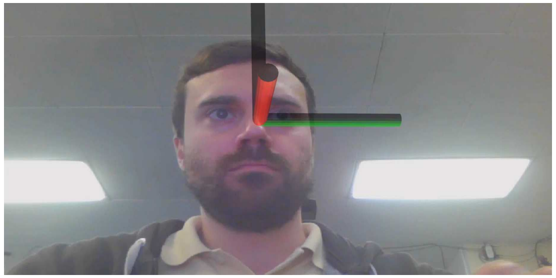


Introduction: Sensing Visual Data in ROS

Human-Centred Robotics

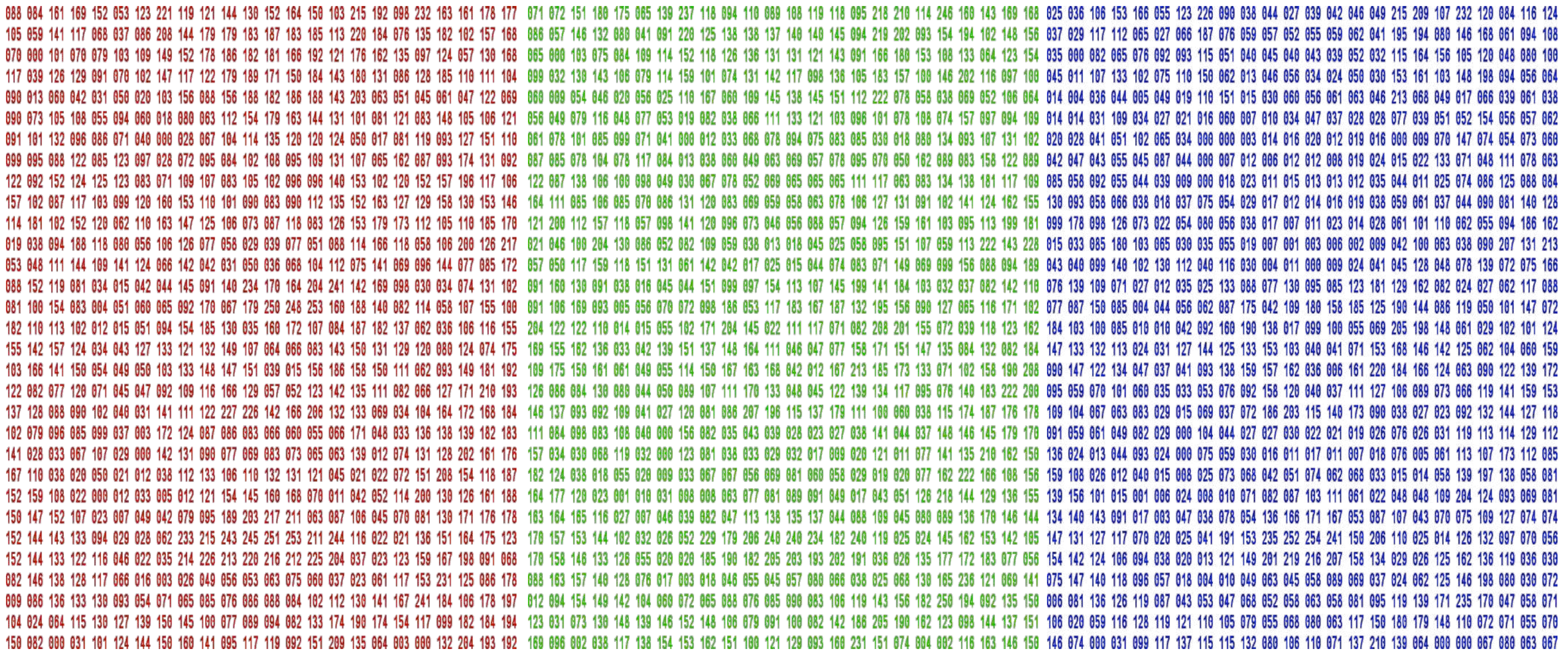
Dr Tobias Fischer

2019-11-07

Computer Vision
The very basics

Computer Vision: The very basics

First encounter

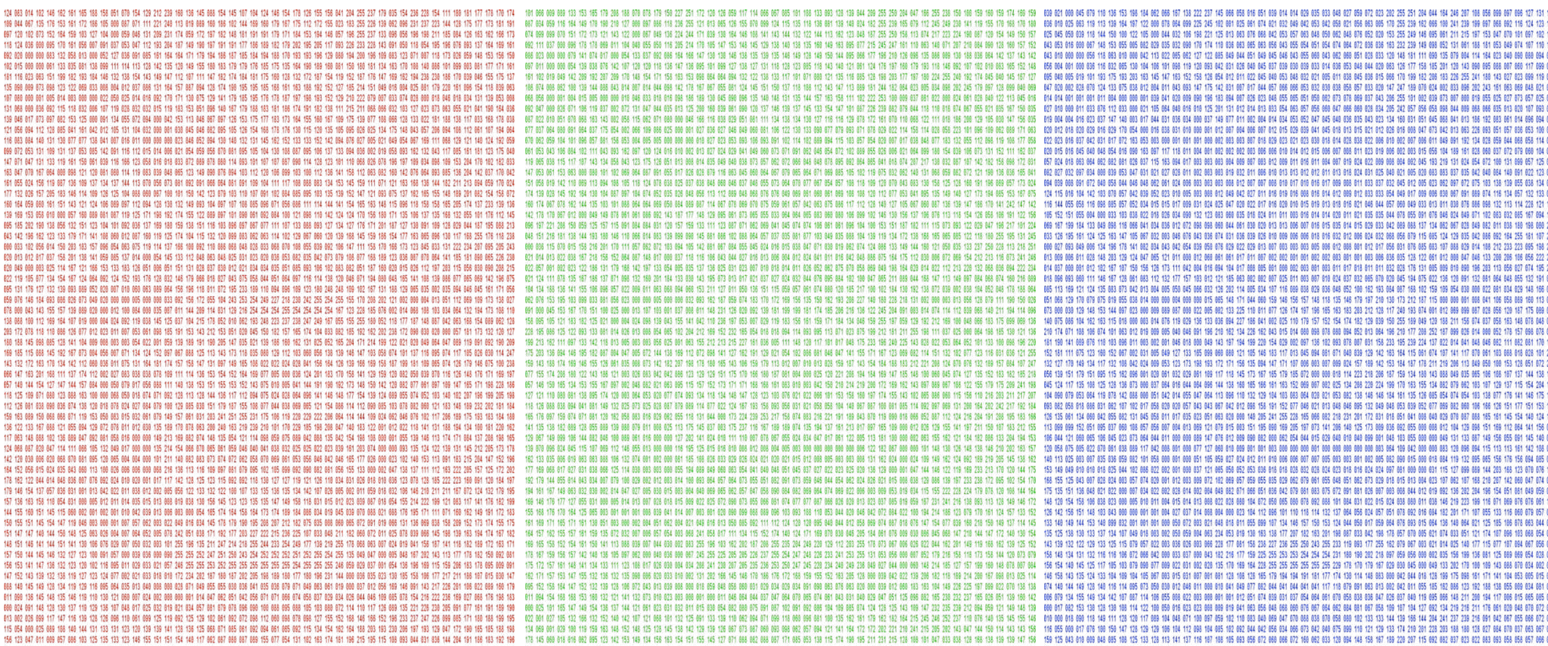


What do you see ?

Slide: Vittorio Ferrari

Computer Vision: The very basics

First encounter



And now ?

Slide: Vittorio Ferrari

Computer Vision: The very basics

First encounter

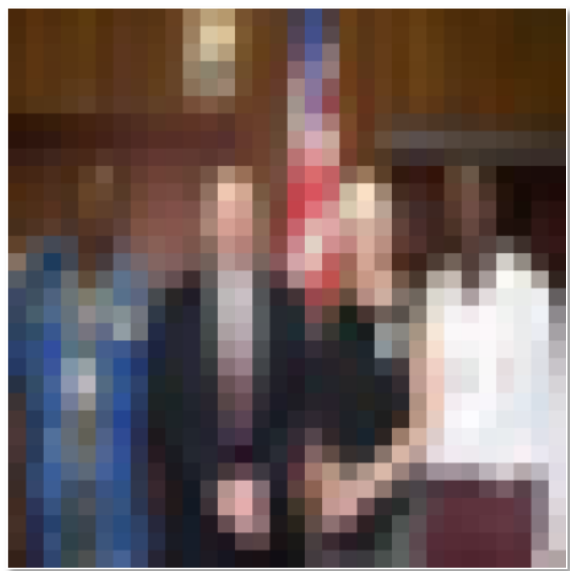
Rapid visual recognition is important !

Slide: Vittorio Ferrari

Computer Vision: The very basics

The goal of computer vision

To extract "meaning" from pixels



Humans are remarkably good at this…

Source: "80 million tiny images" by Torralba et al.

Computer Vision: The very basics

Vision as a source of semantic information

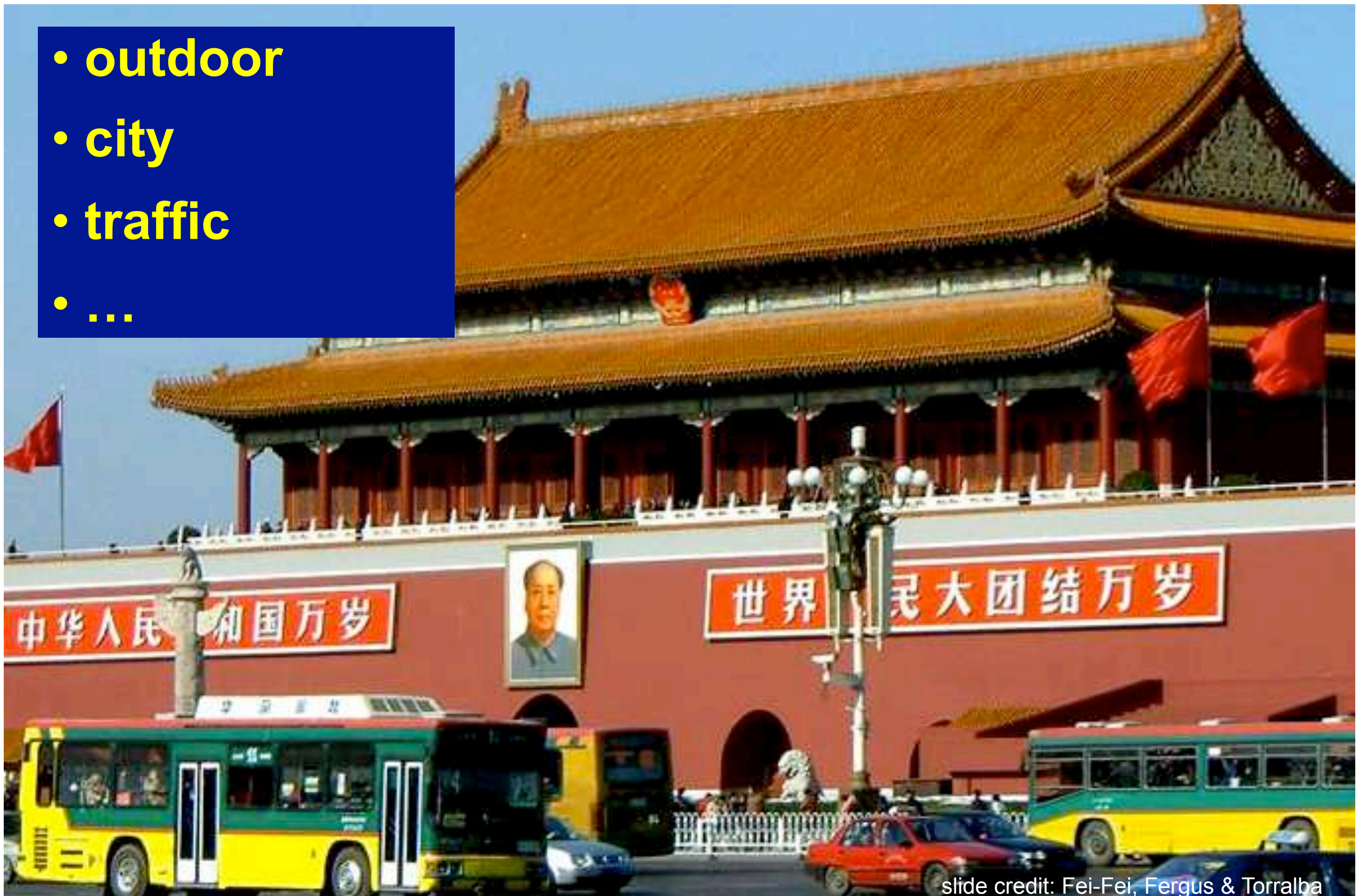
slide credit: Fei-Fei, Fergus & Torralba

Computer Vision: The very basics

Object categorization

Computer Vision: The very basics

Scene and context categorization



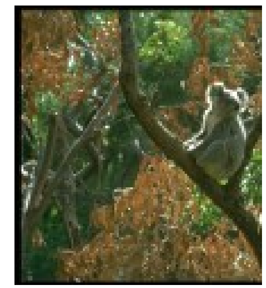
Why is Computer Vision hard?

Why is Computer Vision hard?

Challenges: Robustness

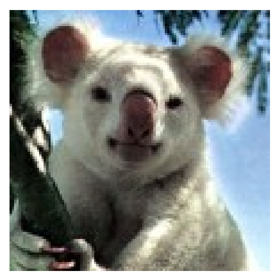
Illumination

Object pose



Clutter

Occlusions



Intra-class variation

Viewpoint

Slide credit: Kristen Grauman

Why is Computer Vision hard?

Challenges: Robustness

- **Detection in Crowded Scenes**
  - **Learn object variability**
    - **Changes in appearance, scale, and articulation**
  - **Compensate for clutter, overlap, and occlusion**

[Leibe, Seemann, Schiele, CVPR'05]

Why is Computer Vision hard?

Challenges: object intra-class variation

slide credit: Fei-Fei, Fergus & Torralba

Why is Computer Vision hard?

Inherent ambiguity of the problem

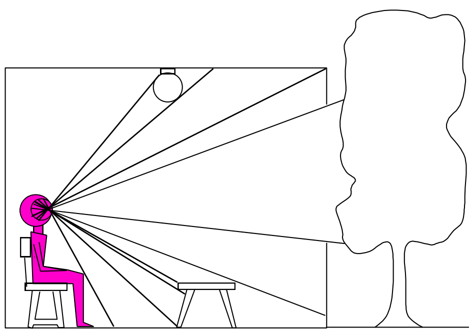
- Many different 3D scenes could have given rise to a particular 2D image

Slide: Vittorio Ferrari

Why is Computer Vision hard?

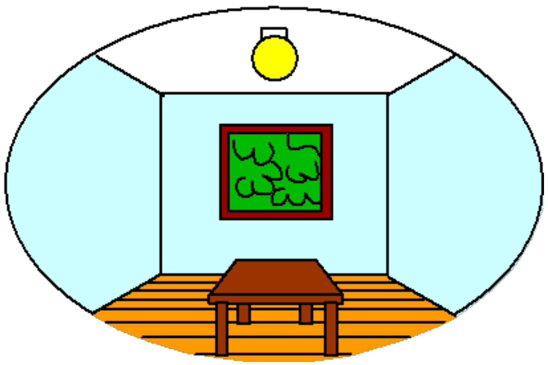
Dimensionality reduction: from 3D to 2D

3D world



Point of observation

2D image



What is preserved?

- Straight lines, incidence

What have we lost?

- Angles, lengths

Slide by A. Efros

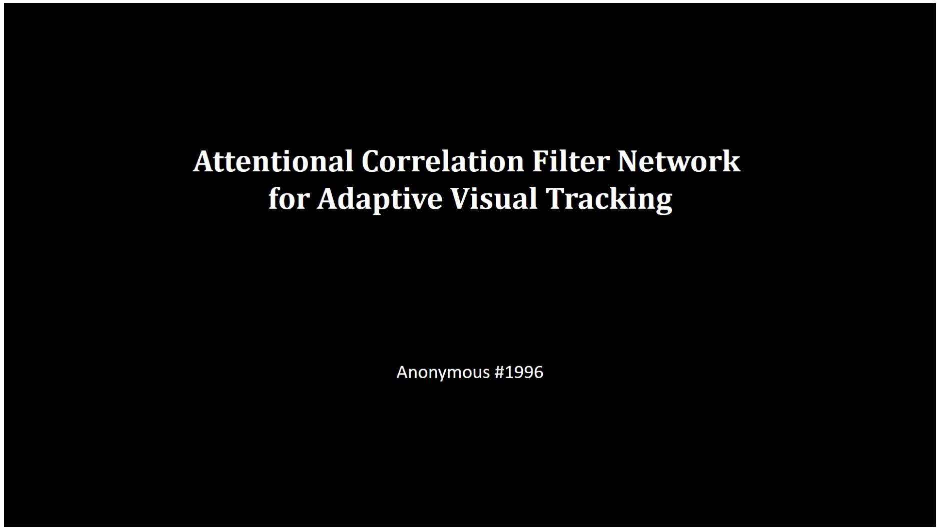
Figures © Stephen E. Palmer, 2002

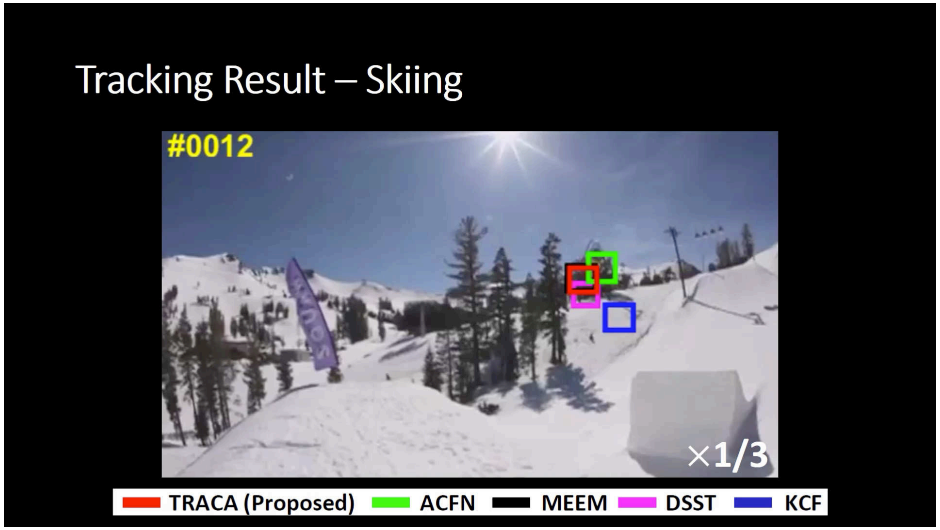
Computer Vision in the Personal Robotics Lab

All code available!

Computer Vision in the Personal Robotics Lab

- Ultra-fast visual object tracking

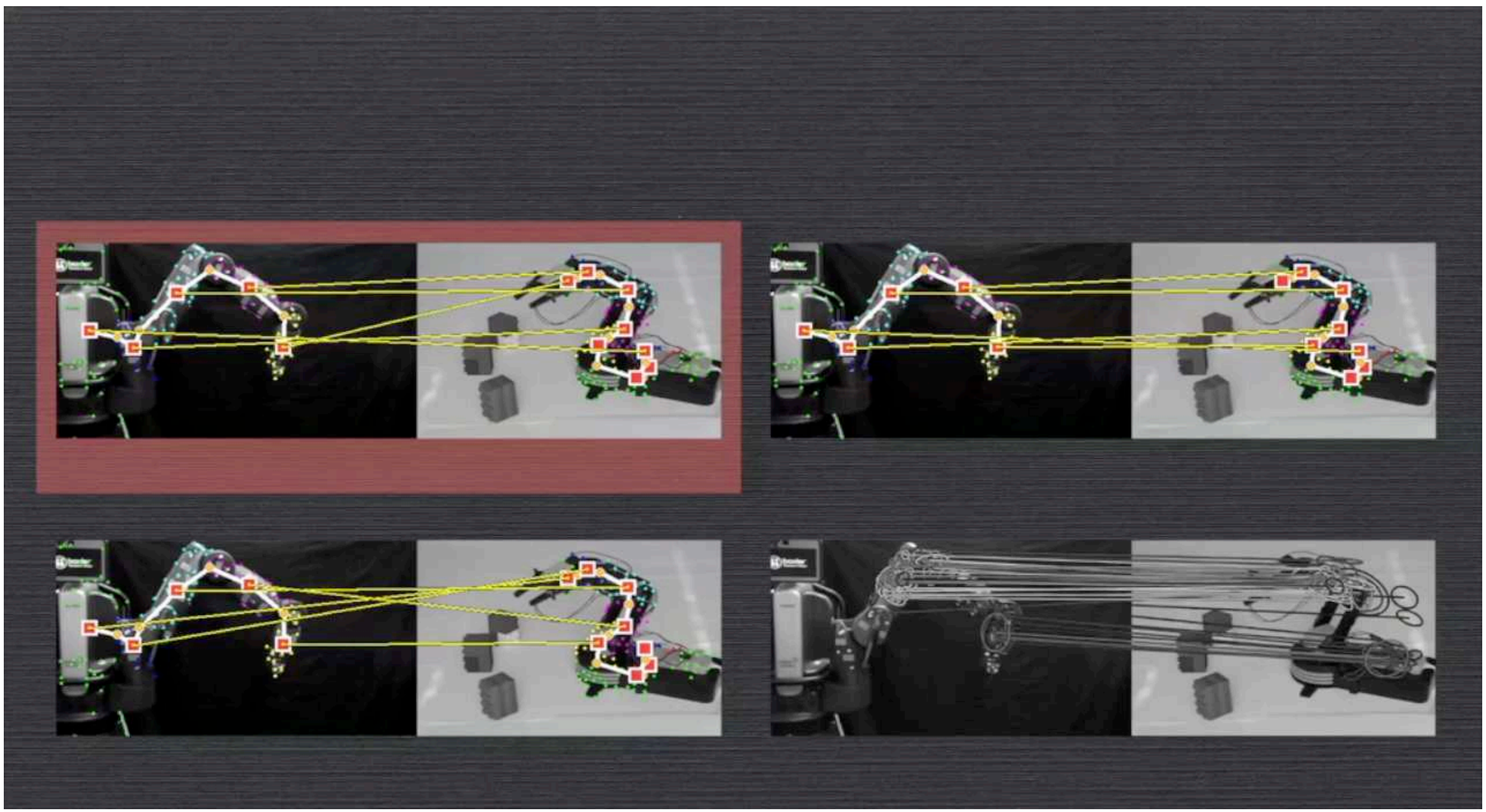




Choi et al. CVPR2017 & CVPR2018

Computer Vision in the Personal Robotics Lab

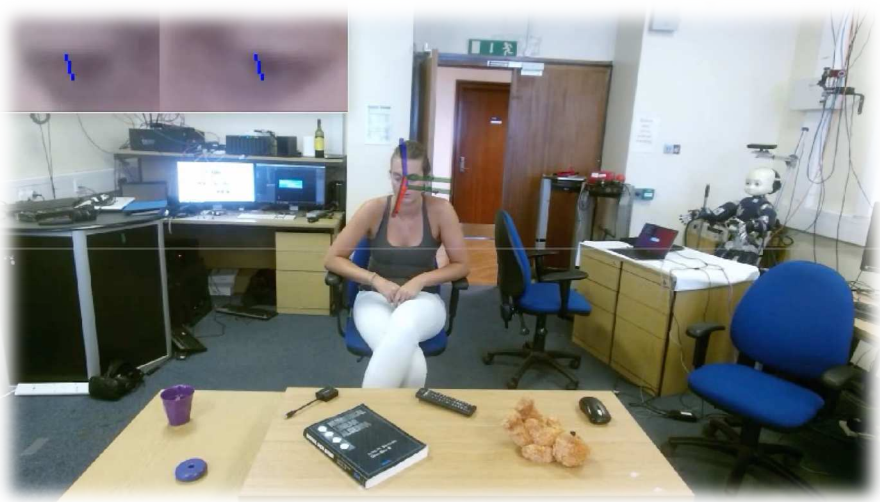
- Robot motion retargeting

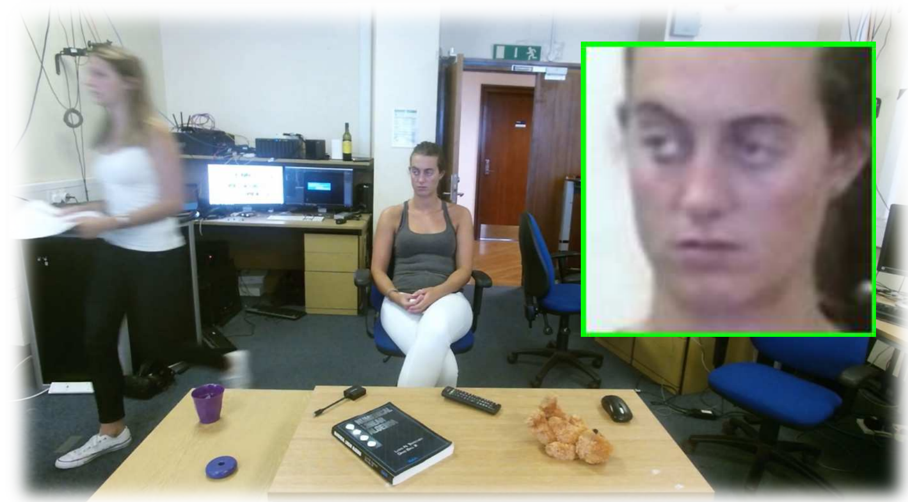


Chang et al. CVPR2016

Computer Vision in the Personal Robotics Lab

- Gaze & blink estimation





Fischer et al. ECCV2018 &
Cortacero et al. ICCV2019W

Other popular computer vision resources

All code available!

Keep in mind you need powerful GPU

Computer Vision with Tensorflow & PyTorch

- Human pose estimation using Detectron2

Detectron2

Computer Vision with Tensorflow & PyTorch

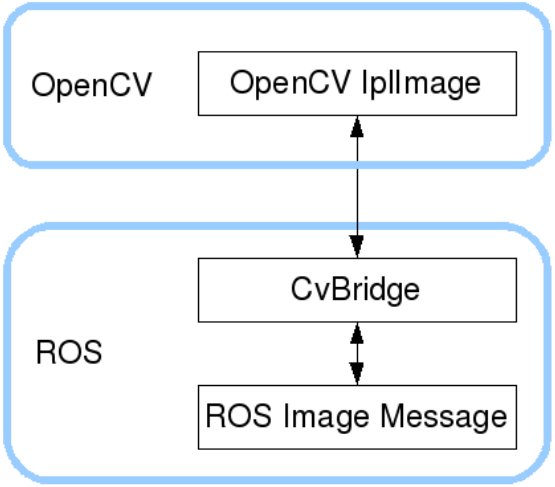
- Object classification & segmentation using Detectron2

Codecentric.AI

Computer Vision & ROS

Computer Vision & ROS

- My recommendation: Python for ease of use
- C++ & Python nodes can interact!
- Most common underlying data structure: Numpy
- Easy conversion between ROS sensor_msgs/Image & OpenCV using cv_bridge



Acquiring images

- There are different packages to acquire images from a camera

- Webcam: uvc_camera or cv_camera

- Kinect / Asus Xtion: openni_launch <- Provides depth

- If one does not work, try another one ..
  wiki.ros.org/Sensors/Cameras

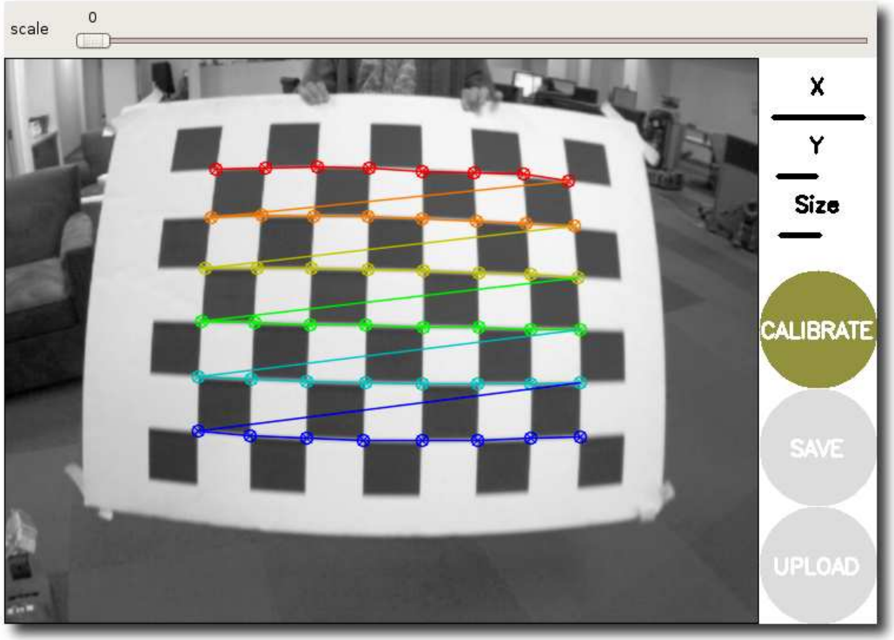
Acquiring images

- Unifying data format: sensor_msgs/Image

```
# This message contains an uncompressed image
# (0, 0) is at top-left corner of image
#

Header header          # Header timestamp should be acquisition time of image
                       # Header frame_id should be optical frame of camera
                       # origin of frame should be optical center of camera
                       # +x should point to the right in the image
                       # +y should point down in the image
                       # +z should point into to plane of the image
                       # If the frame_id here and the frame_id of the CameraInfo
                       # message associated with the image conflict
                       # the behavior is undefined

uint32 height           # image height, that is, number of rows
uint32 width            # image width, that is, number of columns

# The legal values for encoding are in file src/image_encodings.cpp
# If you want to standardize a new string format, join
# ros-users@lists.sourceforge.net and send an email proposing a new encoding.

string encoding         # Encoding of pixels -- channel meaning, ordering, size
                        # taken from the list of strings in include/sensor_msgs/image_encodings.h

uint8 is_bigendian      # is this data bigendian?
uint32 step             # Full row length in bytes
uint8[] data            # actual matrix data, size is (step * rows)
```

Acquiring images

- It is often beneficial to make sure your camera is calibrated: http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration



- We have a checkerboard pattern in the lab

Pointers to other materials

- https://github.com/osrf/rosbook
  “Programming Robots with ROS” Chapter 12: Follow-Bot
- https://industrial-training-master.readthedocs.io/en/melodic/_source/session5/OpenCV-in-Python.html
- ROS Wiki
- OpenCV
- PCL

Practical session

Practical session

1. Subscribing & publishing images, interfacing with OpenCV

2. Face detection with OpenCV

3. Obtaining 3D coordinates using Kinect-like sensor
   Make sure you calibrate color AND depth cameras:
   http://wiki.ros.org/openni_launch/Tutorials/IntrinsicCalibration
   http://wiki.ros.org/openni_launch/Tutorials/ExtrinsicCalibration

4. TensorFlow specifics in ROS

5. PyTorch example