Master in Computer Vision Barcelona

Project Module 6 Coordination

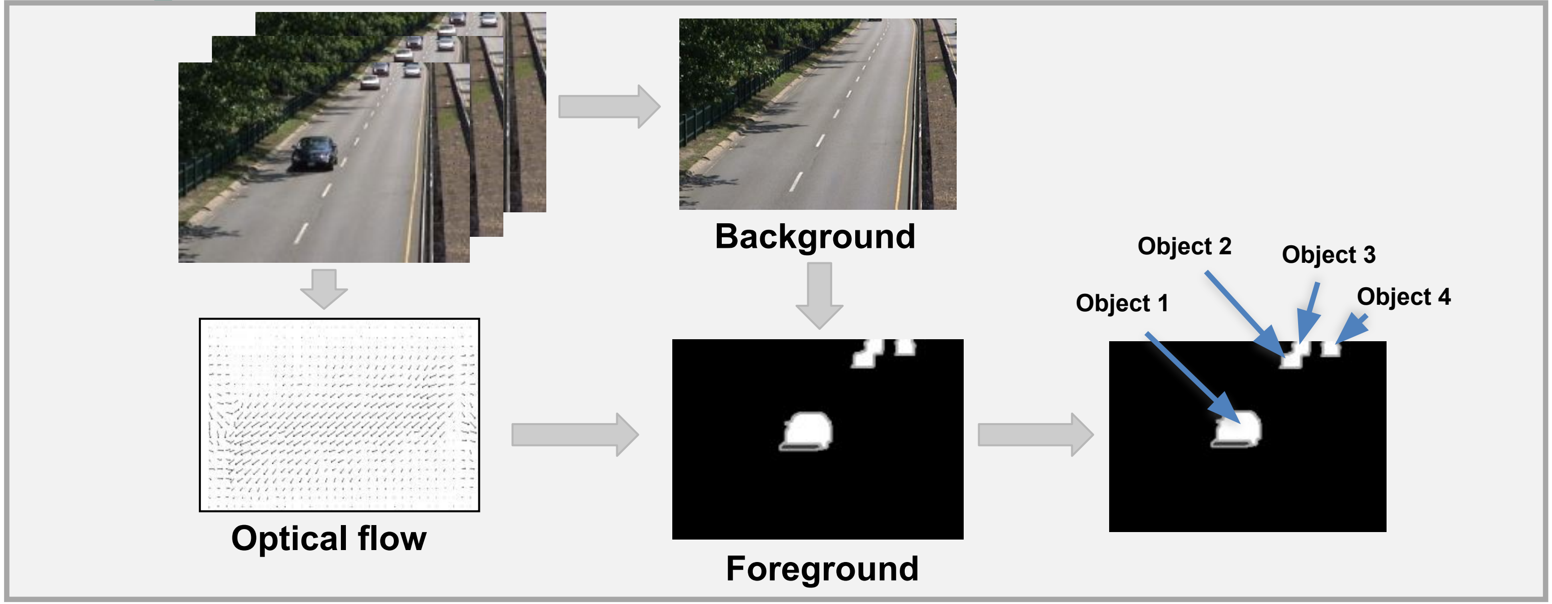
Video Surveillance for Road Traffic Monitoring

J. Ruiz-Hidalgo / X. Giró

j.ruiz@upc.edu / xavier.giro@upc.edu

Week 2: Tasks Description

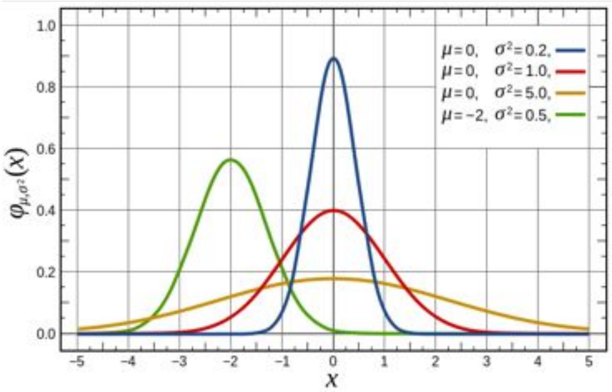
Project Schedule



Week 1	Week 2	Week 3	Week 4	Week 5	Week 6
• Introduction • DB • Evaluation metrics	• Background estimation • Stauffer & Grimson	• Segmentation • Object Detection • Tracking	• Optical flow • Tracking	• Multiple cameras • Speed	• Presentation workshop

Goals Week 2

- **Background estimation**
  - Model the background pixels of a video sequence using a simple statistical model to classify the background / foreground
    - Single Gaussian per pixel
    - Adaptive / Non-adaptive
  - The statistical model will be used to preliminarily classify foreground



- **Comparison with more complex models**

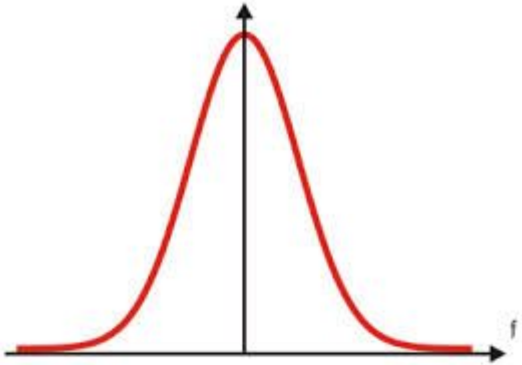
Tasks

- Task 1.1: Gaussian distribution
- Task 1.2 & 1.3: Evaluate results
- Task 2.1: Recursive Gaussian modelling
- Task 2.2: Evaluate and compare to non-recursive
- Task 3: Compare with state-of-the-art
- Task 4: Colour sequences

Sequence S03 - C010

Task 1.1: Gaussian modelling

- **1 Gaussian function to model each background pixel**
  - First 25% of the test sequence to model background
  - Mean and variance of pixels



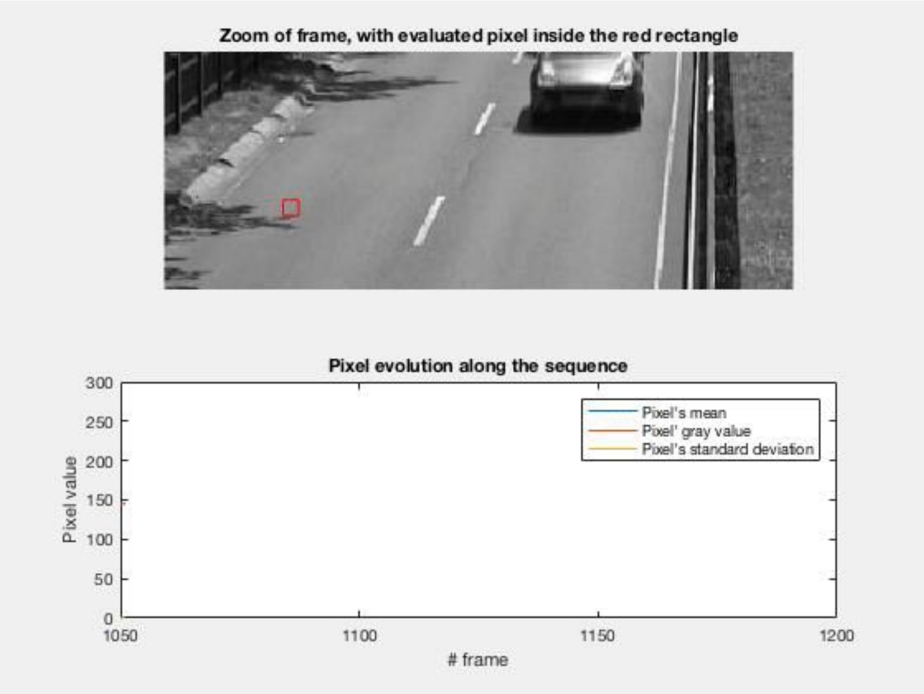
- **Second 75% to segment the foreground and evaluate**

for all pixels i **do**
    if $|I_i - \mu_i| \geq \alpha \cdot (\sigma_i + 2)$ **then** $\qquad \triangleright$ +2 to prevent low values of σ_i
        pixel $\rightarrow$ Foreground
    else
        pixel $\rightarrow$ Background
    end if
end for

Task 1.1: Gaussian modelling (baselines)

Team 1 2016-2017

Using all pixels for the computation of the mean and deviation:



It can be seen that the standard deviation increases greatly throughout the sequence, due to the driving car and the mean is slightly changed.

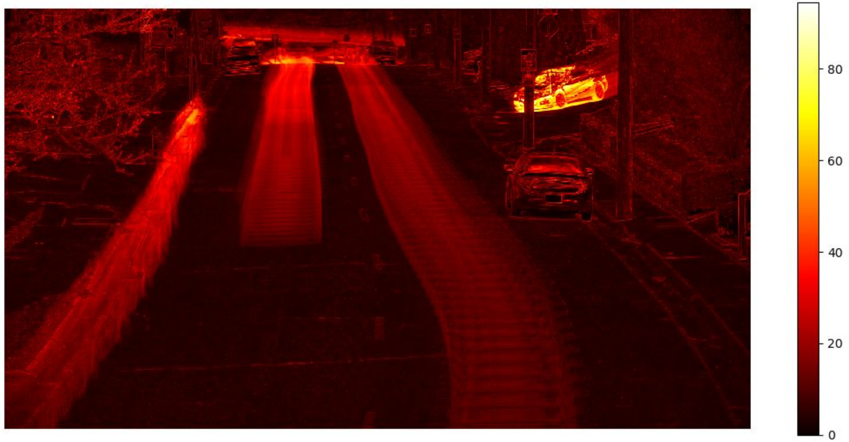
Task 1.1: Gaussian modelling (Team 4 2020 baseline)

We take the first 25% of the total frames in the video. We stack them and compute the mean and the variance with the np.mean and np.std functions from numpy.

To calculate mAP we will sort our results by are as it is faster and we saw last week that the results were almost the same.

Mean

Variance



Task 1.2: $AP_{0.5}$ vs Alpha

- **Evaluate Task 1**
  - $AP_{0.5}$ on detected connected components
  - Filter noise and group in objects + bounding box
  - Over alpha threshold
  - Decide (and explain) if parked/static cars are considered
    - Use annotation provided last week (previous students)

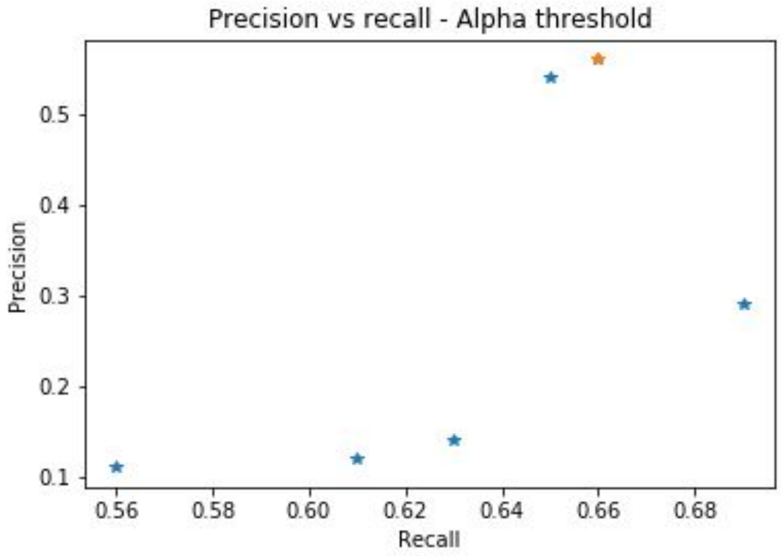
Task 1.2: $mAP_{0.5}$ vs Alpha (baselines)

Team 1 / 2018-2019

Best threshold: Alpha = 11

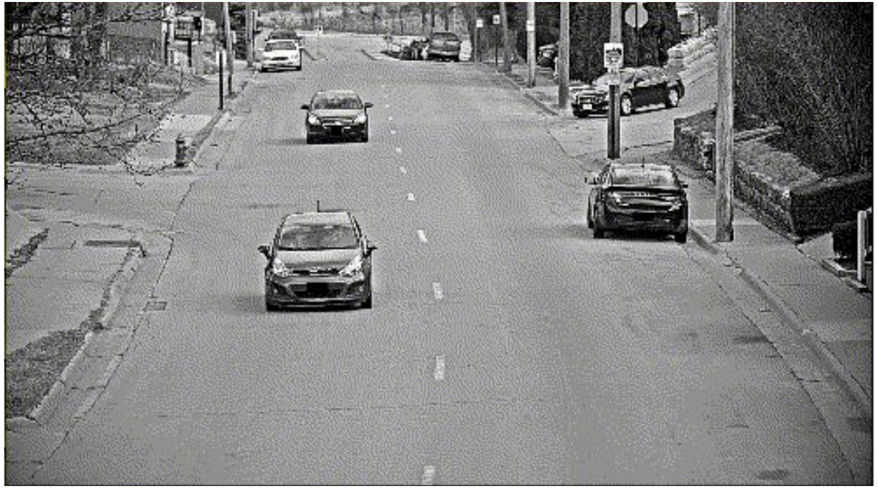
mAP = 0.29

Alpha	Precision	Recall	F1-score
3.5	11%	56%	18%
4	12%	61%	21%
5	14%	63%	22%
7	29%	69%	41%
11	56%	66%	61%
15	54%	66%	60%



Task 1.2: $mAP_{0.5}$ vs Alpha (baselines)

Team 3 / 2018-2019

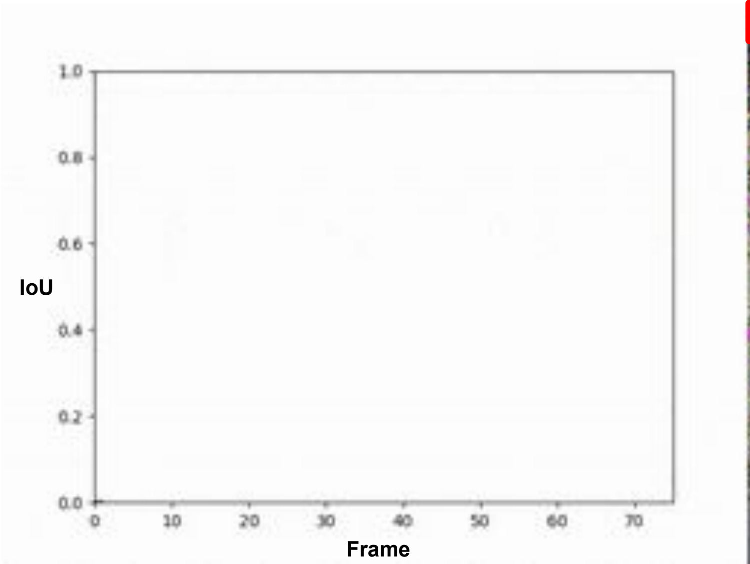


For each frame:

1. Create a mask setting as **foreground** the pixels where: $|I_i - \mu_i| \le \alpha(\sigma^2 + 2)$
2. Consider as background the pixels that are not in the provided **ROI**
3. Apply **morphological filtering** to the mask for filling holes and filtering noise
4. Detect all **connected components** and **filter** them by height, width and ratio*
5. Apply a **bounding box** surrounding each of the filtered connected components
6. Compute mAP of detections against ground truth ones.

Task 1.2: mAP vs alpha (Team 5 2020 baseline)

Yellow: real bgseg algorithm (dim)
Fucsia: bgseg treated with morph



For low values of α, as the threshold is very restrictive many areas that don't have moving objects are detected as foreground. This is due to slight variations on the illumination of the scene. This also happens because the noise introduced by the video compression.

For Higher values of α, we generally obtain better results as only the moving objects are detected.

Once a certain value of α is reached, the performance starts decreasing as moving objects with colors similar to the background stop being detected as foreground

Task 2.1: Adaptive modelling

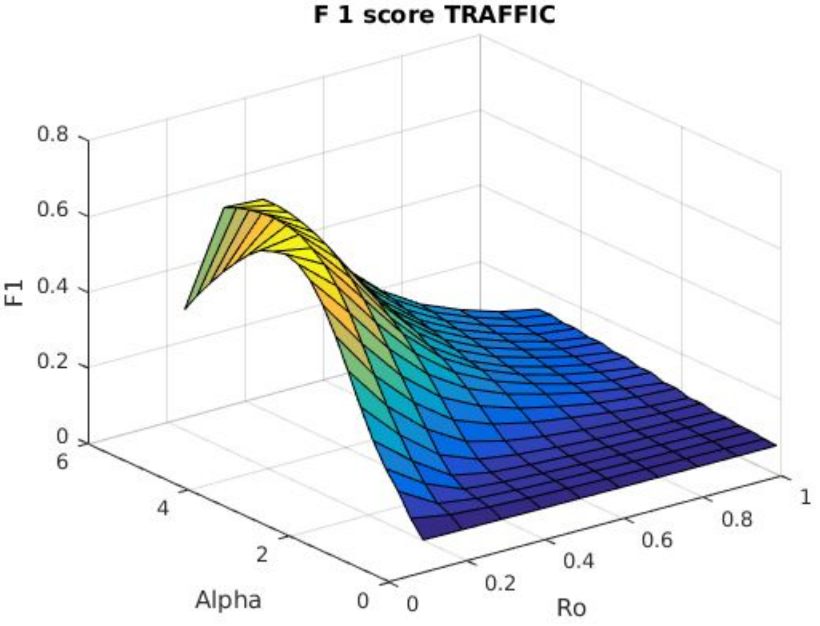
- **Adaptive modelling**
  - First 25% frames for training
  - Second 75% left background adapts

$$
\begin{aligned}
&\textbf{if } \text{pixel } i \in \text{Background } \textbf{then}\\
&\quad \mu_i = \rho \cdot I_i + (1-\rho) \cdot \mu_i\\
&\quad \sigma_i^2 = \rho \cdot (I_i - \mu_i)^2 + (1-\rho) \cdot \sigma_i^2\\
&\textbf{end if}
\end{aligned}
$$

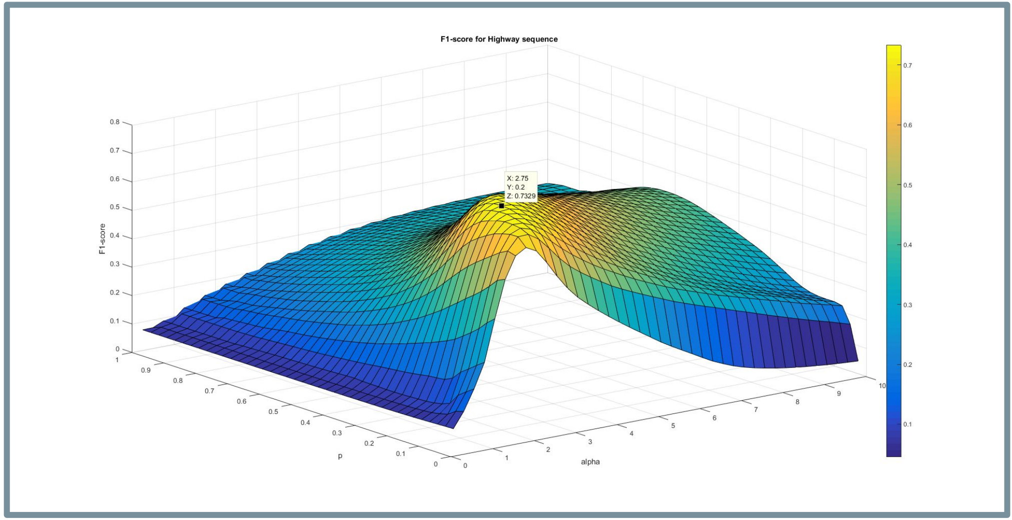
- **Best pair of values (α, ρ) to maximize mAP**
  - Possible two methods:
    - Obtain first the best α for non-recursive, and later estimate **ρ** for the recursive cases
    - Optimize (α, ρ) together with [grid search or random search] (discuss pros & cons).

Task 2.1: Adaptive modelling (baselines)

Team 1 /
2015-2016

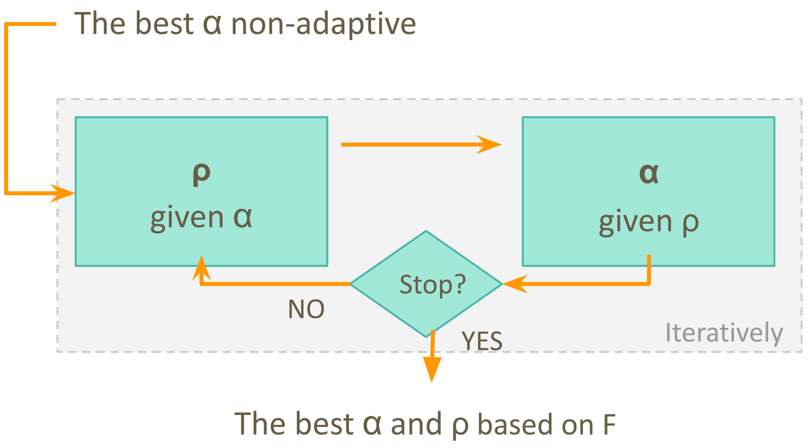


Team 3 /
2015-2016



Task 2.1: Adaptive modelling (baselines)

Team 2 / 2015-2016

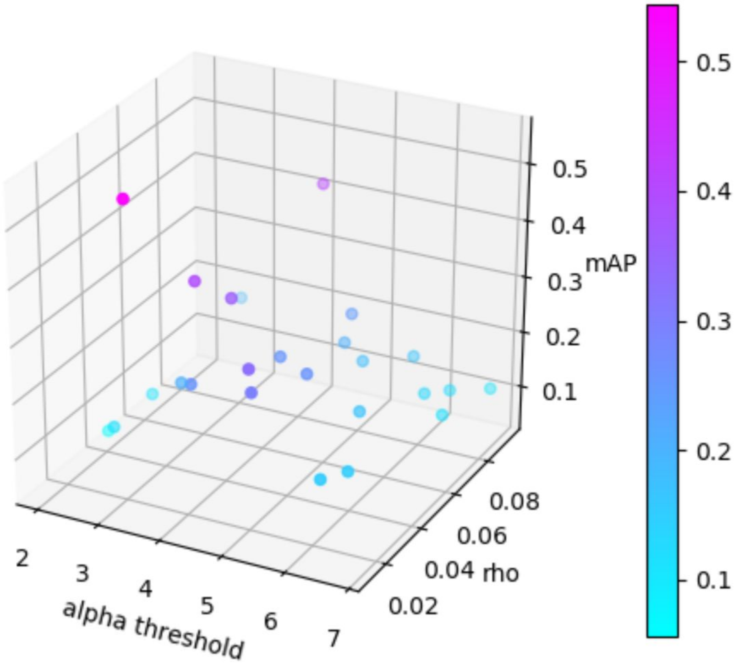


The maximum F is obtained when the system converges:

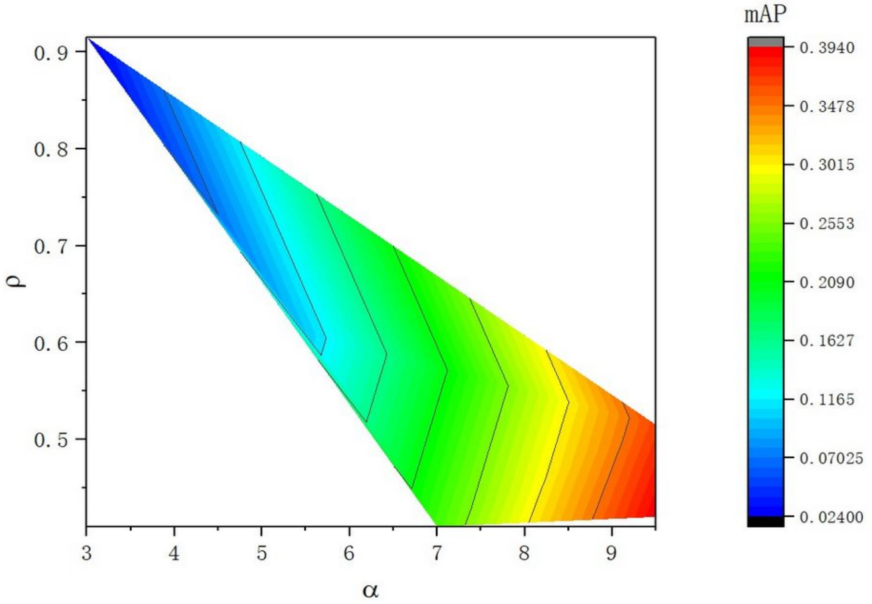
- ➔ |F(t-1) - F(t)| < tolerance
- ➔ number of iterations = maximum number of iterations

Task 2.1: Adaptive modelling (baselines)

Team 2 / 2019-2020



Team 3 / 2019-2020



Task 2.2: Comparison of adaptive vs non

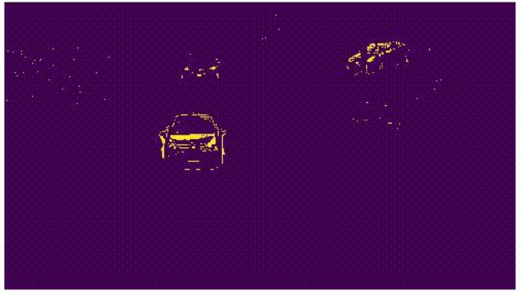
- Compare both the adaptive and non-adaptive version and evaluate them over mAP measures

Task 2.2: Comparison (baselines)

Comparing non-adaptive and adaptive foreground detection obtained mask (before post-processing):

Non-adaptive

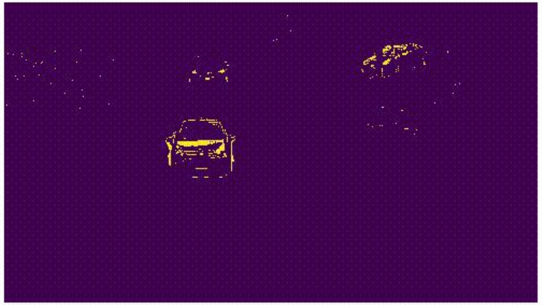
Background illumination changes have a higher impact on the foreground detection.



mAP = 0.2959 (alpha = 1.9737)

Adaptive

Background noise tends to disappear, but foreground might not be detected.



mAP = 0.4319 (alpha = 1.75 / rho = 0.3981)

Task 2.2: Comparison (baselines)

Team 3 / 2019-2020

Non-adaptive

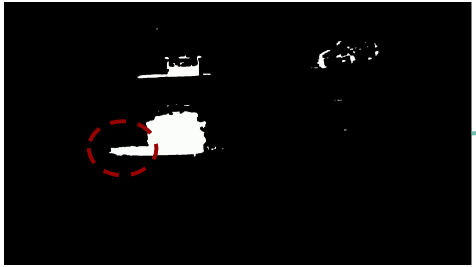
Alpha = 2, mAP = 0.2162

Adaptive

Alpha = 9.5, Rho = 0.42, mAP = 0.394

Be creative!

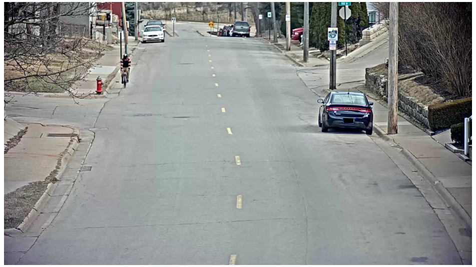
Team 4 / 2018-2019



The model can be further improved by:

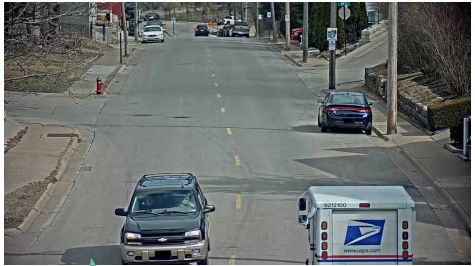
- Detecting and **removing shadows** to reduce false positives
- Refine/reconstruct connected components boundaries, so as to:
  - Avoid cutting them
  - E.g.: thin lines in horizontal and vertical to close objects (morphology)
- (*) Naive median + Gaussian filtering does not reduce compression artifacts
  - Use a specific ‘**deblocking**’ algorithm & check if it helps
- To better model illumination changes, use a **variable background model** (GMMs)

Exaggerated by gif compression!



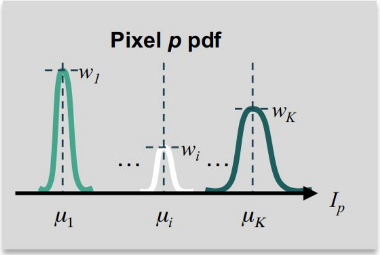
μ_1 , σ_1, w_1

...



μ_i , σ_i, w_i

μ_k , σ_k, w_k



Task 3: Comparison with state-of-the-art

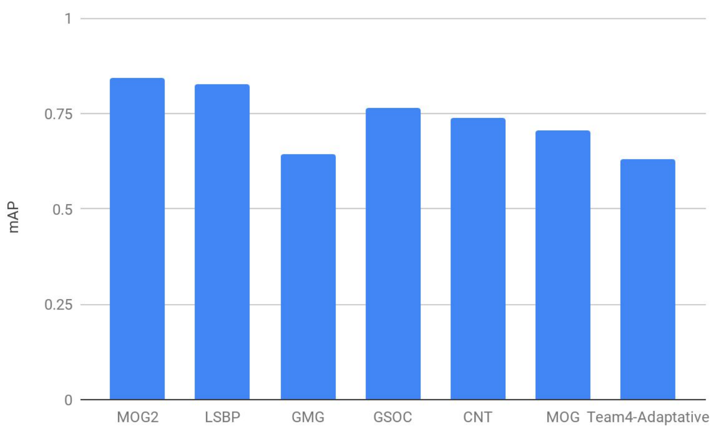
- **Compare with state-of-the-art**
  - P. KaewTraKulPong et al. *An improved adaptive background mixture model for real-time tracking with shadow detection.* In Video-Based Surveillance Systems, 2002. Implementation: BackgroundSubtractorMOG (OpenCV)
  - Z. Zivkovic et al. *Efficient adaptive density estimation per image pixel for the task of background subtraction*, Pattern Recognition Letters, 2005. Implementation: BackgroundSubtractorMOG2 (OpenCV)
  - L. Guo, et al. *Background subtraction using local svd binary pattern*. CVPRW, 2016. Implementation: BackgroundSubtractorLSBP (OpenCV)
  - St-Charles, Pierre-Luc, and Guillaume-Alexandre Bilodeau. *Improving Background Subtraction using Local Binary Similarity Patterns.* Applications of Computer Vision (WACV), 2014. Implementation: LOBSTER (GitHub)
  - M. Braham et al. *Deep background subtraction with scene-specific convolutional neural networks*. In International Conference on Systems, Signals and Image Processing, 2016. No implementation (https://github.com/SaoYan/bgsCNN similar?)
- Evaluate to comment which method (single Gaussian programmed by you or state-of-the-art) performs better

Task 3: Comparison with state-of-the-art (All teams)

Best AP_{50} (best configuration for you: adaptive, non-adaptive, other)

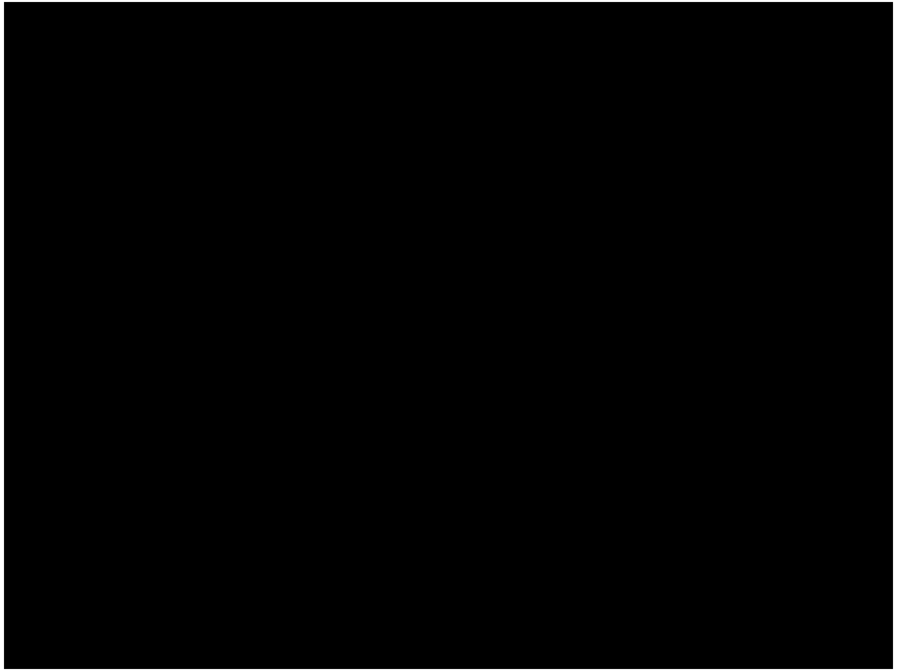
Team ID	Others	Best yours
Team 1		
Team 2		
Team 3		
Team 4		
Team 5		
Team 6		

Task 3: Comparison with state-of-the-art (baselines)



Mixture gaussian models, **MOG2**, can model better the background as the use various gaussians for that purpose. The algorithm adapts better to:

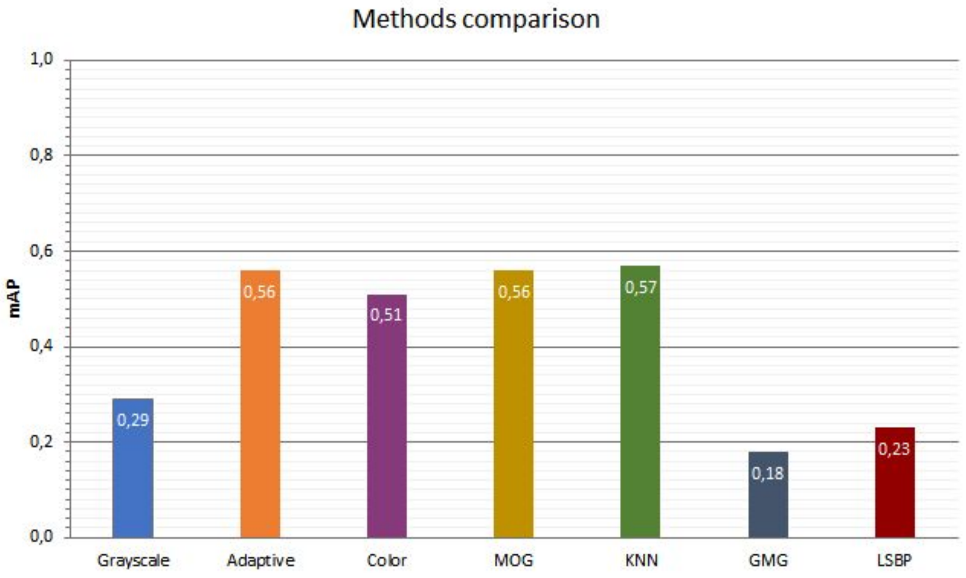
- Shadows are detected as a separate object than foreground, but discarded with image post-processing
- Moving objects on the background (such as trees or plants)
- Illumination (or camera exposure) changes



Results for MOG2

Task 3: Comparison with state-of-the-art (Team 4 2020 baseline)

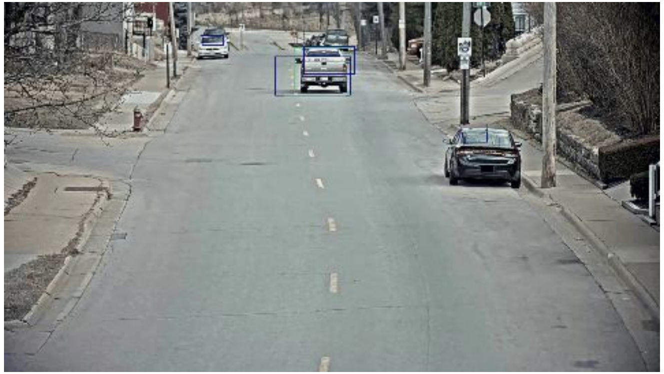
We think the reason why we have obtained similar results is that our method has been fine-tuned for this specific video (alpha, rho) whilst the one from OpenCV is using the default parameters. To make a fair comparison we should be comparing with a test video.



Video shows qualitative results from our best model (grayscale adaptive) and the KNN algorithm from OpenCV.

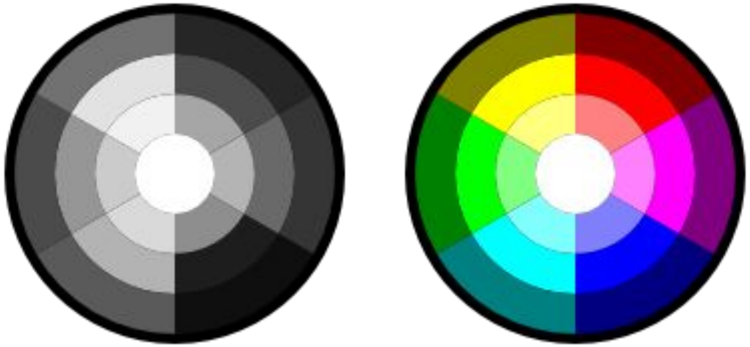
KNN

Our implementation



Task 4: Colour sequences

- Update your implementation to support colour sequences
  - Decide colour space? RGB vs YUV? other?
  - Number of Gaussians needed?



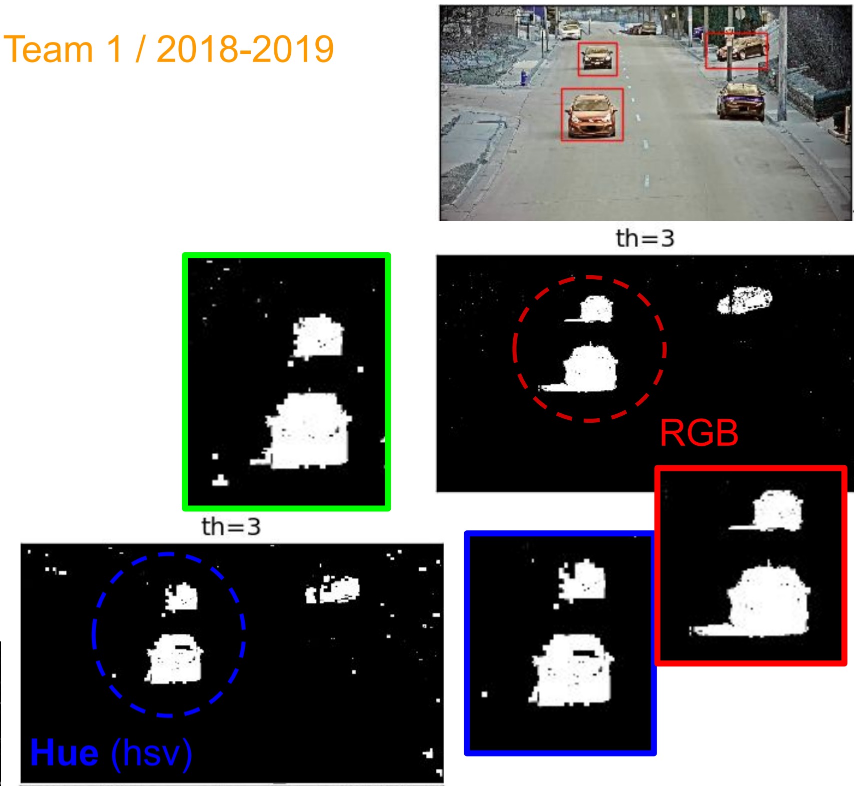
Task 4: Color (baselines)

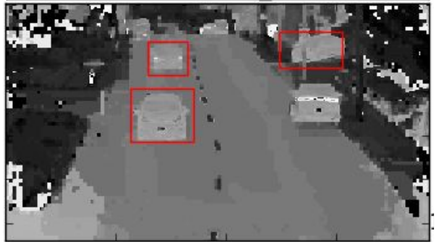
Team 1 / 2018-2019

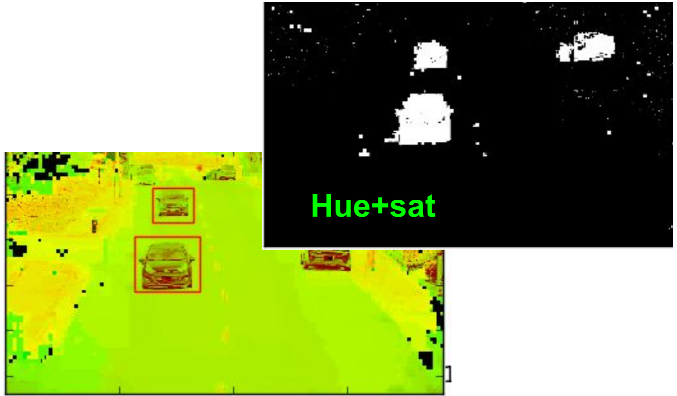
Taking advantage of the chromatic components of other color- space , for example:

- Hue, Saturation in the hsv
- A,B in Lab
- Cr,Cb in YCrCb

* taking into account that all those color spaces were transformed from rgb, therefore they are not ideal







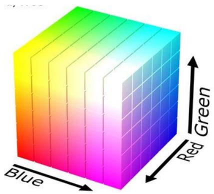
Hue is able to distinguish between shadows and foregrounds, because the **chrome** in both cases stays the same.

Task 4: Color (Team 2 2020 baseline)

Adaptive and non adaptive implementations have been generalized to use color information, modelling pixel statistics (mean and variance) for each of the considered channels.

Using color components should help obtain better foreground segmentation, as it shouldn't consider movement from illumination changes.

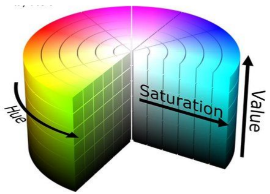
RGB



R: Red Color
G: Green component
B: Blue Component

All channels contain chroma and lightness information

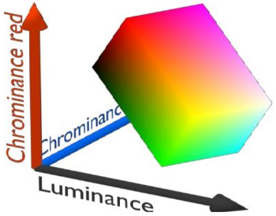
HSV



H: Hue (Dominant Wavelength)
S: Saturation (color shade)
V: Value (Intensity)

Chroma (contained in H) is independent of the light intensity

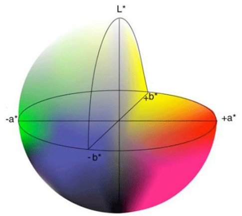
YUV / YCrCb



Y: Luminance
U: color component R - Y
V: color component B - Y

Chroma is independent of the light intensity *(both computed from RGB)*

LAB



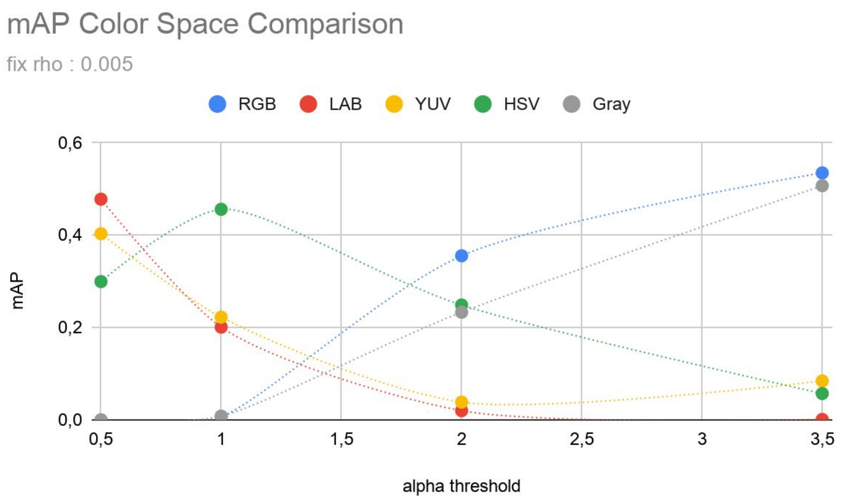
L : Lightness (Intensity)
A : color from Green to Magenta
B : color from Blue to Yellow

Chroma is independent of the light intensity

Task 4: Color (Team 2 2020 baseline)

Quantitative comparison

- Parameters **alpha**, **rho** and postprocessing filters vary depending on the color space.
- *LAB* and *YUV* obtain similar results, as expected due to their similarity. These two color spaces work best with smaller alpha values. However, as small values get a noisier segmentation, post processing is critical to obtain a good foreground estimation.
- *HSV* achieves best mAP with alphas around 1 but with the lower values compared to other spaces.
- *RGB* achieves the best mAP with high alpha values, but it is the most penalized in small values.



Best $AP_{0.5}$ = **0,5348** using RGB with 3 gaussians

	mAP	Precision	Recall	Gaussians
RGB	0,5329	0,8048	0,3474	3
HSV	0,4559	0,6869	0,3346	3
LAB	0,4778	0,6026	0,3672	3
YUV	0,4883	0,7180	0,3130	2

**Results obtained using our adaptive model and opening + closing postprocessing*

Scoring Rubric

Task	Description	Max. Score
T1.1	Gaussian. Implementation	2
T1.2	Gaussian. Discussion	1
T2.1	Adaptive modelling	2
T2.2	Adaptive vs non-adaptive models	1
T3	Comparison with the state of the art	2
T4	Colour sequences	2

Deliverables

- **Report on completed tasks by editing the GDrive slides.**
- **Code used for the week assignment**

- **17th March** (TODAY)
  - **Fill the intra-group evaluation form for Week 1.**
- **23rd March at 15h (Wednesday)**
  - **Put slides on Google Docs template**
  - **Fill the intra-group evaluation for Week 2 on this form**