Should I say hello?

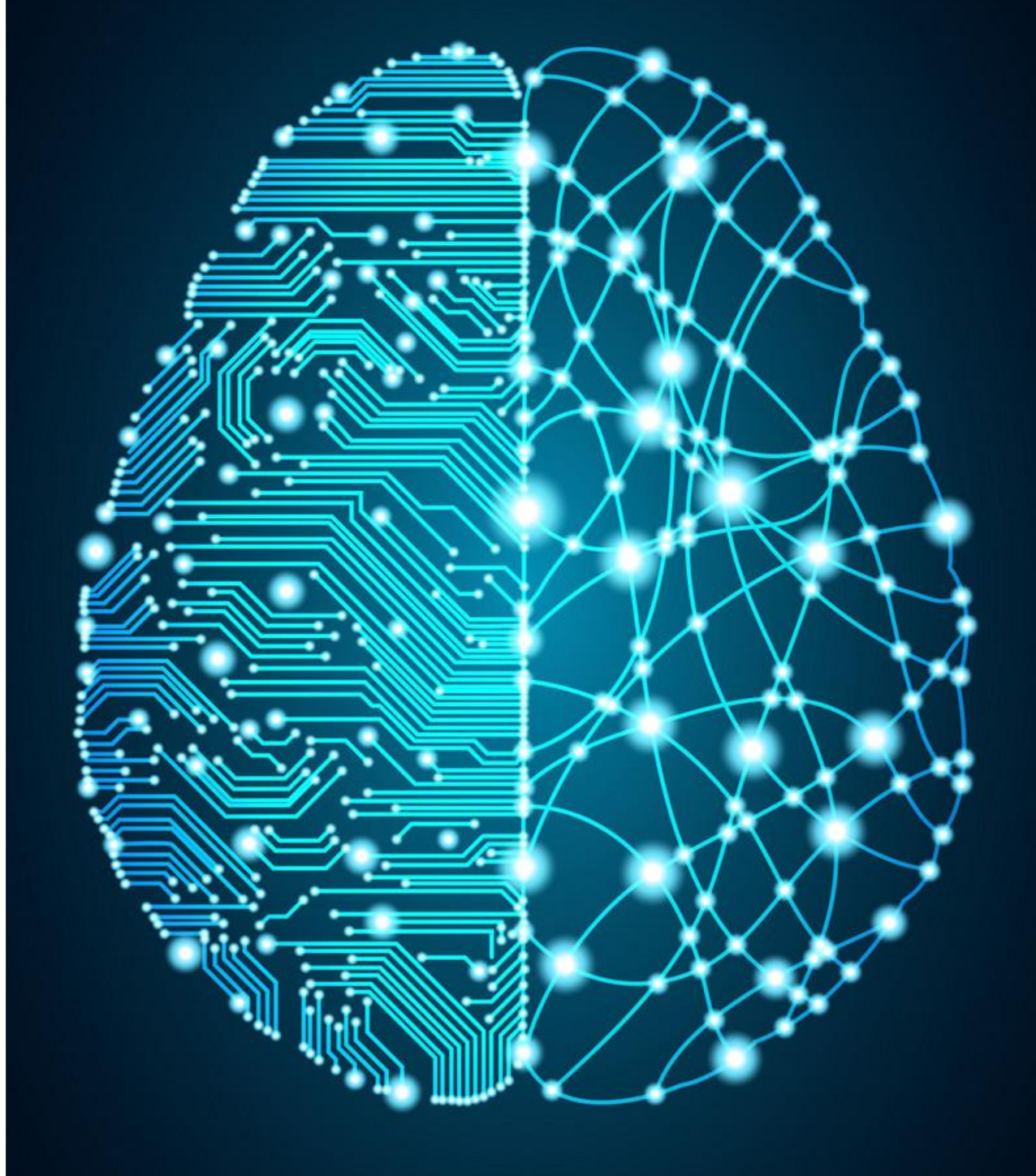
The very basics of human-robot interaction

- Who should I greet? Not all persons might be interested in a conversation.
- As humans we do not have to think about this.
- The robot should be able to start the interaction.
- It needs intelligence to do so!
- Social AI comes into play.

How smart am I?

The requirements for greeting

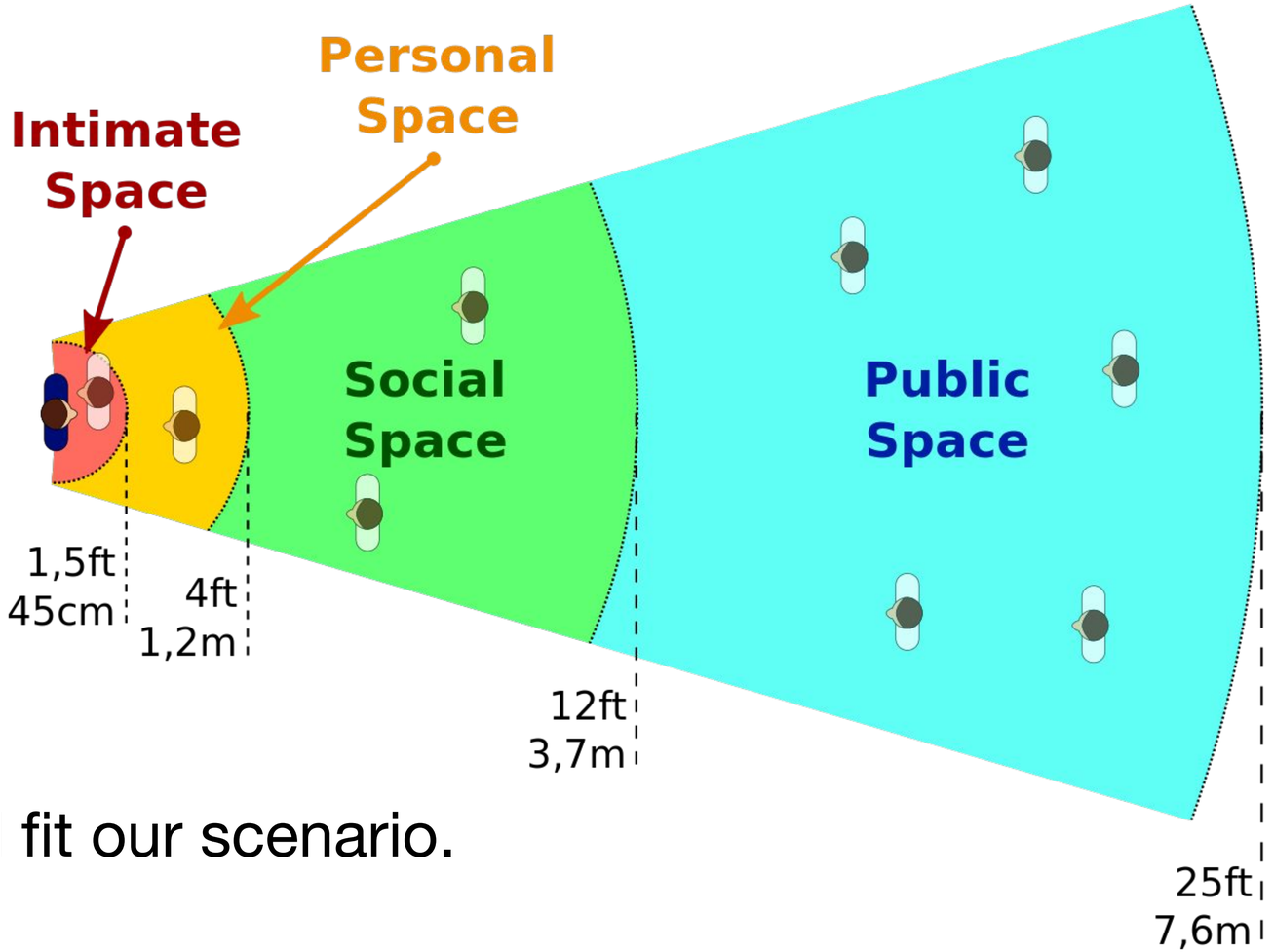
- Detect the presence of a person.
  - How many persons are there?
- Locate and identify the person(s).
- Decide to greet a person or not
  - Which person do I greet?
  - When?
- Do the actual greeting action within an appropriate timespan.



Shall I greet this person?

Theoretical foundation from social psychology

- Hall's Proxemic Theory[1]
- *Intimate relationship; e.g. lovers*
- *Good friends, family*
- *Impersonal business; acquaintances*
- *Public speaking*
- Social space / personal space could fit our scenario.
- Additional requirement: distance to a person.




Person 1
at 2.34m
Person 0
at 2.35m

How to perceive depth from 2D images?

A human-inspired approach

- We as humans need the signal from both eyes to perceive depth —> use stereo images instead of mono images with robots!

- Objects closer to camera ‘move’ more between two images.

- Basic technique, cost-efficient and fast.

How to perceive depth from 2D images?

- **f**: Focal length
- **B**: Distance between camera's
- **X**: Observed point
- **x-x'**: Disparity
- **z**: Depth

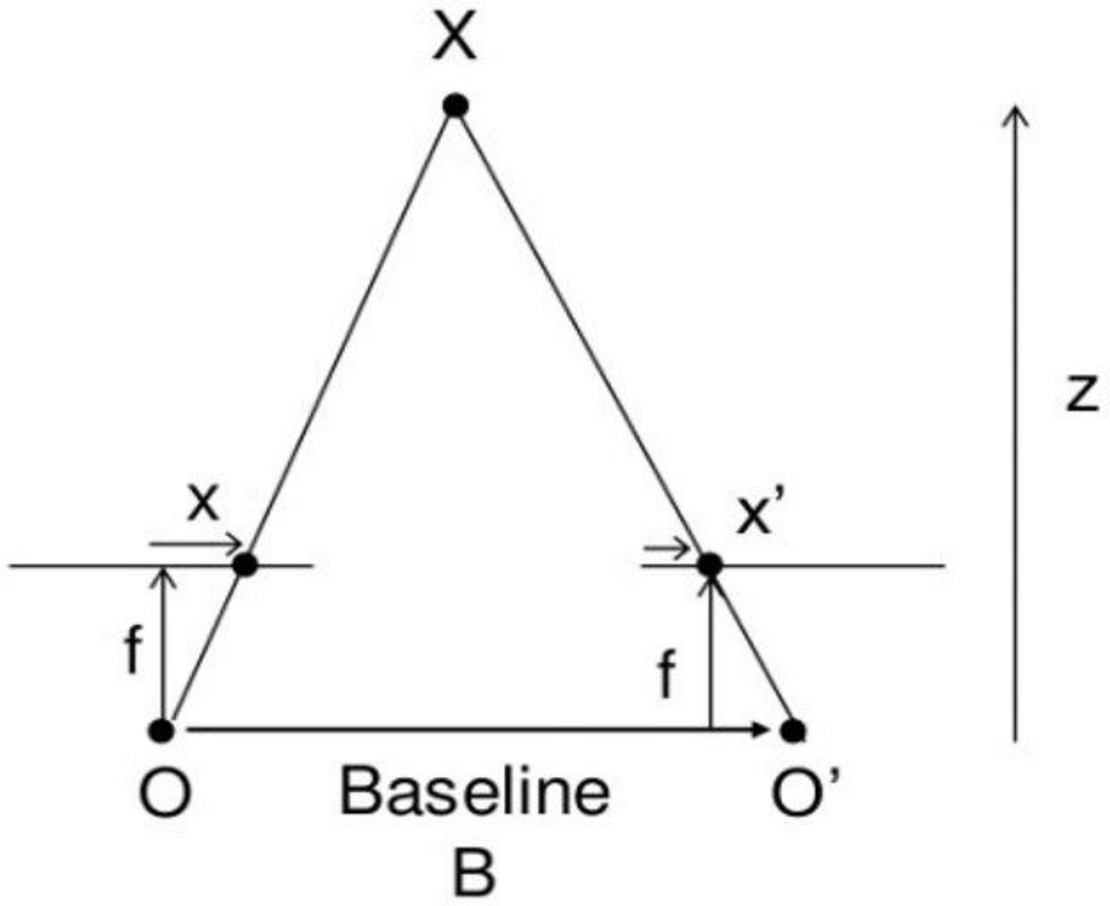


Image source: opencv.org

How to perceive depth from 2D images?

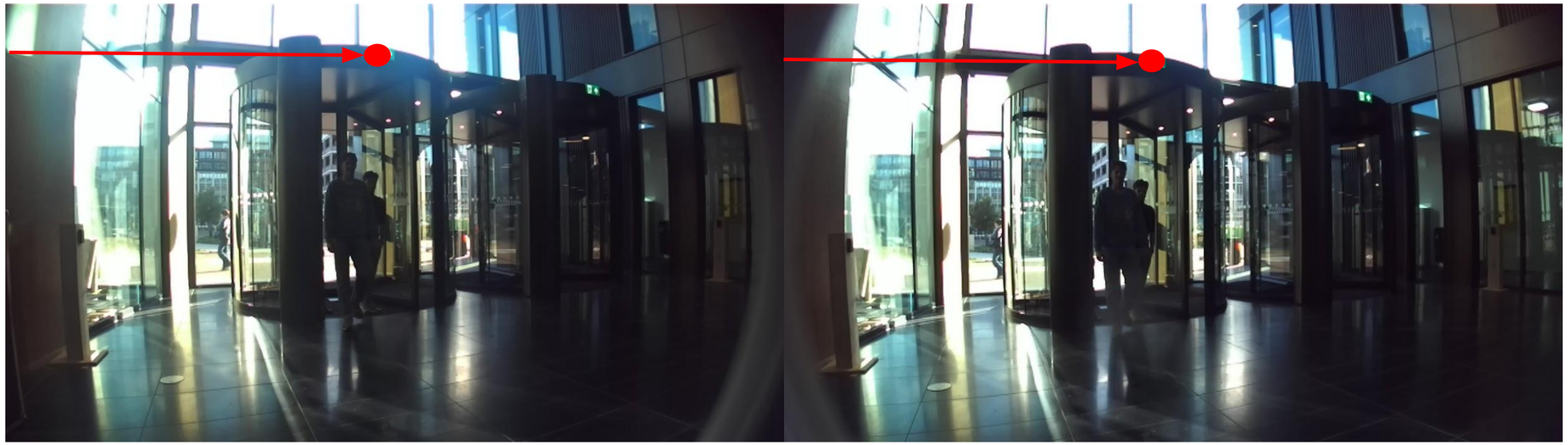
A human-inspired approach

How to perceive depth from 2D images?

A human-inspired approach

How to perceive depth from 2D images?

A human-inspired approach



How to perceive depth from 2D images?

- **f**: Focal length
- **B**: Distance between camera's
- **X**: Observed point
- **x-x'**: Disparity
- **z**: Depth

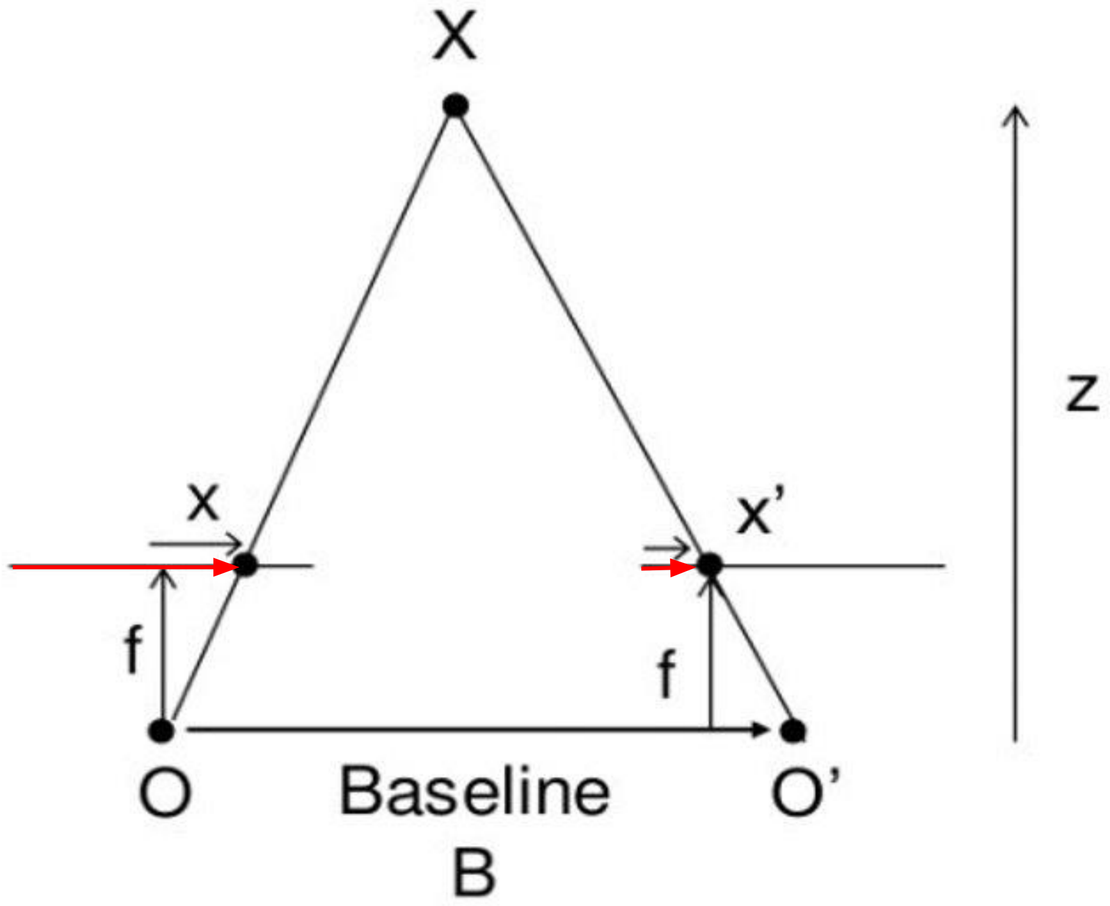


Image source: opencv.org

How to perceive depth from 2D images?

- **f**: Focal length
- **B**: Distance between camera's
- **X**: Observed point
- **x-x'**: Disparity
- **z**: Depth

$$z = \frac{f \cdot B}{x - x'} = \frac{f \cdot B}{d}$$

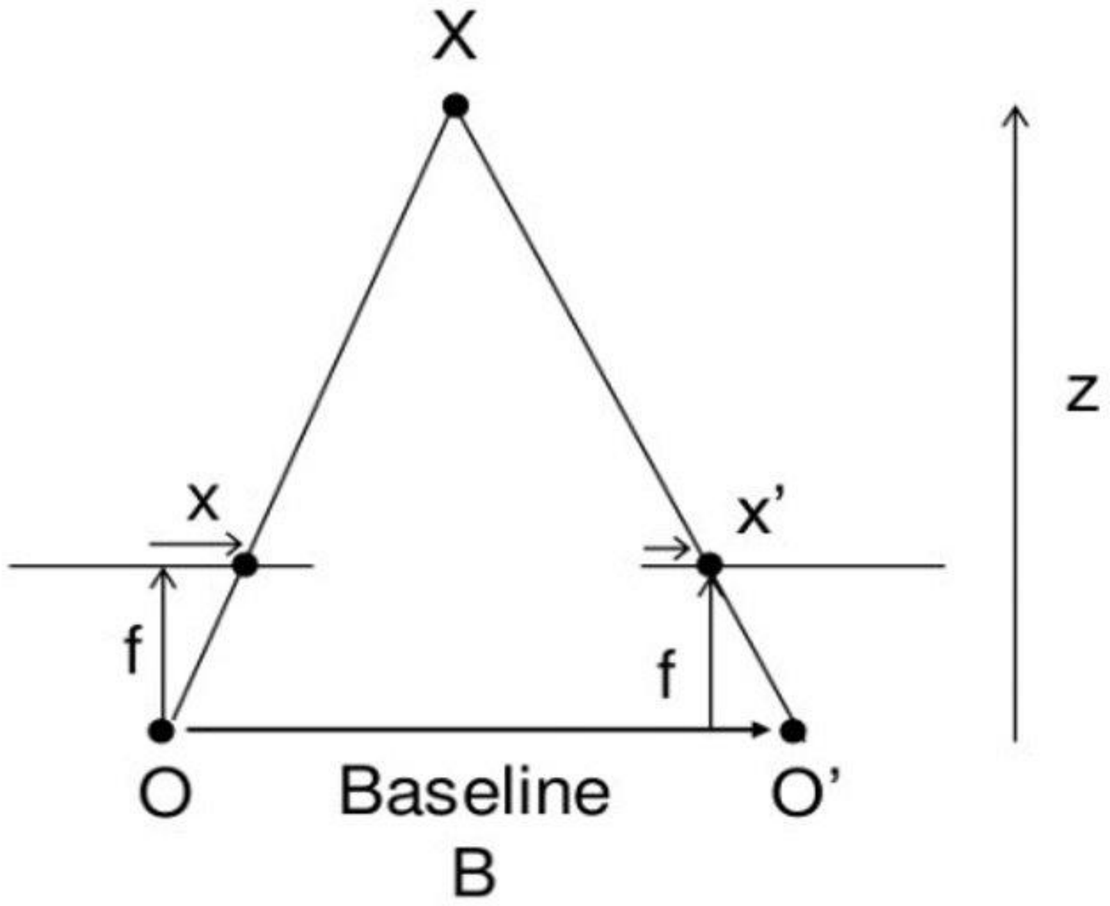
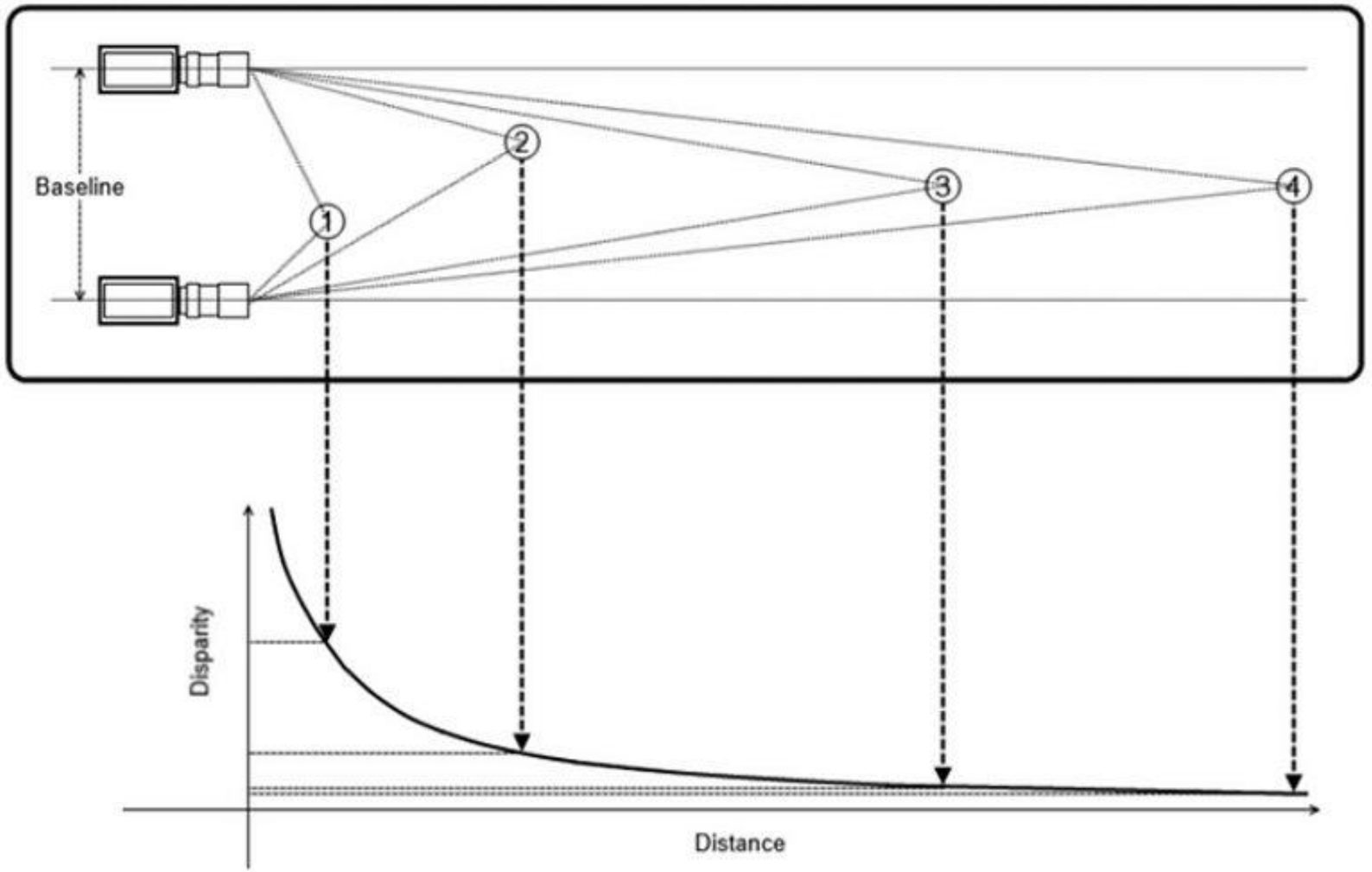


Image source: opencv.org

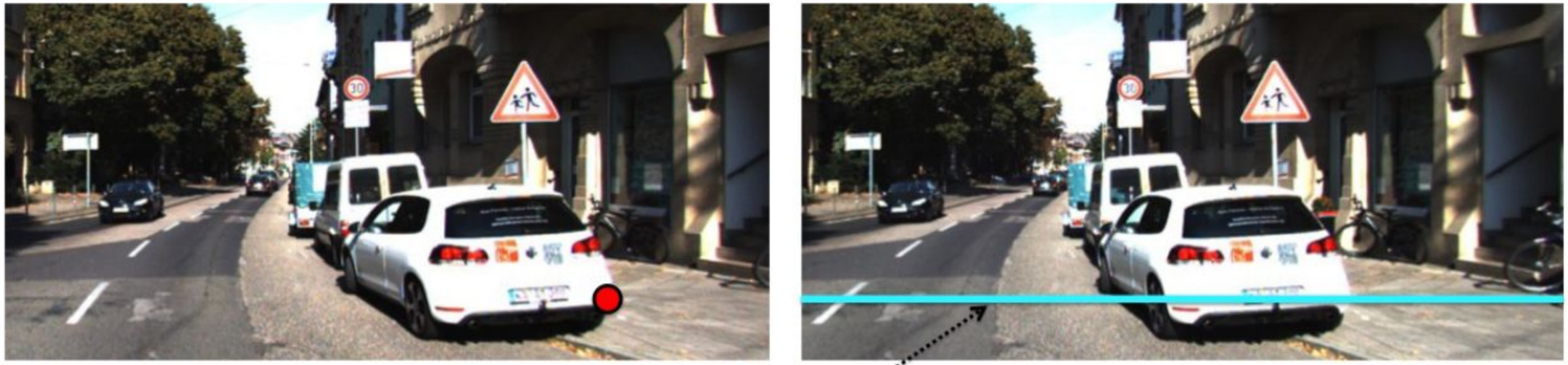
How to perceive depth from 2D images?



1
2

Image source: https://www.oreilly.com/library/view/learning-opencv-3/9781491937983/

How to perceive depth from 2D images?



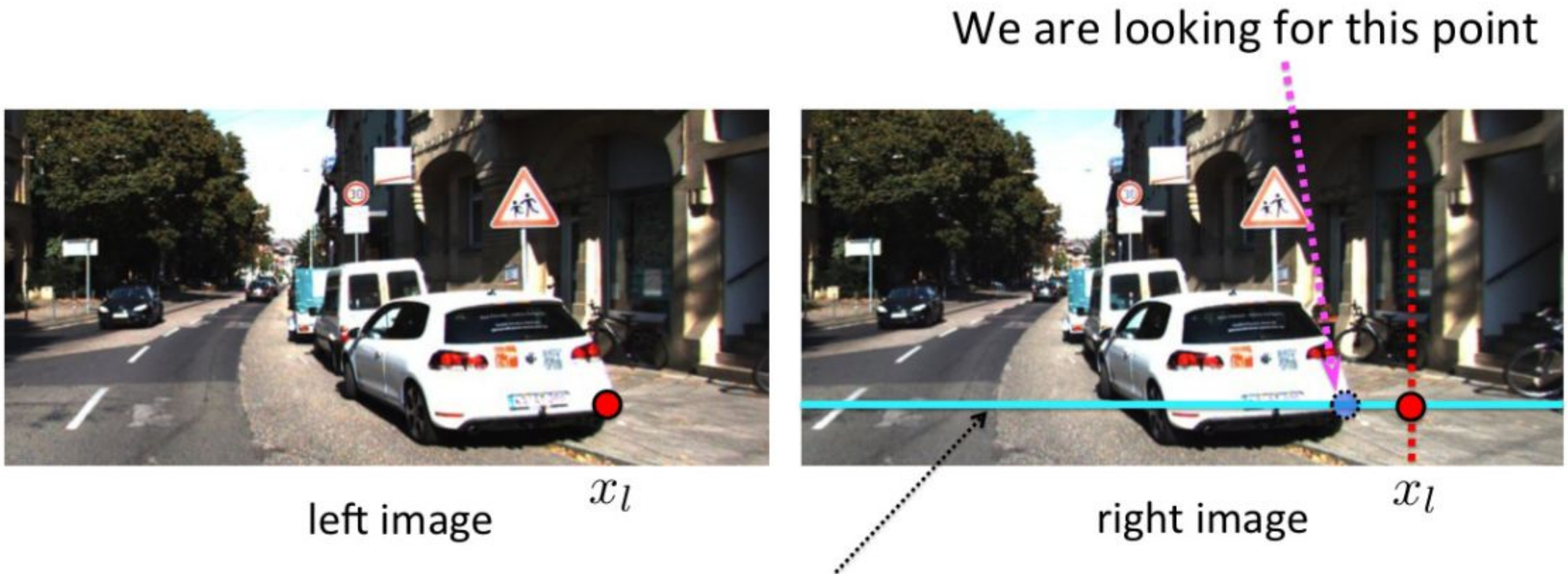
left image

right image

the match will be on this line (same y)

source: http://www.cs.toronto.edu/~fidler/slides/2015/CSC420/lecture12_hres.pdf

How to perceive depth from 2D images?



source: http://www.cs.toronto.edu/~fidler/slides/2015/CSC420/lecture12_hres.pdf

Pipeline for stereo depth

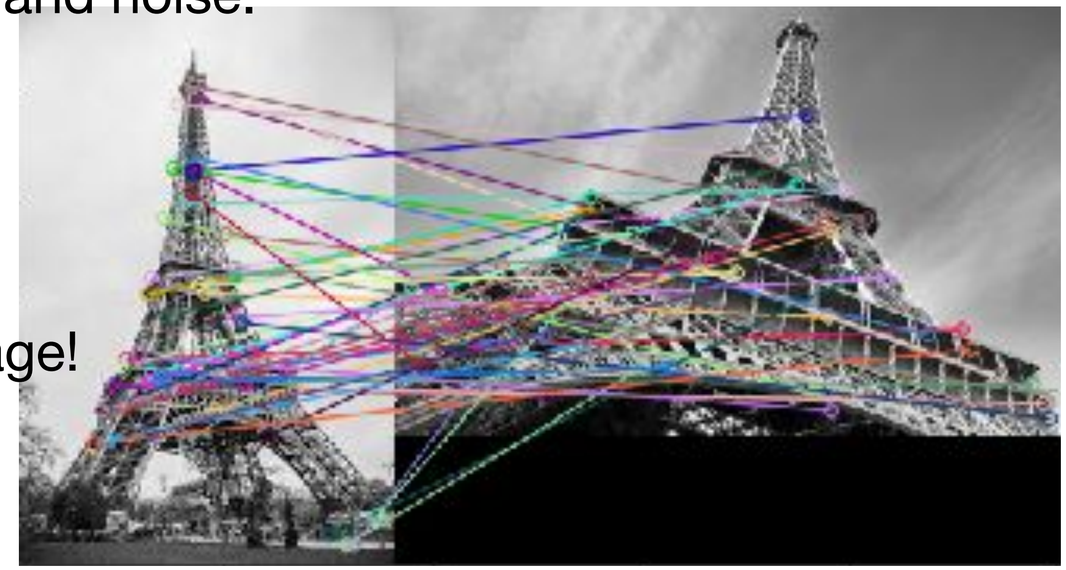
The devil is in the details

- Calibrate the camera (checkerboard)
- Take stereo image
- Split image into two separate images
- Align both images with calibration matrices
- Match every pixel in left image with a pixel in right image (occlusion is a problem)
- Create disparity map to obtain per-pixel depth information:
- Calibration and alignment are great challenges. Use more points for calibration!

$SIFT^2$ for calibration

Distinctive Image Features from Scale-Invariant Keypoints

- Checkerboard ± 70 points, but not uniformly spread across the image!
- Keypoint descriptor based on local image gradients.
- Very robust descriptor due to invariance to rotation and scale.
  - Also robust to changes in lighting, affine distortion, and noise.
- Match keypoints from both images and find the transformation matrix.
- Can use hundreds of keypoints, across the whole image!

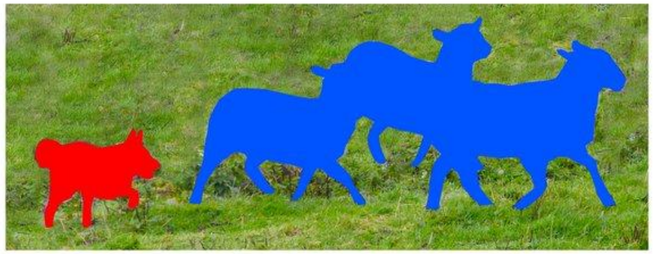


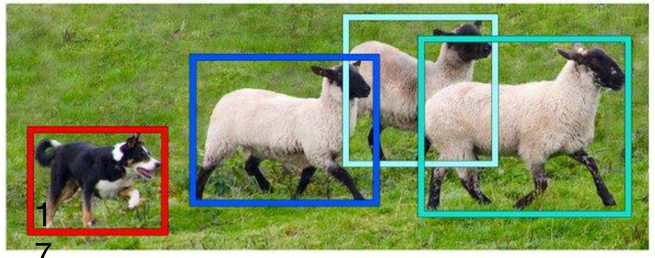
Depth is cool, but what is a person?

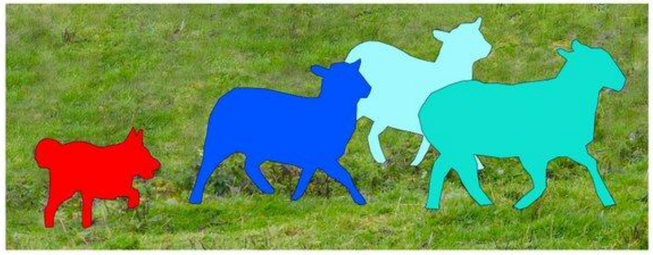
Instance segmentation in a nutshell

- From image recognition to instance segmentation.
- We need to be able to distinguish between people.
- Object instance detection not sufficient; why not?
- Use the instance mask for depth.
- But, what if the mask is not perfect?

Image Recognition

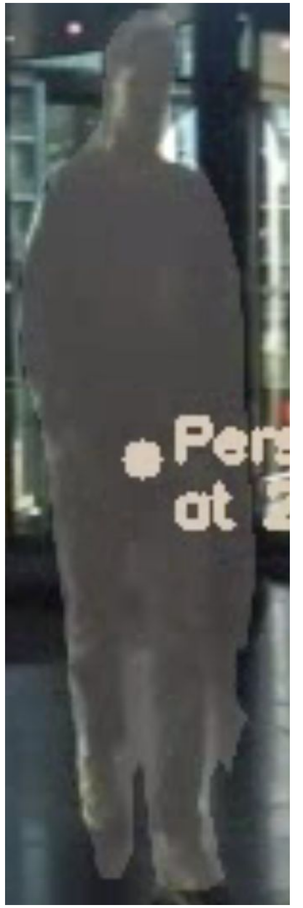

Semantic Segmentation


Object Detection


Instance Segmentation

What metric to use?

A: Mean
B: Median
C: Maximum



What metric to use?

A: Mean

B: Median

C: Maximum

How to do instance segmentation?

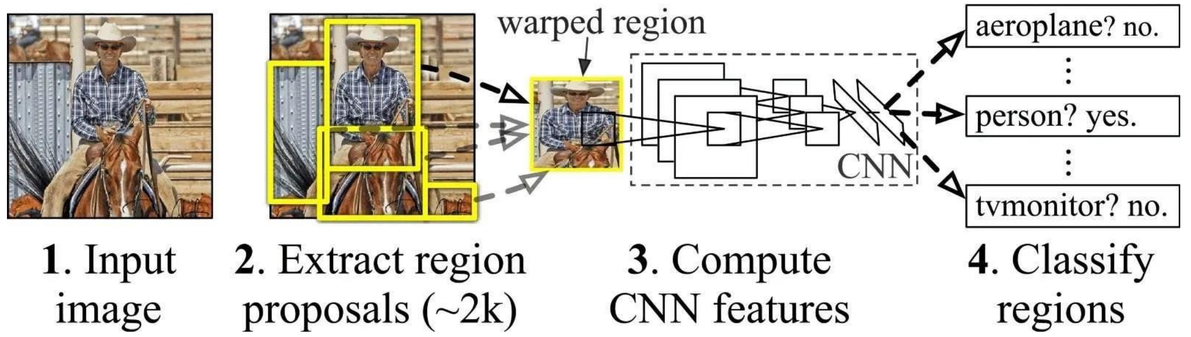
Lots of research done in CV field

- Detectron2[3] library from Facebook Research.
- Mask-RCNN[4] structure with ResNet50[5] backbone.
- Many pretrained models are available.

A little dive into Mask-RCNN

It is all based on Convolutional Neural Networks

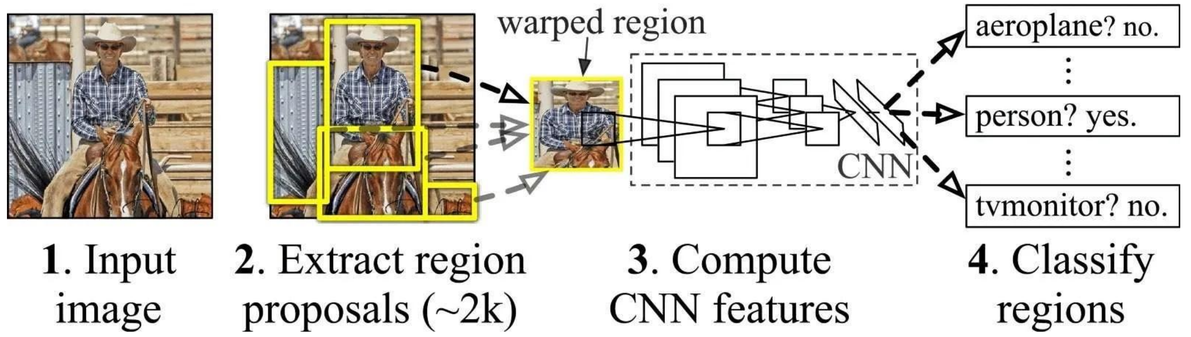
- R-CNN[6] foundation of Mask-RCNN.
- Many forward passes make it slow.



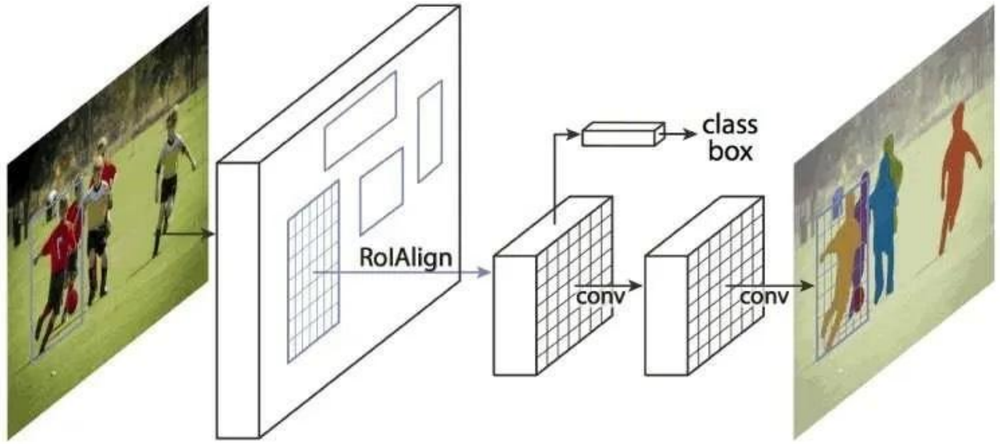
A little dive into Mask-RCNN

It is all based on Convolutional Neural Networks

- R-CNN[6] foundation of Mask-RCNN.
- Many forward passes make it slow.



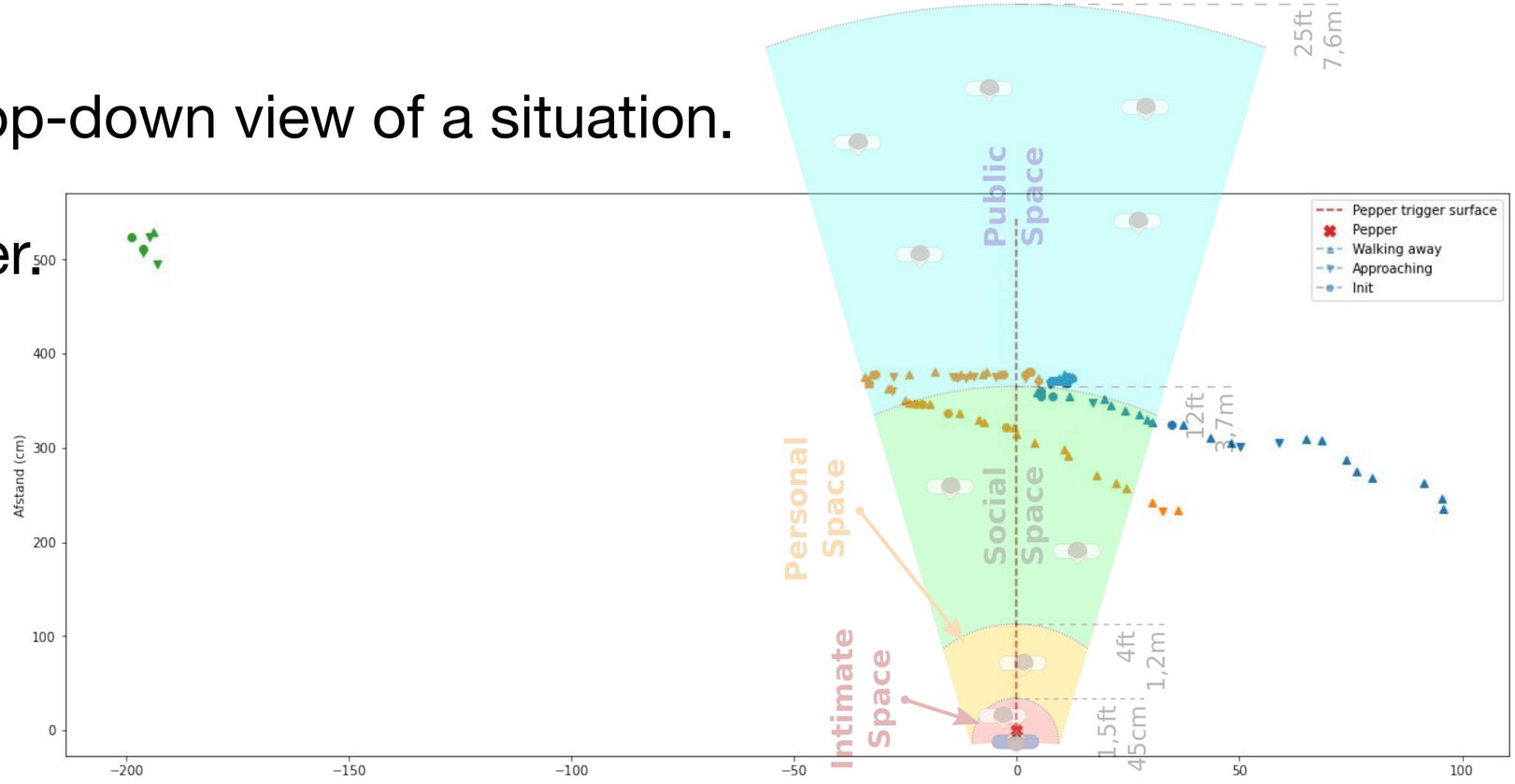
- Re-use features from conv layers + RPN.
- Additional branch for mask prediction.



Combining it all

Depth, instance segmentation, Hall's Proxemic…

- We can now generate a top-down view of a situation.
- Distance relative to Pepper.
- Distinguish persons.
- Approaching or not.
- Hall's spaces.
- Path prediction.
- Shall I say hello?



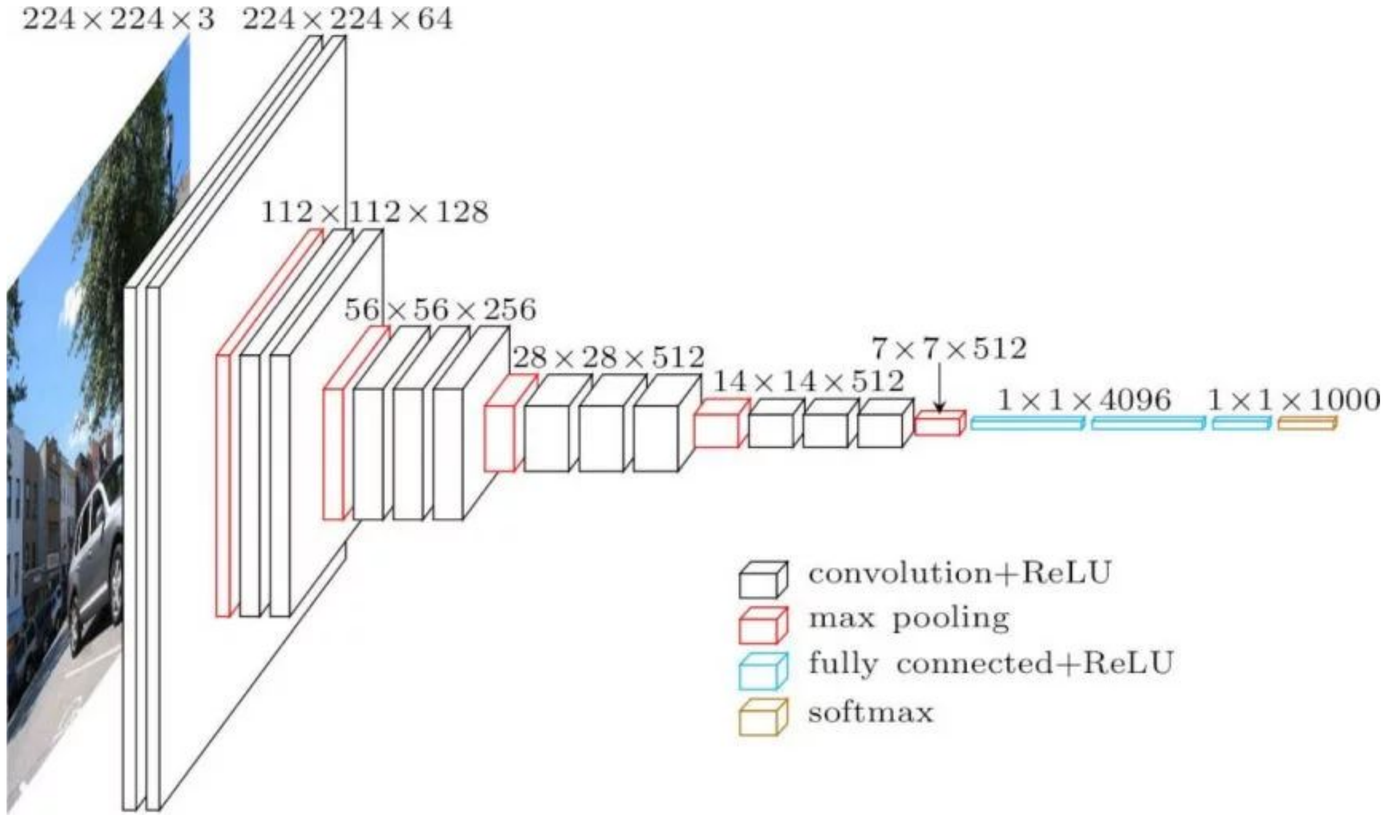
Remaining challenges

We have still some work to do

- How reliable is the pipeline?
- How can we make the whole pipeline fast enough to run in real-time?
- How can we make the whole pipeline fast enough to run in real-time?
- What happens when more people (e.g. 5+) enter the frame?
- How can we track people better?

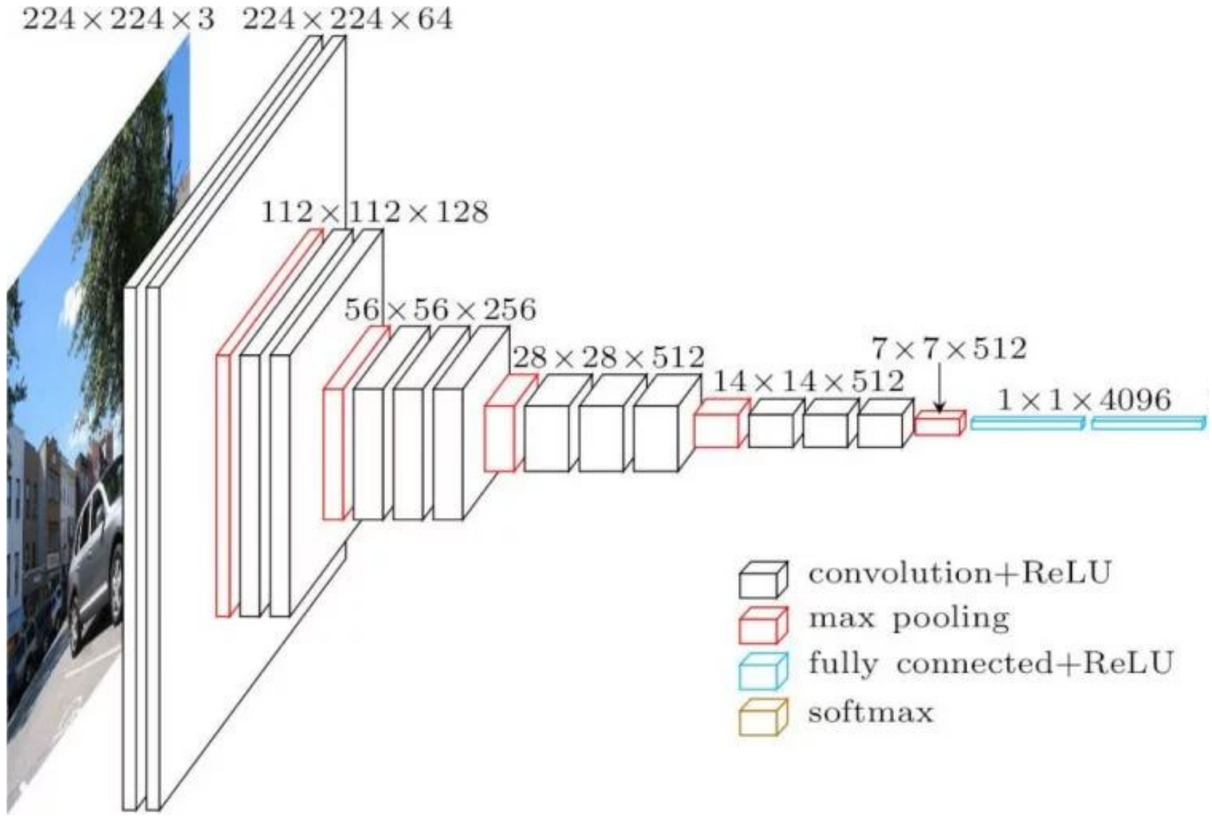
Next step: Face recognition

- Trained using large datasets of faces
  Learn to identify faces of the same person

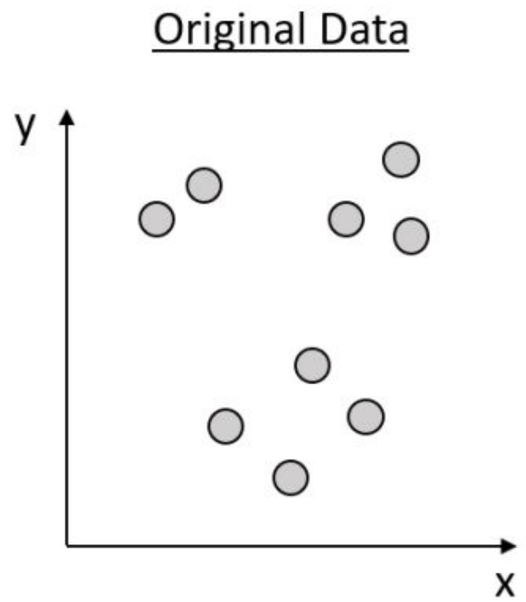


Next step: Face recognition

- Trained using large datasets of faces
  Learn to identify faces of the same person

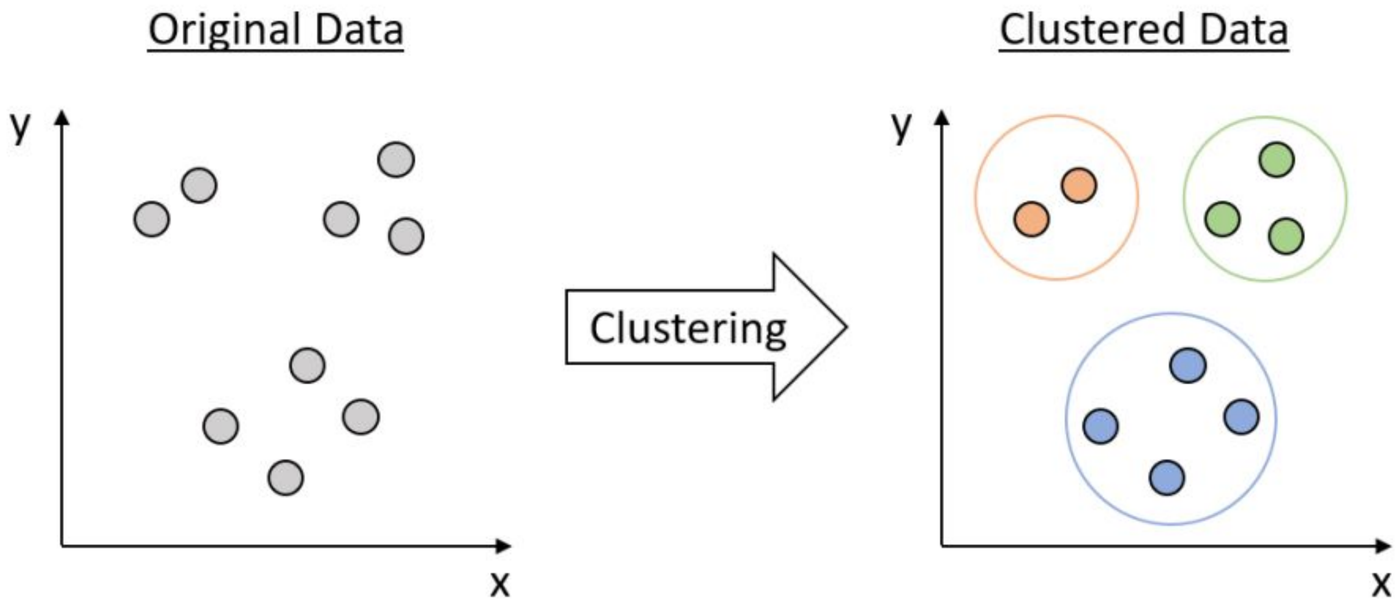


Next step: Face recognition



Next step: Face recognition

- Use clustering to recognize persons



The Social Interaction Cloud Framework

Provides components:

- Face detection
- Face recognition

```
# Connect to the services
desktop = Desktop()
face_rec = DNNFaceDetection()

# Feed the camera images into the face recognition component
face_rec.connect(desktop.camera)

# Send back the outputs to this program
desktop.camera.register_callback(on_image)
face_rec.register_callback(on_faces)

while True:
    img = imgs_buffer.get()
    faces = faces_buffer.get()

    for face in faces:
        draw_bbox_on_image(face, img)

    cv2.imshow('', img)
    cv2.waitKey(1)
```

References

1. Hall, E. T., Birdwhistell, R. L., Bock, B., Bohannan, P., Diebold, A. R., Durbin, M., Edmonson, M. S., Fischer, J. L., Hymes, D., Kimball, S. T., La Barre, W., McClellan, J. E., Marshall, D. S., Milner, G. B., Sarles, H. B., Trager, G. L., & Vayda, A. P. (1968). Proxemics [and Comments and Replies]. Current Anthropology, 9(2/3), 83–108. https://doi.org/10.1086/200975

2. Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, 60(2), 91–110. https://doi.org/10.1023/b:visi.0000029664.99615.94

3. Wu, Y., Kirillov, A., Massa, F., Lo, W. Y., & Girshick, R. (2019). Detectron2. GitHub. Retrieved November 30, 2021, from https://github.com/facebookresearch/detectron2

4. He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. 2017 IEEE International Conference on Computer Vision (ICCV). Published. https://doi.org/10.1109/iccv.2017.322

5. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Published. https://doi.org/10.1109/cvpr.2016.90R-CNN

6. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2016). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(1), 142–158. https://doi.org/10.1109/tpami.2015.2437384