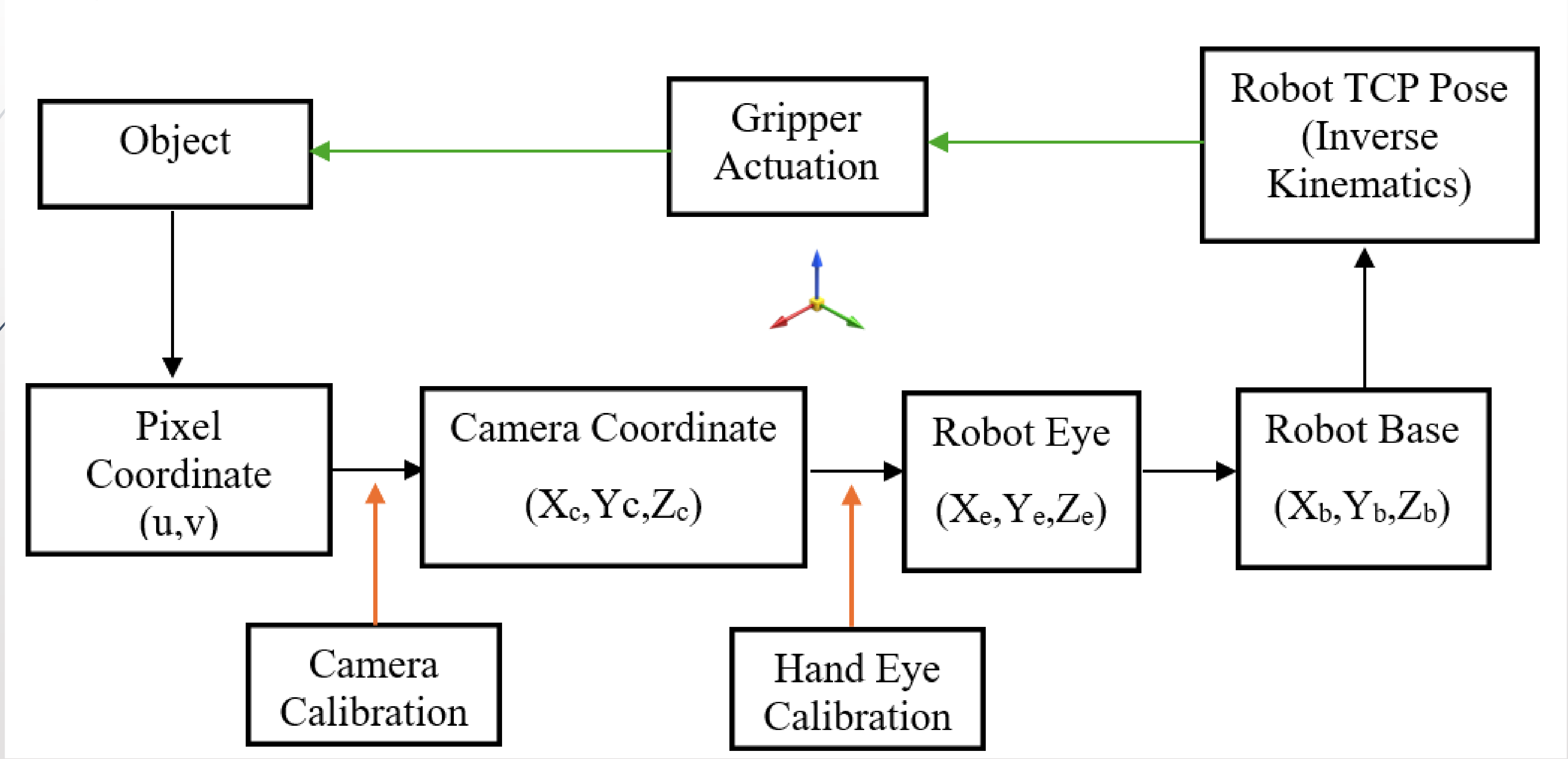
  - With respect to the tool flange of the robot.

  - Converts points from the world coordinate to the camera coordinate frame.



- Changes with the physical location/orientation of the camera.

Why Camera Intrinsics and Extrinsics?



Camera Extrinsics Calibration

- **Design of Digital Twin:** In RoboDK
- **Camera and Robot Setup:**
- **Image Capture and Robot Pose Recording:**

  - The pattern kept at ta fixed position.

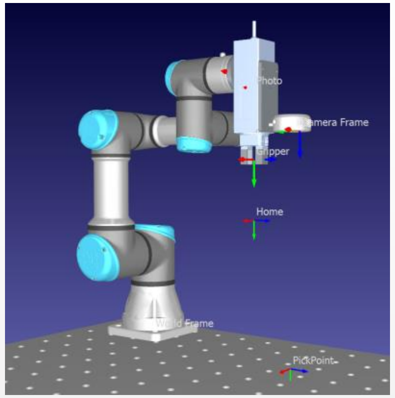
  - Image and the corresponding Pose matrix of the robot recorded.

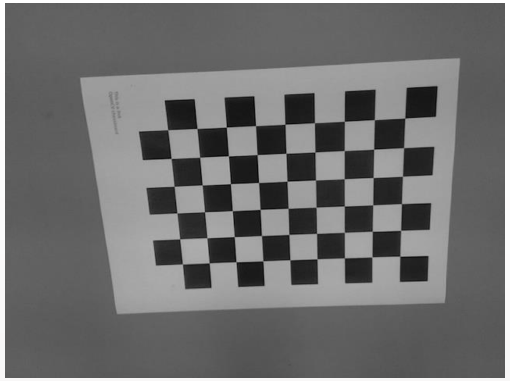
  - Total no. of Samples recorded: 8
- **Pattern Detection:** find_chessboard() OpenCV function

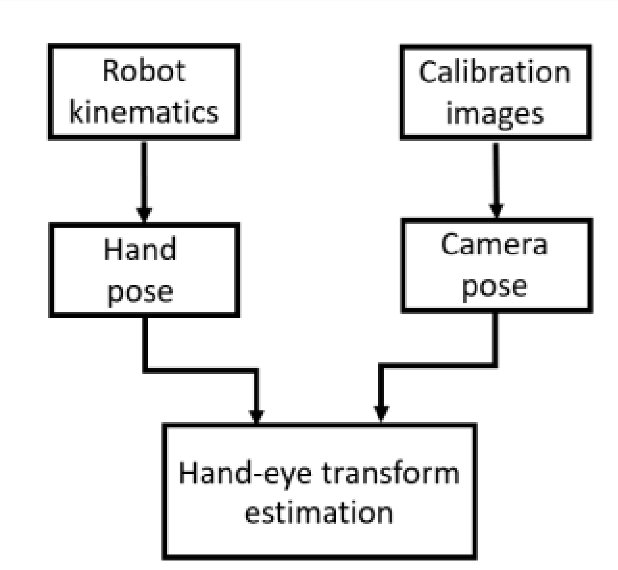
  - returns the Camera Pose for every recorded images.
- **Compute Hand Eye Matrix:**

  - calibrateHandEye() OpenCV function

  - returns the Pose of Camera wrt. to the robot tool flange







$$\begin{bmatrix} 1.000 & -0.005 & -0.019 & -1.766 \\ 0.019 & 0.016 & 1.000 & 103.341 \\ -0.005 & -1.000 & 0.017 & 124.898 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

System Design (Hardware)

- **Universal Robot** : Model: UR3E; Lightweight and compact collaborative robot ;

  6 Axis (Base, Shoulder, Elbow, Wrist 1,2 and 3), Reach : 500mm

- **RGI14 DH Robotics Gripper**: Two jaw moves parallel to each other; Stroke: 14mm; 24VDC

- **Intel RealSense L515 Camera:** RGB resolution: 1920 X 1080; 30fps; 2MP

- **Relay**: Coil-24VDC; Contact- 220VAC

System Design (Hardware)

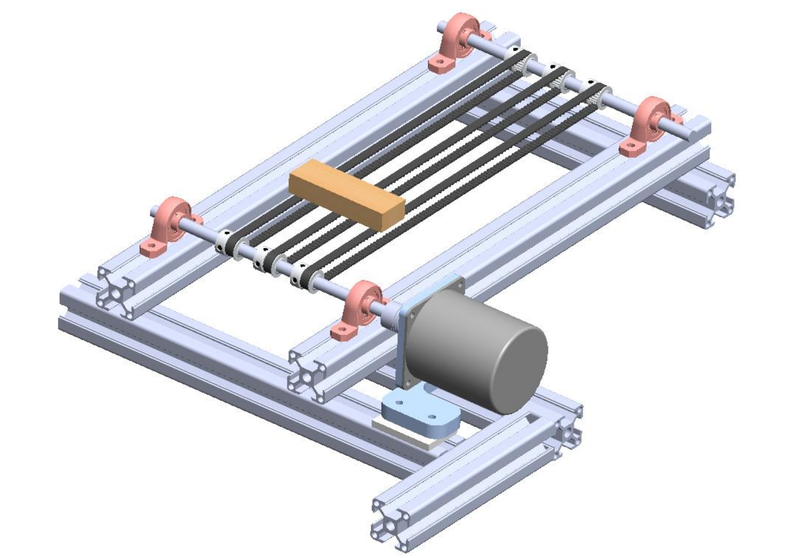
- **Conveyor Model** :

  - Power: Driven by a 500W DC powered motor 220V AC motor

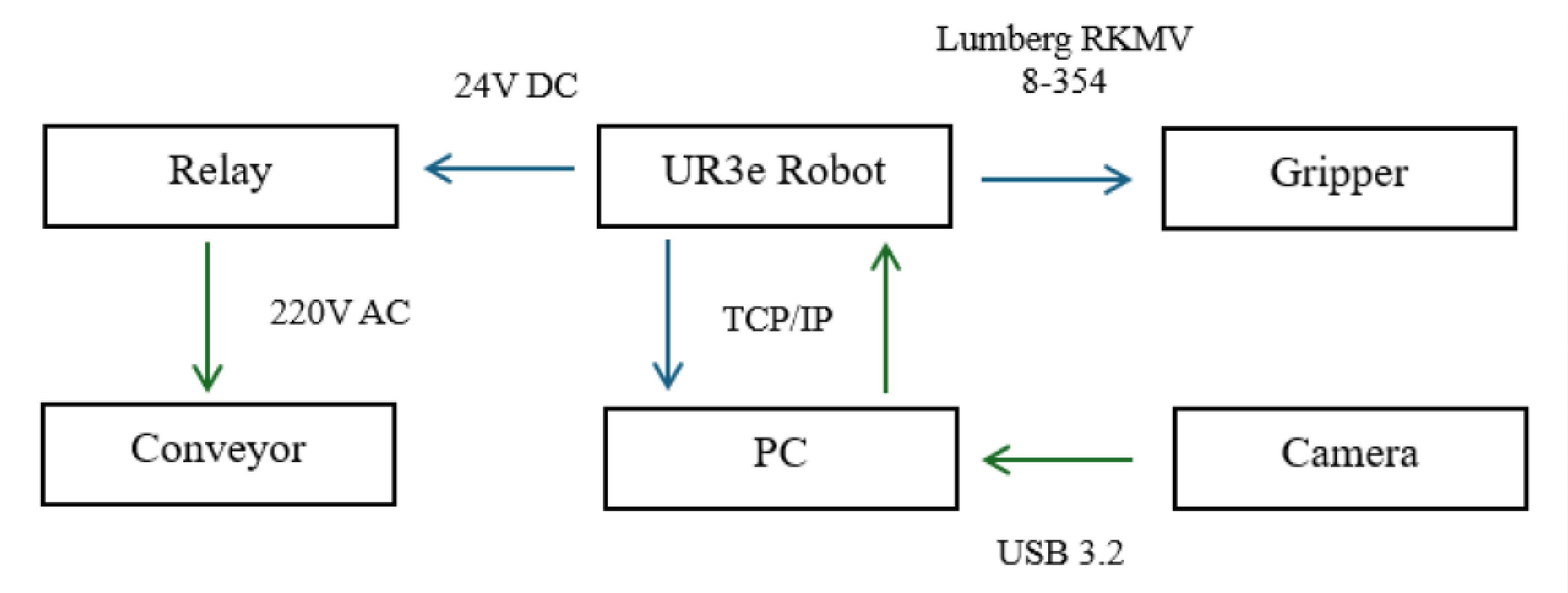
  - Control : 24V DC relay

  - Transport Objects of Dimensions : (75 X 25 X 15) mm

S.N.	Materials	Quantity	Specifications
1	Timing Belt Pulley	6	2GT 20 Teeth 8mm Bore
2	Closed Timing Belt	3	752GT; Width: 6mm
3	Bearing Block	4	608ZZ
4	Flexible Coupling	1	8mm Bore
5	Aluminium Shaft	2	Diameter: 8mm; Length: 200mm
6	Relay Block	1	24V DC Coil/ Contact Points: 220V AC
7	Motor	1	220VAC; 50rpm
8	Aluminium Slotted Profiles	2mtr.	30mm X 30mm
9	Frame Connections	10	30X30 8mm Slot Brackets



System Design (Hardware Integration)



System Design (Software)

- **RoboDK** : Software for robotics and industrial automation applications;

  - Simulation and programming environment for industrial robots;

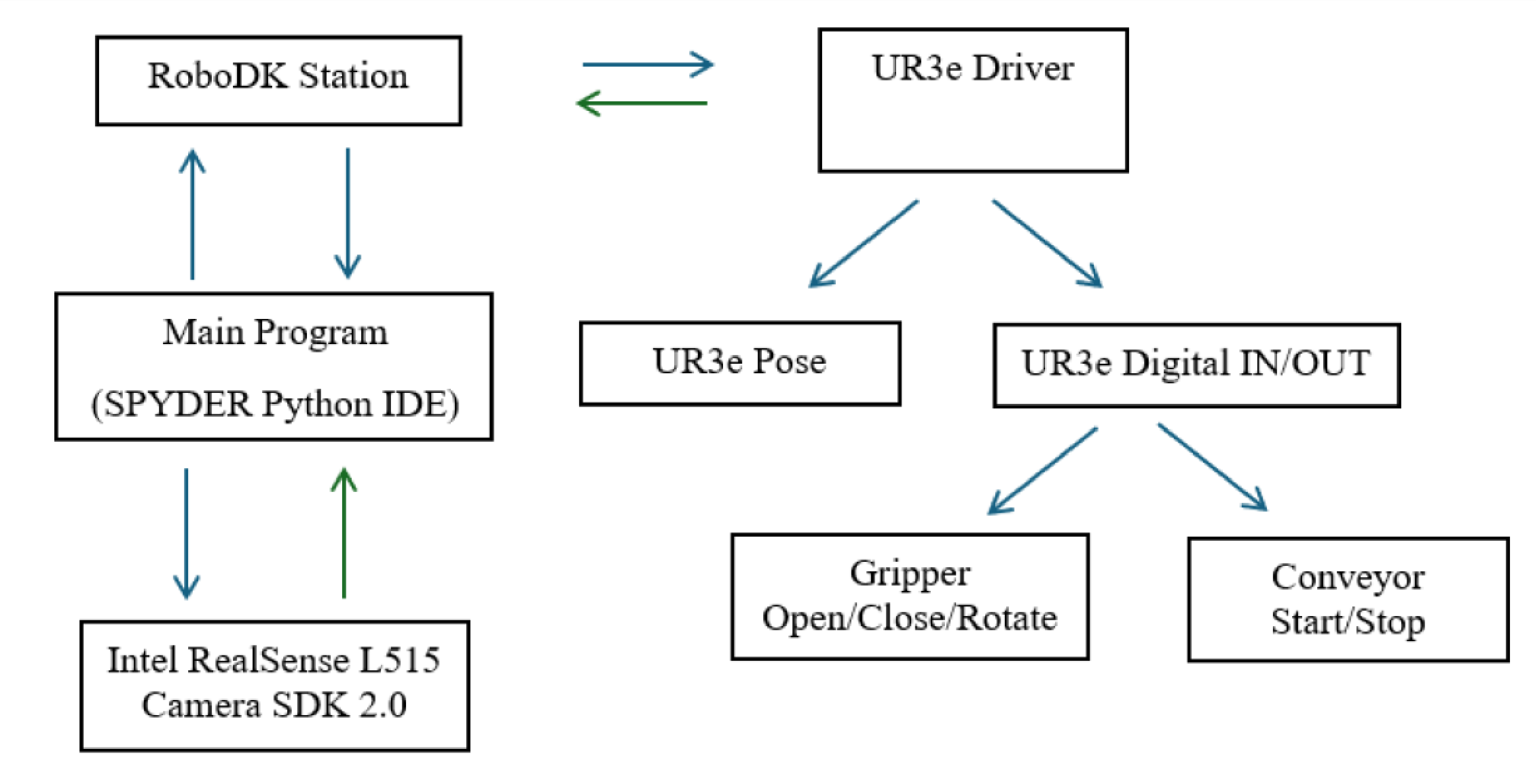
  - Educational License (Universidad de Oviedo)

- **Anaconda Spyder IDE**: Open-source integrated scientific Python programming environment.

- **SolidWorks:** 3D CAD Modelling software (Student Version)

- **MATLAB**: Numerical Computing, Data analysis and algorithm development (Student License)

System Design (Software Integration)



Robolink.py : Main interface to communicate with RoboDK API;
Allows to create, modify and control objects in RoboDK environment.

Robomath.py: Mathematical functions for robotics specific transformations and operations.

Implementation (System Setup)

UR3E Robot Connection to PC:

URScript Commands are send over TCP/IP socket communication.

Connection of Gripper with Robot: Lumberg RKMV 8-354 cable. (Only Power and Digital I/O pins)

Digital IN 1	Digital IN 2	Description
0	0	Gripper Closed
0	1	Gripper Closed and Rotate 90° (Vertical)
1	0	Gripper Open
1	1	Gripper Open and Rotate 90° (Horizontal)

Connection of Intel RealSense camera with PC: USB 3.2 Cable to PC; Image streams at (640,480)

Implementation (System Setup)

Digital Twin Setup:

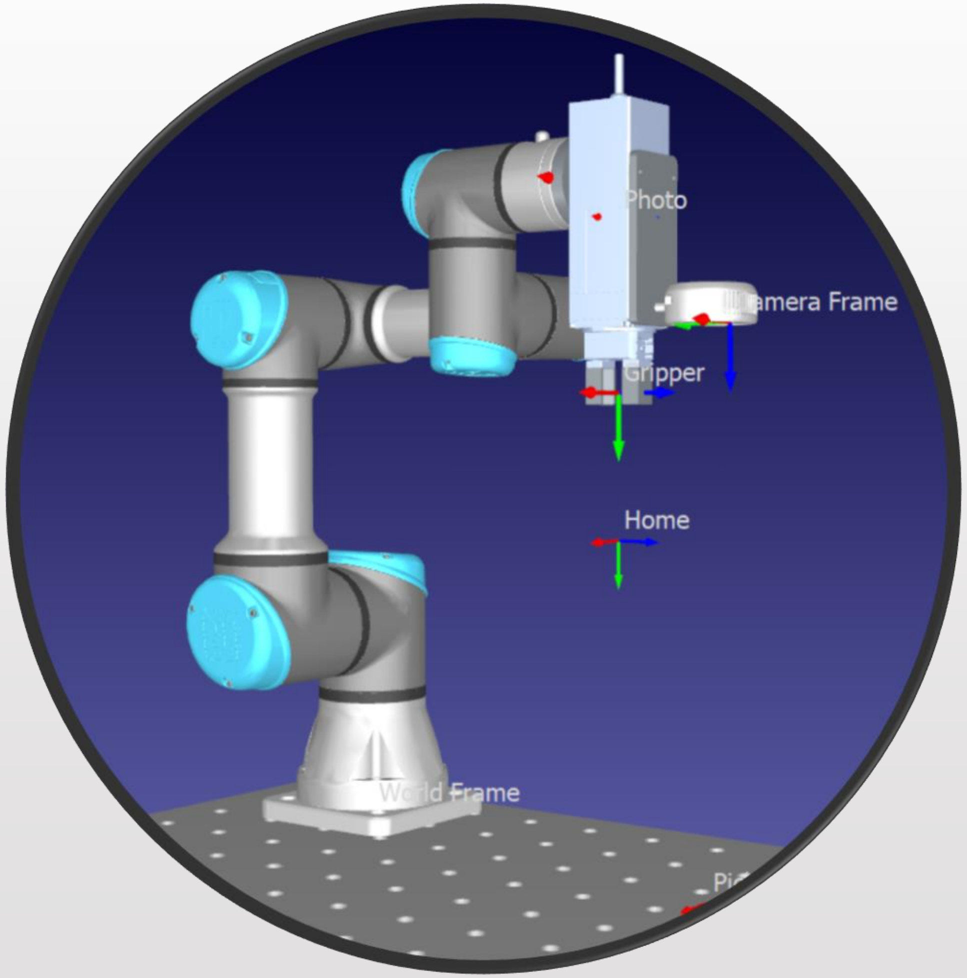
Coordinate Frames: (wrt. World Frame)

1. World Frame: (0,0,0,0,0,0)
2. UR3E Base Frame: (0,0,15,0,0,0)
3. Robot Flange Eye: (305.3, -131.05, 480.8, -69.28, 69.28, -69.28)
4. Gripper Tool Centre Point: (0,145,35,0,0,0)
5. Camera Eye Frame (wrt. Flange Eye) :

   (-1.766, 103.341, 124.898, -89.484, 0.516 , 0.286)

Components: (wrt. World Frame)

1. Base plate: (0,0,0,0,0,0)
2. Table: (250,-250,0,0,0,0)
3. Gripper Camera Assembly (wrt. Flange Eye) : (0,0,0,0,0,0)
4. Conveyor Assembly: (210,175,-220,90,0,0)



Implementation (System Setup)

RoboDK Station Tree:

Targets:

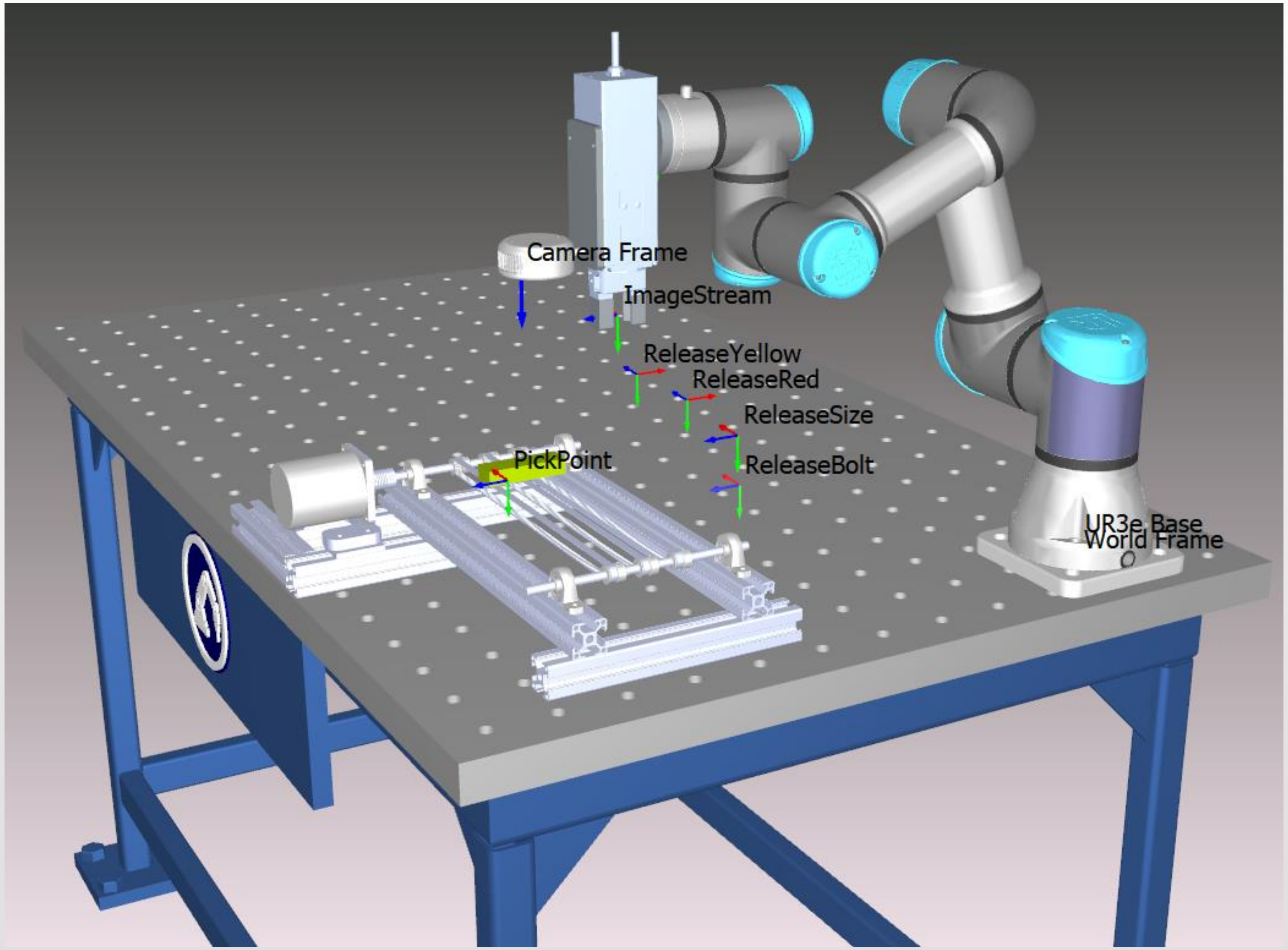
- Specific Points in robot work-space
- Includes parameters such as position, orientation, speed etc.
- Precise and repeatable robot movements

1. Image/Video Capture: [400.30, -131.05, 250.00, -69.28, 69.28, -69.28]
2. Pick-Point: Dynamic target (Calculated from Pixel to World Frame Transformation; target_set())
3. Release Red:
4. Release Yellow:
5. Release Object Size:
6. Release Deficient Bolt:

Implementation (System Setup)

RoboDK Station Tree:

Targets:



Implementation (System Setup)

RoboDK Station Tree:

Programs: movej (X, Y, Z, u, v, w, acceljoints, speedjoints)

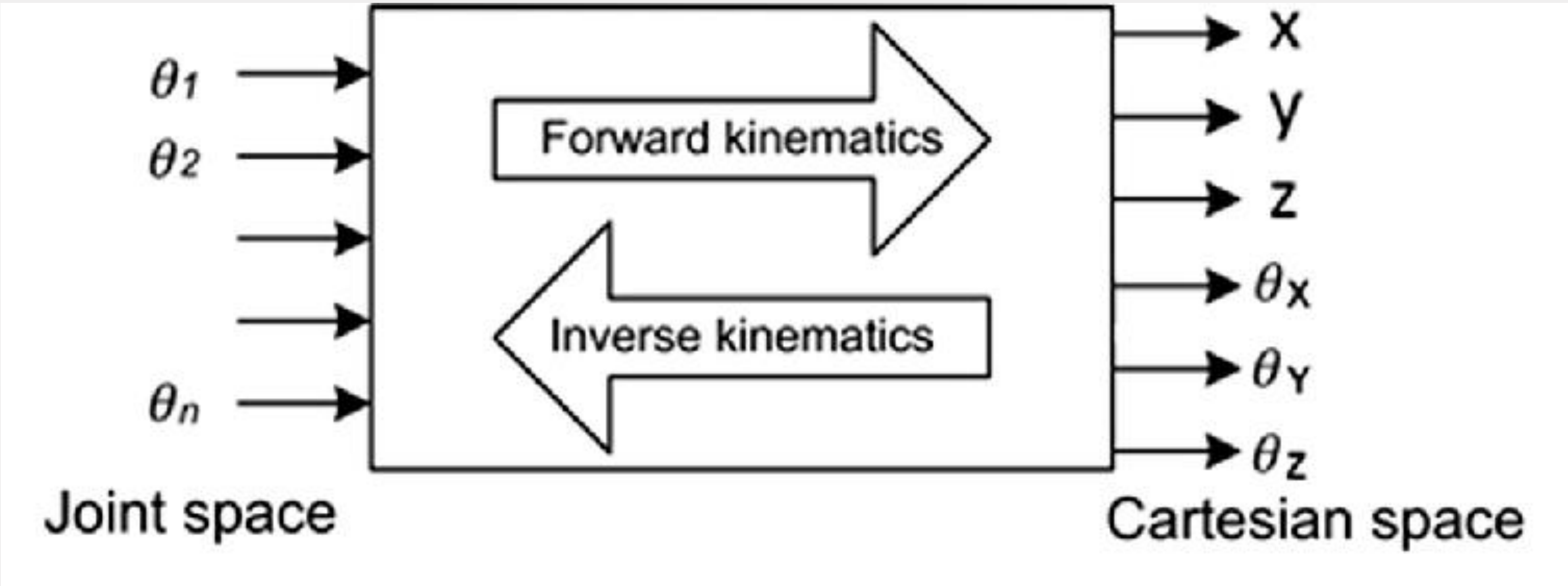
1. Image Capture Position: Moves the TCP to Image capture target
2. Pick_Position: Moves the TCP to calculated target Centroid
3. Gripper Open:
4. Gripper Close:
5. Gripper_TurnH:
6. Gripper_TurnV:
7. Convey_ON:
8. Convey_OFF:

Normal Speed: deg/s and deg/s^2

```
robot_item.setSpeedJoints(200)
robot_item.setAccelerationJoints(350)
```

Pick_Position Speed:

```
robot_item.setSpeedJoints(100)   # Reduce
robot_item.setAccelerationJoints(200)
```



Implementation (Pick and Place Workflow)

target_set(u, v, color, orient): Pixel to World Coordinate Transformation & Robot Movements

Arguments: (u, v) : Pixel Coordinates (Centroid); **color:** [Red/Yellow/NOI/Bolt]; **orient:** [H/V]

Step 1: Convert Image Coordinates to Camera Coordinates (Camera Intrinsic Matrix)

Depth Value (TCP to Object) : 512mm (Table Surface) ; 190mm (Conveyor Belt)

Step 2: Convert Camera Coordinates to World Coordinates (Camera Extrinsic Matrix)

Step 3: Convert World Coordinates to Robot Base Coordinates (Robot Image/Video Pose)

$$Hand\_base\_flange = \begin{bmatrix} 0.000 & 0.000 & 1.000 & 305.299 \\ -1.000 & 0.000 & 0.000 & -131.050 \\ -0.000 & -1.000 & 0.000 & 620 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

target_pos is the (X, Y, Z, -90,0,90) position of the target in the robot's base coordinate system.

Implementation (Robot Movement Workflow)

target_set(u, v, color, orient): Pixel to World Coordinate Transformation & Robot Movements

target_pos is the (X, Y, Z, -90,0,90) position of the target in the robot's base coordinate system.

Step 1: Set the Target Pose for the Robot.

Step 2: Orient and Open Gripper Based on object orientation

Step 3: Close the Gripper to Pick Up the Object

Step 4: Move to Release Position Based on Object Color/Size/ Deficient bolts

Step 5: Open the Gripper to Release the Object

Step 6: Move back to Image/Video Stream Position

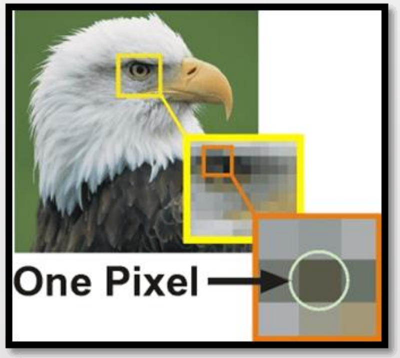
Computer Vision

- **Extracting meaningful information (manipulating and analyzing to extract valuable features)**
- **Filtering and Convolution:**

  - Sliding a kernel over the image and perform mathematical operations

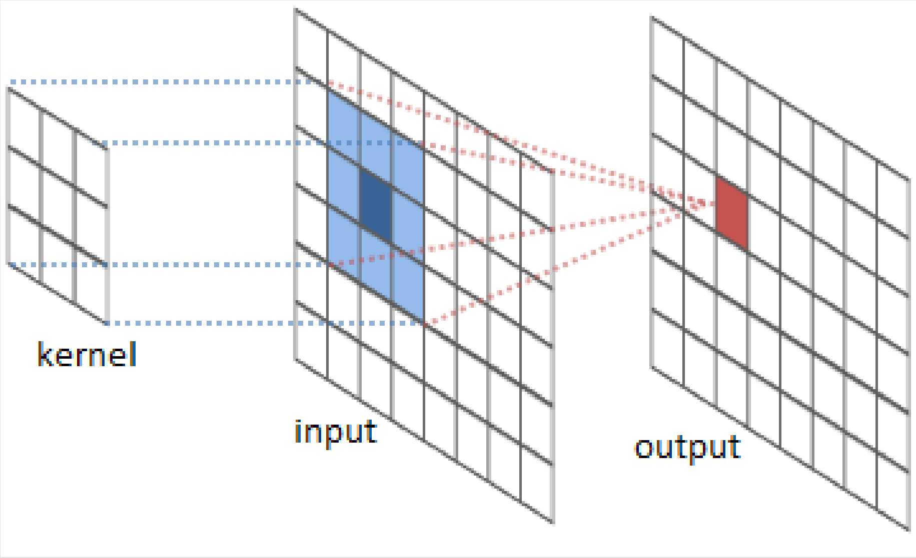
  - Blurring, Sharpening and noise reduction

  - **Gaussian blur**: Reduce noise and detail in an image by averaging the pixel values with their neighbors



$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

3×3 Gaussian Kernel



Computer Vision

- **Color Scales and Conversion:**

  - **Red-Green-Blue:** (R,G,B) where each values range from 0 to 255 .

  - **Grayscale:** Shades of Grey with intensity ranging from 0(Black) to 255 (White)

  - **Blue-Green-Red:** (B,G,R) Inverted RGB.

  - **HSV(Hue, Saturation, Value):**

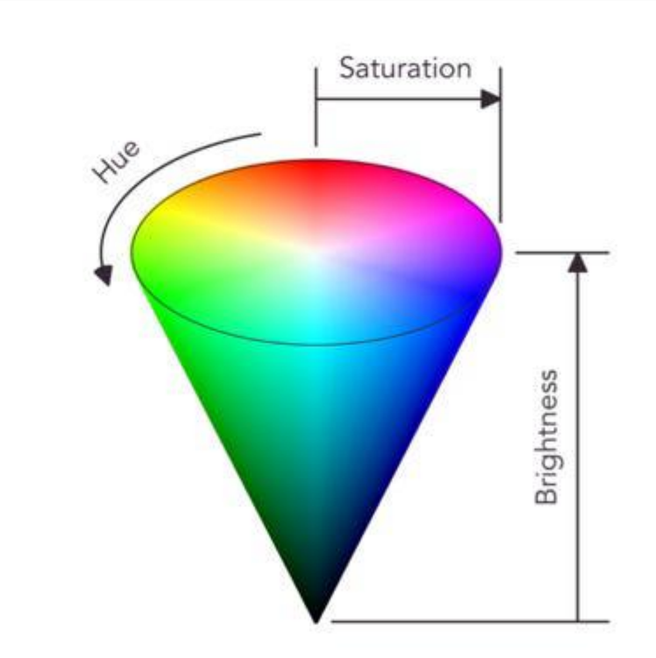
$$Gray = 0.299R + 0.587G + 0.114B$$

- **Hue:** color portion of the model : number from 0-360 deg.

  Red (0,60); Yellow (61, 120); Green(121, 180); Blue(241, 300)

- **Saturation:** Amount of gray in a particular color. (0-100%)

- **Value:** Describes the brightness or intensity of color (0-100%)



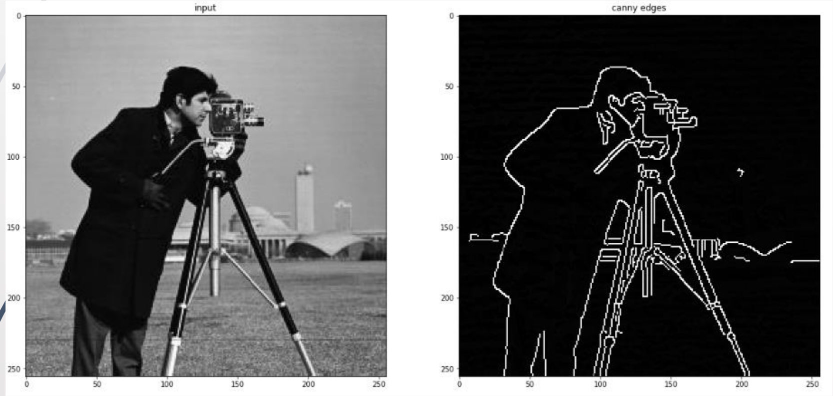
Computer Vision

- **Edge Detection:**

  - Identify image boundaries and transitions (edges, curves or contours)

  - Object Detection, shape analysis and feature extraction.

  - Popular edge detection algorithms: Canny edge detector and Sobel operator.



- **Thresholding:**

  - Conversion of colored/grayscale image to binary image (0/255).

  - A binary mask of 0 and 1 applied to the original image to extract Region of interest.

  -

Computer Vision

- **Color Masking:**

  - Defining a range of colors in the image to extract and conversion of image into a binary format

  - Pixels that fall within the specified color range are set to 1.

  - Pixels that don't fall within the specified range are set to 0.

  - Color mask is applied to the image by multiplying the image by the binary mask.

- **RGB to HSV:**

  - Efficient for Image Segmentation

  - RGB images best suited for simple colors ; HSV image color space better to segment complex colors.

Implementation (Image Processing Workflow)

Step 1: Image Capture ; Size: (640X480)

Step 2: Conversion to HSV Color Scale

Step 3: Color Mask to retrieve the Object Profile

$$Lower\_blue = (32, 50, 80); \quad Upper\_blue = (140, 255, 255)$$

Step 4: Conversion to GrayScale

Step 5 : Inverse Thresholding (250,255)

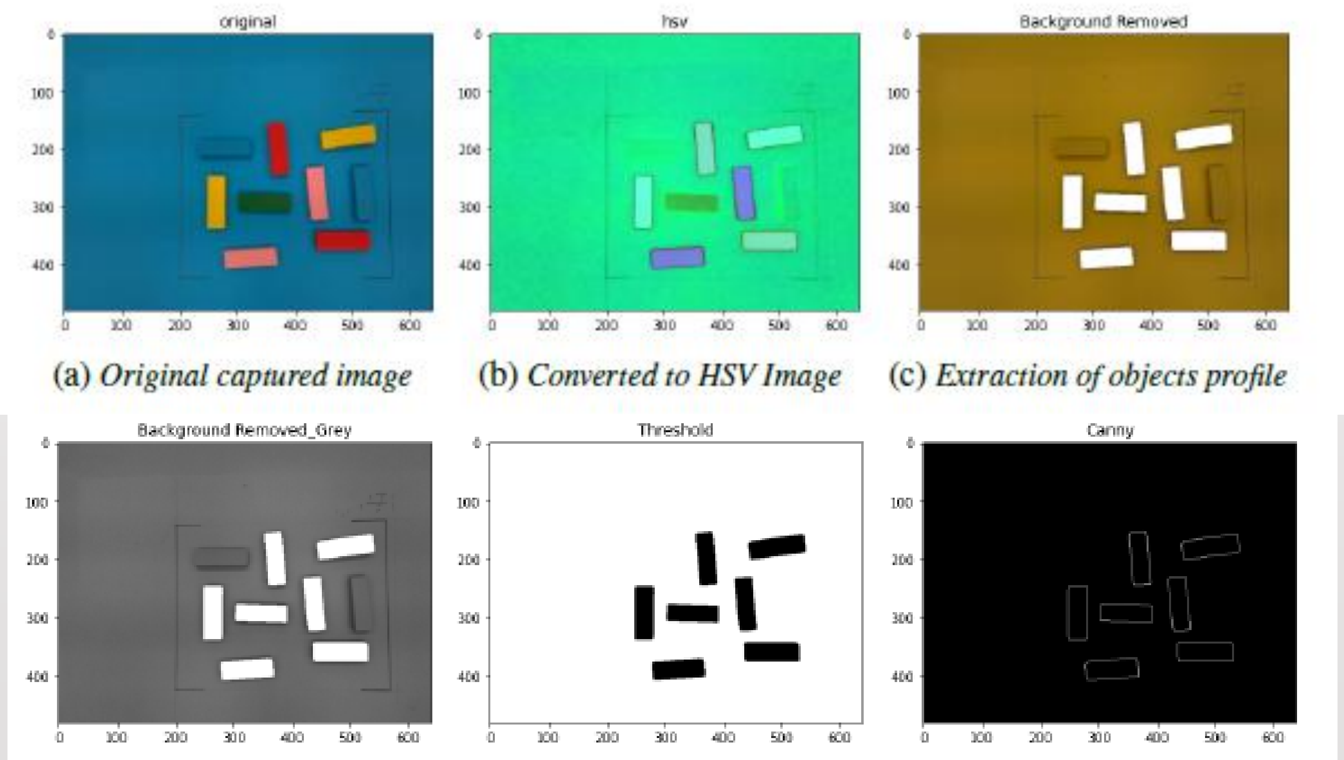
Step 6: Gaussian Blur

Step 7: Canny Edge Detection

Step 8 : Rectangular Contour Detection

Step 9 : Filter out contou

Step 9: Centroid and Aspect Ratio Calculation



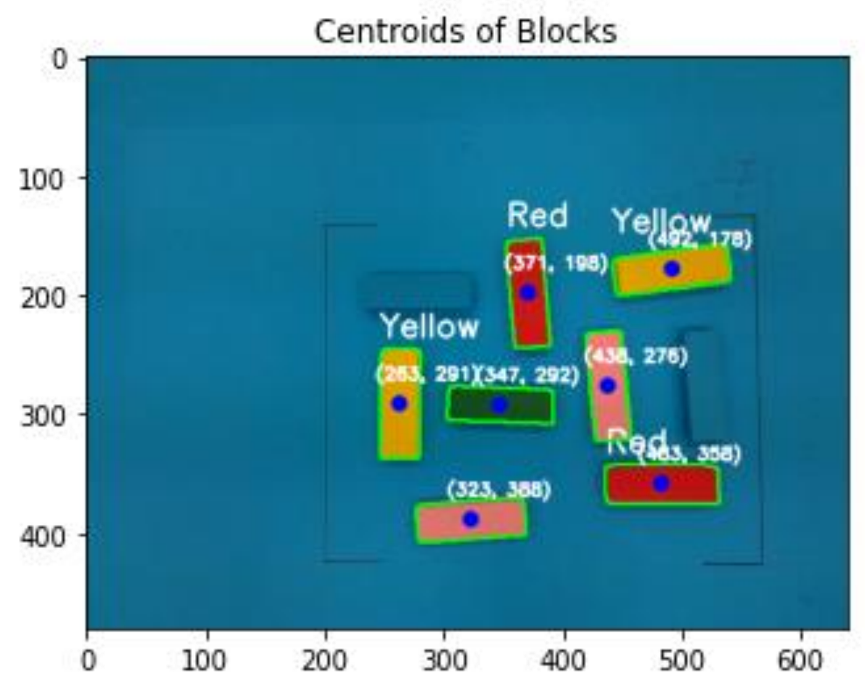
(a) *Original captured image* (b) *Converted to HSV Image* (c) *Extraction of objects profile*

(a) *Objects profile grayed* (b) *Inverse Thresholded Image* (c) *Canny Edge Detection*

Implementation (Image Processing Workflow)

Step 10: Average RGB value Calculation of ROI (Region of Interest)

Step 11: Centroid, color and orient passed to target_set ()



$$For\ Color\ Red:\ (RGB\ value) = (R \geq 100, G \leq 50, B \leq 50)$$
$$For\ Color\ Yellow:\ (RGB\ value) = (R \geq 100, G \geq 100, B \leq 50)$$

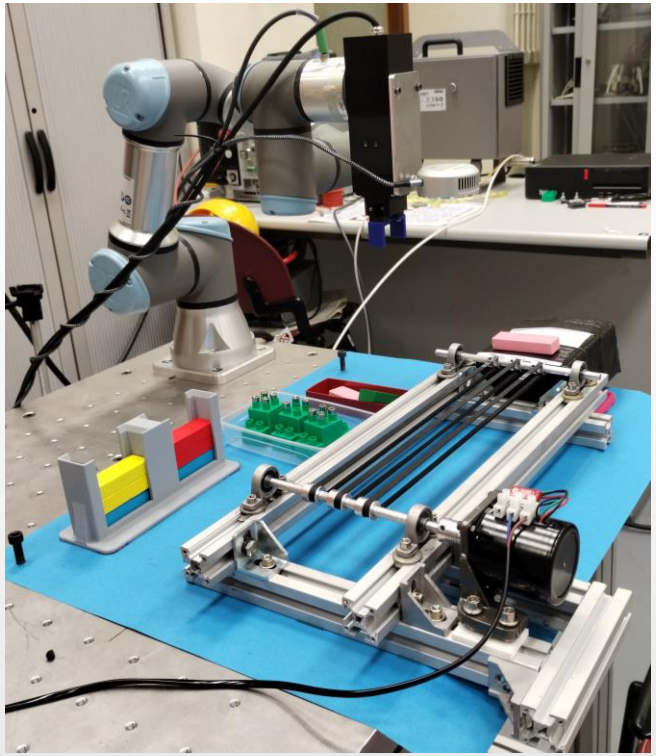
S.N.	Centroids	Color	Orientation	Average RGB	Target Coordinates (wrt UR3e Base)
1	(483,358)	Red	H	(167,26,28)	(344.16,-250.90,0.64)
2	(438,276)	Light Pink	V	(182,107,108)	NOI
3	(492,178)	Yellow	H	(191,146,12)	(497.66,-259.35,2.95)
4	(371,198)	Red	V	(155,40,47)	(481.12,-155.93,4.65)
5	(347,292)	Dark Green	H	(18,77,40)	NOI
6	(323,388)	Light Pink	H	(188,117,119)	NOI
7	(263,291)	Yellow	V	(146,133,39)	(402.25,-63.30,5.13)

Table 4.3. Centroids and Targets for detected objects

target_pos is the (X, Y, Z, -90,0,90)

Inspection on Conveyor

- **Actual Setup of the System**



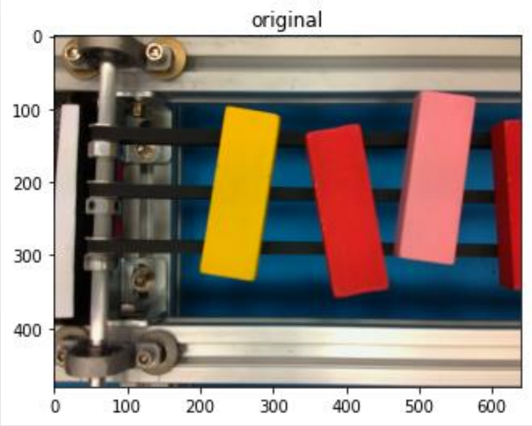
- **Configuration Setup:**

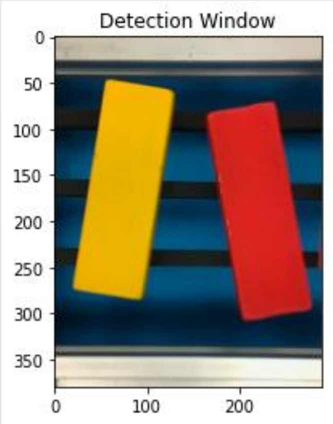
Video_ACK_PRCS = True
Conv_Move = True
Robot_Move = True

- **Scenario Setup:**

Program_RedYellow = True/False
Program_Object_Size = True/False
Program_Missing_Bolt = True/False

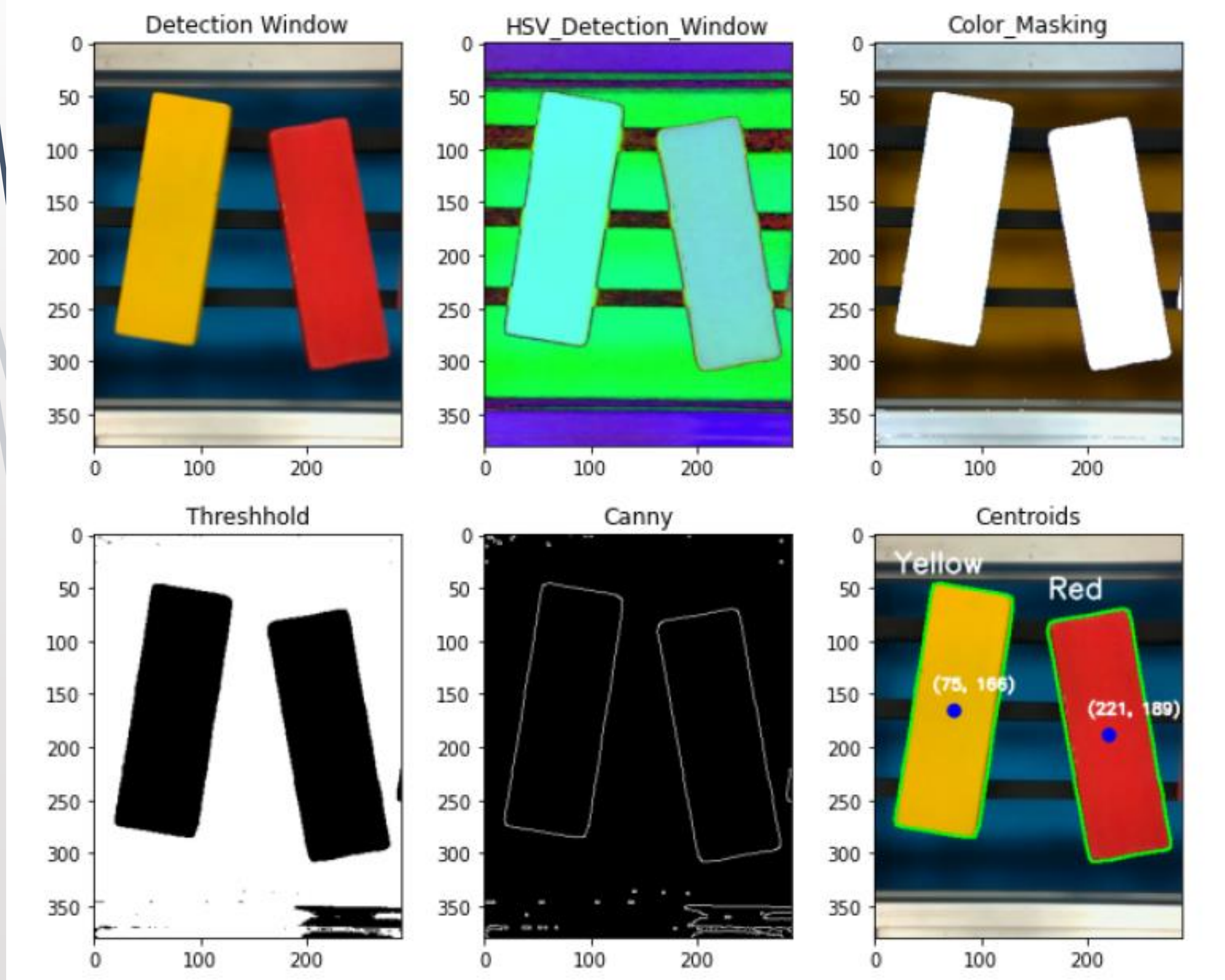
- **Original Image Size:** (640 X 480)
- **Detection Window:** (280 X 380)



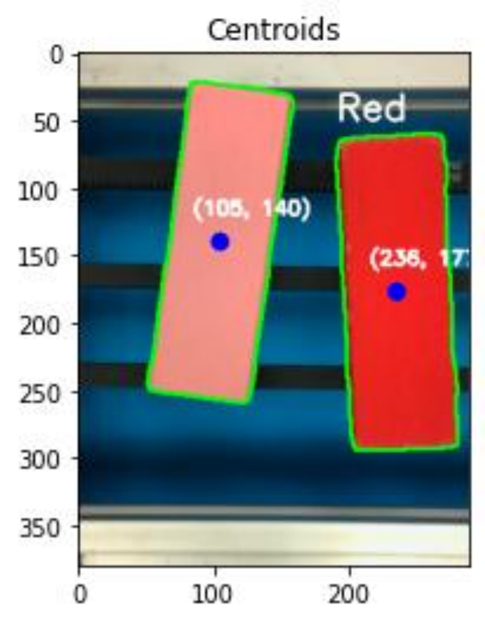


Inspection on Conveyor (Red/Yellow Objects)

- The program picks up only the red and yellow objects from the conveyor and allows other colored objects to pass through.

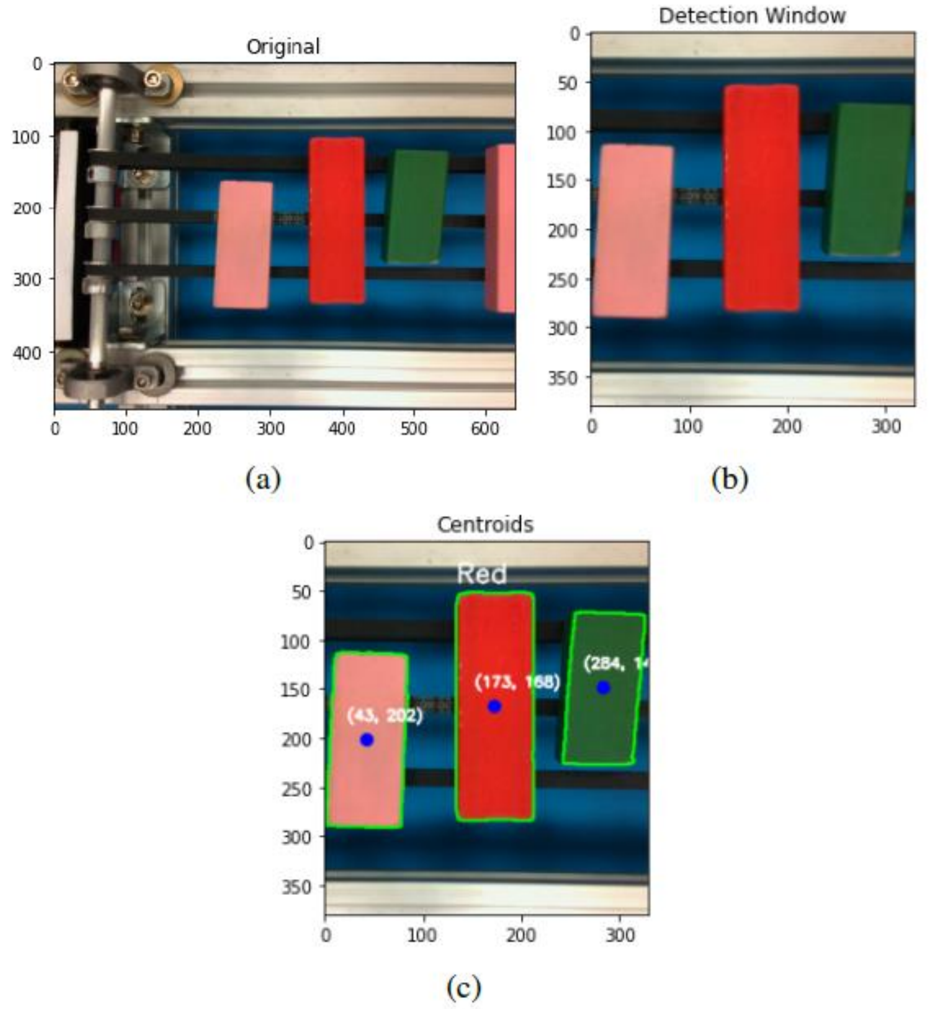


Program_RedYellow = True
Program_Object_Size = False
Program_Missing_Bolt = False



Inspection on Conveyor (Based on Object-Size)

- The program rejects objects less than the standard size objects on the conveyor.



a) Original Image b) Cropped Image c) Centroid and Object Detection based on object size

Program_RedYellow = False
Program_Object_Size = True
Program_Missing_Bolt = False

Standard Contour area of objects: ≈ 18000
Rejection criteria for Contour area: ≤ 16000

Objects Rejected:

- Centroid coordinate : (43,202); Contour Area: 13357.5
- Centroid coordinate : (284,148); Contour Area: 11045.0

Inspection on Conveyor (Based on Deficient Bolts)

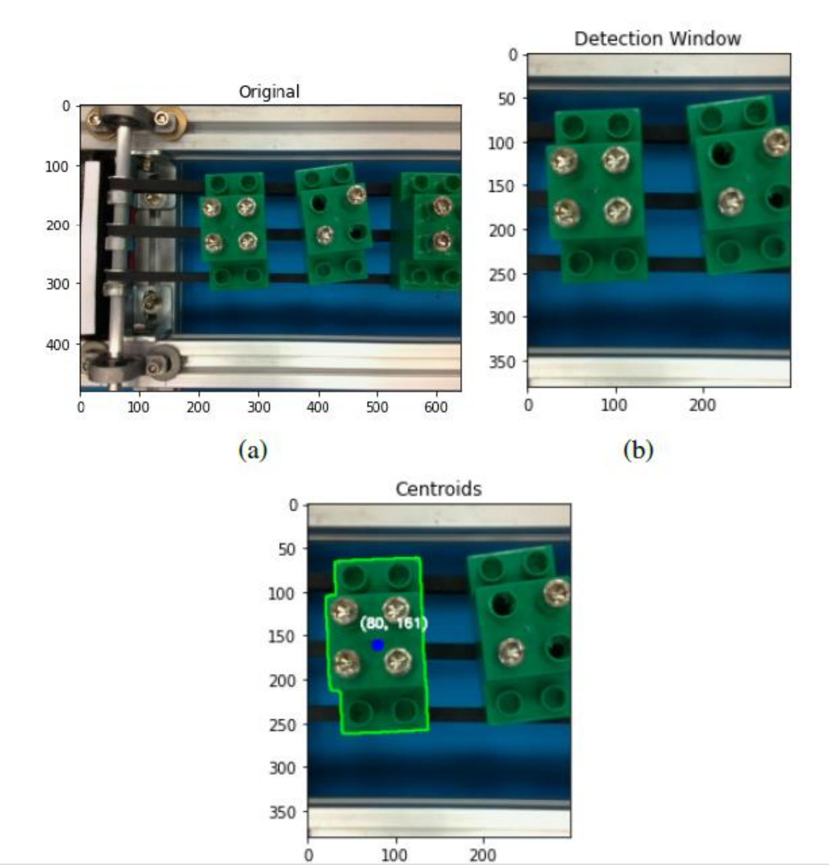
- The program rejects objects fitted with **less than 4 bolts** on the conveyor.

Program_RedYellow = False
Program_Object_Size = False
Program_Missing_Bolt = True

- **Phase 1 :** Object Detection and Centroid Calculation :
  - Using the previously explained algorithm
- **Phase 2:** Bolt Detection Image Processing algorithm:
  - search_bolt() function

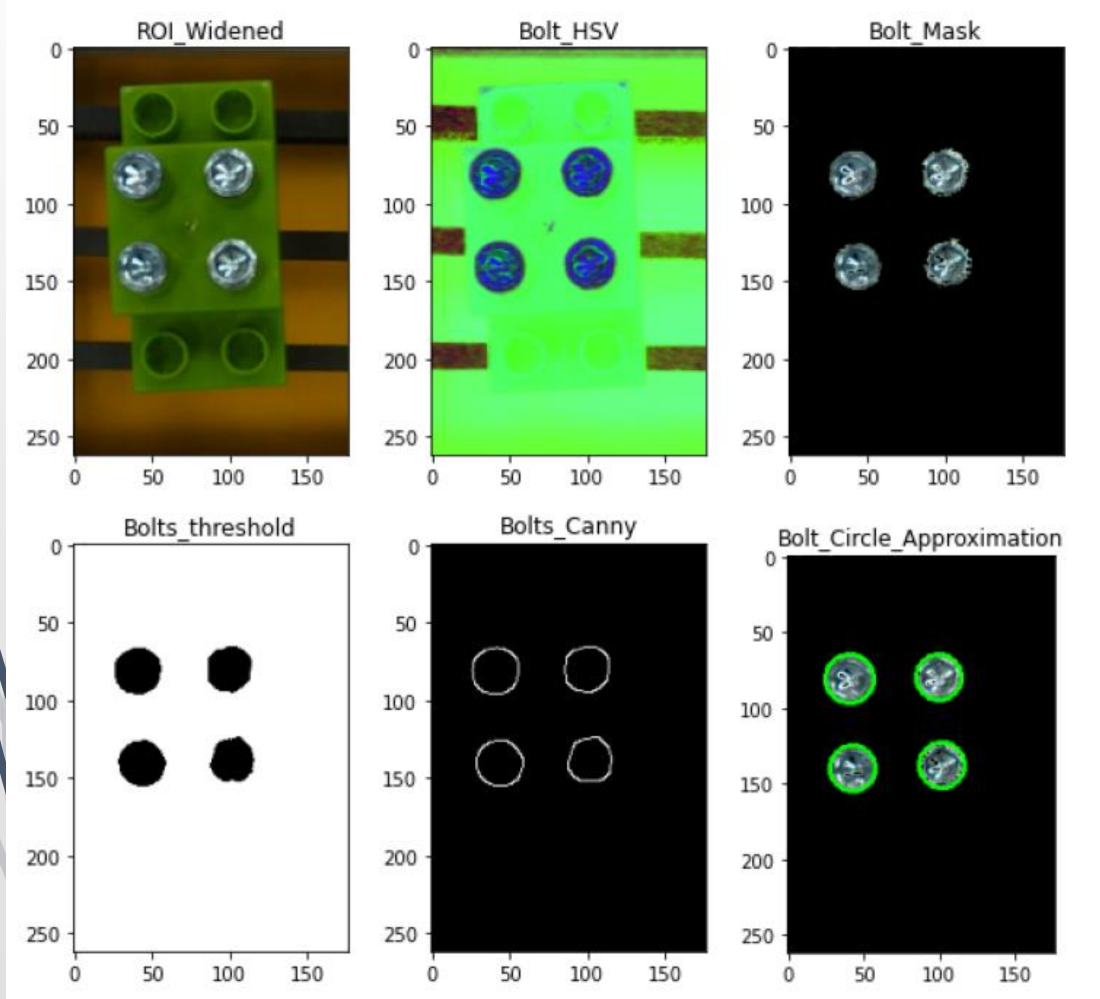
Different Features:

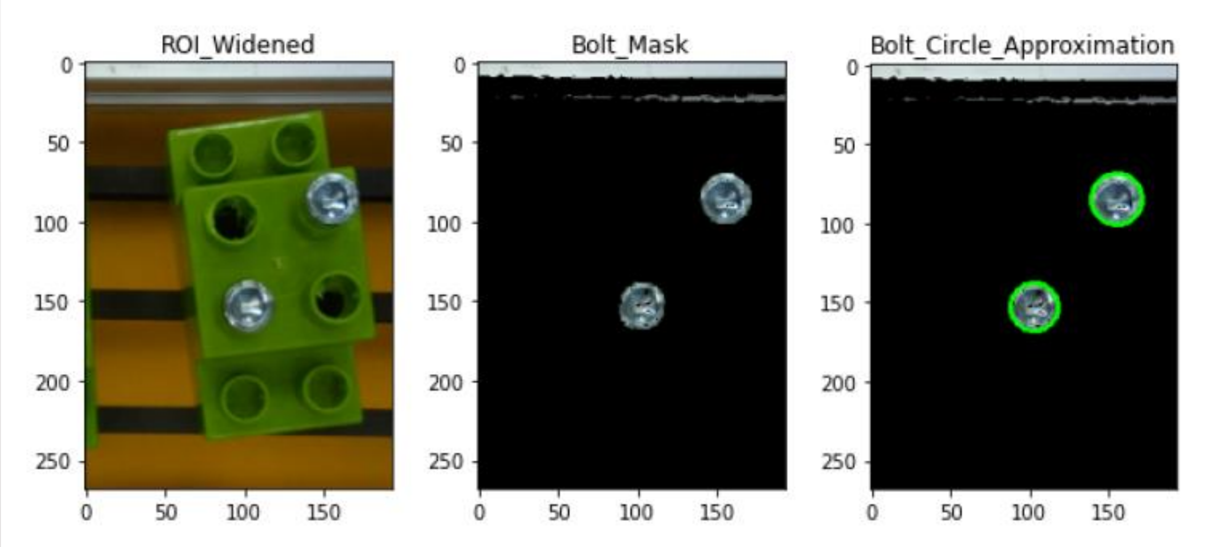
- Widened ROI :
- Mask : $Lower\_silver = (32,50,80); Upper\_silver = (140,255,255)$
- Histogram Equalization: Increase Contrast and Visibility (Redistribute Pixel Values)
- Median Blurring: Preserve Edges and remove noise
- Circle Approximation: Radius (12,20) circular contours-approximated and drawn..



Inspection on Conveyor (Based on Deficient Bolts)

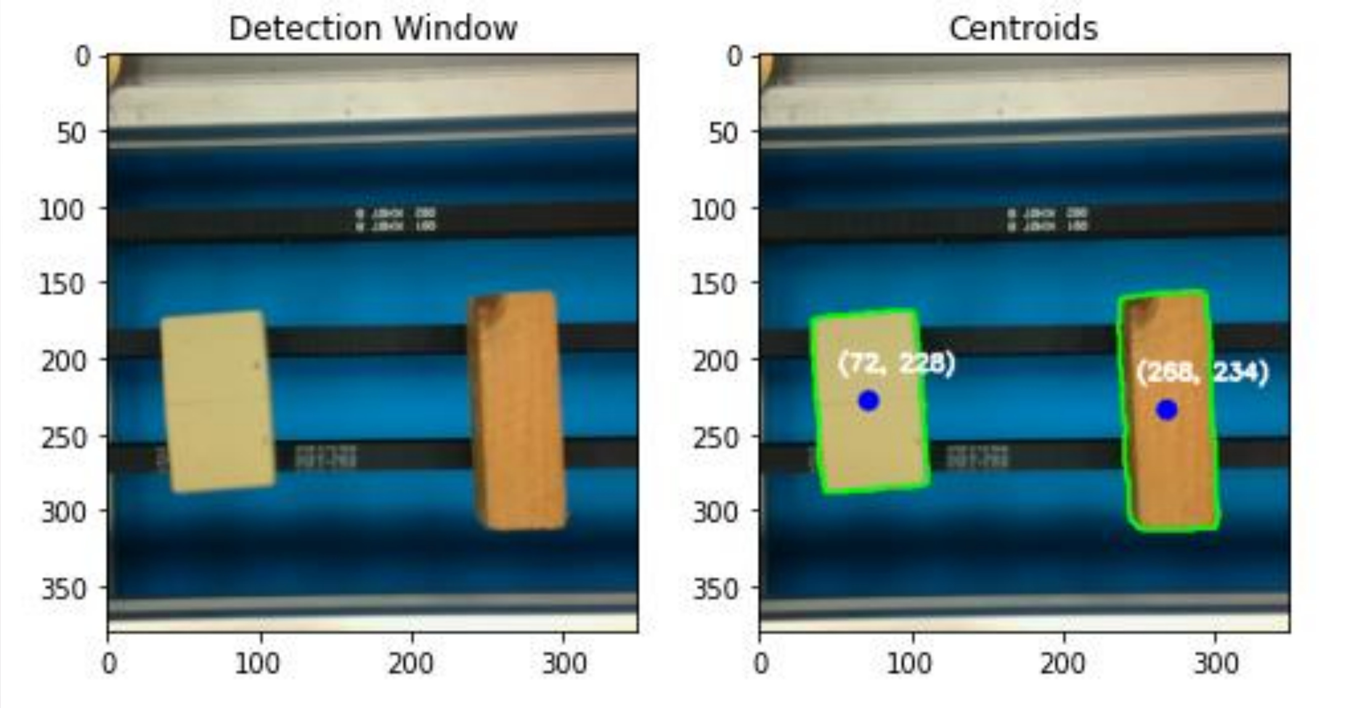
- The program rejects objects fitted with **less than 4 bolts** on the conveyor.
- **Phase 2:** Bolt Detection Image Processing algorithm: search_bolt() function





Inspection on Conveyor (Discussion)

- **Different Rectangular Objects**
- Algorithm tested on different size and colored objects to analyze the detection and centroid calculations.

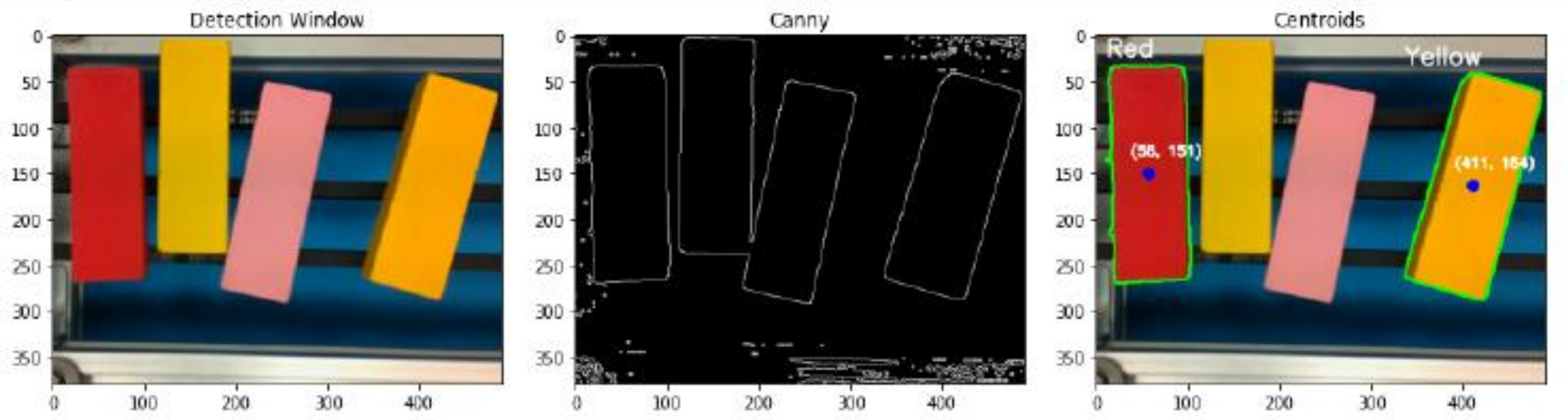


Inspection on Conveyor (Discussion)

Effect of DetectionWindow Increase/Decrease and Object Spacings

- Detection window size impacts the accuracy of pick-place operation
- Objects placed far from the vertical axis of the camera eye accounts for additional area.

  (Slight shift in centroid point)



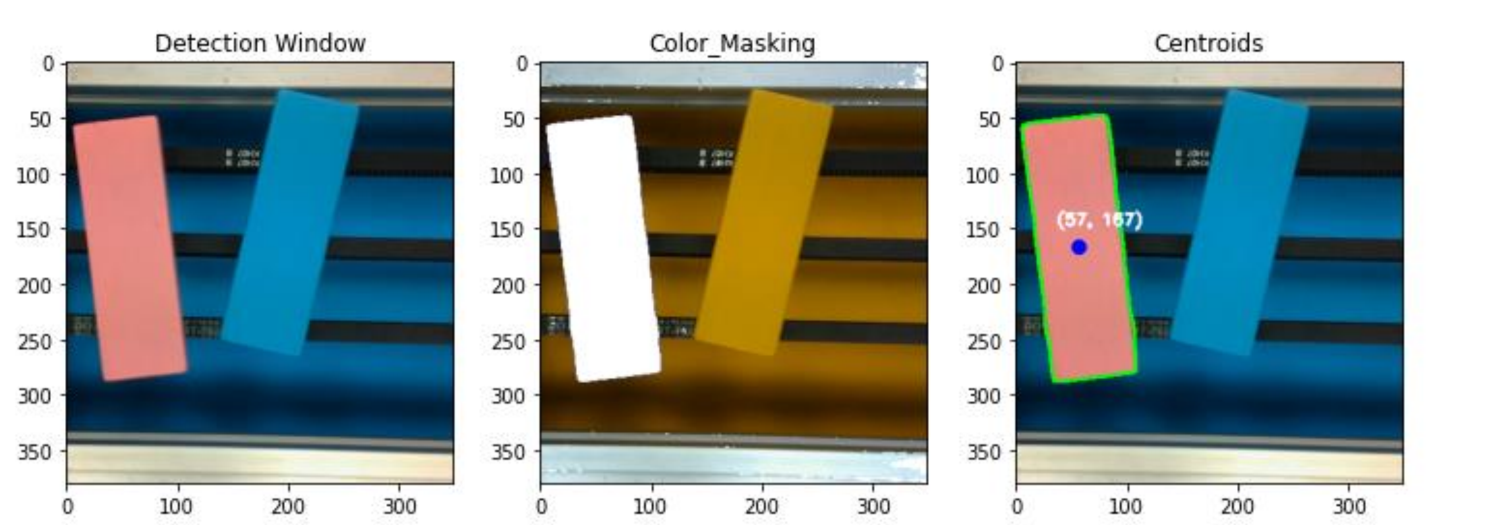
- Objects touching Each other: None Objects Detected
- Minimum Distance between the Objects:

Object Width (w): 25mm
Jaw span (s) : 38mm
Clearance (c): 6 mm (accounting for minor inaccuracies and safe operation)
Minimum Object Spacing: (s/2)-(w/2)+ c = 12.5 mm

Inspection on Conveyor (Discussion)

Detection of coloured objects similar to the background

- Objects colored similar to the background are difficult to detect. (Difficulties in Color Masking)



Normally **Blue and Green backgrounds** are used for digital image processing. The choice for using a blue background was made due to the below advantageous points:

- **Less Reflective:** Minimize Reflections
- **Low Light Situations:** Less bright and require less light.

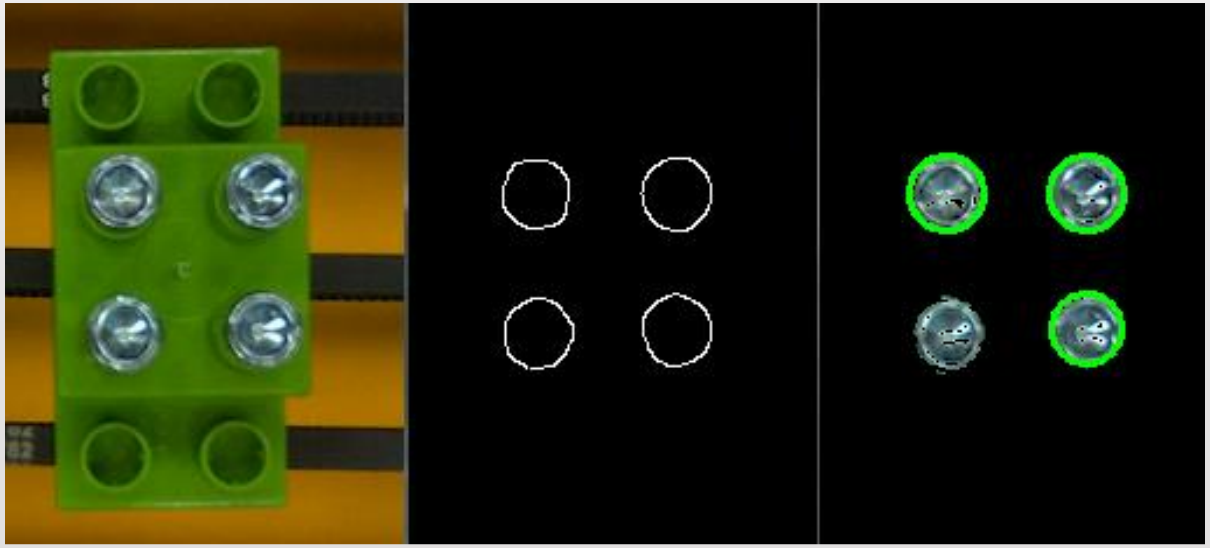
Inspection on Conveyor (Discussion)

False Positives for Detection of Bolts

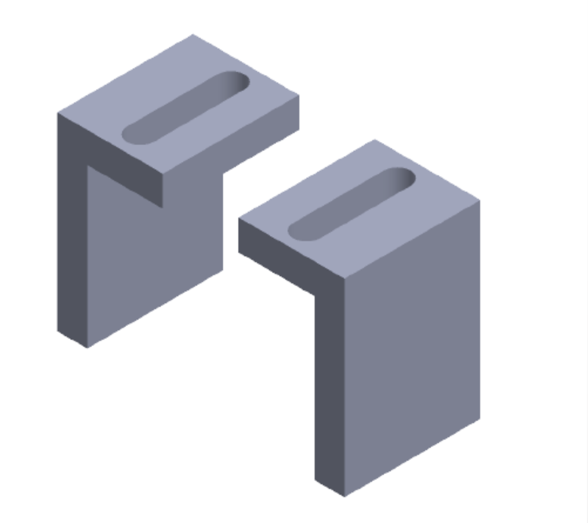
- Performance stable in Static Imaging
- Video Streaming :Images processed at 30 fps; Masking causes a flickering effect and unstable edges.

Applied solutions:

- Histogram Equalization and Median Blurring Added to the algorithm: reduce noise and jitter in the stream.
- Circles are approximated rather than directly using the raw canny edges result.



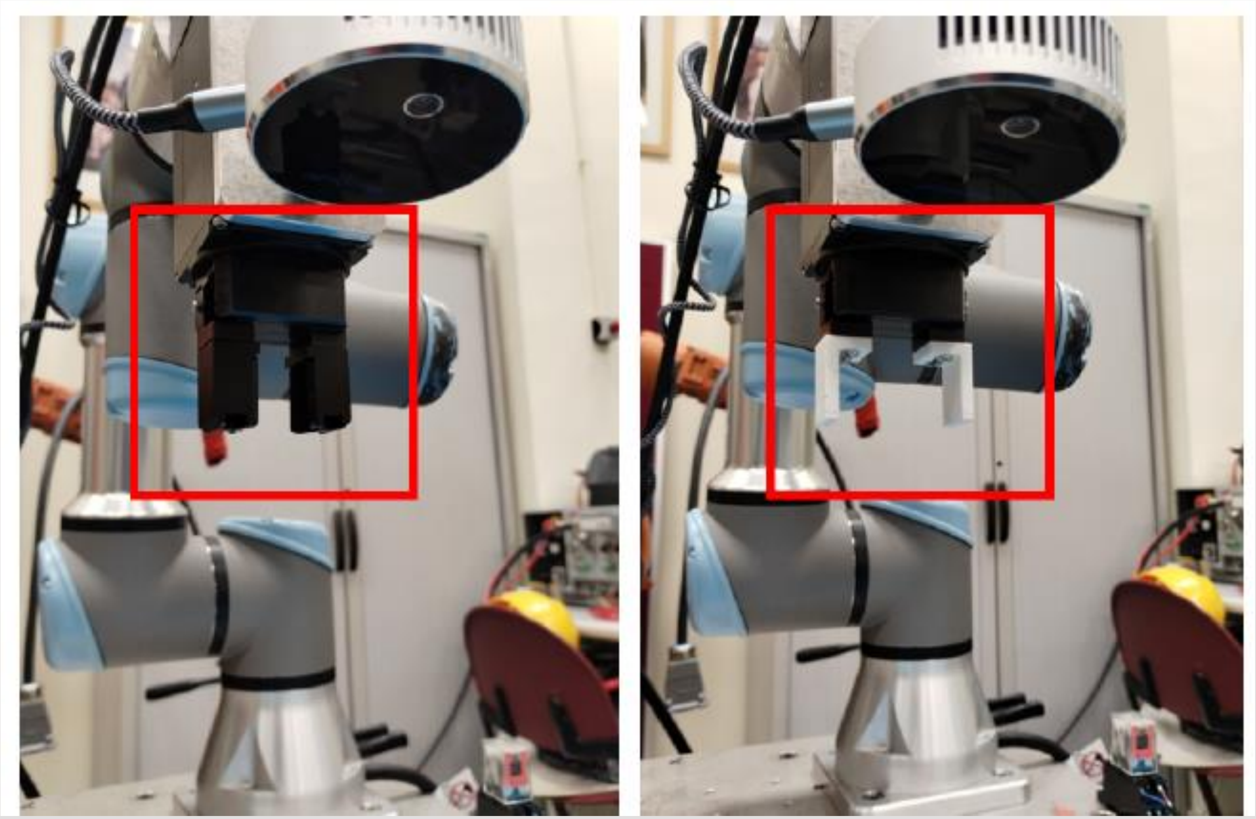
Challenges Faced



Gripper Jaw Span:

Object Dimensions: 75 x 25 x 15 mm

- **Previously:** Gripper Fully Open: **14mm**, Gripper Fully Closed: 2mm
- **Modified:** Gripper Fully Open: **38mm**, Gripper Fully Closed: 22mm



Challenges Faced

Gripper Rotation angle

- Programmed to rotate only 90 degrees horizontally and vertically.
- With the increase in Jaw Span: ±18 degrees; allowed slight deviations (Horizontally and Vertically).

Camera Depth Range Limitation:

- Minimum measuring distance of 50 cm
- Due to the limited reach of the robot arm, the depth data could not be extracted effectively.
- Objects placed on table and conveyor; depth remains constant; manually fed into the transformation calculations.

Conclusions

- Successful integration of Robot, Gripper-Camera assembly and Conveyor with Vision algorithm.
- Successful Implementation of advanced image processing to accurately detect blocks and extract centroids to perform pick and place operation.
- Employing Contour area calculations, reliable method for inspection of objects based on size established.
- Detection of deficient bolts achieved using depth image analysis; vital for quality assurance in manufacturing.

Future Directions

- Implementation of Eye to Hand Camera Setup (i.e. fixed camera eye).
- Integration of different gripper design(soft, vacuum or pneumatic) to explore pick and place operation with wide variety of objects.
- Using Machine learning to enhance the adaptability to perform complex object detections; improves system flexibility.
- Modifying the conveyor belt to reduce image jitters; a encoder could be added to detect and track objects for efficient inspection.
- High resolution cameras to capture detailed features; advanced detection algorithms combined with classical image processing to enhance reliability


Thank you for your attention! Have a good day!