Pick and Place Robotic-Arm

intended for object sorting task.

بشر الدبش

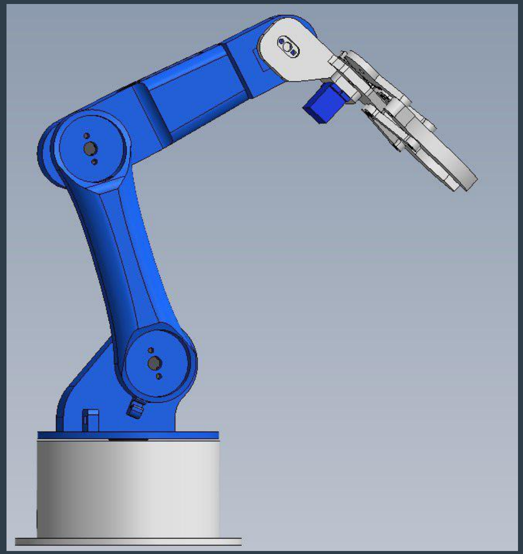
رهف الزيات

سلام عريضة

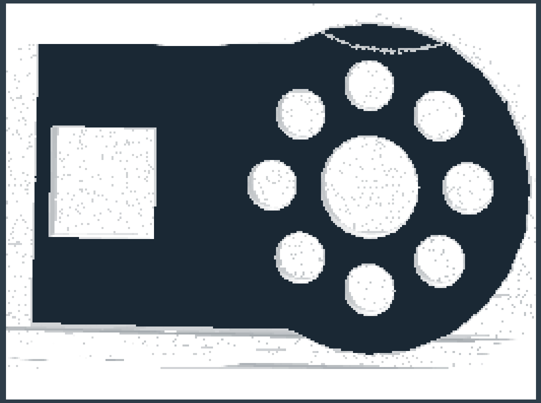
سنا غالغاي

Hardware Design

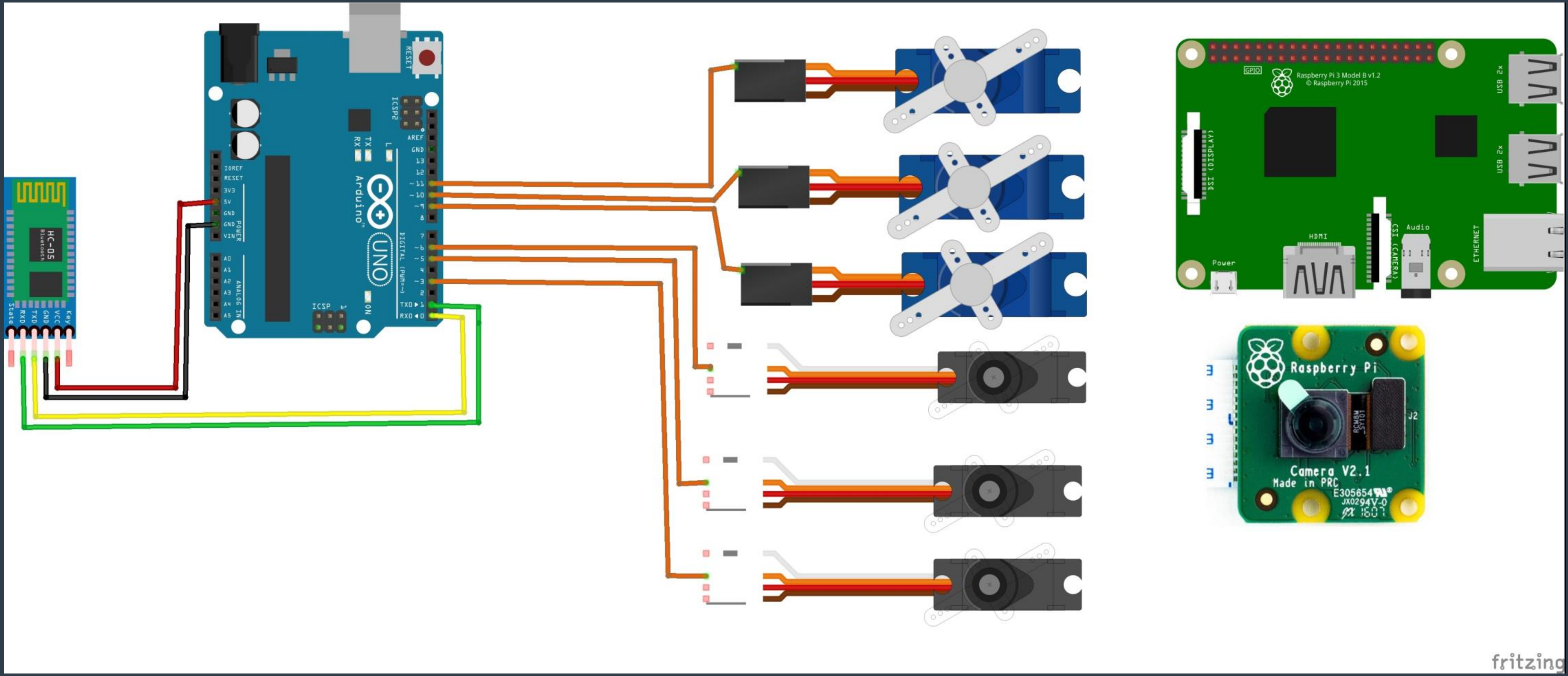
- 3D model of the arm



- 3D printed part used for Camera Mounting

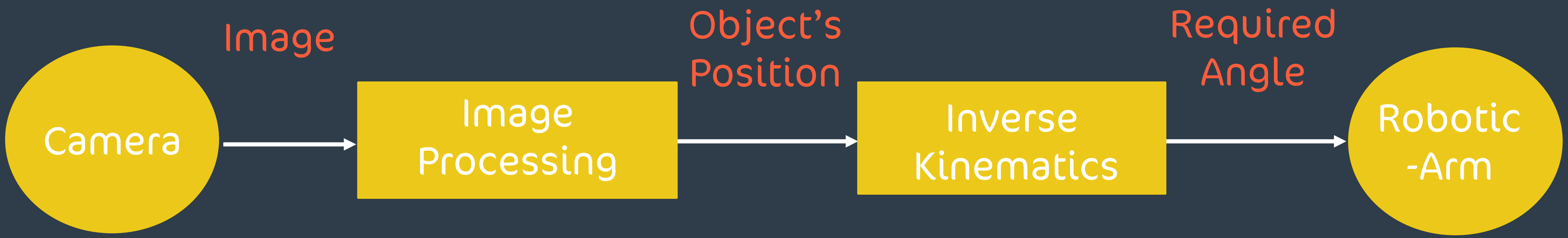


Wiring



Arduino
UNO
Raspberry Pi 3 Model B v1.2
© Raspberry Pi 2015
Raspberry Pi
Camera V2.1
Made in PRC
HC-05
fritzing

Black Box Chart



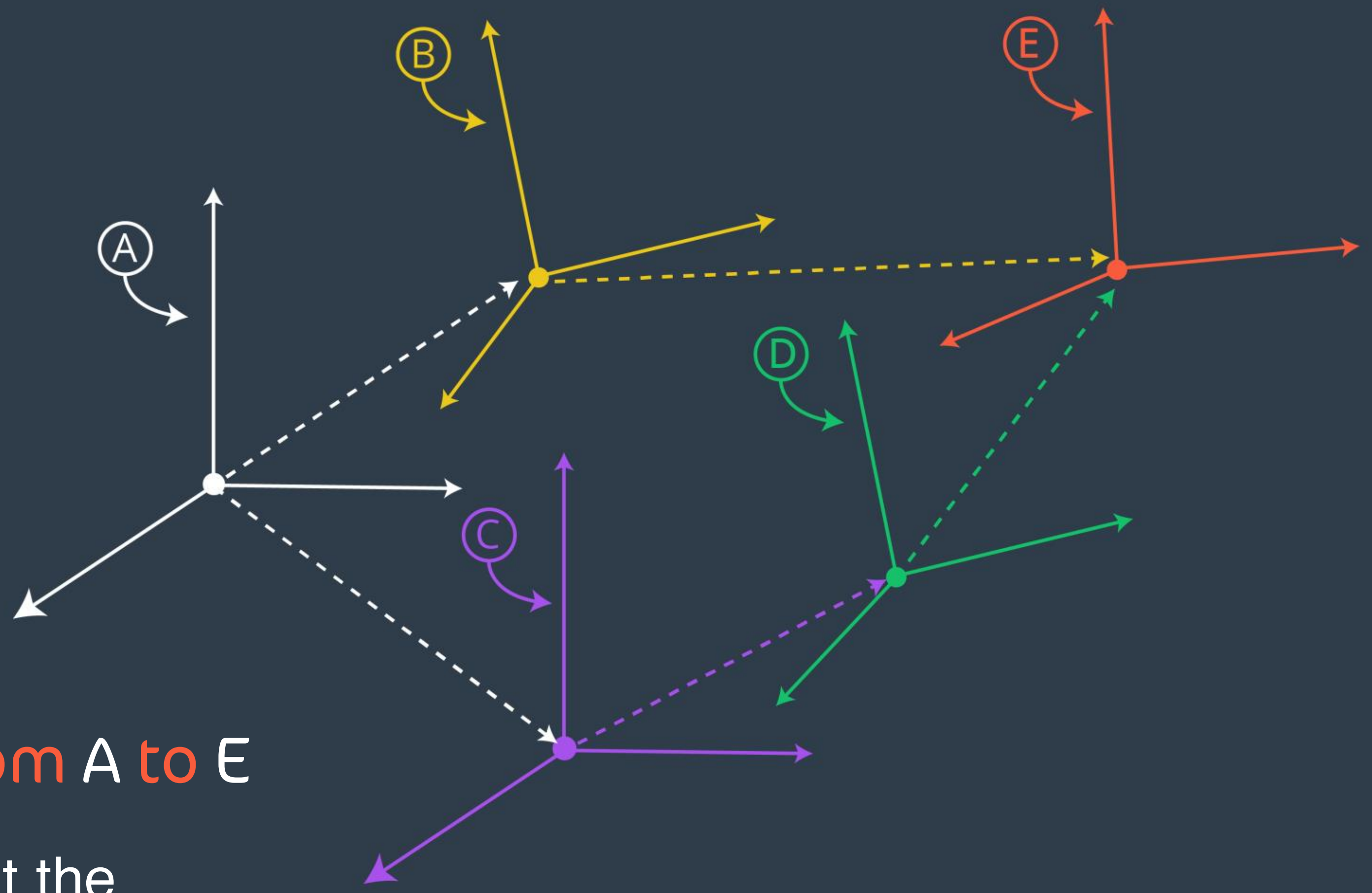
Kinematics


Joint
Space
$(\theta_1, \theta_2, \ldots)$
Forward Kinematics
Inverse Kinematics
Cartesian
Space
(x, y, z)

Composition of Homogeneous Transforms

- To create the transform between frames A and E

Clearly, there are two path



$$T_E^A = T_B^A T_E^B$$

$$T_E^A = T_C^A T_D^C T_E^D$$

$$T_B^A T_E^B = T_C^A T_D^C T_E^D$$

T_E^A : is the Transformation from A to E

In this way we are able to represent the position and the orientation of any point in E relative to our base frame A

Forward Kinematics

The FK problem boils down to the composition of homogeneous transforms.

We start with the base link and move link by link to the end effector.

$$T_N^0 = T_1^0 T_2^1 T_3^2 \dots T_N^{N-1}$$

DH Method

- The method only requires four parameters to describe the position and orientation of neighboring reference frames which are:

$\alpha_{i-1}(twist\ angle) = angle\ between\ \hat{z_i} - 1\ and\ \hat{Z}_{\{i\}}\ measured\ about\ \hat{X}_{i-1}\ in\ a\ right - hand\ sense.$

$a_{i-1}(link\ length) = distance\ from\ \hat{z_i} - 1\ to\ \hat{Z}_{\{i\}}\ measured\ along\ \hat{X}_{i-1}\ where\ \hat{X}_{\{i-1\}}\ is\ perpendicular\ to\ both\ \hat{Z}_i - 1\ to\ \hat{Z}_{\{i\}}$

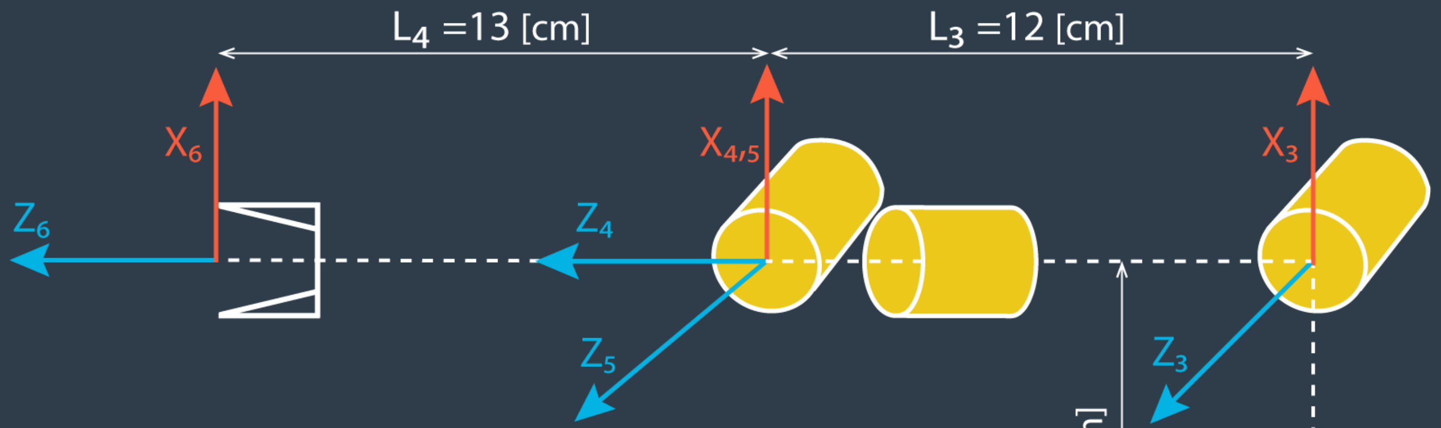
$d_i(link\ offset) = signed\ distance\ from\ \hat{X}_i - 1\ to\ \hat{X}_{\{i\}}\ measured\ along\ \hat{Z}_i.$

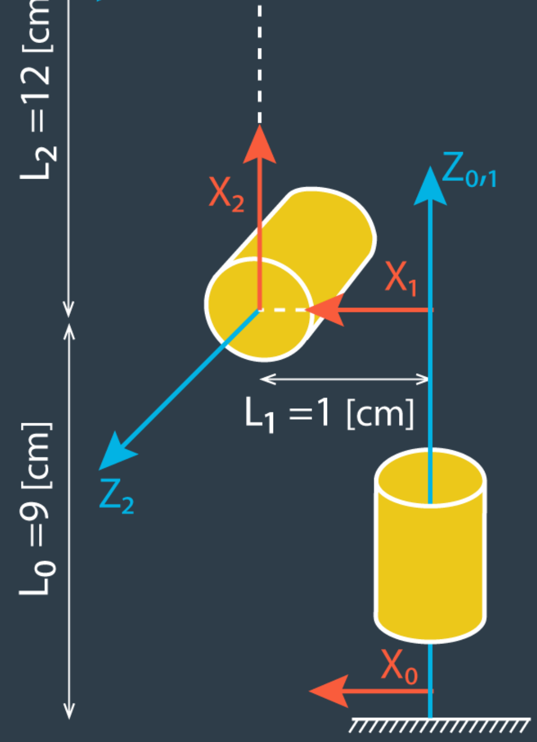
Note that this quantity will be a variable in the case of prismatic joints.

$\theta_i(joint\ angle) = angle\ between\ \hat{X}_i - 1\ to\ \hat{X}_{\{i\}}\ measured\ about\ \hat{Z}_i\ in\ a\ right - hand\ sense.$

Note that this quantity will be a variable in the case of revolute joint.

DH-Parameters





i	α_{i-1}	a_{i-1}	d_i	θ_i
T_1^0	0	0	L_0	θ_1
T_2^1	-90	a_1	0	$\theta_2 - 90$
T_3^2	0	a_2	0	θ_3
T_4^3	-90	0	L_3	θ_4
T_5^4	$+90$	0	0	θ_5
T_6^5	-90	0	L_5	0

Homogenous Matrix for DH Method

Given the DH parameters we can represent the homogeneous transform from frame $\boldsymbol{i-1}$ to frame $\boldsymbol{i}$ is constructed as a sequence of four basic transformations, two rotations and two translations.

$$T_i^{i-1} = R_x(\alpha_{i-1})T_x(a_{i-1})R_z(\theta_i)T_z(d_i)$$

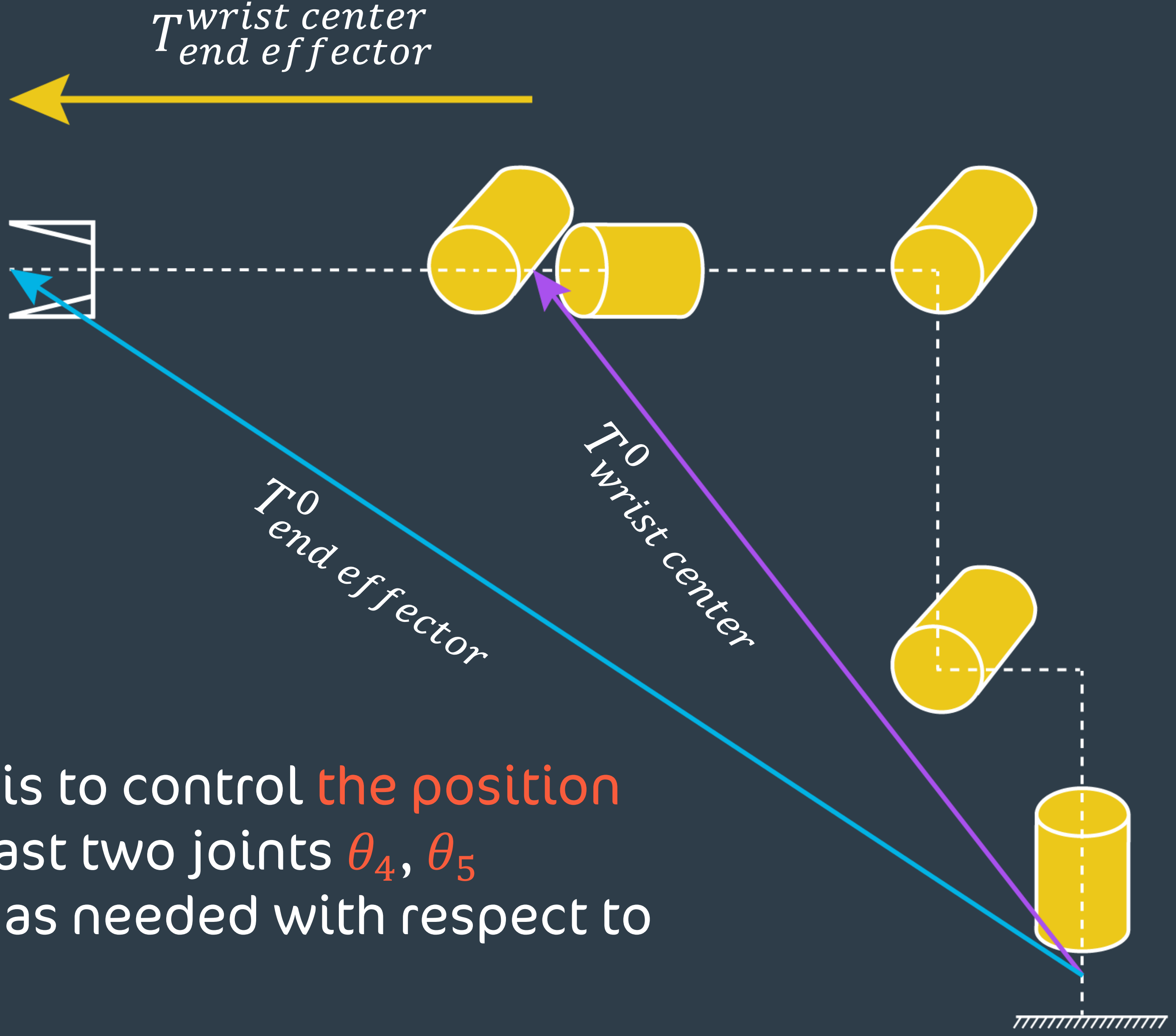
Multiplying all four Transformations results:

$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

To get the over all Transformation

$$T_6^0 = T_1^0T_2^1T_3^2T_4^3T_5^4T_6^5$$

Inverse kinematics



- This design Kinematically decouples the position and orientation of end effector

- The first three joints $\theta_1, \theta_2, \theta_3$ is to control the position of the wrist center while the last two joints θ_4, θ_5 would orient the end effector as needed with respect to the Wrist center.

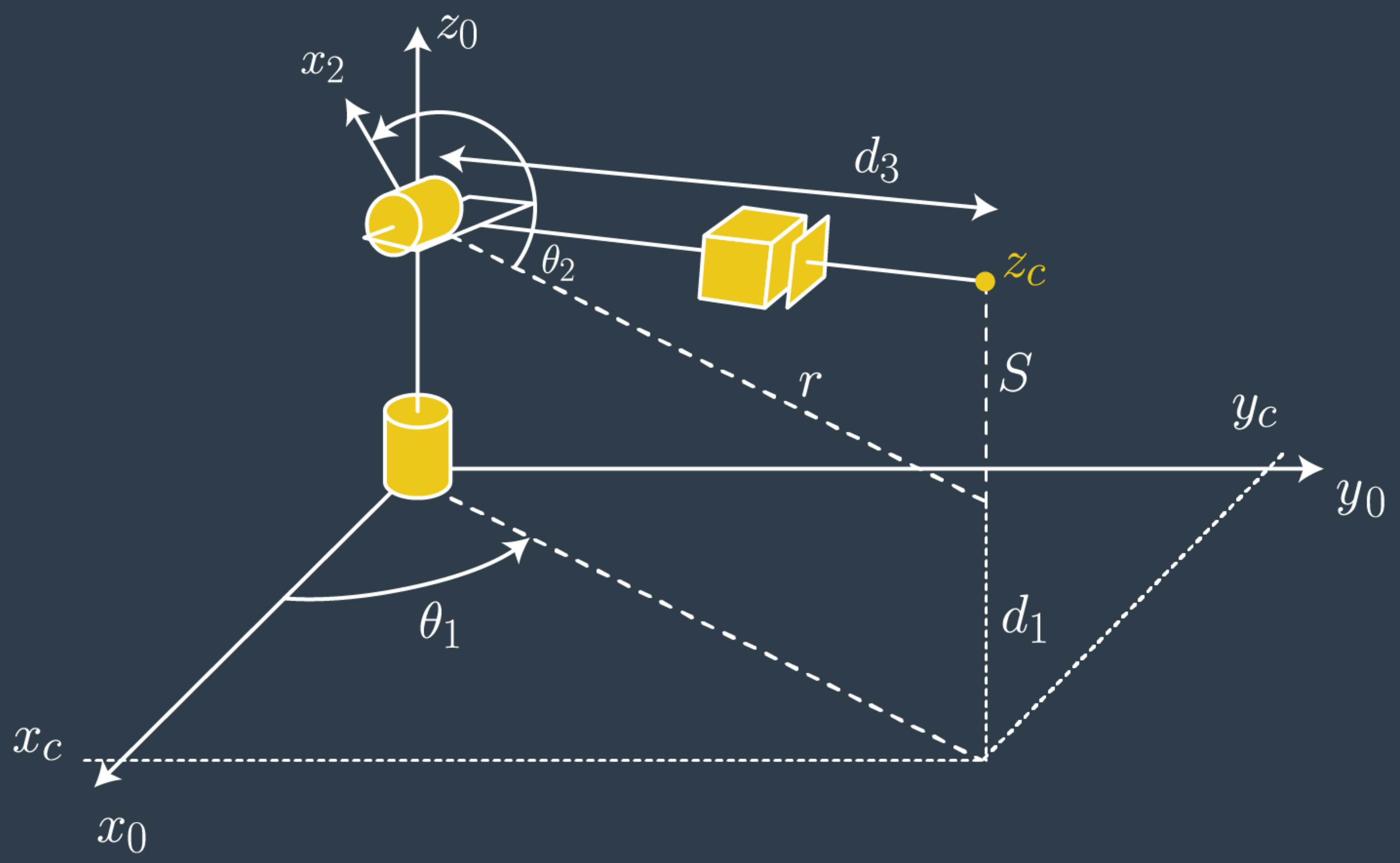
Calculating Angles

θ_1, θ_2 and θ_3 can be calculated in a Closed-form solutions.
In order to calculate θ_4 and θ_5:

Overall rotation $R_6^0 = R_3^0 R_6^3$

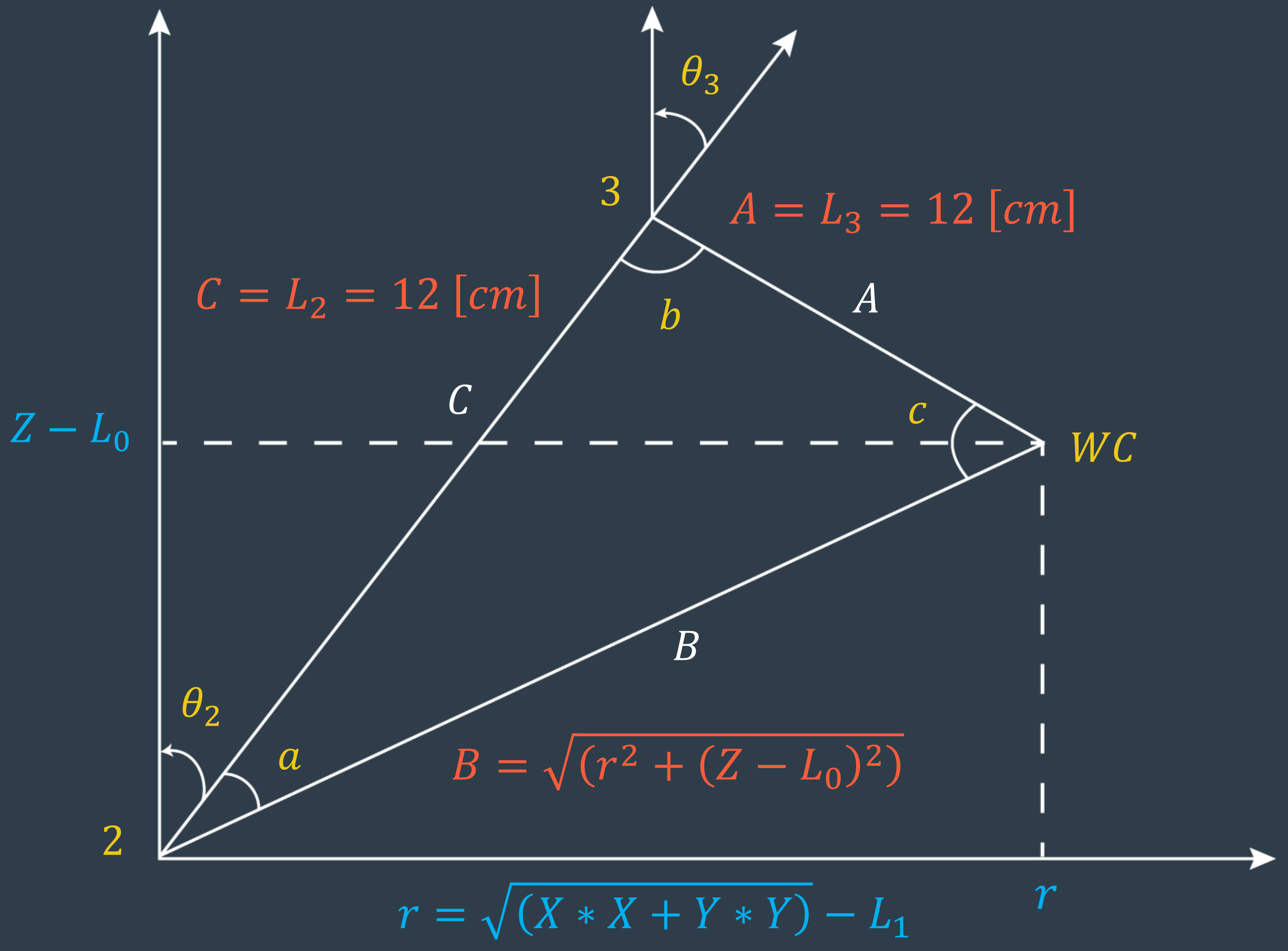
$$R_6^3 = (R_3^0)^{-1}\, R_6^0 = (R_3^0)^T\, R_6^0$$

And then we choose the correct solution among the set of possible solutions.



$$\theta_1 = arc\,tan\frac{y_c}{x_c}$$

Side View



$$a = arc\cos\frac{(B^2 + C^2 - A^2)}{(2 * B * C)}$$

$$b = arc\cos\frac{(A^2 + C^2 - B^2)}{(2 * A * C)}$$

$$\theta_2 = \frac{\pi}{2} - a - arc\tan\frac{Z - L_0}{r}$$

$$\theta_3 = \frac{\pi}{2} - b$$

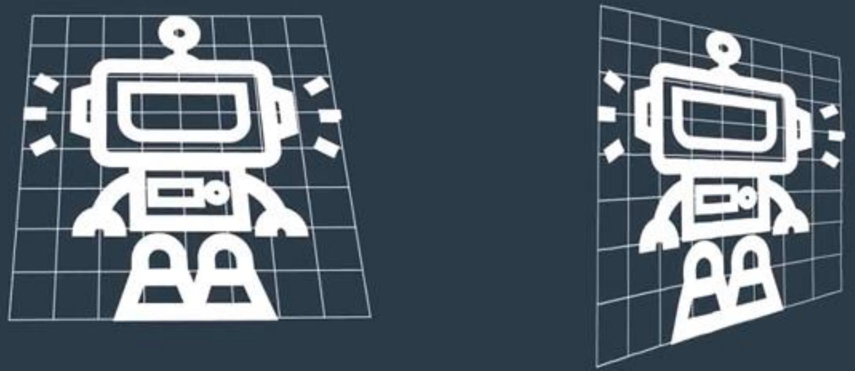
Pinhole camera model

the Transformation from 3D object points, P of X, Y and Z to X and Y is done by a transformative matrix called the camera matrix(C)

Types of Distortion in Cameras

1 Radial Distortion

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$



2 Tangential Distortion

$$x_{corrected} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

Tangential distortion occurs when the lens and the image plane are not parallel, it will create an image that looks tilted so that some objects appear farther away or closer than they actually are.

Camera Calibration

- distortion changes the size and shape of the object in an image.
- In order to calibrate the camera we can take pictures of known shapes, then we'll be able to detect and correct any distortion errors.
- A chessboard is great for calibration because it's regular, high contrast pattern makes it easy to detect automatically. And we know how an undistorted flat chessboard looks like.

- Since our camera is calibrated, we can define something like:

$$\mathit{Pixel\ to\ mm\ ratio} = \frac{\mathit{size\ of\ a\ known\ object\ in}\ \boldsymbol{pixel}}{\mathit{size\ of\ the\ object\ in\ the\ real\ world}\ (\boldsymbol{mm})}$$

$$\boldsymbol{x_{mm}} = \boldsymbol{x_{pixel}} * \mathit{Pixel\ to\ mm\ ratio}$$

$$\boldsymbol{y_{mm}} = \boldsymbol{y_{pixel}} * \mathit{Pixel\ to\ mm\ ratio}$$

- We calculated the coordinates of the object with respect of the image coordinate system

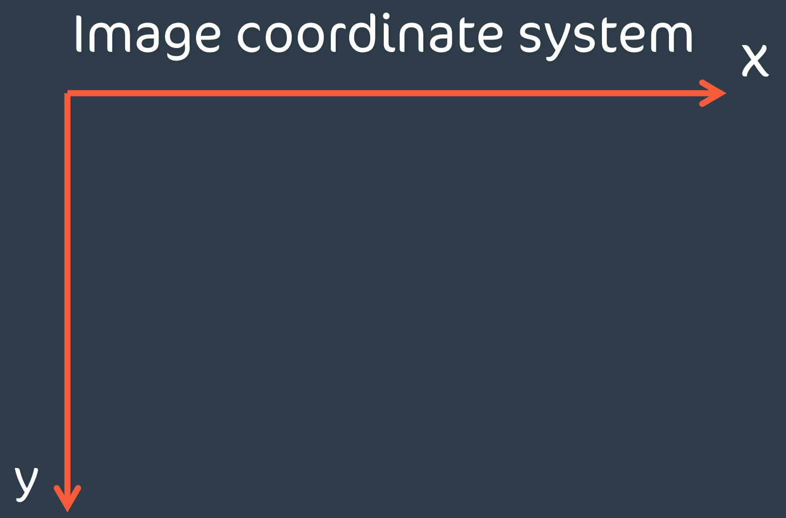
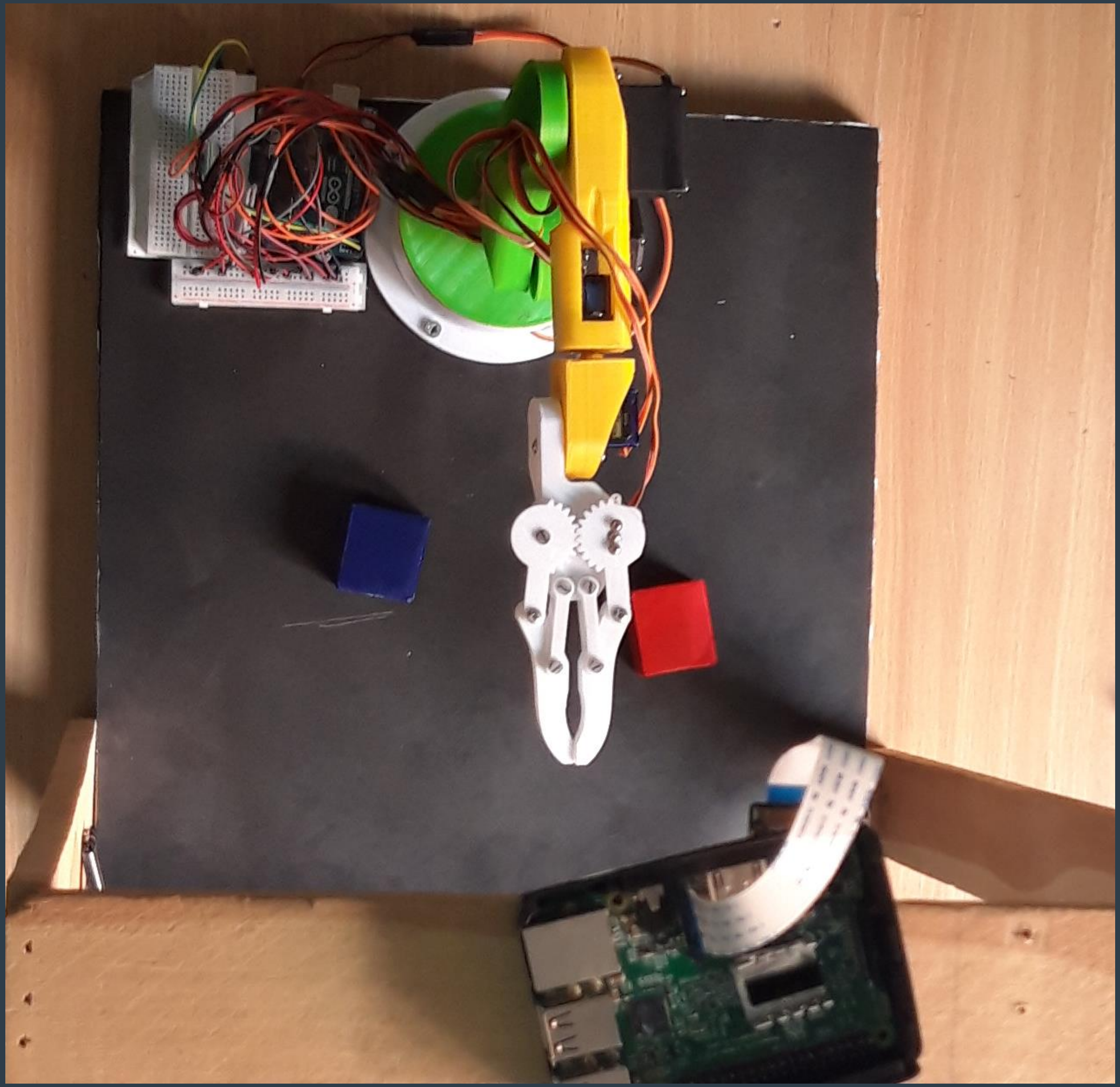


Photo shows the camera and the arm locations



Transformation from image coordinate to the arm reference coordinate T_{arm}^{image}

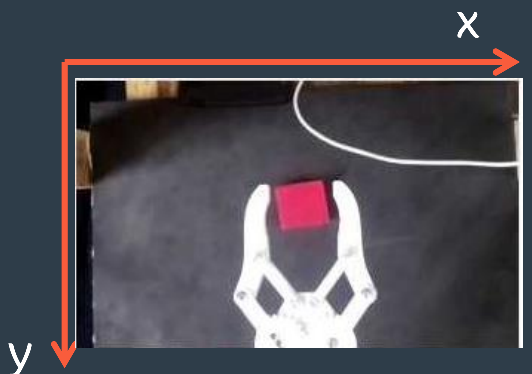
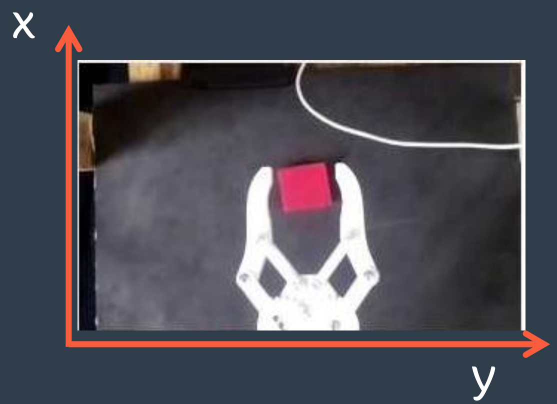


Image coordinate system

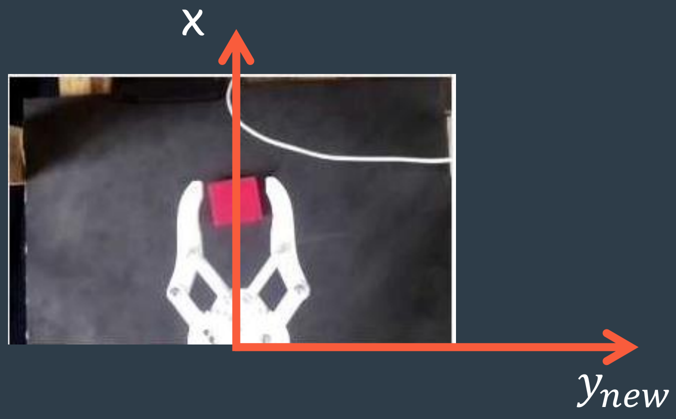
Correcting coordinate system

(to meet our needs)



Shifting y-axis

$$y_{new} = y - \frac{y_{max}}{2}$$

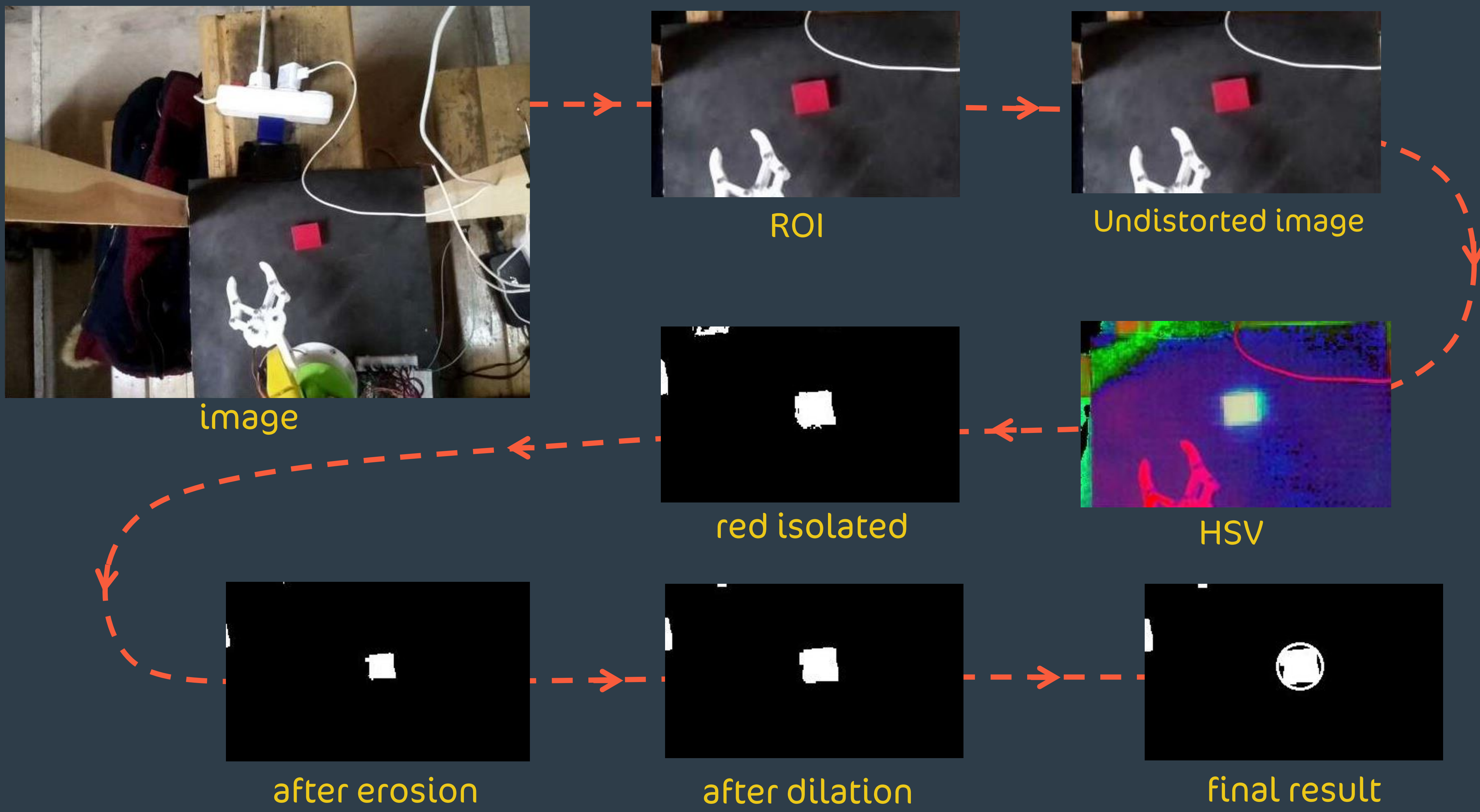


$$x_{arm} = x_{image} + distance\ from\ arm\ to\ image\ on\ x - axis\ (mm)$$

$$y_{arm} = y_{image} + 0$$

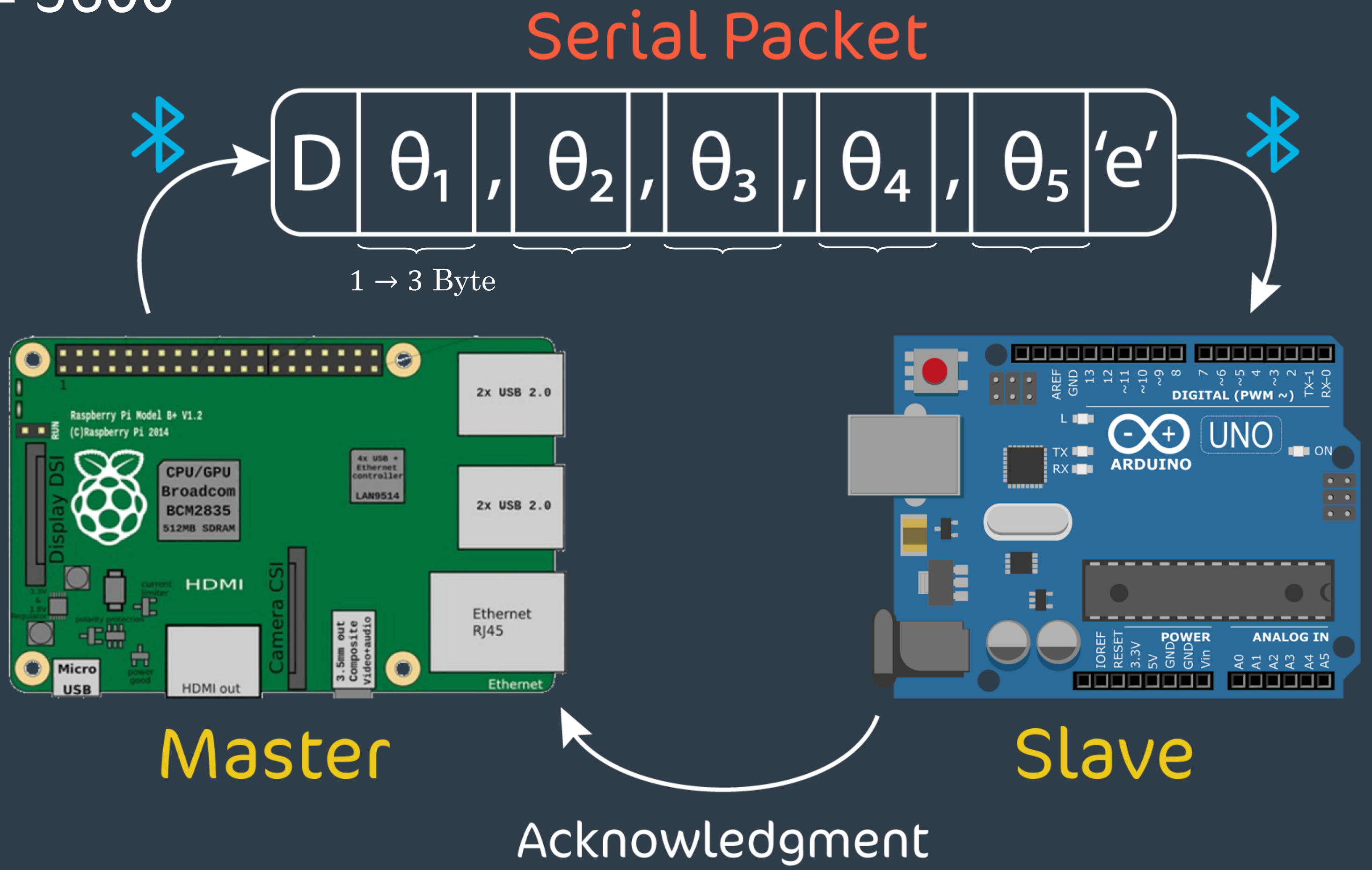
- We don't get any Z-axis information from the camera

Computer Vision Processes



Bluetooth Communication

- D (direction): to determine where the object should be placed.
- ‘e’ ≡ special character to determine the “end” of the packet
- Acknowledgment ≡ ready to receive new data.
- Baud rate = 9600



PYTHON CODE

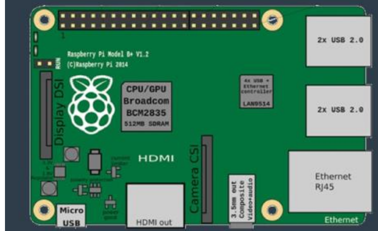
- Meets PEP-8 Standards
- Multi threaded
- OOP

Main three things are

- Name convention
- Indentation level
- Documentation

Tries to apply
S.O.L.I.D
Principles

Power Consumption

	MG-995 Motor	SG90	RaspberryPi 3 Model B	RaspberryPi Camera v2
				
Max Current Draw	$1200\ mA$	$700\ mA$	$1\ Amp$	$250\ mA$
Torque	$9.4\ Kg/cm$	$1.3\ Kg/cm$		

References

- Robotics Software Engineer - Udacity nano degree.
- Self-Driving Car Engineer - Udacity nano degree.
- Stackoverflow.
- Github.
- OpenCV docs.

- How to Mechatronics
- Robotics Tutorial 1
- Robotics Tutorial 2