Master in
Computer Vision
Barcelona

Project
Module 6
Coordination

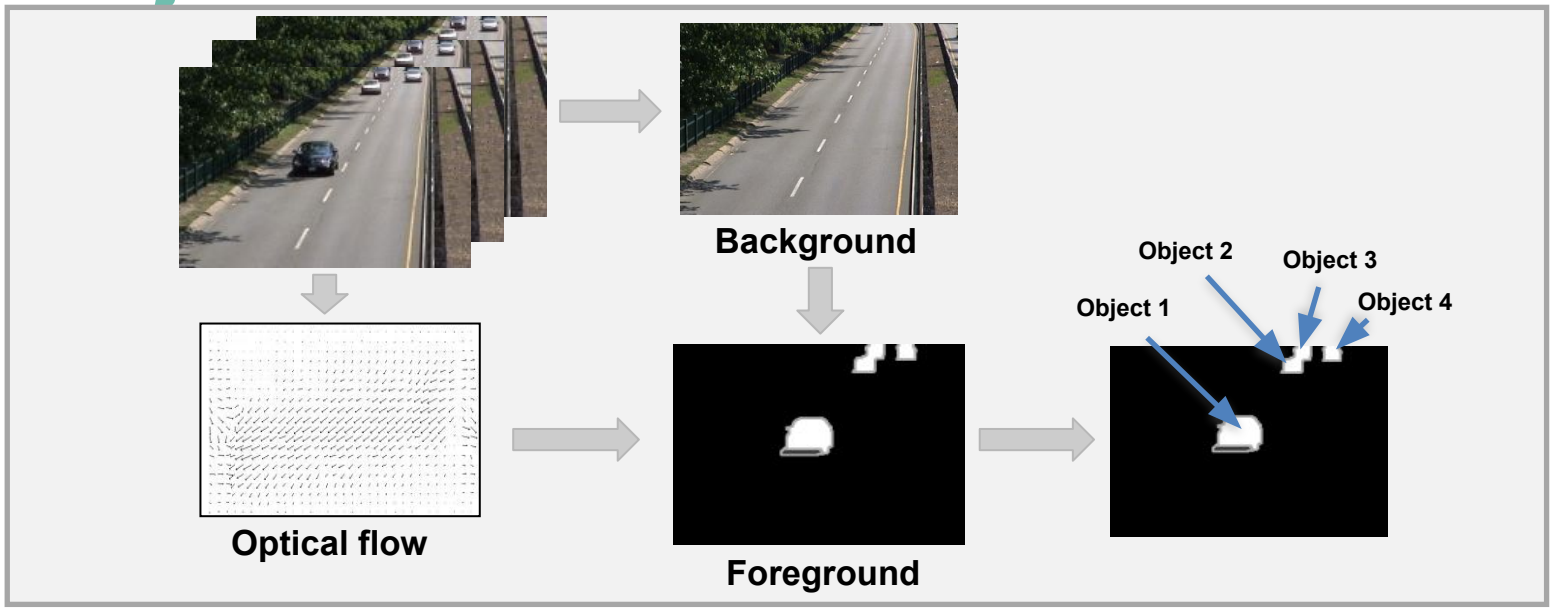
Video Surveillance for Road Traffic Monitoring

J. Ruiz-Hidalgo / X. Giró

Week 2: Review of submissions

j.ruiz@upc.edu / ramon.morros@upc.edu

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

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

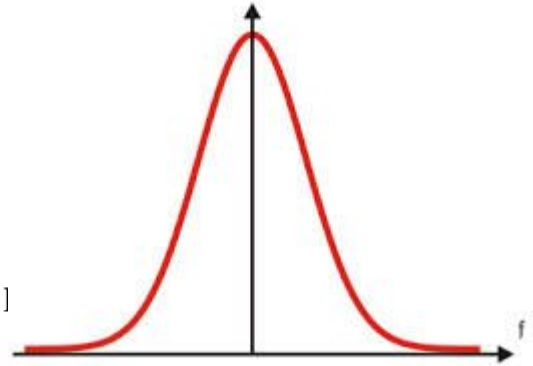
Teams & Repos (please declare any change)

Repo	Students
Team 1	Rachid Boukir, Josep Bravo, Alex Martín, Guillem Martinez, Miquel Romero
Team 2	Álvaro Budria, Alex Carrillo, Sergi Masip, Adrià Molina
Team 3	Albert Barreiro, Manel Guzmán, Jiaqiang Ye Zhu and Advait Dixit
Team 4	Julia Ariadna Blanco Arnaus, Marcos Muñoz González, Abel García Romera and Hicham El Muhandiz Aarab
Team 5	Razvan-Florin Apatean, Michell Vargas, Kyryl Dubovetskyi, Ayan Banerjee and Iñigo Auzmendi
Team 6	Guillem Capellera, Ana Harris, Johnny Núñez, Anna Oliveras

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

for all pixels i **do**
  if $|I_i - \mu_i| \geq \alpha \cdot (\sigma_i + 2)$ **then** ▷ +2 to prevent l
    pixel → Foreground
  else
    pixel → Background
  end if
end for

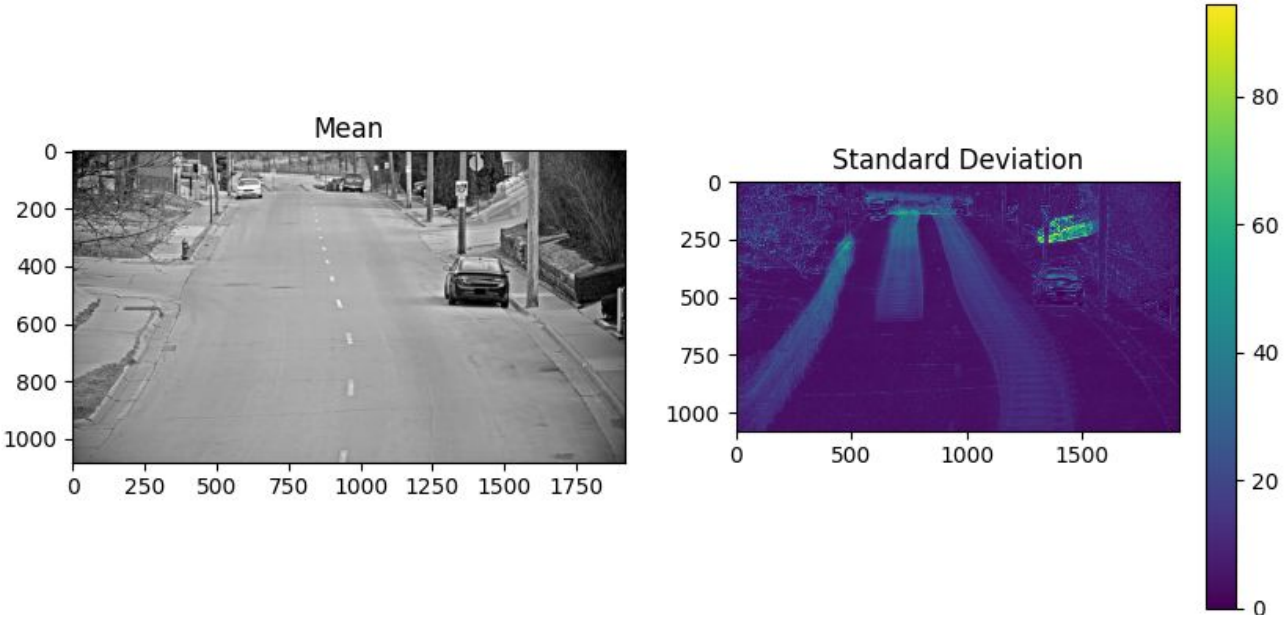


- **Second 75% to segment the foreground**
  - Group into objects → Bounding Boxes

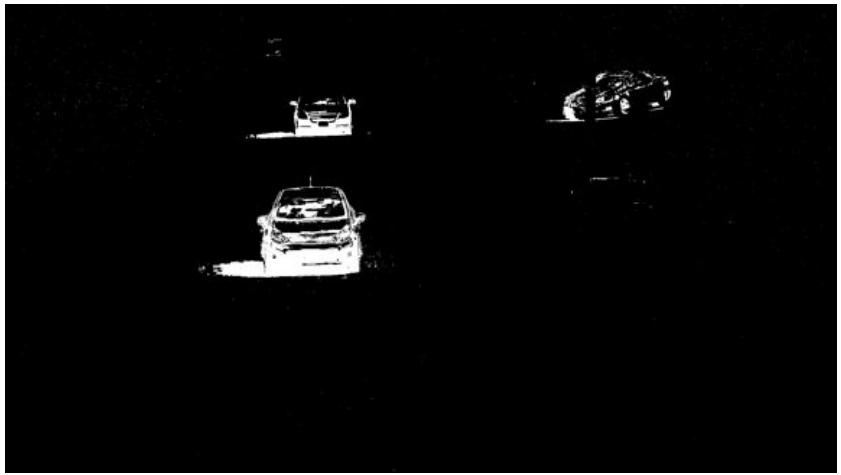
Task 1.1: Gaussian modelling (Team X) max 3 slides

Task 1.1: Gaussian modelling (Team 1) [1/2]

Using the first 25% frames of the sequence we build a gaussian model, obtaining a Mean image (the background) and the standard deviation which will be used to establish whether a pixel is foreground or not.



Task 1.1: Gaussian modelling (Team 1) [2/2]



Results with alpha=4

We can see how the model is very sensible to illumination changes and noise.

It would be necessary to find the best alpha parameter and enhance them with morphological operations.

Task 1.1: Gaussian modelling (Team 2 1/3)

In the plot we can see the mean and standard deviation of a pixel in the image at each time step.

Note how despite the significant fluctuations in the pixel value throughout much of the sequence, the mean value stays more or less constant, whereas the standard deviation increases a lot.

This means that the background model incorporates much of the variance coming from a car, not the background itself. We do not take care of this issue in this practicum, but it is very important to ensure that the mean and std. of a pixel are calculated with background only.

This issue could be mitigated by using a longer window to compute the statistics.

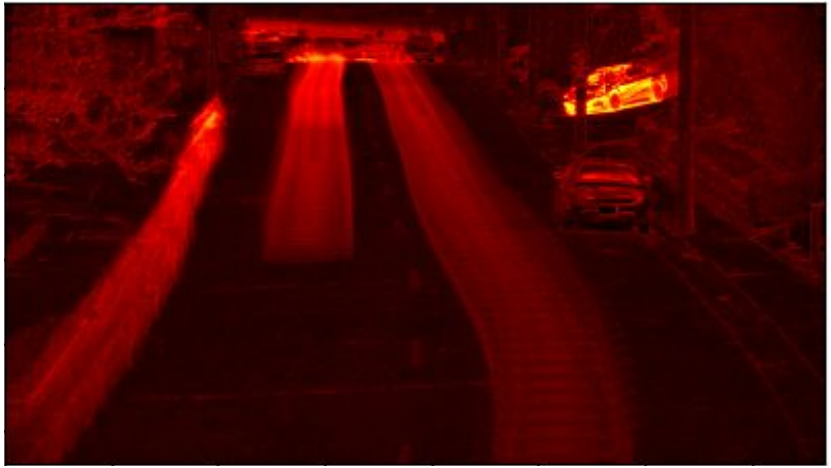
Task 1.1: Gaussian modelling (Team 2 2/3)

Notice how in the standard deviation map, the road and the intersection point (places where cars are present during the “training” phase of the method) have high variance values. This results in less robust background estimates, as the statistics from background and foreground are effectively mixed and blurred together.

Note also the ghost car in the mean map at the intersection point (upper left of the image). During the training phase, a car stops there for a few seconds, resulting in a mean pixel value that is different from the actual background. Again, this reduces the robustness of the method, as it confuses the statistics of foreground and background. This very same effect can be seen to a lesser extent in the darker lines on the road, corresponding to the passing of cars, whose color is incorporated into the mean.

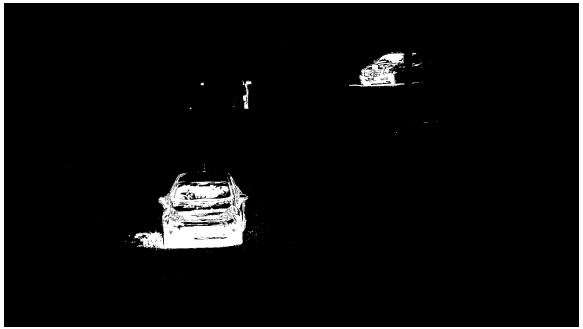
Per-pixel estimated mean

Per-pixel estimated standard deviation



Task 1.1: Gaussian modelling (Team 2 3/3)

Frame 546



Frame 1219

We do not evaluate our model on parked cars, because our model simply cannot detect them.

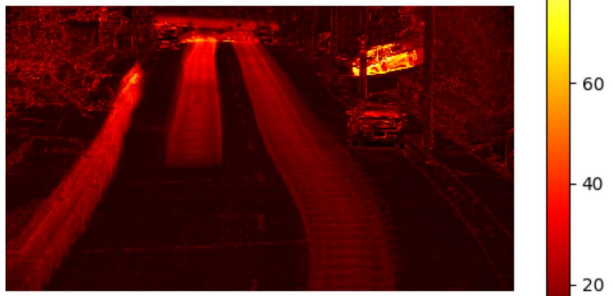
It is impossible for static cars to be considered as background by the simple Gaussian model (notice on the left figures how the parked car on the right is ignored in the segmentation), as it only takes into account the per-pixel mean and the standard deviation. The parked cars have very low std, so the model is robust there.

There is a fair a mount of noise in the non-filtered segmentation maps: salt-and pepper noise, holes in the masks, partially segmented objects, and temporally inconsistent segmentation.

This motivates the use of post-processing on the masks.

Task 1.1: Gaussian modelling (Team 3) 1/3

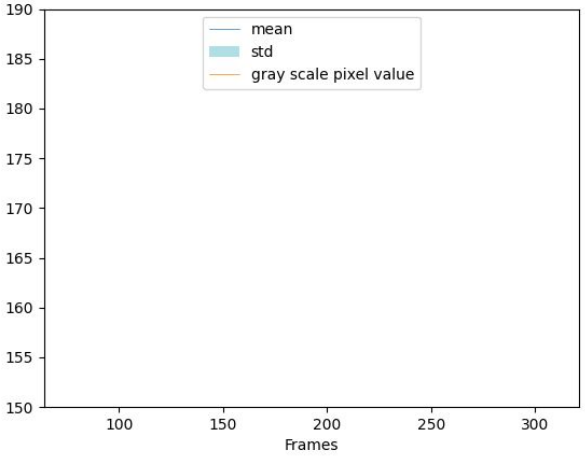
Mean



Variance

- The pixels with **higher** standard deviation are on the road because that's where there is **more movement** (traffic of cars).
- The model as seen is very **sensitive** to the **movement** of the cars and even the bicycles which greatly affects the performance of the model.
- We can also see that the mean image includes the **parked cars**.Their presence can make it **difficult** for the algorithm to accurately detect moving objects in subsequent frames since the parked cars can be mistakenly identified as part of the background. We also have **marks** of a car that stopped for a long time during the creation of the background model.

Task 1.1: Gaussian modelling (Team 3) 2/3



- As seen in the graph, the standard deviation and mean don't change much when the bicycle drives past the selected point, even though there is fluctuation in the grayscale values.
- But when the car passes away, the standard deviation changes very rapidly when the car drives by(enters the frame and exits after some time) but the mean is relatively stable

Task 1.1: Gaussian modelling (Team 3) 3/3

alpha = 4

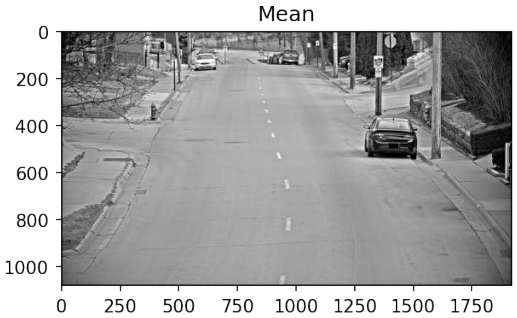
alpha = 6

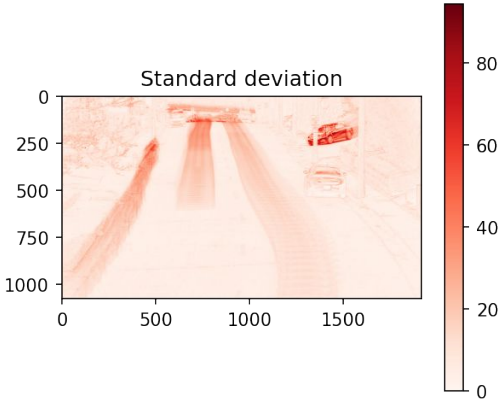
alpha = 10

- Alpha controls the **separation** between foreground and background. Large Alpha puts a higher **constraint** in the pixels to be considered foreground and Lower Alpha values get a **noisier** foreground and are prone to **false** detections
- Model is very **sensitive** to illumination and small differences, **difficult** to distinguish foreground from background. **Morphological** filters like closing greatly help to improve the accuracy of the model.
- **Bicycles** and **moving** cars appearing during training **affect** the performance of the model.

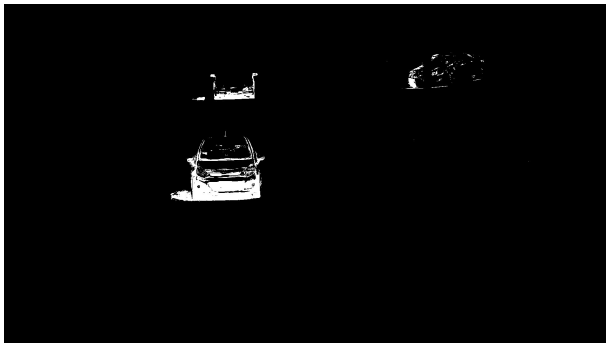
Task 1.1: Gaussian modelling (Team 4)

We use the first 25% of the frames to model the background using the mean and std:





Then we segment the foreground and background of the 75% remaining sequence:



Using alpha=5 as first initial value (visually selected, since in task 1.2 we will use mAP to choose the best one).

Task 1.1: Gaussian modelling (Team 5) [1/3]

We estimated the background using the first 25% of the frames. To avoid storing the first 25% of images, which would lead to excessive memory utilization, we computed the mean and standard deviation values using the single pass online Welford's algorithm.

Frame **Accumulated mean** **Accumulated std**

In the evolution, it can be seen how the moving object from the first 25% of the frames has significantly affected the final standard deviation. As the background in this task is not adaptive, it could lead to problems in detections.

Task 1.1: Gaussian modelling (Team 5) [2/3]

Final mean

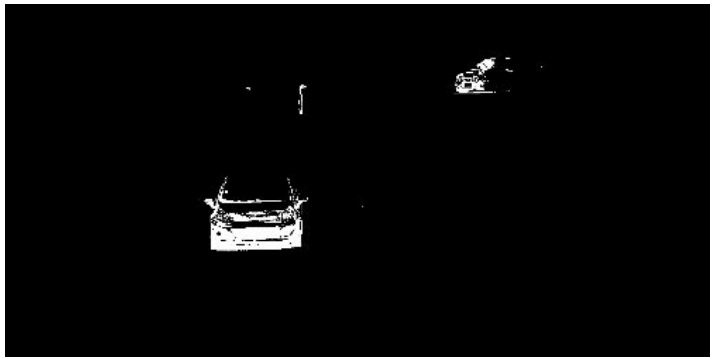
Final standard deviation

It can be seen that the moving object significantly influenced the final standard deviation of the background estimation. Although it may not be as noticeable in the final mean as it is in the standard deviation, artifacts such as transparent cars may appear due to the large number of frames at the end with the stationary car.

Task 1.1: Gaussian modelling (Team 5) [3/3]

Frame

Foreground Estimation (alpha = 10)



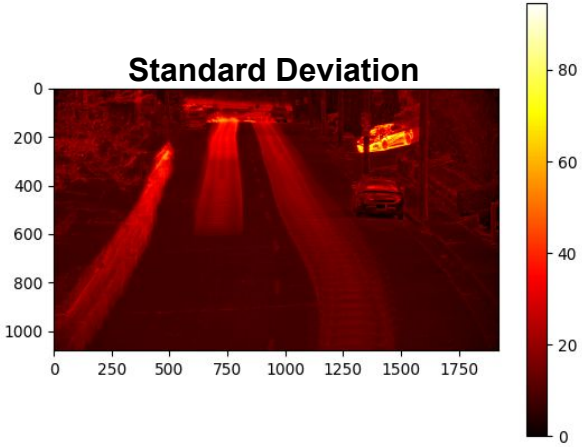
We can estimate the foreground pixels in the remaining video frames by using the final mean and standard deviation values that were estimated from the first 25% of the video frames.

Task 1.1: Gaussian modelling (Team 6) [1/3]

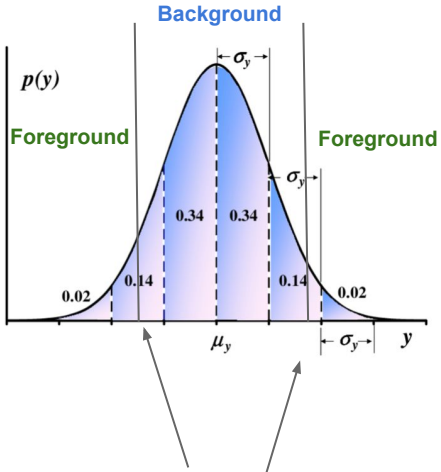
Foreground subtraction is an important technique in video processing that involves separating the foreground of a video sequence from the background. To achieve this, we employ the use of a Gaussian modelling algorithm.

Mean

Standard Deviation



After pre-processing a subset of frames from the video sequence (25%), we can model the Gaussian distribution for each pixel. As a result, each pixel will have an associated Gaussian distribution that can be used to subtract the foreground from the video sequence.



Permissivity regulate how far the pixel intensity can be from the mean considering the std

The Gaussian modelling algorithm assumes that the gray or color values in a video sequence follow a Gaussian distribution, which is defined by a mean and a standard deviation (std). By pre-processing a subset of frames from the sequence, we can model this Gaussian distribution for each pixel.

pixel value

standard deviation

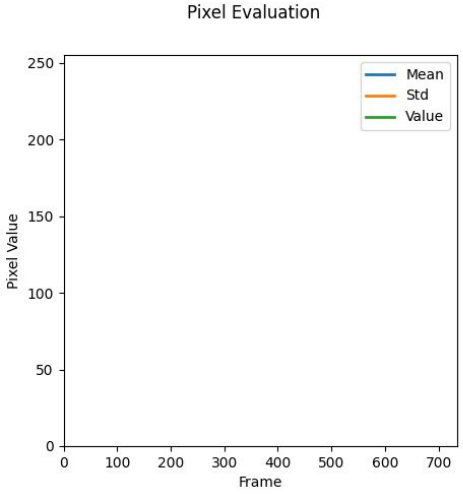
$$| I_i - u_i | < \alpha(\sigma_i + 2) \quad \text{Background}$$

$$| I_i - u_i | \geq \alpha(\sigma_i + 2) \quad \text{Foreground}$$

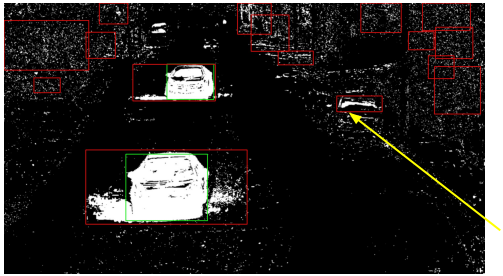
mean of pixel i at the 25% video sequence

permissivity

Task 1.1: Gaussian modelling (Team 6) [2/3]



There is a slight movement in the camera that causes a large movement in the pixel value. When vehicles appear, pixel gray mean oscillates and the std increases



Parked cars issue

We decided not to evaluate our model on parked cars since we will not detect them. In fact, their std is very low and they will not be detected by the Gaussian modelling.
In addition, it is application-dependent, If we want to detected moving cars, parked cars can be considered as background. If you look for stationary cars, it should be considered as foreground.

In some cases, parked/static cars may also cause shadows or reflections that can change the appearance of the background. This can be a problem for background estimation algorithms, as they may mistake the shadows/reflections for foreground objects. For example, there are some frames that appear with a lot of noisy.

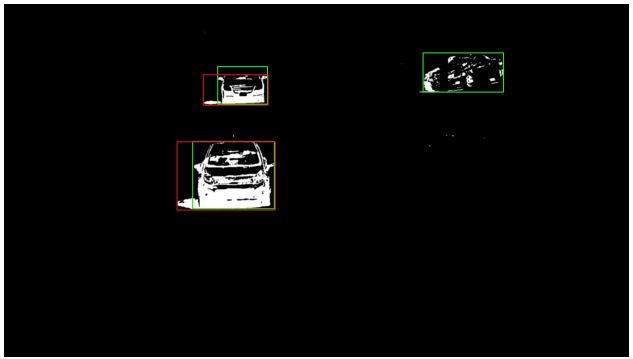
Task 1.1: Gaussian modelling (Team 6) [3/3]

Ground truth
Predicted

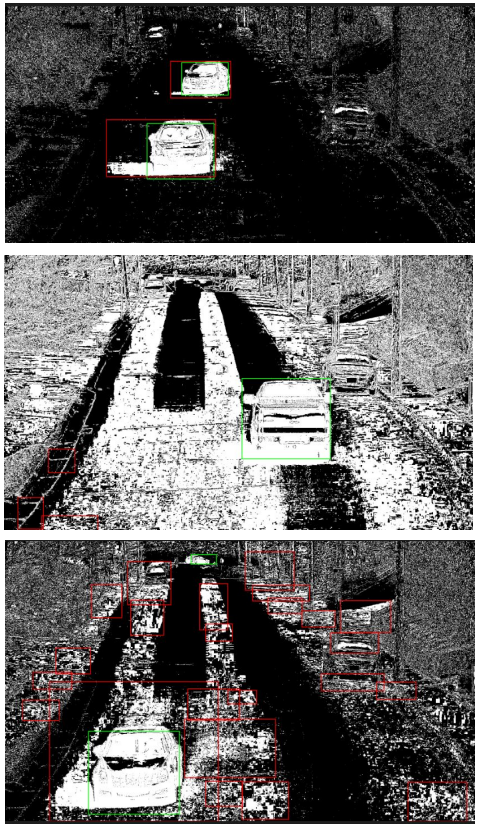
Once the background is estimated, we add some post-processing steps to generate a good object detection. In this case, cars in motion, as we do not consider parked cars.

- Noise filtering to the foreground mask. This consisted of a combination of a median blur and a gaussian blur filters, with kernels of 3 and 5 respectively.
- Find contours was used on the masks to obtain the bounding box detections for each frame.
- Bounding boxes were filtered based on the aspect ratio (removing too vertical boxes). Finally, we applied non-maximum suppression.

with post processing



without post processing and alpha = 3



Task 1.1: Feedback

	feedback
Team 1	Nice and clear instructions in github How did you select the alpha param? value? Is there mask post-processing? Not mentioned. Missing explanation of detecting BB Too simple analysis
Team 2	Nice instructions in github. Post-processing mentioned but not explained. Missing explanation of detecting BB Good analysis
Team 3	Informative Github readme file, but lacking instructions on how to run the script. Bonus point for the plot of the evolution of the std Post-processing mentioned but not explained. Missing explanation of detecting BB Good analysis
Team 4	Nice and clear notebooks in github. Why is the std image smaller? Missing discussion on the resulting mean/std background. Missing explanation of detecting BB
Team 5	Nice and clear instructions in github. Bonus point for computing std on one pass (Welford's algorithm)

Task 1.2: Evaluation

- **Evaluate Task 1**
  - mAP on detected connected components
  - Filter noise and group in objects
  - Over alpha threshold
  - Decide (and explain) if parked/static cars are considered

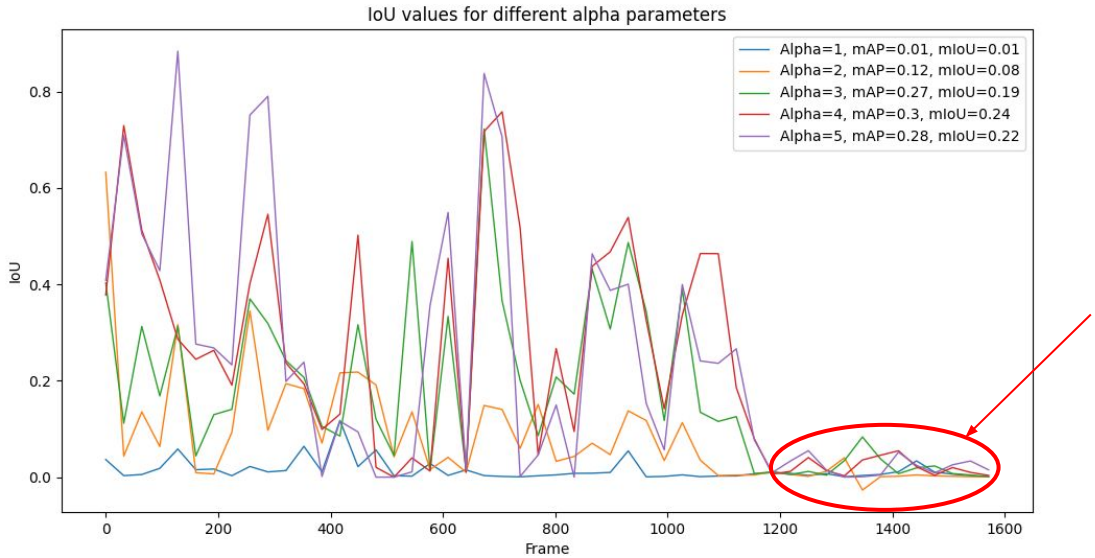
Task 1.2: $AP_{0.5}$ vs α (All teams)

Team ID	$AP_{0.5}$	α	Others
Team 1	0.3	4	
Team 2	.2338	3	
Team 3	0.32	10	
Team 4	0.22	8	
Team 5	0.235	8	Morphological filtering and removal of components that are too big or too small (explained in slide 33 text).
Team 6	0.405	5	

Task 1.2: mAP vs alpha (Team X) - max 3 slides

Task 1.2: mAP vs alpha (Team 1) - [1/3]

A search through the alpha parameter has been executed in order to select the one with the best mAP. Additionally, morphological operations (opening and closing) have been considered for the enhancement of the subtraction.

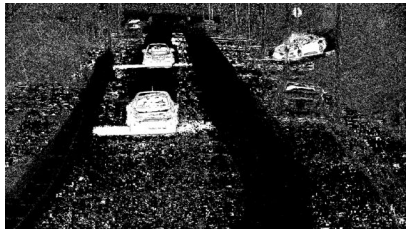


Data points have been interpolated for the sake of visualization

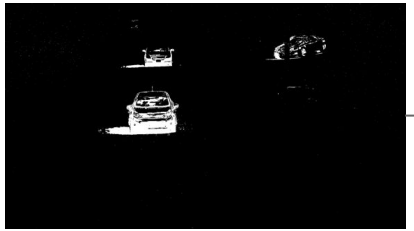
The best performing alpha found has been 4, with an mAP of 0.3 and an mIoU of 0.24.

It is visible how those parts of the sequence that are very distant of the mean because of **illumination changes or noise** have very poor IoU as this model is **not capable of subtracting the background in those conditions**.

Task 1.2: mAP vs alpha (Team 1) - [2/3]

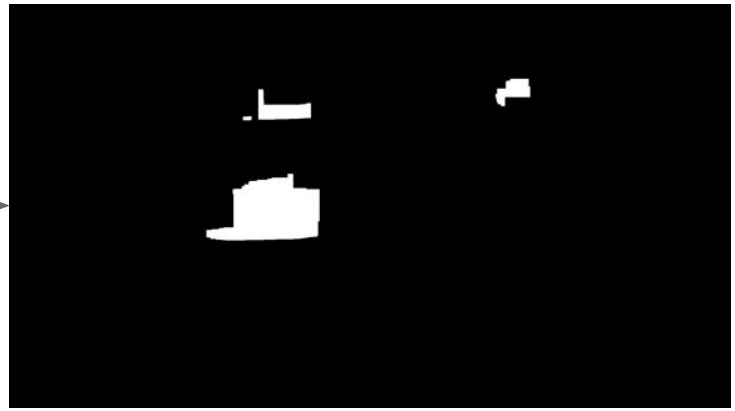


Alpha=1

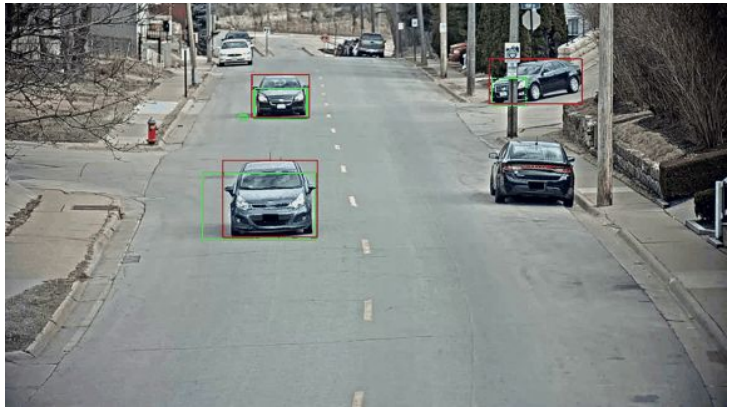


Alpha=4

Morphological enchancement



Alpha=4,
mAP=0.3,
mIoU=0.24



This model, with the optimal alpha, **still struggles in the same situations previously mentioned**.

As it is visible, **parked cars were not considered** as they are part of the background.

Task 1.2: mAP vs alpha (Team 1) - [3/3]

As it was visible in the previous chart, if we visualize the last frames which seem to be very badly detected, we see that **a change of illumination produces such results**.

Detections and masks from the last 400 frames

Task 1.2: mAP vs alpha (Team 2 1/3)

Postprocessing for noise removal

We apply several noise removal postprocessing techniques to mitigate the different types of noise we identified in the segmentation masks.

- **holes in the masks** and **partially segmented objects** → take the bounding box that contains the whole connected component, and eliminate overlapping bounding boxes (non-maxima suppression, NMS)

- **salt-and pepper noise** → morphological filtering: opening and closing. Removes artifacts smaller than the structuring element (6x6)

- **temporally inconsistent segmentation** → impose a temporal smoothness constraint, since the motion in consecutive frames is small: remove those bounding boxes that do not have high overlap (>50%) with either the previous nor the next frames.

In the following table, noise removal techniques are incrementally added to the system, so as to assess their effectivenes.

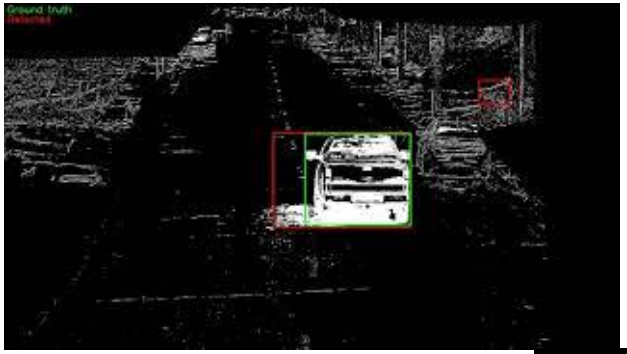
	BBox whole component	N.M.S.	Morphological filtering	Temporal consistency
AP	.1924	.2009	.2222	.2293
it/s	~10.5	~10	~8.5	~8.5 it/s + 0.15 s

All postprocessing techniques are applied frame-wise, except for the temporal prior one, which is applied after all frames have been processed.

* $\alpha = 4$.
* Speed is subject to hardware specifications

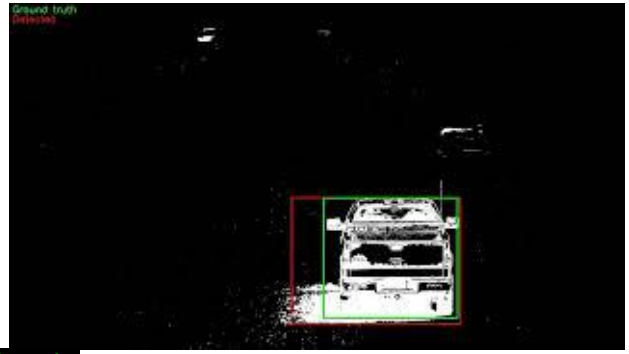
There is a tradeoff between the improvement in AP and the computational cost of the postprocessing. The most effective techniques are morphological filtering followed by NMS.

Task 1.2: mAP vs alpha (Team 2 2/3)



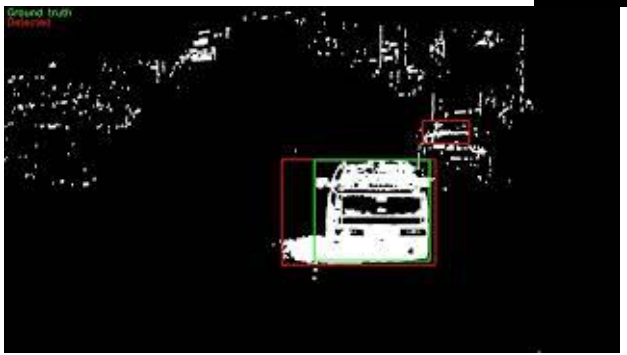
non-filtered

ground truth
detection

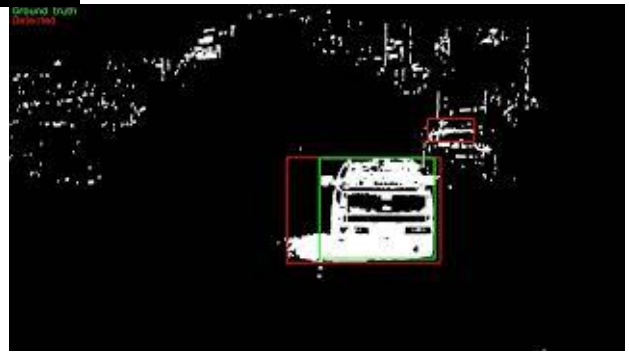


* $\alpha = 4$

NMS



morphological filtering



temporal filtering

Additive effect of post-filtering techniques - Qualitative results

The non-filtered sequence contains a highly noisy segmentation, as well as a huge number of overlapping detected boxes.

The non-maximum suppression (NMS) reduces the number of detected boxes to make it more coherent with the scene.

The morphological filtering succesfully gets rid of much of the small irregular noise. It also homogenizes the segmentation masks for the cars.

Finally, the temporal filtering adds consistency to the predicted boxes along the remporal axis. It is harder to see, but in some frames some of the detections are adequately discarded.

In passing, we remark in some frames the illumination changes. This might be due to clouds in the sky or camera noise. In those frames, the background deviates from its mean value, so we get a lot of noise in the background areas. A preprocessing of the frames to make them homogeneous in illumination along the time axis could further improve results.

Task 1.2: mAP vs alpha (Team 2 3/3)

Alpha	Precision	Recall	F1-score	AP	IoU
1	.0221	.0576	.0319	.0152	.0346
2	.1392	.3016	.1905	.1466	.1327
3	.1995	**.3659**	.2582	**.2338**	.1645
4	.2209	.3028	.2554	.2293	.1999
5	.3108	.2765	.2927	.2128	.2898
7	**.5305**	.2186	**.3096**	.2015	**.5093**
9	.4426	.1369	.2091	.1311	.4690

Note that because higher α leads to less detected pixels, recall is lower, but precision is higher (with a peak at α = 7). However, for these particular dataset and method, it seems recall aligns better with AP.

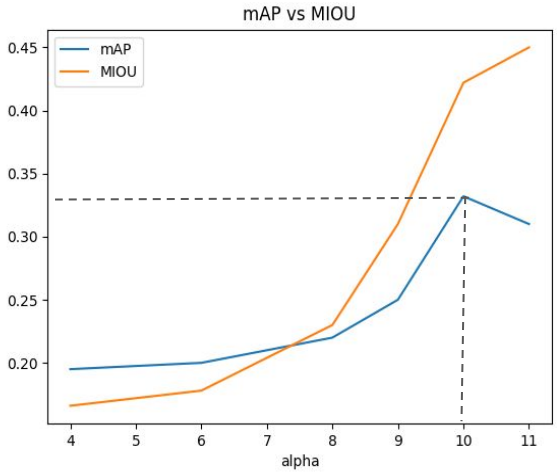
Qualitative evaluation of the effect of α = 1, 2, 4, 8

When **α** is **lower**, the threshold for considering a pixel as foreground is highly non-restrictive, resulting in many background pixels being classified as foreground, which visually looks like noisy clouds in the segmentation map.

When **α** is **higher**, less pixels are classified as foreground. However, there is a tradeoff. If we increase α too much, very few pixels are then classified as foreground, resulting in only partial segmentations of the cars. We note that for high α, even when detecting few pixels as foreground, the bounding boxes are not too bad, because we take the bounding box enclosing a whole connected component in the segmentation mask.

Task 1.2: Gaussian modelling (Team 3) 1/2

Alpha	Precisión	Recall	mAP
4	0.45	0.16	0.195
6	0.45	0.17	0.201
8	0.48	0.18	0.22
10	0.60	0.26	0.32
11	0.57	0.25	0.30

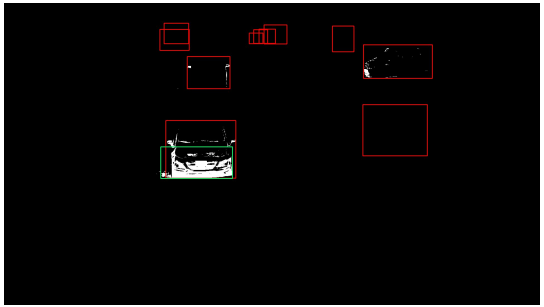


As we can see, increasing alpha may improve the quality of the predicted bounding boxes in terms of IoU, but it can also negatively impact the overall performance of the detection in terms of mAP. One of the reasons is that IOU does not take into account false detections.

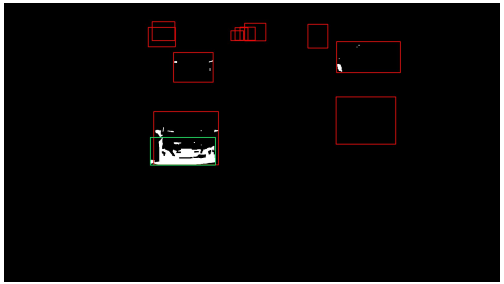
Task 1.2: Gaussian modelling (Team 3) 2/2

Ground truth

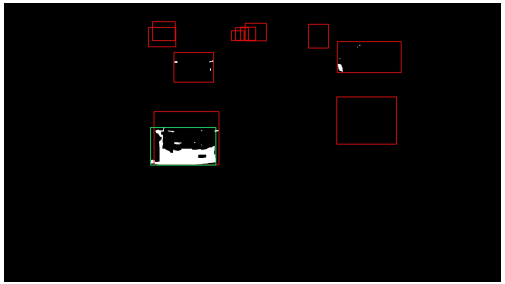
Prediction



No filtering, alpha =10, map= 0.21



Closing(5x5), alpha =10, map= 0.23

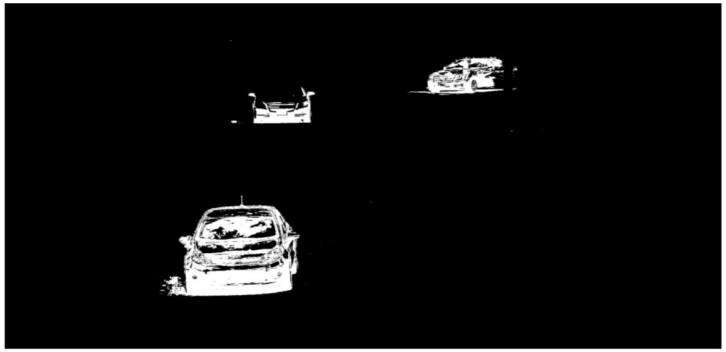


Closing(9x9), alpha =10, map= 0.32

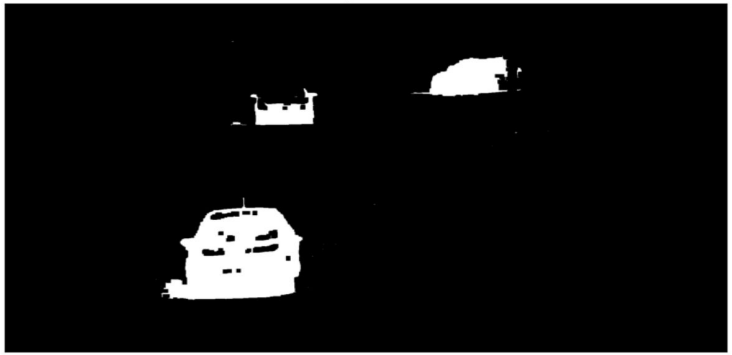
- Applying a **morphological filter** like **closing** greatly helps to increase the accuracy of the model. As shown in the above images, there is a 2 percent increase in the mAP value for the same alpha when we apply the filter.
- Further, the **size of the filter** also makes a big **impact** on the **performance** of the model. **Increasing** the filter size helps the model to perform better and there is jump of 9 percent in the mAP.
- The model is **not** able to detect **parked** cars as it considers it to be a part of the **background** and cars that are very far in the scene.
- We can further improve results by using **adaptive** Gaussian filters.

Task 1.2: mAP vs alpha (Team 4) [1/2]

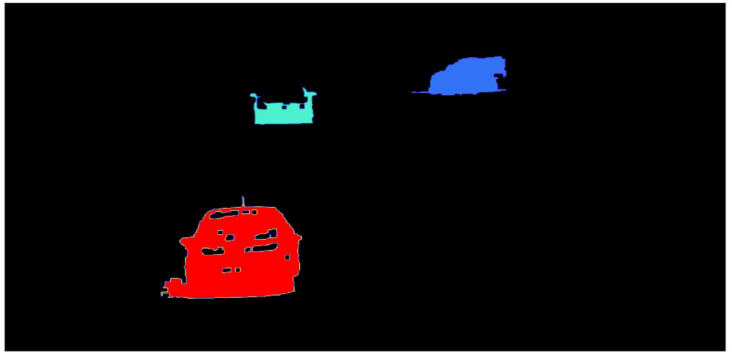
1.Segmenting foreground with the 1.1 approach:



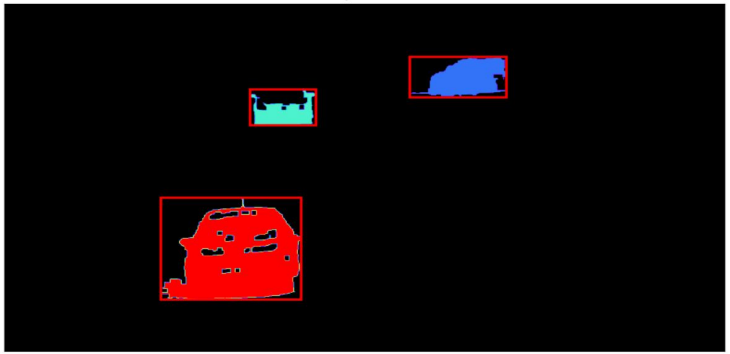
2. Improving the result using morphology



3. Getting the connected components:



4. Obtaining the bboxes of the connected regions:

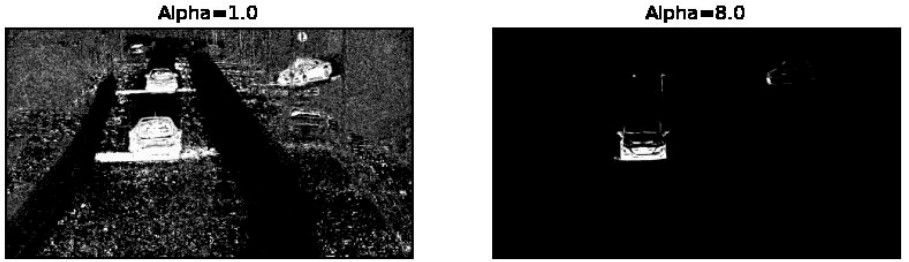


Task 1.2: mAP vs alpha (Team 4) [2/2]

We perform the same 4 steps for each frame of the video and then we compute the mAP.

We do it with several values of alpha to determine the optimal one in terms of mAP:

alpha	mAP
1.0	0.0
5.0	0.1339
8.0	0.2192
10.0	0.1599
15.0	0.0909



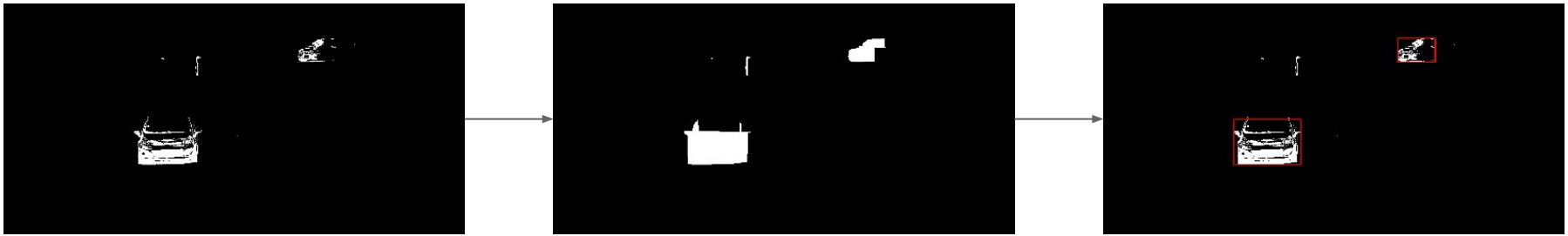
A visual comparison between the worst and best alphas (in terms of mAP)

Task 1.2: mAP vs alpha (Team 5) - [1/3]

Ground truth box

Detected box

Estimate the foreground **Filter noise and aggroup** **Obtain bboxes**



- From the annotations we removed the parked cars, because with the foreground we can not detect those cars (are part of the background).
- We used morphological filters to filter the foreground.
- We used Optuna to perform a probabilistic parameter search to find the optimal values for the filter size, alpha value, and minimum component area. The mean average precision at 50% (mAP50) was used as the evaluation criterion.
- Bounding boxes were obtained by identifying the connected components, and components that were too small or too large were removed.

Task 1.2: mAP vs alpha (Team 5) - [2/3]

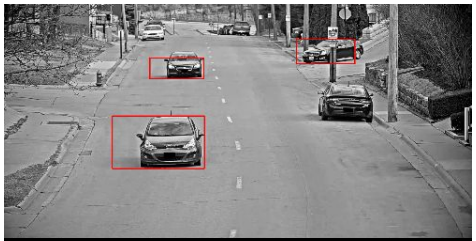
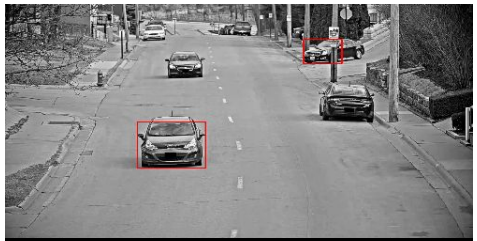
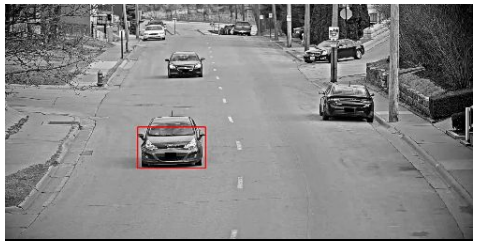
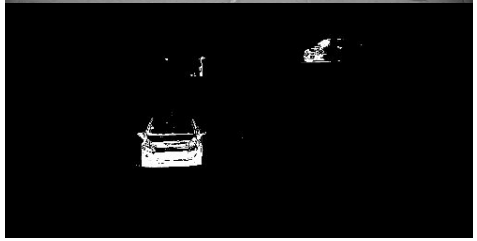
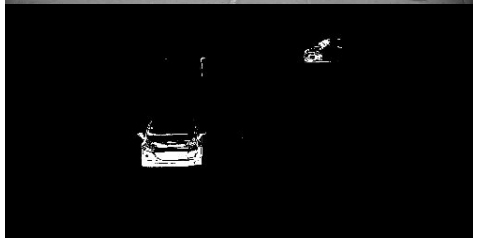
Alpha	mAP50	mIoU
2	0.026	0.06
4	0.128	0.147
6	0.156	0.171
8	**0.235**	**0.172**
10	0.163	0.137
12	0.145	0.109
14	0.077	0.084

We achieved the best mAP results using an alpha value of 8. Post-processing methods were found to have a significant impact on performance, so the same post-processing was used in all these experiments to maintain consistency.

Task 1.2: mAP vs alpha (Team 5) - [3/3]

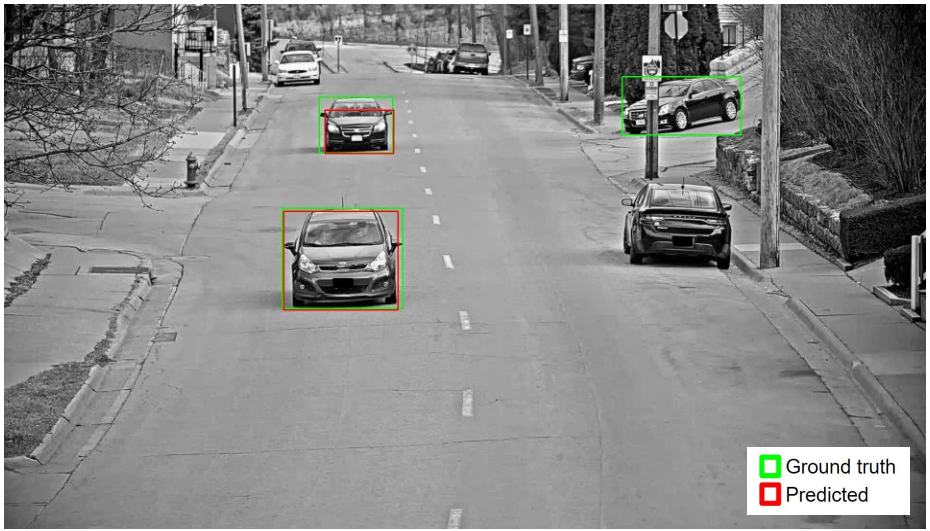
Ground truth box

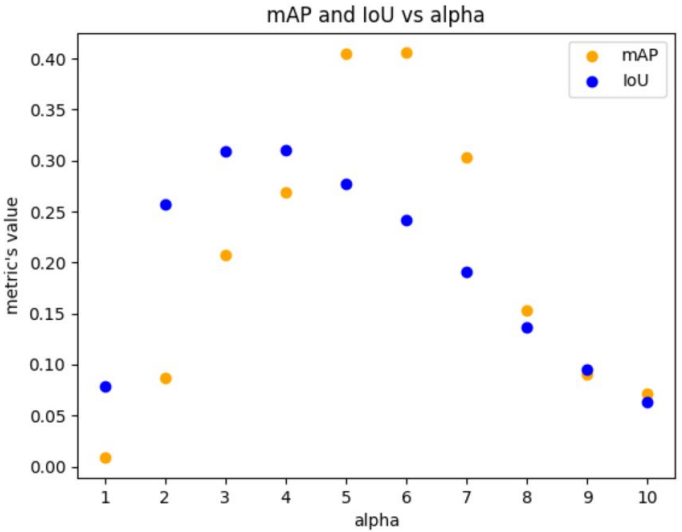
Detected box

	alpha = 4	alpha = 8	alpha = 12
BBox detection			
Foreground estimation			

- With alpha low values (as 4), there was too much noise and we could not detect the objects correctly, often even detecting the noise as object.
- Using alpha high values (as 12), the noise was less, but often the object pixels were also taken as background.
- Our best alpha value was 8. We did not obtain very good results, because in order to remove the noise, we had to select a high alpha and remove the small object (not detecting the cars that are far-away). This could be improved with an adaptive background.

Task 1.2: mAP vs alpha (Team 6) [1/2]





alpha	mAP	IoU
1	0.008	0.079
2	0.087	0.257
3	0.208	0.309
4	0.270	0.310
5	**0.405**	**0.278**
6	0.406	0.242
7	0.303	0.191
8	0.153	0.136
9	0.091	0.096
10	0.072	0.063

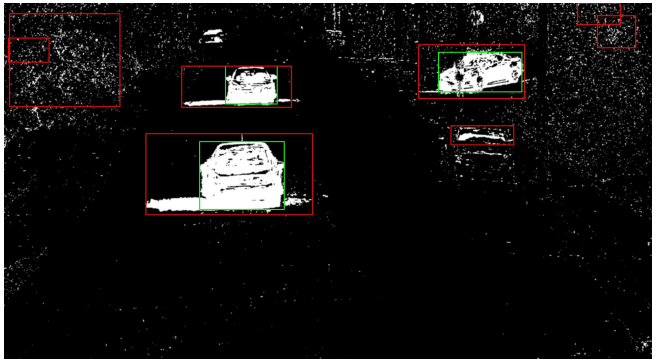
Very low alpha values turn out to be very sensitive to image noise generating lower scores. On the other hand, if we increase the alpha value too much, we lose information. The **optimum** value obtained is an **alpha value of 5**, since with alphas 5 and 6 we get the best mAP, but the IoU is much better with alpha 5.

Task 1.2: mAP vs alpha (Team 6) [2/2]

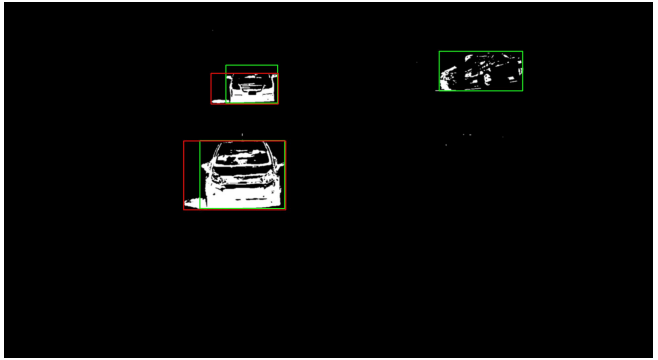
Ground truth
Predicted

- We can see that by increasing the alpha some elements of the cars are not able to be modelled as foreground.
- On the other hand, much more noise, and therefore more elements considered as foreground, is obtained by decreasing the alpha value.
- Even with the optimum value of alpha (=5), we are not able to detect some cars far away from the camera.

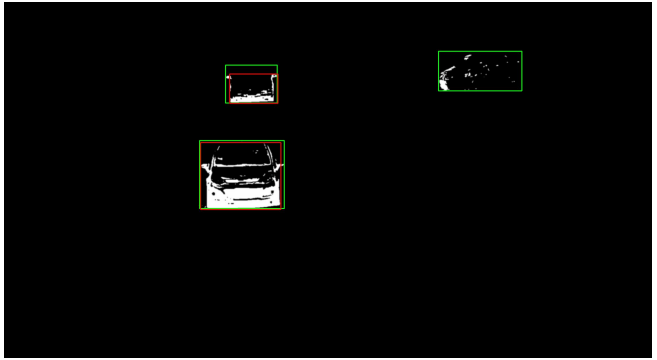
alpha = 1



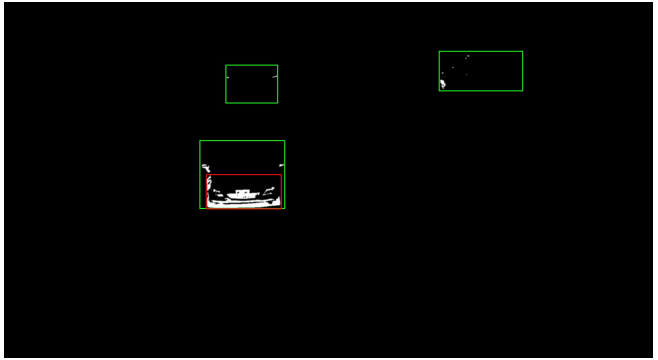
alpha = 3



alpha = 5



alpha = 10



Task 1.2: Feedback

	feedback
Team 1	Exhaustive search for alpha. Good! But only in development set. Beware of overfitting. Figure with values of IoU vs. alpha. Good. Missing details in morphology operators used. Which ones did you use?
Team 2	Nice post-processing with NMS, morphological filtering and temporal smoothing. Missing details in morphology kernels used. Table (why not a plot?) with values of IoU and mAP vs. alpha. Very good Good analysis (alpha, post-processing, etc.)
Team 3	IOU does not take into account false detections?? Table (why not a plot?) with values of mAP, P, R vs. alpha. Good Details in morphology techniques and kernels used are given. Good
Team 4	Good slide explaining results but missing details in morphology operators used. Which ones did you use? Table with mAP/alpha, why not a plot? 0.22 mAP(alpha=8) Not clear what do you do with parked parks in the ground truth?
Team 5	Remove car park from annotations what is optuna? (missing reference) Missing details in morphological operators (which one did you use?) Table (why not plot) 0.23 mAP (alpha=8)
Team 6	parked cars? Bonus point for deciding mAP also with IoU. Plot and values shown

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**
  - 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 which is best…).

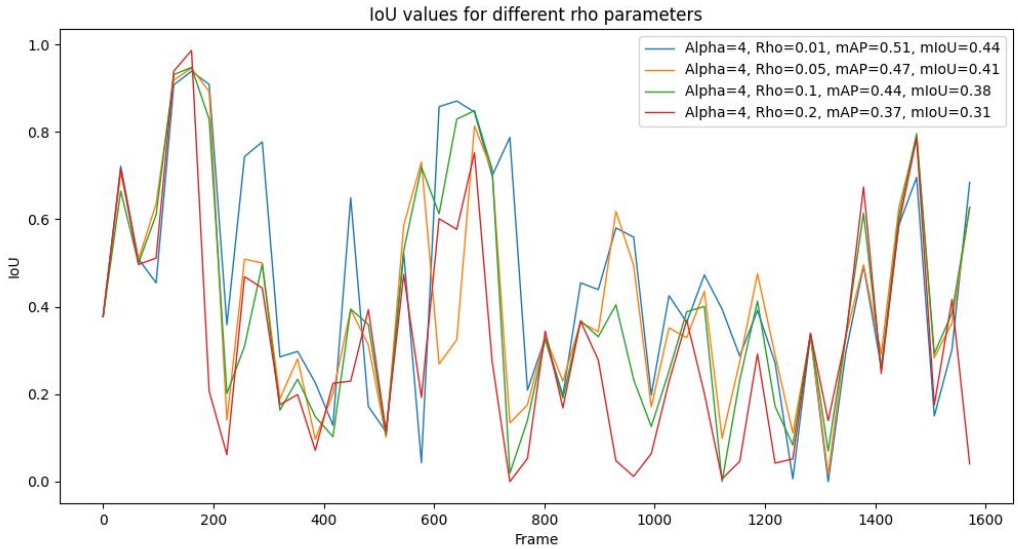
Task 2.1: Adaptive modelling (All teams)

Team ID	$AP_{0.5}$	α	ρ
Team 1	0.64	3	0.05
Team 2	0.40	4	0.02
Team 3	0.5852	3.3	0.043
Team 4	0.34	4	0.1
Team 5	0.52	4	0.0175
Team 6	0.784	4	0.025

Task 2.1: Adaptive modelling (Team X) - max 3 slides

Task 2.1: Adaptive modelling (Team 1) - [1/3]

We first searched the optimal rho value for the optimal alpha found in the previous task.
We found out that in general small rho values performed better.



Data points have been interpolated for the sake of visualization

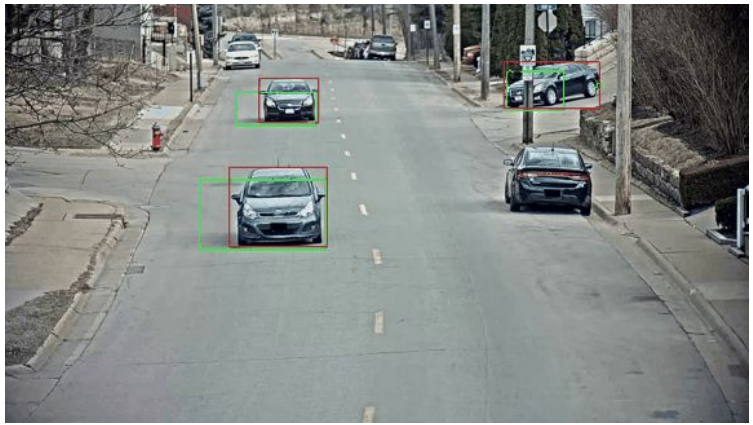
Every single combination tried performed better than the non-adaptive modelling with mAP=0.3

In general all combinations were pretty similar, with rho=0.01 a mAP of 0.51 and mIoU of 0.44 was achieved.

Nonetheless, **the chart is still very spiky** which proves that the model is still not perfect, as it is still **sensible to sudden changes**.

Task 2.1: Adaptive modelling (Team 1) - [2/2]

When using the second method to search the optimal alpha and rho value combined we found that an alpha=3 instead of 4, which was the optimal at non-adaptive, performs better.

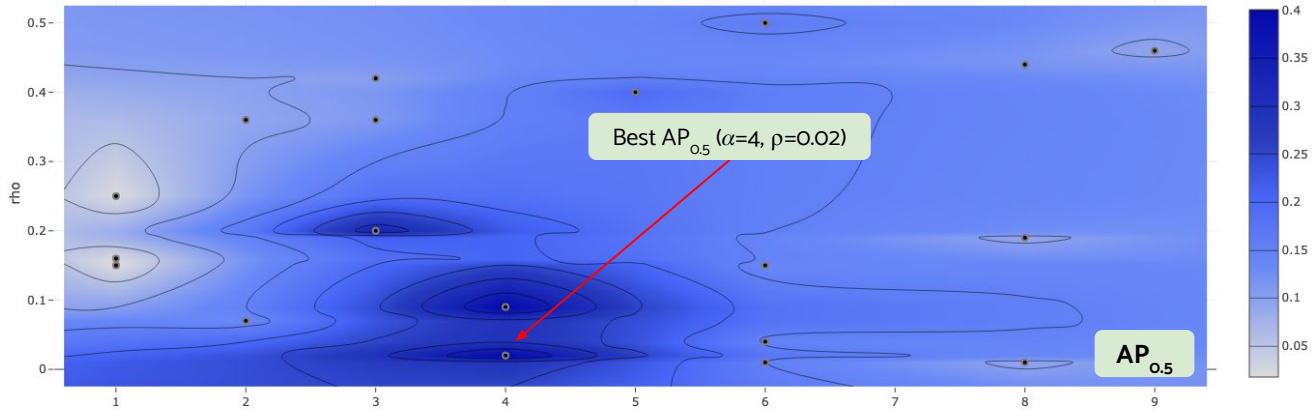


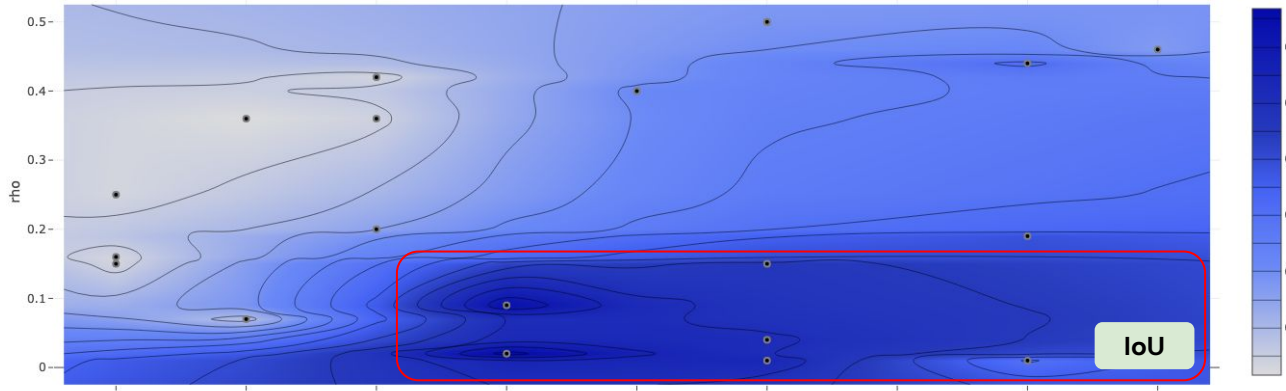
Alpha=3, Rho=0.05, mAP=0.64, mIoU=0.5

We find that **occlusions between moving objects is problematic** as it is impossible to detect them as two different objects.

Task 2.1: Adaptive modelling (Team 2) - [1/2]

For the adaptive modelling, we use Optuna's hyperparameter optimization framework to optimize (α, ρ) together, with multi-objective optimization to maximize the 5 metrics at the same time: precision, recall, F1, $AP_{0.5}$ and IoU. After some preliminar tests, we noted that higher values for ρ (i.e. between 0'5 to 1) resulted in a bad performance among all the metrics, so we further investigated a more promising and smaller range between 0 and 0'5 for ρ.





Importance	α	ρ
Precision	.53	.47
Recall	.84	.16
F1	.76	.24
$AP_{0.5}$	.62	.38
IoU	.59	.41

Among all the metrics, the α parameter is the most important one contributing to the performance of the adaptive model. In terms of AP, an α=4 similar to the best non-adaptive case (α=3) yields best results, combined with a very small ρ=0'02. In terms of IoU performance, the choice of α is slightly wider (between 4 and 8), while also keeping a small ρ to get good results.

Task 2.1: Adaptive modelling (Team 2) - [2/2]

Alpha	Rho	Time [s]	Precision	Recall	F1-score	$AP_{0.5}$ ↓	IoU
4	**0.02**	226	.8253	.4731	.6014	**.4006**	.6680
4	0.09	271	.8356	.4574	.5912	**.3989**	.6383
3	0.20	314	.1878	.5325	.2776	**.3255**	.1496
5	0.40	247	.2456	.3889	.3011	**.1937**	.2111
6	0.50	243	.2023	.3347	.2522	**.1589**	.1830
2	0.07	285	.0776	.3316	.1258	**.1258**	.0798
3	0.42	287	.0245	.024	.0419	**.1003**	.0330

In all the adaptive modelling experiments, the following **post-filtering techniques are applied**:

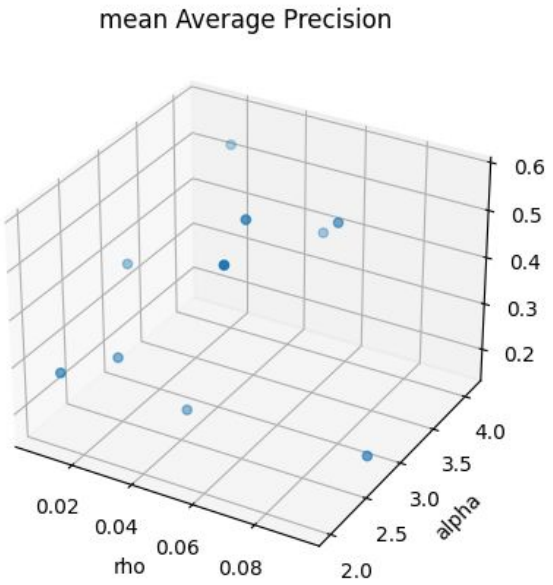
- Non-Maximum Suppression (NMS)
- Morphological filtering
- Temporal filtering

We show here a summary of best and worst results from the Optuna study with +30 trials. Overall, the best performance in terms of a balance between all the metrics is achieved with an α between 3 and 4 and a small value of ρ no higher than 0'2. Note that the best configuration (α=4, ρ=0.02) achieves not only the best AP, but also the best F1-score and IoU. We also observe in the last two rows that a change in the α (the param with highest importance) really influences the performance, degrading almost all metrics when using α=[2, 3] and similar ρ=[0'07, 0'42].

Task 2.1: Adaptive modelling(Team 3) 1/2

In order to get the best adaptive model we have perform a random search. We have taken that because Random Search is a more efficient approach in most cases, especially when the hyperparameter space is large and the importance of each hyperparameter is unknown.

The results are illustrated on the following figure



The plot show that majority of random pairs perform better than the non-adaptive algorithm.

The best result is obtained with

Alpha: 3.3
Rho: 0.043

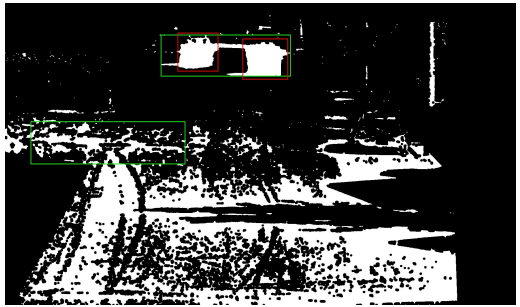
That lead a mAp = 0.5852

Task 2.1: Adaptive modelling(Team 3) 2/2

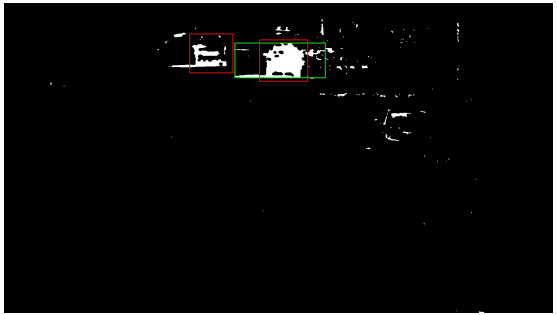
The following illustrations shows some examples of the results processed with the random search.

Ground truth

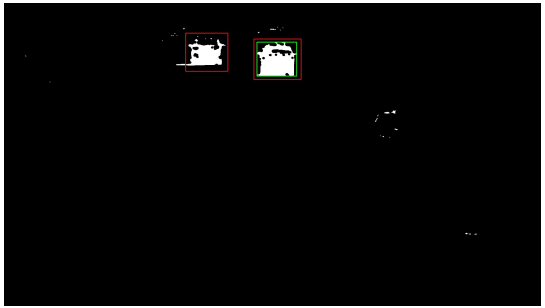
Prediction



alpha 2.5 rho 0.0029
mAP=0.1198



alpha 3.3 rho 0.0043
mAp = 0.5812



alpha: 3.3, rho: 0.072,
mAP: 0.5714

Task 2.1: Adaptive modelling (Team 4)

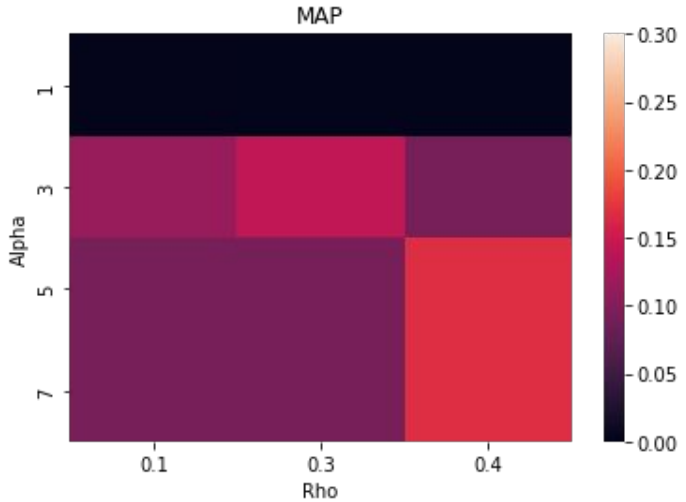
First of all, we determined which was the best combination of rho and alpha on the adaptative model case. Due to time constraints, we carried out a small grid search using the following values:

rho: 0.1, 0.3, 0.5
alpha: 1, 3, 5, 7

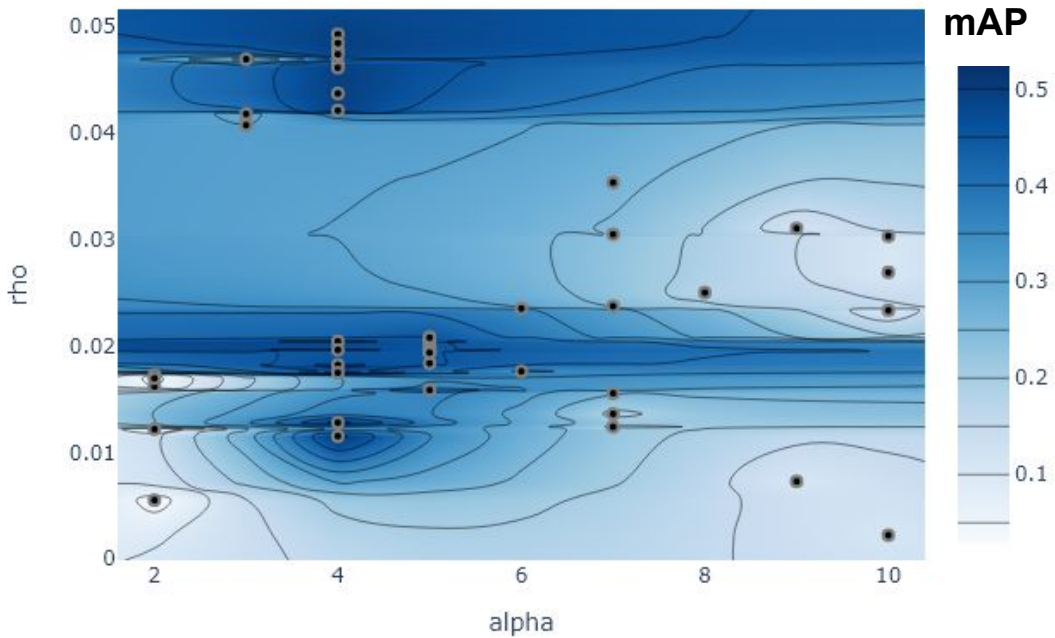
As we can see, the best combination has been rho = 0.4, alpha = 5.

It is worth mentioning that the grid search was done over the output without any kind of pre-processing, so it is quite possible that there exist better alpha and rho combinations given a specific kind of preprocessing.

	Parameters	**MAP**
Best MAP	rho =, alpha =	0.34
Worst MAP	rho =, alpha =	



Task 2.1: Adaptive modelling (Team 5) [1/2]



- In the adaptive model, we used the same morphological filters for post-processing, but we reduced the minimum area of the connected components to detect small, distant cars.
- We then conducted a probabilistic search (50 trials) using Optuna to determine the best alpha and rho values. The results indicated that the optimal alpha value was 4. Similarly, we found that values between 0.01 and 0.05 produced similar results for rho, with the best value being 0.0175.

Best combination

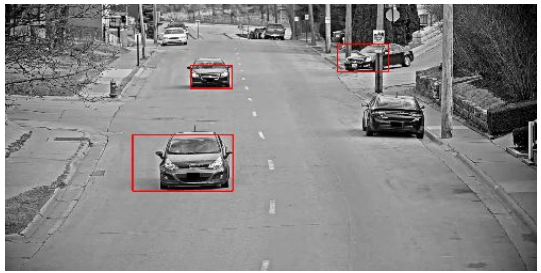
mAP	mIoU	rho	alpha
0.52	0.49	0.0175	4

Task 2.1: Adaptive modelling (Team 5) [2/2]

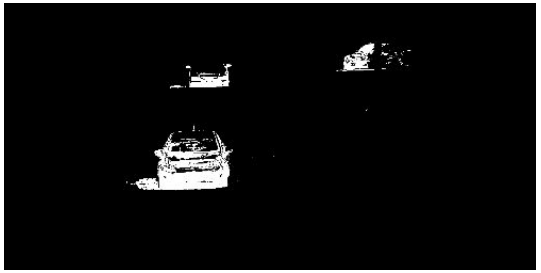
Ground truth box

Detected box

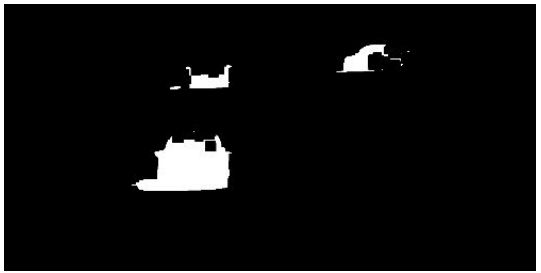
BBox detection



Foreground estimation



Filtered foreground



When examining the estimates and detections, it becomes apparent that this model significantly reduces noise. As a result, smaller alpha values and connected components can be used for bounding box detection, enabling the detection of even the smallest cars.

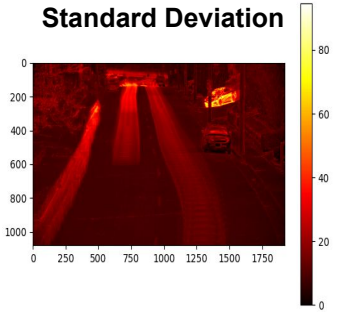
Task 2.1: Adaptive modelling (Team 6) [1/2]

We use the same pipeline as the Task 1 but here we change the way to model background. In Adaptive modelling the parameters **mean** and **standard deviation** computed using the 25% of the first frames, are updated recursively, according to the pixel value at each timestep.

Compute the mean and the standard deviation of the first 25% frames for training

Mean

Standard Deviation



pixel value

standard deviation

$$| I_i - u_i | < \alpha(\sigma_i + 2)$$ Background

$$| I_i - u_i | \geq \alpha(\sigma_i + 2)$$ Foreground

mean of pixel i at the 25% video sequence

permissivity

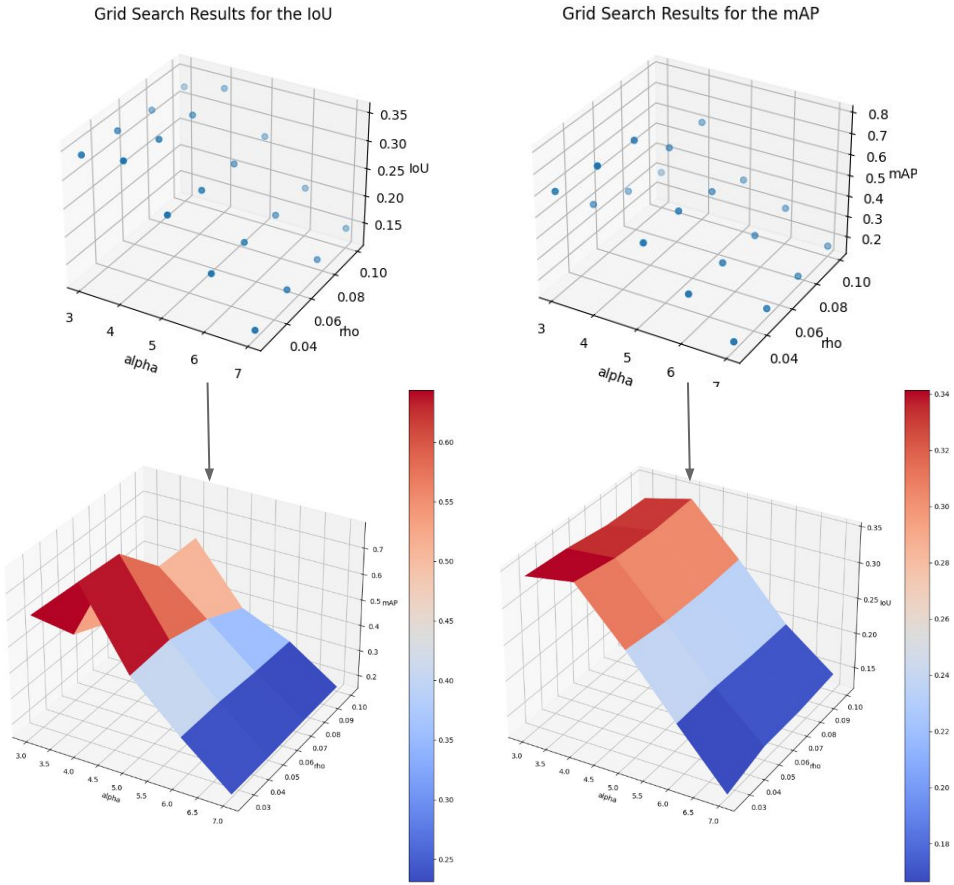
if pixel $i \in$ Background **then**

$$\mu_i = \rho \cdot I_i + (1 - \rho) \cdot \mu_i$$

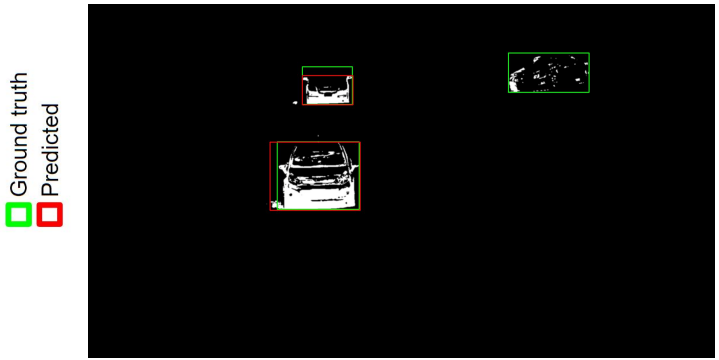
$$\sigma_i^2 = \rho \cdot (I_i - \mu_i)^2 + (1 - \rho) \cdot \sigma_i^2$$

end if

Task 2.1: Adaptive modelling (Team 6) [2/2]



- We first computed a grid search with the previous best alpha = 5 to know more or less the range of values of the rho, to later do a gird search with the two parameters together.
- We found out that the best was with alpha = 4 and rho = 0.025, increasing a lot the quality of the results with respect to the previous method
- To visualize better the results we computed the surface plots for the mAP and the IoU



with best alpha and rho

Task 2.1: Feedback

	feedback
Team 1	Nice plot of IoU vs frame for different parameter combination Not clear if joint optimization is a grid search. Details are missing.
Team 2	Optunas used for hyperparameter optimization.Good. Nice figure of the parameter spaces. Optimizing for P,R redundant with F1 No qualitative analysis of results
Team 3	Random search used for optimization. Lacks details on this process (iterations, etc.) No qualitative analysis of results
Team 4	Table not filled. what happened? (only 1 slide) you tried rho=0.3 and the best is rho=0.4? No qualitative evaluation.
Team 5	Bonus point for a nice hyper-parameter search. Quantitative and qualitative evaluation is shown.
Team 6	Good explanation of the task with equations. Qualitative and quantitative evaluation is shown. Good explanation of how you found the range of rho parameter.

Task 2.2: Comparison adaptive vs non

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

Task 2.2: Comparison adaptive vs non (Team X) - max 3 slides

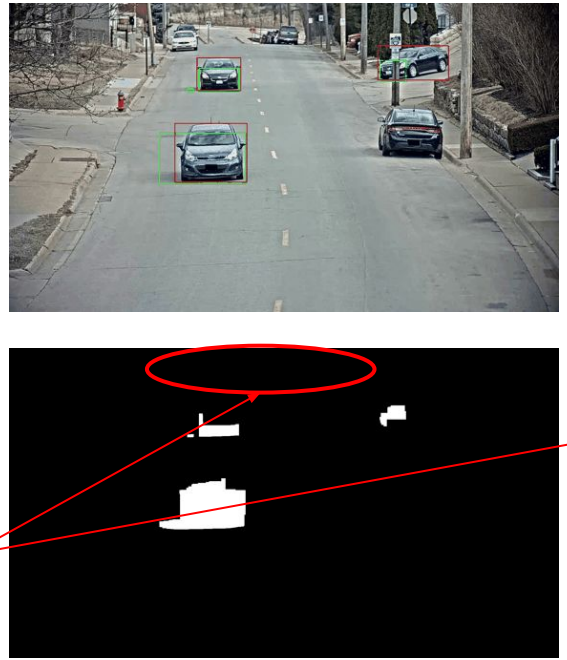
Task 2.2: Comparison adaptive vs non (Team 1) - [1/2]

None of the two models was able to not consider shadows as part of the detection.

Non-adaptive is more sensible to noise.

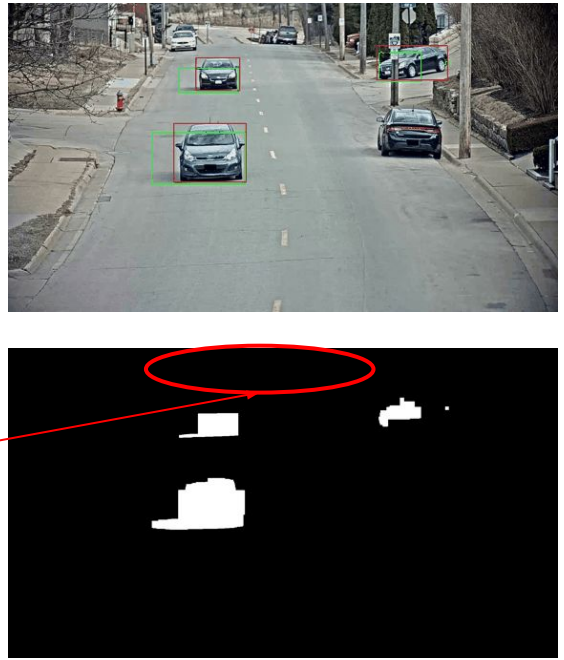
Non-adaptive seems to be **unable to detect cars further in the scene**, probably because they resemble to the background.

Non-adaptive



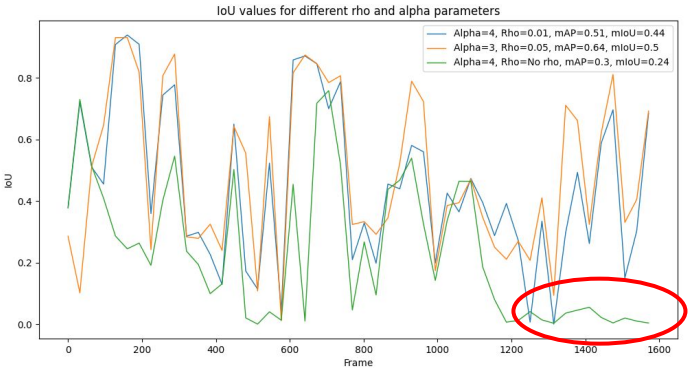
mAP=0.3, mIoU=0.24

Adaptive



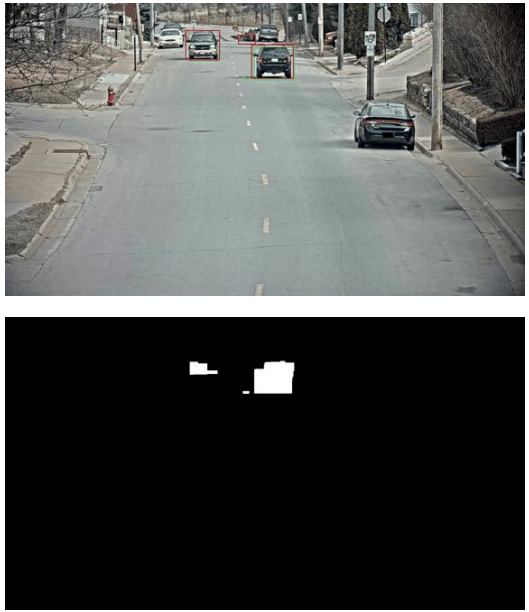
mAP=0.64, mIoU=0.5

Task 2.2: Comparison adaptive vs non (Team 1) - [2/2]



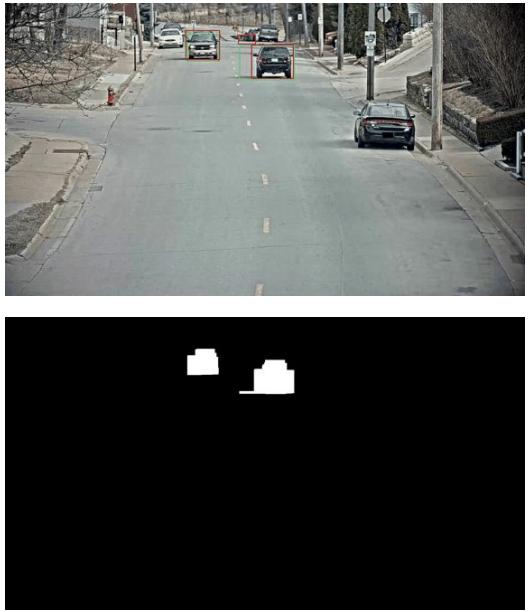
When we compare the problematic part of the sequence is when we see clear improvements, the adaptive is able to adapt the subtraction while the non-adaptive completely fails.

Non-adaptive



mAP=0.3, mIoU=0.24

Adaptive



mAP=0.64, mIoU=0.5

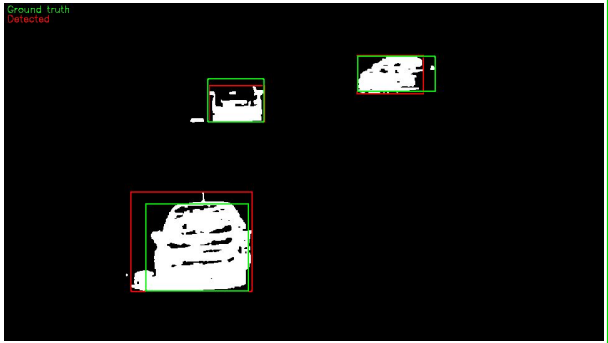
Task 2.2: Comparison adaptive vs non (Team 2) [1/2]

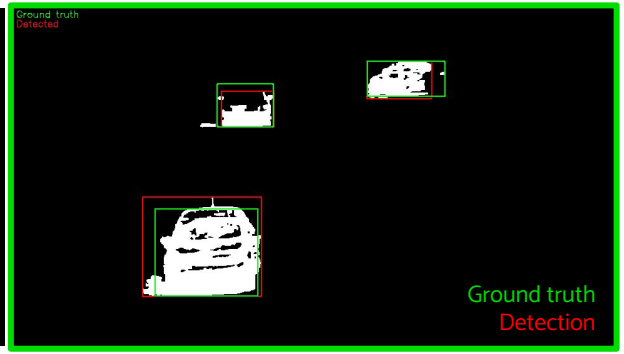
α	Precision	Recall	F1-score	AP	IoU
1	.0221	.0576	.0319	.0152	.0346
2	.1392	.3016	.1905	.1466	.1327
3	.1995	**.3659**	.2582	**.2338**	.1645
4	.2209	.3028	.2554	.2293	.1999
5	.3108	.2765	.2927	.2128	.2898
7	**.5305**	.2186	**.3096**	.2015	**.5093**
9	.4426	.1369	.2091	.1311	.4690

α	ρ	Precision	Recall	F1-score	$AP_{0.5}\downarrow$	IoU
4	**0.02**	.8253	.4731	.6014	**.4006**	.6680
4	0.09	.8356	.4574	.5912	**.3989**	.6383
3	0.20	.1878	.5325	.2776	**.3255**	.1496
5	0.40	.2456	.3889	.3011	**.1937**	.2111
6	0.50	.2023	.3347	.2522	**.1589**	.1830
2	0.07	.0776	.3316	.1258	**.1258**	.0798
3	0.42	.0245	.024	.0419	**.1003**	.0330

In terms of quantitative results, there is a clear improvement when using an adaptive modelling (right table) rather than a non-adaptive one (left table). Comparing best results from both, we observed an increase of 2x on AP, 4x on IoU and 2x on F1-score on the adaptive case. We can state that using an online approximation to update the model and then classify each pixel based on the Gaussian distribution is more effective to determine if it is considered part of the background model. However, this is not always true for some choices of α and ρ (last rows on red), as they can degrade the model performance even below the worst results obtained in an non-adaptive approach.

Task 2.2: Comparison adaptive vs non (Team 2) [2/2]

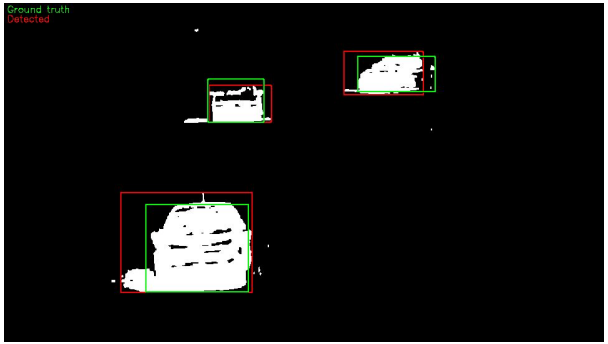


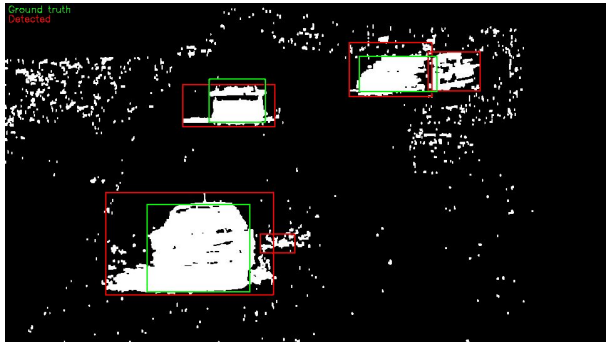


Non-adaptive
(α=4)

Best set of params
for both approaches

Adaptive
(α=4, ρ=0.02)





Non-adaptive
(α=3)

Worst set of params for
adaptive approach:

Adaptive
(α=3, ρ=0.42)

When qualitative comparing the best set of parameters, there is a clear winner when using an adaptive modelling. In the non-adaptive case, we observe some frames with significant noise in the background when the scene changes rapidly (new car entering the road), causing a lot of wrong detections. However, in the adaptive case, there is almost no trace of noise. Moreover, the adaptive modelling achieves to reduce the detection of the shadows, and it also fills more holes inside the objects, providing more compact detections.

However, if we look at the two sequences below, we note that the adaptive modelling also fails when using a bad choice of parameters α and ρ. We can observe that in the adaptive case there is a higher level of noise in the entire ROI, causing more wrong detections than in the non-adaptive case. As seen on the previous slides, quantitative results also conclude that in this specific case the adaptive approach provides worst metrics than non-adaptive.

Task 2.2: Comparison adaptive vs non(Team 3) 1/2

If we compare the best results obtained quantitatively, with the adaptive vs non- adaptive one, in the following tables, we observe that the adaptive performs nearly the double as the non-adaptive one.

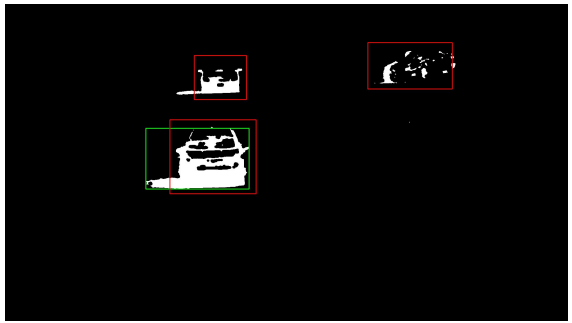
Alpha	Rho	Adaptive	mAP
4	-	No	0.32
3.3	0.0043	Yes	0.5812

Task 2.2: Comparison adaptive vs non(Team 3) 2/2

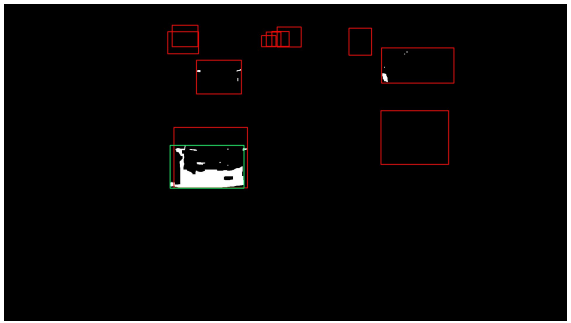
We can compare qualitatively the difference between the adaptive vs non-adaptative in the following frames. The main differences are at the **deep background** where in the non-adaptive algorithm there is no cars while in the adaptive ones there are

Ground truth

Prediction

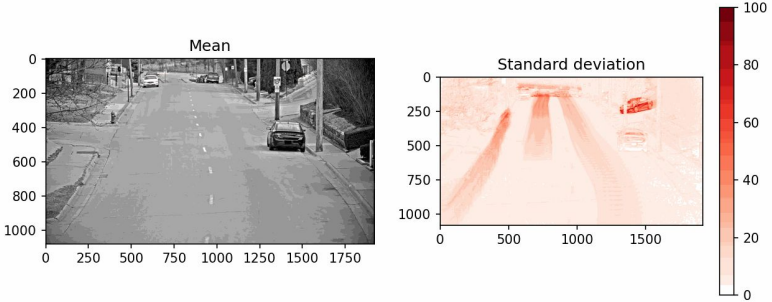


adaptive

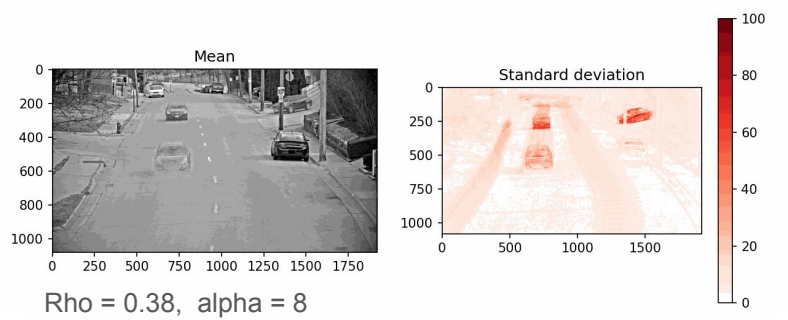


non-adaptive

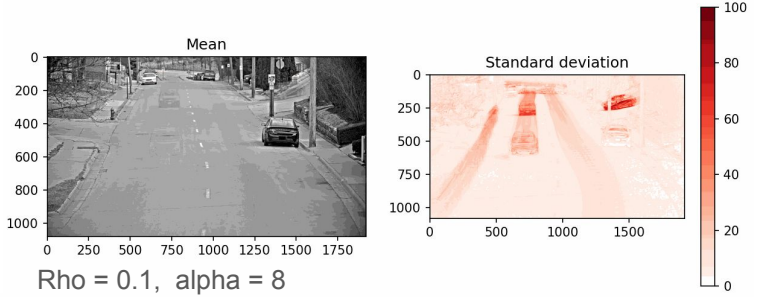
Task 2.2: Comparison adaptive vs non (Team 4) [1/2]



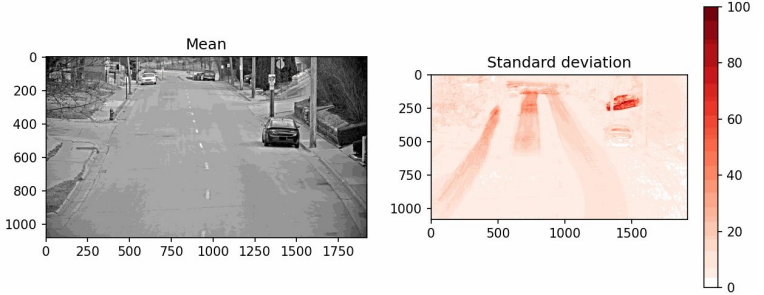
Rho = 0 (non adaptative), alpha = 8



Rho = 0.38, alpha = 8



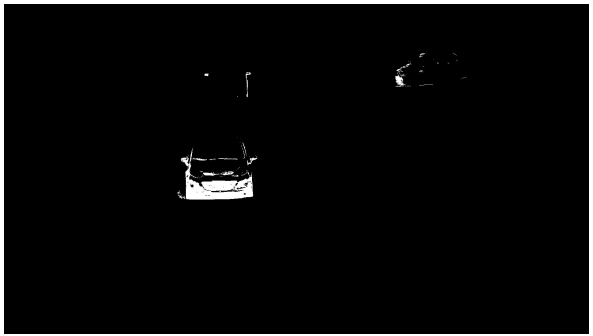
Rho = 0.1, alpha = 8



We are using a single channel and pixel-based approach, which gives us results far from perfect. Because of this, parts of the foreground that got mislabelled as the background generate artifacts in the model of the background, as we are estimating our background using assumptions from our mask result.
In particular, we can see how there are traces of the cars, specially when using bigger values for rho, as the model adapts faster to changes and includes the cars in the background estimate. Because of this we decided to use smaller alphas so that pixels would be rather classified as foreground instead and generate less interferences in the model.
We can see that in the last image this issue got mostly solved. However, the region where two cars cross in concurrently partially estimated a car as the background because two cars cross it in nearby frames.

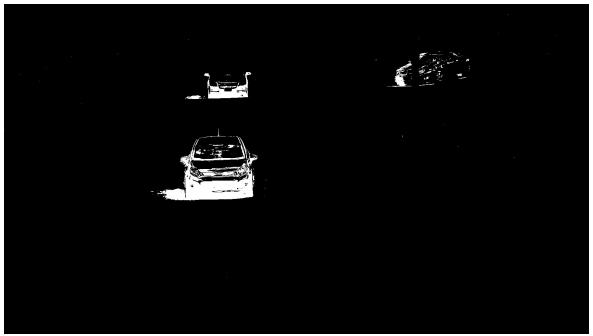
Task 2.2: Comparison adaptive vs non (Team 4) [2/2]

Non-adaptative case



Rho = 0 (non adaptatve), alpha = 8

Adaptative case



Rho = 0.1, alpha = 4

In this case, we can see that the first car is slightly more visible, although the shadows are more visible as we are using a smaller alpha and thresholding more part of the image as foreground.

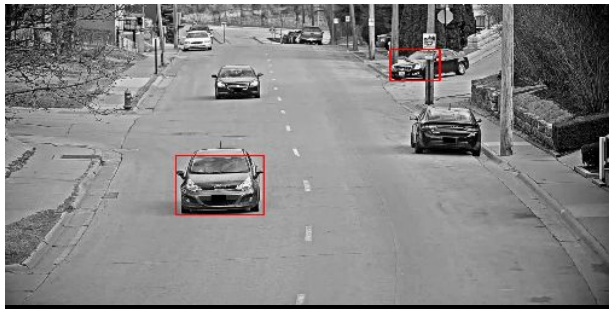
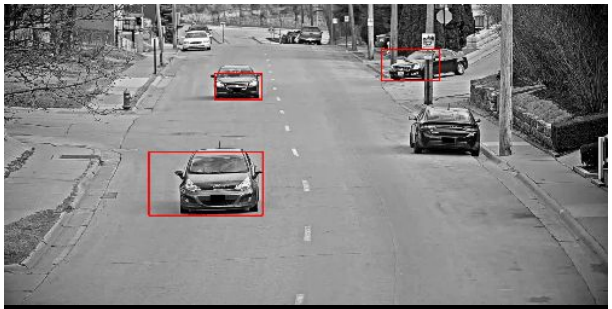
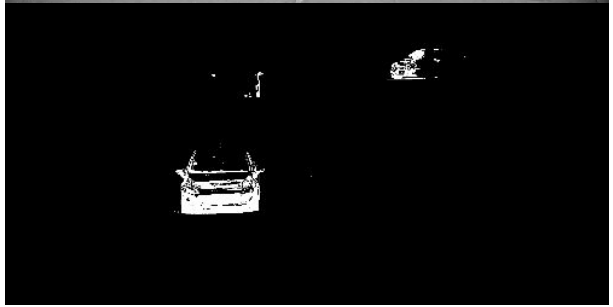
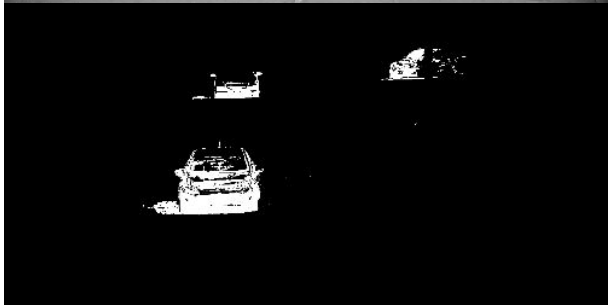
Furthermore, the cars on the back now appear in the output mask.

Approach	**Parameters**	**MAP**
Non-adaptative case	alpha=8	0.219
Adaptative case	rho=0.1, alpha=4	0.34

Task 2.2: Comparison adaptive vs non (Team 5) [1/3]

Ground truth box

Detected box

	Static	Adaptive
BBox estimation		
Foreground estimation		
	mAP = 0.235, mIoU = 0.172	**mAP = 0.52, mIoU = 0.49**

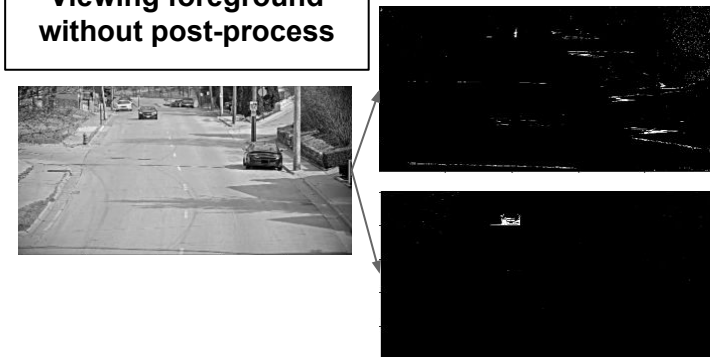
Task 2.2: Comparison adaptive vs non (Team 5) [2/3]

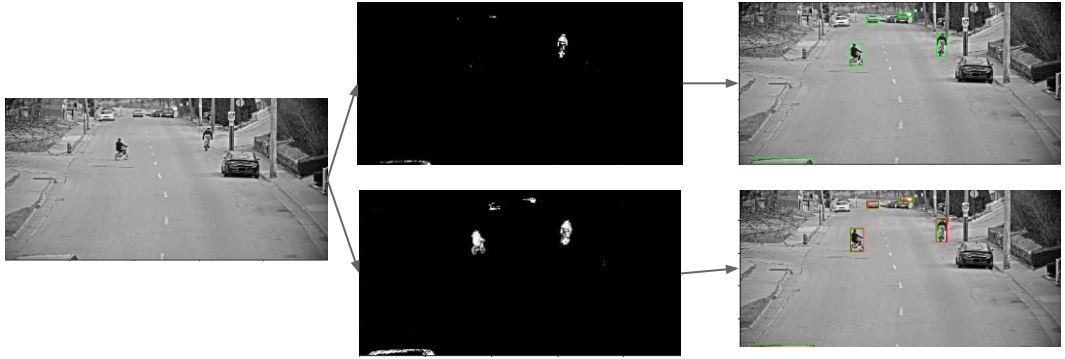
TOP: ADAPTIVE, **BOT:** STATIC

Ground truth box

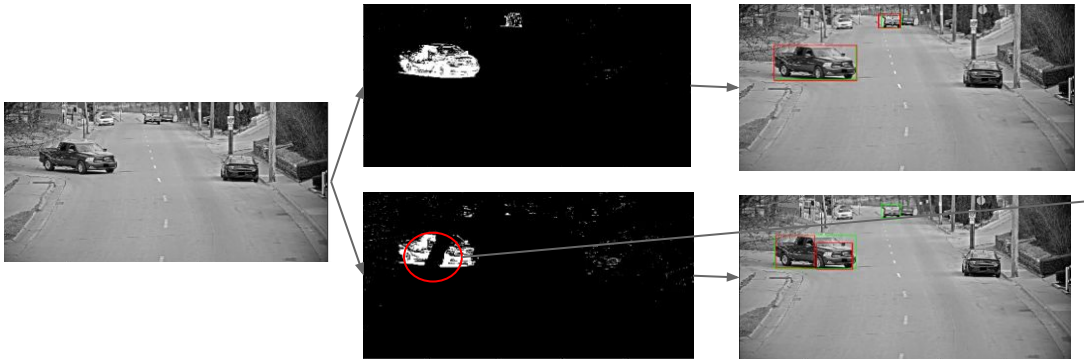
Detected box

Viewing foreground without post-process





The static model produces a foreground estimation with a significant amount of noise that must be removed using a major post-processing technique. In contrast, the adaptive model generates a foreground estimation with less noise, enabling the detection of smaller objects.



INITIAL ESTIMATED STD

Poor initial estimations in the static model can result in the failure to detect some objects. In this particular case, it caused the detection of one car as two separate objects, which was not observed in the adaptive model.

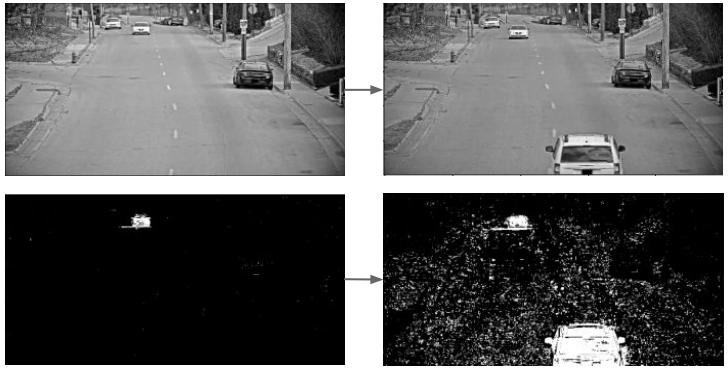
Task 2.2: Comparison adaptive vs non (Team 5) [3/3]

ADAPTIVE FAILURES

Viewing foreground without post-process

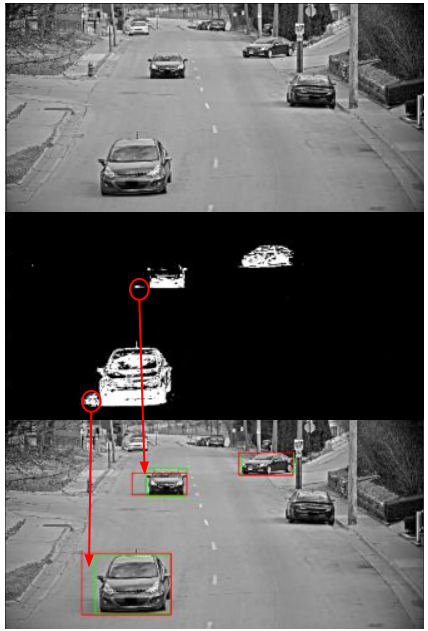
Ground truth box

Detected box



The adaptive model is not completely robust to sudden, significant changes in illumination. It requires several iterations to adapt to these changes effectively.

When two objects overlap, it may not be possible to detect them separately.



Shadows are sometimes detected as objects, resulting in a IoU score for the detected boxes. This issue could be addressed by utilizing color spaces that have a separated luminance channel.

Task 2.2: Comparison adaptive vs non (Team 6) [1/2]

Non Adaptive

alpha	mAP	IoU
1	0.008	0.079
2	0.087	0.257
3	0.208	0.309
4	0.270	0.310
5	**0.405**	**0.278**
6	0.406	0.242
7	0.303	0.191
8	0.153	0.136
9	0.091	0.096
10	0.072	0.063

Adaptive

alpha	rho	mAP	IoU
3.0	0.025	0.610	0.340
3.0	0.05	0.410	0.335
3.0	0.075	0.336	0.326
3.0	0.1	0.296	0.323
4.0	**0.025**	**0.784**	**0.351**
4.0	0.05	0.770	0.340
4.0	0.075	0.606	0.336
4.0	0.1	0.601	0.339
5.0	0.025	0.493	0.280
5.0	0.05	0.499	0.272
5.0	0.075	0.453	0.270
5.0	0.1	0.374	0.272
6.0	0.025	0.319	0.200
6.0	0.05	0.315	0.202
6.0	0.075	0.301	0.199
6.0	0.1	0.293	0.198
7.0	0.025	0.161	0.124
7.0	0.05	0.161	0.140
7.0	0.075	0.165	0.140
7.0	0.1	0.166	0.145

Here we can see the quantitative results of the two grid searches.

Wen using adaptive modeling we see a clear improvement in the results in general.

Also, note that the best result improves a lot the mAP when using Adaptive Gaussian modeling.

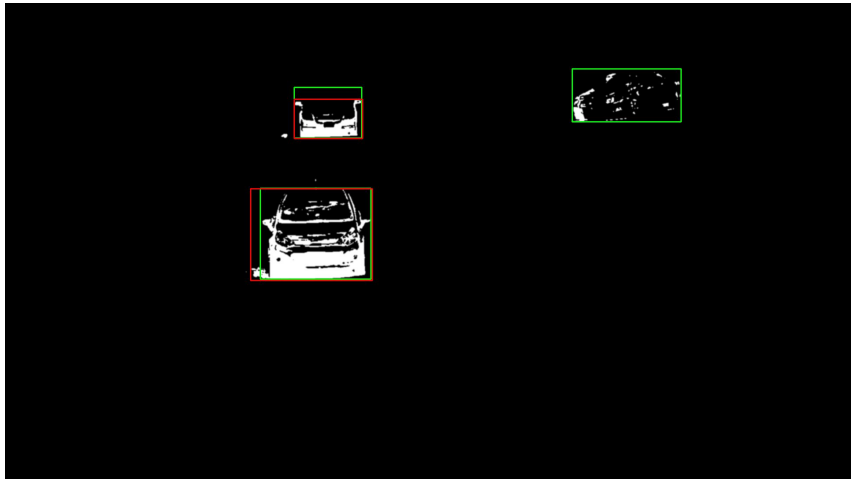
Averall, we can clearly state that adaptive methods work better but there is a need to optimize the two parameters, alpha and rho.

Task 2.2: Comparison adaptive vs non (Team 6) [2/2]

Best wit Gaussian modeling

alpha = 5
mAP = **0.405** IoU = 0.278

Best with Adaptive Gaussian modeling



alpha = 4 rho = 0.025
mAP = **0.784** IoU = 0.513

- We can see how adaptive modeling performs better even when there are illumination changes (like at the end of the sequence).
- Note also how we are able to detect the cars far away from the camera better and the method is less sensible to noise
- Both models not be effective at removing non-Gaussian noise, such as impulsive noise or random spikes in the video sequence.
- Adaptive Gaussian gets more density objects.

Task 2.2: Feedback

	feedback
Team 1	Good discussion with the help of the IoU vs frame plot Qualitative analysis given Conclusion: non-adaptive less robust to noise, non-adaptive fails at detecting small cars.
Team 2	Excellent quantitative and qualitative analysis. Table could be simplified. Conclusion: clear improvement for adaptive, non-adaptive less robust to noise,
Team 3	Analysis maybe too simple. Conclusion: non-adaptive fails at detecting small cars.
Team 4	Good discussions on the differences between the adaptive and none with mean/std figures. side-by-side qualitative and quantitative evaluation Conclusion: car more visible but also shadows, small cars at the back appear in the adaptive case
Team 5	side-by-side qualitative and quantitative evaluation Conclusion: adaptive generates foreground with less noise. Bonus point: Study of the cases where adaptive fails, shadows.
Team 6	Better to use a 2D/3D plot to represent instead of table. side-by-side qualitative and quantitative evaluation Conclusion: adaptive better with far away cars, noise is not removed, cars more visible.

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)
  - 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

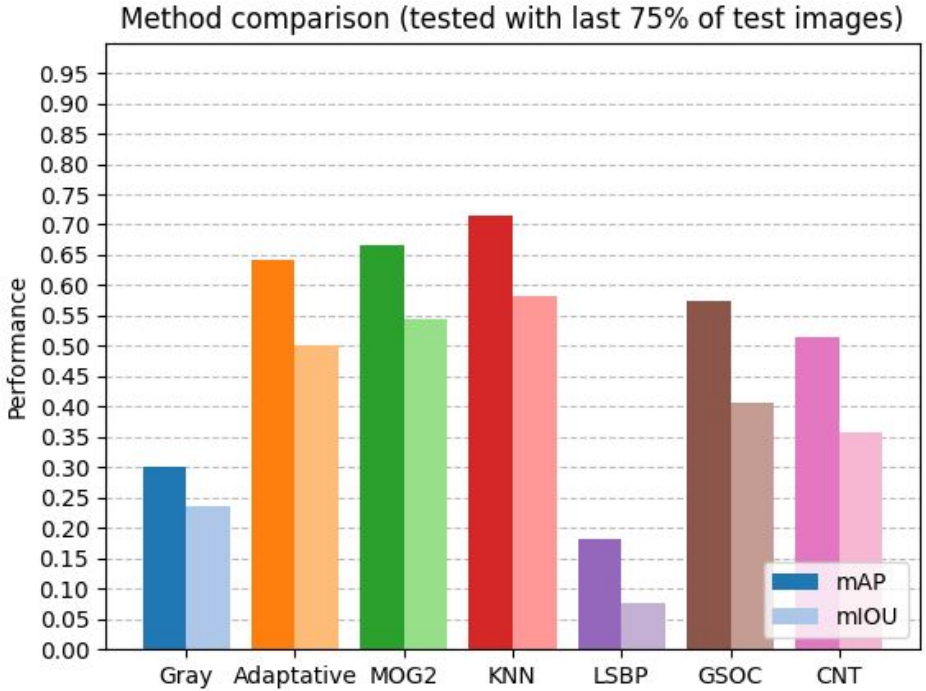
Task 3: Comparison with state-of-the-art (All teams)

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

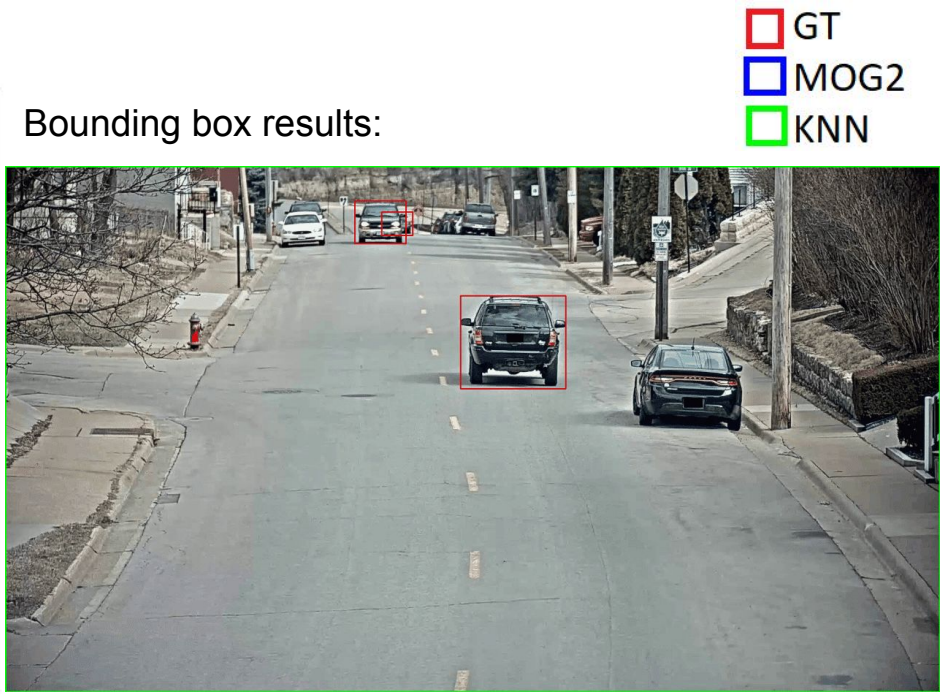
Team ID	Others	Best yours
Team 1	0.715 (KNN)	0.64 (adaptive)
Team 2	0.37 (KNN)	0.40 (adaptive)
Team 3	0.522 (KNN)	0.58(adaptative)
Team 4	0.3909 (MOG2)	0.34 (adaptive)
Team 5	0.3574 (KNN)	0.52 (adaptive)
Team 6	0.780(MOG2)	0.784(adaptive)

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

Task 3: Comparison with state-of-the-art (Team 1) - [1/3]



Bounding box results:



Task 3: Comparison with state-of-the-art (Team 1) - [2/3]

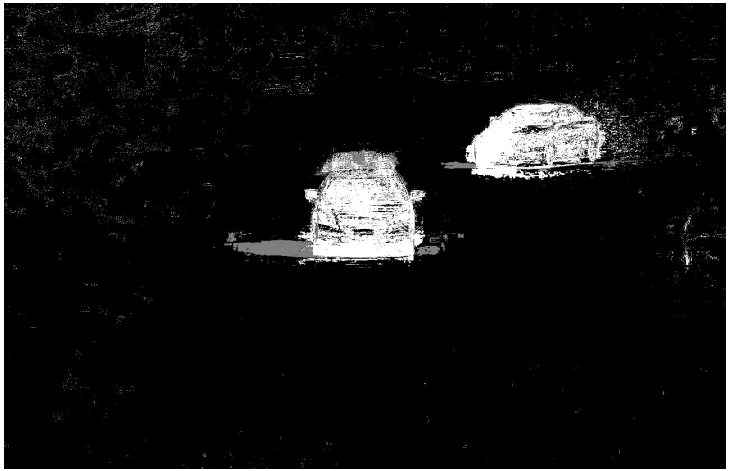
Similarities of KNN and MOG2:

- They both use adaptive background models that can learn and update the statistical properties of the background over time.
- Both algorithms can handle complex backgrounds with changing lighting conditions and moving objects.
- Both algorithms detect shadows as a separate object from the foreground, which can help avoid false positives in the results.

Big difference in our scenario:

- KNN performs better at handling slow illumination changes in the background because it relies on the history of pixel values. As seen in previous slide, in our case KNN generates less noisy bounding boxes.

	MOG2	**KNN**
Approach used to model the background	Mixture of Gaussian distributions	K-nearest neighbors
Performs better	There is a lot of background motion	Better suited for scenarios with less background motion (our case)
Computationally expensive	15it/s	17it/s



Task 3: Comparison with state-of-the-art (Team 1) - [3/3]

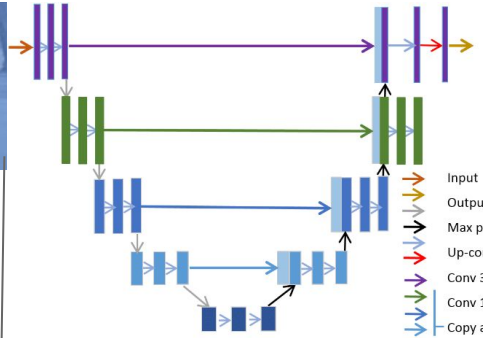
U-Net MOG2 mask prediction refinement experiment

- Using the rgb input image, and MOG2 mask output, we tried training a UNET to achieve masks more similar to ground truth, to avoid holes and noise in the mask, that end up being multiple bboxes (lot's of FP).
- The UNET was trained with the 25% of training set not used in the prediction.


holes


noise


RGB + MOG2 mask
4 channel image
(w,h,RGBM)



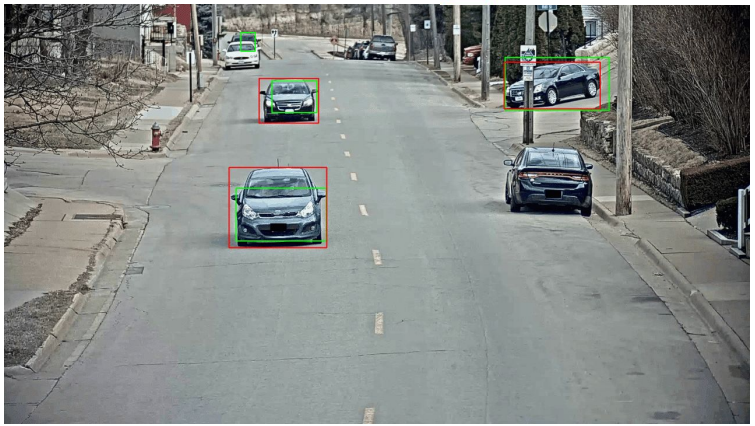

Prediction epoch 16
(binary mask)


Ground truth

mAP: 0.6647
mIoU: 0.5452

mAP: 0.6754
mIoU: 0.4752

Bounding box results:



- UNET **did not learned to remove noise** even though GT had no noise. Probably it was due to the fact that in train set there was no noise.
- UNET did learn to do **better masks without holes**
- UNET is **not a viable option for background subtraction** because it takes 23 minutes to process the entire dataset, whereas MOG2 only takes a couple of minutes.

Task 3: Comparison with state-of-the-art (Team 2) [1/2]

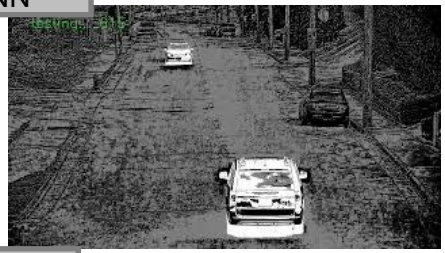
Method	Time [s]	Precision	Recall	F1-score	AP	IoU
Single Gaussian*	-	.19	.36	.25	.23	.16
Adaptive Gaussian*	3min46s	**.82**	**.47**	**.60**	**.40**	**.66**
KNN	2m18s	.55	.25	.32	**.37**	.47
MOG2	2m34s	.53	.24	.30	.37	.44
LSBP	5m59s	.20	.31	.21	.18	.12

Surprisingly, there's no already implemented method in the SOTA which outperformed the Adaptive Gaussian approach. In the qualitative analysis (see next slide) we observe sudden changes in illumination which could be affecting the method in situations where adaptive gaussian could adapt.

* Higher AP

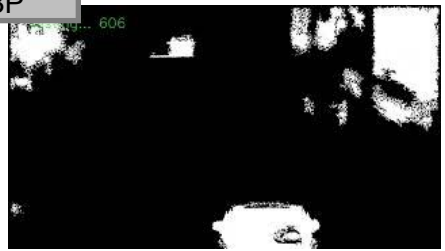
Task 3: Comparison with state-of-the-art (Team 2) [2/2]

KNN



MOG2

LSBP



Both MOG2 and the KNN approach yield similar results and interactions. This methods are not robust to changes in illumination in the scene, which could be the reason for the poor performance as we expected them to actually outperform our method.<

Although LSBP yields the worst performance with the worst efficiency it seems less sensitive to changes and noise. The drop in performance may be due to a more aggressive segmentation and, therefore, elimination of many True Positives for the sake of False Positive reduction.

Task 3: Comparison with state-of-the-art (Group 3) 1/3

SOTA Implemented methods:

CNT (Change Detection based on Temporal Coherence)

GSOC (GrabCut based on Separating Objects and Background using Contrast)

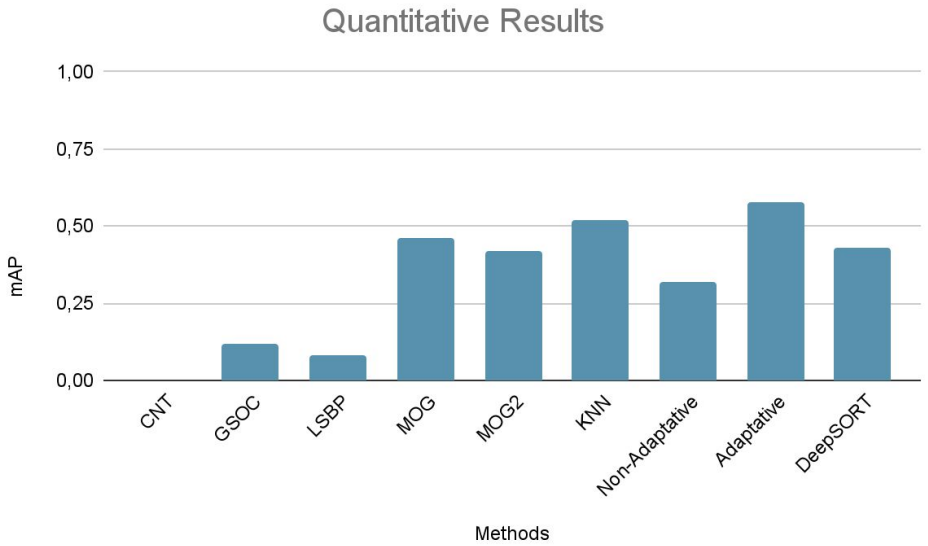
LSBP (Local Saliency-based Propagation)

MOG (Mixture of Gaussians)

MOG2 (Mixture of Gaussians version 2)

KNN (K-Nearest Neighbors):

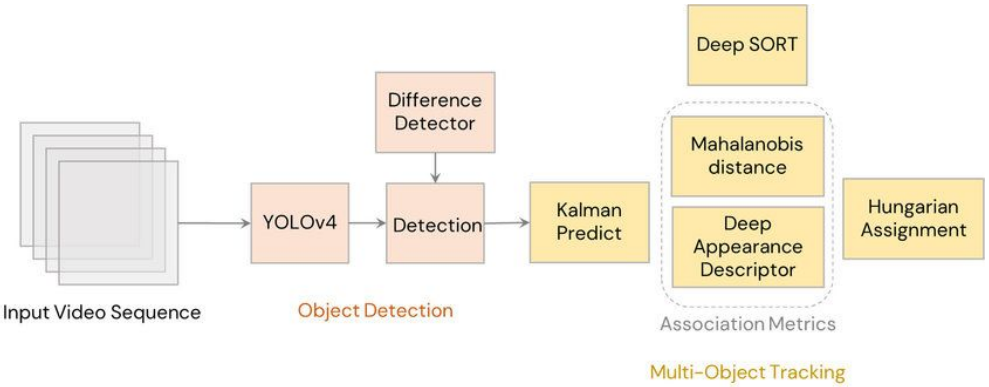
DeepSORT (Deep Learning for Single Object Tracking)



Task 3: Comparison with state-of-the-art (Group 3) 2/3

BONUS Part

DeepSORT is a computer vision tracking algorithm for tracking objects while assigning an ID to each object. DeepSORT is an extension of the **SORT** (Simple Online Realtime Tracking) algorithm. DeepSORT introduces deep learning into the SORT algorithm by adding an appearance descriptor to reduce identity switches, Hence making tracking background estimation more efficient. Deep Learning for Single Object Tracking is a state-of-the-art method that uses deep learning for tracking and **can perform well in complex scenarios but requires large amounts of training data and computational resources.**



DeepSORT

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

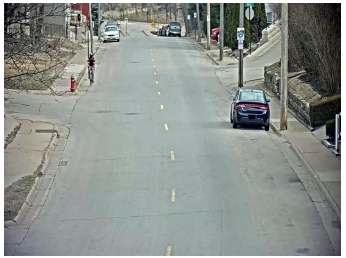
We can conclude:

- **CNT**: Change Detection based on Temporal Coherence is simple and fast but may not perform well in complex scenarios. But didn't worked for us due to the implementation.
- **GSOC**: GrabCut based on Separating Objects and Background using Contrast is interactive and produces high-quality results but is time-consuming and may not be suitable for large-scale applications.
- **LSBP**: Local Saliency-based Propagation uses local saliency information for more accurate results but may require more computational resources.
- **MOG/MOG2**: Mixture of Gaussians models the background as a mixture of Gaussians and can handle complex scenarios. MOG2 is an improved version that can handle more complex scenarios than MOG.
- **KNN**: K-Nearest Neighbors is simple and fast but may not perform well in complex scenarios in this case it performed well
- **DeepSORT**: Deep Learning for Single Object Tracking is a state-of-the-art method that uses deep learning for tracking and can perform well in complex scenarios but requires large amounts of training data and computational resources.
- The choice of method depends on the specific application and the complexity of the background. There is no one "best" method for background removal. But in this case, **the SOTA method that performed better for us is the KNN but didn't overperfed the own adaptative method**

GSOC

LSBP

The difference between performances respect to other groups may be due to the random selection of the different clips of the video, the postprocessing of the frames and also we have considered in most of the frames only the moving cars

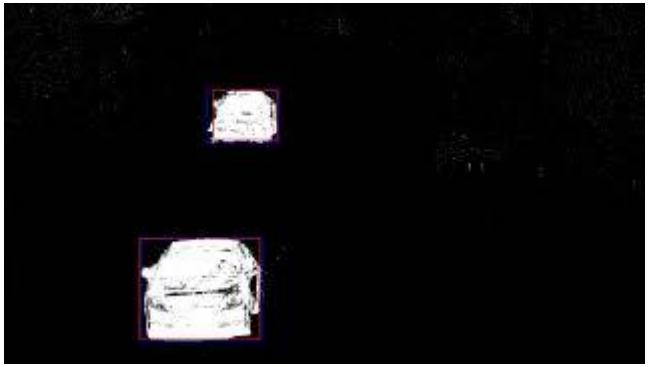


KNN

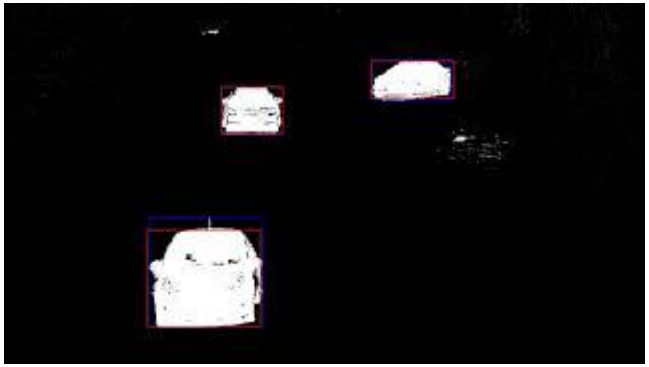
MOG

MOG2

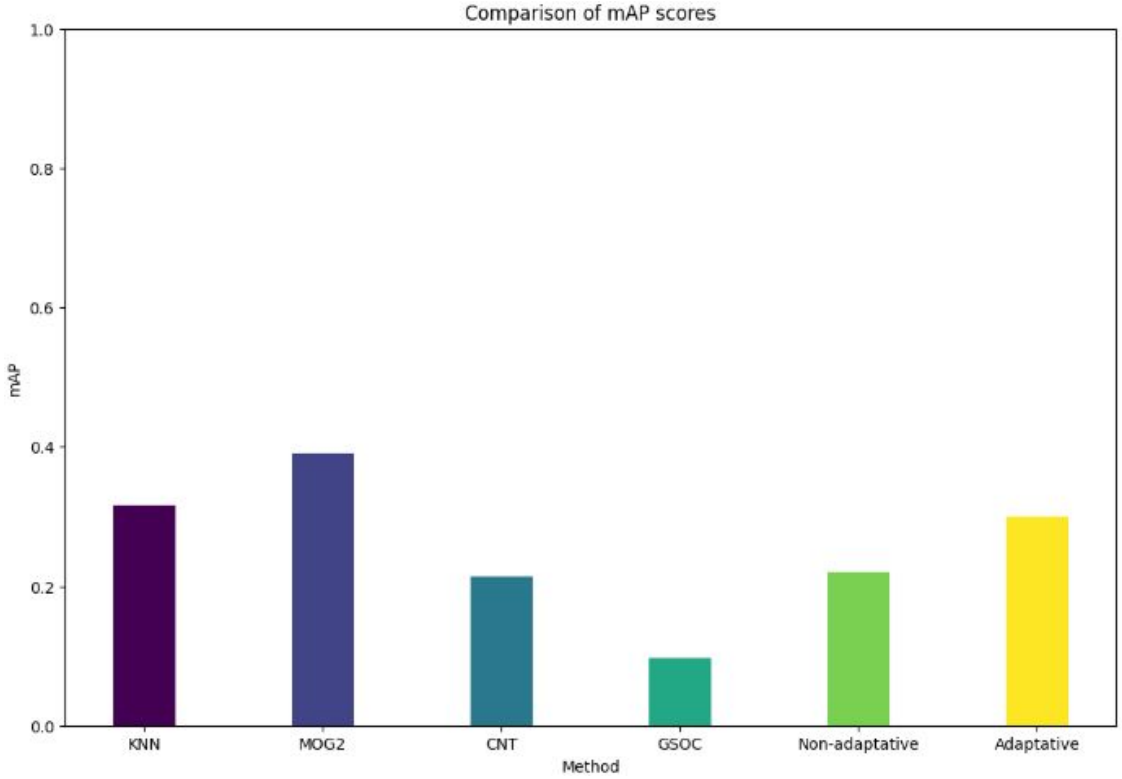
Task 3: Comparison with state-of-the-art (Team 4) 1/3



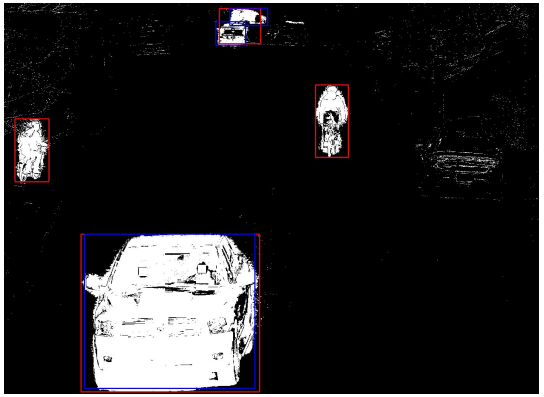
MOG2



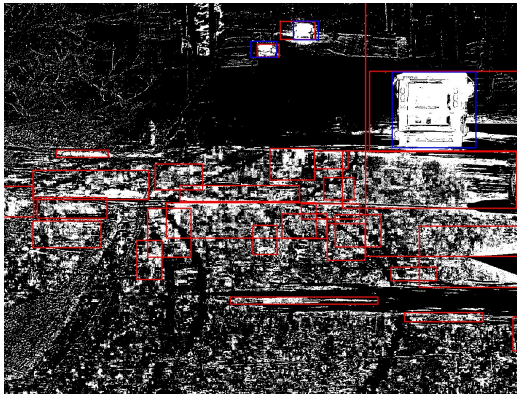
KNN



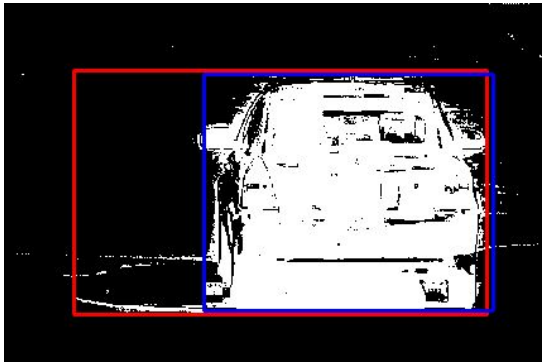
Task 3: Comparison with state-of-the-art (Team 4) 2/3



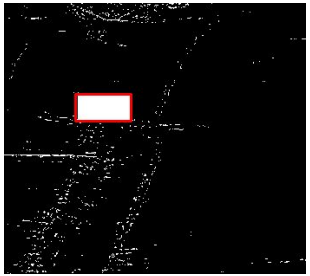
In KNN and MOG2 bicycles are also detected



In some frames, we still have a lot of false positives due to changes in illumination.

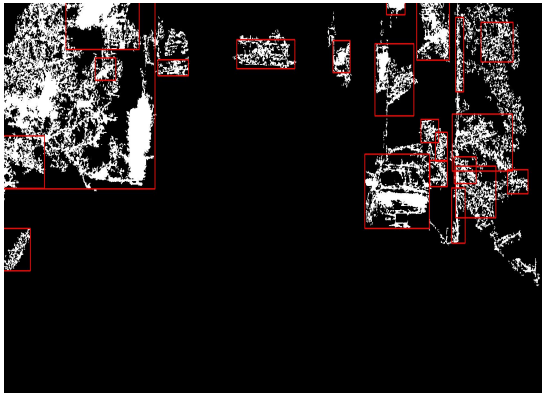


KNN and MOG2 still leave some pixels related to the shadows despite the attempts to remove them, resulting in a lower IoU, and therefore, in a lower mAP

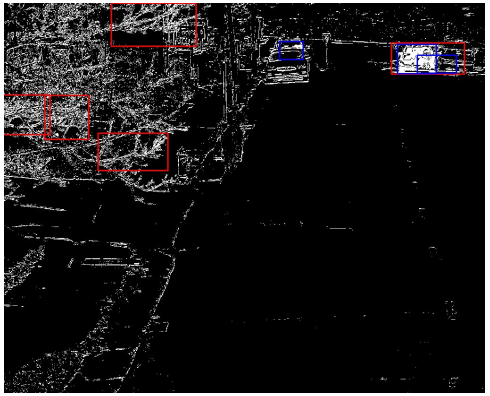


Artifacts appear in some frames

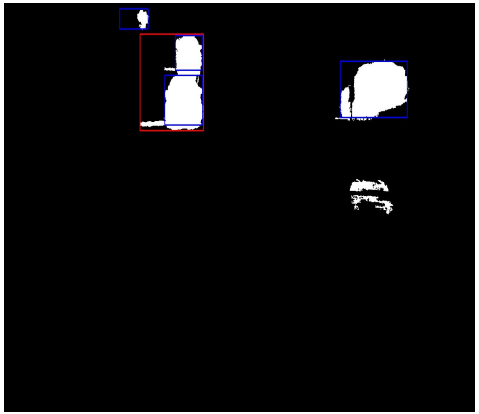
Task 3: Comparison with state-of-the-art (Team 4) 3/3



GSOC initially has problems with detections from trees and parked cars



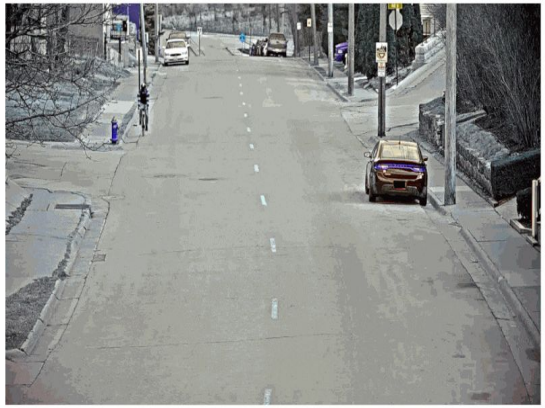
CNT also has this problem when there are small changes in illumination (parts of the trees and parked cars are detected)



In GSOC the trees and the parked cars progressively disappear as the background is updated

Task 3: Comparison with state-of-the-art (Team 5) 1/3

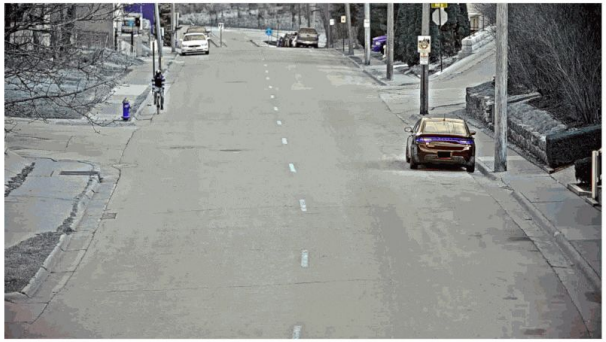
GMG: Very old approach, lots of false positive



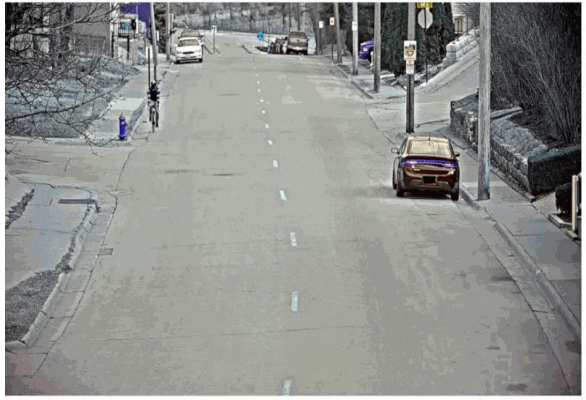
LSBP: improve the performance from GMG, suffers from the effects of the artifacts.

Ground truth box

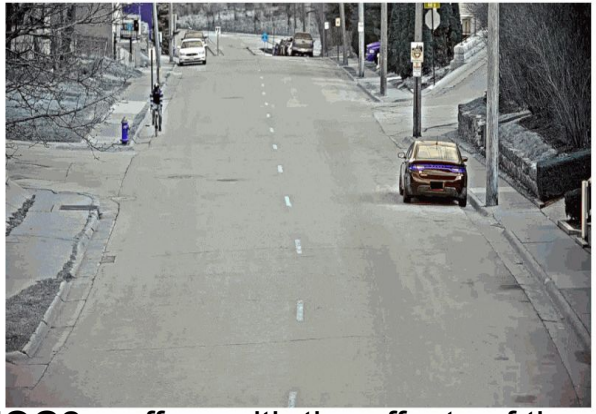
Detected box



KNN: provides better bounding box results compare to the other state-of-the-art approaches

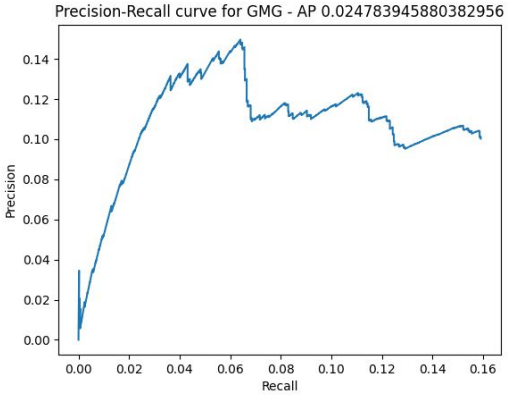


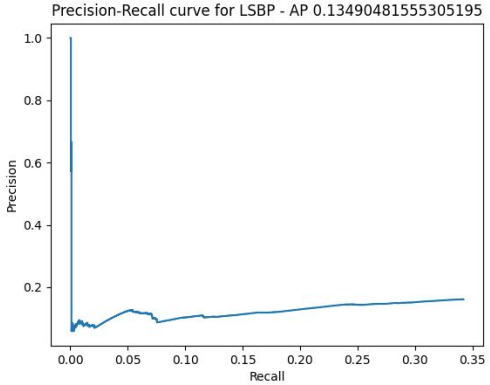
MOG: Also suffering from artifacts, sometimes bycycles are also detected.

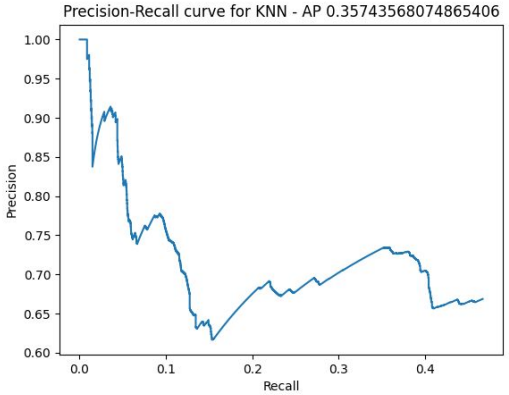


MOG2: suffers with the effects of the shadow

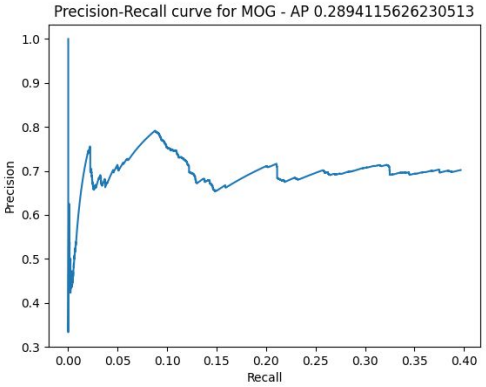
Task 3: Comparison with state-of-the-art (Team 5) 2/3

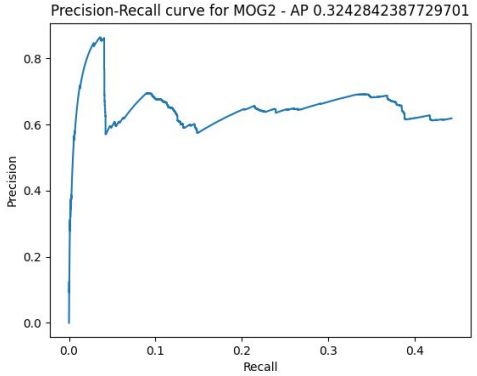


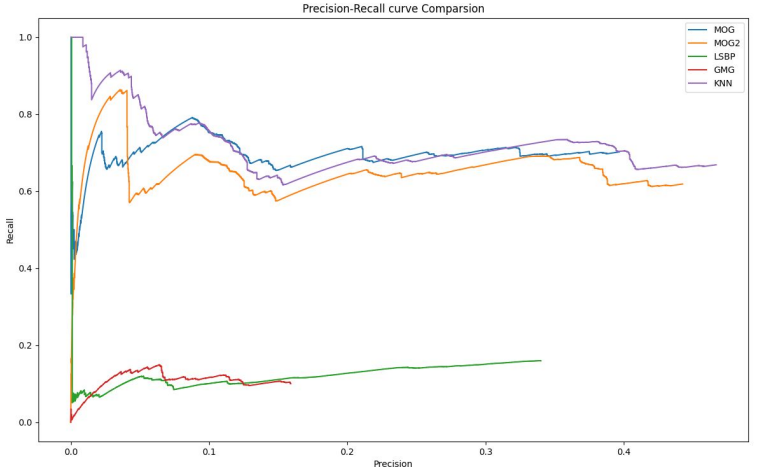




MOG2 and KNN provides almost the same performance but KNN wins the race whereas, MOG2 is more stable







GMG and LSBP are neither stable and not meet the desired result.

Task 3: Comparison with state-of-the-art (Team 5) 3/3

- **GMG:** A bank of Kalman filters and Gale-Shapley matching are used to approximate a solution to the multi-target tracking problem in their algorithm, which also incorporates statistical background image estimation and per-pixel Bayesian segmentation. Tracks can be selectively filtered using a heuristic confidence model based on dynamic data. But generate a lot of false positive in this particular sequence.

- **LSBP**: For more accurate results, Local Saliency-based Propagation makes use of information about local saliency, but it may require more computational power. It also suffers with artifacts effect.

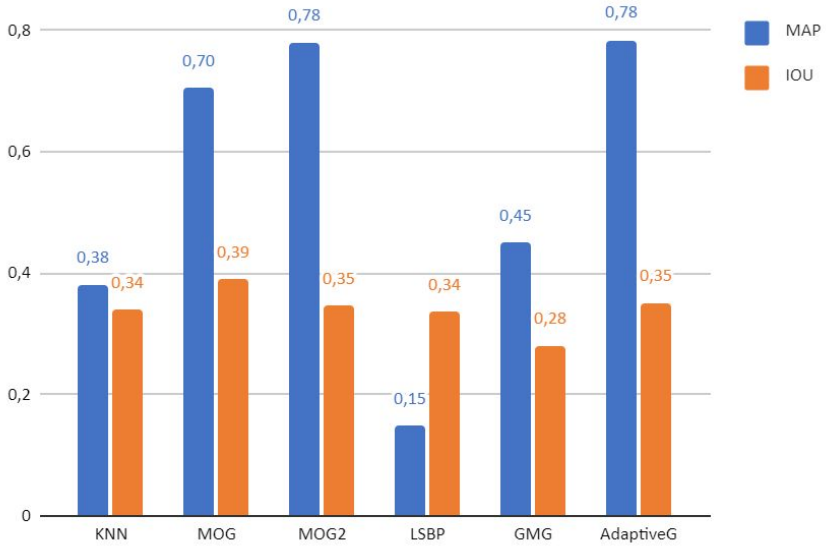
- **MOG:** A way to make this adaptive background mixture model better. By reinvestigating the update conditions, they used various conditions at various stages. Their system can learn more quickly and accurately as a result of this, and it can also effectively adapt to changing environments. Their background model served as the basis for a computational color space.

- **MOG2:** Their adaptive algorithm using Gaussian mixture probability density. Recursive equations were used to constantly update the parameters and but also to simultaneously select the appropriate number of components for each pixel.

- **KNN:** It presented recursive equations that are utilized to simultaneously select the appropriate number of components for each pixel and to continuously update the parameters of a Gaussian mixture model. It helps to improve the kernel density estimation. We found it as the best state-of-the-art approach.

Comparison of background subtraction methods on near Infra-Red spectrum video sequences

Task 3: Comparison with state-of-the-art (Team 6) [1/2]



- Of the SOTA models the best result was obtained with the MOG2 method. Which reached a similar performance to our Adaptive Gaussian model.
- The same post-processing was applied to all methods.

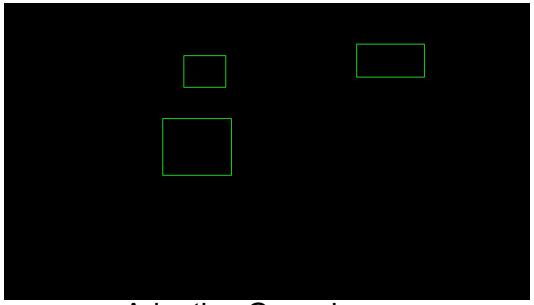
We tested the following SOTA methods, as proposed in the slides and two extra ones, KNN and GMG

- **MOG**: A pixel-based method for background subtraction that models each pixel as a mixture of Gaussian distributions.
- **MOG2**: An improved version of MOG that is more robust to noise and adapts better to changes in lighting conditions.
- **LSBP**: A local binary pattern-based method for background subtraction that uses a binary pattern to describe each pixel in the image.
- **KNN**: A pixel-based method for background subtraction that uses a K-nearest neighbors algorithm to classify pixels as foreground or background.
- **GMG**: A pixel-based method for background subtraction that models each pixel as a mixture of Gaussian distributions and adapts its model online to changes in the background.

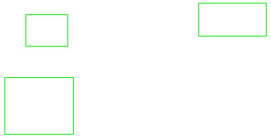
We used all the implementations of this methods provided by OpenCV.

Task 3: Comparison with state-of-the-art (Team 6) [2/2]

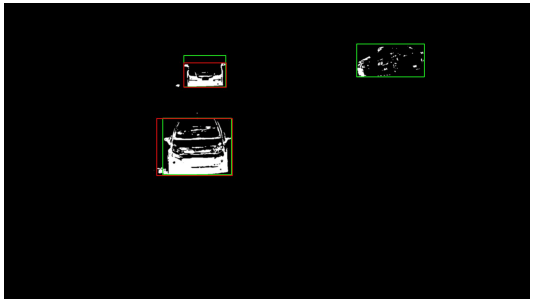
MOG



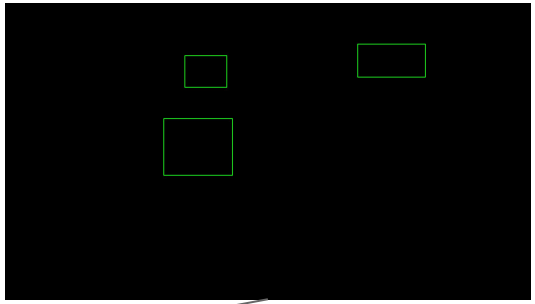
KNN



Adaptive Gaussian



MOG2

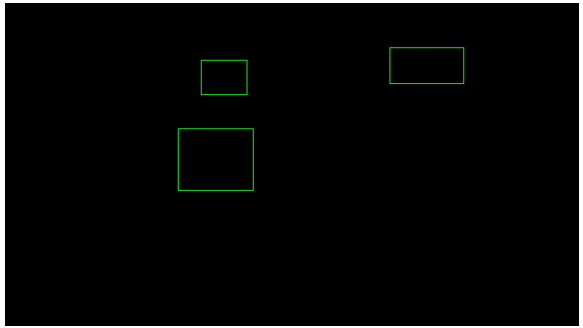


As we saw in the quantitative analysis, MOG2 performs the best, similar to our Adaptive Gaussian implementation. The main difference is that the adaptive is able to correct shadows, unlike the MOG2.

Despite having obtained a big jump between the mAP of the MOG and KNN, the visualization of both is quite similar, we can find a bigger difference in the segmentation of the shadows of moving cars, making the MOG method more robust. Small details seem to segment better with the KNN algorithm, but in our case this only generates more noise in the detection.

Note that MOG2 is able to detect the shadows (in gray) from the detections, so a further post processing should be added. In fact, if we remove the pixels detected as shadows, we have more accurate BB but in some cases we lose some detections like the cars far away from the camera, achieving a similar mAP and IoU than before.

MOG2 + removing the detected shadows

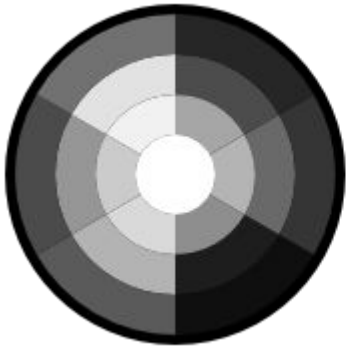


Task 3: Feedback

	feedback
Team 1	Methods without references, not full names. Table explaining the difference between KNN and MOG2 methods. Good. Bonus point for implementing unet to improve the results. But details missing! implementation?
Team 2	Methods without references, not full names. implementation?
Team 3	Methods with names but no references. Deep Sort for FG? Other SOTA methods explored. Good. implementation?
Team 4	Methods without references, not full names. Better to compare with less but reference and explain the differences. What artefacts are you talking about? implementation?
Team 5	Bonus point to include an explanation of the methods, missing references Used precision recall curves, why do you include them instead of using AP? implementation?
Team 6	Bonus point for including an explanation of the methods and imlpementation. Good discussion of results and differences between methods.

Task 4: Color sequences

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



Task 4: Color (All teams)

Best AP_{50} (color space)

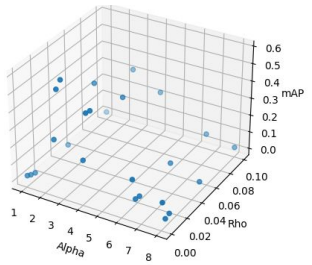
Team ID	Best
Team 1	0.61 (XYZ(2 gaussians))
Team 2	.4981 (α=5.5, ρ=0.59, RGB(3 gaussians), all channels, simple voting)
Team 3	0.68 (RGB(3 gaussians))
Team 4	
Team 5	0.31 (A-channel from LAB (1 gaussian))
Team 6	0.42 (RGB)

Task 4: Color (Team X) - max 2 slides

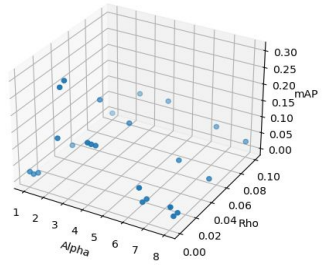
Task 4: Color (Team 1)

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

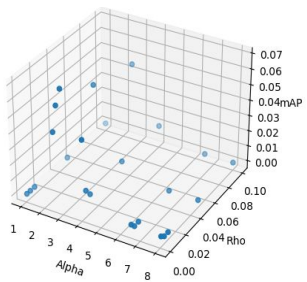
RGB



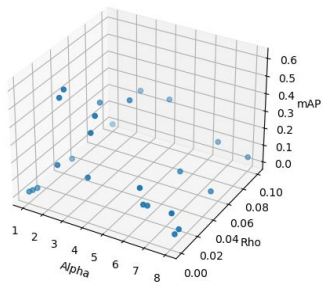
HSV(HS)



YUV (UV)



XYZ (XZ)



All best values obtained with:

- alpha = 2.5
- rho = 0.01

Color space	Num. Gaussians	mAP	mIoU
RGB	3	0.5861	0.4590
HSV	2	0.3608	0.2678
YUV	2	0.0683	0.0355
XYZ	2	0.6112	0.4710

- Best performance obtained with the first and last channel of the XYZ color using 2 gaussians. The worst performance is given with YUV using 2 gaussians also. We avoid using the brightness or luminance component in the HSV, YUV and XYZ.
- The best performance obtained in all the color spaces was using the same alpha and rho parameters. Moreover the relationship between the mAP and the two parameters is similar in all them.

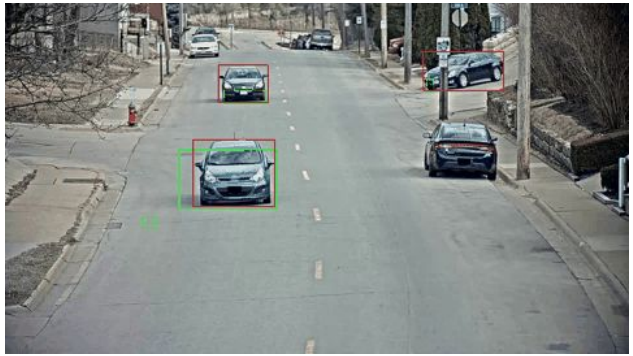
Task 4: Color (Team 1)

Ground truth box

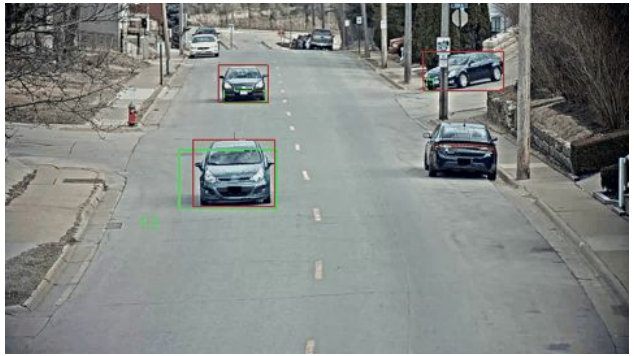
Detected (annotations with noise) box

*GIFs for the first 400 frames of the test split

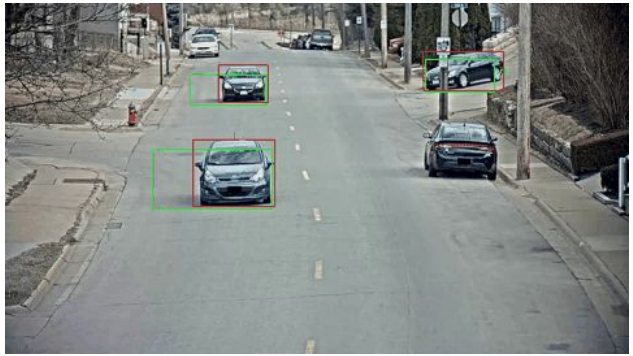
RGB



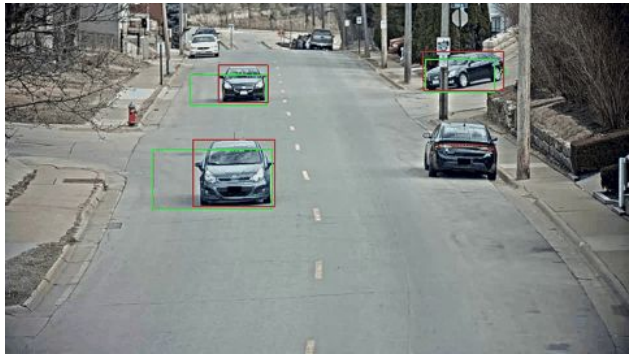
HSV(HS)



YUV (UV)



XYZ (XZ)



Making a qualitative analysis we can observe the following:

- In all the color spaces almost all the near objects are being detected, but with the far objects this depends on the color space.
- The difference in performance comes from the precision of the bounding boxes predicted and how much detections are caused by noise.
- Other object like the kids in the bikes causes the model to fail in the prediction of the car bounding box.

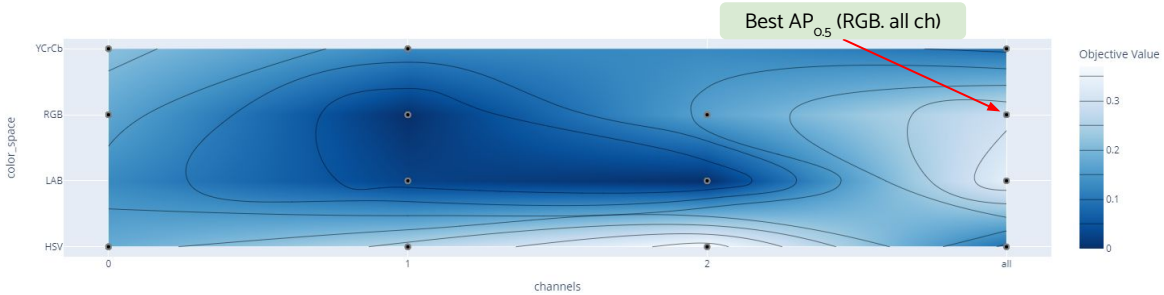
Task 4: Color (Team 2) - [1/2]

Alpha	Rho	Color space	Channel (ch)	Voting	Time [s]	Precision	Recall	F1-score	$AP_{0.5}$	IoU
4	0.02	gray	-	-	-	.8253	.4731	.6014	.4006	.6680
5.5	**0.59**	**RGB**	**all**	**simple**	**410**	**.8652**	**.5854**	**.6983**	**.4981**	**.6516**
2	0.07	LAB	all	unanimous	398	.5261	.2838	.3687	.1918	.5337
3	0.20	HSV	1 (S)	-	178	.4336	.2171	.2893	.1714	.4513

In all the color experiments, the following **post-filtering techniques are applied**:

- Non-Maximum Suppression (NMS)
- Morphological filtering
- Temporal filtering

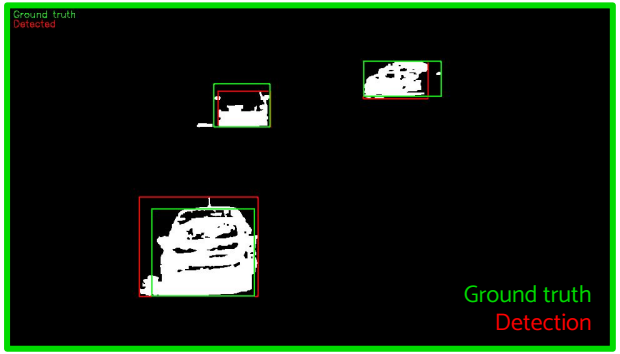
First row (baseline) corresponds to the best adaptive gaussian parameters found in task 2.



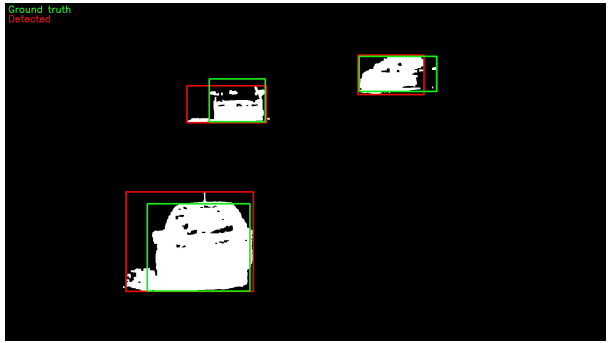
We conducted an Optuna search this time with color space, channel (1st, 2nd, 3rd, or all), and voting system (simple or unanimous) as hyperparameters. Voting is used only when all channels are selected. In simple voting, a majority of channels decides whether a pixel is foreground or not. In unanimous voting, all channels must agree to decide a pixel is foreground. Due to time constraints, we performed 50 runs of the Optuna study on the first 400 frames of the test partition. However, the best configurations found for each color space (in the table) were computed on the whole partition.

We **searched for the rho and alpha again**, as they **may have different impacts on the final result depending on the color space**. Our search results showed that using 3 Gaussians in the RGB space with simple voting system achieved the best results (interestingly, with a high rho), with approximately 10% improvement in AP w.r.t. the baseline. **The model may have benefited from the color information to differentiate the cars from the mostly gray background**. We were surprised that the **HSV did not perform well**, **despite being more robust to brightness changes and shadows**. We believe that post-processing techniques should also be adjusted for different color spaces, as they may alter critical information.

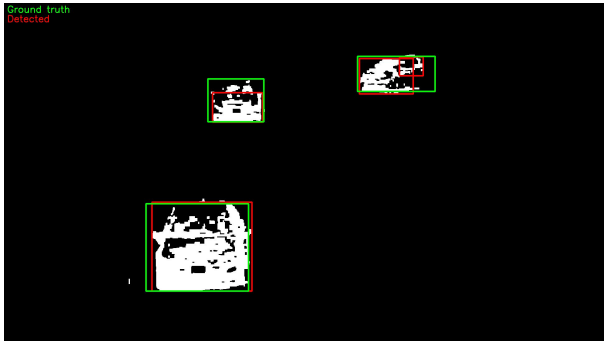
Task 4: Color (Team 2) - [2/2]



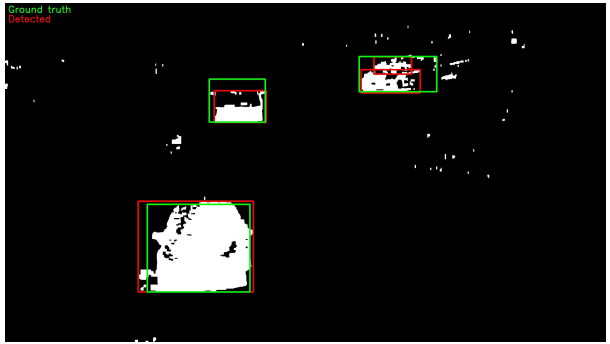
Baseline
(gray)



RGB
(all ch, simple vote)



LAB
(all ch, unanimous vote)



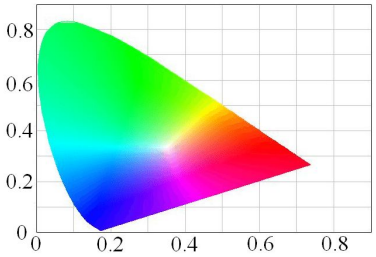
HSV
(ch 1)

We can observe that the RGB is able to improve the mAP because it handles small objects a bit better, despite showing a few false positives (bikers and noise) and detecting shadows.

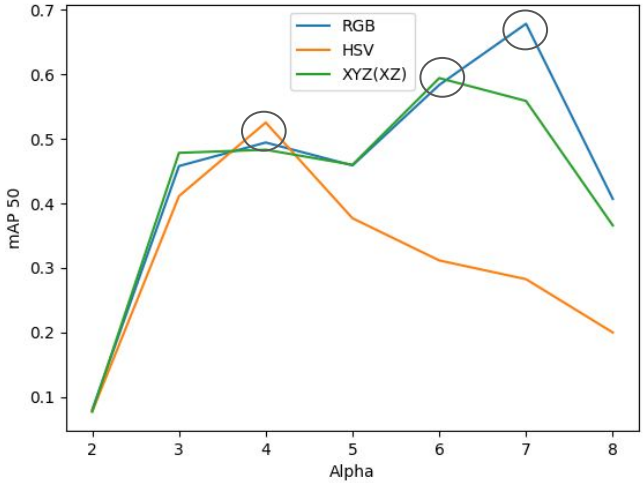
The other color spaces display more noise or, in some cases, unconnected components. Nonetheless, they do not detect shadows. HSV with channel 1 (saturation), for instance, rely on the saturation of the scene, which is mostly gray and non-saturated.

Again, the post-processing, such as the morphological operations, should be adjusted for each color space to achieve the optimum performance. In addition, we think that setting aspect ratio priors to discard false positives like bikers or other non-car objects would be useful.

Task 4: Color (Team 3) 1/2



XYZ representation



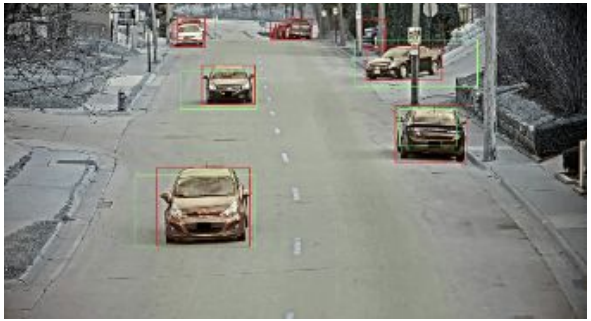
Adaptive modelling using colors, with **ρ** = 0.005

- In case of XYZ, Y corresponds to the relative luminance and X and Z carry information about how the cones in the human eye respond to color frequencies. While modelling the background we only use XZ expecting not to obtain shadows in the predictions.
- We used **adaptive** modelling in combination of different colors spaces to obtain a model of the background.
- Using RGB gives **better** performance than section 2, even without parameter random search. We can expect even better results with random search.
- In next slide we show the predictions using the **optimum** alphas for every colour space.

Task 4: Color (Team 3) 2/2

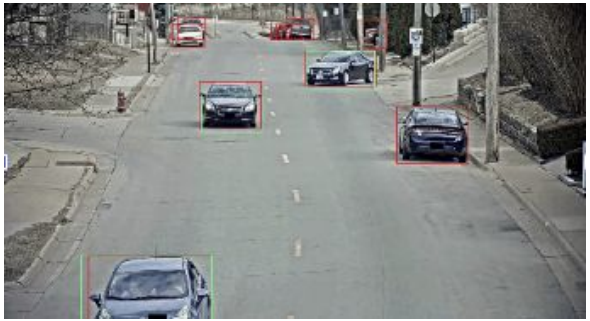
Ground truth

Prediction



RGB

HSV



XYZ (XZ)

- Although RGB detects **shadows** in the predictions, it obtained the **best** mAP. It partially detects the parked car that is nearest to the camera.
- The HSV is prone to **instability** because the hue component can introduce significant **noise** due to the way it is computed and can only detect objects **near** to the camera.
- As we expected, XYZ doesn't detect **shadows** and also seems the most **stable** and accurate when it predicts but it has many **false negatives** than RGB.

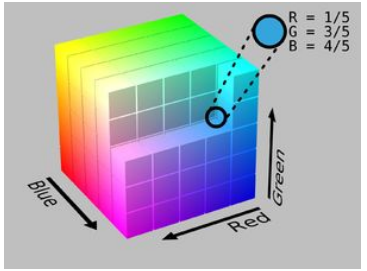
Task 4: Color (Team 4) - [1/2]

In our previous tasks, since we are only using the grayscale channel, car shadows appeared in the foreground segmentation as their values were too different to the ones of the background. Because of this, we tried to **use perceptual colourspaces** that keep the values in a separate channel, such as LAB (Luminance channel) and HSV (Value channel). This way, we can exclude them from the background model.

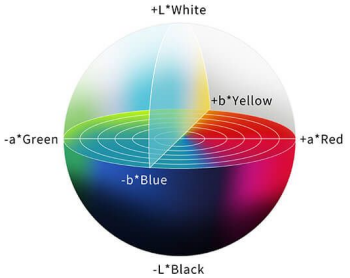
In particular, we tried to compute the masks modelling the background using the following channels:

- A and B of the LAB colourspace
- H and S of the HSV colourspace.
- R, G and B of the RGB colourspace. This colourspace does not take into account the shadow-removal approach, but we included it in our experiments to see how much it would differ from using perceptual spaces.

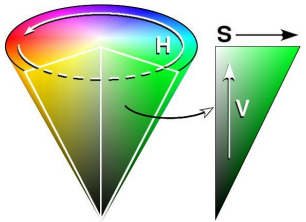
RGB: Channels are red, green and blue



LAB: Channels are luminance, a and b



HSV: Channels are the hue, saturation and value.



RGB mask

LAB mask

HSV mask

Task 4: Color (Team 4) - [2/2]

RGB boxes and IoU

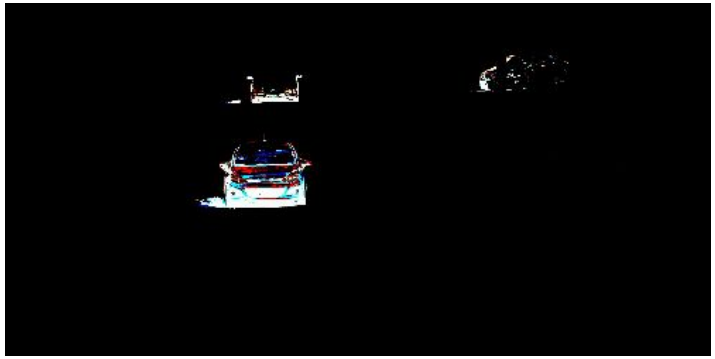
LAB boxes and IoU

HSV boxes and IoU

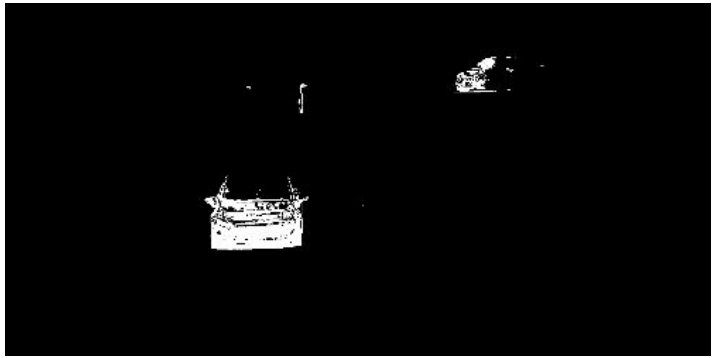
As we can see,

We did not have enough time to optimise the rho and alpha values, so these results might be suboptimal.

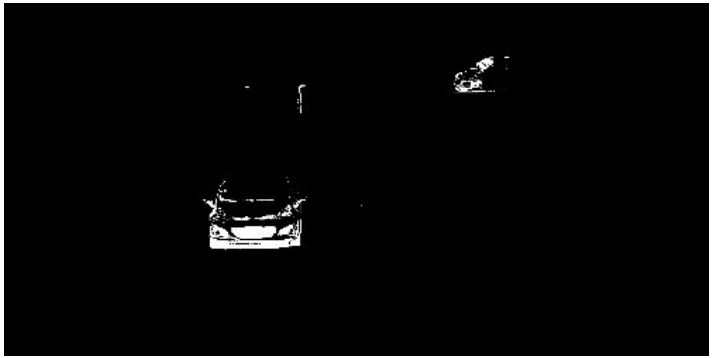
Task 4: Color (Team 5)



Left-top gif: foreground estimated in each color channels individually. Colorful pixels represents areas, where motion was detected, but **not in every** channel. White pixels represent areas where motion is present in **all** channels.



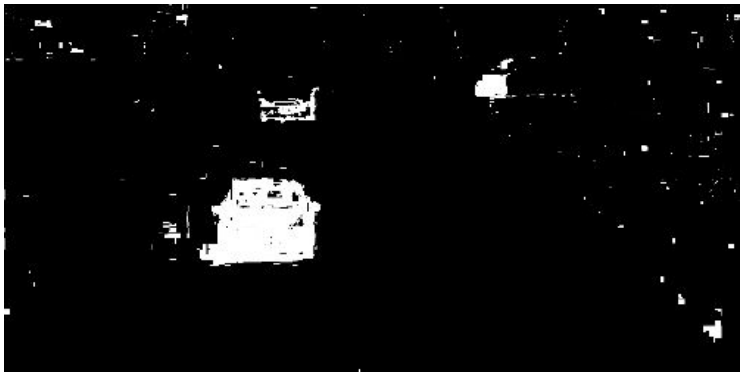
Pixel is marked as a foreground if motion is detected in **any** channel
→ better Recall, worse Precision



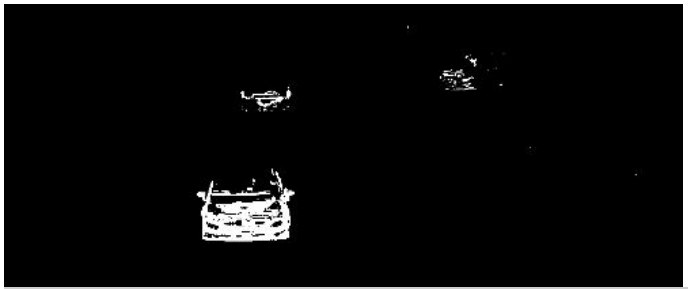
Pixel is marked as a foreground if motion is detected in **all** channels
→ worse Recall, better Precision

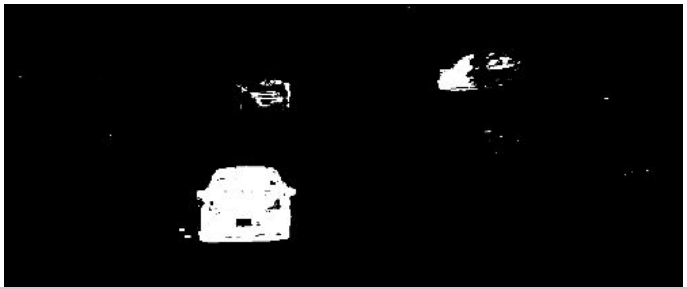
Task 4: Color (Team 5)

Method	Alpha (optimized)	mAP	IoU
3-channel estimation. Foreground if motion in **any** channel.	10	0.2256	0.1929
3-channel estimation. Foreground if motion in **all** channels.	10	0.1562	0.1664
Hue (from HSV)	3	0.0249	0.0671
A (from LAB)	2	**0.3190**	**0.2704**
B (from LAB)	2	0.1962	0.2055



Gif above: a foreground estimated from a Hue channel. It appears very rough and “blocky”, probably due to the chroma information loss after a compression



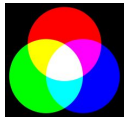


Gifs to the left: a foreground by A-, and B-channels of LAB color model.

NOTE: color-based estimation could show its advantage on uncompressed videos. Compressed videos lead into a severe chroma-information loss.

Task 4: Color (Team 6) [1/2]

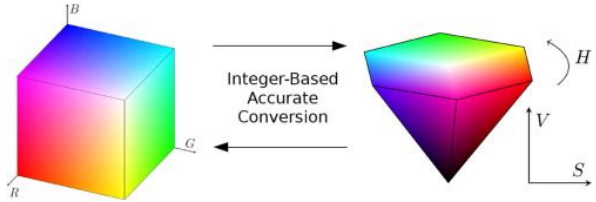
For detecting moving cars in a fixed camera recording of a street using Gaussian modeling, we can try different color spaces and channels to find the best option that works on our specific case. Here we state the four color spaces and the channels we will consider.



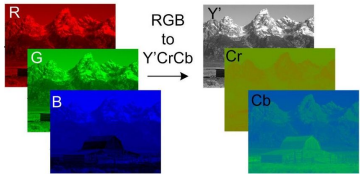
RGB: If the background is uniform in color, any of the channels can be used. However, if the background contains some texture or variations in color, it may be better to use a combination of channels to capture the details of the scene. Seems that all **three channels (R, G, and B)** are used to model the background.

YUV: The **chrominance** channels **(U and V)** are used to model the background, as they are less sensitive

HSV: The **saturation (S)** channel is used to model the background, as it is less sensitive to changes in brightness and more sensitive to changes in color caused by moving objects.



YCrCb: We use the **chrominance** channels **(Cr and Cb)** to model the background, as they are less sensitive to changes in brightness and more sensitive to changes in color caused by moving objects.



Task 4: Color (Team 6) [2/2]

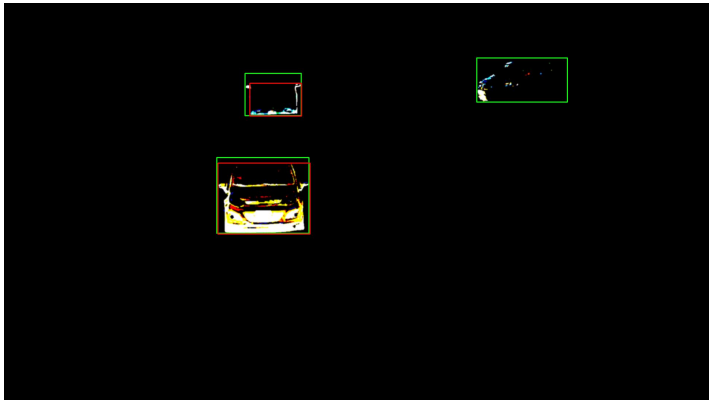
We ran several experiments with the Gaussian Model by taking into account the color channels we explained at the previous slide. Once estimated the foreground, we perform the same denoising as the first task and then, to compute the binary mask to detect contours (see task 1), we compute the mean between the channels.

RGB performs slightly better than the other color spaces. Also, achieves higher scores in terms of mIoU than the grayscale version of gaussian modelling. In this video we can see that using RGB allows to segment almost perfectly the silhouette of the objects that are near to the camera. We see almost no noise outside of the cars we are tracking. Far away objects are still challenging using colorspace.

Color Space	Gaussians	α	mAP	mIoU
RGB	3	5	0.42	0.28
RGB	3	6	0.35	0.29
HSV	1	5	0.39	0.23
HSV	1	3	0.23	0.28
YCrCb	2	6	0.41	0.24
YCrCb	2	5	0.41	0.28
YUV	2	6	0.41	0.24
YUV	2	5	0.41	0.28

Table with best mAP and mIoU per color space values using alpha = 3, 4, 5, 6, 7

RGB (3 gaussians) alpha = 6



Predicted Ground truth

We considered always the same alphas for all the channels. A possible future research could be to consider an alpha for each channel. Also, instead of computing the mean to construct the binary mask, we could consider other voting techniques.
It should be noted that using more than one channel (RGB, YCrCb, YUV) increases the computation expense considerably.

Task 4: Feedback

	feedback
Team 1	Several color spaces tested. Good! Missing comparison with gray.
Team 2	Comparison with gray provided. Good explanation and justification on the color channels used.
Team 3	Good explanation and justification on the color channels used. Missing comparison with gray.
Team 4	“all teams” table not filled. Also figures missing. (do not leave task reporting to the last minute) Reference the figures if not yours.
Team 5	Missing color channel info, are you using RGB? Discussion on how to select foreground pixel from multiple gaussians. Did you try AB jointly? why not? Missing comparison with gray.
Team 6	Good explanation and justification on the color channels used. No comparison with gray. Bonus point for talking about possible improvements. Bonus point for discussion about compression.