Image-Based Situation Awareness

Sakari Lampola

Sakari Lampola

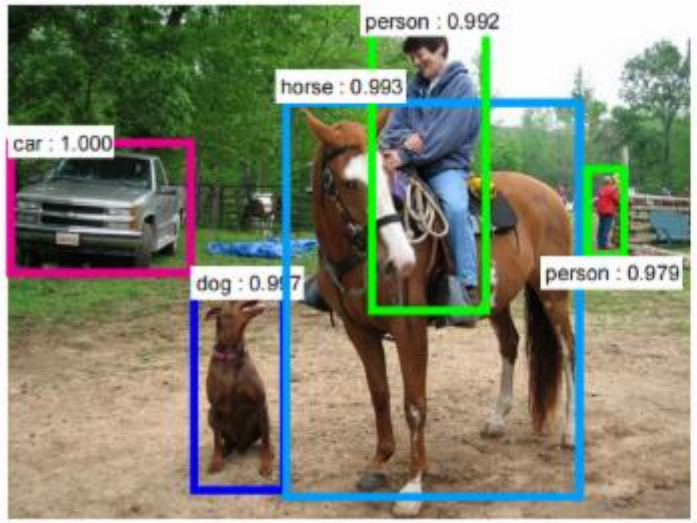
- Master of Science 1983 (TTKK, system theory)
- TTKK 1980-1983 (assistant, researcher)
- Tampereen Sähkölaitos 1980-1982 (student trainee)
- Imatran Voima Oy 1984-1988 (group leader)
- Kymenlaakso University of Applied Sciences 1988-1991 (teacher)
- Oy Cybersoft Ab 1989-2012

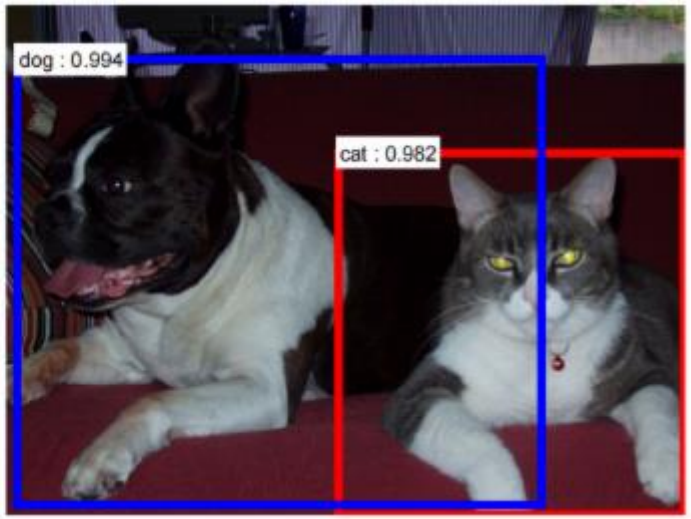
Motivation: Understanding Image

Understanding leads to action

Understanding enables prediction

Object detection helps understanding





Beyond state of the art

- External knowledge
- Expectations
- Probabilities
- Emotions
- …

Understanding Image: Image Captioning

Image captioning examples

man in black shirt is playing guitar.

construction worker in orange safety vest is working on road.

two young girls are playing with lego toy.

boy is doing backflip on wakeboard.

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Image captioning models

Deep Visual-Semantic Alignments for Generating Image Descriptions

Andrej Karpathy Li Fei-Fei
Department of Computer Science, Stanford University
{karpathy,feifeili}@cs.stanford.edu

Abstract

We present a model that generates natural language descriptions of images and their regions. Our approach leverages datasets of images and their sentence descriptions to learn about the inter-modal correspondences between language and visual data. Our alignment model is based on a novel combination of Convolutional Neural Networks over image regions, bidirectional Recurrent Neural Networks over sentences, and a structured objective that aligns the two modalities through a multimodal embedding. We then describe a Multimodal Recurrent Neural Network architecture that uses the inferred alignments to learn to generate novel descriptions of image regions. We demonstrate that our alignment model produces state of the art results in retrieval experiments on Flickr8K, Flickr30K and MSCOCO datasets. We then show that the generated descriptions significantly outperform retrieval baselines on both full images and on a new dataset of region-level annotations.

Figure 1. Motivation/Concept Figure: Our model treats language as a rich label space and generates descriptions of image regions.

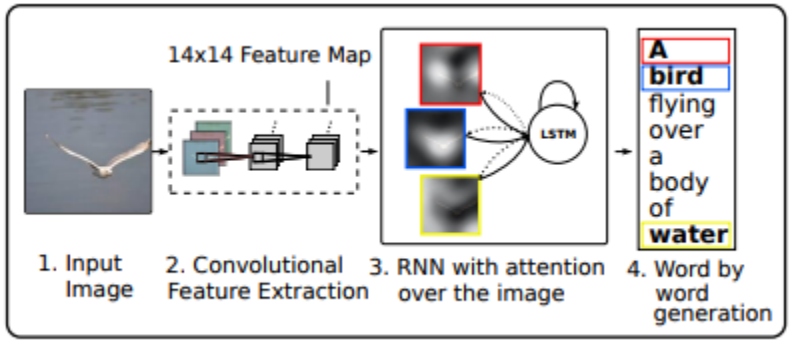
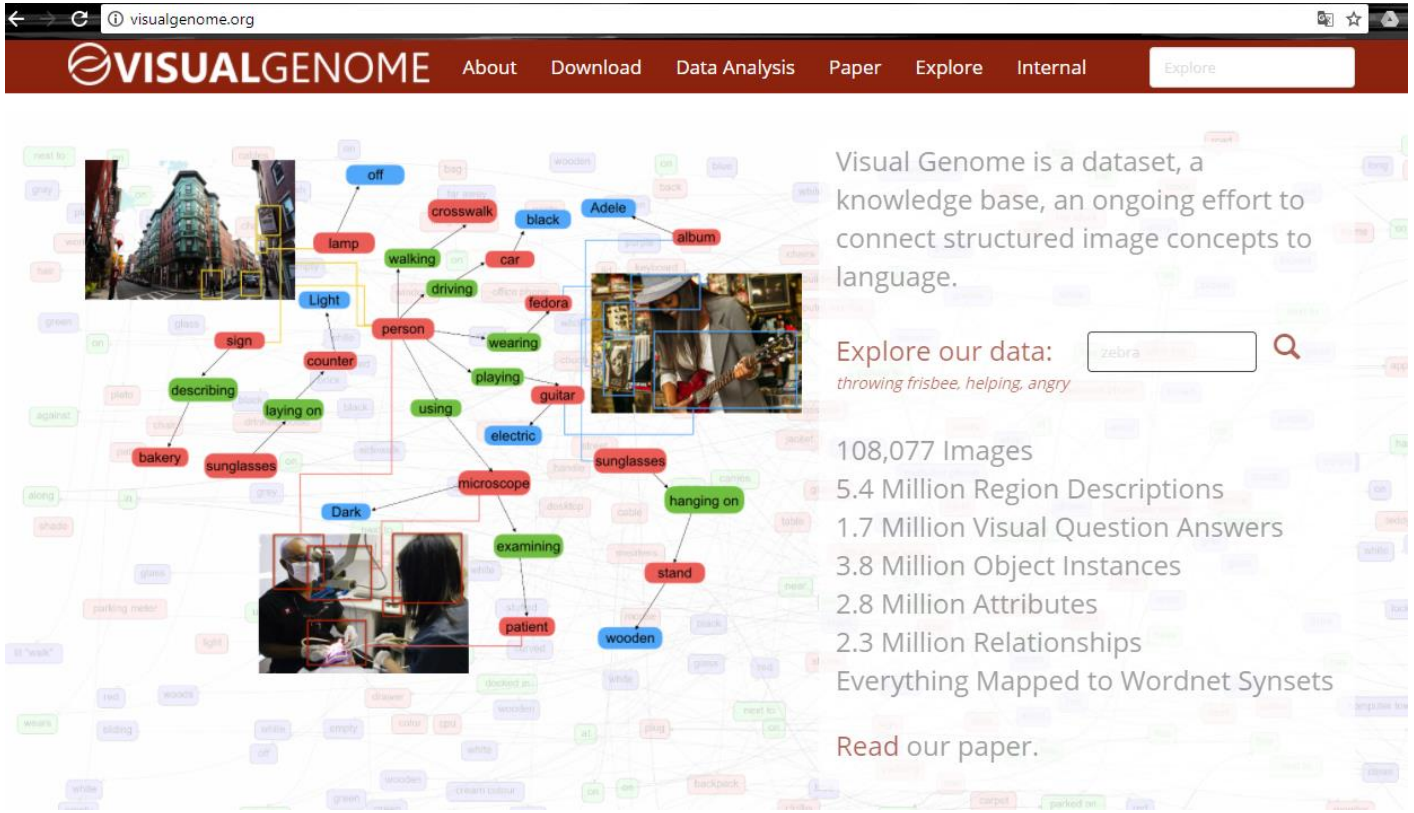


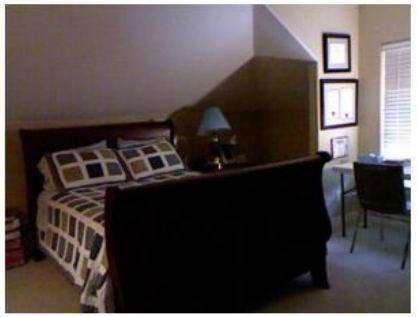
Image captioning datasets



Understanding Image:
Visual Question Answering (VQA)

VQA examples





What is on the right side of the cabinet?	How many drawers are there?	What is the largest object?
Neural-Image-QA: bed	3	bed
Language only: bed	6	table

Table 7. Examples of questions and answers. Correct predictions are colored in green, incorrect in red.



What is on the refrigerator?	What is the colour of the comforter?	What objects are found on the bed?
Neural-Image-QA: magnet, paper	blue, white	bed sheets, pillow
Language only: magnet, paper	blue, green, red, yellow	doll, pillow

VQA questions categories

Object Presence
Is there a traffic light in the photo?

Subordinate Object Recognition
What animal is in the picture?

Scene Classification
What is the weather like?

Sport Recognition
What sport are they playing?

Activity Recognition
What is the dog doing?

Other Attributes
What is the fence made of?

Counting
How many dogs are there?

Positional Reasoning
What is to the left of the woman?

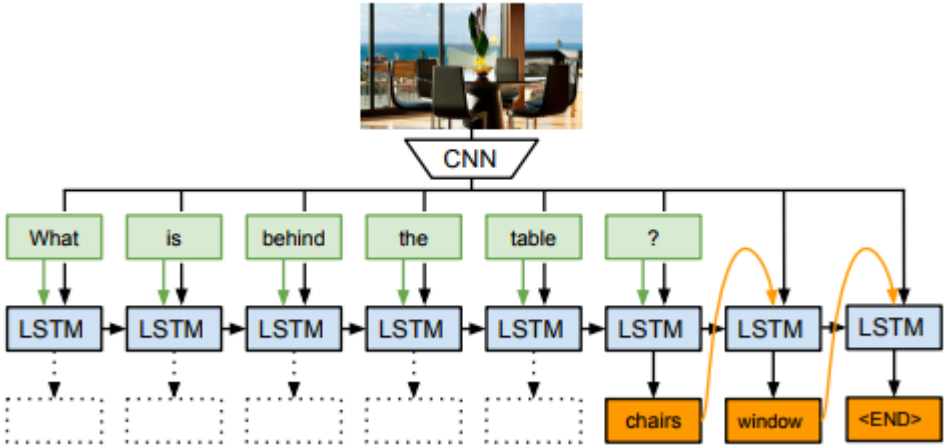
Absurd
What color is the couch?

Sentiment Understanding
How is the woman feeling?

Color Attributes
What color are the woman's shorts?

Utility/Affordance
What object can be thrown?

VQA models and datasets



	Images	Questions	Annotation Source	Question Types	Unique Answers	MC/OE
DAQUAR	1,449	16,590	Both	3	968	OE
COCO-QA	123,287	117,684	Auto	4	430	OE
COCO-VQA	204,721	614,163	Manual	3	145,172	Both
Visual7W	47,300	327,939	Manual	7	25,553	MC
Visual Genome	108,000	1,773,358	Manual	6	207,675	OE
TDIUC (This Paper)	167,437	1,654,167	Both	12	1,618	OE

Understanding Image: Visual Dialog

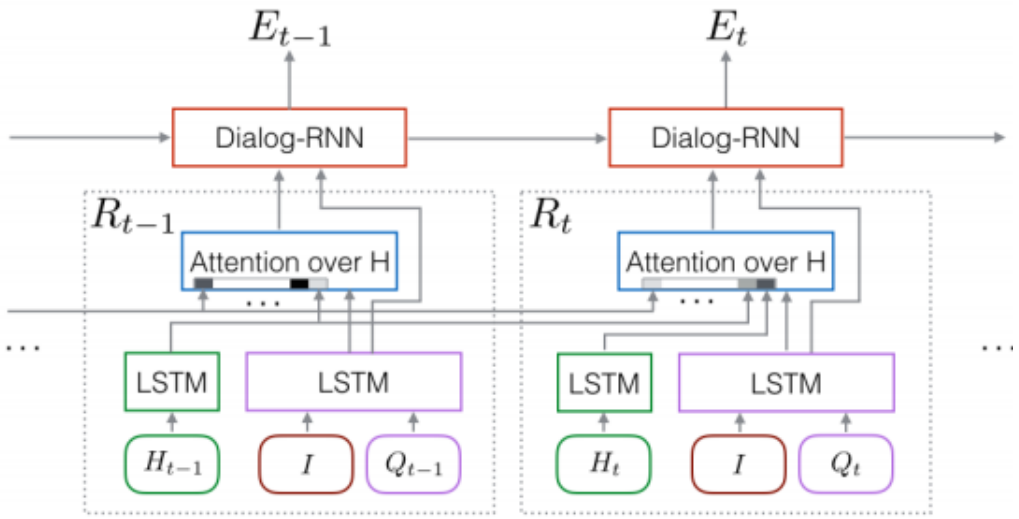
Visual dialog example

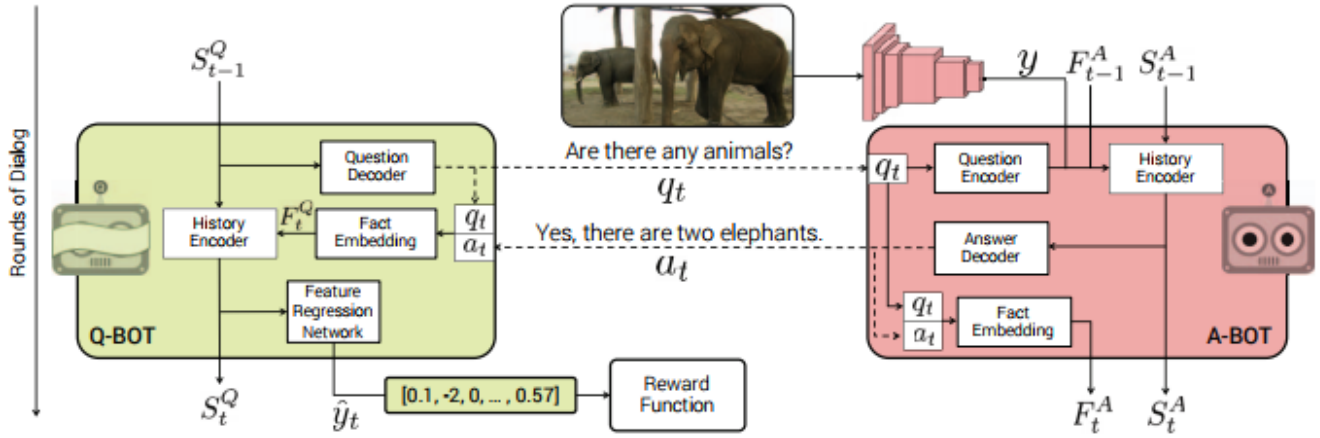
Visual dialog example

Image + Caption	Human Dialog		SL-pretrained		RL-full-QAf	
	What are their genders?	1 man 1 woman	How many people?	2	Are people male or female or both?	I think 1 is male, can't see other 1 ...
	Are they both adults?	Yes	Male or female?	I can't tell	What are they wearing?	They are all wearing snow pants and jackets
	Do they wear goggles?	Looks like sunglasses	What are they wearing?	I can't tell, they are far away	Is it snowing?	It does not appear to be snowing at moment
	Do they have hats on?	Man does	What color are skis?	I can't tell	Can you see any trees?	Yes, there are lot of trees in background
	Are there any other people?	No	Are they wearing goggles?	Yes	Can you see any buildings in background?	No, I can not
	What color is man's hat?	Black	Are they wearing goggles?	Yes	Can you see any trees?	Yes, there are lot of trees in background
	Is it snowing now?	No	Are they wearing goggles?	Yes	Can you see any buildings in background?	No, I do not see any buildings
	What is woman wearing?	Blue jacket and black pants	Are they wearing goggles?	Yes	Can you see any trees?	Yes, there are lot of trees in background
	Are they smiling?	Yes	Are they wearing hats?	Yes	Can you see any buildings in background?	No , I do not see any buildings
	Do you see trees?	Yes	Are they wearing goggles?	Yes	Does it look like they are in park?	It does not appear to be
A couple of people in the snow on skis.						
	How old is man?	In his teens	How old is man?	I can't see his face, I don't know	What color is skateboard?	It is hard to tell but I think it's black
	Is he wearing helmet?	No	What is he wearing?	shorts and t shirt	Is he wearing helmet?	Yes
	Is he wearing any protective gear?	No	What color is his shirt?	white	Is he wearing knee pads?	No, no knee pads
	What color is his hair?	Dirty blonde or dark brown	what color are his pants?	black	Is it day or night?	It is daytime
	What color is his shirt?	Black with red letters	what color is his shirt?	blue	Is it sunny?	It is bright out, but I see no shadows, so I can not
	Is he wearing shorts?	Yes	What color are his shoes?	Black	Can you see any other people?	There are 2 other people in background
	What color are his shorts?	Denim	What color are his shoes?	Black	Is it day or night?	It is daytime
	Are there other people visible?	No	What color are his shoes?	Black	Is it sunny?	Yes, it is sunny
	What color are his shoes?	I can't see them	What color are his shoes?	Black	What color is skateboard?	It is hard to tell but I think it's black
	Is this at skate park?	Yes	What color are his shoes?	Black	Is he wearing helmet?	Yes, he is wearing helmet
A man riding a skateboard down the side of a ramp.						

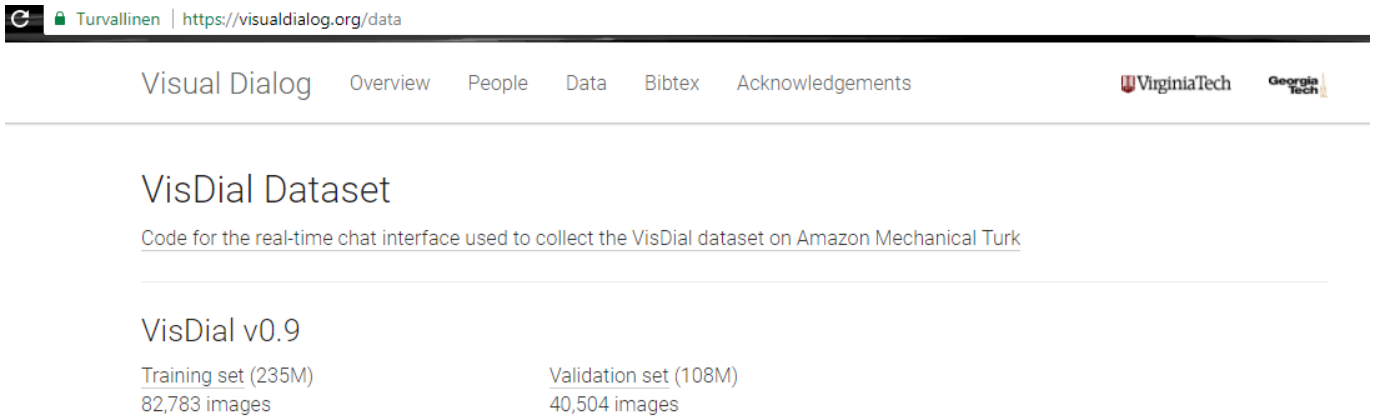
Table 1: Selected examples of Q-BOT-A-BOT interactions for SL-pretrained and RL-full-QAf. RL-full-QAf interactions are diverse, less prone to repetitive and safe exchanges ("can't tell", "don't know" *etc.*), and more image-discriminative.

Visual dialog models





Visual dialog datasets



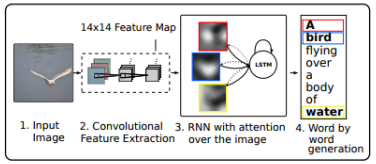
Undestanding Image: The Big Problem

Clever Hans

(by Larry Zitnick, 29.10.2017)

Current methods are inexplicable mappings from image to language

?



?

construction worker in orange safety vest is working on road.

one man and two women sitting in living room

man and woman are playing wii game while woman sits on couch with wine glass in her hand

group of people sitting on couch with their laptops

Understanding Image: Visual Reasoning

Visual reasoning example

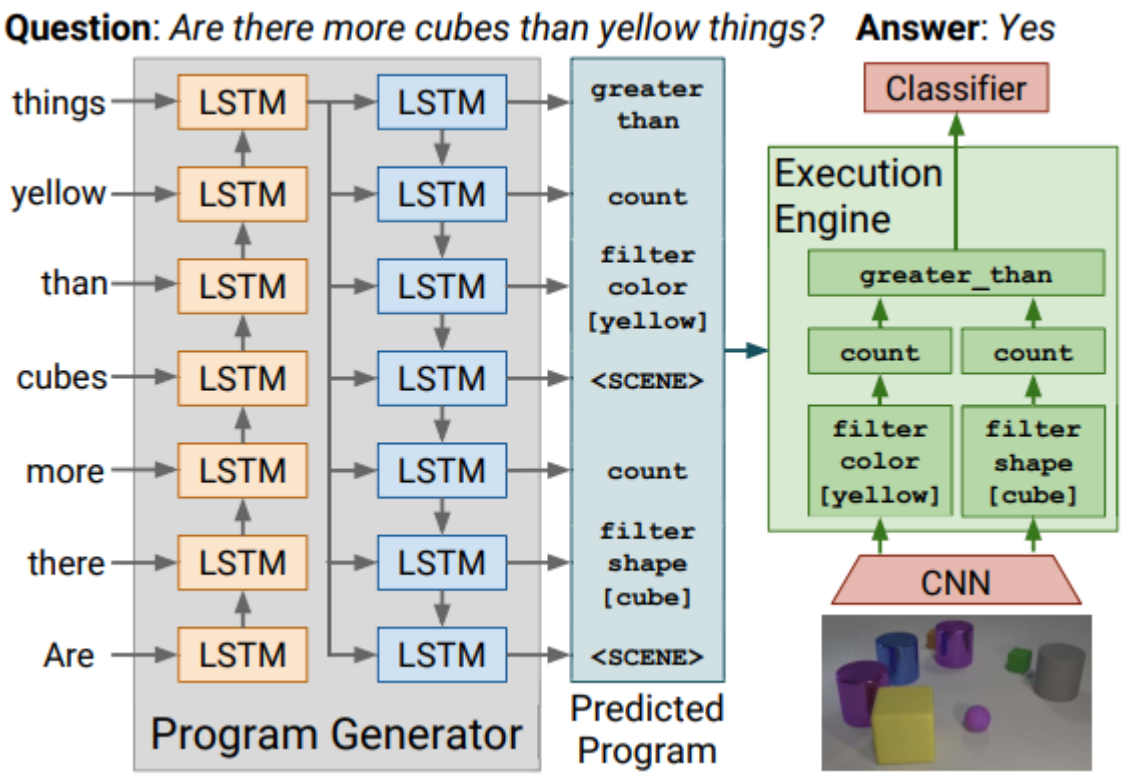
How many chairs are at the table?

Is there a pedestrian in my lane?

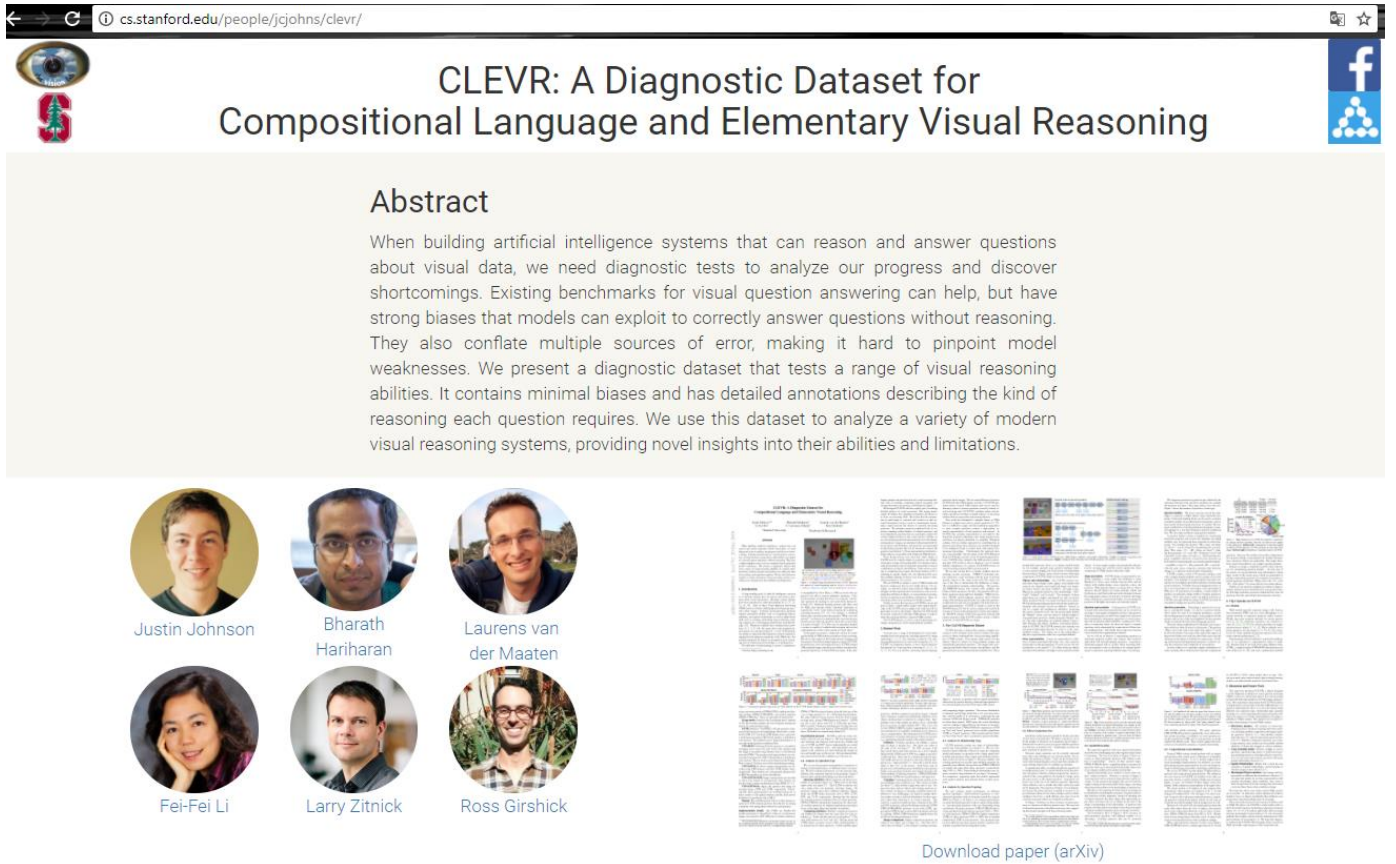
Is the person with the blue hat touching the bike in the back?

Is there a matte cube that has the same size as the red metal object?

Visual reasoning models



Visual reasoning datasets



Thesis

Preliminary work 2017

- Richard Szeliski: “Computer Vision: Algorithms and Applications”
- Sebastian Thrun, Wolfram Burgard, Dieter Fox: “Probabilistic Robotics”
- Kevin P. Murphy: ”Machine Learning. A Probabilistic Perspective”
- Stanford CS231n: ”Convolutional Neural Networks for Visual Recognition” (YouTube)
- Udacity: ”Introduction to Computer Vision” (YouTube)
- Udacity: ”Intro to Machine Learning” (YouTube)
- Udacity: ”Machine Learning” (YouTube)
- Udacity: ”Computational Photography” (YouTube)
- Daphne Koller: ”Probabilistic Graphical Models” (YouTube)
- Reviews on calculus, linear algebra, statistics, probability, …
- Kaggle competition
- Matlab, Python
- ICCV 2017

General information

Title:
Image-Based Situation Awareness

Doctoral Programme:
TUT Engineering Sciences, TTITO

Faculty:
Department of Automation Science and Engineering

Supervisor:
Professor Risto Ritala

Approved:
8.12.2017

Schedule:
2018-2021

Goal

“Develop methods which can perceive the elements in the environment within a volume of time and space, comprehend their meaning and project their status in the near future. The methods will be tested on a system which will help visually impaired people to understand their environment better. The system will be demonstrated on existing hardware.”

Reserch questions

1) By which methods an image-based perception of the environment can be produced, including object detection and localization in the field of view context?
2) By which methods a comprehension of the situation can be produced? This includes estimating object locations and their velocities in a 3d coordinate system, also for currently hidden objects.
3) By which methods a future projection of the situation can be created?
4) By which methods the comprehension and the future projection can be translated into human language?
5) How are uncertainty and reliability considered and represented?

Prototype

OrCam MyEye 2.0

“For blind and partially sighted people, an artificial vision device with a lightweight smart camera that instantly reads text aloud - from any surface – and recognizes faces, products, and money notes in real time.”

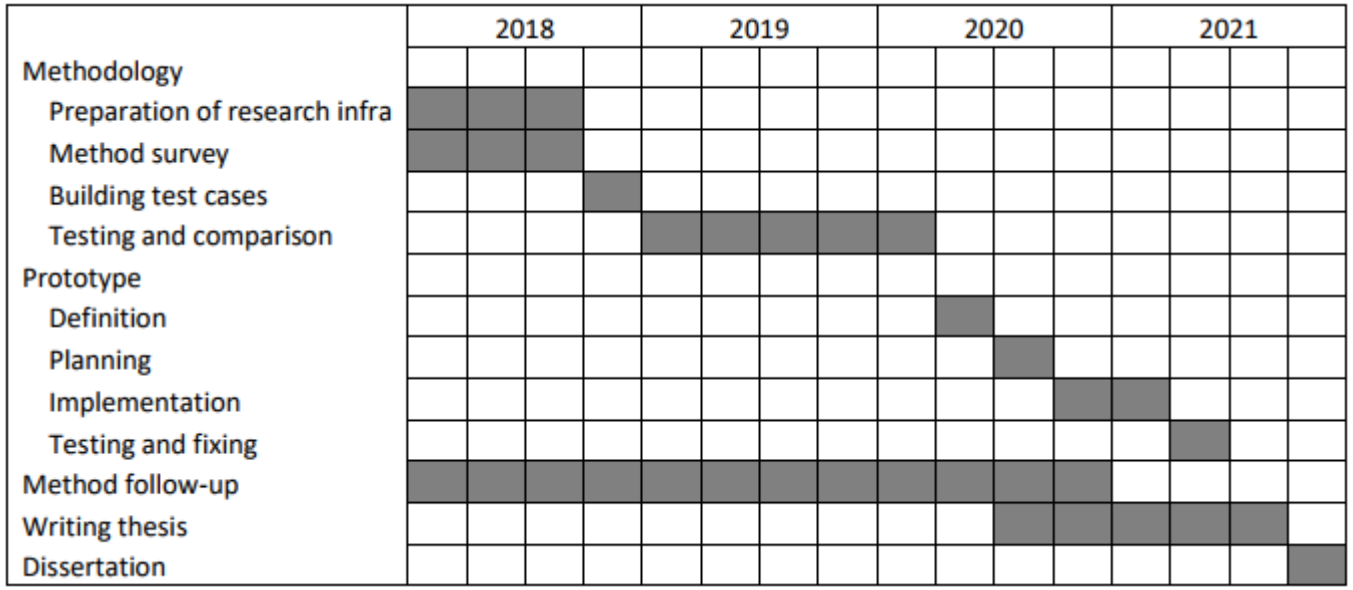
Thesis Prototype

A reliable and fast system which can help a visually impaired person to understand her environment better by accurate and rich description. This is accomplished by

- capturing images
- detecting objects
- comprehending situation
- describing environment
- projecting future
- implying warnings

The system will be demonstrated on physical hardware, preferably mobile.

Schedule


2018
2019
2020
2021
Methodology
Preparation of research infra
Method survey
Building test cases
Testing and comparison
Prototype
Definition
Planning
Implementation
Testing and fixing
Method follow-up
Writing thesis
Dissertation

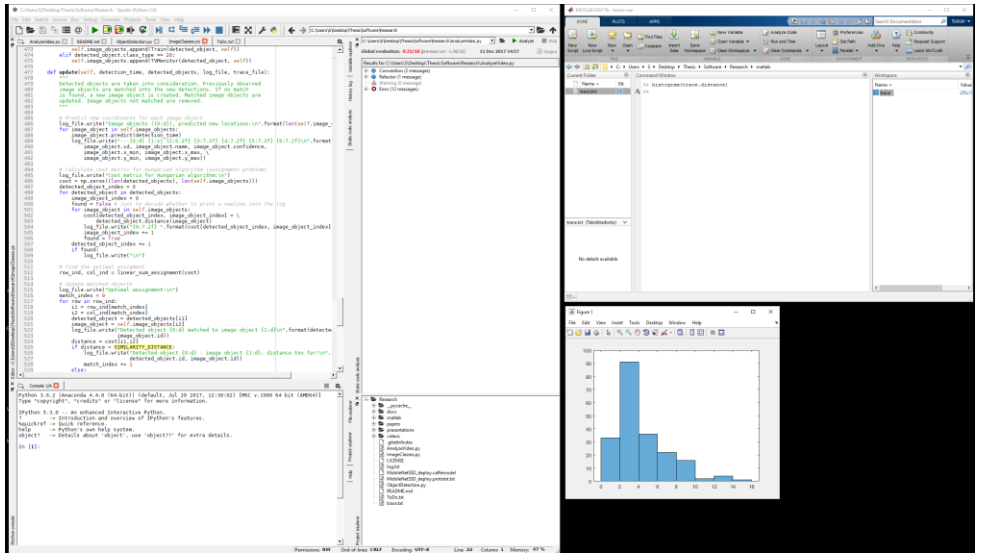
Supporting studies

2018-2019
TTI-90006 Orientation to Doctoral Studies, 3 cr
BMT-51506 Human Visual System, 5 cr
ASE-7516 Dynamic Planning with Incomplete Information, 5 cr
2019-2020
SGN-31007 Advanced Image Processing, 5 cr
ASE-7536 Model-Based Estimation, 5-7 cr
SGN-25006 Vector Space Methods for Signal and Image Processing, 5 cr
2020-2021
MAT-75006 Artificial Intelligence, 7 cr
SGN-43006 Knowledge Mining and Big Data, 5 cr
TTI-90016 Essay on Philosophy of Science/Research Ethics, 2 cr

Goodfellow, Bengio, Courville: “Deep Learning”
Udacity, Youtube courses on machine learning and computer vision

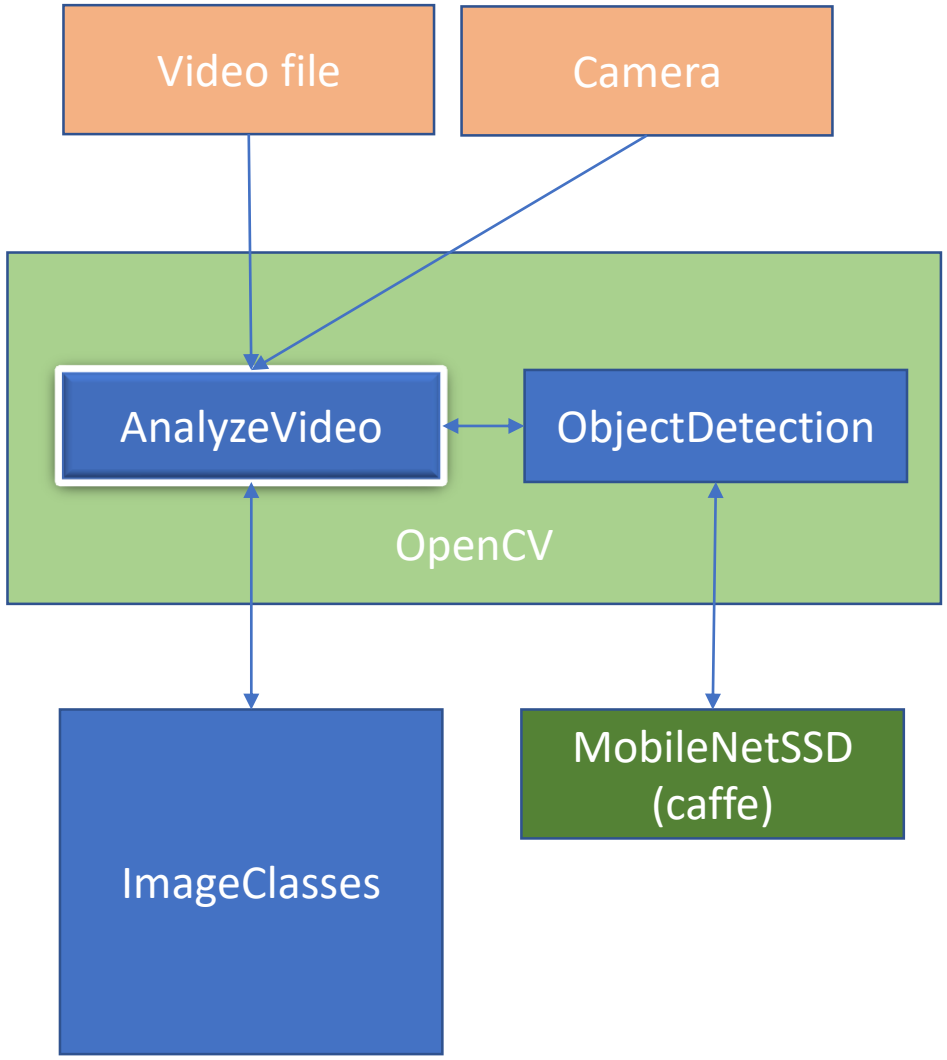
Research environment

- Software
  - Matlab
  - Python, Anaconda, Spyder
  - OpenCV
  - Keras, TensorFlow, (TensorFlow Lite, if required)
  - NumPy, Pandas, MatPlotLib, SkLearn, SciPy, SeaBorn,…
  - (PyTorch, Caffe2Go, if required)
  - Visual Studio - prototype user interface
- Hardware
  - Intel i7-4790 4GHz / 16Gt Ram
  - NVIDIA GeForce GTX 780 Ti (CUDA)
  - (Google Cloud Platform, if required)



Research environment

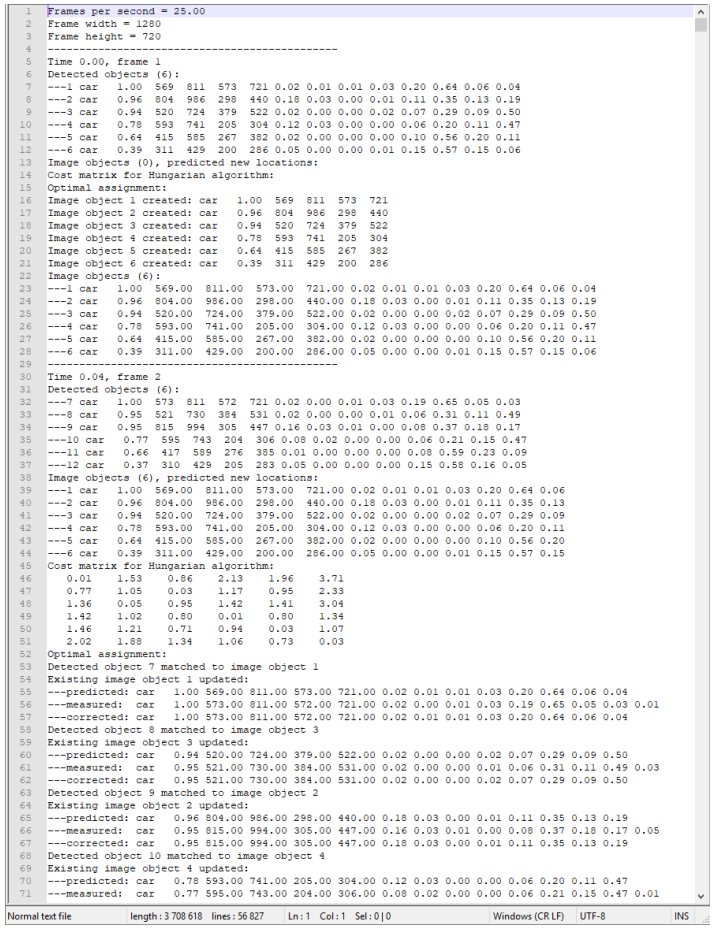
Software architecture



Research environment

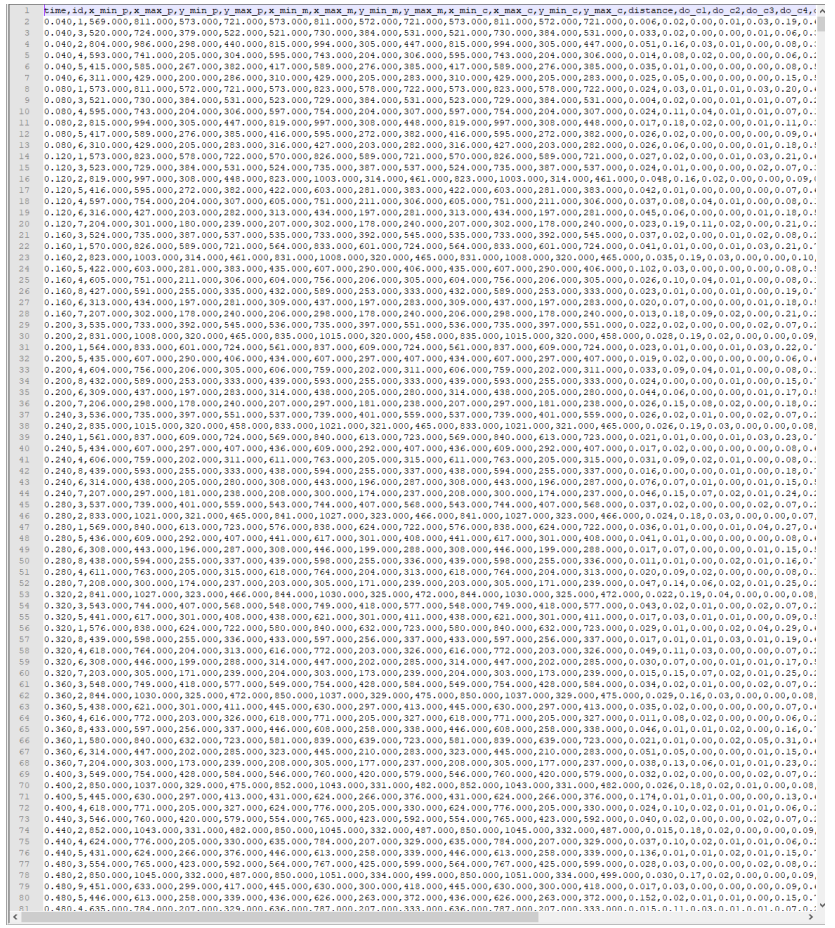
- Video processing, frame by frame
- Python 3.6.2, OpenCV 3.3.0
- Single Shot Detector + MobileNets (Google)

Research environment



```
Frames per second = 25.00
Frame width = 1280
Frame height = 720
-------------------------------------------------
Time 0.00, frame 1
Detected objects (6):
---1 car   1.00  569  811  573  721 0.02 0.01 0.01 0.03 0.20 0.64 0.06 0.04
---2 car   0.96  804  986  298  440 0.18 0.03 0.00 0.01 0.11 0.35 0.13 0.19
---3 car   0.94  520  724  379  522 0.02 0.00 0.00 0.02 0.07 0.29 0.09 0.50
---4 car   0.78  593  741  205  304 0.12 0.03 0.00 0.00 0.06 0.20 0.11 0.47
---5 car   0.64  415  585  267  382 0.02 0.00 0.00 0.00 0.10 0.56 0.20 0.11
---6 car   0.39  311  429  200  286 0.05 0.00 0.00 0.01 0.15 0.57 0.15 0.06
Image objects (0), predicted new locations:
Cost matrix for Hungarian algorithm:
Optimal assignment:
Image object 1 created: car   1.00  569  811  573  721
Image object 2 created: car   0.96  804  986  298  440
Image object 3 created: car   0.94  520  724  379  522
Image object 4 created: car   0.78  593  741  205  304
Image object 5 created: car   0.64  415  585  267  382
Image object 6 created: car   0.39  311  429  200  286
Image objects (6):
---1 car   1.00  569.00  811.00  573.00  721.00 0.02 0.01 0.01 0.03 0.20 0.64 0.06 0.04
---2 car   0.96  804.00  986.00  298.00  440.00 0.18 0.03 0.00 0.01 0.11 0.35 0.13 0.19
---3 car   0.94  520.00  724.00  379.00  522.00 0.02 0.00 0.00 0.02 0.07 0.29 0.09 0.50
---4 car   0.78  593.00  741.00  205.00  304.00 0.12 0.03 0.00 0.00 0.06 0.20 0.11 0.47
---5 car   0.64  415.00  585.00  267.00  382.00 0.02 0.00 0.00 0.00 0.10 0.56 0.20 0.11
---6 car   0.39  311.00  429.00  200.00  286.00 0.05 0.00 0.00 0.01 0.15 0.57 0.15 0.06
-------------------------------------------------
Time 0.04, frame 2
Detected objects (6):
---7 car   1.00  573  811  572  721 0.02 0.00 0.01 0.03 0.19 0.65 0.05 0.03
---8 car   0.95  521  730  384  531 0.02 0.00 0.00 0.01 0.06 0.31 0.11 0.49
---9 car   0.95  815  994  305  447 0.16 0.03 0.01 0.00 0.08 0.37 0.18 0.17
---10 car   0.77  595  743  204  306 0.08 0.02 0.00 0.00 0.06 0.21 0.15 0.47
---11 car   0.66  417  589  276  385 0.01 0.00 0.00 0.00 0.08 0.59 0.23 0.09
---12 car   0.37  310  429  205  283 0.05 0.00 0.00 0.00 0.15 0.58 0.16 0.05
Image objects (6), predicted new locations:
---1 car   1.00  569.00  811.00  573.00  721.00 0.02 0.01 0.01 0.03 0.20 0.64 0.06
---2 car   0.96  804.00  986.00  298.00  440.00 0.18 0.03 0.00 0.01 0.11 0.35 0.13
---3 car   0.94  520.00  724.00  379.00  522.00 0.02 0.00 0.00 0.02 0.07 0.29 0.09
---4 car   0.78  593.00  741.00  205.00  304.00 0.12 0.03 0.00 0.00 0.06 0.20 0.11
---5 car   0.64  415.00  585.00  267.00  382.00 0.02 0.00 0.00 0.00 0.10 0.56 0.20
---6 car   0.39  311.00  429.00  200.00  286.00 0.05 0.00 0.00 0.01 0.15 0.57 0.15
Cost matrix for Hungarian algorithm:
   0.01    1.53    0.86    2.13    1.96    3.71
   0.77    1.05    0.03    1.17    0.95    2.33
   1.36    0.05    0.95    1.42    1.41    3.04
   1.42    1.02    0.80    0.01    0.80    1.34
   1.46    1.21    0.71    0.94    0.03    1.07
   2.02    1.88    1.34    1.06    0.73    0.03
Optimal assignment:
Detected object 7 matched to image object 1
Existing image object 1 updated:
---predicted: car   1.00 569.00 811.00 573.00 721.00 0.02 0.01 0.01 0.03 0.20 0.64 0.06 0.04
---measured:  car   1.00 573.00 811.00 572.00 721.00 0.02 0.00 0.01 0.03 0.19 0.65 0.05 0.03 0.01
---corrected: car   1.00 573.00 811.00 572.00 721.00 0.02 0.01 0.01 0.03 0.20 0.64 0.06 0.04
Detected object 8 matched to image object 3
Existing image object 3 updated:
---predicted: car   0.94 520.00 724.00 379.00 522.00 0.02 0.00 0.00 0.02 0.07 0.29 0.09 0.50
---measured:  car   0.95 521.00 730.00 384.00 531.00 0.02 0.00 0.00 0.01 0.06 0.31 0.11 0.49 0.03
---corrected: car   0.95 521.00 730.00 384.00 531.00 0.02 0.00 0.00 0.02 0.07 0.29 0.09 0.50
Detected object 9 matched to image object 2
Existing image object 2 updated:
---predicted: car   0.96 804.00 986.00 298.00 440.00 0.18 0.03 0.00 0.01 0.11 0.35 0.13 0.19
---measured:  car   0.95 815.00 994.00 305.00 447.00 0.16 0.03 0.01 0.00 0.08 0.37 0.18 0.17 0.05
---corrected: car   0.95 815.00 994.00 305.00 447.00 0.18 0.03 0.00 0.01 0.11 0.35 0.13 0.19
Detected object 10 matched to image object 4
Existing image object 4 updated:
---predicted: car   0.78 593.00 741.00 205.00 304.00 0.12 0.03 0.00 0.00 0.06 0.20 0.11 0.47
---measured:  car   0.77 595.00 743.00 204.00 306.00 0.08 0.02 0.00 0.00 0.06 0.21 0.15 0.47 0.01
```

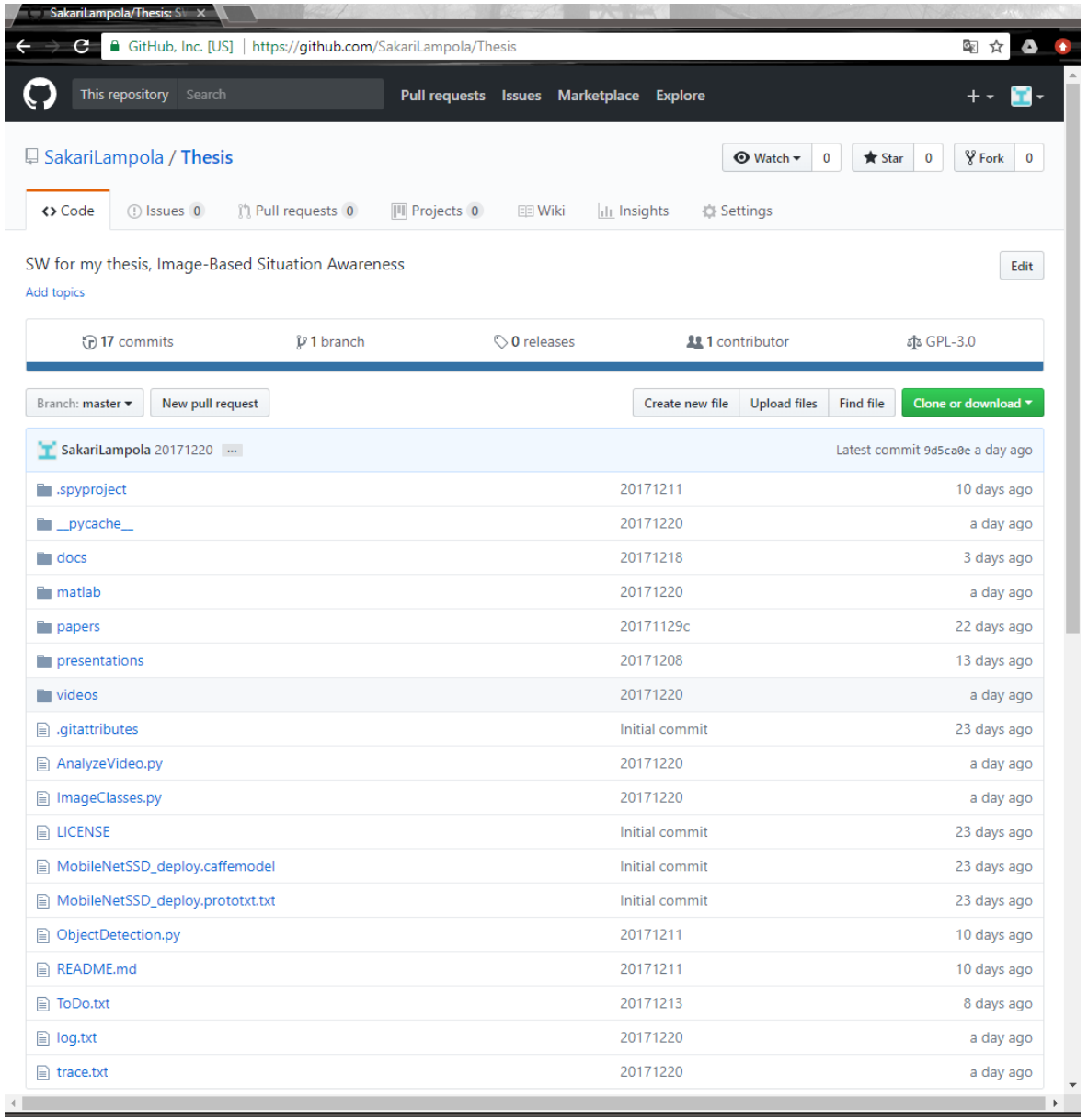
Normal text file | length : 3 708 618 | lines : 56 827 | Ln : 1 | Col : 1 | Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS



```
time,id,x_min_p,x_max_p,y_min_p,y_max_p,x_min_m,x_max_m,y_min_m,y_max_m,x_min_c,x_max_c,y_min_c,y_max_c,distance,do_c1,do_c2,do_c3,do_c4,
0.040,1,569.000,811.000,573.000,721.000,573.000,811.000,572.000,721.000,573.000,811.000,572.000,721.000,0.006,0.02,0.00,0.01,0.03,0.19,0.
0.040,3,520.000,724.000,379.000,522.000,521.000,730.000,384.000,531.000,521.000,730.000,384.000,531.000,0.033,0.02,0.00,0.00,0.01,0.06,0.
0.040,2,804.000,986.000,298.000,440.000,815.000,994.000,305.000,447.000,815.000,994.000,305.000,447.000,0.051,0.16,0.03,0.01,0.00,0.08,0.
0.040,4,593.000,741.000,205.000,304.000,595.000,743.000,204.000,306.000,595.000,743.000,204.000,306.000,0.014,0.08,0.02,0.00,0.00,0.06,0.
0.040,5,415.000,585.000,267.000,382.000,417.000,589.000,276.000,385.000,417.000,589.000,276.000,385.000,0.035,0.01,0.00,0.00,0.00,0.08,0.
0.040,6,311.000,429.000,200.000,286.000,310.000,429.000,205.000,283.000,310.000,429.000,205.000,283.000,0.025,0.05,0.00,0.00,0.00,0.15,0.
0.080,1,573.000,811.000,572.000,721.000,573.000,823.000,578.000,722.000,573.000,823.000,578.000,722.000,0.024,0.03,0.01,0.01,0.03,0.20,0.
0.080,3,521.000,730.000,384.000,531.000,523.000,729.000,384.000,531.000,523.000,729.000,384.000,531.000,0.004,0.02,0.00,0.01,0.01,0.07,0.
0.080,4,595.000,743.000,204.000,306.000,597.000,754.000,204.000,307.000,597.000,754.000,204.000,307.000,0.024,0.11,0.04,0.01,0.01,0.07,0.
0.080,2,815.000,994.000,305.000,447.000,819.000,997.000,308.000,448.000,819.000,997.000,308.000,448.000,0.017,0.18,0.02,0.00,0.01,0.11,0.
0.080,5,417.000,589.000,276.000,385.000,416.000,595.000,272.000,382.000,416.000,595.000,272.000,382.000,0.026,0.02,0.00,0.00,0.00,0.09,0.
0.080,6,310.000,429.000,205.000,283.000,316.000,427.000,203.000,282.000,316.000,427.000,203.000,282.000,0.026,0.06,0.00,0.00,0.01,0.18,0.
0.120,1,573.000,823.000,578.000,722.000,570.000,826.000,589.000,721.000,570.000,826.000,589.000,721.000,0.027,0.02,0.00,0.01,0.03,0.21,0.
0.120,3,523.000,729.000,384.000,531.000,524.000,735.000,387.000,537.000,524.000,735.000,387.000,537.000,0.024,0.01,0.00,0.00,0.02,0.07,0.
0.120,2,819.000,997.000,308.000,448.000,823.000,1003.000,314.000,461.000,823.000,1003.000,314.000,461.000,0.048,0.16,0.02,0.00,0.00,0.09,0.
0.120,5,416.000,595.000,272.000,382.000,422.000,603.000,281.000,383.000,422.000,603.000,281.000,383.000,0.042,0.01,0.00,0.00,0.00,0.07,0.
0.120,4,597.000,754.000,204.000,307.000,605.000,751.000,211.000,306.000,605.000,751.000,211.000,306.000,0.037,0.08,0.04,0.01,0.00,0.08,0.
0.120,6,316.000,427.000,203.000,282.000,313.000,434.000,197.000,281.000,313.000,434.000,197.000,281.000,0.045,0.06,0.00,0.00,0.01,0.18,0.
0.120,7,204.000,301.000,180.000,239.000,207.000,302.000,178.000,240.000,207.000,302.000,178.000,240.000,0.023,0.19,0.11,0.02,0.00,0.21,0.
0.160,3,524.000,735.000,387.000,537.000,535.000,733.000,392.000,545.000,535.000,733.000,392.000,545.000,0.037,0.02,0.00,0.01,0.02,0.06,0.
0.160,1,570.000,826.000,589.000,721.000,564.000,833.000,601.000,724.000,564.000,833.000,601.000,724.000,0.041,0.01,0.00,0.01,0.03,0.21,0.
0.160,2,823.000,1003.000,314.000,461.000,831.000,1008.000,320.000,465.000,831.000,1008.000,320.000,465.000,0.035,0.19,0.03,0.00,0.00,0.10,
0.160,5,422.000,603.000,281.000,383.000,435.000,607.000,290.000,406.000,435.000,607.000,290.000,406.000,0.102,0.03,0.00,0.00,0.00,0.08,0.
0.160,4,605.000,751.000,211.000,306.000,604.000,756.000,206.000,305.000,604.000,756.000,206.000,305.000,0.026,0.10,0.04,0.01,0.00,0.08,0.
0.160,8,427.000,591.000,255.000,335.000,432.000,589.000,253.000,333.000,432.000,589.000,253.000,333.000,0.023,0.01,0.00,0.00,0.00,0.15,0.
0.160,6,313.000,434.000,197.000,281.000,309.000,437.000,197.000,283.000,309.000,437.000,197.000,283.000,0.020,0.07,0.00,0.00,0.01,0.18,0.
0.160,7,207.000,302.000,178.000,240.000,206.000,298.000,178.000,240.000,206.000,298.000,178.000,240.000,0.013,0.18,0.09,0.02,0.00,0.21,0.
0.200,3,535.000,733.000,392.000,545.000,536.000,735.000,397.000,551.000,536.000,735.000,397.000,551.000,0.022,0.02,0.00,0.00,0.02,0.07,0.
0.200,2,831.000,1008.000,320.000,465.000,835.000,1015.000,320.000,458.000,835.000,1015.000,320.000,458.000,0.028,0.19,0.02,0.00,0.00,0.09,
0.200,1,564.000,833.000,601.000,724.000,561.000,837.000,609.000,724.000,561.000,837.000,609.000,724.000,0.023,0.01,0.00,0.01,0.03,0.22,0.
0.200,5,435.000,607.000,290.000,406.000,434.000,607.000,297.000,407.000,434.000,607.000,297.000,407.000,0.019,0.02,0.00,0.00,0.00,0.06,0.
0.200,4,604.000,756.000,206.000,305.000,606.000,759.000,202.000,311.000,606.000,759.000,202.000,311.000,0.033,0.09,0.04,0.01,0.00,0.08,0.
0.200,8,432.000,589.000,253.000,333.000,439.000,593.000,255.000,333.000,439.000,593.000,255.000,333.000,0.024,0.00,0.00,0.01,0.00,0.15,0.
0.200,6,309.000,437.000,197.000,283.000,314.000,438.000,205.000,280.000,314.000,438.000,205.000,280.000,0.044,0.06,0.00,0.00,0.01,0.17,0.
0.200,7,206.000,298.000,178.000,240.000,207.000,297.000,181.000,238.000,207.000,297.000,181.000,238.000,0.026,0.15,0.08,0.02,0.00,0.18,0.
0.240,3,536.000,735.000,397.000,551.000,537.000,739.000,401.000,559.000,537.000,739.000,401.000,559.000,0.026,0.02,0.01,0.00,0.02,0.07,0.
0.240,2,835.000,1015.000,320.000,458.000,833.000,1021.000,321.000,465.000,833.000,1021.000,321.000,465.000,0.026,0.19,0.03,0.00,0.00,0.08,
0.240,1,561.000,837.000,609.000,724.000,569.000,840.000,613.000,723.000,569.000,840.000,613.000,723.000,0.021,0.01,0.00,0.01,0.03,0.23,0.
0.240,5,434.000,607.000,297.000,407.000,436.000,609.000,292.000,407.000,436.000,609.000,292.000,407.000,0.017,0.02,0.00,0.00,0.00,0.08,0.
0.240,4,606.000,759.000,202.000,311.000,611.000,763.000,205.000,315.000,611.000,763.000,205.000,315.000,0.031,0.09,0.02,0.01,0.00,0.08,0.
0.240,8,439.000,593.000,255.000,333.000,438.000,594.000,255.000,337.000,438.000,594.000,255.000,337.000,0.016,0.00,0.00,0.01,0.00,0.18,0.
0.240,6,314.000,438.000,205.000,280.000,308.000,443.000,196.000,287.000,308.000,443.000,196.000,287.000,0.076,0.07,0.01,0.00,0.01,0.15,0.
0.240,7,207.000,297.000,181.000,238.000,208.000,300.000,174.000,237.000,208.000,300.000,174.000,237.000,0.046,0.15,0.07,0.02,0.01,0.24,0.
0.280,3,537.000,739.000,401.000,559.000,543.000,744.000,407.000,568.000,543.000,744.000,407.000,568.000,0.037,0.02,0.00,0.00,0.02,0.07,0.
0.280,2,833.000,1021.000,321.000,465.000,841.000,1027.000,323.000,466.000,841.000,1027.000,323.000,466.000,0.024,0.18,0.03,0.00,0.00,0.08,
0.280,1,569.000,840.000,613.000,723.000,576.000,838.000,624.000,722.000,576.000,838.000,624.000,722.000,0.036,0.01,0.00,0.01,0.04,0.27,0.
0.280,5,436.000,609.000,292.000,407.000,441.000,617.000,301.000,408.000,441.000,617.000,301.000,408.000,0.041,0.01,0.00,0.00,0.00,0.08,0.
0.280,6,308.000,443.000,196.000,287.000,308.000,446.000,199.000,288.000,308.000,446.000,199.000,288.000,0.017,0.07,0.00,0.00,0.01,0.15,0.
0.280,8,438.000,594.000,255.000,337.000,439.000,598.000,255.000,336.000,439.000,598.000,255.000,336.000,0.011,0.01,0.00,0.00,0.02,0.01,0.16,0.
0.280,4,611.000,763.000,205.000,315.000,618.000,764.000,204.000,313.000,618.000,764.000,204.000,313.000,0.020,0.09,0.02,0.00,0.00,0.08,0.
0.280,7,208.000,300.000,174.000,237.000,203.000,305.000,171.000,239.000,203.000,305.000,171.000,239.000,0.047,0.14,0.06,0.02,0.01,0.25,0.
0.320,2,841.000,1027.000,323.000,466.000,844.000,1030.000,325.000,472.000,844.000,1030.000,325.000,472.000,0.022,0.19,0.04,0.00,0.00,0.08,
0.320,3,543.000,744.000,407.000,568.000,548.000,749.000,418.000,577.000,548.000,749.000,418.000,577.000,0.043,0.02,0.01,0.00,0.02,0.07,0.
0.320,5,441.000,617.000,301.000,408.000,438.000,621.000,301.000,411.000,438.000,621.000,301.000,411.000,0.017,0.03,0.01,0.01,0.00,0.09,0.
0.320,1,576.000,838.000,624.000,722.000,580.000,840.000,632.000,723.000,580.000,840.000,632.000,723.000,0.029,0.01,0.00,0.02,0.04,0.29,0.
0.320,8,439.000,598.000,255.000,336.000,433.000,597.000,256.000,337.000,433.000,597.000,256.000,337.000,0.017,0.01,0.01,0.03,0.01,0.19,0.
0.320,4,618.000,764.000,204.000,313.000,616.000,772.000,203.000,326.000,616.000,772.000,203.000,326.000,0.049,0.11,0.03,0.00,0.00,0.07,0.
0.320,6,308.000,446.000,199.000,288.000,314.000,447.000,202.000,285.000,314.000,447.000,202.000,285.000,0.030,0.07,0.00,0.01,0.01,0.17,0.
0.320,7,203.000,305.000,171.000,239.000,204.000,303.000,173.000,239.000,204.000,303.000,173.000,239.000,0.015,0.15,0.07,0.02,0.01,0.25,0.
0.360,1,548.000,749.000,418.000,577.000,549.000,754.000,428.000,584.000,549.000,754.000,428.000,584.000,0.034,0.02,0.01,0.00,0.02,0.07,0.
0.360,2,844.000,1030.000,325.000,472.000,850.000,1037.000,329.000,475.000,850.000,1037.000,329.000,475.000,0.029,0.16,0.03,0.00,0.00,0.08,
0.360,5,438.000,621.000,301.000,411.000,445.000,630.000,297.000,413.000,445.000,630.000,297.000,413.000,0.035,0.02,0.00,0.00,0.00,0.07,0.
0.360,4,616.000,772.000,203.000,326.000,618.000,771.000,205.000,327.000,618.000,771.000,205.000,327.000,0.011,0.08,0.02,0.00,0.00,0.06,0.
0.360,8,433.000,597.000,256.000,337.000,446.000,608.000,258.000,338.000,446.000,608.000,258.000,338.000,0.046,0.01,0.01,0.02,0.00,0.16,0.
0.360,1,580.000,840.000,632.000,723.000,581.000,839.000,639.000,723.000,581.000,839.000,639.000,723.000,0.021,0.01,0.00,0.02,0.05,0.31,0.
0.360,6,314.000,447.000,202.000,285.000,323.000,445.000,210.000,283.000,323.000,445.000,210.000,283.000,0.051,0.05,0.00,0.00,0.01,0.15,0.
0.360,7,204.000,303.000,173.000,239.000,208.000,305.000,177.000,237.000,208.000,305.000,177.000,237.000,0.038,0.13,0.06,0.01,0.01,0.23,0.
0.400,3,549.000,754.000,428.000,584.000,546.000,760.000,420.000,579.000,546.000,760.000,420.000,579.000,0.032,0.02,0.00,0.00,0.02,0.07,0.
0.400,2,850.000,1037.000,329.000,475.000,852.000,1043.000,331.000,482.000,852.000,1043.000,331.000,482.000,0.026,0.18,0.02,0.01,0.00,0.08,
0.400,5,445.000,630.000,297.000,413.000,431.000,624.000,266.000,376.000,431.000,624.000,266.000,376.000,0.174,0.01,0.01,0.00,0.00,0.13,0.
0.400,4,618.000,771.000,205.000,327.000,624.000,776.000,205.000,330.000,624.000,776.000,205.000,330.000,0.024,0.10,0.02,0.01,0.01,0.06,0.
0.440,3,546.000,760.000,420.000,579.000,554.000,765.000,423.000,592.000,554.000,765.000,423.000,592.000,0.040,0.02,0.00,0.00,0.02,0.07,0.
0.440,2,852.000,1043.000,331.000,482.000,850.000,1045.000,332.000,487.000,850.000,1045.000,332.000,487.000,0.015,0.18,0.02,0.00,0.00,0.09,
0.440,4,624.000,776.000,205.000,330.000,635.000,784.000,207.000,329.000,635.000,784.000,207.000,329.000,0.037,0.10,0.02,0.01,0.01,0.06,0.
0.440,5,431.000,624.000,266.000,376.000,446.000,613.000,258.000,339.000,446.000,613.000,258.000,339.000,0.136,0.01,0.01,0.02,0.01,0.15,0.
0.480,3,554.000,765.000,423.000,592.000,564.000,767.000,425.000,599.000,564.000,767.000,425.000,599.000,0.028,0.03,0.00,0.00,0.02,0.08,0.
0.480,2,850.000,1045.000,332.000,487.000,850.000,1051.000,334.000,499.000,850.000,1051.000,334.000,499.000,0.030,0.17,0.02,0.00,0.00,0.09,
0.480,9,451.000,633.000,299.000,417.000,445.000,630.000,300.000,418.000,445.000,630.000,300.000,418.000,0.017,0.03,0.00,0.00,0.00,0.09,0.
0.480,5,446.000,613.000,258.000,339.000,436.000,626.000,263.000,372.000,436.000,626.000,263.000,372.000,0.152,0.02,0.01,0.01,0.00,0.15,0.
0.480,4,635.000,784.000,207.000,329.000,636.000,787.000,207.000,333.000,636.000,787.000,207.000,333.000,0.015,0.11,0.03,0.01,0.01,0.07,0.
```

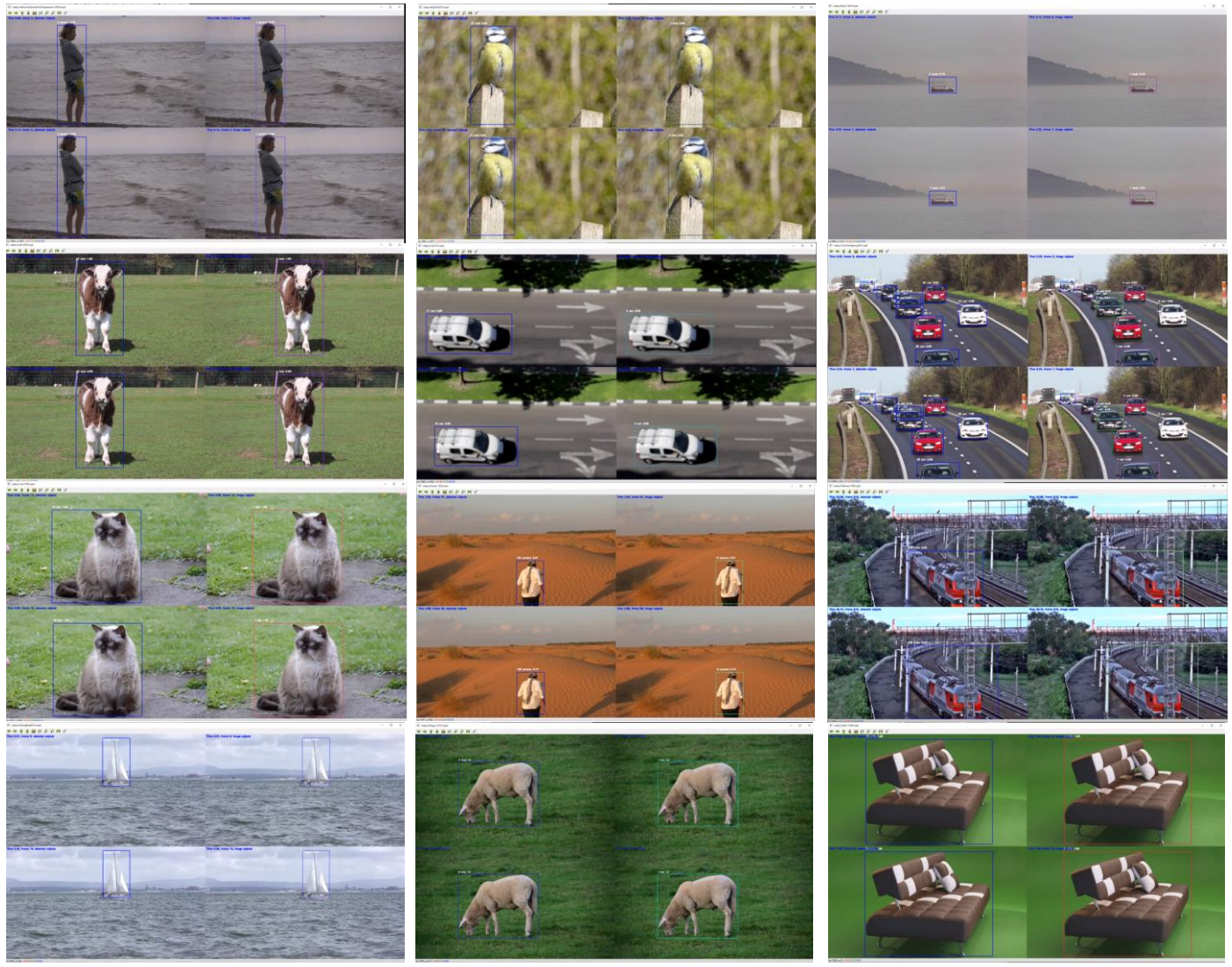
- Detailed log for every frame
- Trace file for external analysis

GitHub



https://github.com/SakariLampola/Thesis

Demos



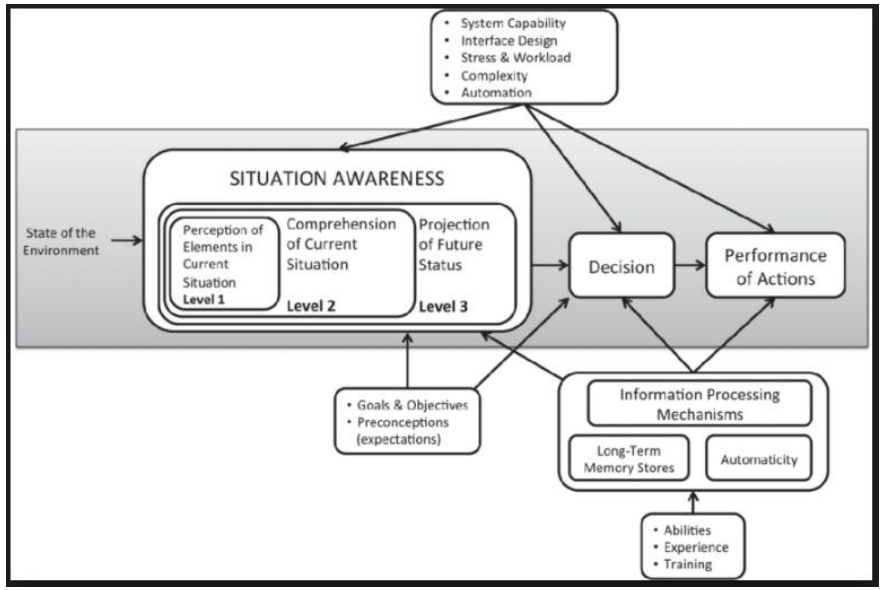
YouTube channel: Sakari Lampola

Situation Awareness

Definition

Wikipedia:

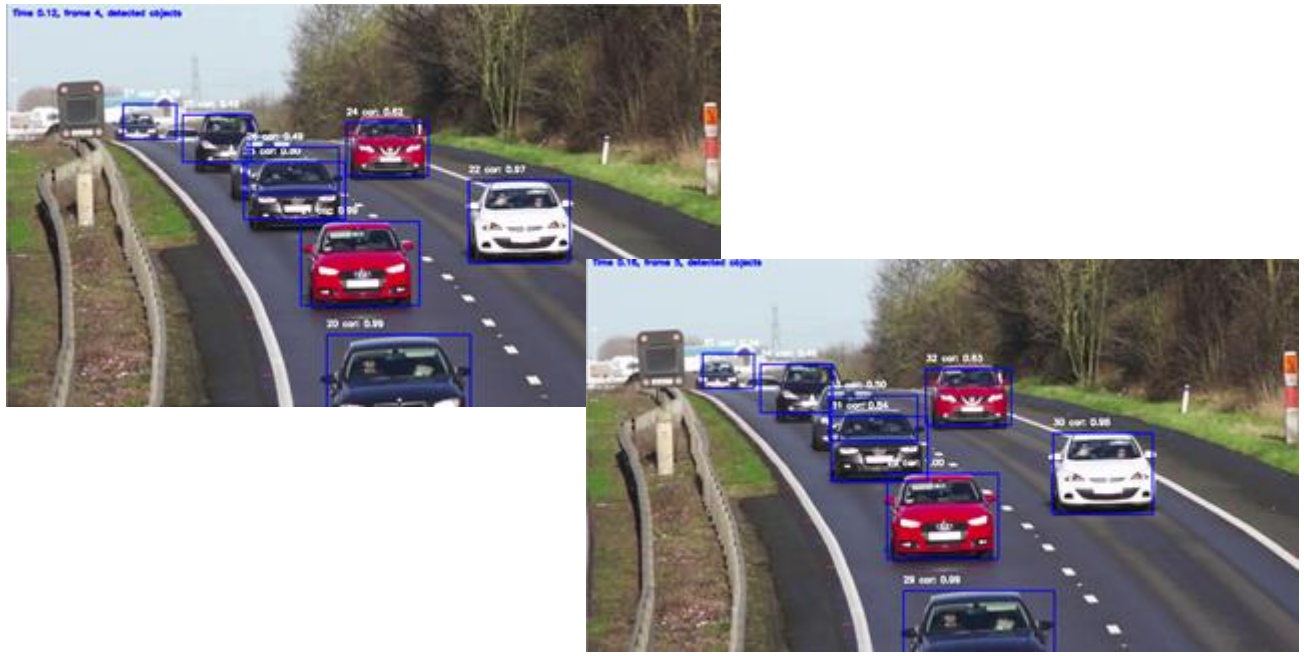
Situational awareness or **situation awareness** (SA) is the perception of environmental elements and events with respect to time or space, the comprehension of their meaning, and the projection of their status after some variable has changed, such as time, or some other variable, such as a predetermined event.
It is also a field of study concerned with understanding of the environment critical to decision-makers in complex, dynamic areas from aviation, air traffic control, ship navigation, power plant operations, military command and control, and emergency services such as fire fighting and policing; to more ordinary but nevertheless complex tasks such as driving an automobile or riding a bicycle.



Levels

Endsley's model:

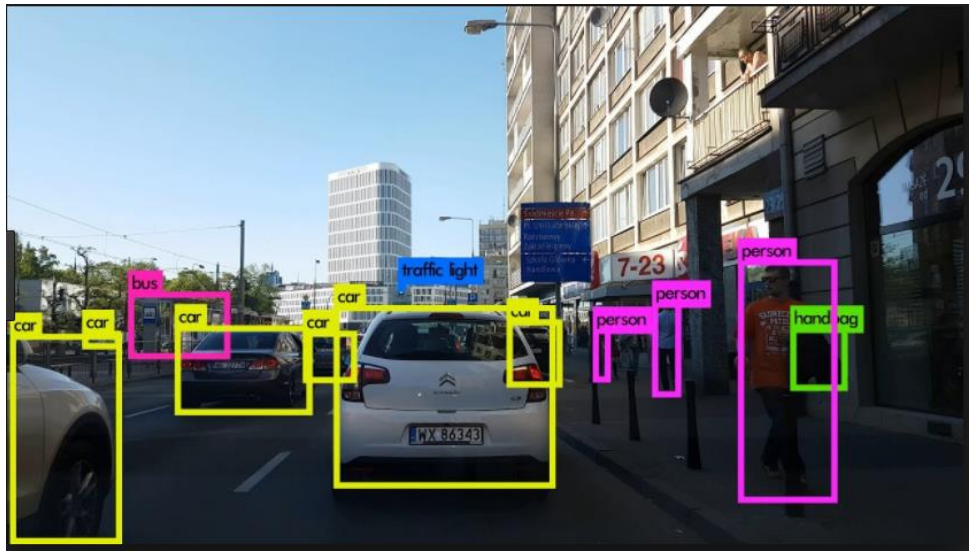
1. Perception of the elements in the current situation
2. Comprehension of the current situation
3. Projection of future status of the situation



N. A. Stanton, P. R. G. Chambers, and J. Piggott, "Situational awareness and safety," Safety Sci., vol. 39, no. 3, pp. 189–204, 2001

Perception

Perception of the elements in the current situation



Object detection

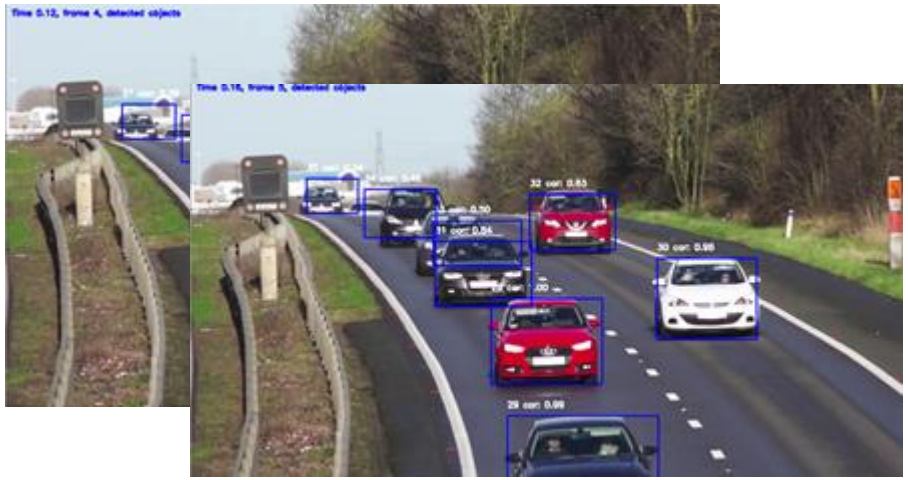
Algorithms:
- Faster R-CNN
- YOLO
- SSD
- …

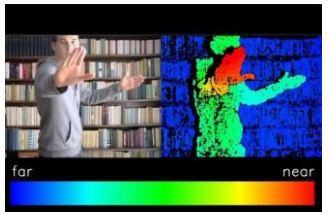
Output
- Classification
- Bounding box
- Reliability index

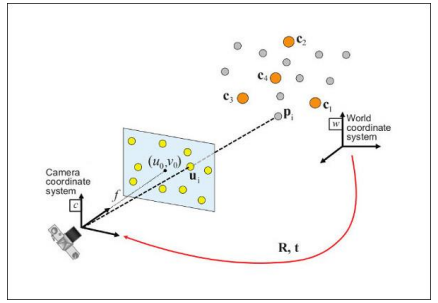
“The first step in achieving SA is to perceive the status, attributes, and dynamics of relevant elements in the environment. Thus, Level 1 SA, the most basic level of SA, involves the processes of monitoring, cue detection, and simple recognition, which lead to an awareness of multiple situational elements (objects, events, people, systems, environmental factors) and their current states (locations, conditions, modes, actions).”

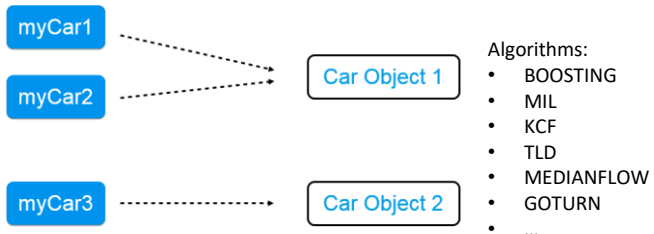
Comprehension

Comprehension of the current situation



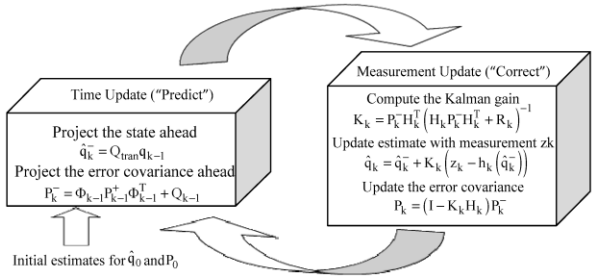






Algorithms:

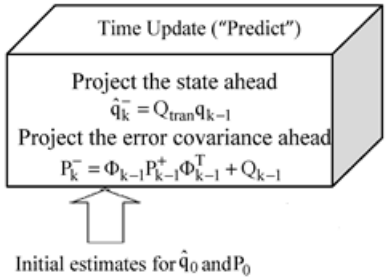
- BOOSTING
- MIL
- KCF
- TLD
- MEDIANFLOW
- GOTURN
- …

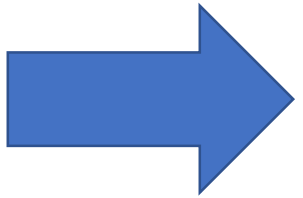


"The next step in SA formation involves a synthesis of disjointed Level 1 SA elements through the processes of pattern recognition, interpretation, and evaluation. Level 2 SA requires integrating this information to understand how it will impact upon the individual's goals and objectives. This includes developing a comprehensive picture of the world, or of that portion of the world of concern to the individual."

Projection

Projection of future status of the situation





Language

"The third and highest level of SA involves the ability to project the future actions of the elements in the environment. Level 3 SA is achieved through knowledge of the status and dynamics of the elements and comprehension of the situation (Levels 1 and 2 SA), and then extrapolating this information forward in time to determine how it will affect future states of the operational environment."

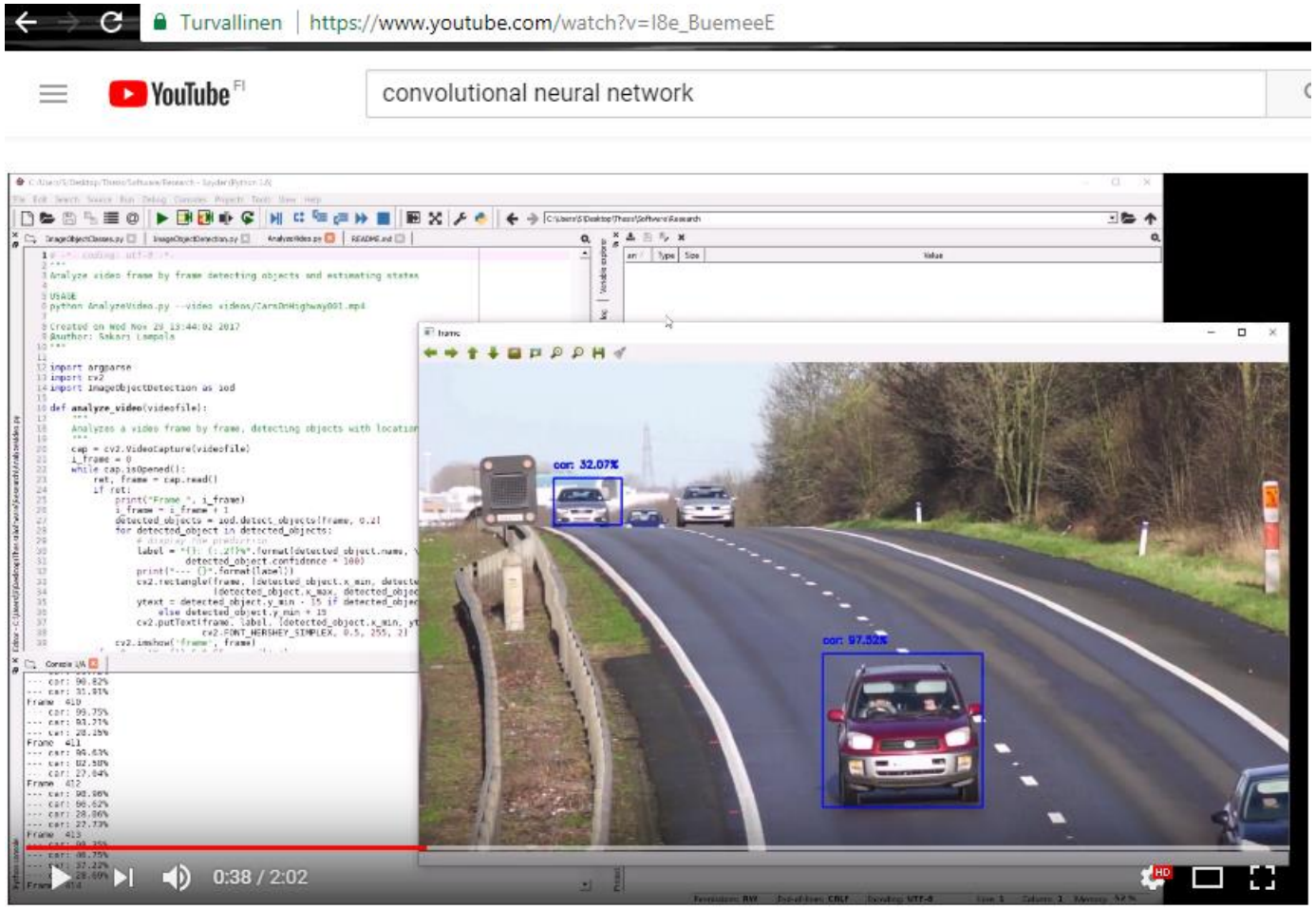
Work in Progress

Perception

“The first step in achieving SA is to perceive the status, attributes, and dynamics of relevant elements in the environment. Thus, Level 1 SA, the most basic level of SA, involves the processes of monitoring, cue detection, and simple recognition, which lead to an awareness of multiple situational elements (objects, events, people, systems, environmental factors) and their current states (locations, conditions, modes, actions).”

Perception

Object detection



Object Detection Using SSD and MobileNets

Perception

SSD + MobileNet

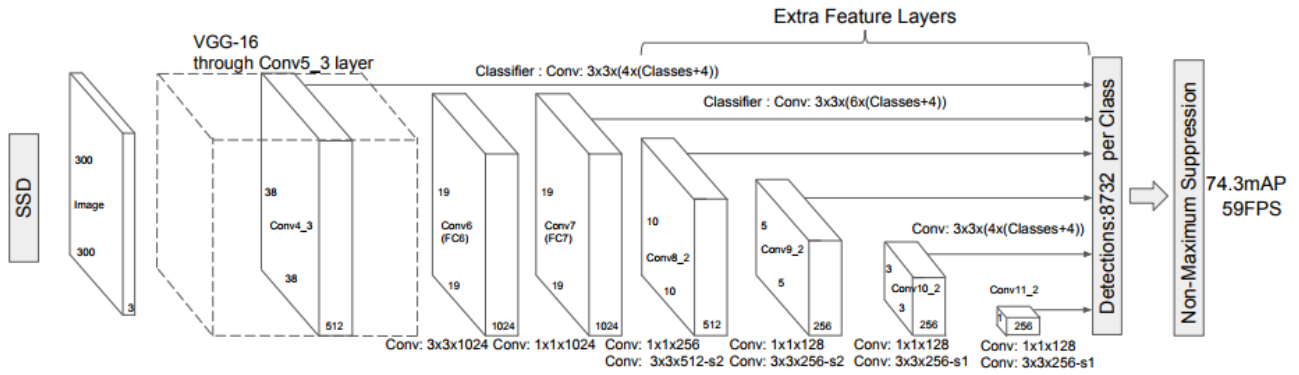


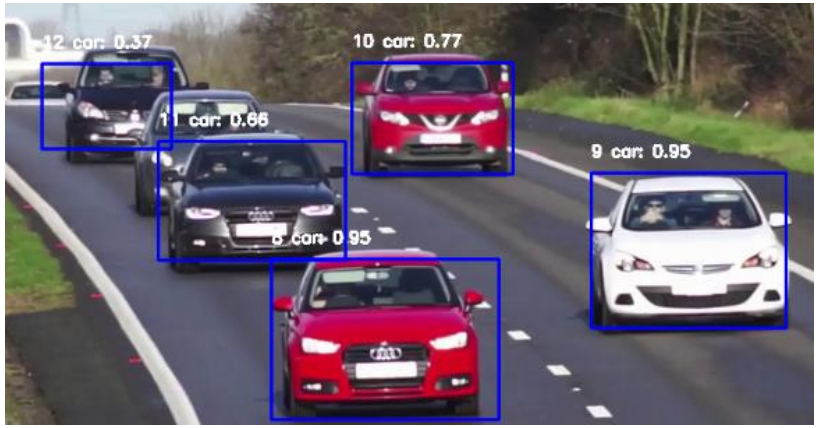
Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

```
CLASS_NAMES = ["background", "aeroplane", "bicycle", "bird", "boat",
               "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
               "dog", "horse", "motorbike", "person", "pottedplant", "sheep",
               "sofa", "train", "tvmonitor"]
```

Perception

Result



Detected object

- x_min
- x_max
- y_min
- y_max
- class
- confidence

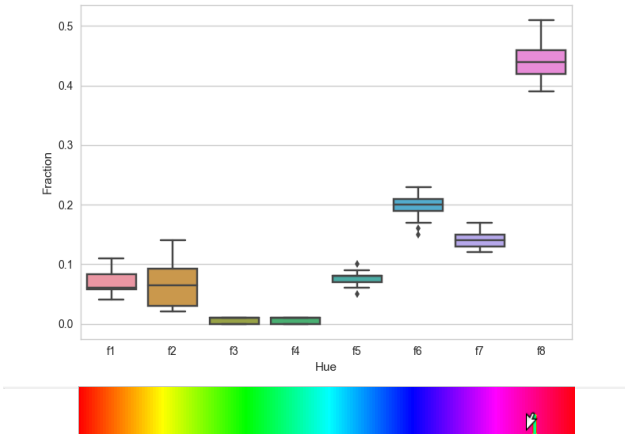
+

- appearance

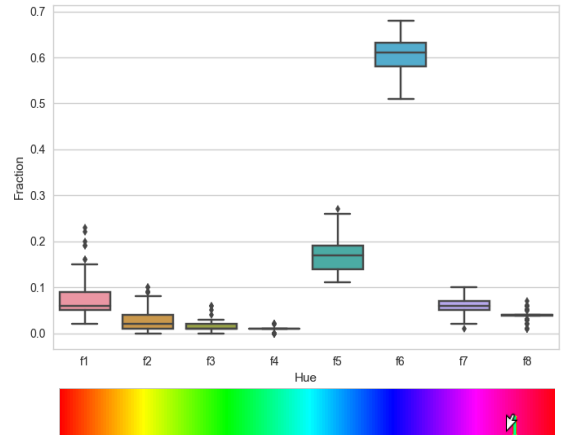
Perception

Appearance : hue histogram, 8 bins

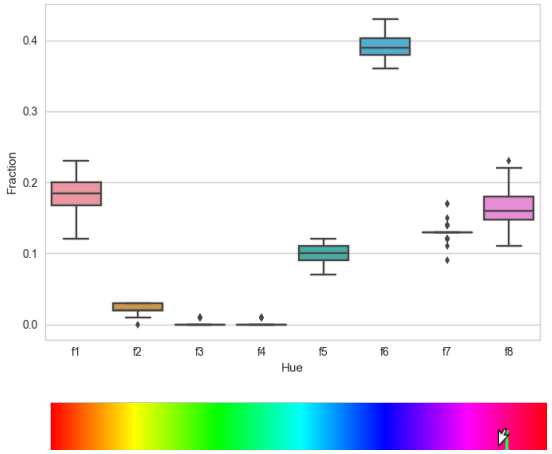
Red car



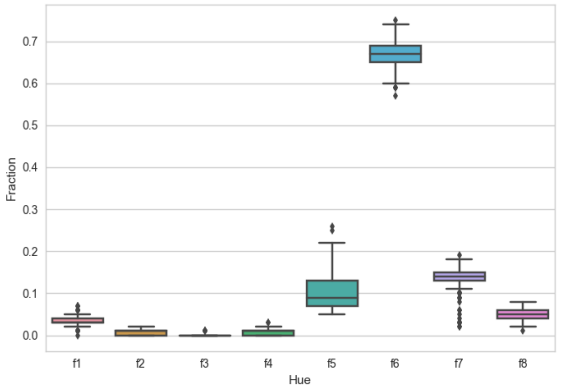
Blue car



White car



Black car

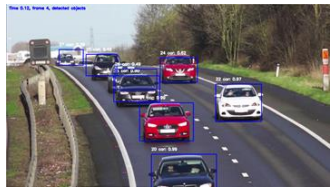


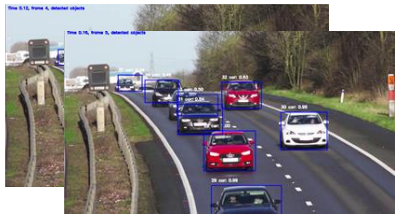
Comprehension

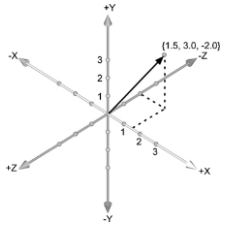
“The next step in SA formation involves a synthesis of disjointed Level 1 SA elements through the processes of pattern recognition, interpretation, and evaluation. Level 2 SA requires integrating this information to understand how it will impact upon the individual's goals and objectives. This includes developing a comprehensive picture of the world, or of that portion of the world of concern to the individual.”

Comprehension

Entity definitions







Detected Object → Image Object → World Object

Detected object:
Output from an object detector, no persistency.

Image object:
Persistent object created and updated by detected objects.

World object:
Object living in a 3d world with state: including location, velocity and acceleration. Updated by projected image object.

Comprehension

Image object identity

Perception

Image object

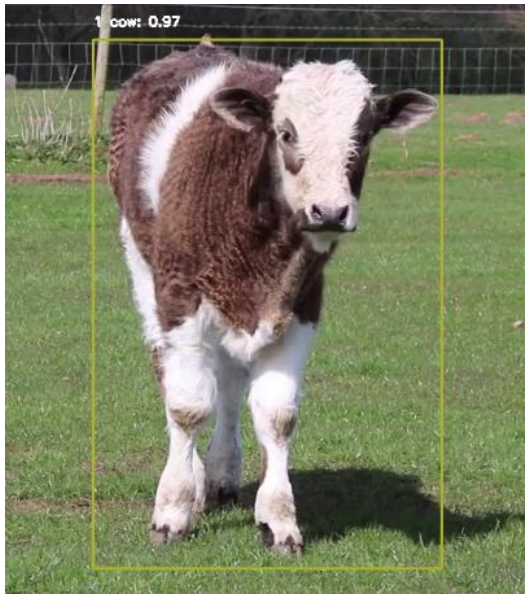
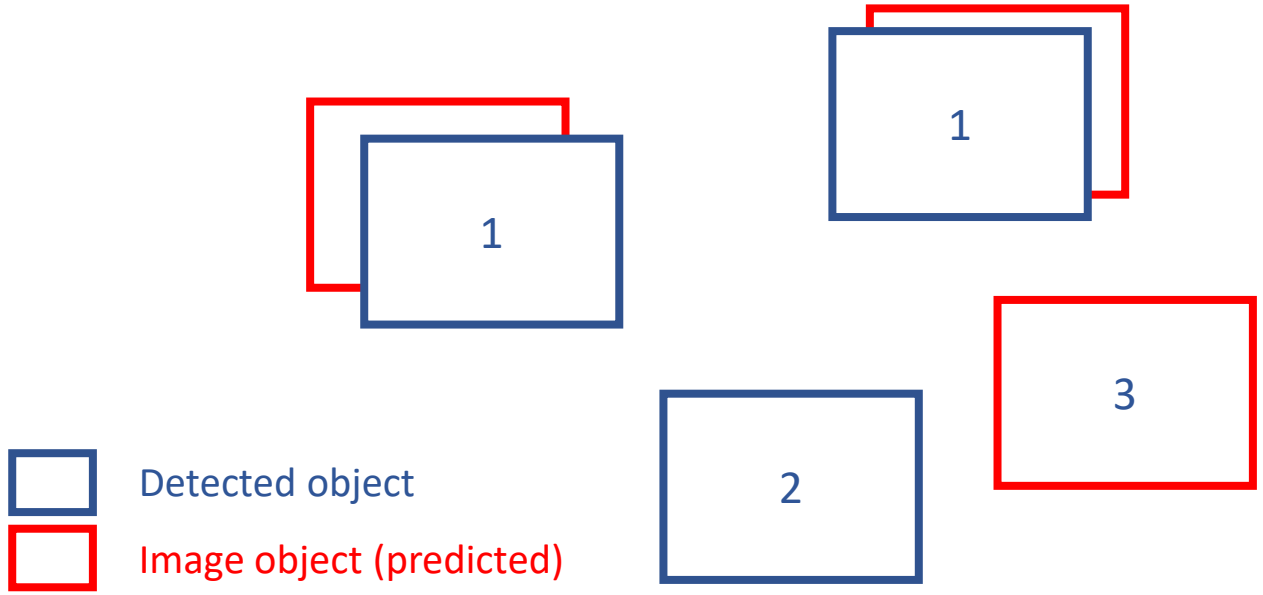


Image object

- id
- status
- x_min
- x_max
- y_min
- y_max
- vx_min
- vx_max
- vy_min
- vy_max
- class
- confidence
- appearance

Comprehension

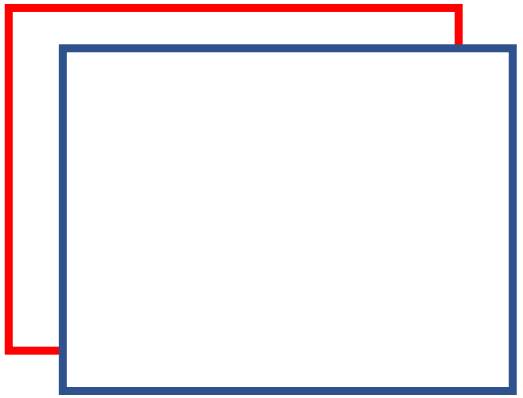
Matching detected and image objects



1. Detected object and image object are ”near” each other. Image object updated according to the detected object.
2. Detected object without any nearby image objects. A new image object is created according to the detected object.
3. Image object without any nearby detected objects. The image object status is changed to ’Hidden’. Movement prediction continues until out of frame.

Comprehension

Distance metrics



Detected object

Image object (predicted)

```
def distance(self, image_object):
    """
    Calculates distance (4 corners) to a predicted image object
    """
    dx_min = abs(self.x_min - image_object.x_min)
    dx_max = abs(self.x_max - image_object.x_max)
    dy_min = abs(self.y_min - image_object.y_min)
    dy_max = abs(self.y_max - image_object.y_max)

    horizontal_size = max(image_object.x_max - image_object.x_min, 1)
    vertical_size = max(image_object.y_max - image_object.y_min, 1)

    dx_min /= horizontal_size
    dx_max /= horizontal_size
    dy_min /= vertical_size
    dy_max /= vertical_size

    return (dx_min + dx_max + dy_min + dy_max) / 4.0 # Average of 4 corners
```

Comprehension

Probabilistic method

Detected object

- x_min
- x_max
- y_min
- y_max
- class
- confidence

+

- appearance

Image object 1

- id
- status
- x_min
- x_max
- y_min
- y_max
- vx_min
- vx_max
- vy_min
- vy_max
- class
- confidence
- appearance

Image object 2

- id
- status
- x_min
- x_max
- y_min
- y_max
- vx_min
- vx_max
- vy_min
- vy_max
- class
- confidence
- appearance

$P(\text{Image object i} \mid \text{Detected object}) = N(\mu_{io} - \mu_{do}, V_f)$

How to define V_f?

Feature vector

- x_min
- x_max
- y_min
- y_max
- class ??
- confidence+
- appearance

Comprehension

Matching algorithm

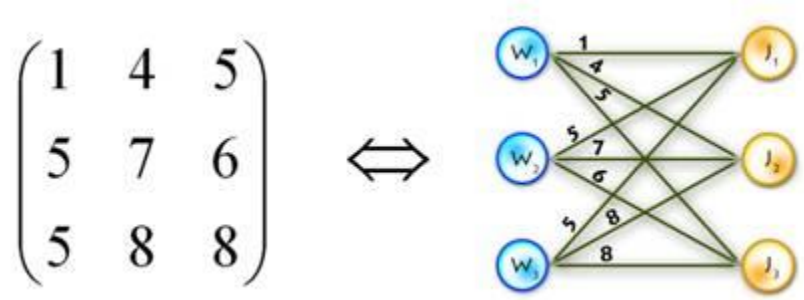
Hungarian algorithm

From Wikipedia, the free encyclopedia

The **Hungarian method** is a combinatorial optimization algorithm that solves the assignment problem in polynomial time and which anticipated later primal-dual methods. It was developed and published in 1955 by Harold Kuhn, who gave the name "Hungarian method" because the algorithm was largely based on the earlier works of two Hungarian mathematicians: Dénes Kőnig and Jenő Egerváry.[1][2]

James Munkres reviewed the algorithm in 1957 and observed that it is (strongly) polynomial.[3] Since then the algorithm has been known also as the **Kuhn–Munkres algorithm** or **Munkres assignment algorithm**. The time complexity of the original algorithm was $O(n^4)$, however Edmonds and Karp, and independently Tomizawa noticed that it can be modified to achieve an $O(n^3)$ running time. Ford and Fulkerson extended the method to general transportation problems.[*citation needed*] In 2006, it was discovered that Carl Gustav Jacobi had solved the assignment problem in the 19th century, and the solution had been published posthumously in 1890 in Latin.[4]

$$\begin{pmatrix} 1 & 4 & 5 \\ 5 & 7 & 6 \\ 5 & 8 & 8 \end{pmatrix}$$



scipy.optimize.linear_sum_assignment

scipy.optimize.linear_sum_assignment(*cost_matrix*) [source]

Solve the linear sum assignment problem.

The linear sum assignment problem is also known as minimum weight matching in bipartite graphs. A problem instance is described by a matrix C, where each C[i,j] is the cost of matching vertex i of the first partite set (a "worker") and vertex j of the second set (a "job"). The goal is to find a complete assignment of workers to jobs of minimal cost.

Formally, let X be a boolean matrix where $X[i, j] = 1$ iff row i is assigned to column j. Then the optimal assignment has cost

$$\min \sum_i \sum_j C_{i,j} X_{i,j}$$

Comprehension

Matching algorithm

```
-------------------------------------------
Time 0.04, frame 2
Detected objects (6):
---7 car   1.00  573  811  572  721 0.02 0.00 0.01 0.03 0.19 0.65 0.05 0.03
---8 car   0.95  521  730  384  531 0.02 0.00 0.00 0.01 0.06 0.31 0.11 0.49
---9 car   0.95  815  994  305  447 0.16 0.03 0.01 0.00 0.08 0.37 0.18 0.17
---10 car   0.77  595  743  204  306 0.08 0.02 0.00 0.00 0.06 0.21 0.15 0.47
---11 car   0.66  417  589  276  385 0.01 0.00 0.00 0.00 0.08 0.59 0.23 0.09
---12 car   0.37  310  429  205  283 0.05 0.00 0.00 0.00 0.15 0.58 0.16 0.05
Image objects (6), predicted new locations:
---1 car   1.00  569.00  811.00  573.00  721.00 0.02 0.01 0.01 0.03 0.20 0.64 0.06
---2 car   0.96  804.00  986.00  298.00  440.00 0.18 0.03 0.00 0.01 0.11 0.35 0.13
---3 car   0.94  520.00  724.00  379.00  522.00 0.02 0.00 0.00 0.02 0.07 0.29 0.09
---4 car   0.78  593.00  741.00  205.00  304.00 0.12 0.03 0.00 0.00 0.06 0.20 0.11
---5 car   0.64  415.00  585.00  267.00  382.00 0.02 0.00 0.00 0.00 0.10 0.56 0.20
---6 car   0.39  311.00  429.00  200.00  286.00 0.05 0.00 0.00 0.01 0.15 0.57 0.15
Cost matrix for Hungarian algorithm:
   0.01    1.53    0.86    2.13    1.96    3.71
   0.77    1.05    0.03    1.17    0.95    2.33
   1.36    0.05    0.95    1.42    1.41    3.04
   1.42    1.02    0.80    0.01    0.80    1.34
   1.46    1.21    0.71    0.94    0.03    1.07
   2.02    1.88    1.34    1.06    0.73    0.03
Optimal assignment:
Detected object 7 matched to image object 1
Existing image object 1 updated:
---predicted: car   1.00 569.00 811.00 573.00 721.00 0.02 0.01 0.01 0.03 0.20 0.64 0.06 0.04
---measured:  car   1.00 573.00 811.00 572.00 721.00 0.02 0.00 0.01 0.03 0.19 0.65 0.05 0.03 0.01
---corrected: car   1.00 573.00 811.00 572.00 721.00 0.02 0.01 0.01 0.03 0.20 0.64 0.06 0.04
Detected object 8 matched to image object 3
Existing image object 3 updated:
---predicted: car   0.94 520.00 724.00 379.00 522.00 0.02 0.00 0.00 0.02 0.07 0.29 0.09 0.50
---measured:  car   0.95 521.00 730.00 384.00 531.00 0.02 0.00 0.00 0.01 0.06 0.31 0.11 0.49 0.03
---corrected: car   0.95 521.00 730.00 384.00 531.00 0.02 0.00 0.00 0.02 0.07 0.29 0.09 0.50
Detected object 9 matched to image object 2
Existing image object 2 updated:
---predicted: car   0.96 804.00 986.00 298.00 440.00 0.18 0.03 0.00 0.01 0.11 0.35 0.13 0.19
```

Comprehension

Matching algorithm

Open question: Role of classical algorithms?

- MeanShift
- CamShift
- Optical flow (Lucas-Kanade)
- Background Subtraction

Possible added value:

- Refined object region
- Support for occasionally missing bounding boxes

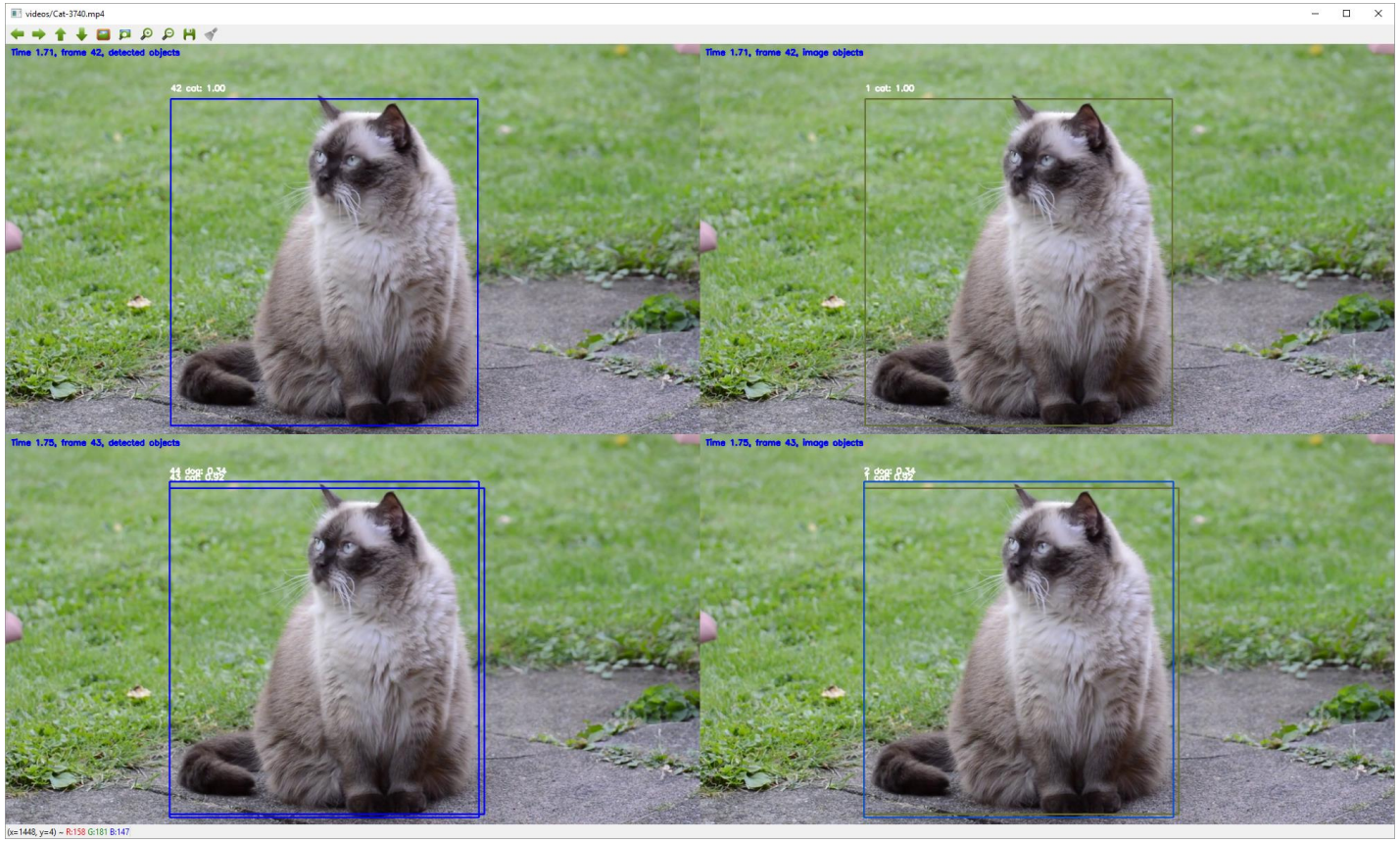
Comprehension

Special cases / Vanishing detected objects

Solution: Image objects have statuses. Their location is predicted according to the previous location and velocity. When no matching detected objects exist, image objects are not immediately removed, but their status is changed (’detected’-> ’hidden’). Prediction continues in following frames.

Comprehension

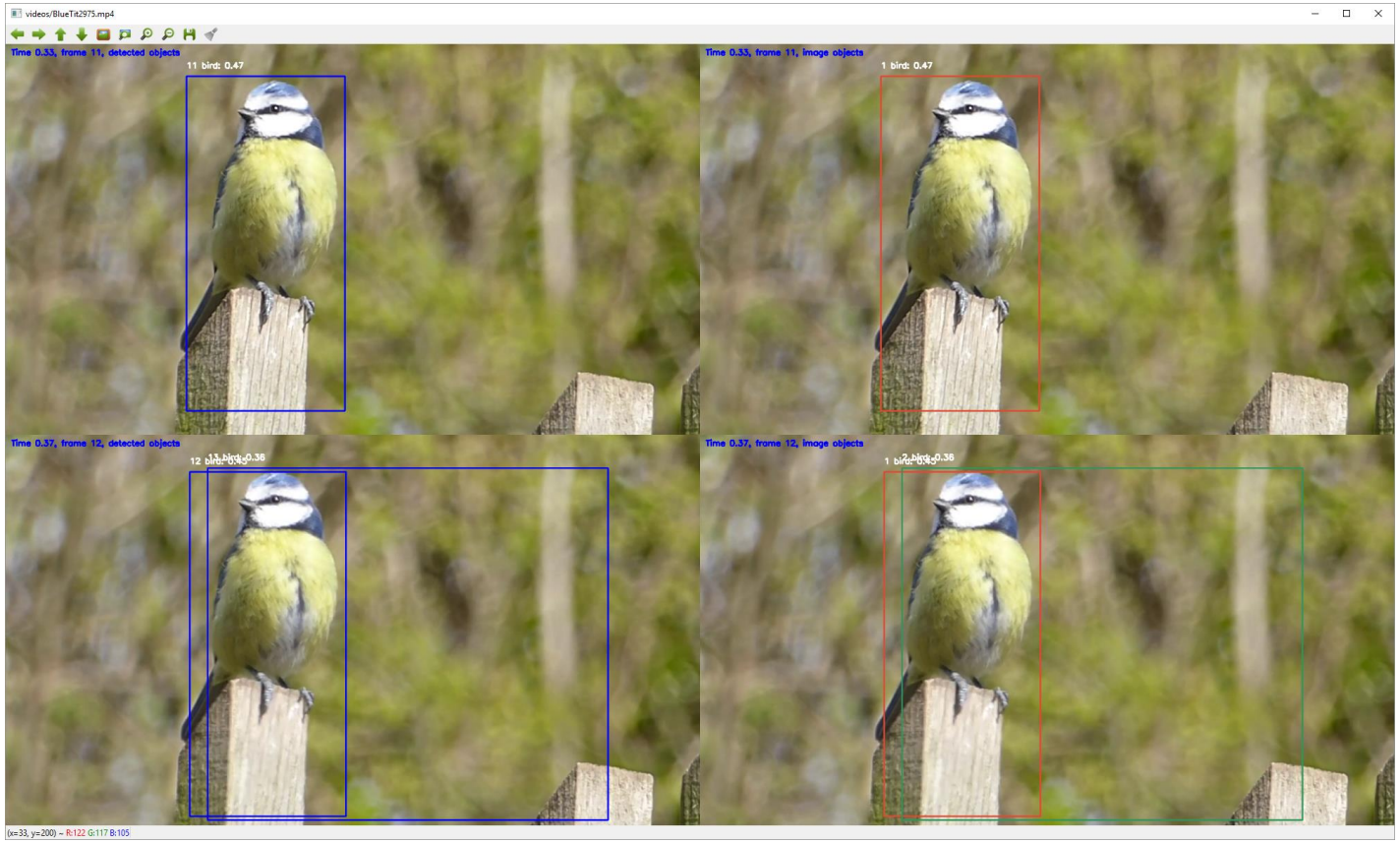
Special cases / Multiple classifications



Solution: The classification with highest confidence is used if multiple detected objects correspond to an image object and no other image object is nearby. The information is available in the Hungarian algorithm cost matrix.

Comprehension

Special cases / Multiple bounding boxes (a)

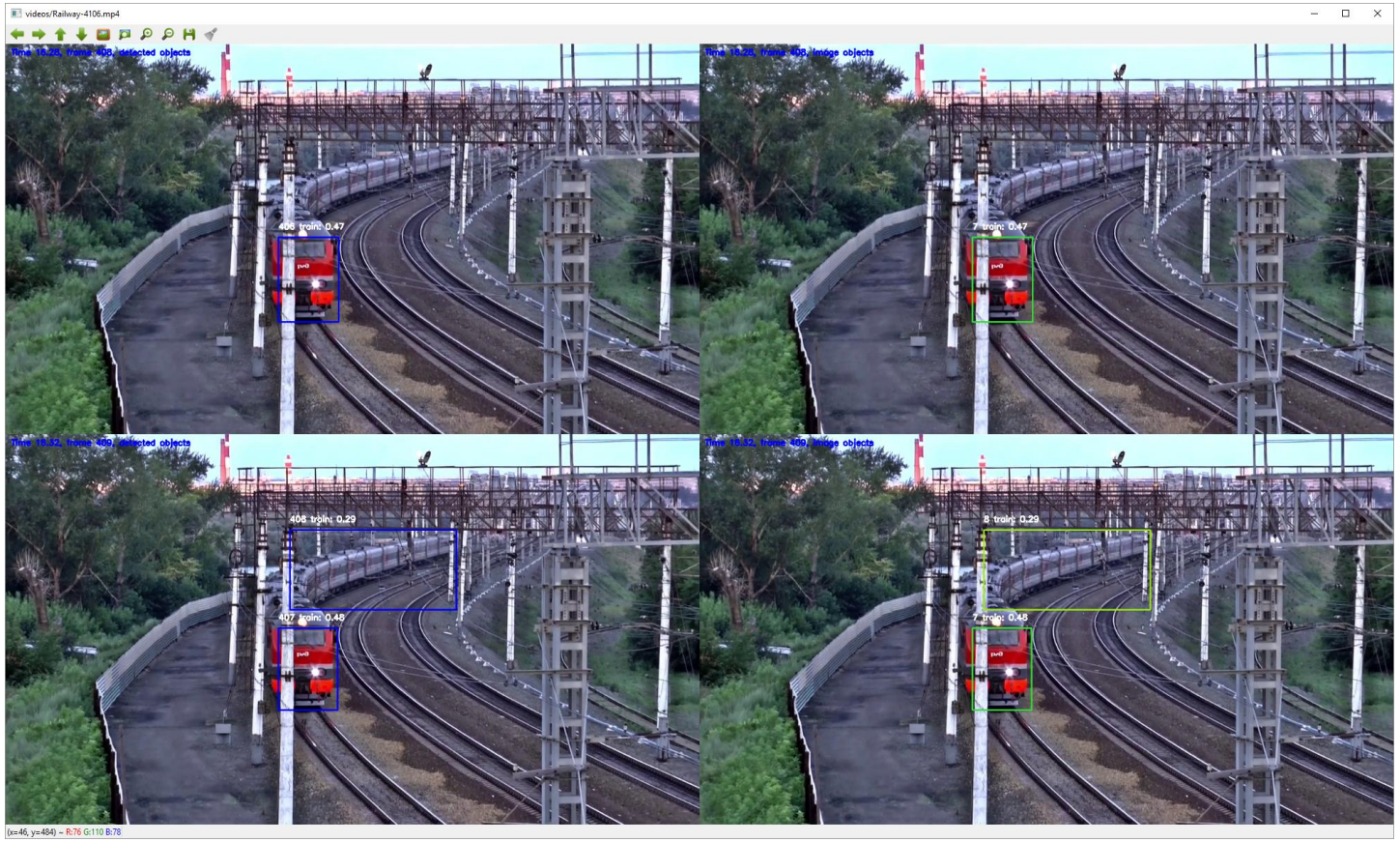


Solution: ?

Open question: What to do?

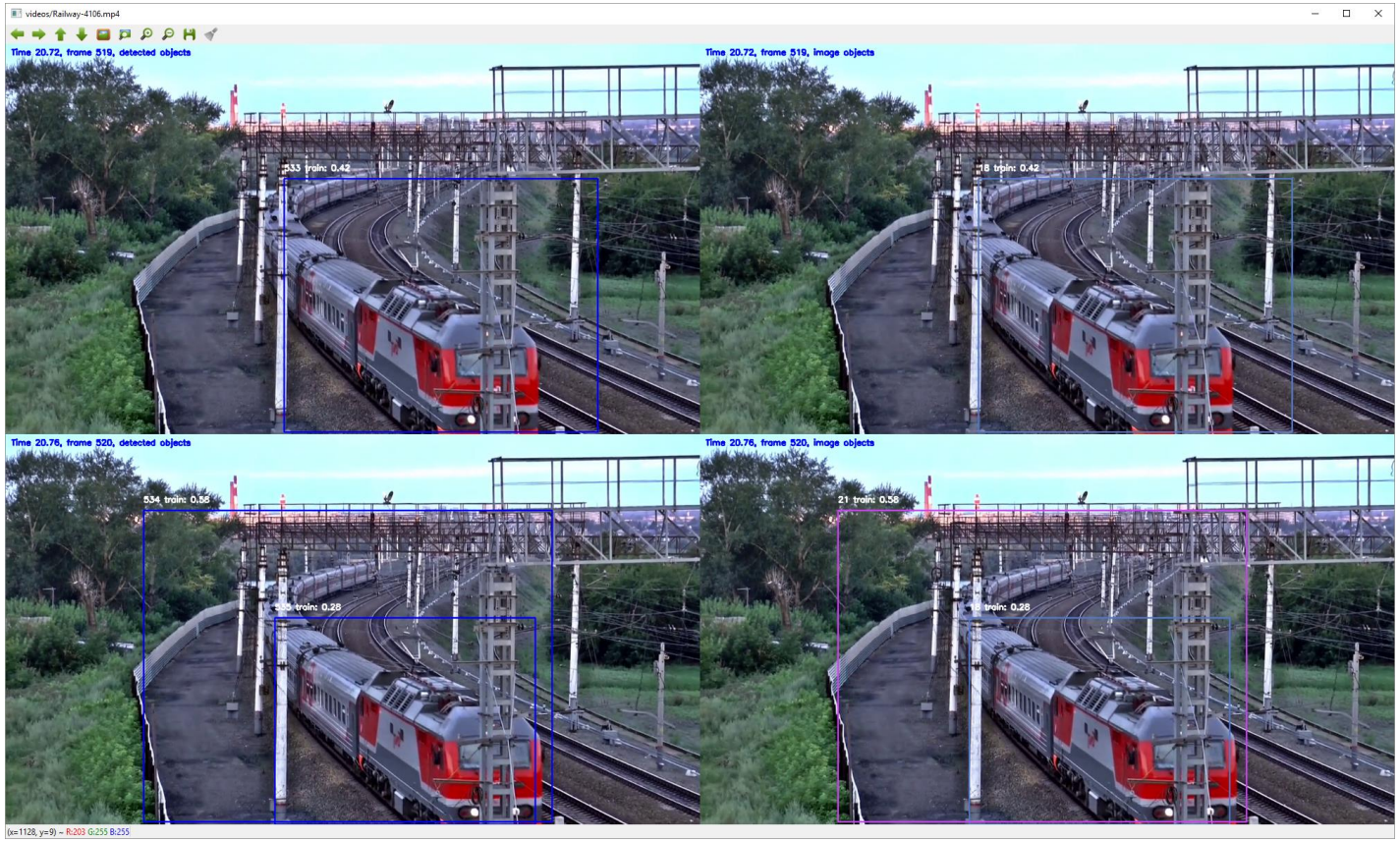
Comprehension

Special cases / Multiple bounding boxes (b)



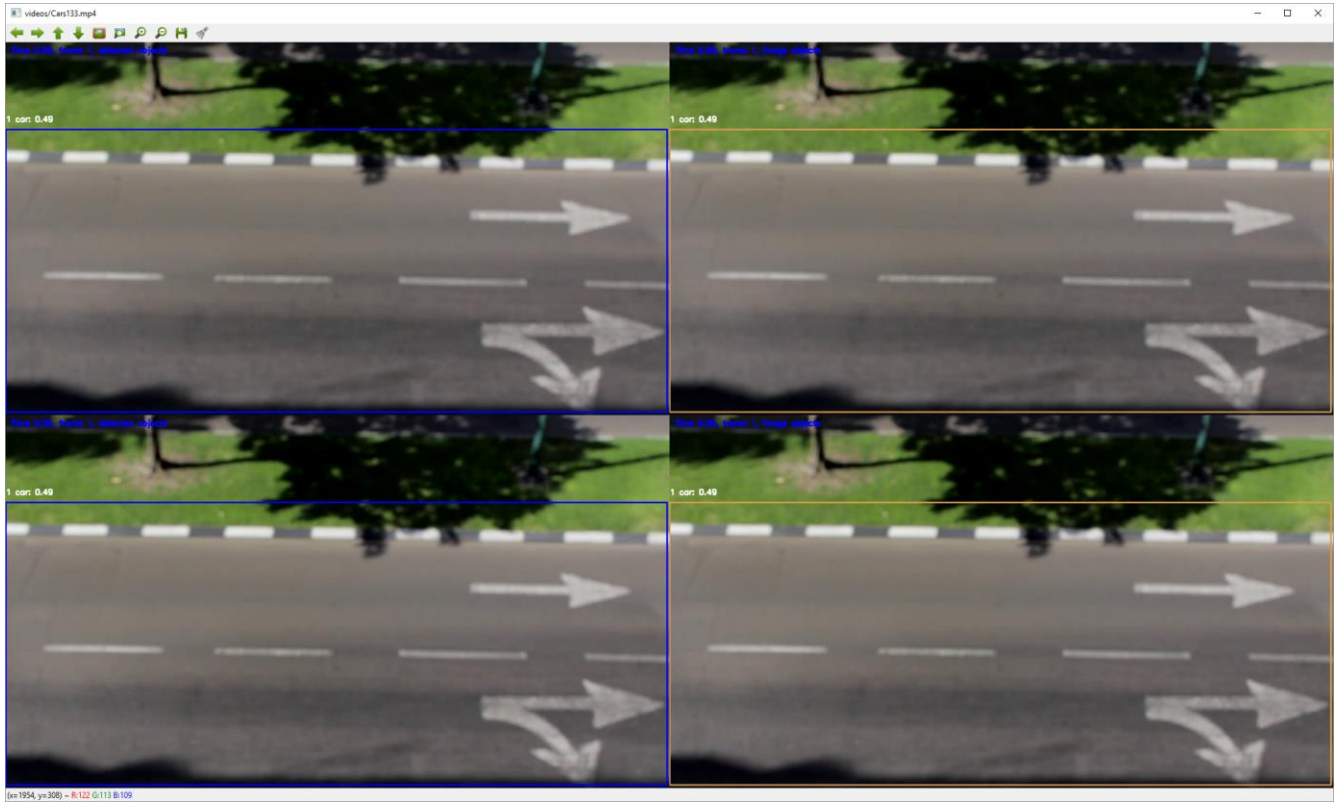
Comprehension

Special cases / Multiple bounding boxes (c)



Comprehension

Special cases / Typical background elements are messed up with object in the classification. (a)



Solution: Classification confidence must be high enough for creating a new image object. Probabilistic estimate based on distance and a priori size.

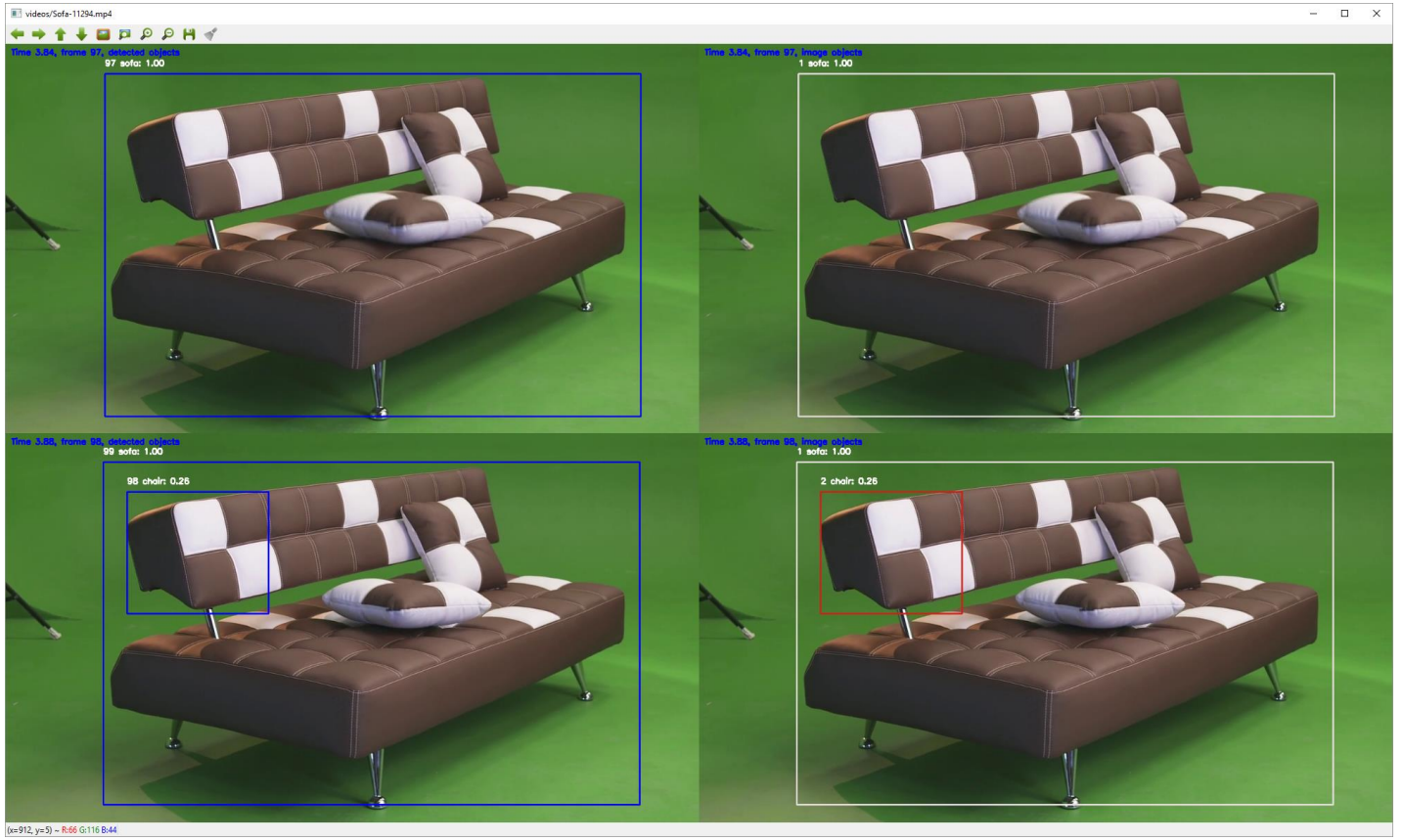
Open question: How to determine the appropriate minimum confidence level?

Comprehension

Special cases / Typical background elements are messed up with object in the classification. (b)

Comprehension

Special cases / False detections inside other objects

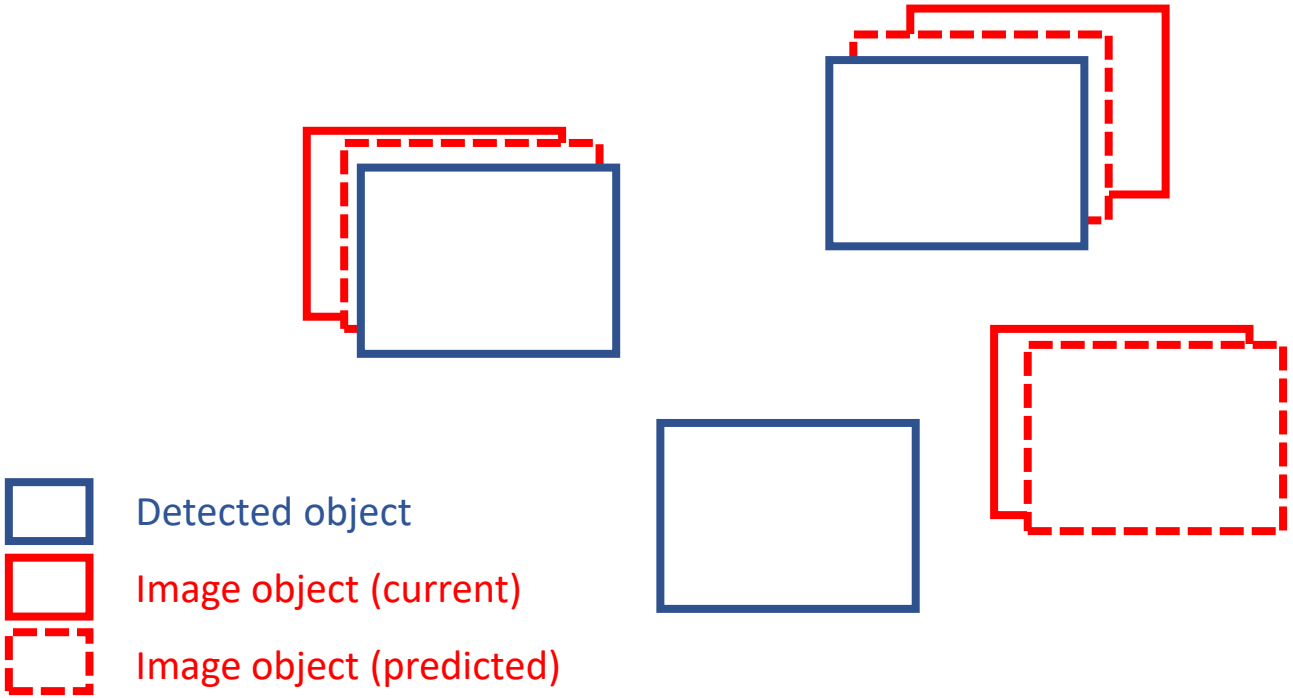


Solution: ?

Open question: What to do?

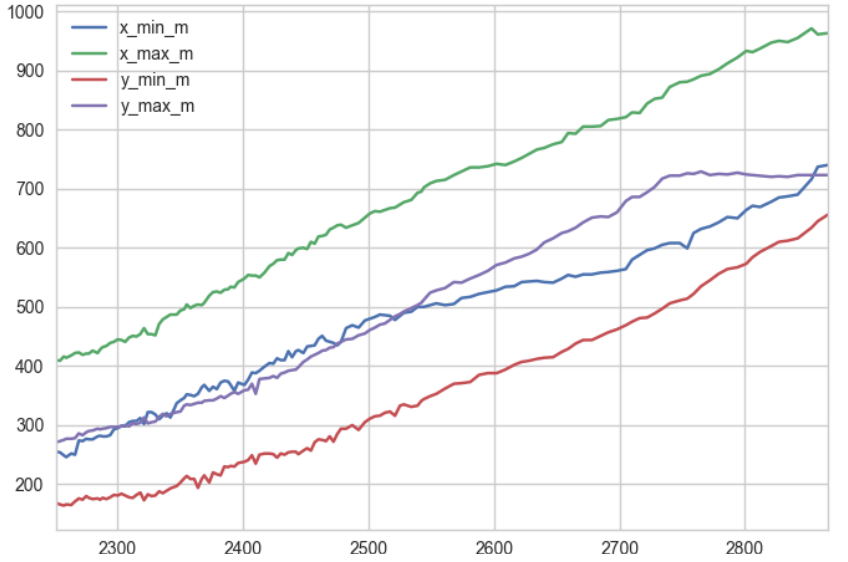
Comprehension

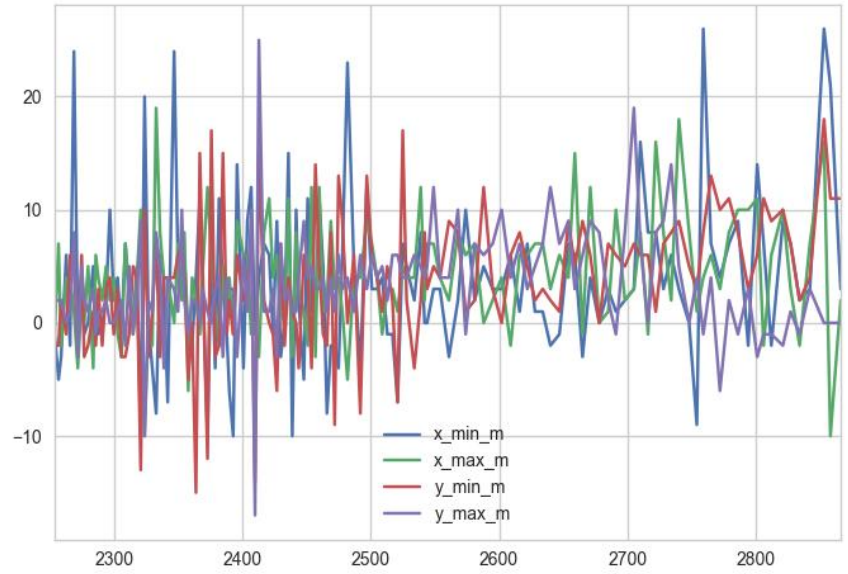
Image object location and velocity prediction



Comprehension

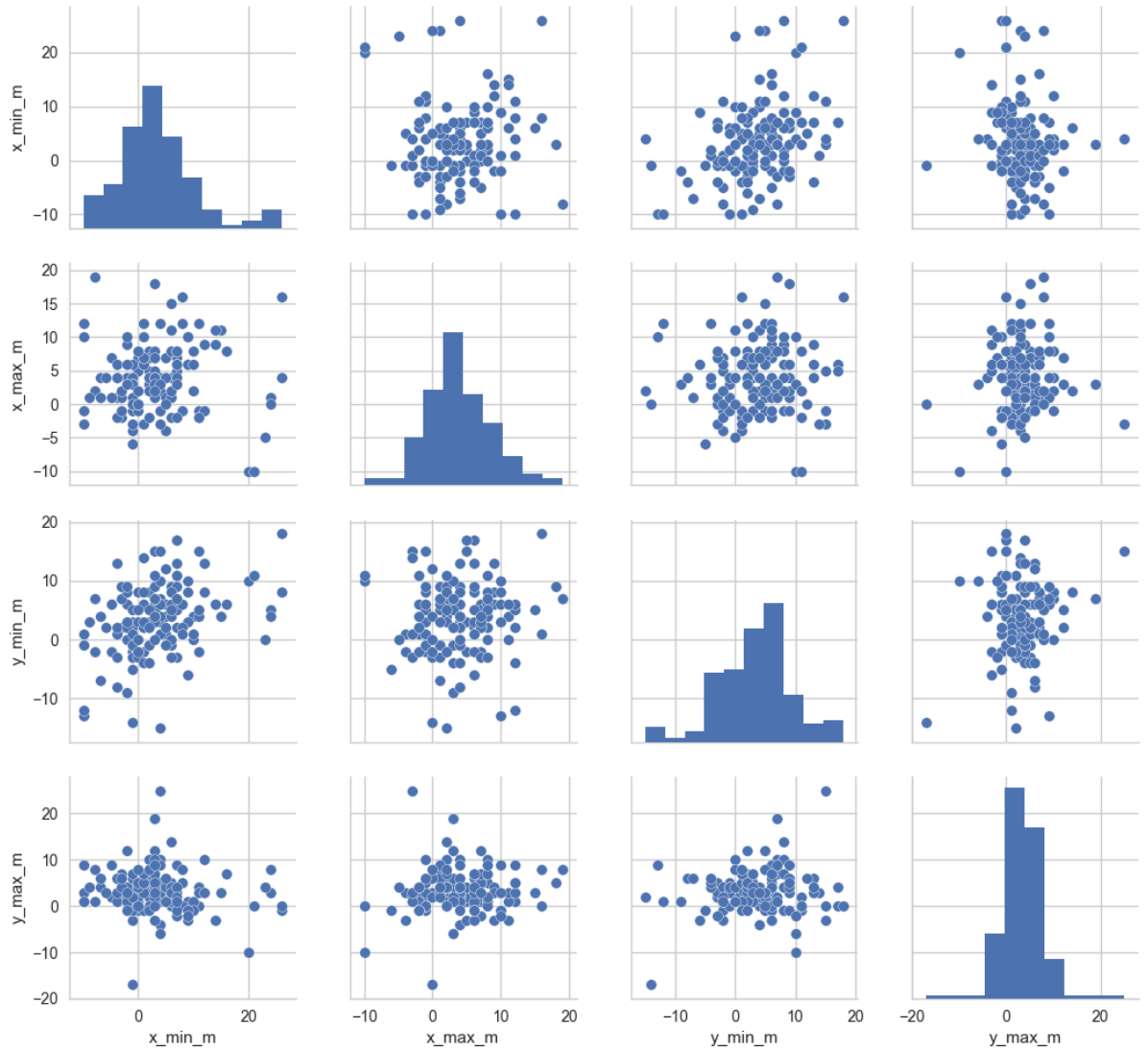
Image object location and velocity prediction





Comprehension

Image object location and velocity prediction



Comprehension

Image object location and velocity prediction

Bounding box corner location

State vector s:

$$s = \begin{bmatrix} l \\ v \end{bmatrix}$$

where

l = location coordinate (x_{min}, x_{max}, y_{min}, y_{max}) of the bounding box corner in the image
v = velocity (vx_{min}, vx_{max}, vy_{min}, vy_{max}) of the bounding box corner in the image

State equation in differential form:

$$\frac{ds(t)}{dt} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} * s(t) + \epsilon(t) = A_1 * s$$

State equation in difference form:

$$s(k+1) = (I + \Delta * A_1) * s(k) + \epsilon(k)$$

$$= \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} * s(k) + \varepsilon(k) = A * s(k) + \varepsilon(k)$$

where Δ is the time increment and ε Gaussian noise with covariance R.

Measurement equation

$$z(k) = [1 \quad 0] * s(k) + \delta(k) = C * s(k) + \delta(k)$$

Where δ is Gaussian noise with covariance matrix Q.

Comprehension

Image object location and velocity prediction

Kalman filter initialization:

$$\mu(0) = \begin{bmatrix} l(0) \\ 0 \end{bmatrix}$$

where l(0) is the first location measurement.

$$\Sigma(0) = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix}$$

where α, β and γ are believed variances of location and velocity.

$$R = \begin{bmatrix} r_1 & 0 \\ 0 & r_2 \end{bmatrix}$$

where r_1, r_2 and r_3 are believed state equation variances of location and velocity.

$$Q = [q]$$

Where q is the believed measurement variance, for example 1.0.

Kalman filter update:

$$\mu_1(k) = A * \mu(k-1)$$

$$\Sigma_1(k) = A * \Sigma(k-1) * A^T + R$$

$$K(k) = \Sigma_1(k) * C^T * (C * \Sigma_1(k) * C^T + Q)^{-1}$$

$$\mu(k) = \mu_1(k) + K(k) * (z(k) - C * \mu_1(k))$$

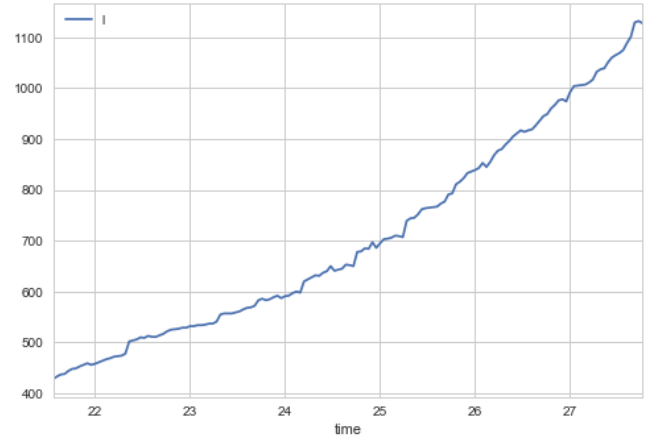
$$\Sigma(k) = \left(I - K(k) * C\right) * \Sigma_1(k)$$

Comprehension

Image object location and velocity prediction

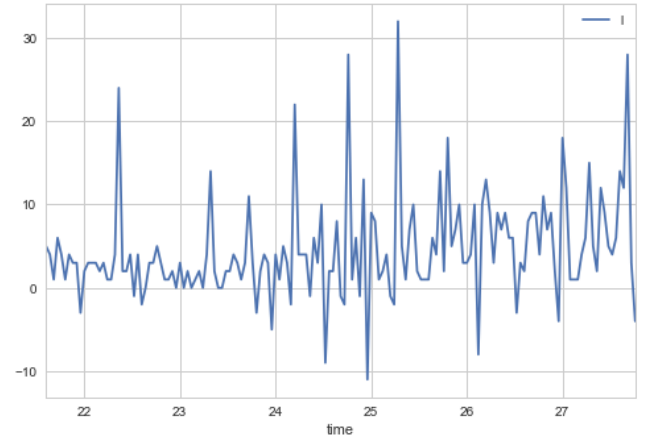
```
In [39]: x_min.plot()
```

```
Out[39]: <matplotlib.axes._subplots.AxesSubplot at 0x20472a436d8>
```




```
In [40]: x_min_diff = x_min.diff(1)
         x_min_diff = x_min_diff.dropna()
         x_min_diff.plot()
```

```
Out[40]: <matplotlib.axes._subplots.AxesSubplot at 0x20472a5ee80>
```

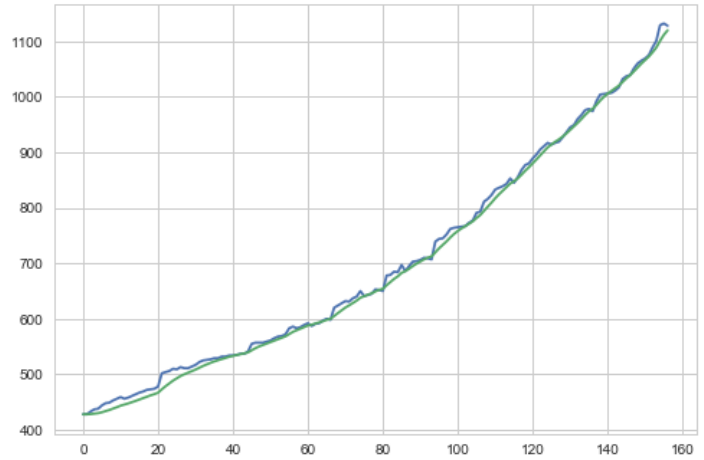


Comprehension

Image object location and velocity prediction

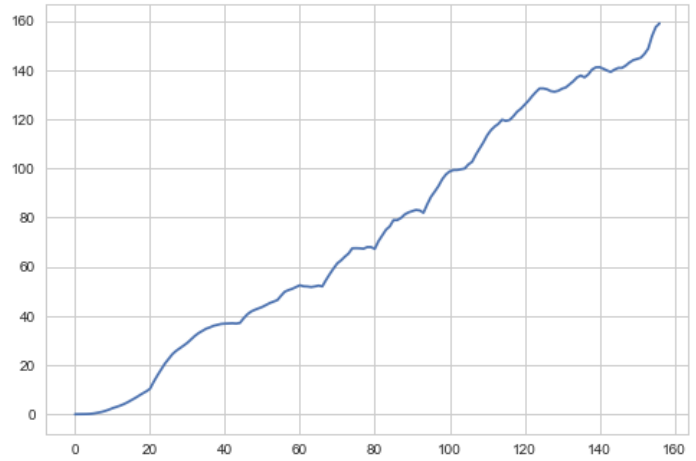
```
In [50]: plt.plot(x_min_m)
         plt.plot(x_min_f)
```

```
Out[50]: [<matplotlib.lines.Line2D at 0x204728d0f98>]
```




```
In [51]: plt.plot(vx_min_f)
```

```
Out[51]: [<matplotlib.lines.Line2D at 0x20472dfb978>]
```

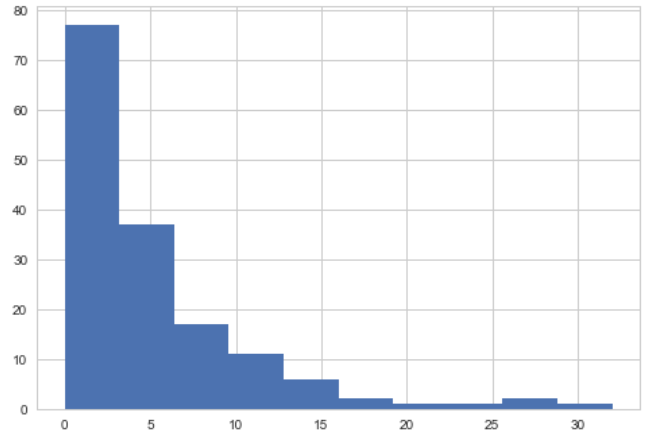


Comprehension

Image object location and velocity prediction

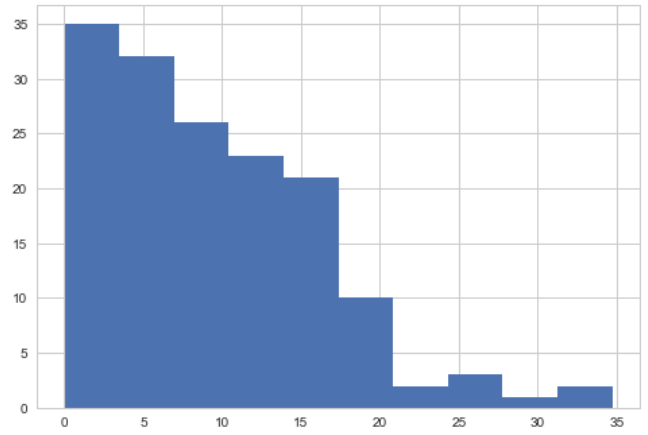
```
In [57]: plt.hist(x_min_error_prev)
         print(np.mean(x_min_error_prev))
```

```
5.32903225806
```




```
In [58]: plt.hist(x_min_error_filter)
         print(np.mean(x_min_error_filter))
```

```
9.44186196322
```



Comprehension

Image object location and velocity prediction

Open questions:

- How to set covariances and initialize sigma?
- How to predict when using previous measurement gives smaller average error?

(Notebook demo)

Next Steps

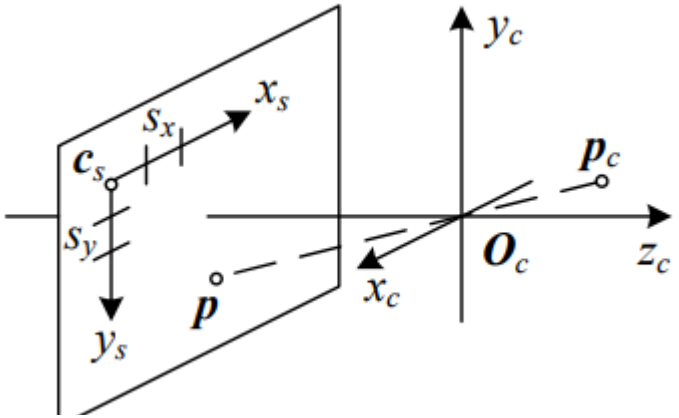
Next steps

Comprehension:

1. Closing the open questions
2. 2d -> 3d transformation
3. World object state estimation

Next steps

2d -> 3d transformation



$$\boldsymbol{K} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}.$$

$$\tilde{\boldsymbol{x}}_s = \alpha \boldsymbol{M}_s^{-1} \boldsymbol{p}_c = \boldsymbol{K} \boldsymbol{p}_c. \tag{2.54}$$

The 3×3 matrix $\boldsymbol{K}$ is called the *calibration matrix* and describes the camera *intrinsics* (as opposed to the camera's orientation in space, which are called the *extrinsics*).

$$\tilde{\boldsymbol{x}}_s = \boldsymbol{K} \begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix} \boldsymbol{p}_w = \boldsymbol{P} \boldsymbol{p}_w,$$

Open question: How to create the reverse transformation?

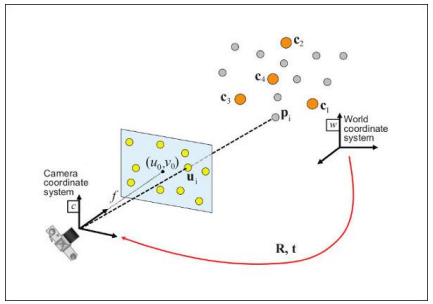
Next steps

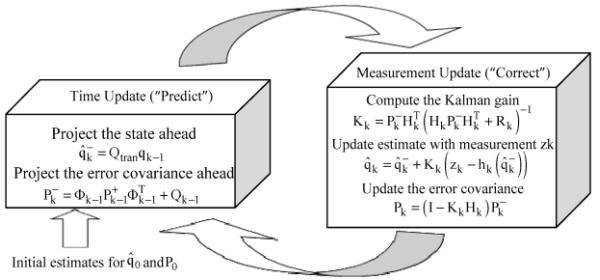
2d -> 3d transformation (a priori size for distance estimation)

```
199 class Aeroplane(ImageObject):
200     """
201     Aeroplane - derived class
202     """
203     name = 'aeroplane'
204     class_type = 1
205     height_min, height_mean, height_max = 1.5, 2.0, 6.0
206     width_min, width_mean, width_max = 2.0, 20.0, 80.0
207     length_min, length_mean, length_max = 3.0, 20.0, 80.0
208     velocity_max, acceleration_max = 300.0, 10.0
209 
210 class Bicycle(ImageObject):
211     """
212     Bicycle - derived class
213     """
214     name = 'bicycle'
215     class_type = 2
216     height_min, height_mean, height_max = 0.6, 1.1, 1.4
217     width_min, width_mean, width_max = 0.4, 0.6, 0.8
218     length_min, length_mean, length_max = 1.2, 1.5, 1.7
219     velocity_max, acceleration_max = 13.0, 1.0
220 
221 class Bird(ImageObject):
222     """
223     Bird - derived class
224     """
225     name = 'bird'
226     class_type = 3
227     height_min, height_mean, height_max = 0.1, 0.25, 2.8
228     width_min, width_mean, width_max = 0.05, 0.5, 1.2
229     length_min, length_mean, length_max = 0.1, 0.2, 1.2
230     velocity_max, acceleration_max = 80.0, 100.0
231 
```

Next steps

World object state estimation





State vector s:

$$s = \begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \\ a_x \\ a_y \\ a_z \end{bmatrix}$$

where

(x, y, z) = location of the world object center point
(v_x , v_y , v_z) = velocity of the object
(a_x , a_y , a_z) = acceleration of the object

Next steps

World object state estimation

State equation in differential form:

$$\frac{ds(t)}{dt} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} * s(t) + \epsilon(t) = A_1 * s$$

State equation in difference form:

$$s(k+1) = (I + \Delta * A_1) * s(k) + \epsilon(k)$$

$$= \begin{bmatrix} 1 & \Delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} * s(k) + \varepsilon(k) = A * s(k) + \varepsilon(k)$$

where Δ is the time increment and ε Gaussian noise with covariance R.

Measurement equation

$$z(k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} * s(k) + \delta(k) = C * s(k) + \delta(k)$$

Where δ is Gaussian noise with covariance matrix Q.

Next steps

World object state estimation

Kalman filter initialization:

$$\mu(0) = \begin{bmatrix} x(0) \\ 0 \\ 0 \\ y(0) \\ 0 \\ 0 \\ z(0) \\ 0 \\ 0 \end{bmatrix}$$

where x(0), y(0), z(0) is the first location measurement.

$$\Sigma(0) = \begin{bmatrix} \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \beta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \alpha & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma \end{bmatrix}$$

where α, β and γ are believed variances of location, velocity and acceleration.

$$R = \begin{bmatrix} r_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & r_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & r_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & r_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & r_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & r_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & r_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r_3 \end{bmatrix}$$

where r_1, r_2 and r_3 are believed variances of location, velocity and acceleration.

$$Q = \begin{bmatrix} q & 0 & 0 \\ 0 & q & 0 \\ 0 & 0 & q \end{bmatrix}$$

Where q is the believed measurement variance.

Next steps

World object state estimation

Kalman filter update:

$$\mu_1(k) = A * \mu(k-1)$$

$$\Sigma_1(k) = A * \Sigma(k-1) * A^T + R$$

$$K(k) = \Sigma_1(k) * C^T (C * \Sigma_1(k) * C^T + Q)^{-1}$$

$$\mu(k) = \mu_1(k) + K(k) * (z(k) - C * \mu_1(k))$$

$$\Sigma(k) = (I - K(k) * C) * \Sigma_1(k)$$

Open questions:

1. Equations ok?
2. Initial values?
3. Different time step for prediction, not equal to inverse video frame rate?

To Be Discussed

Actions

- The role of actions, should it be considered?
- Navigation assistance?

Probability representation

- Probability there is something in the image
  - P(S)
- Probability of class given that there is something
  - P(C|S)
- Probability of location (or state) given the class
  - P(X|C,S)

What is the goal?

“Develop methods which can perceive the elements in the environment within a volume of time and space, comprehend their meaning and project their status in the near future. The methods will be tested on a system which will help visually impaired people to understand their environment better. The system will be demonstrated on existing hardware.”

Goodfellow, Bengio, Courville: “Deep Learning”

Sources of uncertainty

Ingredients:

- Classification belief
- Bounding box noise
- Pixel discretization
- Distance estimation

1. Inherent stochasticity in the system being modeled.
   - ?
2. Incomplete observability.
   - ?
3. Incomplete modeling.
   - ?

Goodfellow, Bengio, Courville: “Deep Learning”

Goal Fine Tuning

“Develop methods which can perceive the elements in the environment within a volume of time and space, comprehend their meaning and project their status in the near future. The methods will be tested on a system which will help visually impaired people to understand their environment better. The system will be demonstrated on existing hardware.”

What is the goal in practice?
What is the output of the system?

Goodfellow, Bengio, Courville: “Deep Learning”

Thank you!

lampola@student.tut.fi
https://github.com/SakariLampola/Thesis