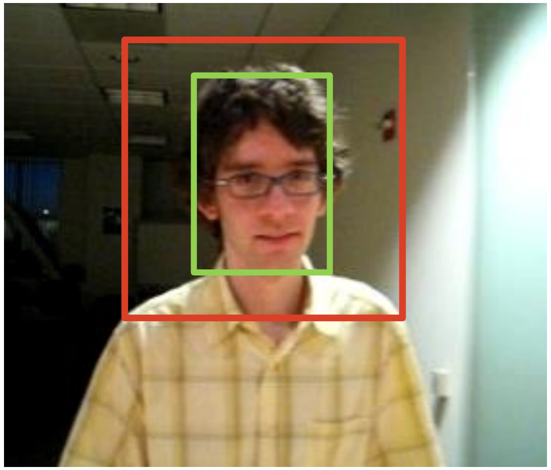
Univerza *v Ljubljani*

Correlation filters and tracking evaluation

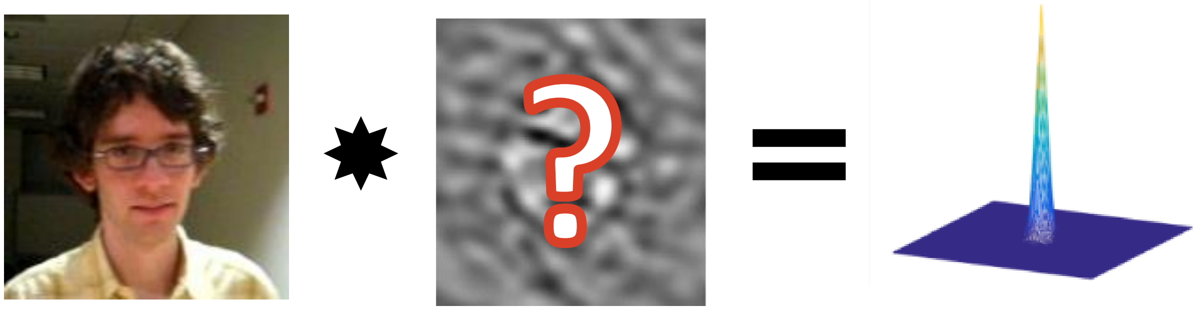
Advanced Computer Vision Methods
Project 3

Visual Cognitive Systems Laboratory,
Faculty of Computer and Information Science,
University of Ljubljana

Correlation filters: idea



Image



Training example: P

Filter: H

Desired response: G

2-D Gaussian

Correlation equation: $\mathbf{G} = \mathbf{P} \star \mathbf{H}$

green bbox: target region, red bbox: search region (filter size = capture range)

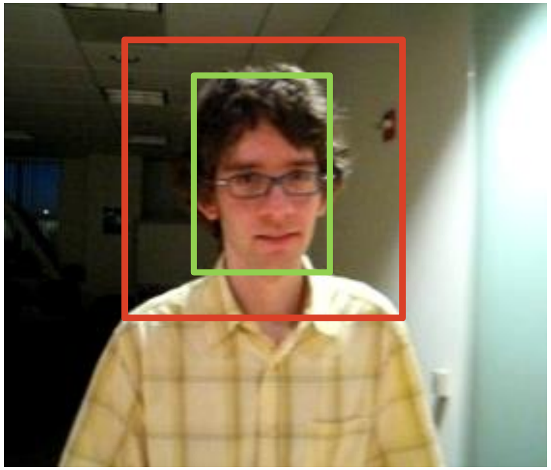
Correlation is slow operation: Fourier-transform trick

$$\hat{\mathbf{G}} = \hat{\mathbf{P}} \odot \hat{\mathbf{H}}^{\dagger} \qquad \hat{\mathbf{A}} = \mathcal{F}(\mathbf{A})$$

Point-wise product

$\dagger$ denotes the complex-conjugate operation

Correlation filters: construction

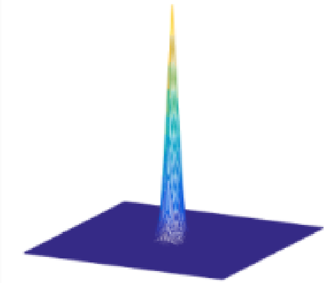


Training example: P

✸

Filter: H

=



Desired response: G

$$\arg\min_{\tilde{\mathbf{H}}} |\mathbf{P} * \tilde{\mathbf{H}} - \mathbf{G}|^2 = \arg\min_{\hat{\mathbf{H}}^\dagger} |\hat{\mathbf{P}} \odot \hat{\mathbf{H}}^\dagger - \hat{\mathbf{G}}|^2$$

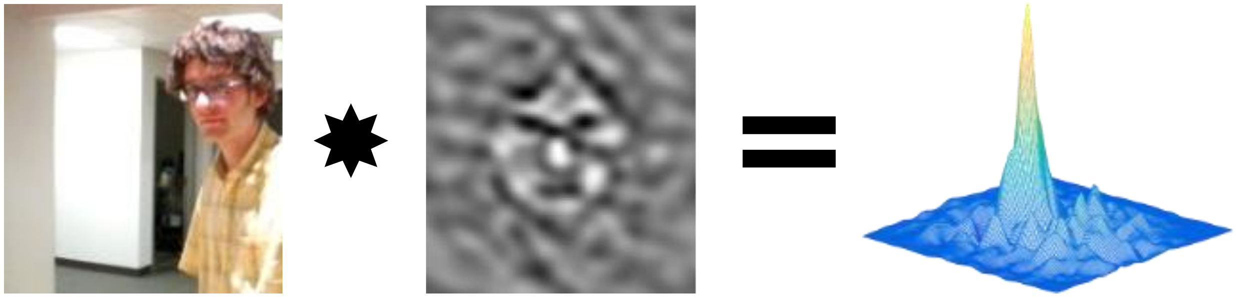
Closed-form solution: $\hat{\mathbf{H}}^\dagger = \dfrac{\hat{\mathbf{G}} \odot \hat{\mathbf{P}}^\dagger}{\hat{\mathbf{P}} \odot \hat{\mathbf{P}}^\dagger}$ ← Point-wise division

the hat symbol ($\widehat{\ }$) denotes the Fourier domain image

$\dagger$ denotes the complex-conjugate operation

Correlation filters: localization

Target position in previous frame

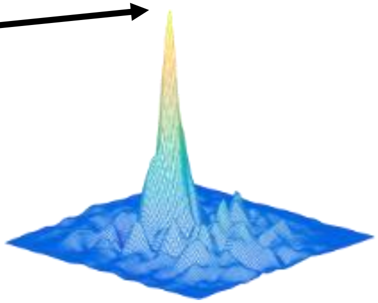


Localization patch: L

Filter: H

Correlation response: G‘

Correlation equation: $\mathbf{G}' = \mathcal{F}^{-1}(\hat{\mathbf{L}} \odot \hat{\mathbf{H}}^{\dagger})$



New position of the target:
position of the maximum peak in G‘

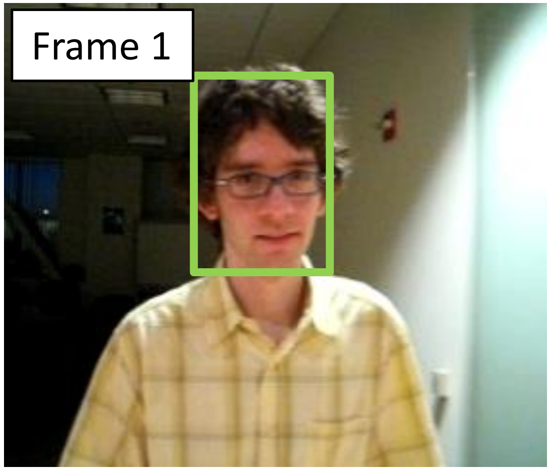
Correlation filters: update

- Construct filter in frame t: $\hat{\tilde{\mathbf{H}}}_t^{\dagger}$
  - The same approach as in frame t = 1
- Update current filter (exponential forgetting):

$$\hat{\mathbf{H}}_t^{\dagger} = (1-\alpha)\hat{\mathbf{H}}_{t-1}^{\dagger} + \alpha\hat{\tilde{\mathbf{H}}}_t^{\dagger}$$

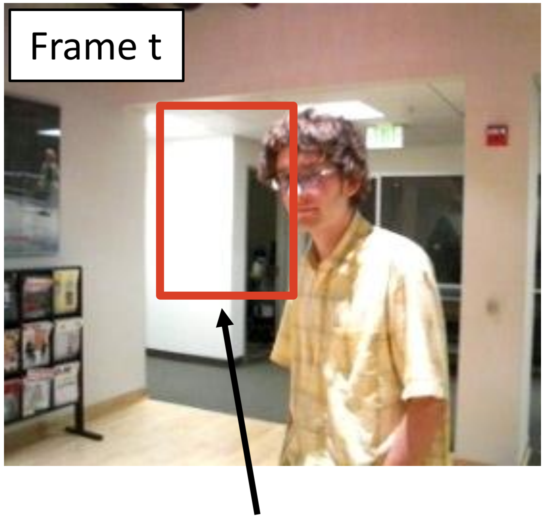
- Observe how the performance changes for different learning rates α

Correlation filters: the pipeline



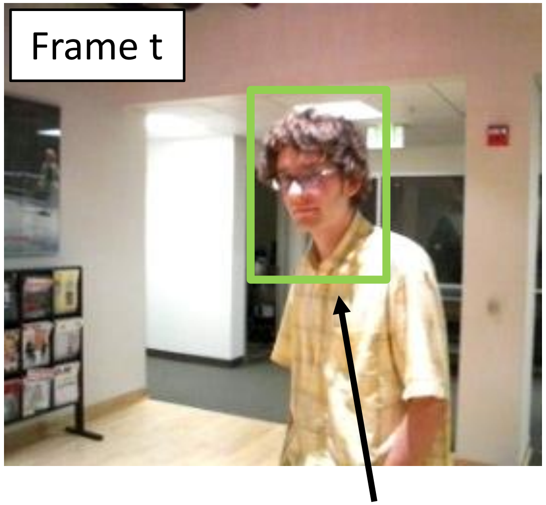
Initialization image and position

Initialize the tracker using this position



Target position in previous frame

Localize target: estimate new position



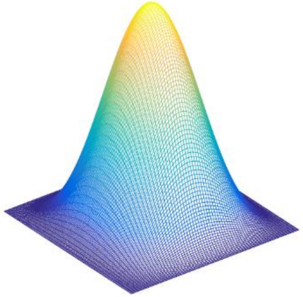
New target position

Extract patch here again and update the filter

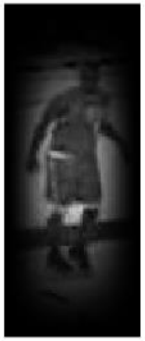
Correlation filters: tips & tricks

- Use cosine window on image patch (P and L)
  - Due to the periodity and to focus on the center
  - Calculate only once, due to the speed

⊙



=

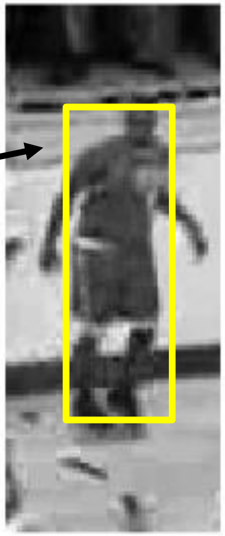


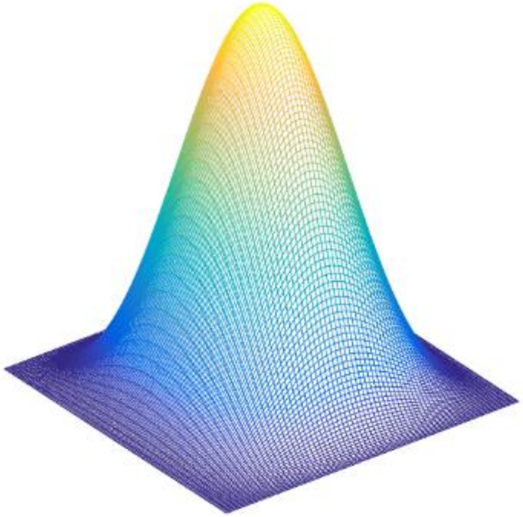
- Calculate ideal Gaussian response **G** only once
  - Due to the speed
- Make sure that you are using 2D Fourier transform

Correlation filters: tips & tricks

- Patch and window can be larger than target

Size of the reported region
is still the same as ground-truth

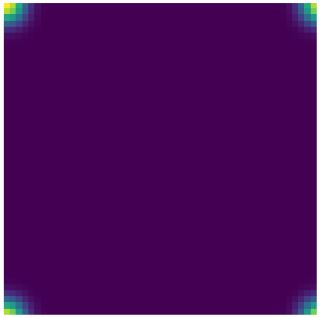




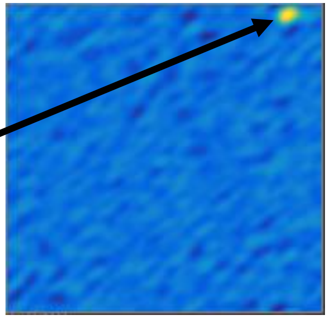
Extracted patch is enlarged
for some factor

CF implementation: Localization

- Gaussian peak (which is used to create filter) is circularly shifted
    - Maximum is in the top-left corner
      output of the create_gauss_peak function from assignment material



- In localization step:
    - `x,y = position of the maximum in correlation response`
    - `if x > width / 2, then x = x – width`
      `if y > height / 2, then y = y – height`
    - `new x = old x + x, new y = old y + y`



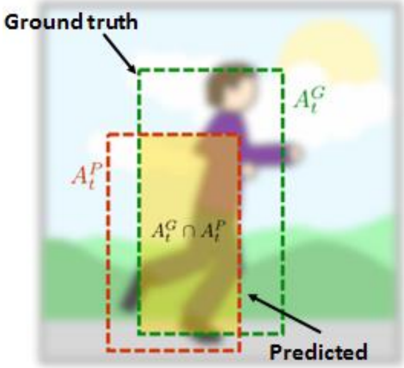
Visual Tracking Evaluation: VOT

- Evaluate your tracker and compare it with others
- 4 Challenges: VOT2013, 2014, 2015, 2016
  http://www.votchallenge.net/
- Toolkit: run experiments, analysis, visualizations
- Datasets: 4 different tracking datasets with ground-truth annotations
- Three measures:
  - Accuracy (average overlap)
  - Robustness (average number of failures)
  - EAO (Expected average overlap)

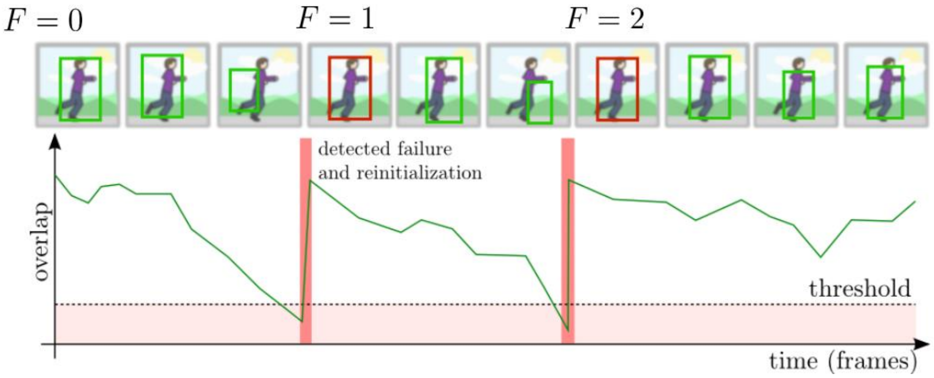
We will use only these two

Visual Tracking Evaluation: Measures

- Accuracy (average overlap)

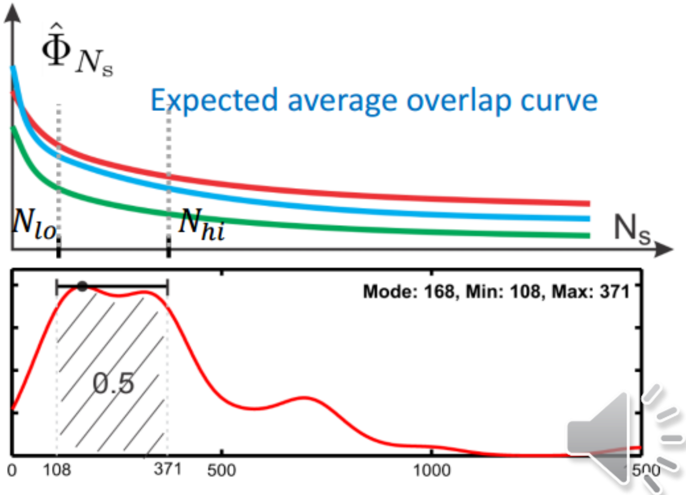


- Robustness (average number of failures - reinitializations)



- EAO (Expected average overlap)

  Combination of both measures



Tracking toolkit lite

- Simpler evaluation protocol (for your assignments), than actual VOT protocol
- Github page: https://github.com/alanlukezic/pytracking-toolkit-lite
- See Github page for instructions how to integrate tracker
- You have to implement your tracker within the Tracker class (see NCC example)
    - Note that this is not the same Tracker class as for the run_tracker.py script (use Tracker class from utils.tracker which is located in the toolkit)

More Advanced: VOT Toolkit

- Integrate a tracker into the VOT toolkit

  https://github.com/votchallenge/vot-toolkit

- Or use the new python version of the VOT toolkit

  https://github.com/votchallenge/vot-toolkit-python

- Documentation:

  http://www.votchallenge.net/howto/

- Note: Requires a C++ compiler

- Obtain results and comparison:

  `run_experiments` / `run_analysis`

- Run **only baseline** experiment