Basic Image Processing
and
Robotic System

TANAKORN KULSRI

Robot Academy

King Monkut’s of University of Technology North Bangkok

Outline

Basic Image Processing

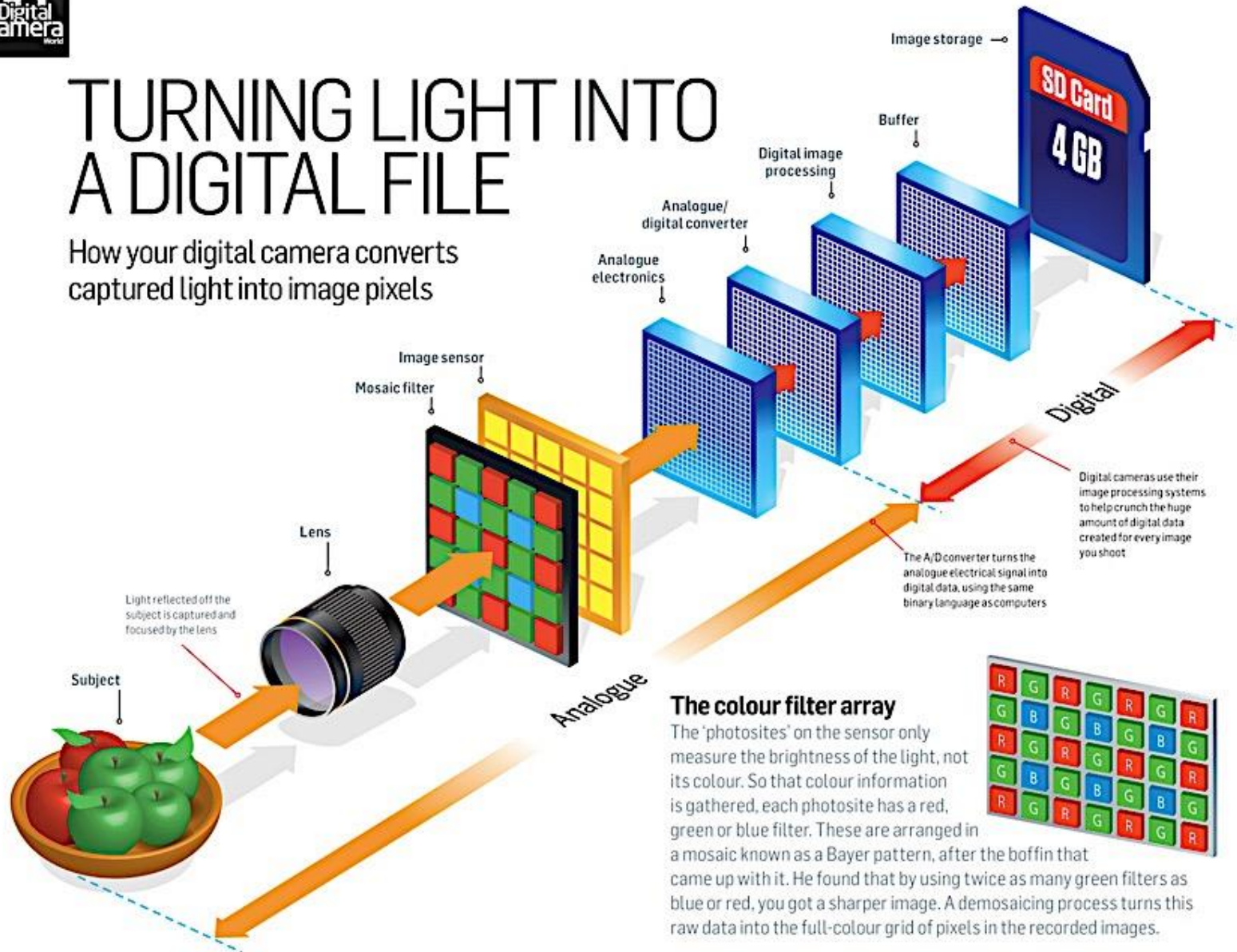
- Concept of Image
- Thresholding
- Morphology
- Contour Detection

Basic Robot Arm System

- Frame and Transformation Matrix
- Workspace

TURNING LIGHT INTO A DIGITAL FILE

How your digital camera converts captured light into image pixels



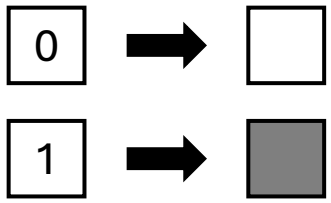
The colour filter array

The 'photosites' on the sensor only measure the brightness of the light, not its colour. So that colour information is gathered, each photosite has a red, green or blue filter. These are arranged in a mosaic known as a Bayer pattern, after the boffin that came up with it. He found that by using twice as many green filters as blue or red, you got a sharper image. A demosaicing process turns this raw data into the full-colour grid of pixels in the recorded images.

www.digitalcameraworld.com

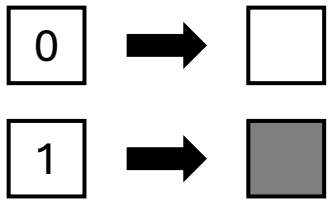
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

What if:



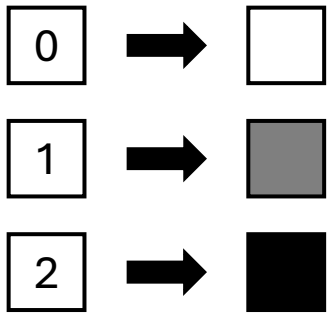
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

What if:



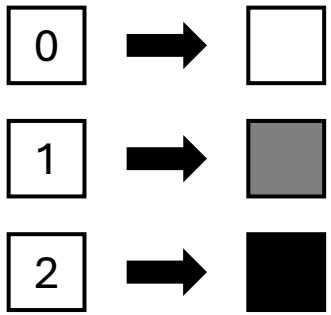
2	2	2	0	0	0	0	0	0	0	2	2	2
2	0	0	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	2
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	2
2	2	2	0	0	0	0	0	0	0	2	2	2

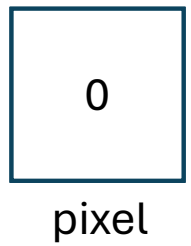
What if:



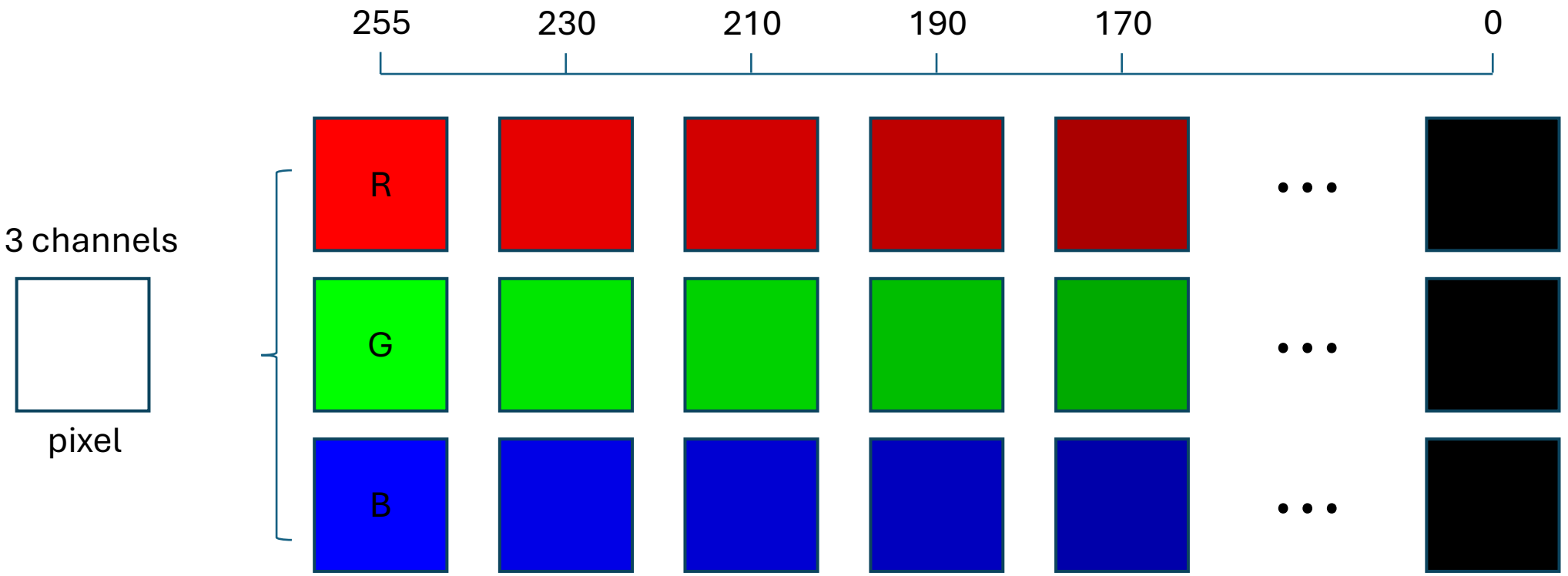
2	2	2	0	0	0	0	0	0	0	2	2	2
2	0	0	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	2
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	2
2	2	2	0	0	0	0	0	0	0	2	2	2

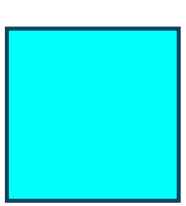
What if:




0
pixel

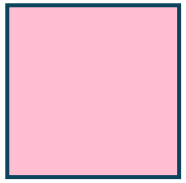
Digital color intensity (light intensity)





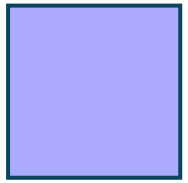
R G B

RGB(0,255,255)



R G B

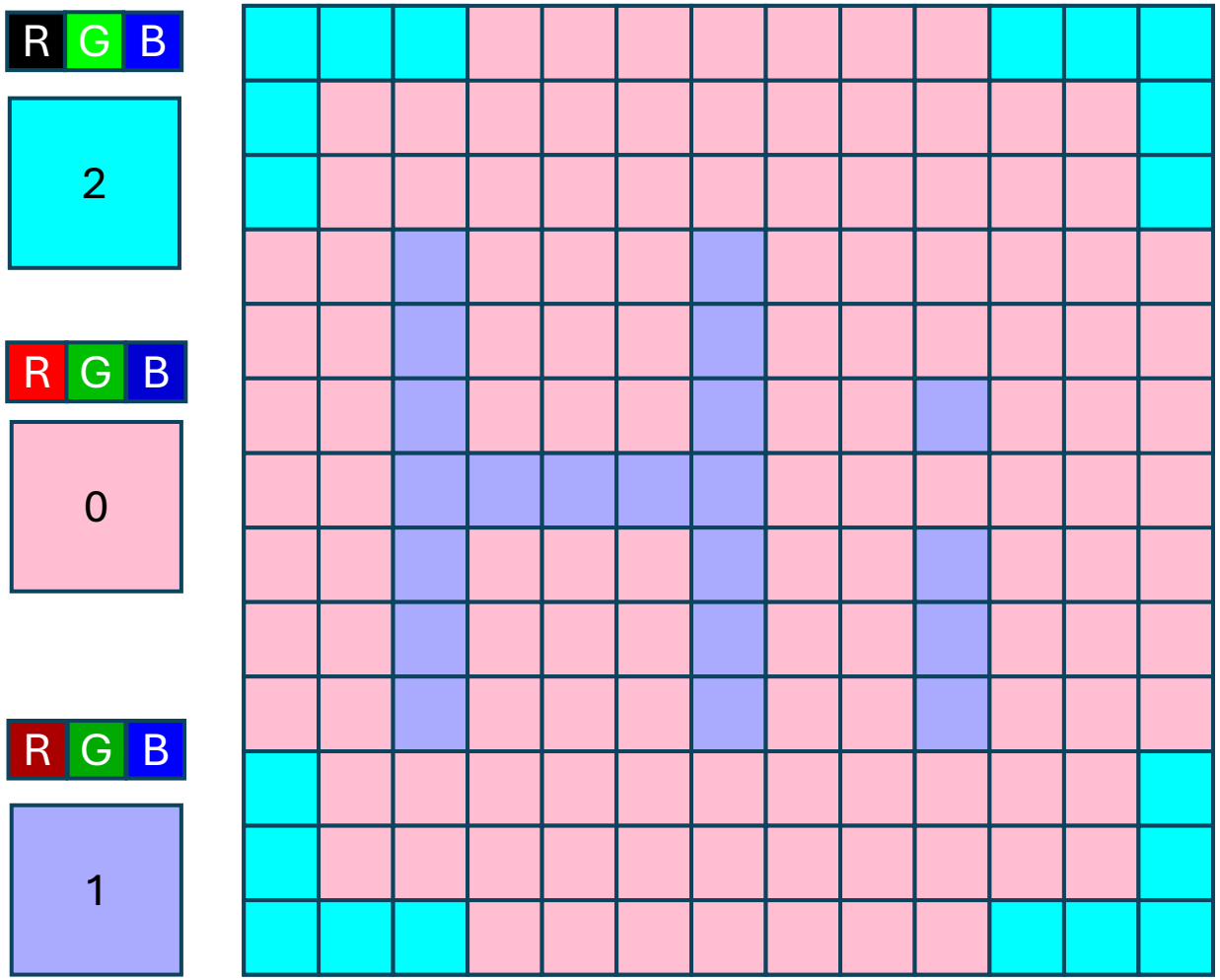
RGB(255,190,210)



R G B

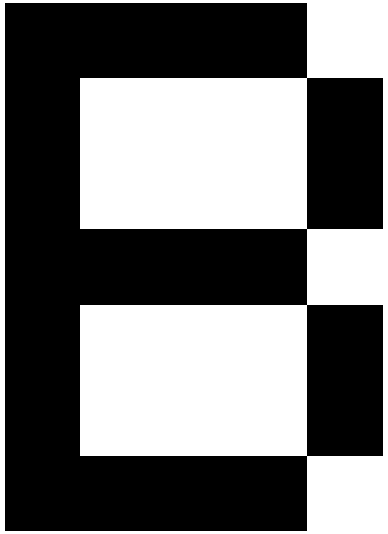
RGB(170,170,255)

2	2	2	0	0	0	0	0	0	0	2	2	2
2	0	0	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	2
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	2
2	2	2	0	0	0	0	0	0	0	2	2	2

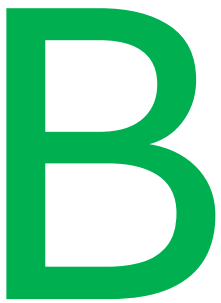


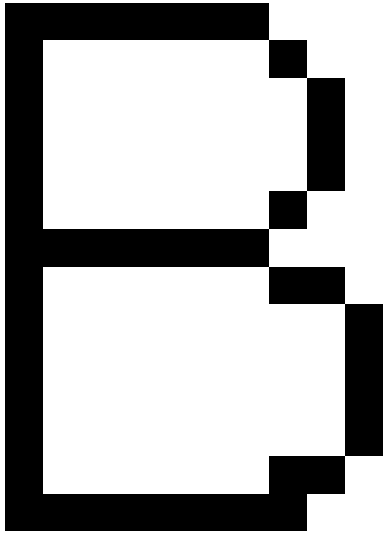
Greeting

Image resolution



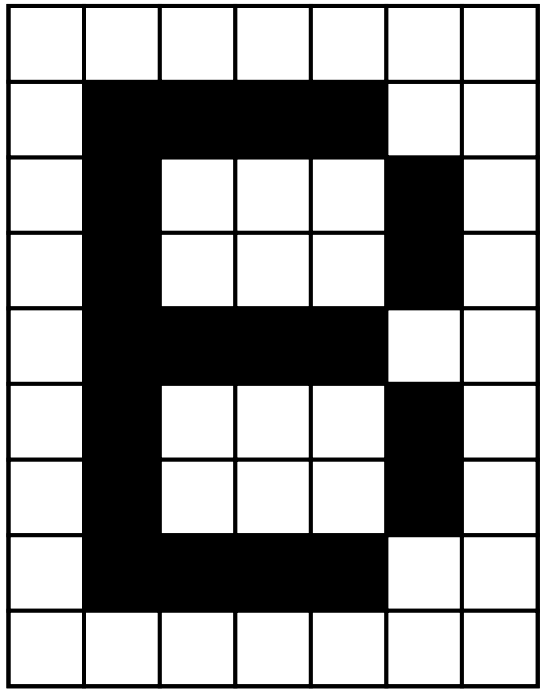
9×7





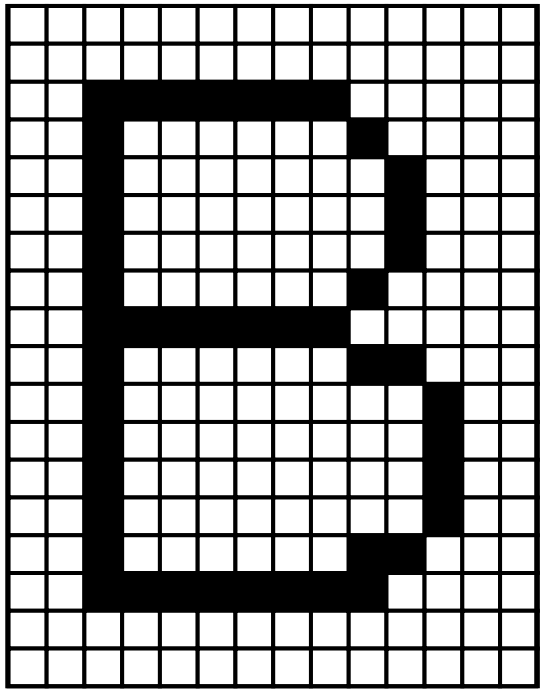
18×14

Image resolution



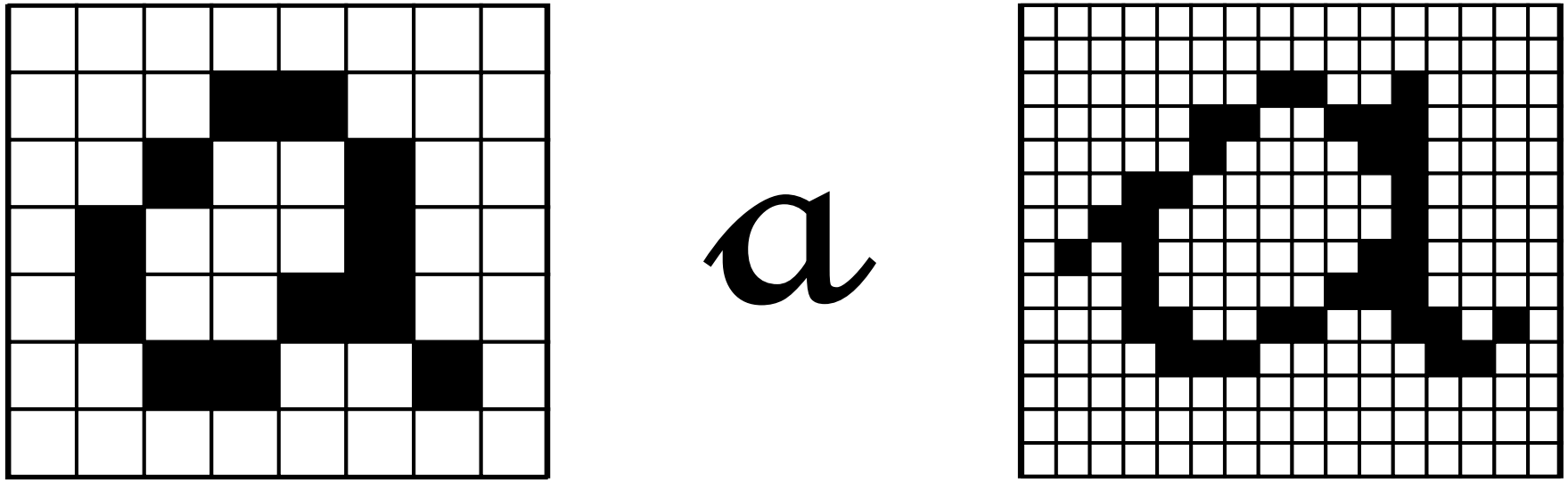
9×7

B



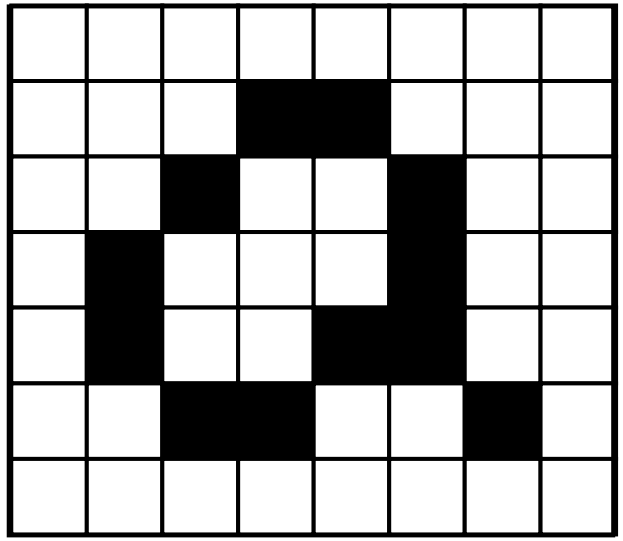
18×14

Image resolution

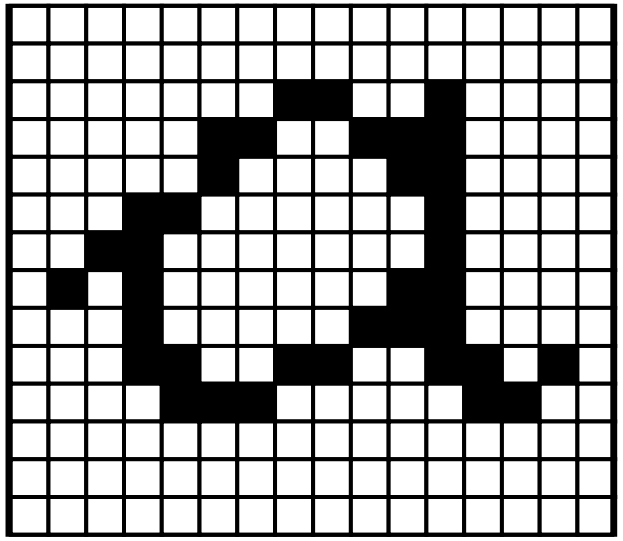


8×7 $\qquad$ 16×14

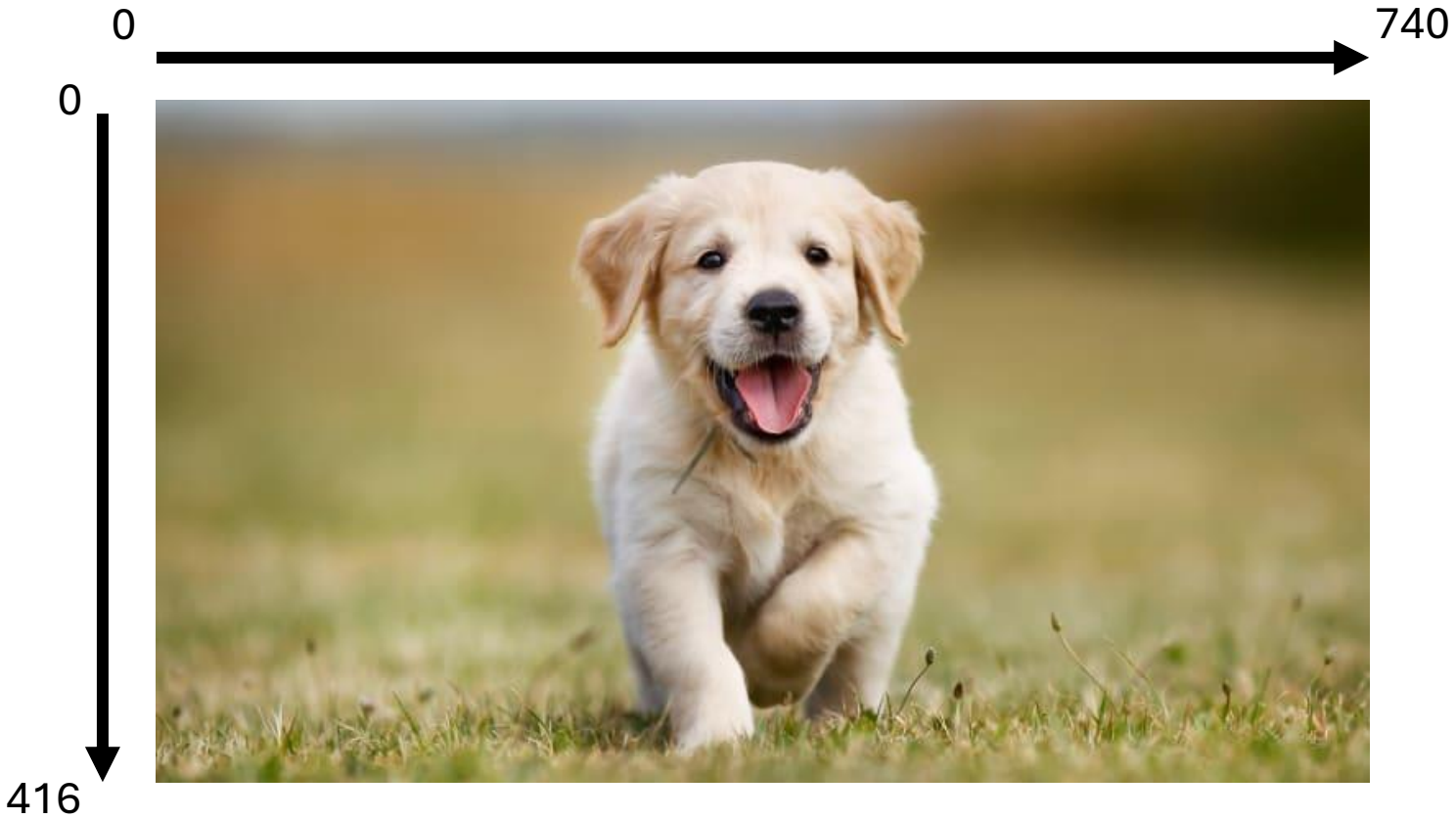
Image resolution



8×7



16×14



416×740

Type of images

Binary image

Gray image

Color image

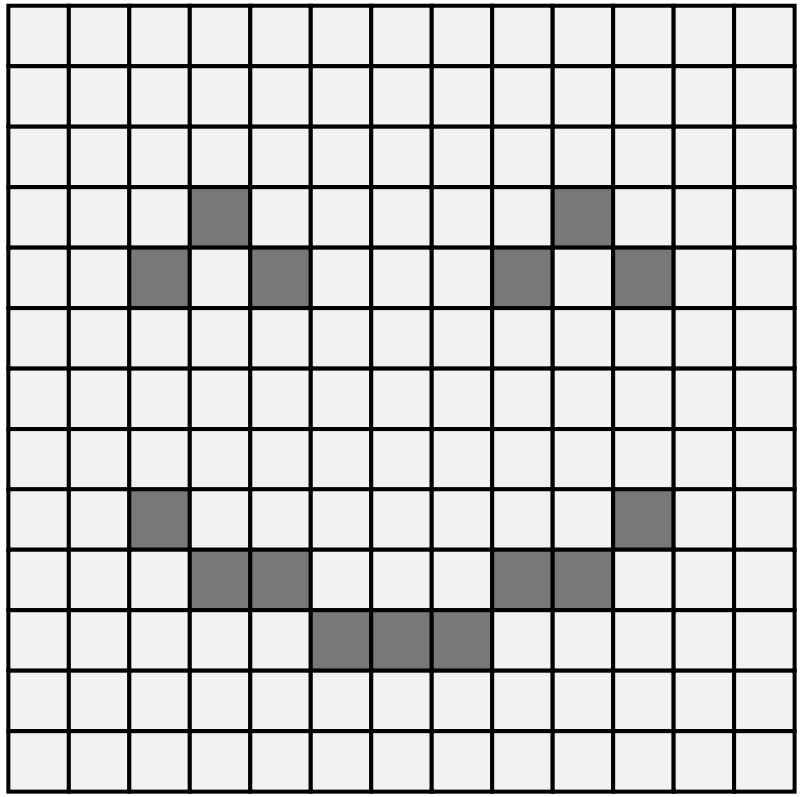
Type of images

Binary		0 or 1
Gray scale		8 bits 0 255
Color scale		8 bits 8 bits 8 bits 0 255 24 bit color format also known as true color format

Try to Coding

```
from ImageProc import arucoProjection
import cv2

cam = arucoProjection(0)

While True :
        projection_image ,marker_image = cam.get_projection_image()

        cv2.imshow('projection', projection_image)
        cv2.imshow('marker', marker_image)

        key = cv2.waitKey(1) & 0xFF

        if key == ord(‘q’) :
                break

cam.stop_camera()
cv2.destroyAllWindows()
```

Thresholding



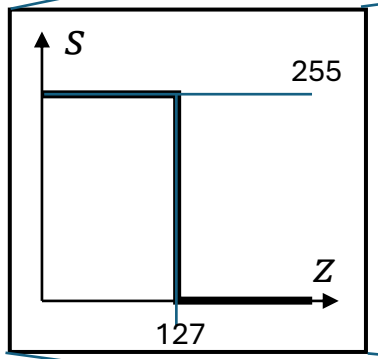
original

242

z

120

threshold

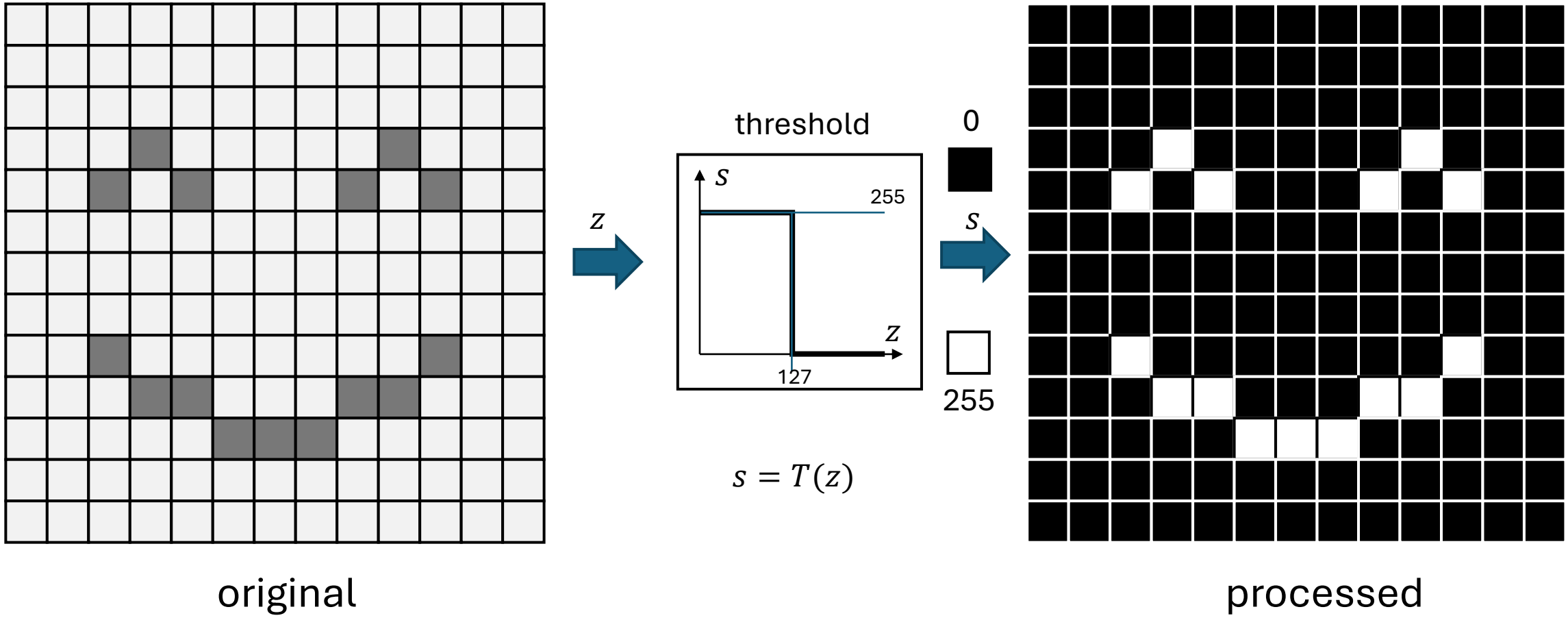


s

```
if(z<127):
  s=255
else:
  s=0
```

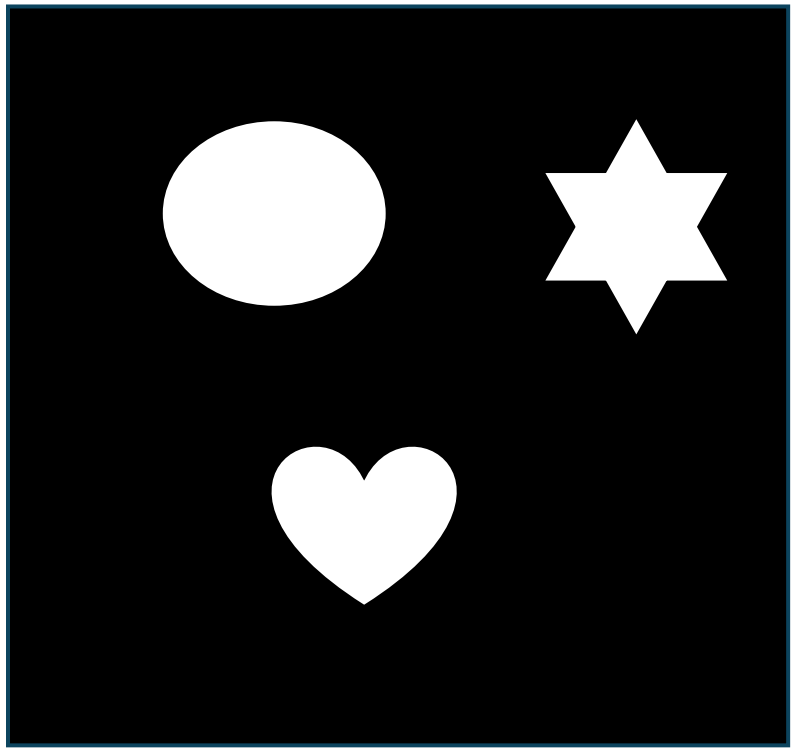
$s = T(z)$

Thresholding



Thresholding

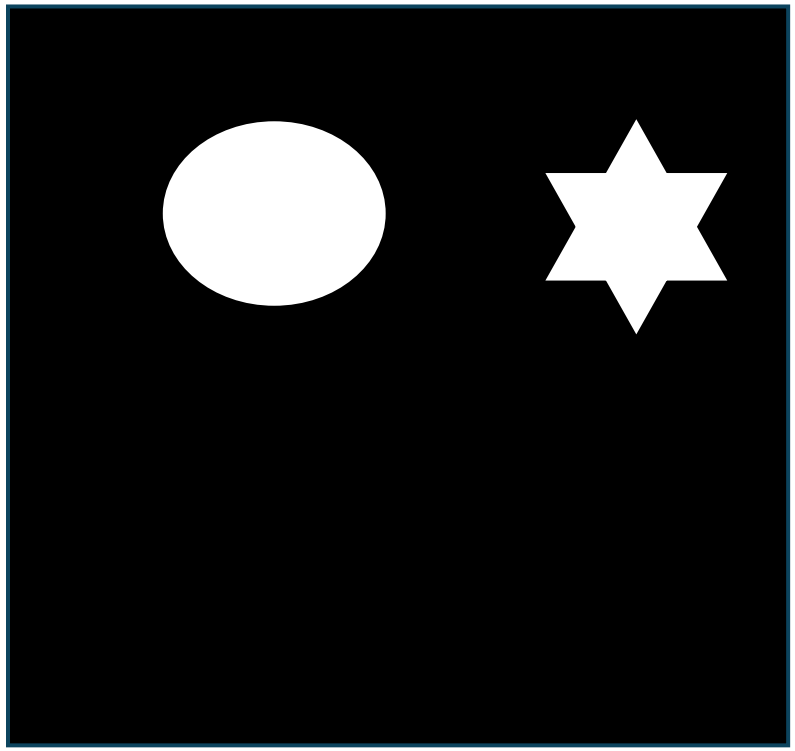
The basic of intensity thresholding



$$g(x,y) = \begin{cases} 255 & if\ Z(x,y) > 100 \\ 0 & otherwise \end{cases}$$

Thresholding

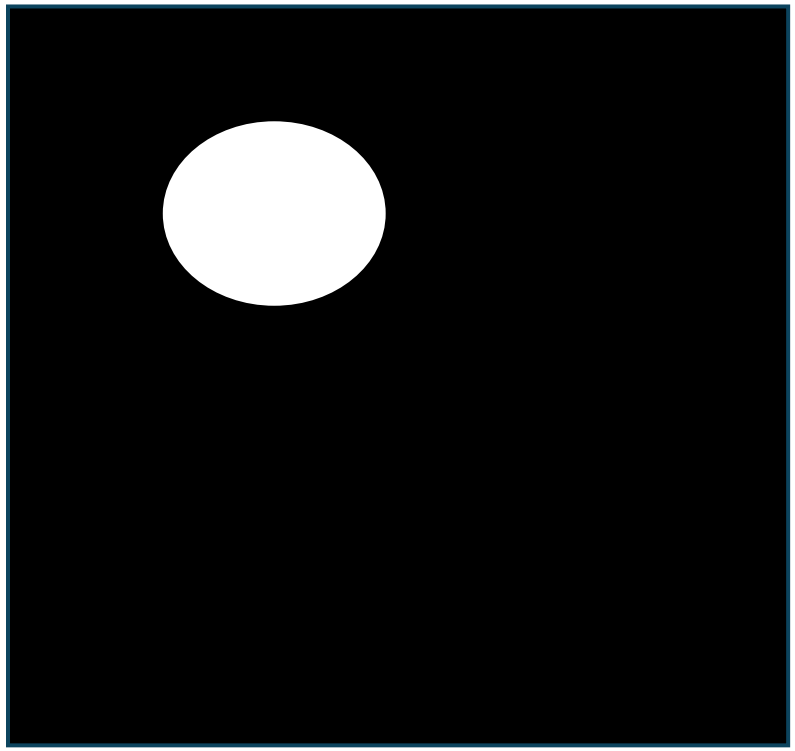
The basic of intensity thresholding



$$g(x,y) = \begin{cases} 255 & if\ Z(x,y) > 160 \\ 0 & otherwise \end{cases}$$

Thresholding

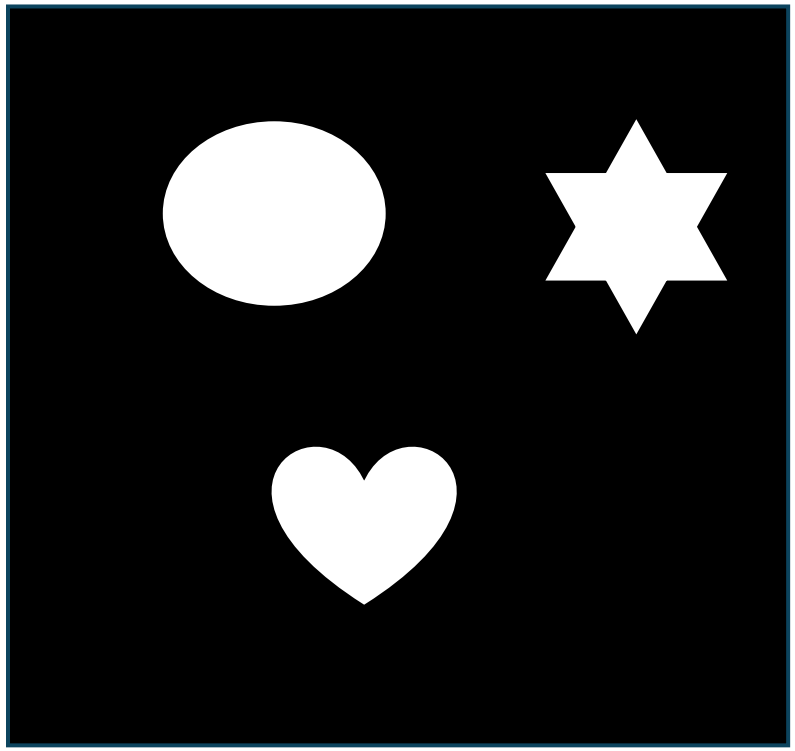
The basic of intensity thresholding



$$g(x,y) = \begin{cases} 255 & if\ Z(x,y) > 160 \\ & \text{And } Z(x,y) < 200 \\ 0 & otherwise \end{cases}$$

Thresholding

The basic of intensity thresholding



$$g(x,y) = \begin{cases} 255 & if\ Z(x,y) > T \\ 0 & otherwise \end{cases}$$

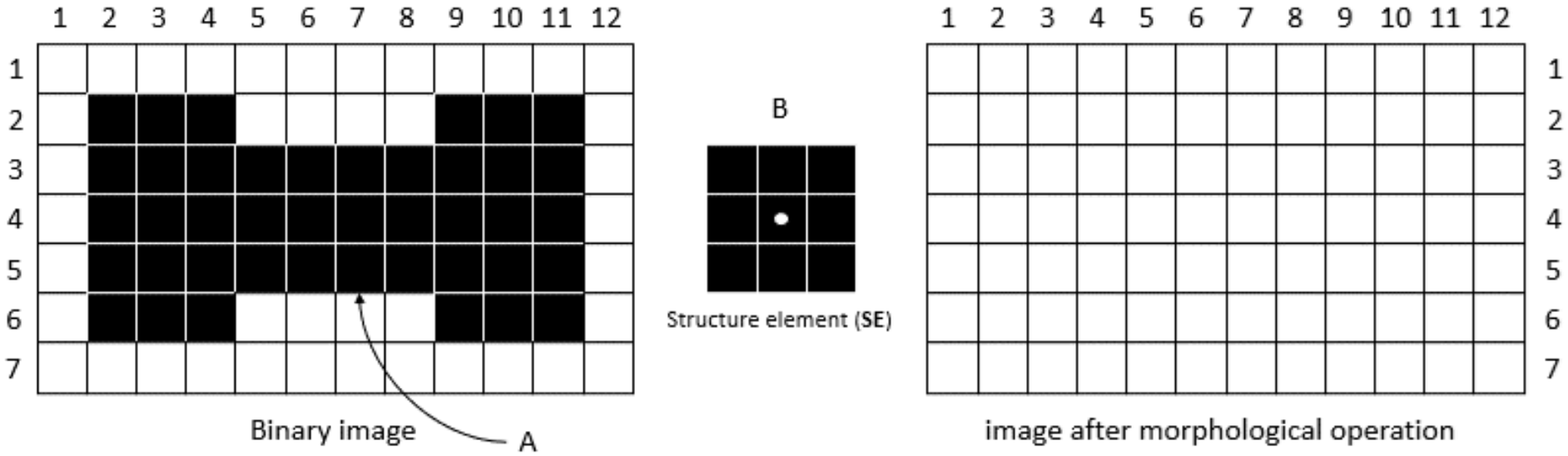
Try to Coding

```
from ImageProc import arucoProjection
import cv2

cam = arucoProjection(0)
threshold_value = 155

While True :
        projection_image ,marker_image = cam.get_projection_image()
        gray_image = cv2.cvtColor(projection_image,cv2.COLOR_BGR2GRAY)
        ret ,thresh_image = cv2.threshold(gray_image,threshold_value,255,cv2.THRESH_BINARY_INV)

        cv2.imshow('projection', projection_image)
        cv2.imshow(‘gray’,gray_image)
        cv2.imshow(‘threshold’,thresh_image)
        cv2.imshow('marker', marker_image)

        key = cv2.waitKey(1) & 0xFF

        if key == ord(‘q’) :
                break

cam.stop_camera()
cv2.destroyAllWindows()
```

Morphological binary image

EROSION

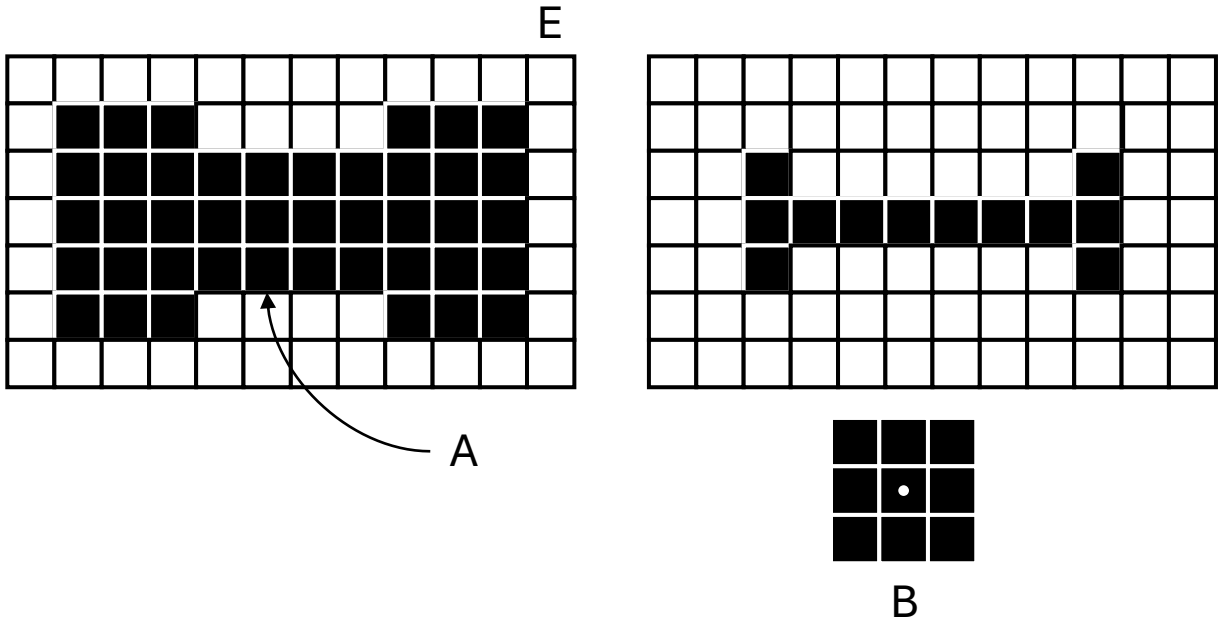
$$A \ominus B = \bigcap_{b \in B} A_{-b}$$



Morphological binary image

EROSION

$$A \ominus B = \bigcap_{b \in B} A_{-b}$$

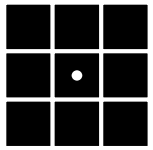


DILATION

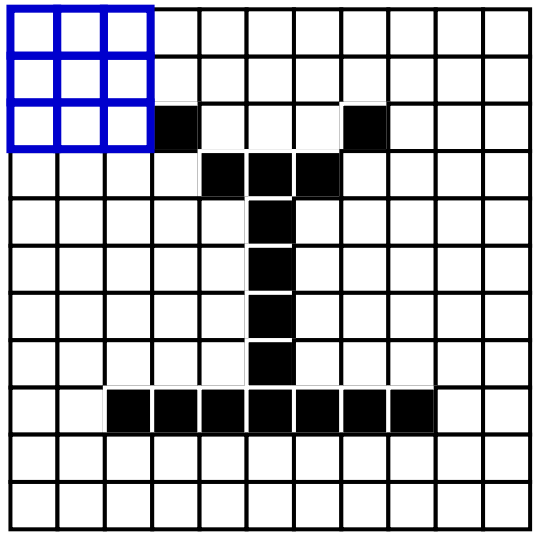
$$A \oplus B = \bigcup_{b \in B} A_{b}$$

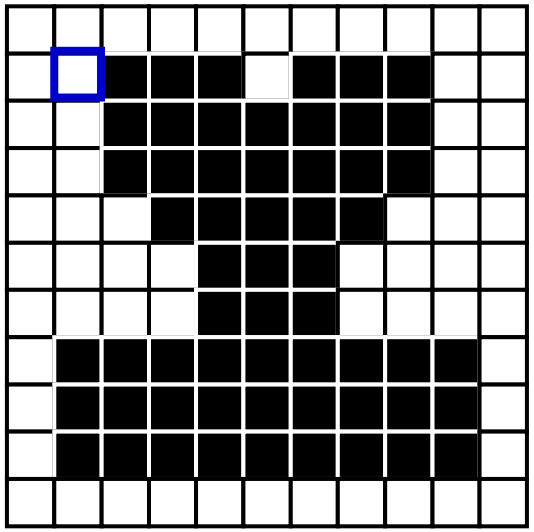
Morphological binary image

$$A \oplus B = \bigcup_{b \in B} A_b$$



B

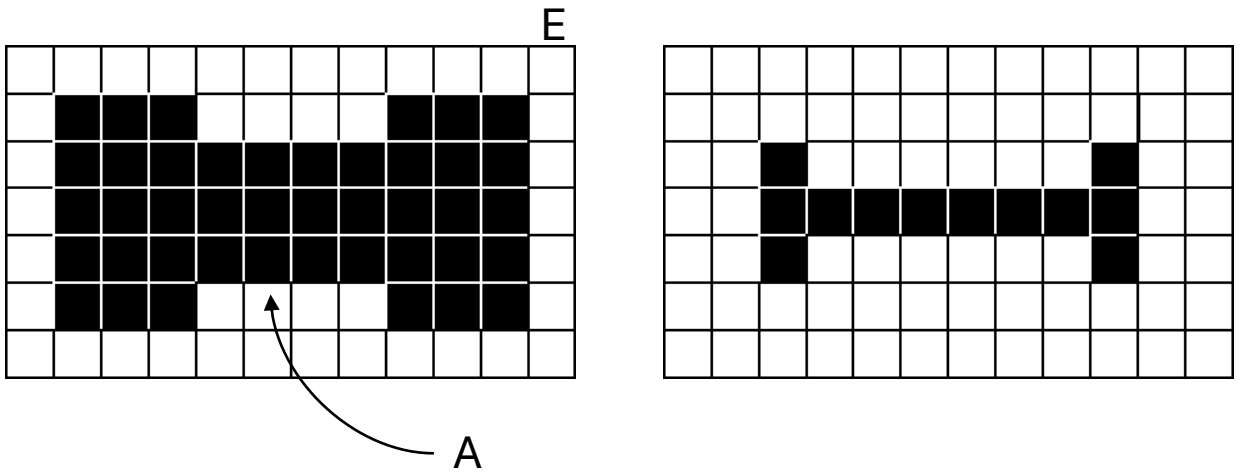




Morphological binary image

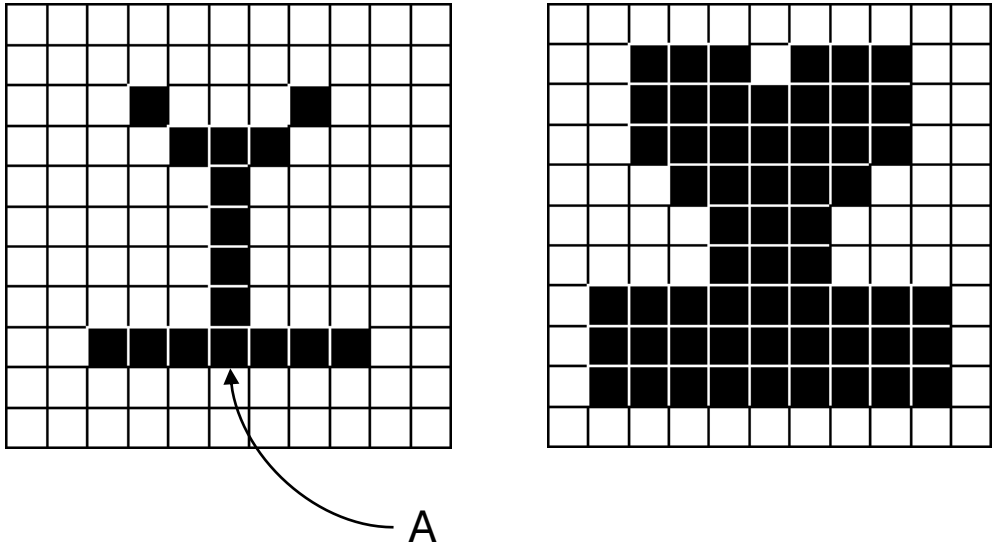
EROSION

$$A \ominus B = \bigcap_{b \in B} A_{-b}$$



DILATION

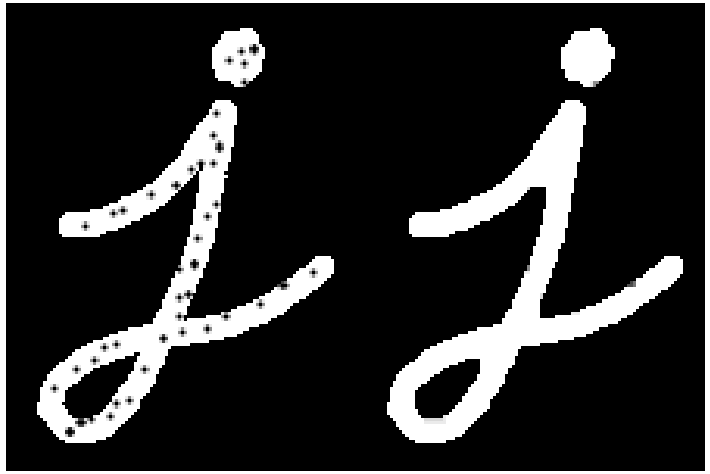
$$A \oplus B = \bigcup_{b \in B} A_{b}$$



Morphological opening and closing

Opening

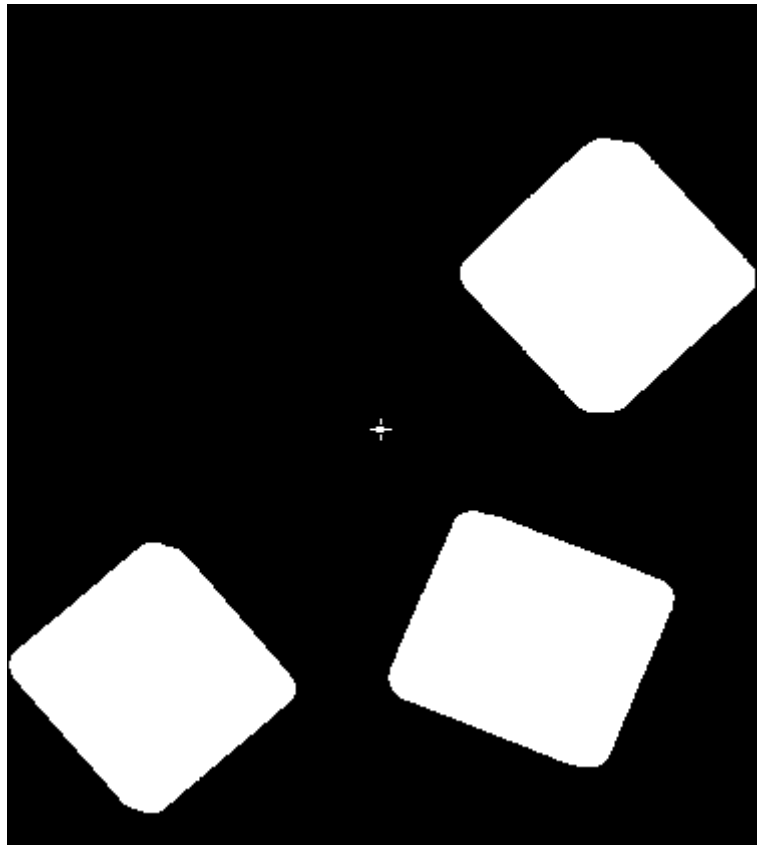
$$A \circ B = (A \ominus B) \oplus B$$



Closing

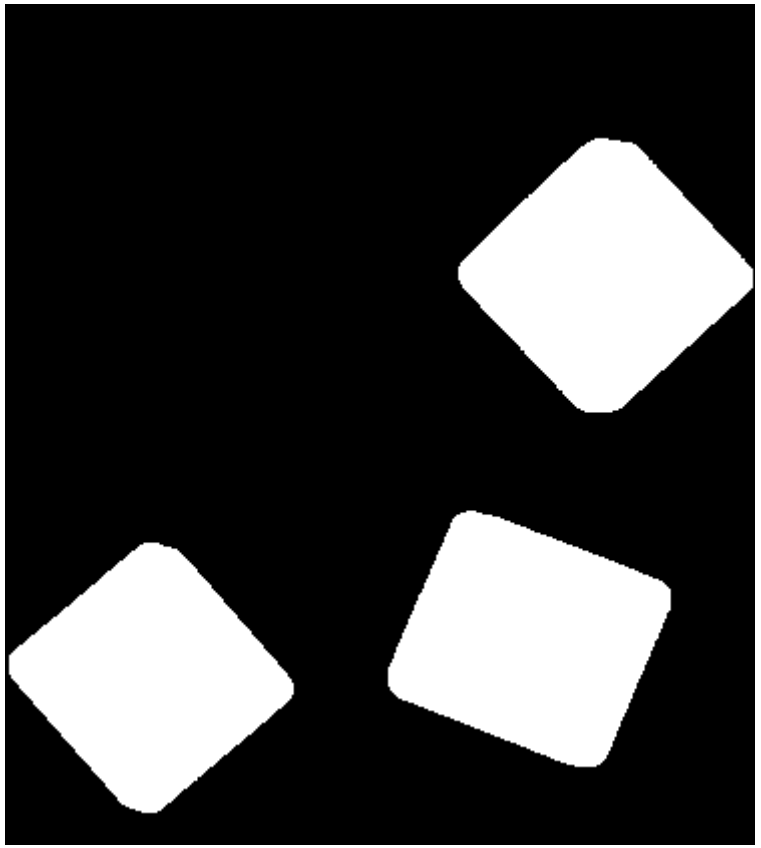
$$A \bullet B = (A \oplus B) \ominus B$$

Morphological binary image



Threshold:155

Morphology

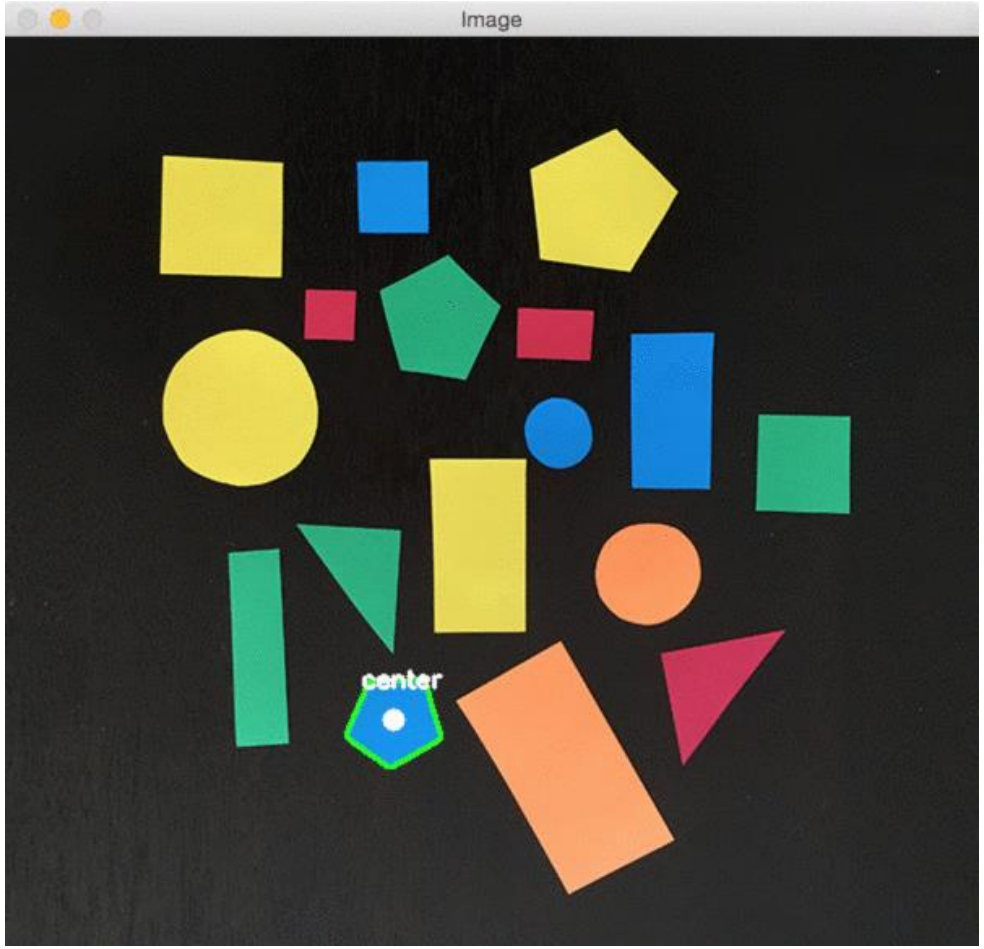


After Morphology

Try to Coding

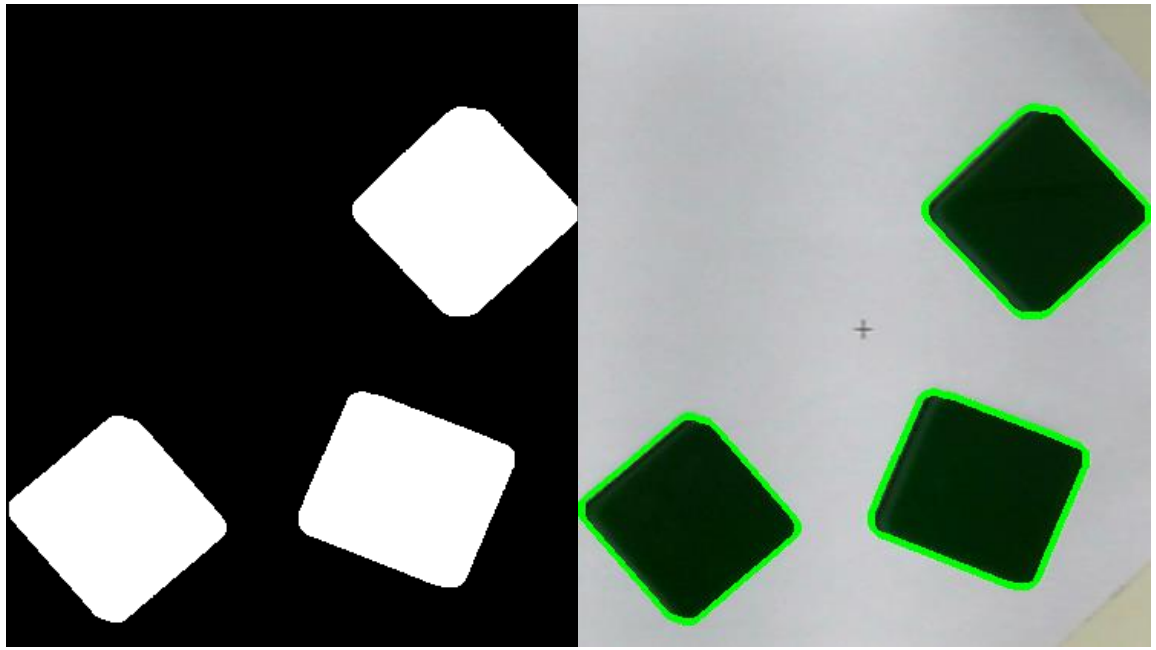
```
from ImageProc import arucoProjection
import cv2

cam = arucoProjection(0)
threshold_value = 155

While True :
        projection_image ,marker_image = cam.get_projection_image()
        gray_image = cv2.cvtColor(projection_image,cv2.COLOR_BGR2GRAY)
        ret ,thresh_image = cv2.threshold(gray_image,threshold_value,255,cv2.THRESH_BINARY_INV)

        kernel = np.ones((5,5),dtype=np.uint8)
        thresh_image = cv2.erode(thresh_image ,kernel ,iterations=1)
        thresh_image = cv2.dilate(thresh_image ,kernel ,iterations=1)

        cv2.imshow('projection', projection_image)
        cv2.imshow(‘gray’,gray_image)
        cv2.imshow(‘threshold’,thresh_image)
        cv2.imshow('marker', marker_image)

        key = cv2.waitKey(1) & 0xFF

        if key == ord(‘q’) :
                break

cam.stop_camera()
cv2.destroyAllWindows()
```

Contour Detection



https://www.pyimagesearch.com/2016/02/01/opencv-center-of-contour/

Contour Detection



Contour feature:

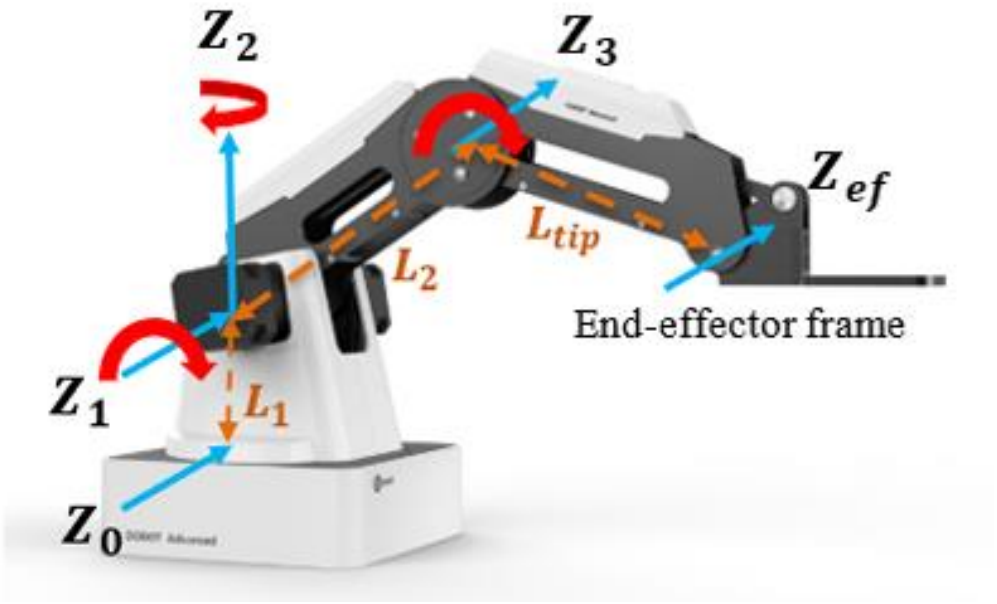
1. **Moments**
2. **Contour Area**
3. **Contour Perimeter**
4. **Contour Approximation (**approximates a contour shape to another shape with less number of vertices)

https://docs.opencv.org/3.4/dd/d49/tutorial_py_contour_features.html

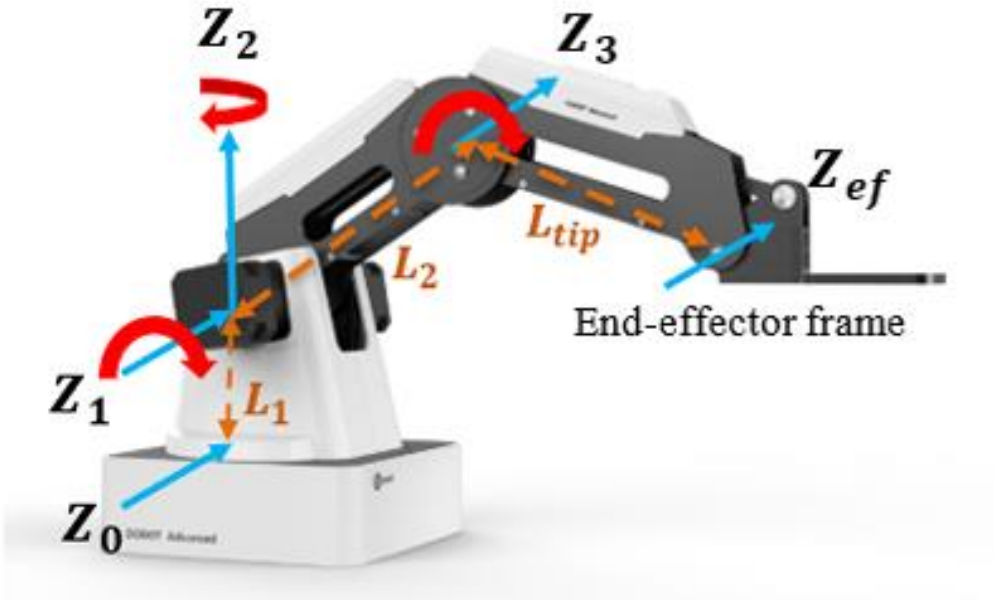
Try to Coding

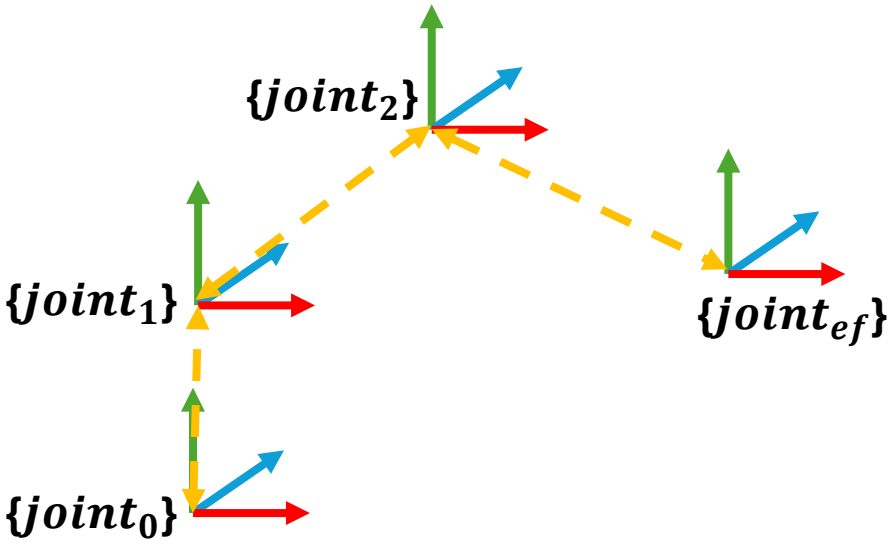
```
from ImageProc import arucoProjection
import cv2

cam = arucoProjection(0)
threshold_value = 155

While True :
        projection_image ,marker_image = cam.get_projection_image()
        gray_image = cv2.cvtColor(projection_image,cv2.COLOR_BGR2GRAY)
        ret ,thresh_image = cv2.threshold(gray_image,threshold_value,255,cv2.THRESH_BINARY_INV)

        kernel = np.ones((5,5),dtype=np.uint8)
        thresh_image = cv2.erode(thresh_image ,kernel ,iterations=1)
        thresh_image = cv2.dilate(thresh_image ,kernel ,iterations=1)

        contours ,hierarchy = cv2.findContours(thresh_image,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
        object_pts ,object_areas = cam.computeContours(contours)

        cv2.drawContours(projection_image ,contours , -1 , (0,255,0) ,3)

        cv2.imshow('projection', projection_image)
        cv2.imshow(‘gray’,gray_image)
        cv2.imshow(‘threshold’,thresh_image)
        cv2.imshow('marker', marker_image)

        key = cv2.waitKey(1) & 0xFF

        if key == ord(‘q’) :
                break

cam.stop_camera()
cv2.destroyAllWindows()
```

Frame and Transformation Matrix

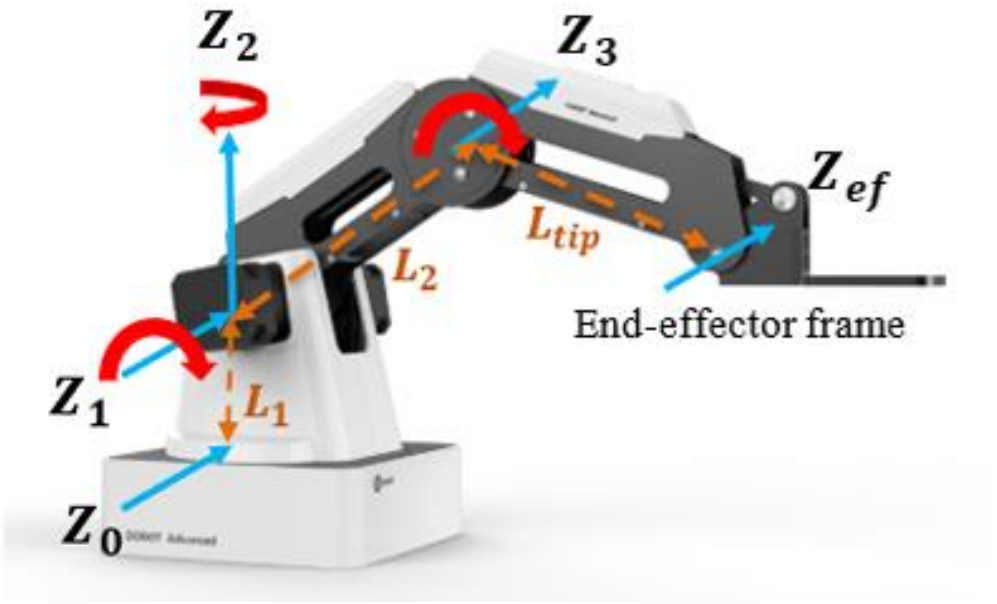


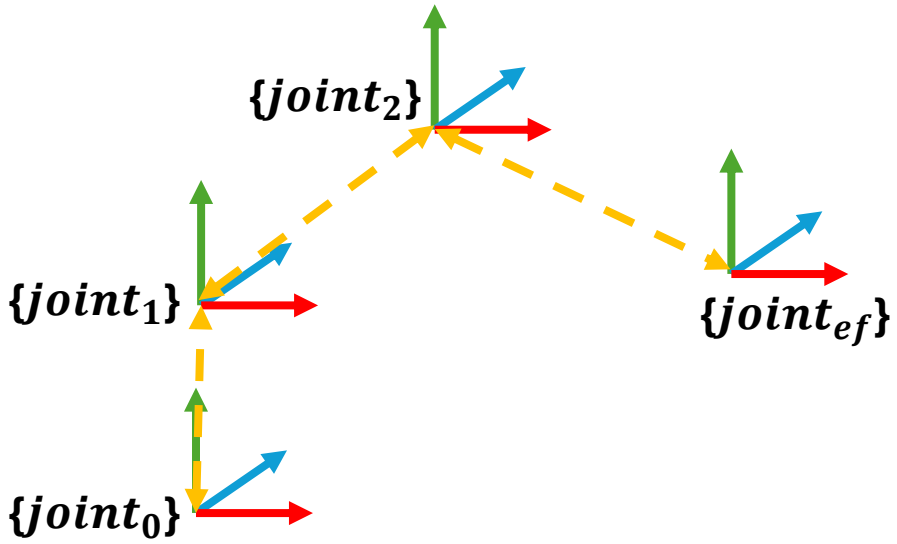
Frame and Transformation Matrix





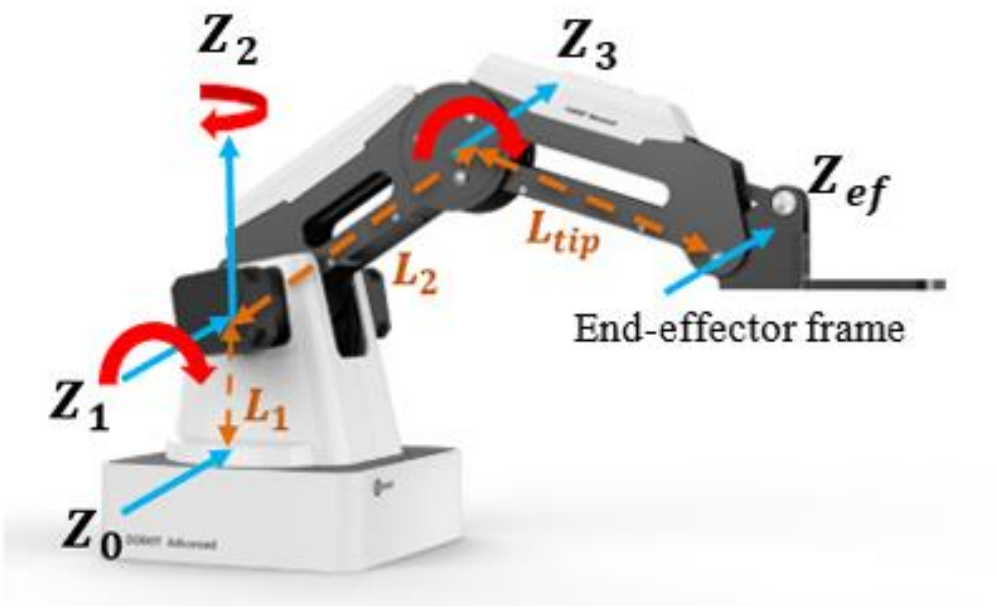
Frame and Transformation Matrix

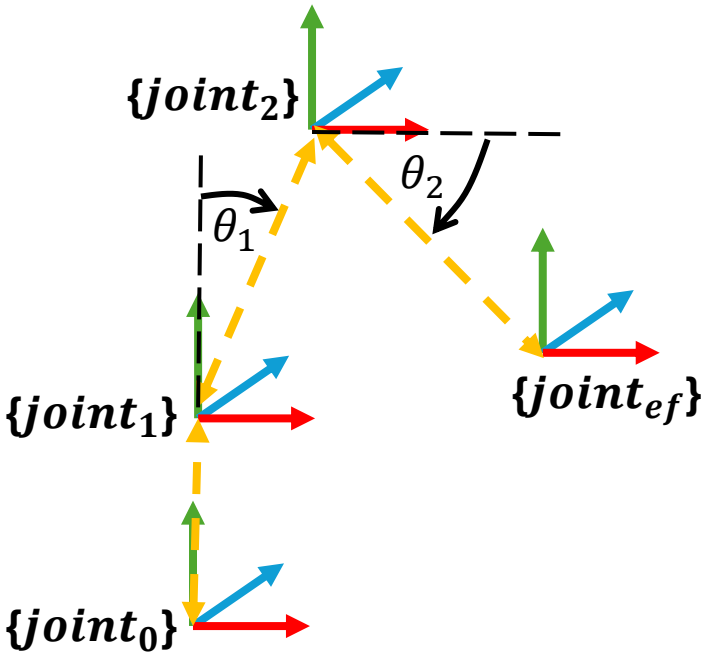




- **Joints Command (forward kinematic)**
- Coordinate Command (inverse kinematic)

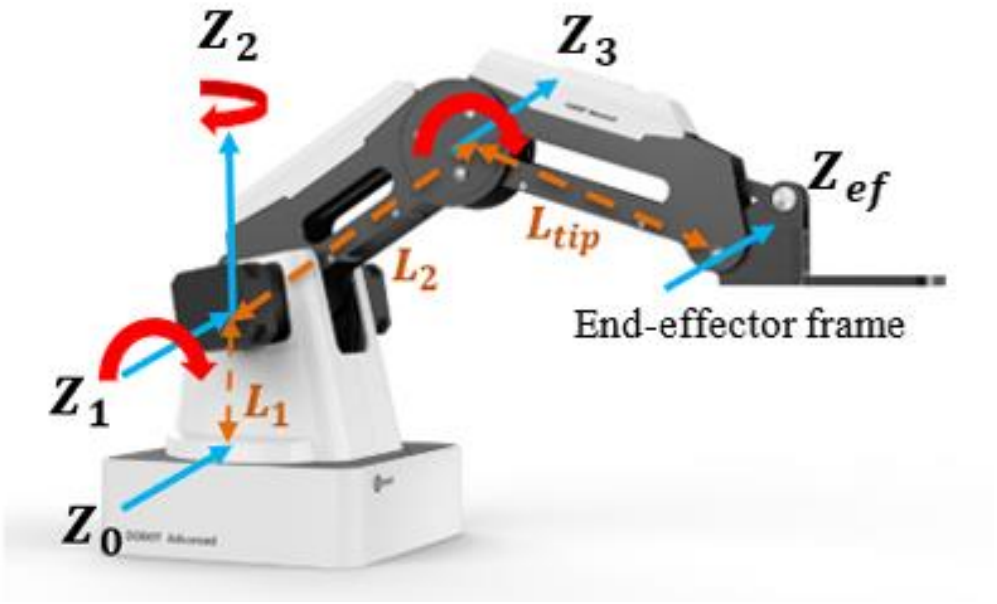
Frame and Transformation Matrix

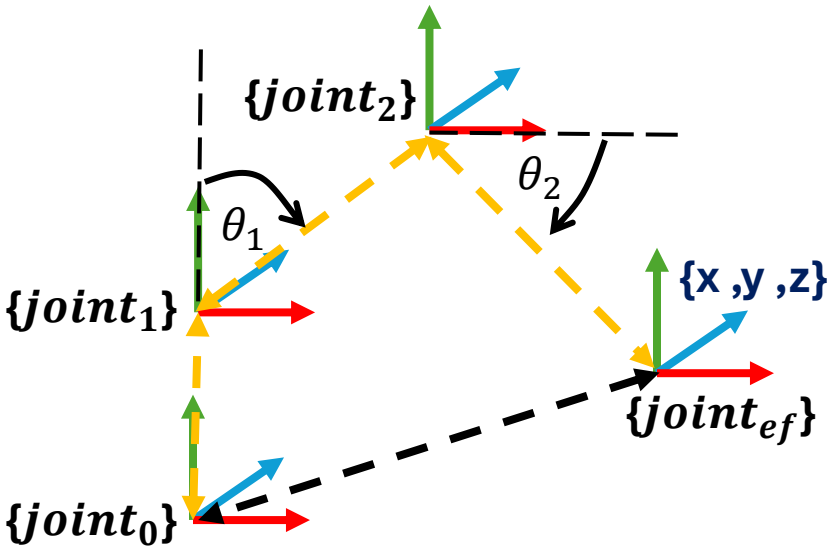




- **Joints Command (forward kinematic)**
- Coordinate Command (inverse kinematic)

Frame and Transformation Matrix





- Joints Command (forward kinematic)
- **Coordinate Command (inverse kinematic)** $\{x, y, z\} \longrightarrow \{\theta_0, \theta_1, \theta_2, \theta_3\}$

input output

Cartesian Trajectory Based Control of Dobot Robot Proceedings of the International Conference on Industrial Engineering and Operations Management Toronto, Canada, October 23-25, 2019

Frame and Transformation Matrix

Command Your Dobot

```
from DobotDriver import DobotDriver

dobot_arm = DobotDriver()

dobot_arm.move_on_robot_coordinate(0.205,0.0,0.15,0.0,wait=True)
```

Frame and Transformation Matrix

Transformation Matrix

$$ {}^{A}_{B}T = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} $$

Rotation Matrix

$$ \boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

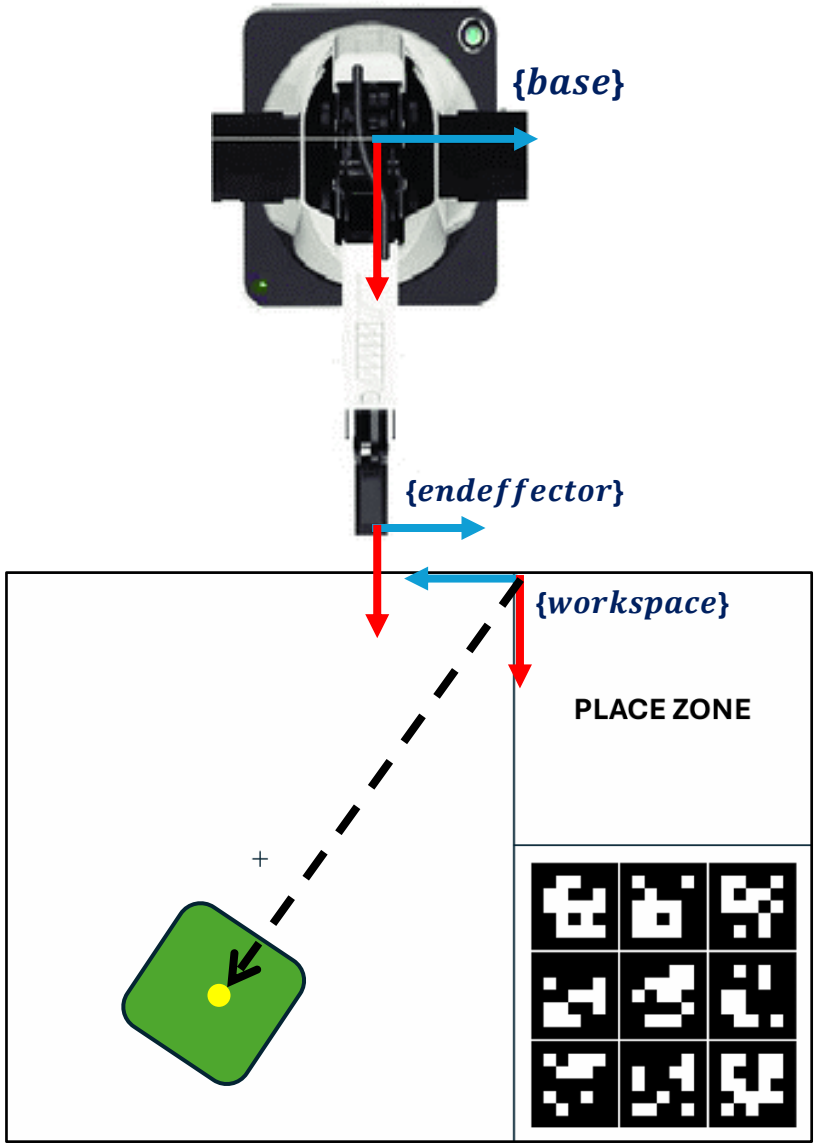
"Descript How a B frame is rotated from an A frame"

Translation Vector

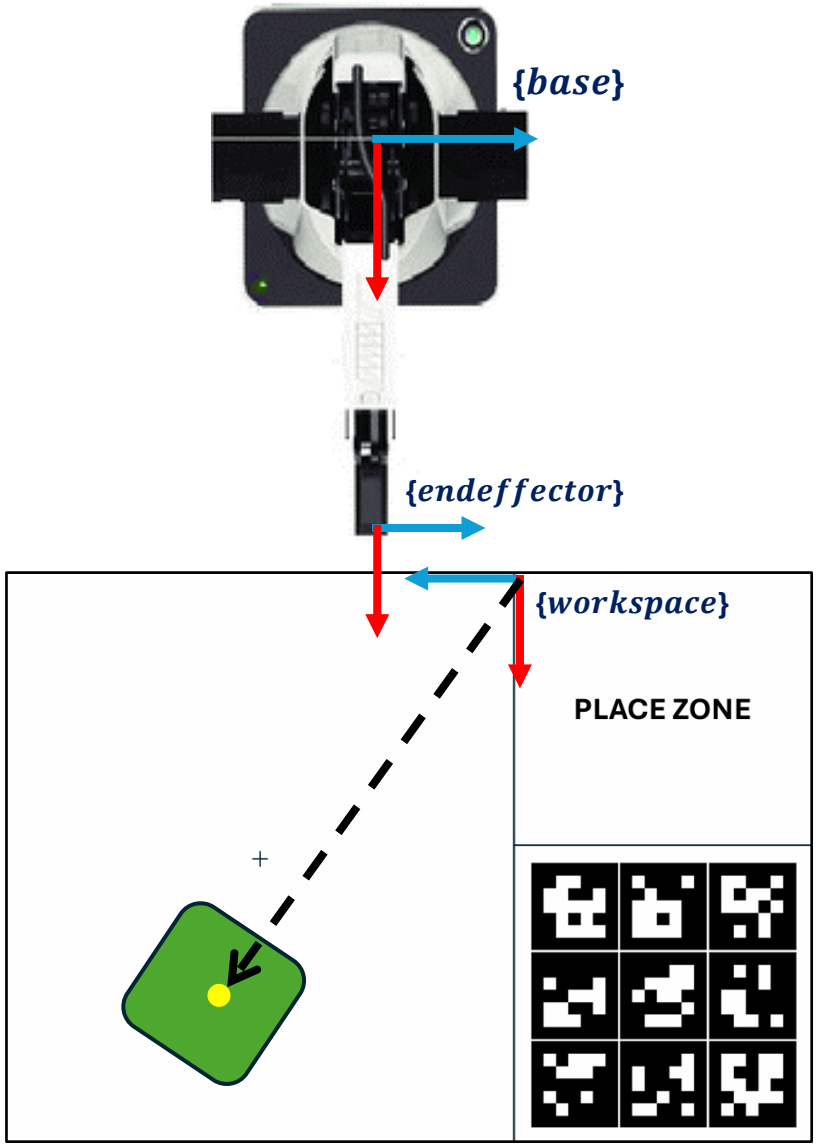
$$ \boldsymbol{T} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} $$

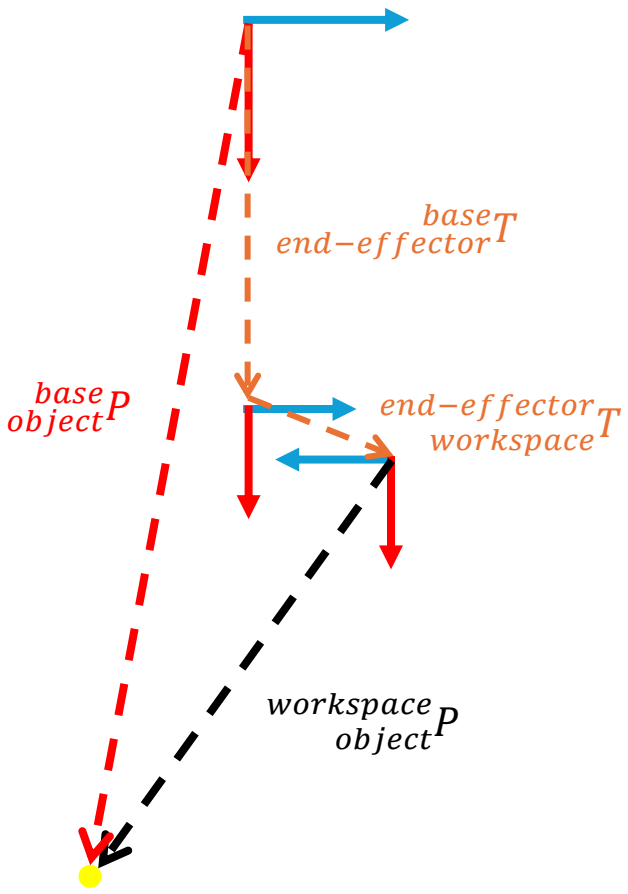
"Descript How a B frame far from an A frame"

Frame and Transformation Matrix



Frame and Transformation Matrix





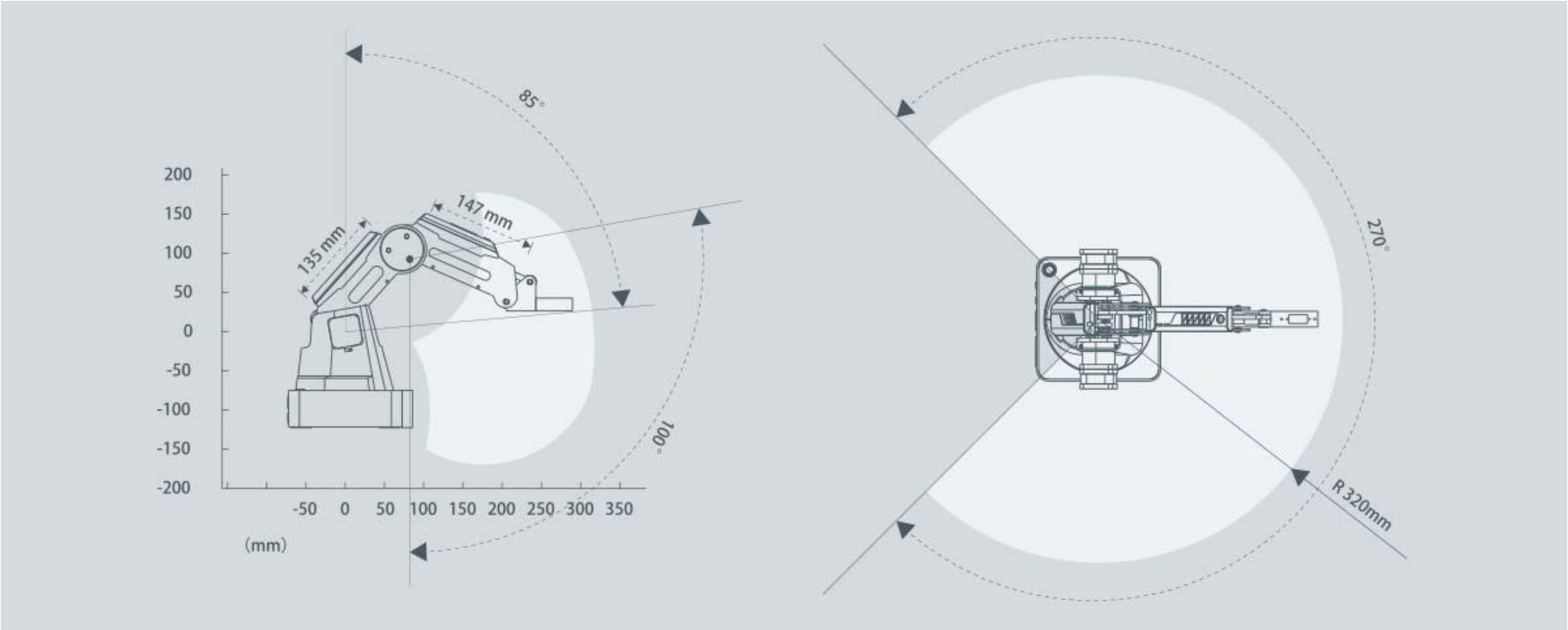
$$^{base}_{object}P = {}^{base}_{end-effector}T \; {}^{end-effector}_{workspace}T \; {}^{workspace}_{object}P$$

Frame and Transformation Matrix

Command Your Dobot

```
from DobotDriver import DobotDriver

dobot_arm = DobotDriver()

dobot_arm.move_on_marker_coordinate(0.1025,0.09,0.01,0.0,wait=True)
```

Workspace



"Your Robot may not move to everywhere"

Cartesian Trajectory Based Control of Dobot Robot Proceedings of the International Conference on Industrial Engineering and Operations Management Toronto, Canada, October 23-25, 2019