Image-Based Situation Awareness Audit 1.3.2018

Sakari Lampola

Previous Audit 11.1.2018

Previous Audit

Open questions:

- Role of classical object tracking alrorithms?
- What to do with multiple bounding boxes around one object?
- Appropriate minimum confidence level?
- What to do with false detections inside other objects?
- What to do with false detections from the background?
- How to set Kalman filter parameters for image object filtering?
- Hungarian algorithms special case for hidden objects

To do:

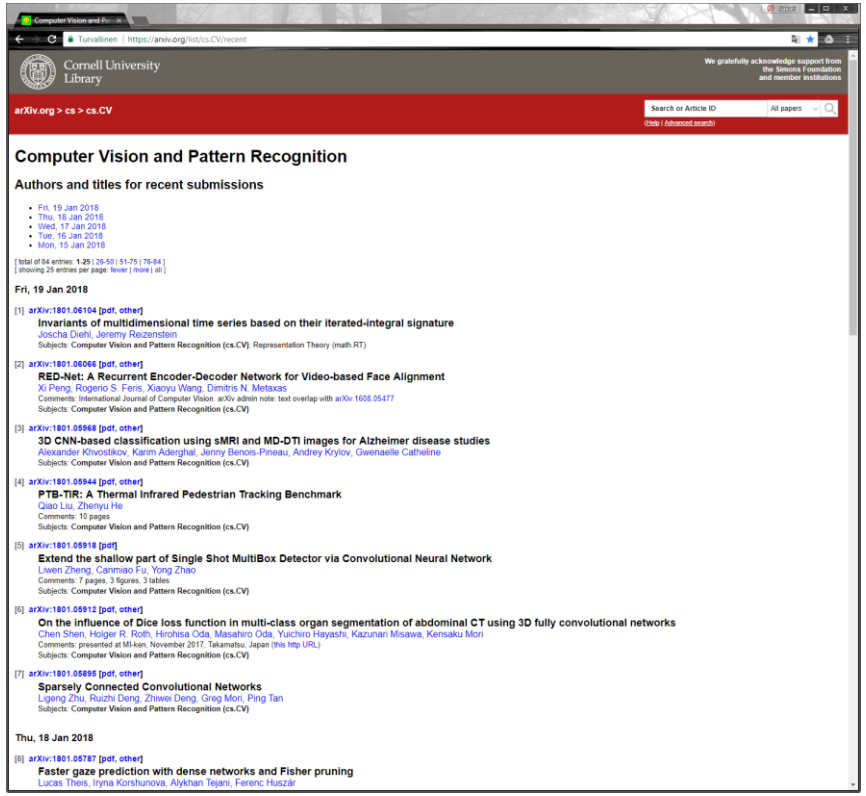
- Close open questions
- Image object status
- Image object velocity estimation
- Probabilistic approach for matching detected and image objects
- 2d -> 3d transformation
- World object state estimation

Other:

- Semantic segmentation
- Organisations to follow: ICCV, ICRA, NIPS, IROS, arXiv
- Camera motion (yaw, pitch)
- Grid or continuos presentation?
- Class specific attributes
- Object history

Work Done

Method Follow-Up



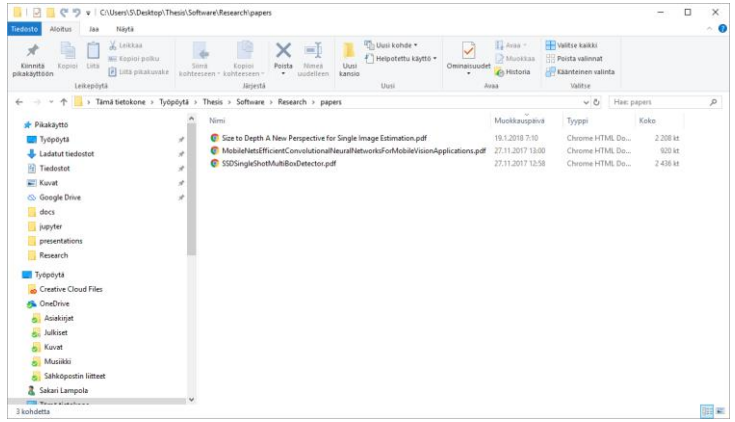


Image Object Velocity Estimation

Image object velocity is necessary for:

- predicting image object locations when matching new measurements
- identifying image objects
- predicting image object locations for hidden objects

Image object

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

Estimation algorithm

Image Object Kalman Filtering

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.

Kalman filter initialization:

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

where l(0) is the first location measurement.

$$\Sigma(0) = \begin{bmatrix} 10.0 & 0 \\ 0 & 10000.0 \end{bmatrix}$$

where 10.0 and 10000.0 are believed initial error variances of location and velocity.

$$R = \begin{bmatrix} 1.0 & 0 \\ 0 & 1.0 \end{bmatrix}$$

where diagonal elements are believed state equation variances of location and velocity.

$$Q = [10.0]$$

Where 10.0 is the believed measurement variance.

Kalman filter update:

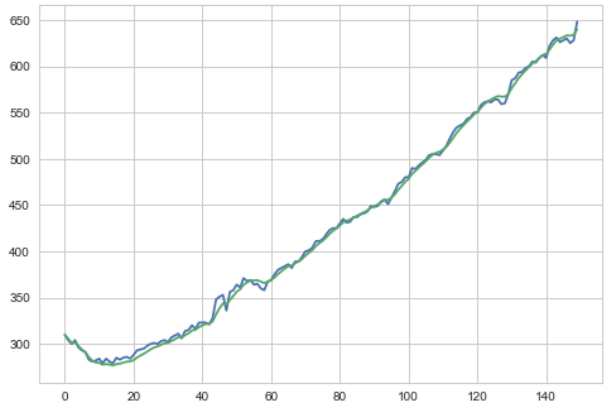
$$\mu_1(k) = A * \mu(k-1)$$
$$\Sigma_1(k) = A * \Sigma(k-1) * A^T + R$$
$$K(k) = \Sigma_1(k) * C^T * (C * \Sigma_1(k) * C^T + Q)^{-1}$$
$$\mu(k) = \mu_1(k) + K(k) * (z(k) - C * \mu_1(k))$$
$$\Sigma(k) = (I - K(k) * C) * \Sigma_1(k)$$

Asiakirjan loppu

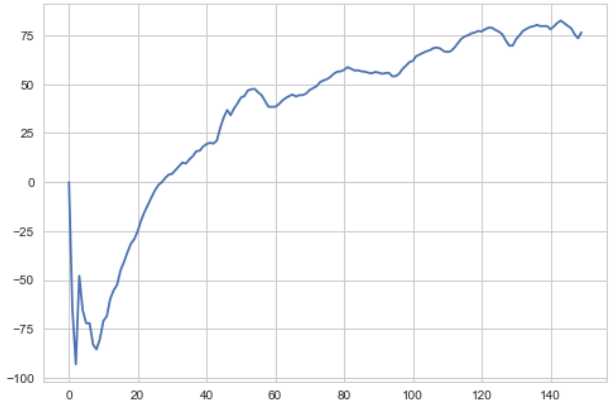
Numerical values are estimated using grid search and 10 step ahead mean prediction error. Values rounded.

Image Object Velocity Estimation

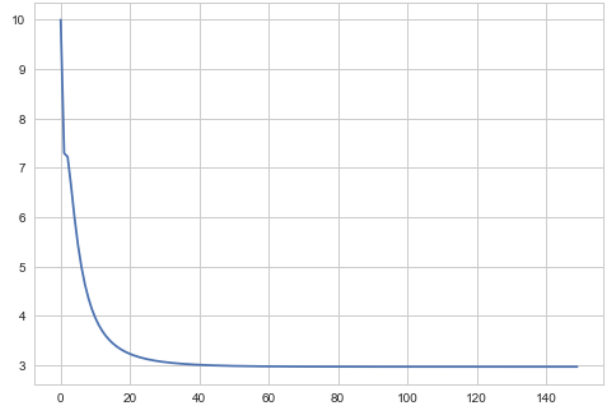
Moving object (car)



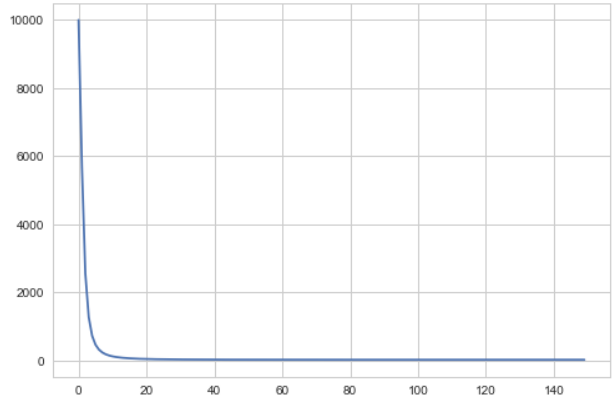
Measured and filtered location (upper left corner)



Estimated velocity



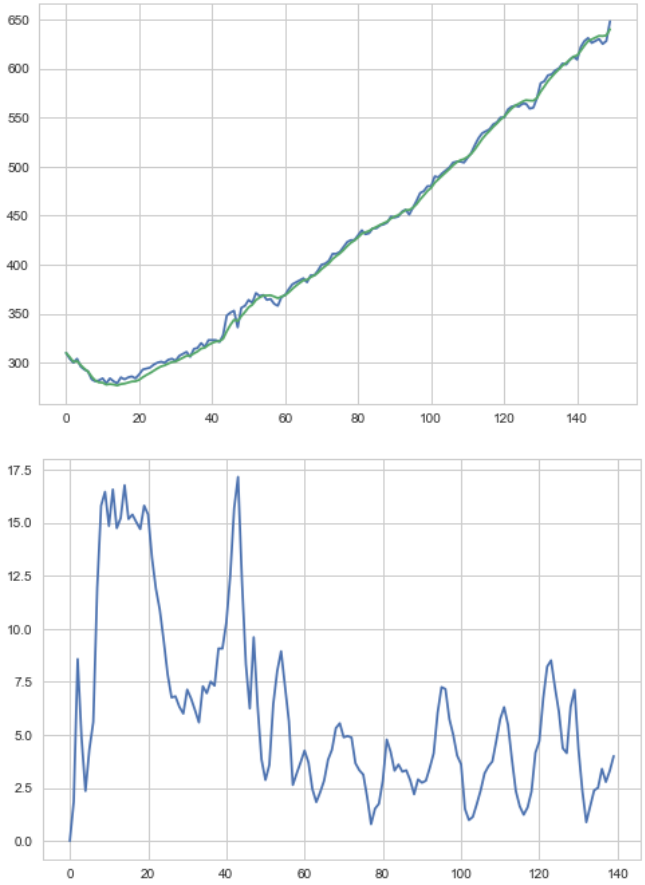
Location variance



Velocity variance

Image Object Velocity Estimation

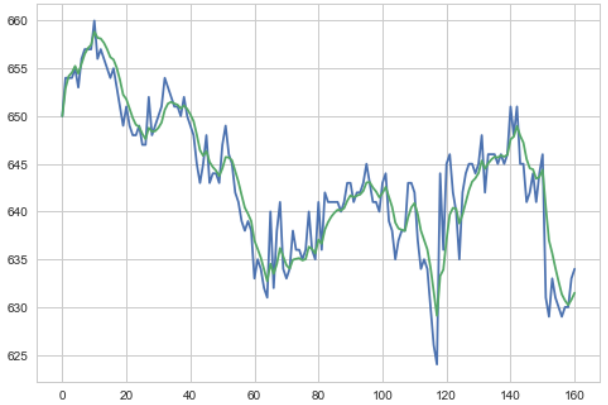
Moving object (car)



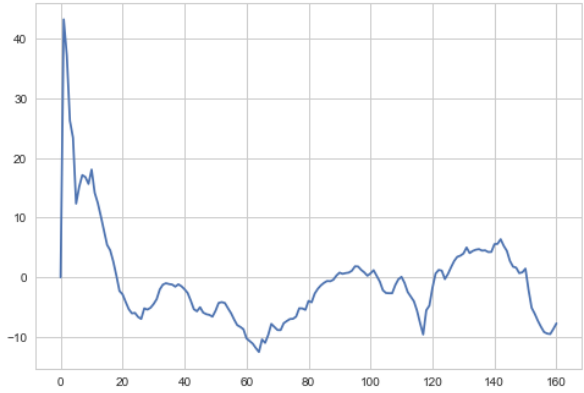
10 step ahead mean prediction error

Image Object Velocity Estimation

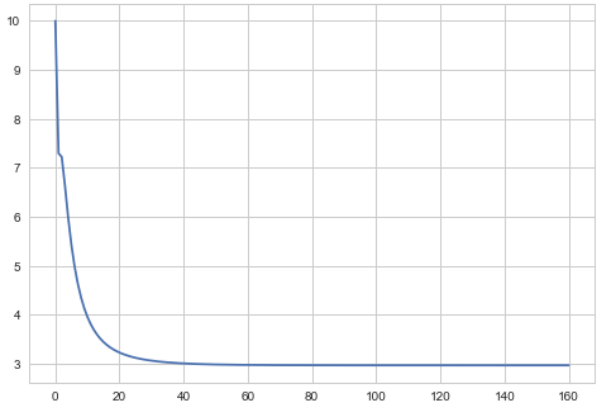
Static object (calf)



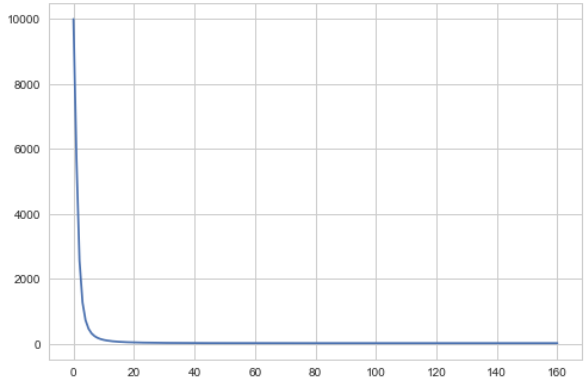
Measured and filtered location (upper left corner)



Estimated velocity



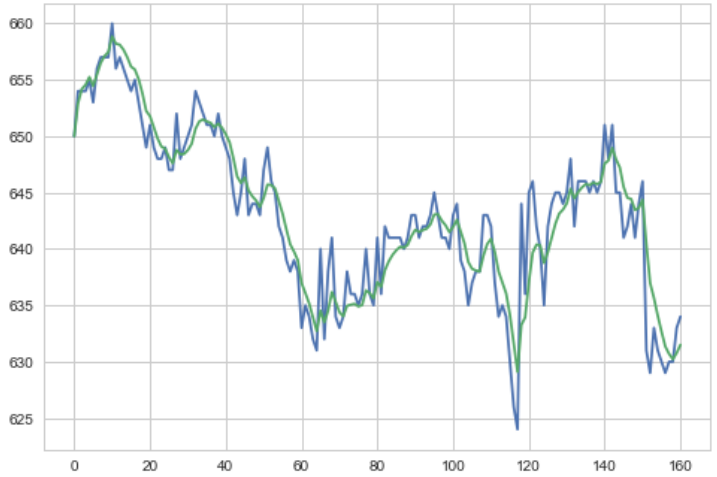
Location variance

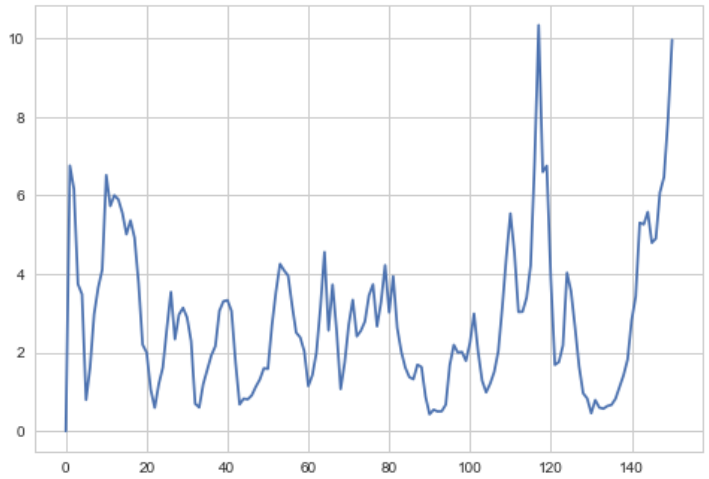


Velocity variance

Image Object Velocity Estimation

Static object (calf)

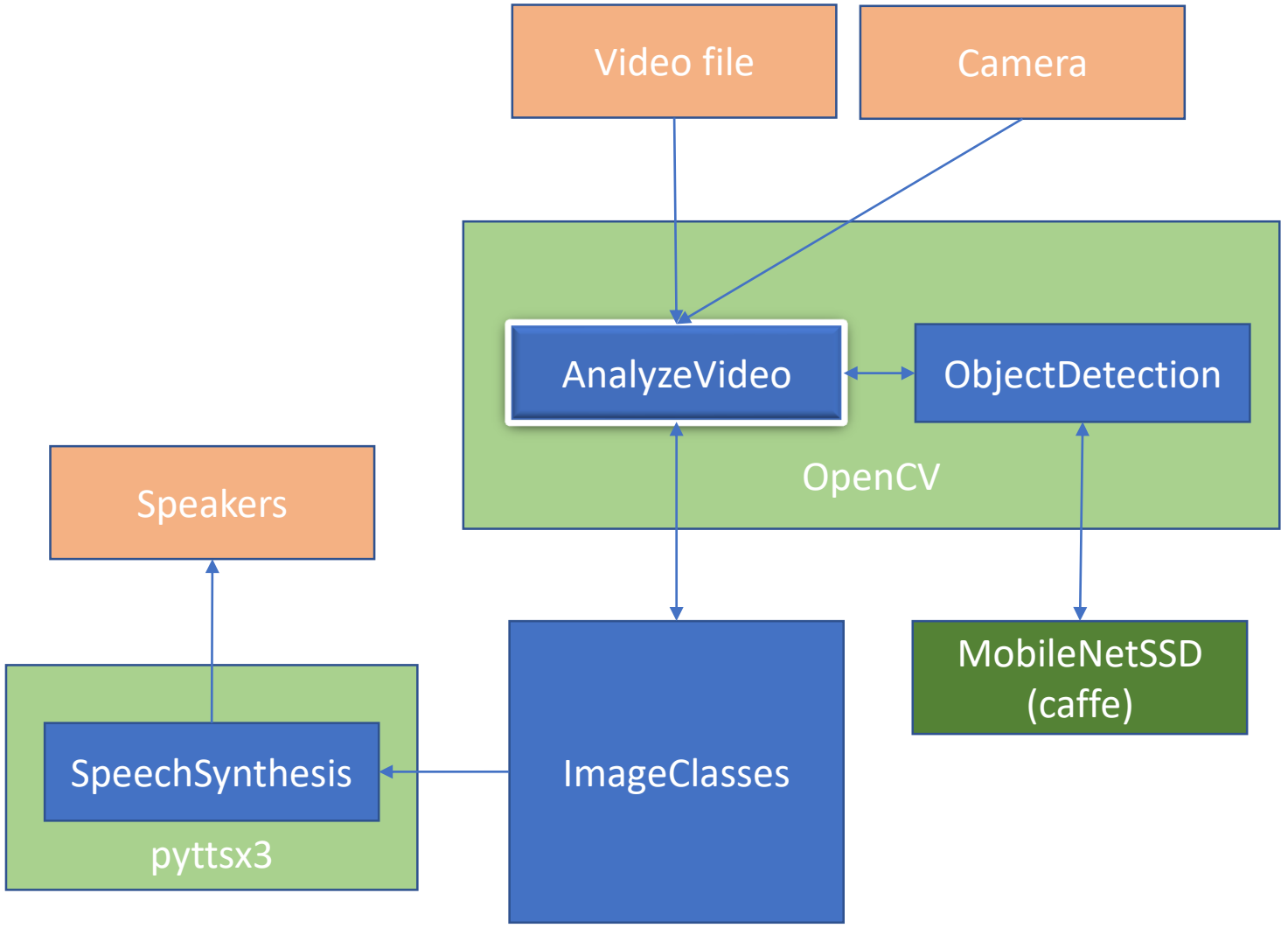




10 step ahead mean prediction error

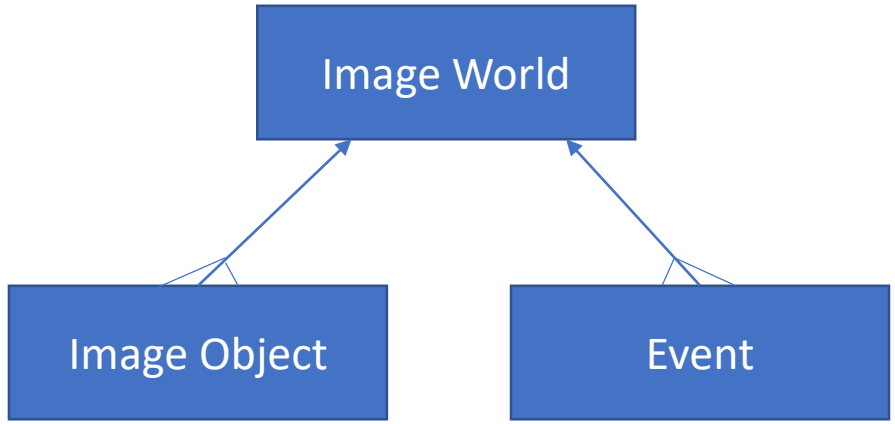
Speech Synthesis

Software Architecture



Speech Synthesis

Entities



Event
- time
- text
- priority

- Event is generated when
  - new image object is created
  - image object status is changed
- Event will pause the video for the duration of speech (not in the final version)
- Events are collected (history)

Confidence Level

SSD Mobilenet implementation:
extract the confidence (i.e., probability) associated with the prediction

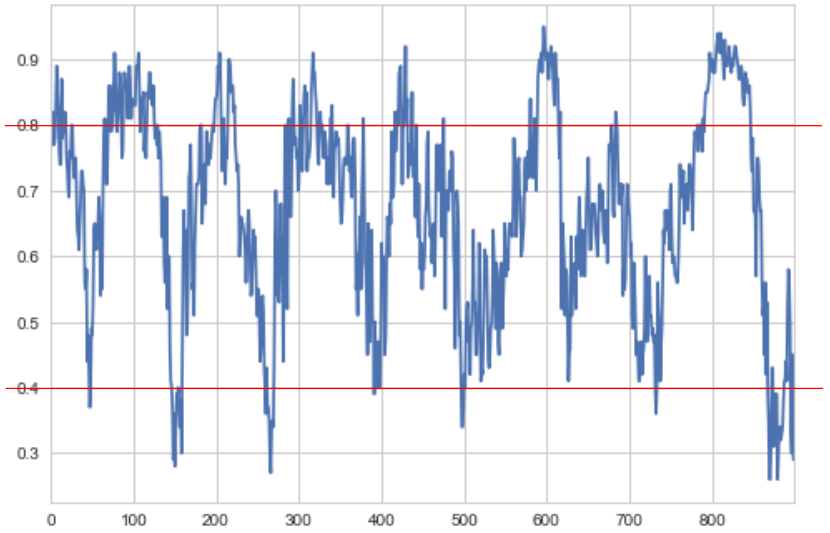
	A	B	C	D	E	F	G	H	I	J
1	Objects detected		Confidence level							
2	Video	Correct	0,00	0,20	0,40	0,60	0,80	0,90	0,95	1,00
3	CarsOnHighway001.mpg	39	49	49	39	36	34	32	32	0
4	Calf-2679.mp4	1	2	2	2	2	1	1	1	0
5	Dunes-7238.mp4	1	7	7	6	5	2	2	2	0
6	Sofa-11294.mp4	1	2	2	1	1	1	1	1	0
7	Cars133.mp4	5	9	9	6	5	5	5	5	0
8	BlueTit2975.mp4	1	3	3	2	1	1	1	1	0
9	Railway-4106.mp4	1	10	10	5	3	3	1	1	0
10	Hiker1010.mp4	1	4	4	0	0	0	0	0	0
11	Cat-3740.mp4	1	3	3	2	2	1	1	1	0
12	SailingBoat6415.mp4	1	1	1	1	1	1	1	1	0
13	AWomanStandsOnTheSeashore-10058.mp4	1	1	1	1	1	1	1	1	0
14	Dog-4028.mp4	1	4	4	2	1	1	1	1	0
15	Boat-10876.mp4	1	2	2	1	1	1	1	0	0
16	Horse-2980.mp4	1	3	3	3	2	2	1	1	0
17	Sheep-12727.mp4	1	1	1	1	1	1	1	1	1

Good value for creating a new image object is between 0.8 and 0.9.

The ’good’ value also depends on other hyperparameters.

Confidence Level

Confidence level has dynamics



create

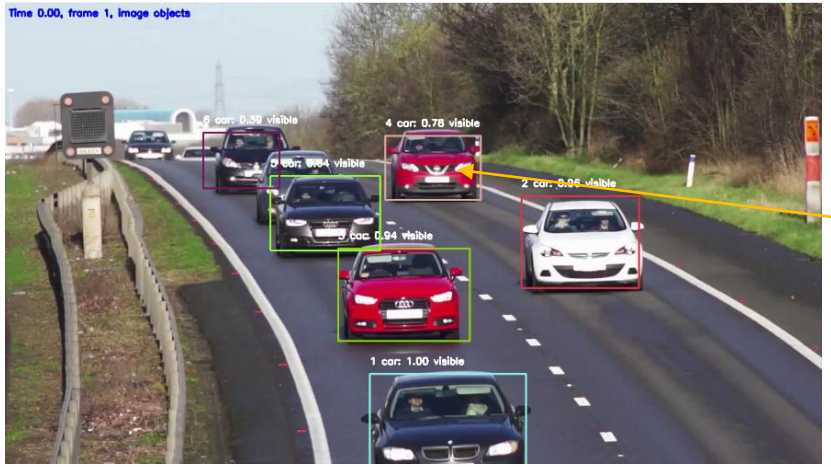
Update (not class)

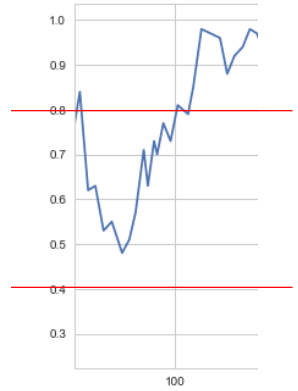
ignore

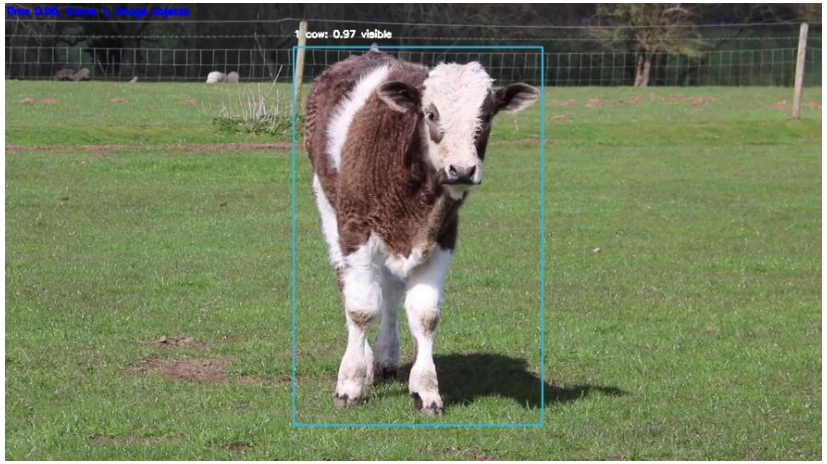
Different levels for creating and updating image object. Hyperparameters:

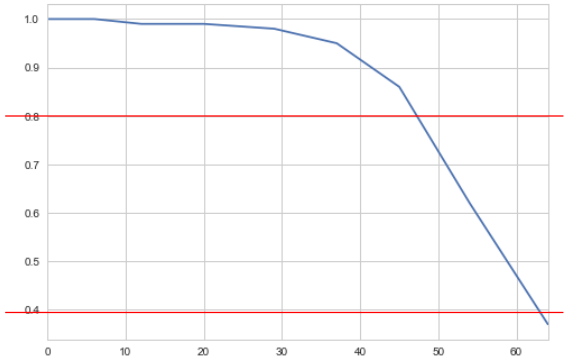
- CONFIDENCE_LEVEL_CREATE (0.8)
- CONFIDENCE_LEVEL_UPDATE (0.4)

Confidence Level

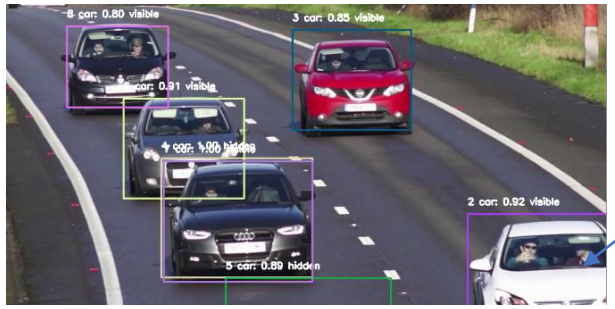








Border Behaviour



Box size and form distorded

- When one of the corners approaches image border, the image object is removed
- Image object is not created if one of the corners is in the border area
- The corresponding world object continues to live in 3d world, with acceleration fixed

time	x_max_c	x_max_m	x_max_p	y_max_c	y_max_m	y_max_p
1.48	1208.859	1209.0	1205.616	646.300	652.0	640.731
1.52	1221.500	1236.0	1212.044	653.697	656.0	649.501
1.56	1232.488	1242.0	1224.941	660.427	661.0	656.939
1.60	1241.599	1246.0	1236.095	668.758	673.0	663.679
1.64	1251.081	1256.0	1245.282	677.391	682.0	672.083
1.68	1258.430	1258.0	1254.848	687.143	694.0	680.794
1.72	1265.965	1266.0	1262.190	694.428	695.0	690.663
1.76	1272.740	1271.0	1269.725	704.340	711.0	697.956
1.80	1280.741	1282.0	1276.471	711.433	711.0	707.979
1.84	1287.573	1286.0	1284.493	717.291	714.0	715.066
1.88	1292.323	1286.0	1291.299	722.517	718.0	720.869
1.92	1292.517	1276.0	1295.946	728.172	725.0	726.022
1.96	1291.385	1273.0	1295.873	731.168	722.0	731.626
2.00	1291.974	1279.0	1294.445	732.465	720.0	734.474
2.04	1291.500	1277.0	1294.826	732.500	718.0	735.572
2.08	1290.547	1276.0	1294.121	733.994	724.0	735.375
2.12	1289.259	1275.0	1292.938	736.016	728.0	736.711
2.16	1289.533	1280.0	1291.424	736.959	727.0	738.606
2.20	1290.113	1282.0	1291.548	737.402	727.0	739.392
2.24	1290.640	1283.0	1292.000	735.994	722.0	739.671

```
In [10]: # image size 1280 * 720
```

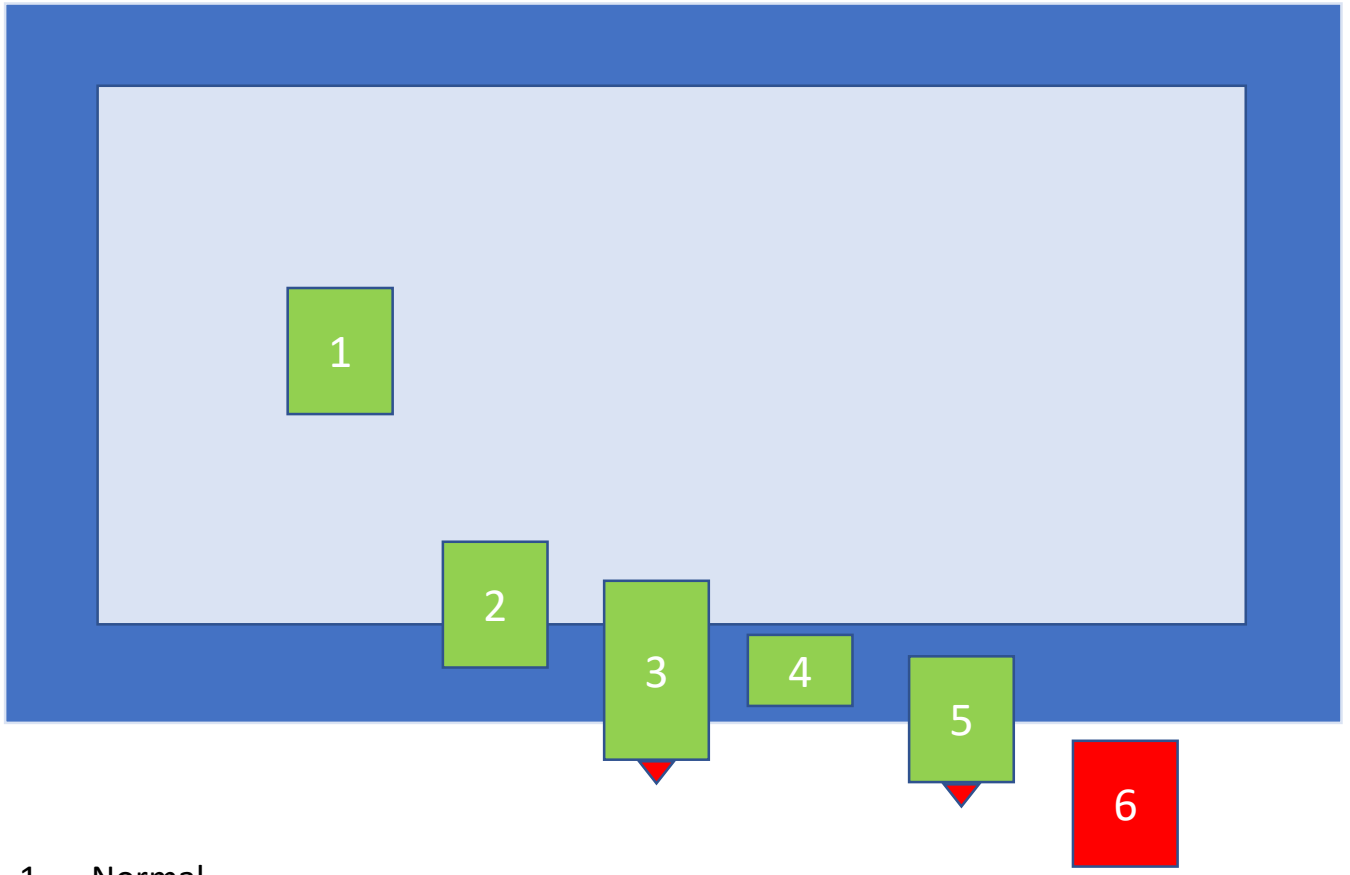
Hyperparameter BORDER_WIDTH

Condition for removal & creation prevention:

x_min < BORDER_WIDTH or
x_max > image_width - BORDER_WIDTH or
y_min < BORDER_WIDTH or
y_max > image_height - BORDER_WIDTH

All types of coordinates applied (measured, predicted and corrected)

Border Behaviour



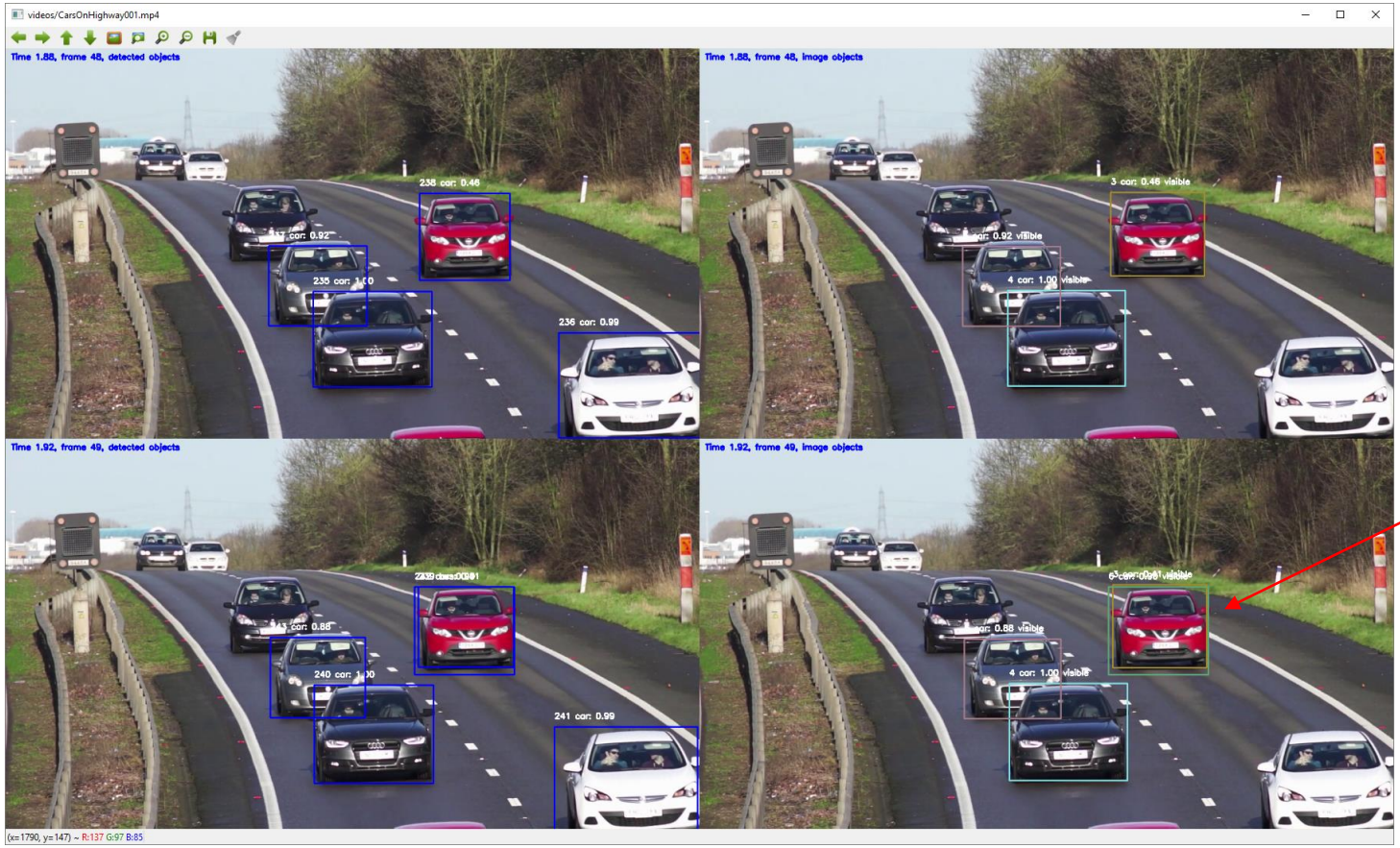
1. Normal
2. Normal
3. Out of screen velocity fixed
4. Normal, image objects not created
5. Out of screen velocity fixed, image objects not created
6. Image object removed

Done for:

- left
- right
- top
- bottom

”Velocity fixed”=No update from the detected object, velocity fixed, location calculated from the velocity

Duplicate Prevention




```
------------------------------------------------
Time 1.92, frame 49
Detected objects (5):
---239 bus   0.41  761  937  273  421 0.04 0.03 0.01 0.01 0.09 0.22 0.14 0.47
---240 car   1.00  569  789  454  635 0.04 0.01 0.00 0.00 0.11 0.55 0.20 0.09
---241 car   0.99 1012 1276  531  725 0.18 0.02 0.00 0.01 0.09 0.42 0.17 0.12
---242 car   0.90  754  938  273  435 0.06 0.03 0.01 0.00 0.08 0.23 0.15 0.44
---243 car   0.88  488  663  366  514 0.04 0.01 0.02 0.01 0.15 0.63 0.09 0.05
Delected object 241 removed due to being in border area:
---  0.99 1012.00 1276.00  531.00  725.00
Image objects (3), predicted new locations:
---3 car   0.46  761.08  933.34  266.59  421.32 0.11 0.04 0.01 0.01 0.07 0.19 0.12 0.45
---4 car   1.00  570.38  787.31  448.50  625.18 0.03 0.00 0.00 0.00 0.08 0.59 0.17 0.12
---5 car   0.92  486.86  667.96  369.59  516.20 0.03 0.01 0.02 0.01 0.15 0.62 0.11 0.06
Cost matrix for Hungarian algorithm:
   0.02    0.93    1.08
   1.14    0.03    0.63
   0.05    0.90    1.04
   1.10    0.51    0.02
```

Duplicate Prevention

Solution:

- Change of class not permitted or is heavily penalized (large distance)
- New image object is not created if the distance is very small to existing image object

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).”

Next Steps

Next steps

Comprehension:

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

To Be Discussed

Method follow-up

- Google search enough?
- Good way of following new papers?

Thank you!

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