ROBOT VISION

AARHUS
UNIVERSITY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

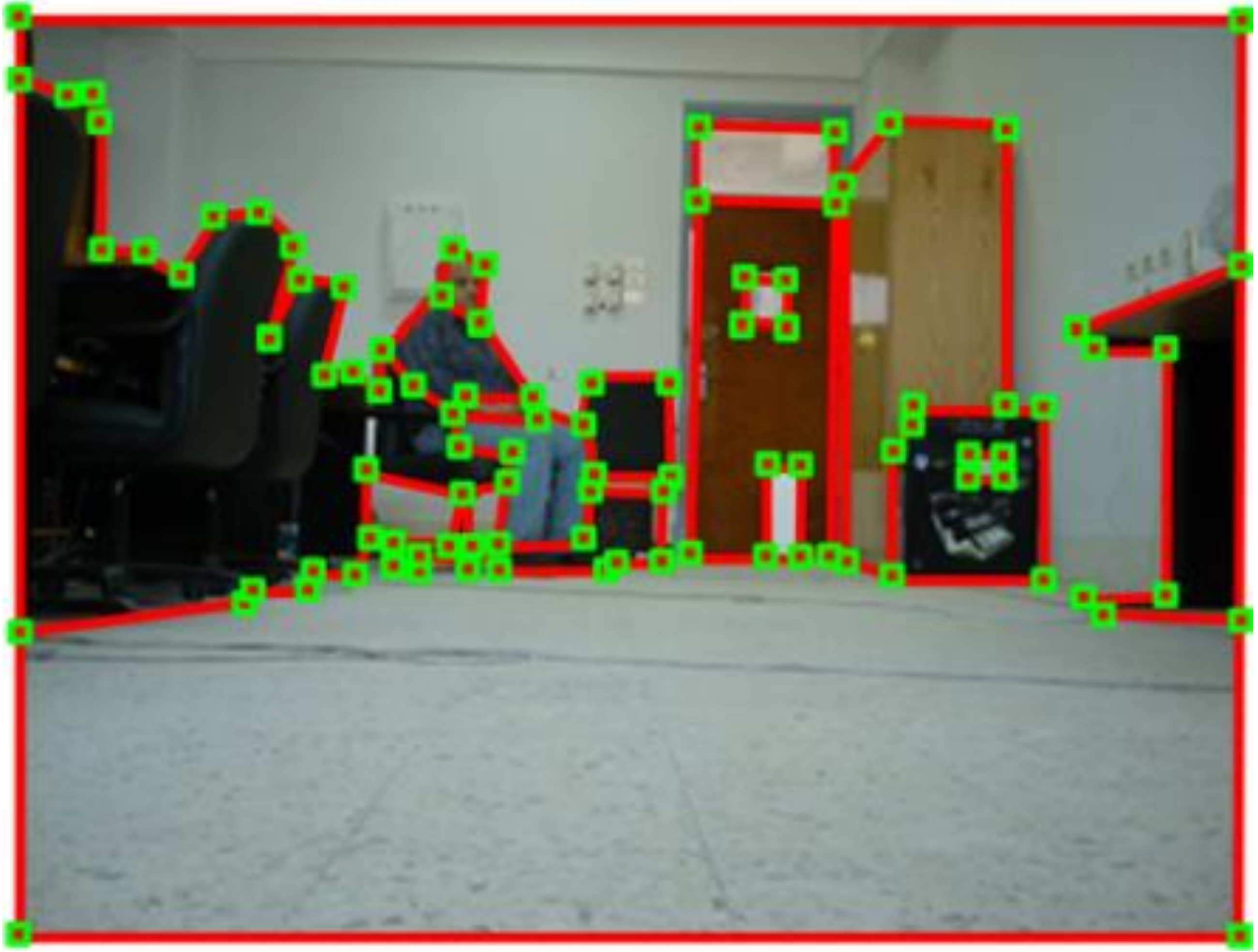
AUTONOMOUS MOBILE ROBOTICS
20 MARCH 2024

ANDRIY SARABAKHA
ASSISTANT PROFESSOR (TENURE TRACK)

SOLIDUM PETIT IN PROFUNDIS · UNIVERSITAS ARHUSIENSIS

ROBOT VISION

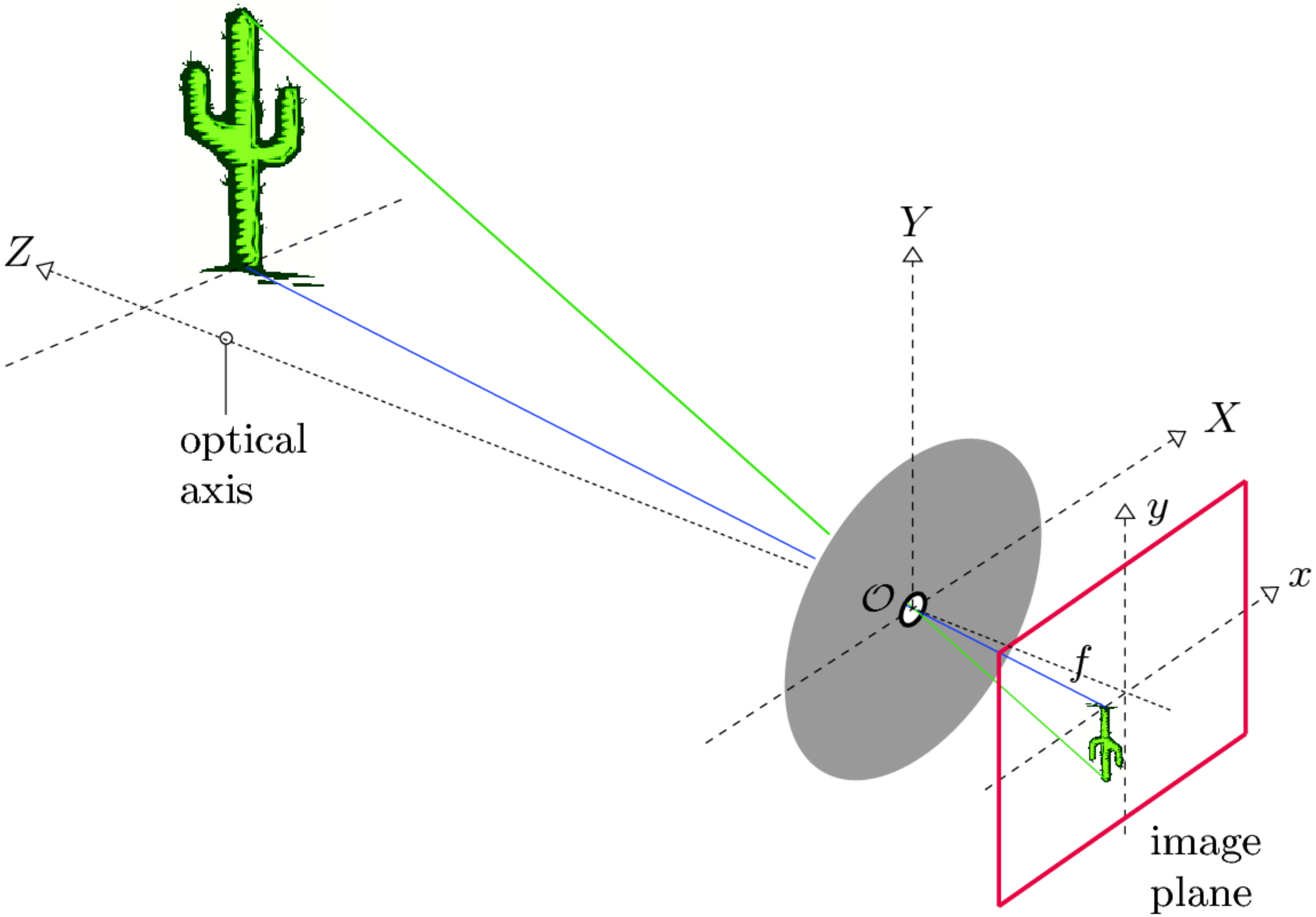
- **Robot vision** is the capability of robots to perceive and interpret visual information from their environment using cameras and image processing algorithms.
- Robot vision allows to
  - understand their surroundings
  - recognize objects
  - navigate in the environments.



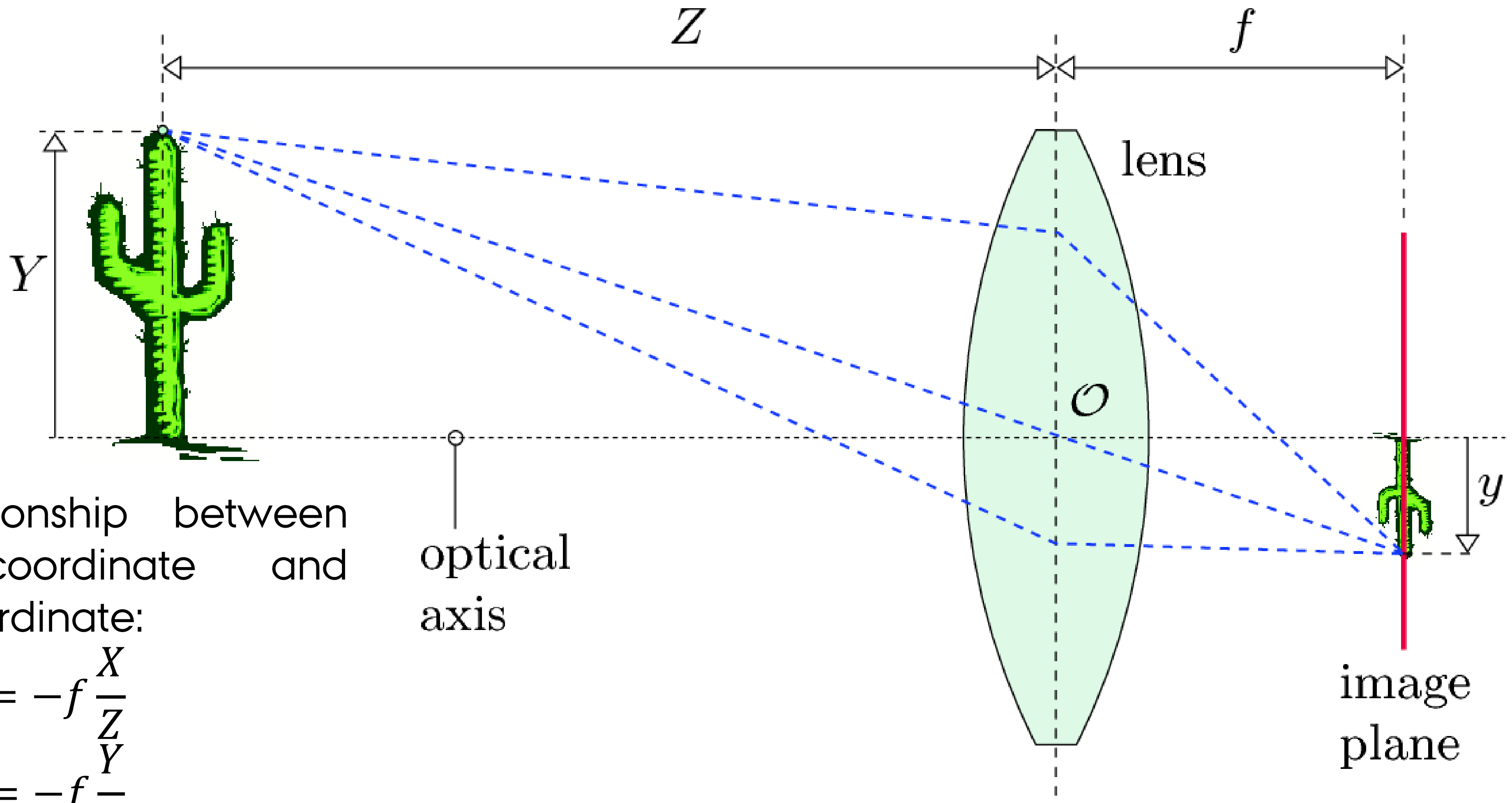
PINHOLE CAMERA

- **Pinhole camera** is the simplest tool to generate images.
- Light from a scene passes through the aperture and projects an inverted image on the *image plane*.
- Camera projects 3D points to a 2D plane:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}$$



THIN LENS CAMERA



- The relationship between object coordinate and image coordinate:

$$x = -f\frac{X}{Z}$$

$$y = -f\frac{Y}{Z}$$

PERSPECTIVE

- **Perspective** is the spatial connection between objects in a photo:

$$\Delta p = f\frac{H}{D}$$

  - Δp – height in pixels
  - f – focal length
  - H – height of the object
  - D – distance to the object

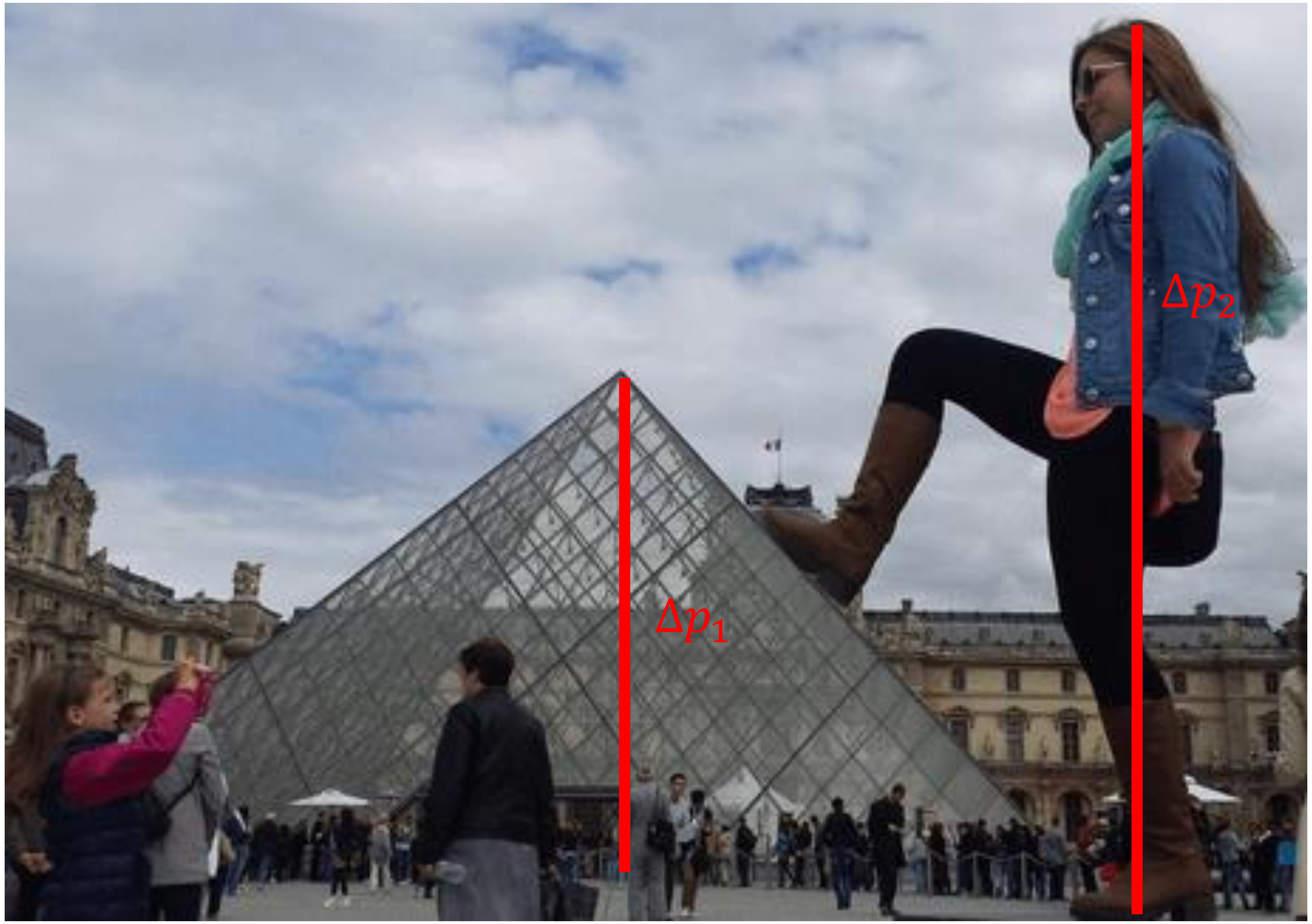
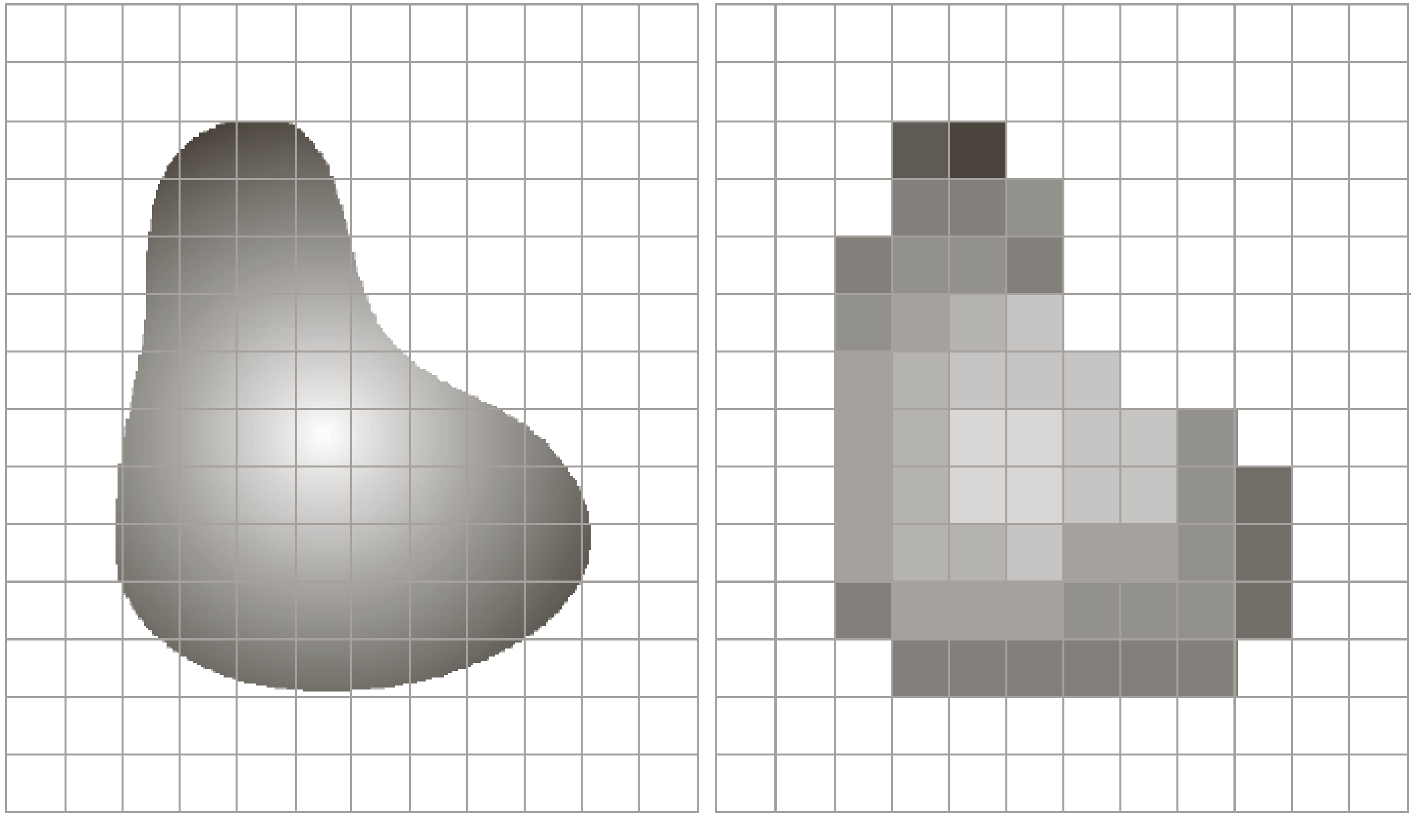


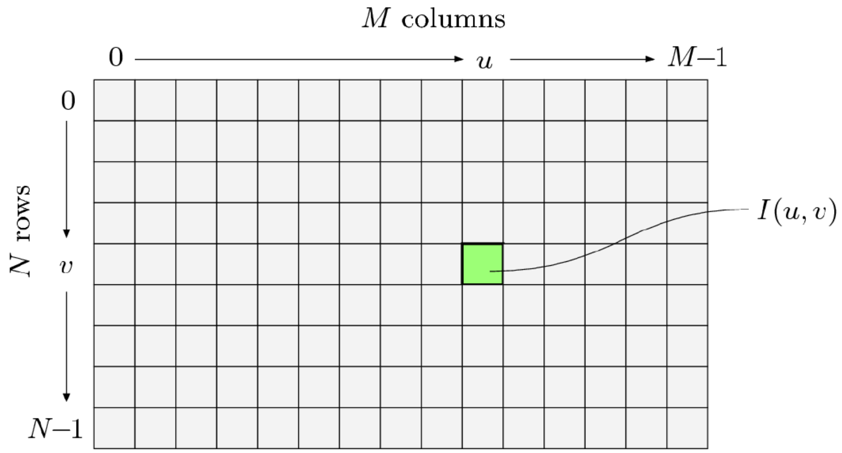
IMAGE DIGITALISATION

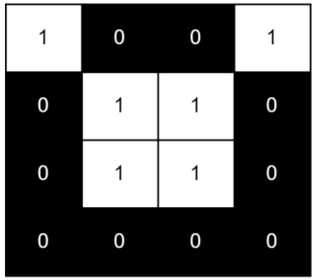
- **Sampling** corresponds to a discretization of the space.
- **Quantization** is a discretization of the intensity values.



DIGITAL IMAGES

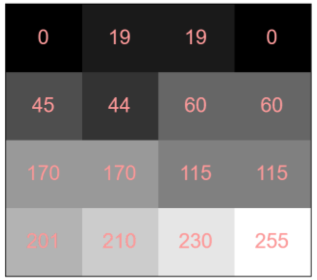
- **Digital image** is a presentation of an image in a form which computer can store (i.e., in form of numbers).
- Digital images are basically of three types:
  - monochrome,
  - grayscale images,
  - color images.





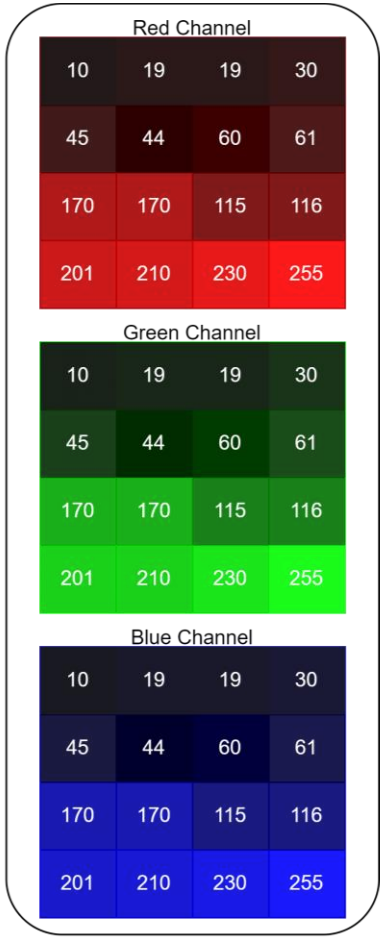
1	0	0	1
0	1	1	0
0	1	1	0
0	0	0	0

Binary image



0	19	19	0
45	44	60	60
170	170	115	115
201	210	230	255

Grayscale image



Red Channel

10	19	19	30
45	44	60	61
170	170	115	116
201	210	230	255

Green Channel

10	19	19	30
45	44	60	61
170	170	115	116
201	210	230	255

Blue Channel

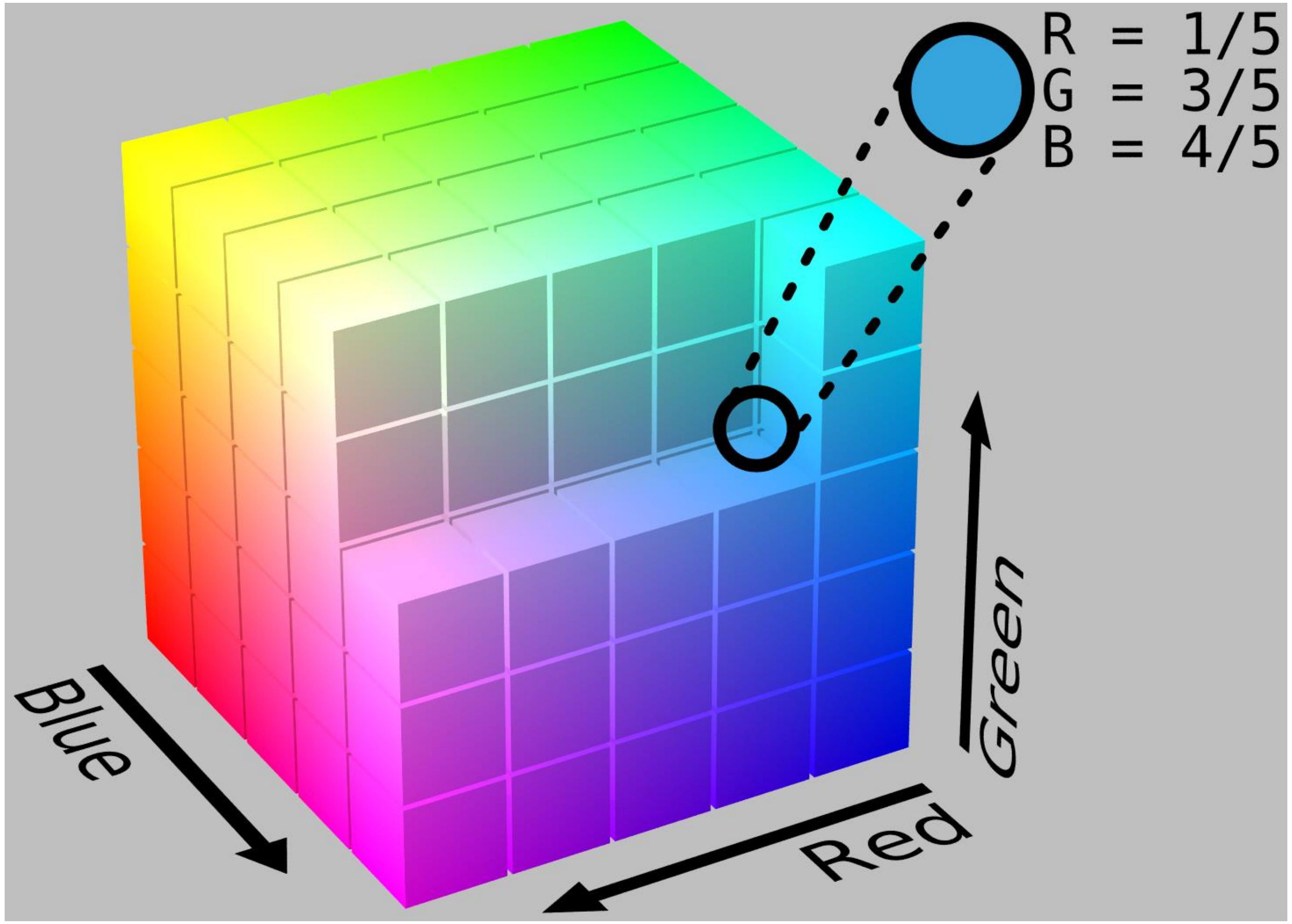
10	19	19	30
45	44	60	61
170	170	115	116
201	210	230	255

RGB image

RGB COLOR MODEL

- **RGB color model** is based on the mixing of primary colors **R**ed, **G**reen and **B**lue to produce a wide array of colors.
- Conversion to grayscale:

$$\text{grayscale} = \frac{R + G + B}{3}$$

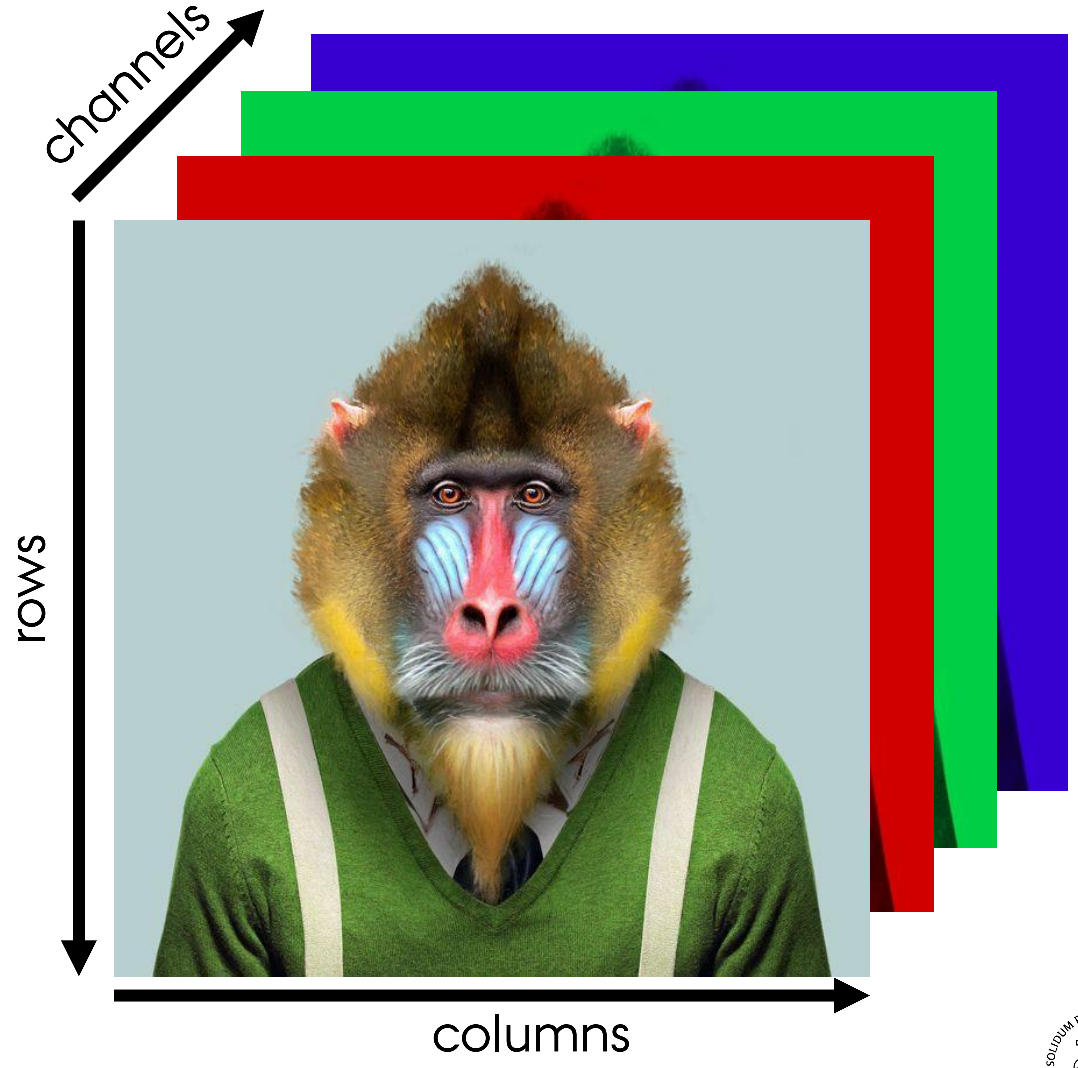


RGB IMAGE

- RGB image is a 3-dimensional array:
  [row, column, channel]

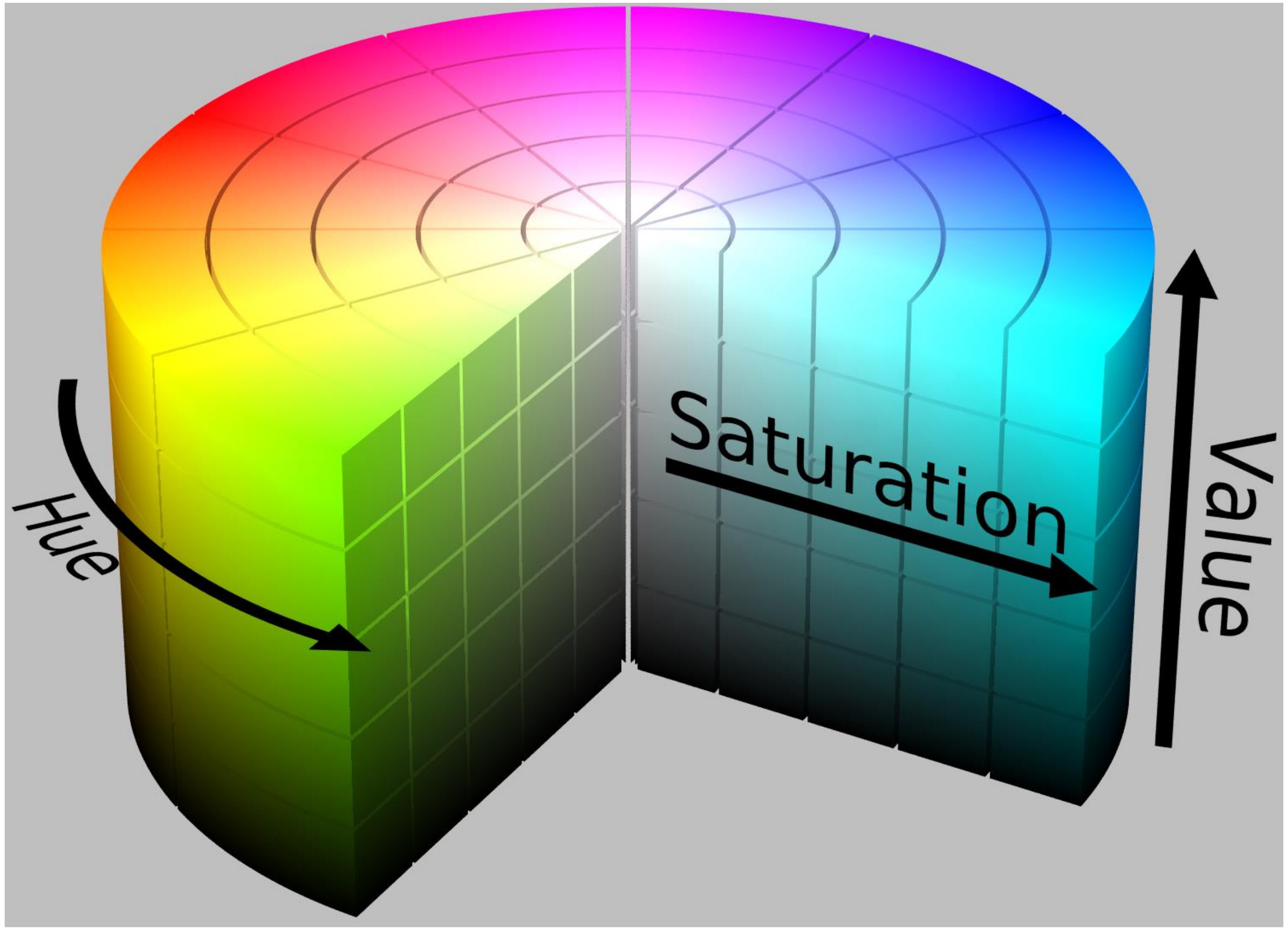
- In MATLAB:

```
red = RGB(:,:,1);
green = RGB(:,:,2);
blue = RGB(:,:,3);
```



HSV COLOR MODEL

- **HSV** (**H**ue, **S**aturation, **V**alue) color model is more intuitive than RGB color model.
- **Hue** corresponds to the color's position on the color wheel.
- **Saturation** corresponds to the color intensity.
- **Value** corresponds to the color brightness.

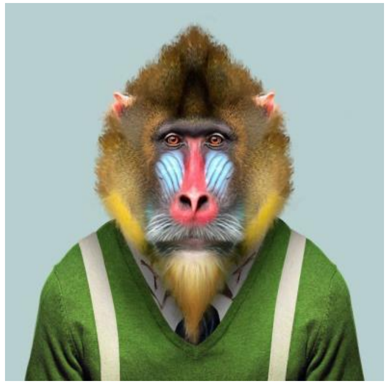


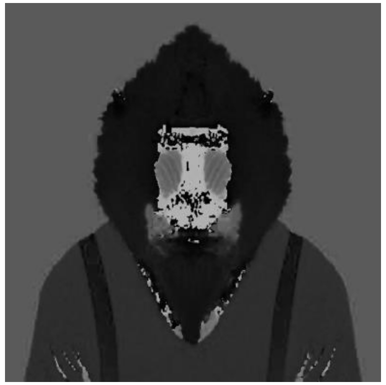
HSV IMAGE

In MATLAB:

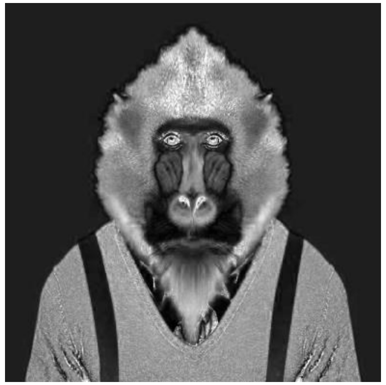
```
HSV = rgb2hsv(RGB);
hue = HSV(:,:,1);
saturation = HSV(:,:,2);
value = HSV(:,:,3);
```

RGB

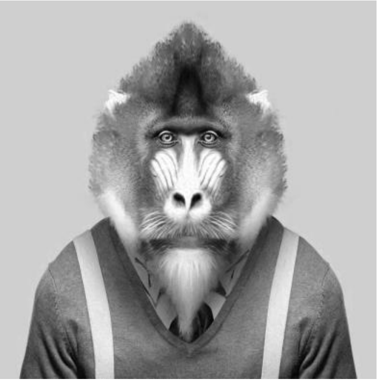




hue



saturation



value

BRIGHTNESS

- **Brightness** of an image is the average over all pixel intensities:

$$B(I) = \frac{1}{3 \cdot w \cdot h} \sum_{u=1}^{h} \sum_{v=1}^{w} \sum_{c=1}^{3} I(u, v, c)$$

low brightness

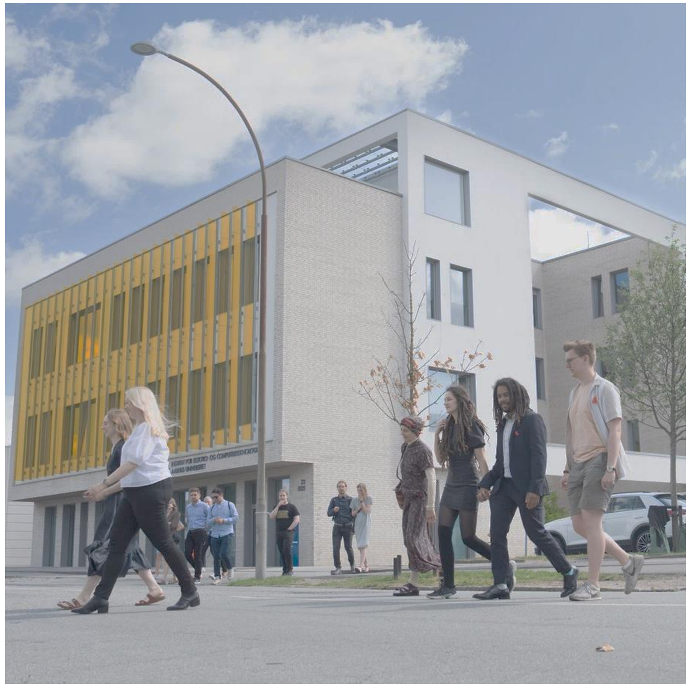
normal brightness

high brightness

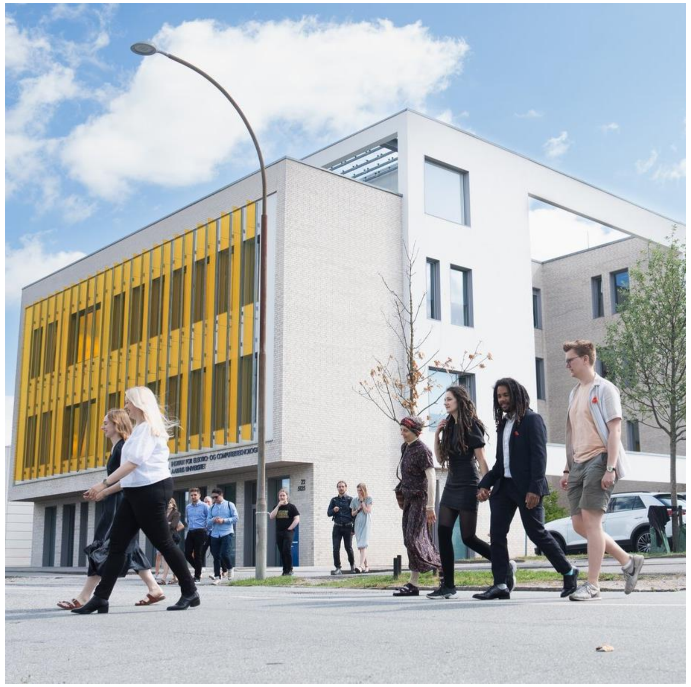
CONTRAST

- **Contrast** of an image is the amount to which different objects in the image can be visually distinguished from one another:

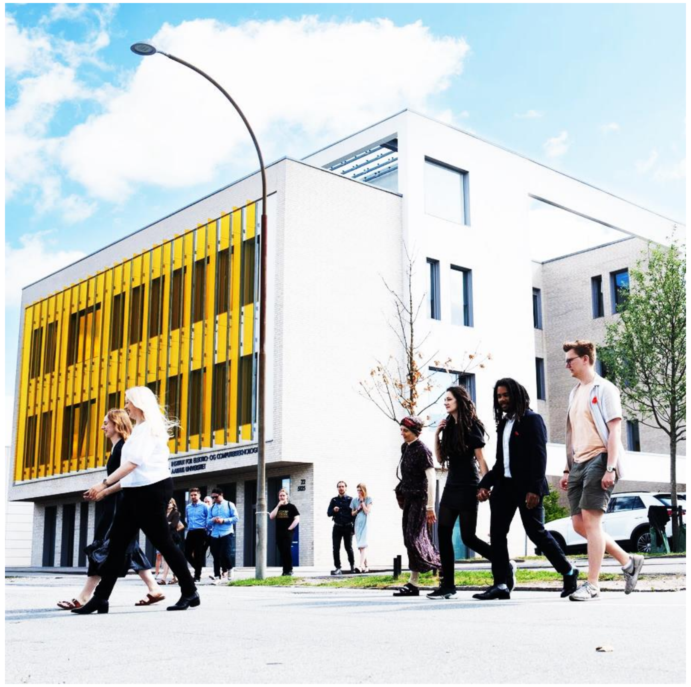
$$C(I) = \frac{\max I - \min I}{\max I + \min I}$$



low contrast



normal contrast



high contrast

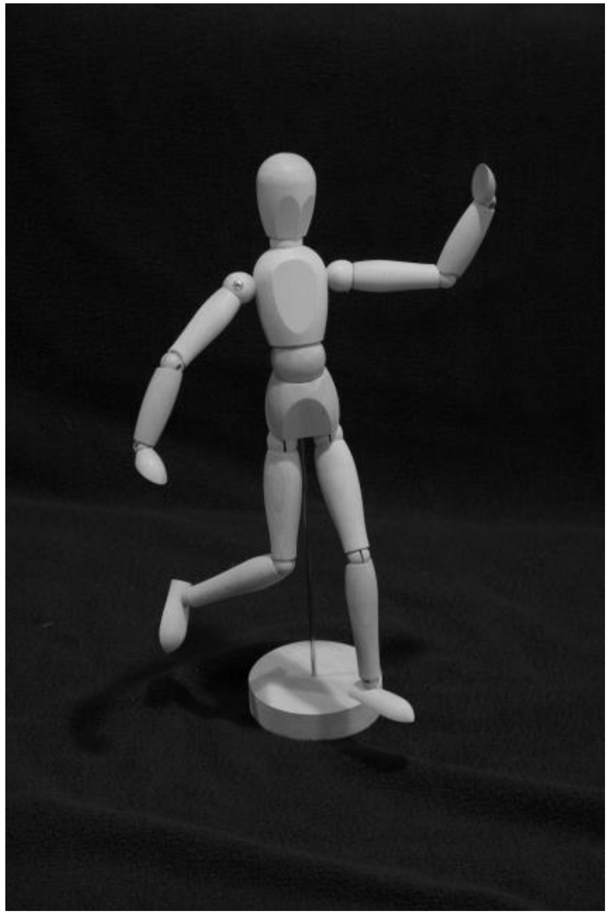
THRESHOLDING

- **Thresholding** converts a grayscale image into a binary image by making every pixel below some threshold a equal to 0 and every pixel above that threshold a equal to 1:

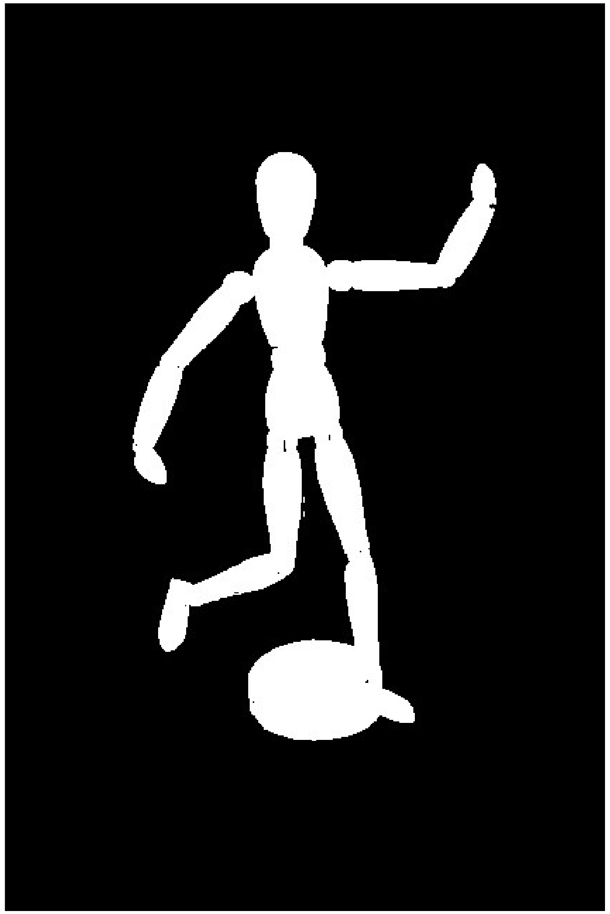
$$f(p) = \begin{cases} 0, \text{if } p < a \\ 1, \text{if } p \geq a \end{cases}$$

- Thresholding in MATLAB:

```
imgBW = imgGrey > a;
```



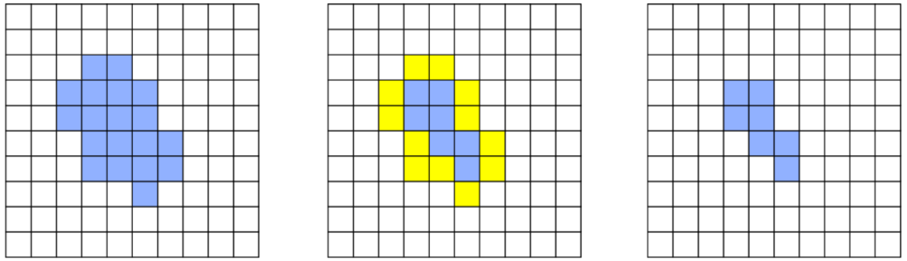
grey-scale image



binary image

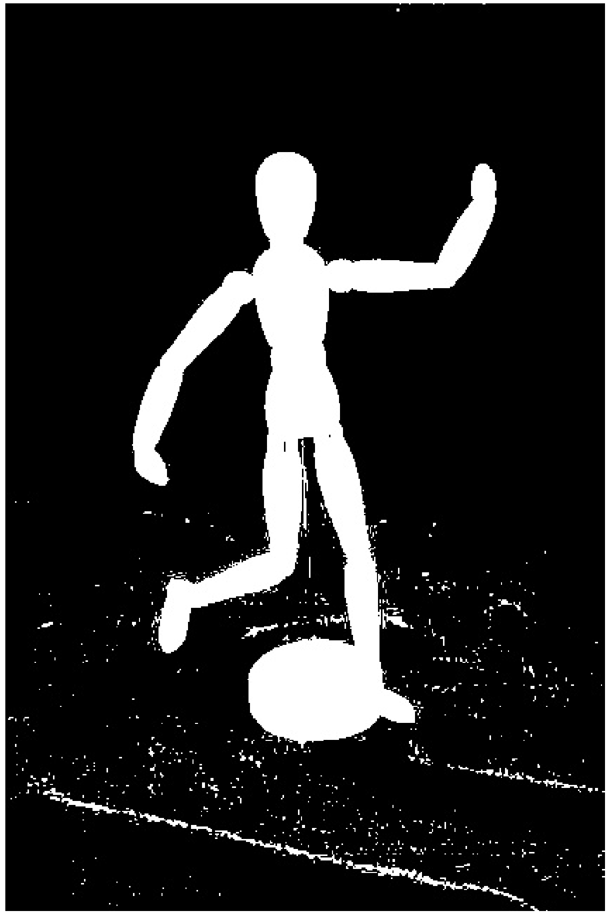
EROSION

- **Erosion** changes a foreground (white) pixel to background (black) if it has a background pixel as a 4-neighbor.

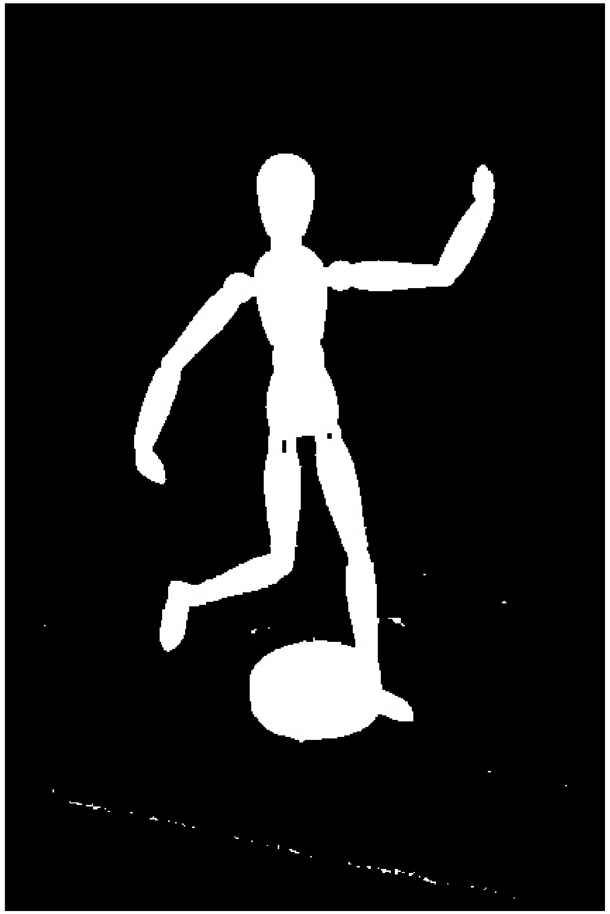


- In MATLAB:

```
J = imdilate(I,SE)
```



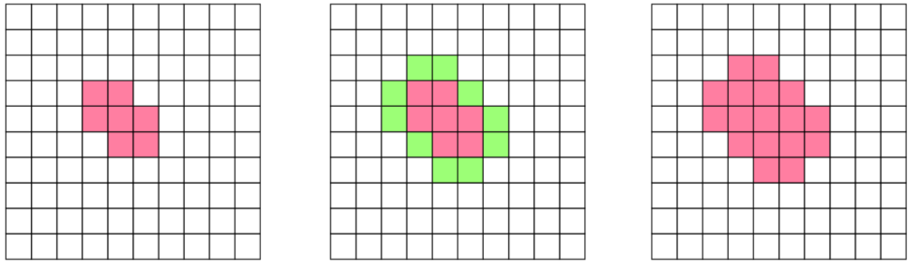
before erosion



after erosion

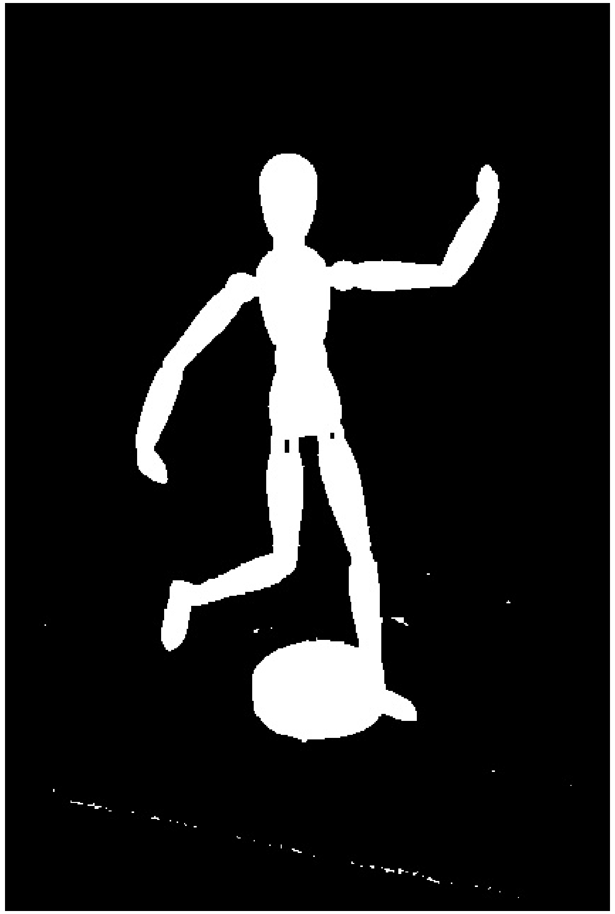
DILATION

- **Dilation** changes a background (black) pixel to foreground (white) if it has a foreground pixel as a 4-neighbor.

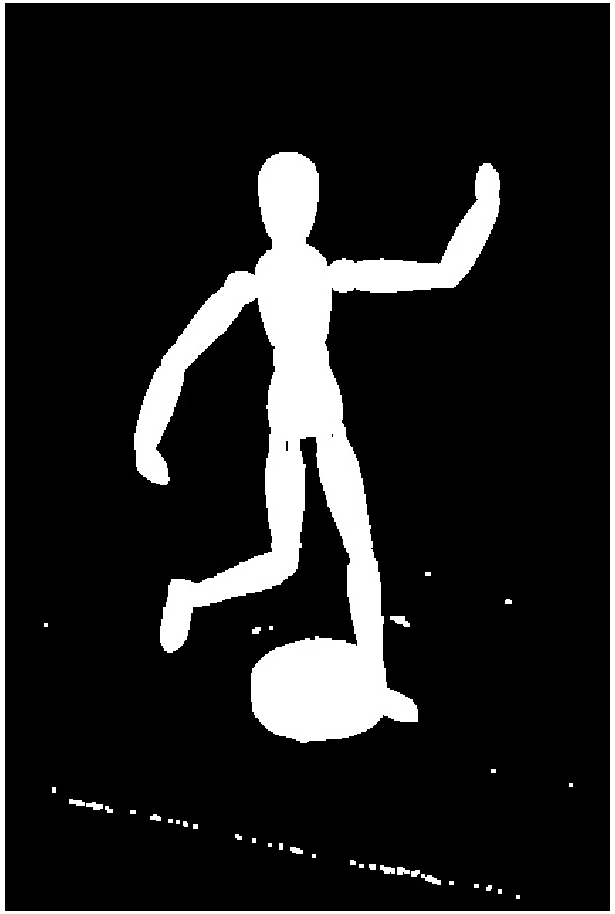


- In MATLAB:

```
J = imerode(I,SE)
```



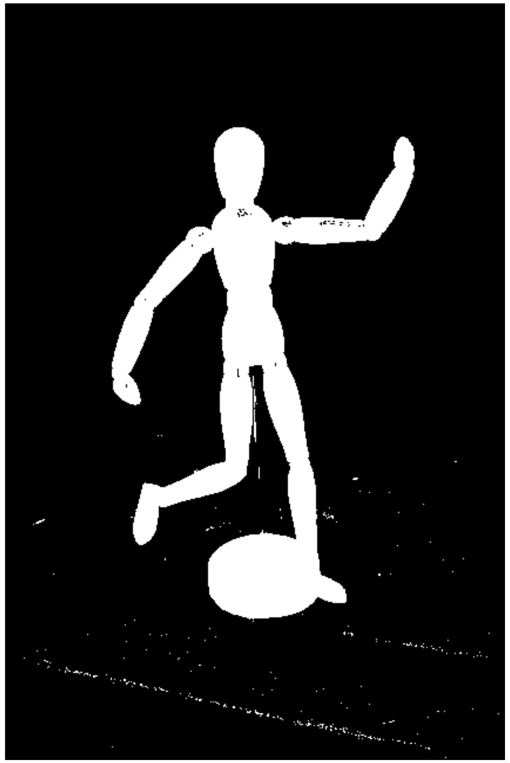
before dilation



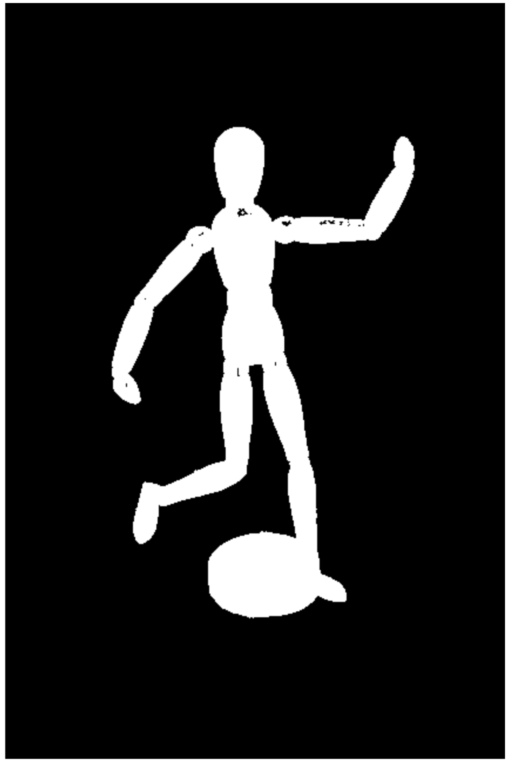
after dilation

OTHER MORPHOLOGICAL OPERATIONS

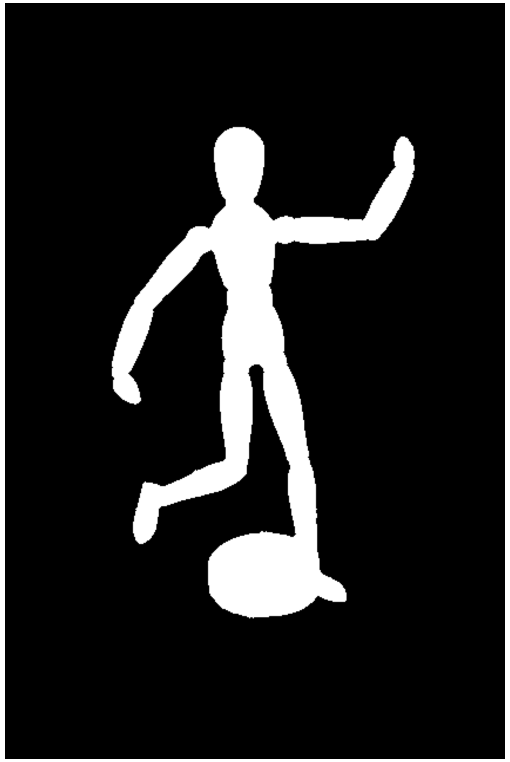
- **Opening**: erosion followed by dilation. Useful for removing small elements.
- **Closing**: dilation followed by erosion. Useful for closing holes.



original image



after opening

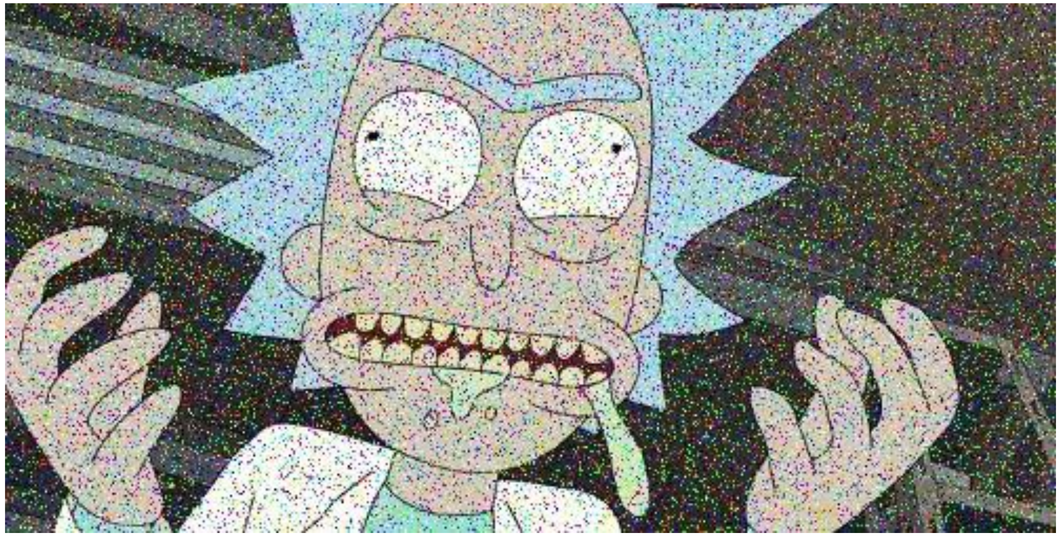


after closing

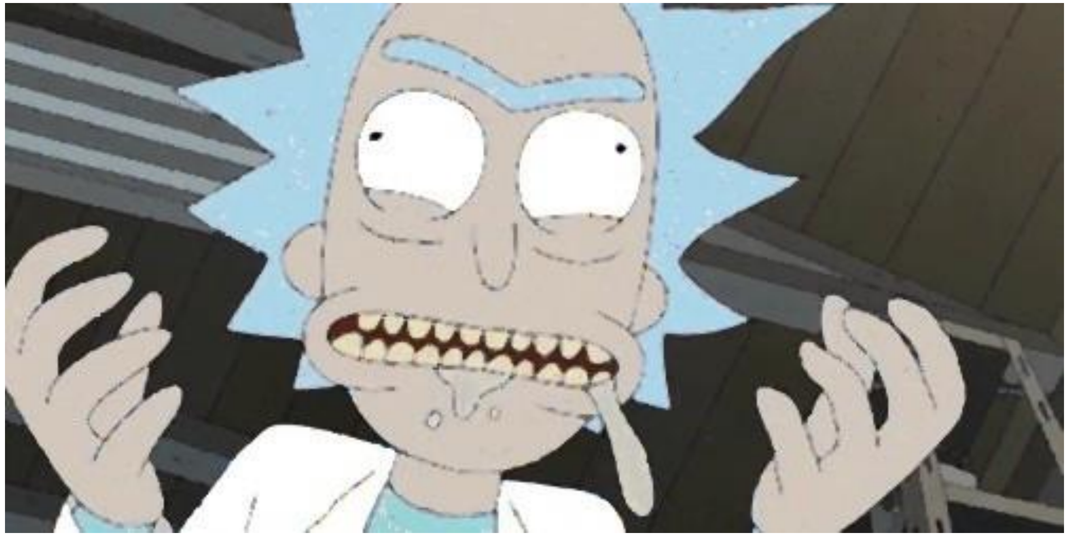
MEDIAN FILTER

- **Median filter** calculates median value for each pixel.
- It is useful to remove white noise.
- In MATLAB:

```
J = medfilt2(I)
```



noisy image

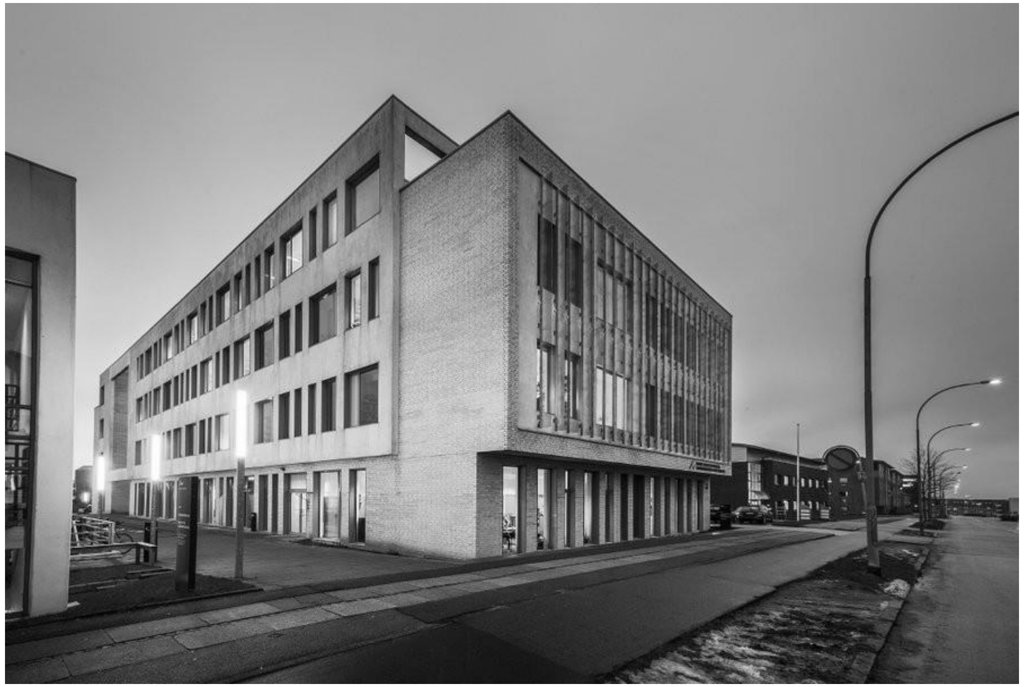


filtered image

SOBEL FILTER

- **Sobel filter** is a popular image processing technique used for detecting edges in images.
- In MATLAB:

```
BW = edge(I)
```



original image

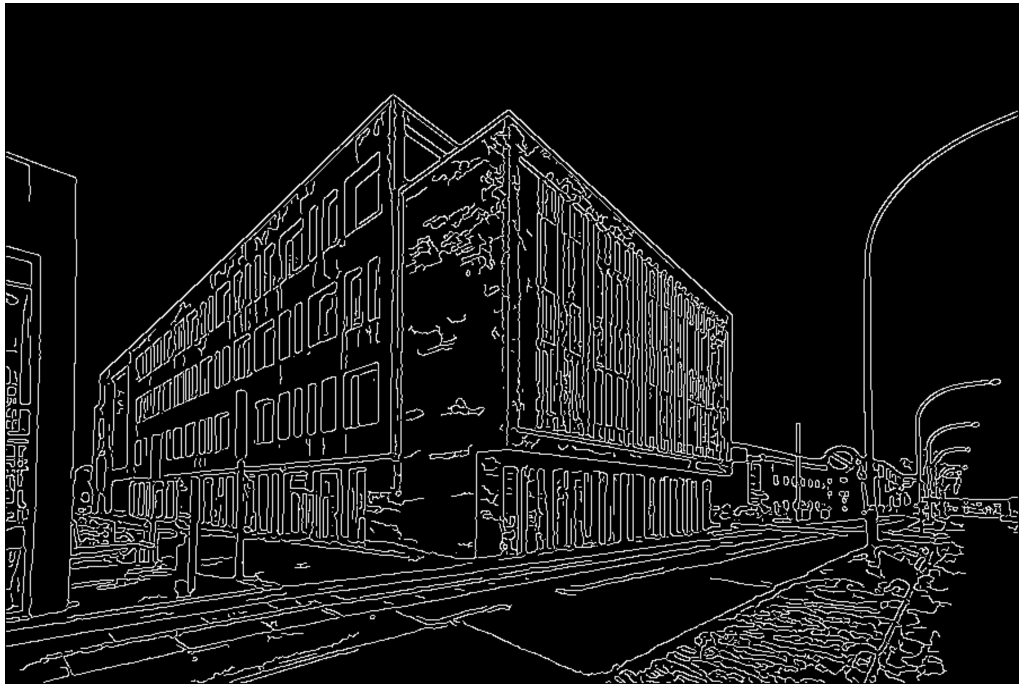


image edges

AARHUS
UNIVERSITY