Master in Computer Vision Barcelona

Project Module 6 Coordination

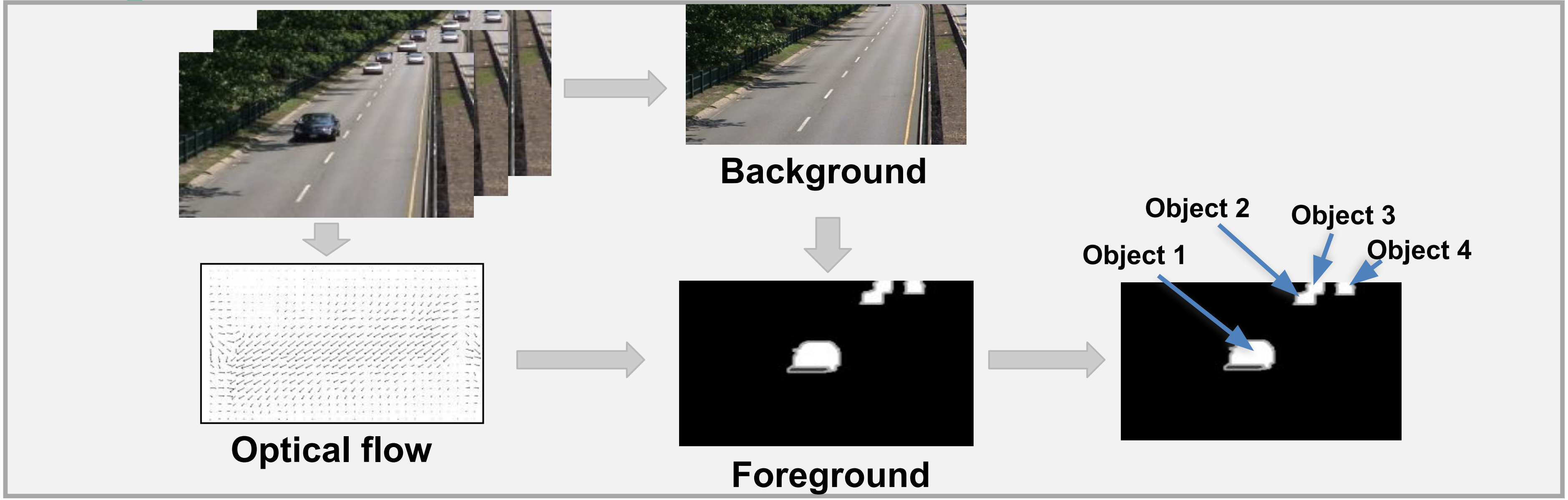
Video Surveillance for Road Traffic Monitoring

J. Ruiz-Hidalgo / X. Giró

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

Week 1: Instructions

Project Schedule

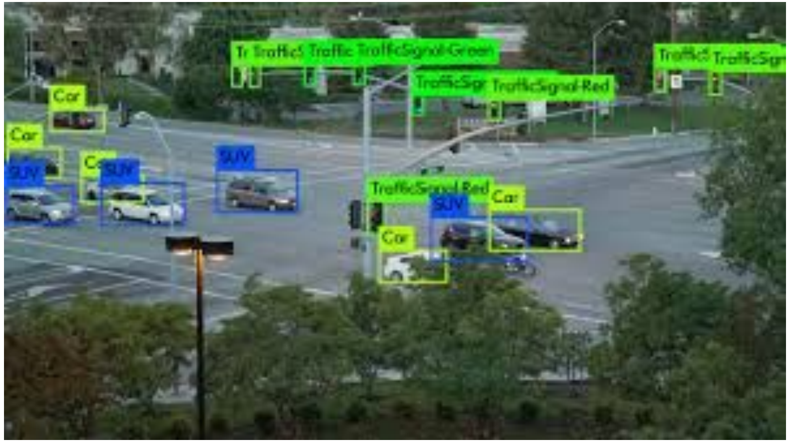


Week 1	Week 2	Week 3	Week 4	Week 5	Week 6
• Introduction • DB • Evaluation metrics	• Background estimation • Stauffer & Grimson	• Object Detection • Tracking	• Optical flow • Tracking	• Multiple cameras • Speed	• Presentation workshop

Master in
Computer Vision
Barcelona

Goals Week 1

- Understand and become familiar with the programming framework used in the project.
- Learn about the databases to be used.
- **Implement the evaluation metrics and graphs used during the module.**
- Read / write sequences of images and associated segmentation ground truth.



OpenCV

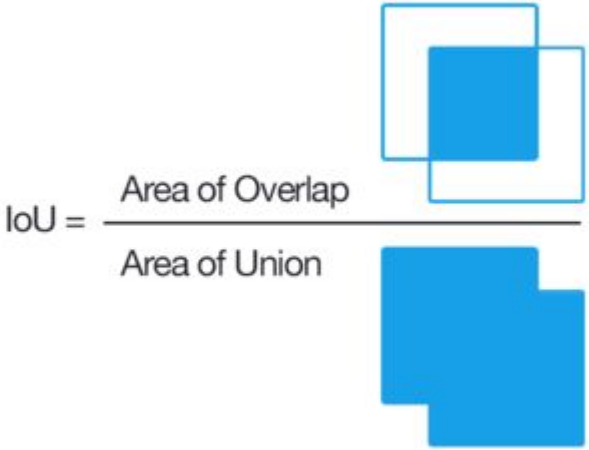
python™

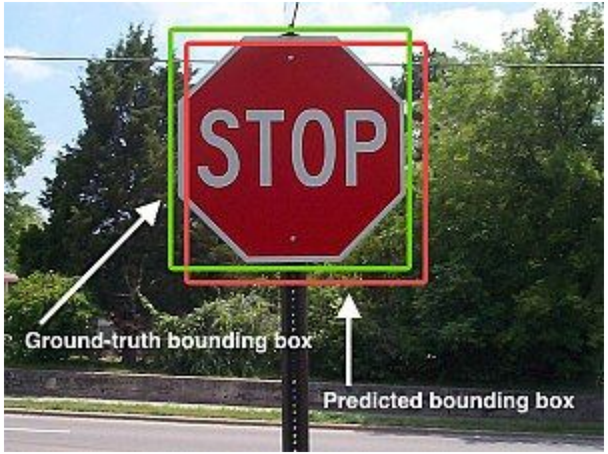
Tasks

- Task 1: Detection metrics.
- Task 2: Detection metrics. Temporal analysis.
- Task 3: Optical flow evaluation metrics.
- Task 4: Visual representation optical flow.

Evaluation metrics: Intersection over Union (IoU)

- aka Jaccard index
- Size of intersection divided by the size of the union
- Evaluate localization





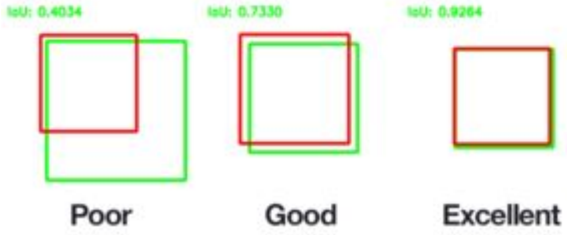
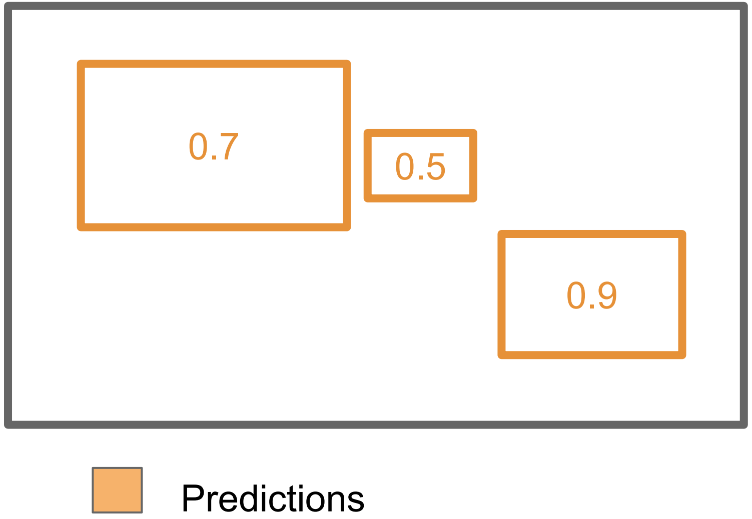


Figure: Pyimagesearch

Metric: Average Precision (AP) for Object Detection

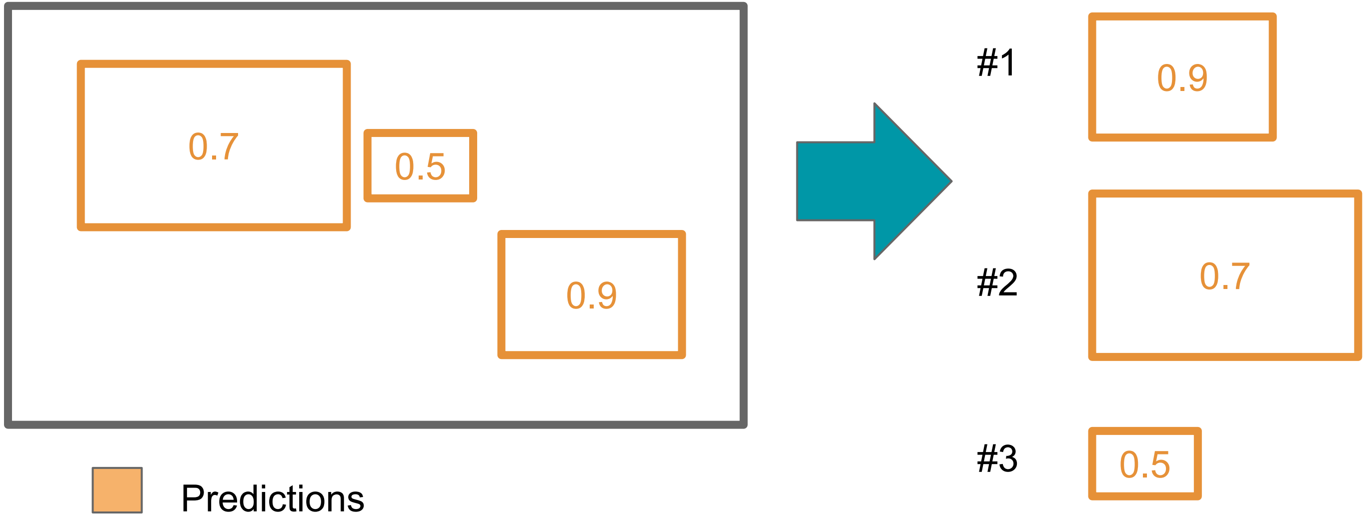
Consider the case in which your object detection algorithm provides you:

- Coordinates for each bounding box.
- A confidence for each bounding box



Metric: Average Precision (AP) for Object Detection

Rank your predictions based on the confidence score of your object detection algorithm:

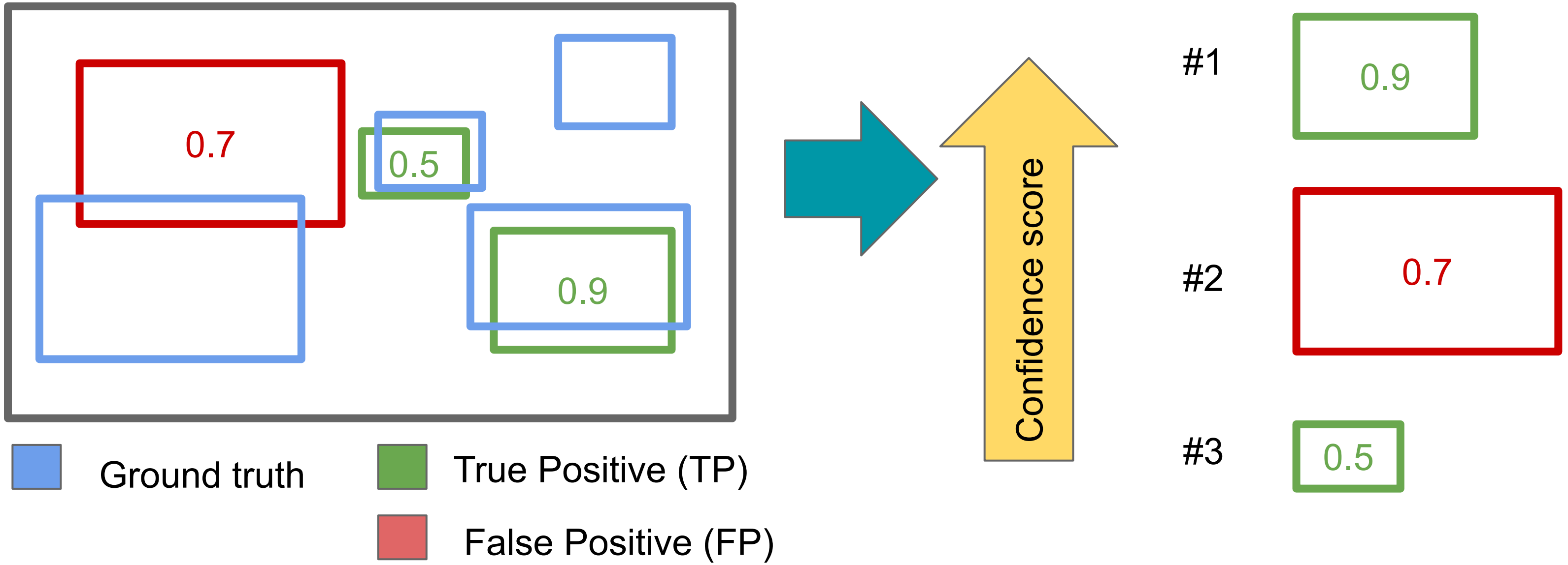


Metric: Average Precision (AP) for Object Detection

Set a criteria to identify whether your predictions are correct.

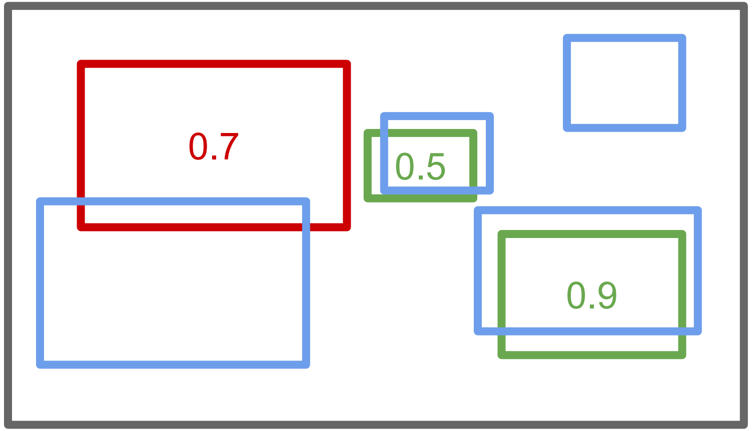
Typically, a minimum IoU with respect to the bounding boxes from the ground truth annotation.

- **In your work, consider IoU > 0.5. This is referred as $AP_{0.5}$.**
- Other popular options: $AP_{0.75}$, or a range of IoU [0.5:0.95] in 0.05 steps
- Each GT box can only be assigned to one predicted box.



Metric: Average Precision (AP) for Object Detection

Compute the point of the Precision-Recall curve by considering as decision thresholds (Thr) the confidence scores of the ranked detections.

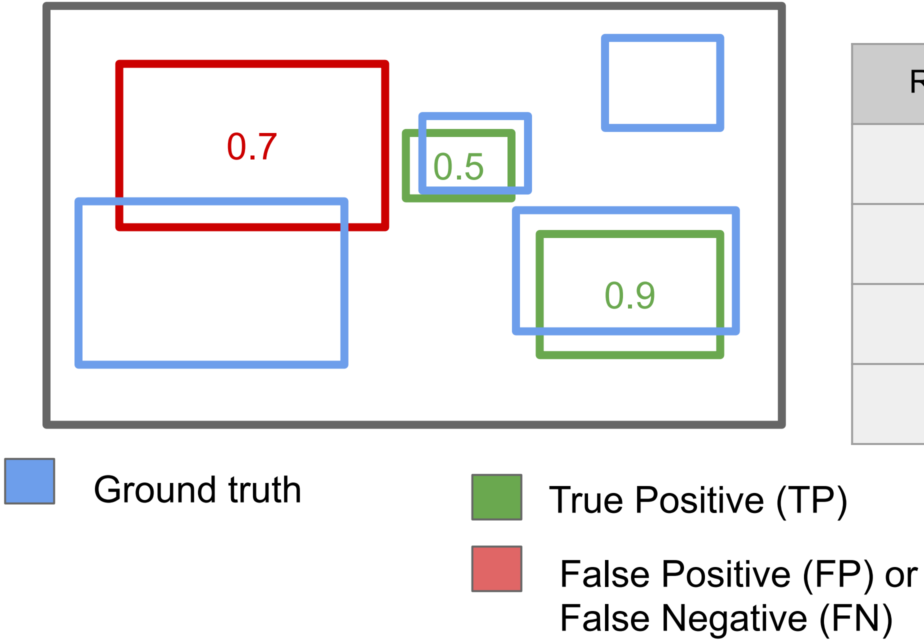


Rank	Correct ?
1	True
2	False
3	True

Threshold	Precision	Recall
0.9	1/1	1/4
0.7	1/2	1/4
0.5	2/3	2/4

Metric: Average Precision (AP) for Object Detection

In the object detection case, in which GT objects may never any predictions, we may consider that trying to find the missing objects with an infinite amount of object proposals would drop precision to 0.0, but would eventually find all objects, so recall would be 1.0

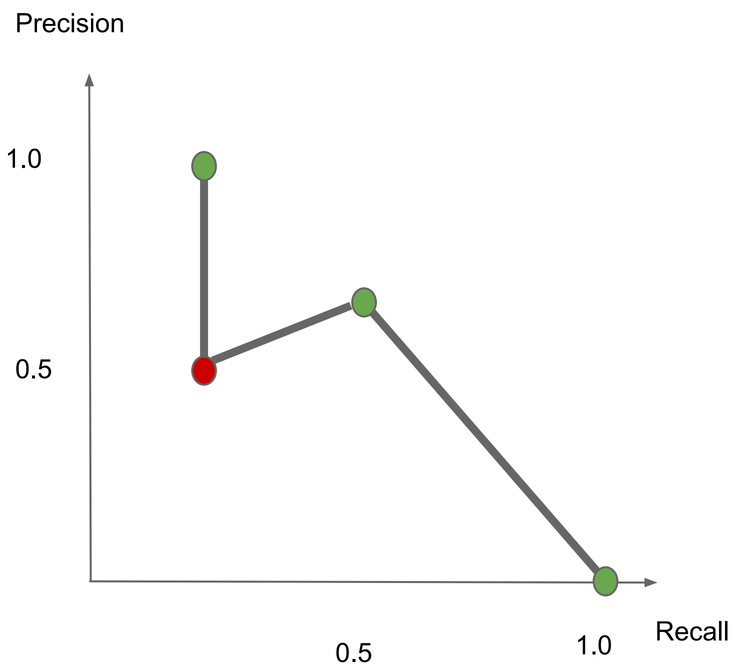


Rank	Correct ?
1	True
2	False
3	True
∞	True(s)

Threshold	Precision	Recall
0.9	1/1	1/4
0.7	1/2	1/4
0.5	2/3	2/4
0.0	$\simeq 0$	1

Table inspired by: Johnatan Hui, [“mAP (mean Average Precision) for Object Detection”](#) (Medium 2018)

Metric: Average Precision (AP) for Object Detection

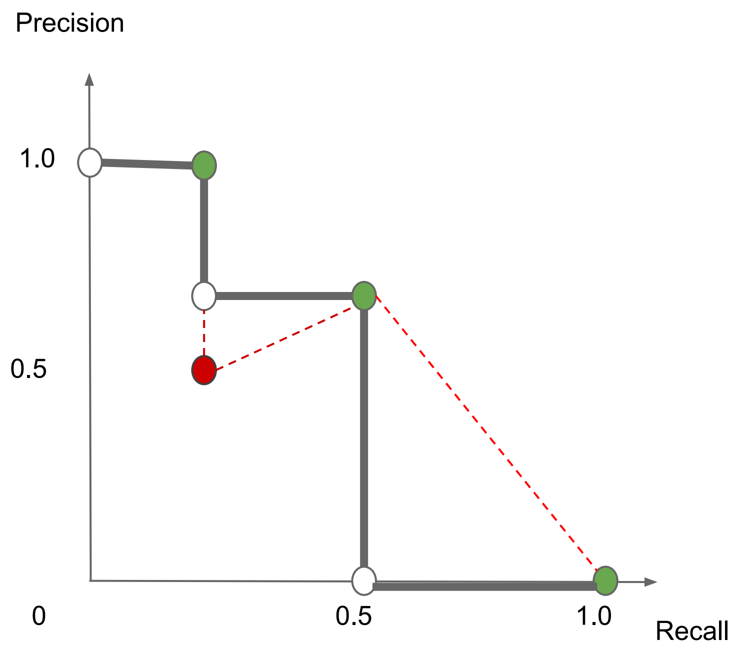


Rank	Correct ?
1	True
2	False
3	True
∞	True(s)

Threshold	Precision	Recall
0.9	1/1	1/4
0.7	1/2	1/4
0.5	2/3	2/4
0.0	≃ 0	1

Metric: Average Precision (AP) for Object Detection

“The precision at each recall level r is interpolated by taking the maximum precision (...) for which the corresponding recall exceeds r.” (from **Pascal VOC**) [ref]



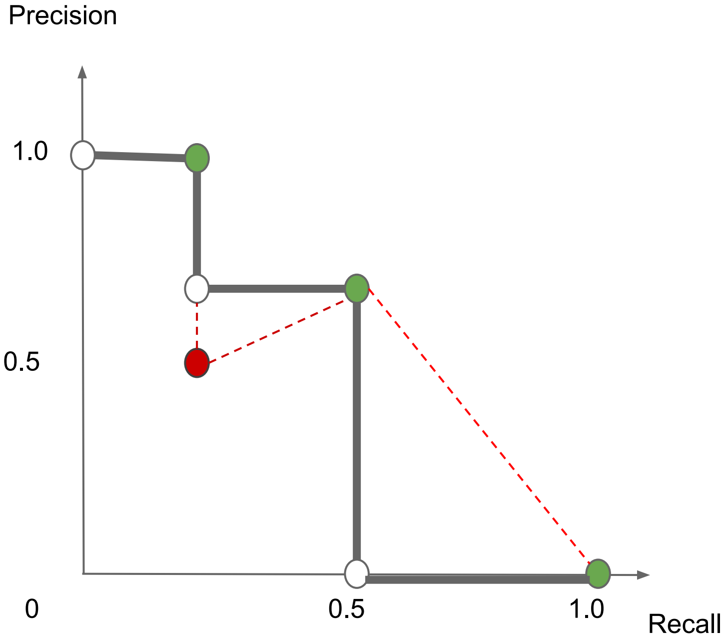
Rank	Correct ?
1	True
2	False
3	True
∞	True(s)

Threshold	Precision	Recall
0.9	1/1	1/4
0.7	1/2	1/4
0.5	2/3	2/4
0.0	≃ 0	1

[ref] Everingham, Mark, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. "The Pascal Visual Object Classes (VOC) challenge." IJCV 2010.

Metric: Average Precision (AP) for Object Detection

Actually, not all PR pairs need to be computed because AP for object detection only requires the PR pairs related to True positives:



Rank	Correct ?
1	True
2	False
3	True
∞	True(s)

Threshold	Precision	Recall
0.9	1/1	1/4
~~0.7~~	~~1/2~~	~~1/4~~
0.5	2/3	2/4
0.0	≃ 0	1

Metric: Average Precision (AP) for Object Detection

- AP approximates the area of the PR curve.
- There are different methods for this approximation that may cause inconsistencies between implementations.
- Popular ones
  - **(ours) “the mean precision at a set of 11 equally spaced recall levels [0, 0.1, ...1]”**
  - “weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight” (scikit-learn).

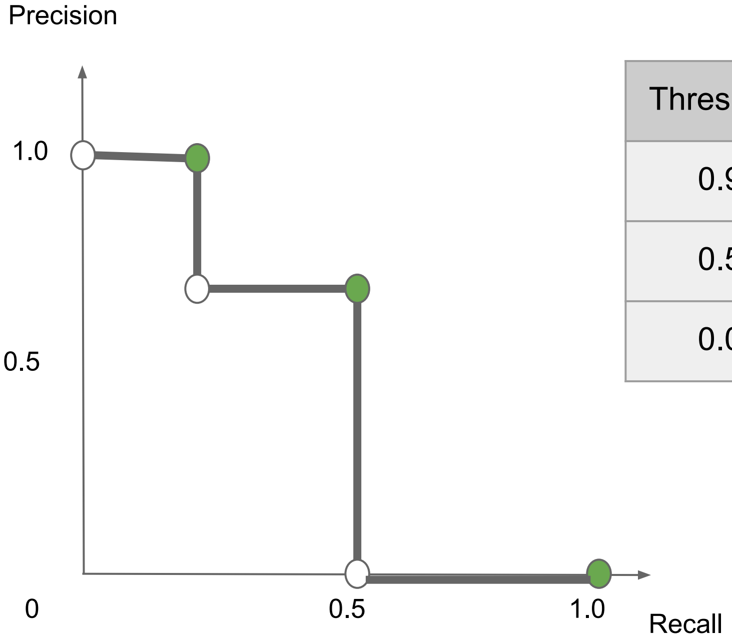
$$\mathrm{AP} = \sum_n (R_n - R_{n-1}) P_n$$

[ref] Everingham, Mark, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. "The Pascal Visual Object Classes (VOC) challenge." IJCV 2010.

Metric: Average Precision (AP) for Object Detection

In our work, we adopt the approach from **Pascal VOC**:

- AP is "the mean precision at a set of 11 equally spaced recall levels [0, 0.1, ...1]"

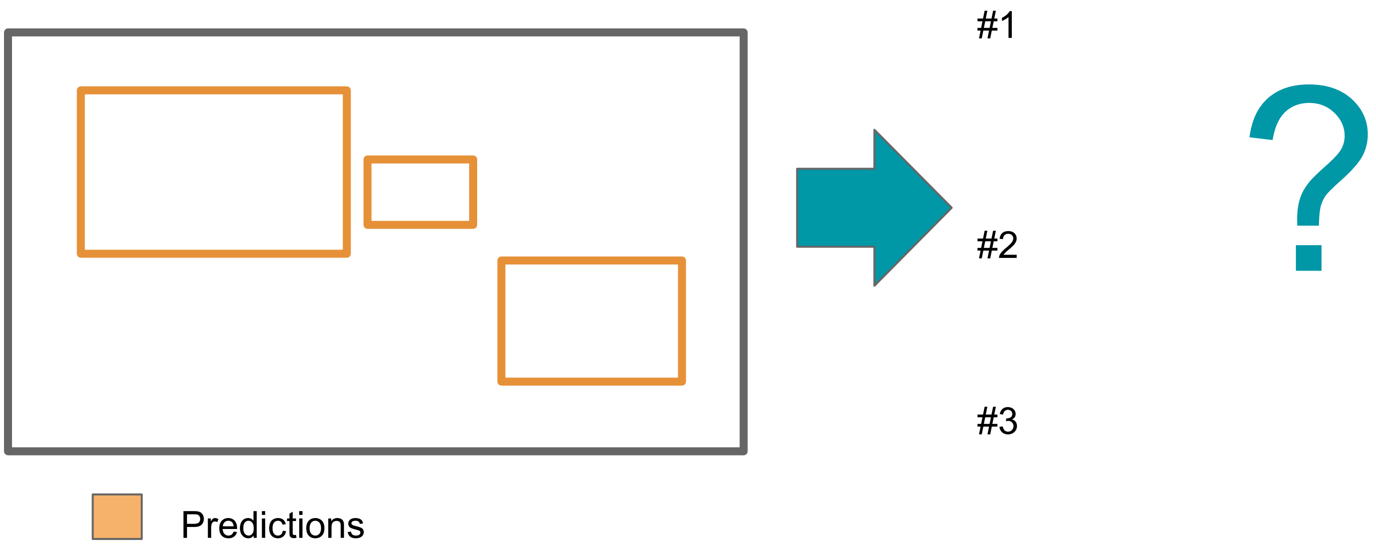


Threshold	Precision	Recall
0.9	1/1	1/4
0.5	2/3	2/4
0.0	≃ 0	1

Recall	Precision
0.0	1.00
0.1	1.00
0.2	1.00
0.3	0.67
0.4	0.67
0.5	0.00
...	0.00
1.0	0.00
AP	0.39

Metric: Average Precision w/o confidence scores

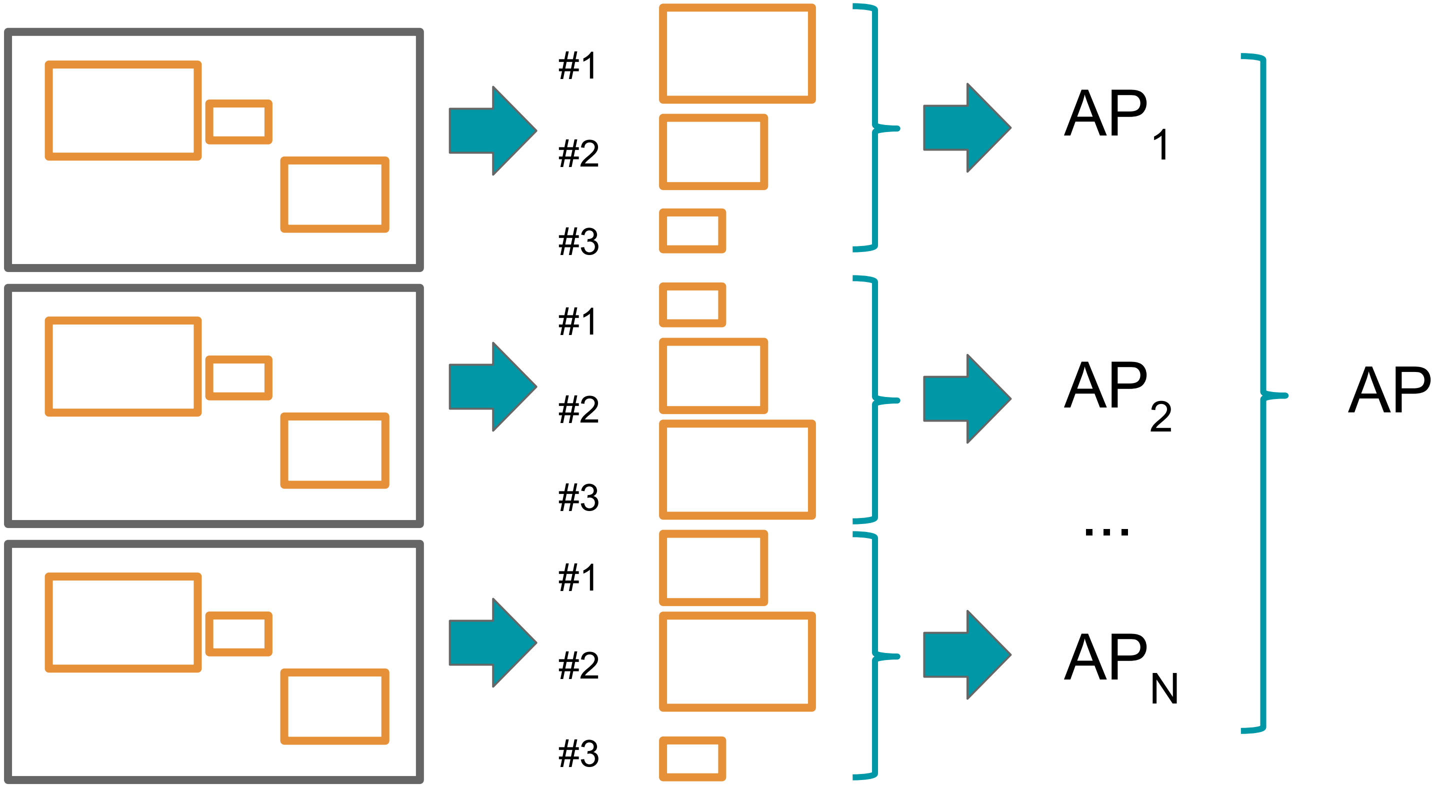
What if your object detection algorithm does not provide any confidence score ?



Metric: Average Precision w/o confidence scores

If your object detection algorithm does not provide any confidence score:

- Generate N random ranks (eg. N=10) and average your metrics across these N runs.



Evaluation metrics: mean Average Precision (mAP)

In the cases of multiple Q classes (eg. car, bike, person…), the mAP averages across the AP(q) of each class:

$$MAP = \frac{1}{|Q|} \sum_{q=1}^{Q} AP(q)$$

- Further readings:
  - Tarang Sangh, [“Measuring Object Detection models — mAP — What is Mean Average Precision?”]() (Medium 2018)

Evaluation metrics: Average Precision (AP)

You can obtain implementations for this Average Precision for Object Detection from:

Detectron 2
(use this one for the project)

TensorFlow

Microsoft CoCo dataset API
pycocotools

T1 General instructions

- Use ground truth from AI city challenge seq S03 - C010
  - Download data for this specific sequence from: UAB Campus Virtual / M6 / Project Materials / AI City Data
- Use an annotated file (thanks to team 1 - 2018/2019) from: UAB Campus Virtual / M6 / Project Materials / Annotation
  - AICity Challenge annotations only have moving objects
- Use ffmpeg to extract frames from the video
- Do **not** consider IDs (tracking) and use single class (classification).
- Only consider “**car**” class for measures (both bikes and cars are annotated)

T1.1 IoU & mAP for (ground truth + noise)

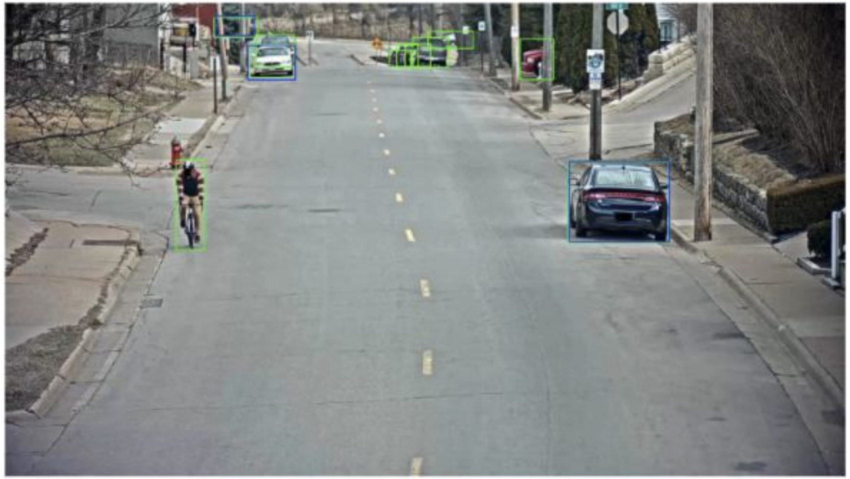
- Generate noisy annotation to the ground truth data by:
    - Adding noise to the size and position of bounding boxes
    - Introduce probability to generate/delete bounding boxes
- Compute the IoU and mAP.
- Study the effect of the parameters governing the noise in your results.

T1.1 IoU & mAP for (ground truth + noise)

Team 5 (2019/2020)

Random frame

stdev box change	probability dropout	probability generate	IoU frame	mAP frame
0.005	0.3	0.1	0.85	1



Same random frame

stdev box change	probability dropout	probability generate	IoU frame	mAP frame
0.005	0.6	0.1	0.82	0.45

Frame example using more probability of bounding boxes dropout: **the mAP of that frame decreases**

T1.1 IoU & mAP for (ground truth + noise)

Team 5 (2019/2020)

Predictions based on modifications from ground truth

random function for box change	max box change	stdev box change	random function for box generate/ dropout	probability dropout	probability generate	stdev generated box position	**mIoU**	**$\mathbf{mAP_{0.5}}$***
truncated normal (-1,1)	1000	0.005	random (0,1)	0.3	0.1	0.2	**0.802**	**0.678**
	1000	0.02		0.3	0.1	0.2	**0.467**	**0.247**
	1000	0.005		0.6	0.1	0.2	**0.785**	**0.414**
	1000	0.005		0.3	0.6	0.2	**0.696**	**0.685**

Generate noise on size and position of predicted boxes decreases both mIoU and mAP

Adding more predicted boxes decreases mIoU, but keeps mAP

This happens because new boxes usually appear in areas where there are not gt boxes

Deleting more predicted boxes decreases the mAP, but keeps mIoU

This happens because now there are more gt boxes without prediction but the ones predicted have good boxes

*** IoU threshold > 0.5**

T1.2 mAP for provided object detections

- Compute the mAP for the provided detections (mask_rcnn, ssd512, yolo3)

T1.2 mAP for provided object detections

Team	Mask RCNN	SSD512	Yolo3
1			
2			
3			
4			
5			
6			

T2 Temporal Analysis of the Results

T2 IoU vs time

- Temporal analysis of the results
  - Graph: IoU vs #frame

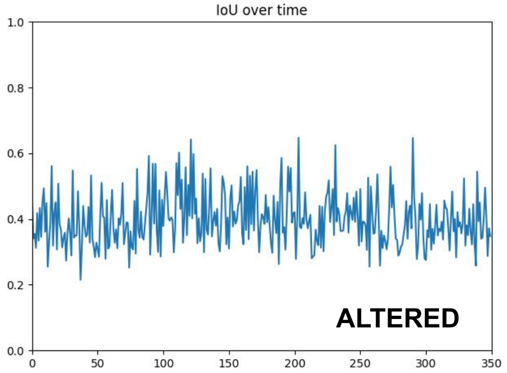
Explain &
Show WHY !

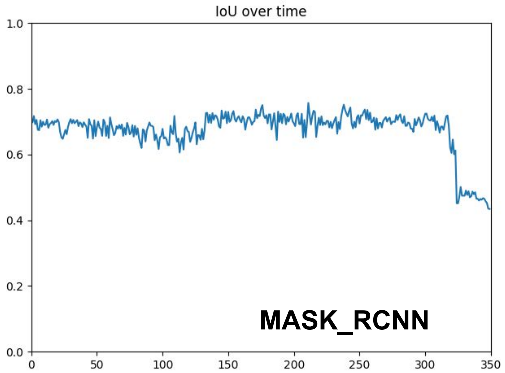
T2 IoU vs time (Team 5)

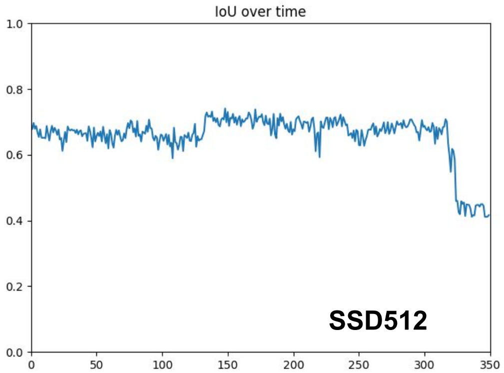
In the altered case, as the detection is randomly modified, the IoU has:

- ➢ Quite low value
- ➢ Lots of noise.

In the detection methods, IoU over time drops in the last frames due to the disappearance of most of the cars.





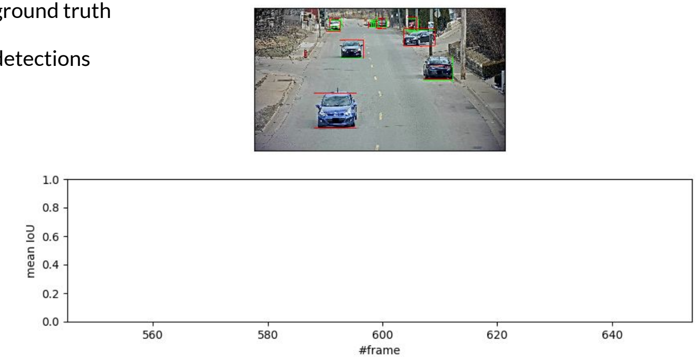


T2 IoU vs time

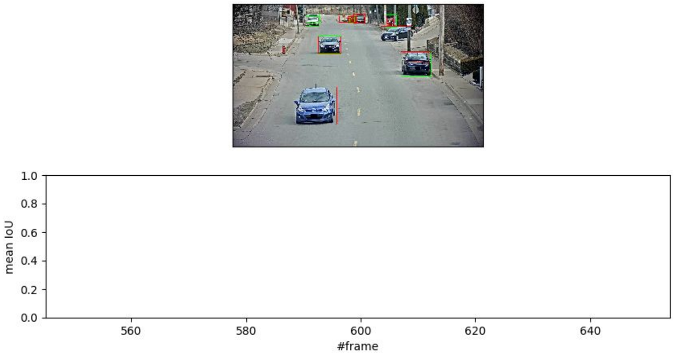
ground truth

detections

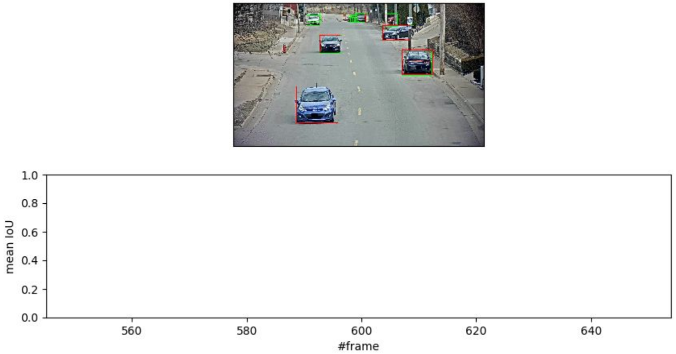
noisy annotations



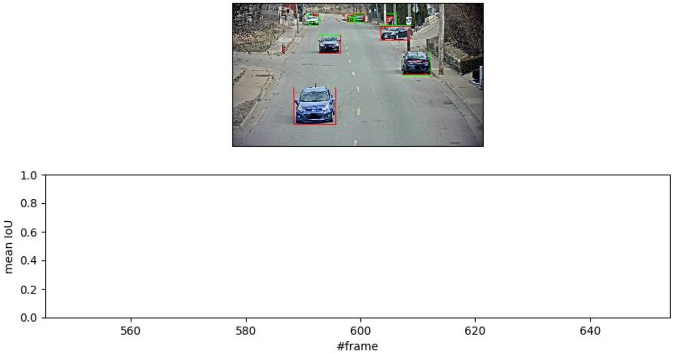
mask_rcnn detections



ssd512 detections



yolo3 detections



T3 Quantitative evaluation of optical flow

- Optical flow estimations using the Lucas-Kanade algorithm.
- Sequences 45 and 157 (image_0) from the KITTI dataset.
- Only 1 estimation / sequence (2 frames!)
- Check the KITTI website for code to read results (dense motion vectors)

GROUND TRUTH

http://www.cvlibs.net/datasets/kitti/
(Flow 2012 > Stereo / Optical flow dataset)

TEST

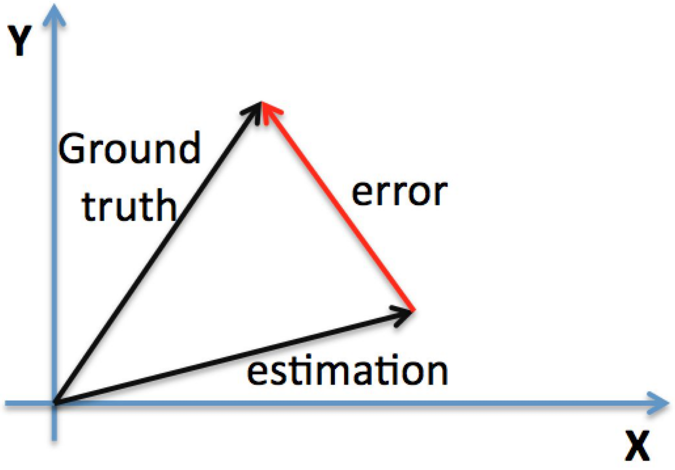
UAB Campus Virtual
M6 Video Analysis > Project Materials
> motion_data

T3.1 MSEN & T3.2 PEPN

- **MSEN:** Mean Square Error in Non-occluded areas
- **PEPN:** Percentage of Erroneous Pixels in Non-occluded areas

Consider only non-occluded areas (NOC). Indicated by *flow_noc* in the data.

Consider erroneous those pixels whose motion vector error is > 3.



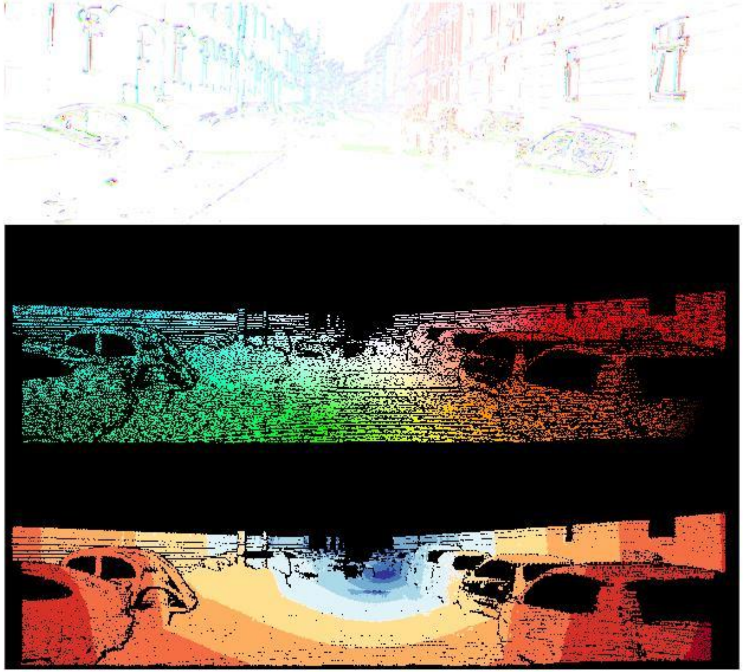
T3.1 MMEN & T3.2 PPEN

	MSEN		PEPN	
Team	**Seq 45**	**Seq 157**	**Seq 45**	**Seq 157**
1				
2				
3				
4				
5				
6				
7				
8				
9				

T3.3 Analysis & Visualizations

Discuss the obtained results and generate visualizations that help understanding them.

T3.3 Analysis & Visualizations (baselines)

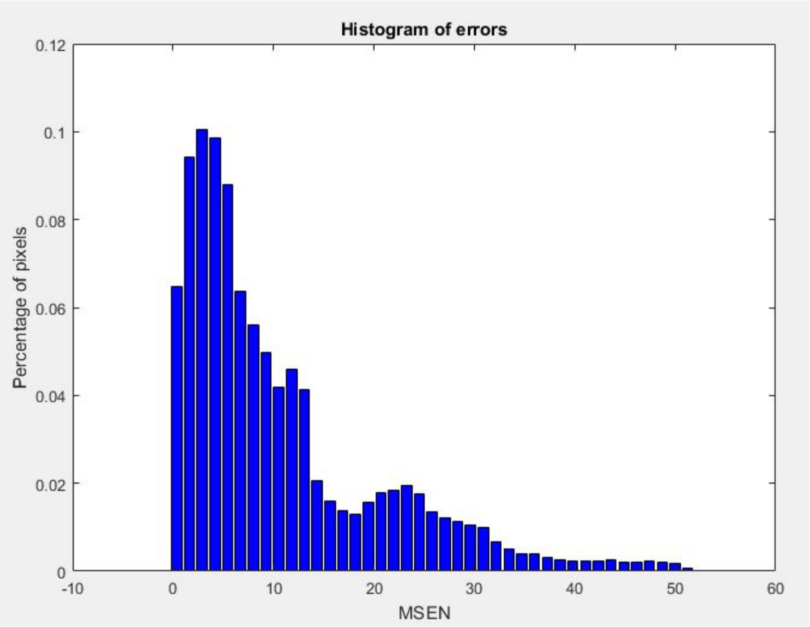


Plots for sequence 45 (estimation, ground truth and error). Reddish values correspond to higher errors.

T3.3 Analysis & Visualizations (baselines)

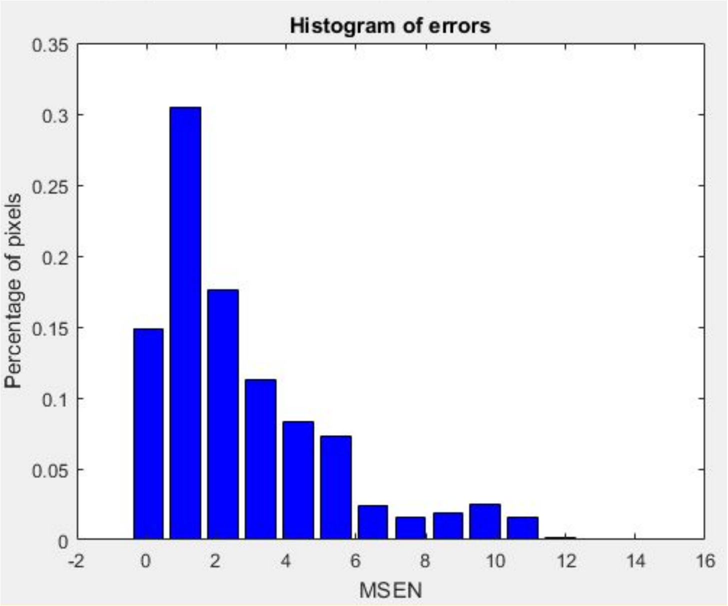
Team 1 (2017/2018)

Sequence 45



Not valid vectors are discarded

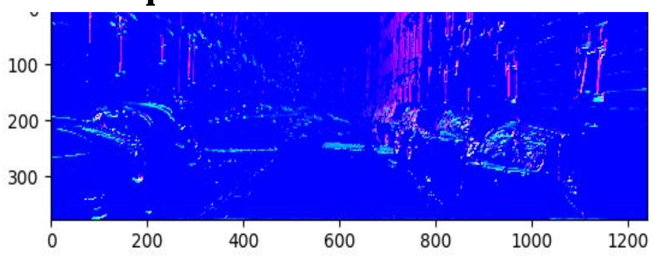
Sequence 157



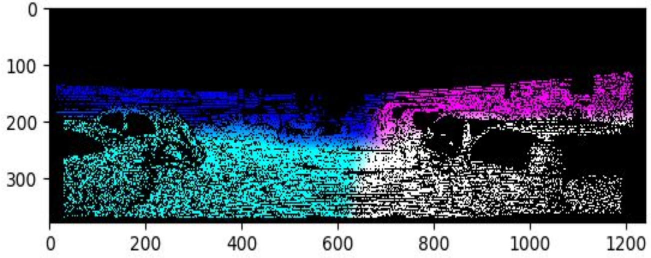
Not valid vectors are discarded

T3.3 Analysis & Visualizations (baselines)

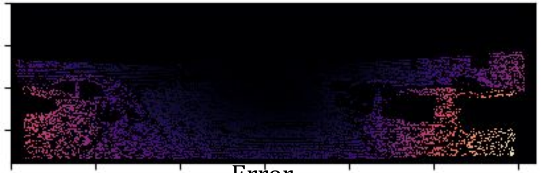
Sequence 045



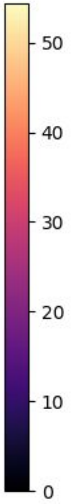
Estimated Optical flow

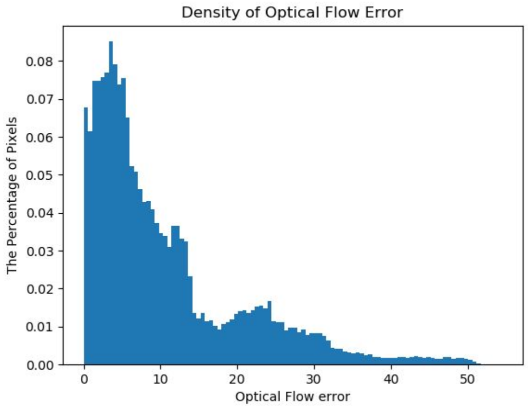


Ground Truth Optical Flow



Error

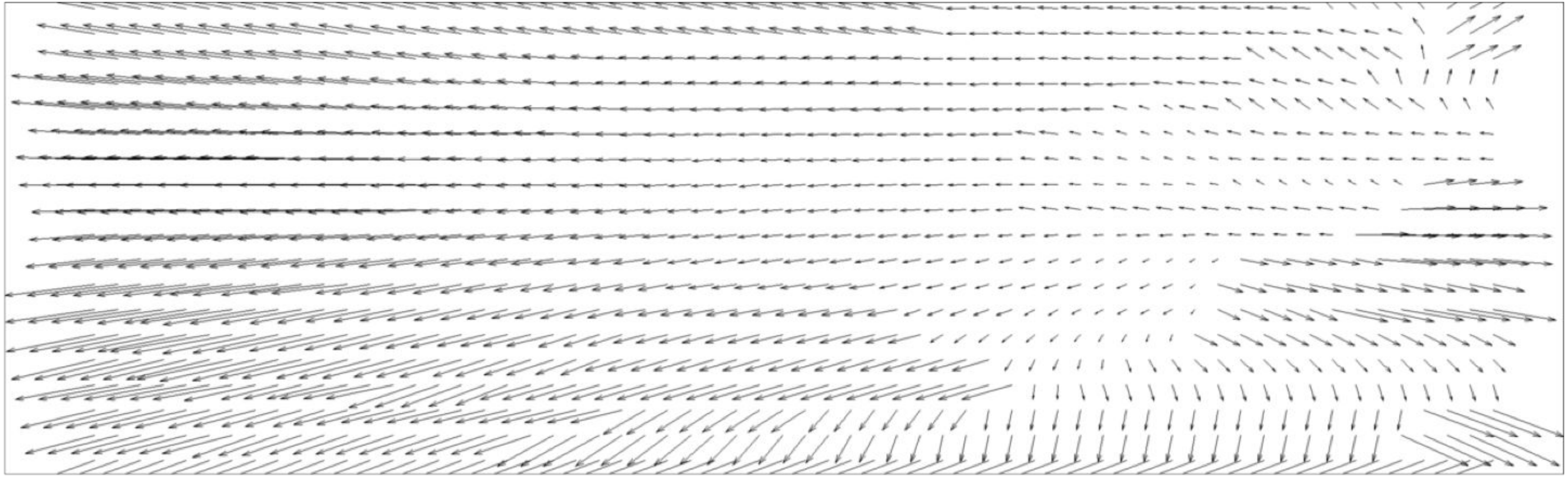




- **Inference**: The histogram above represents the Optical flow error vs the Percentage of pixels we find lesser pixels with more errors in sequence 45 compared to sequence 157. Just above the histogram represents the error image that illustrates more error occurs when objects are closer as shown in sequence 45 as there is more movement and higher disparity. We note that pixels in black in the ground-truth image correspond to 'non valid' pixels, which we use for computing the error.

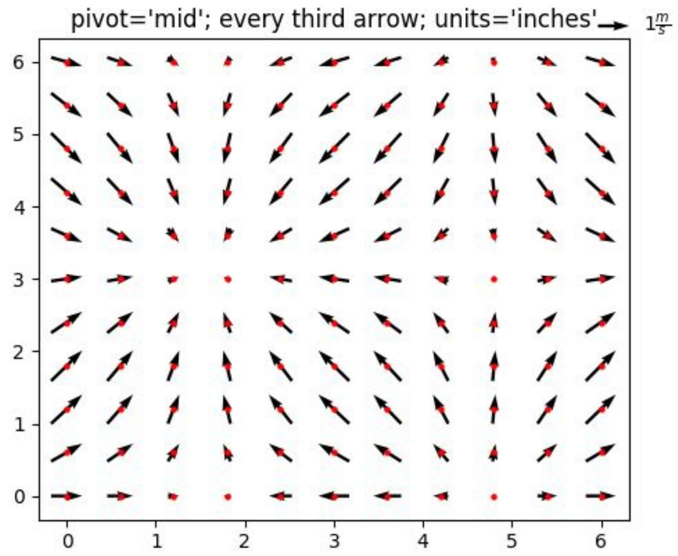
T4 Optical flow plot

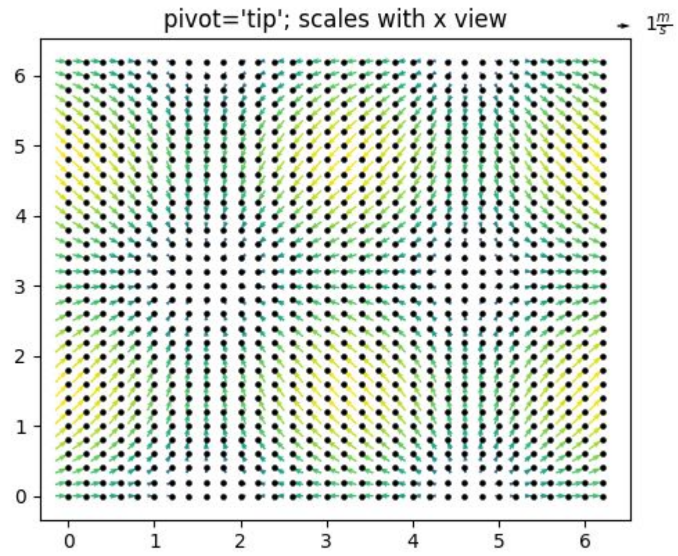
- Plot the optical flow
  - Dense representation -> too many motion vectors
  - Arrows might be confusing, not related to pixels
- Propose a simplification method for a clean visualization.



T4 Optical flow plot

- Tips on how to plot optical flow with Matplotlib
  - [Quiver demo]
  - [matplotlib.axes.Axes.quiver]





T4 Optical flow plot (baseline)

Team 5 (2015/2016)

T4 Optical flow plot (baseline)

Team 7 (2015/2016)

Figure 14. Zoom-in of the result

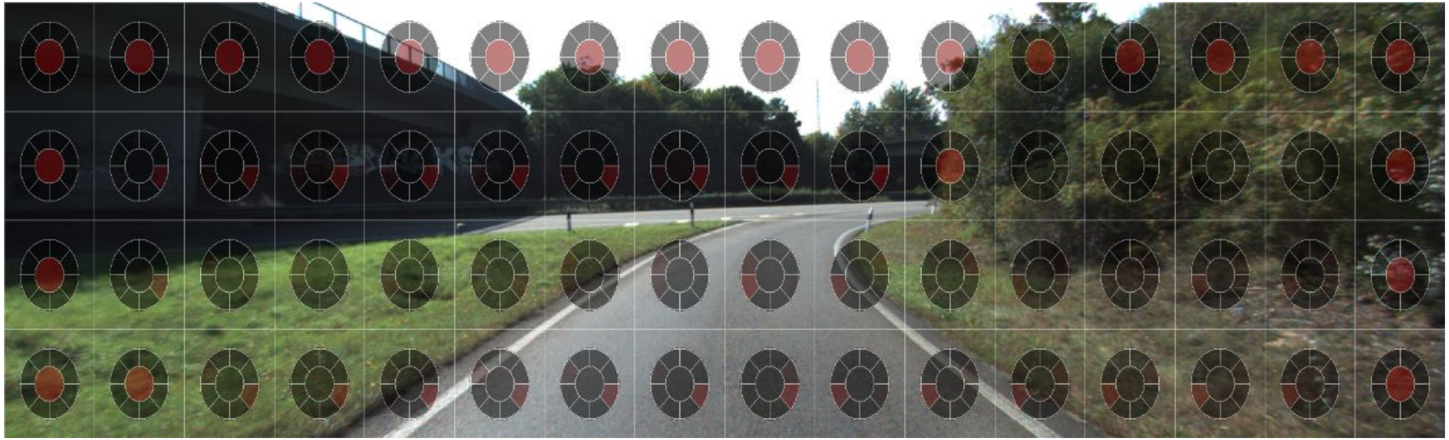
Figure 15. Results of the Task 4

T4 Optical flow plot (baseline)

Team 3 (2017/2018)

We have created a custom way of visualizing optical flow. We noticed that sometimes the arrows provided by the quiver function, although practical and intuitive, tend to obfuscate the original image. This hardens the understanding of the image, as the objects of interest are the ones that are moving and therefore obfuscated. Due to their thin shape, mixing the arrows using transparency is not a valid option, as they become harder to see.

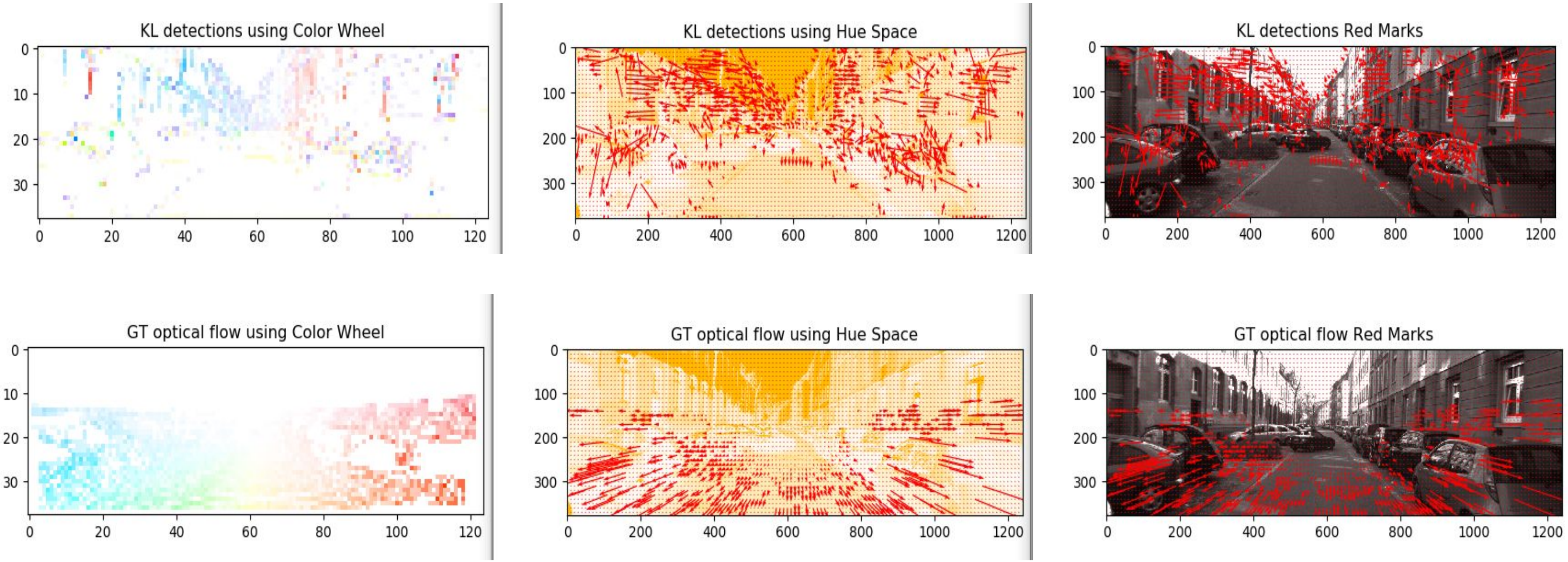
To solve this we have superposed a transparent normalized histogram of the movement over the original image. This histogram counts with 9 bins: a central one for non-moving pixels, and 2 more for each major direction. This allows a quick understanding of both the image and the motion in a single glimpse.



T4 Optical flow plot (baseline)

Team 3 (2019/2020)

- Visualization for Sequence 045

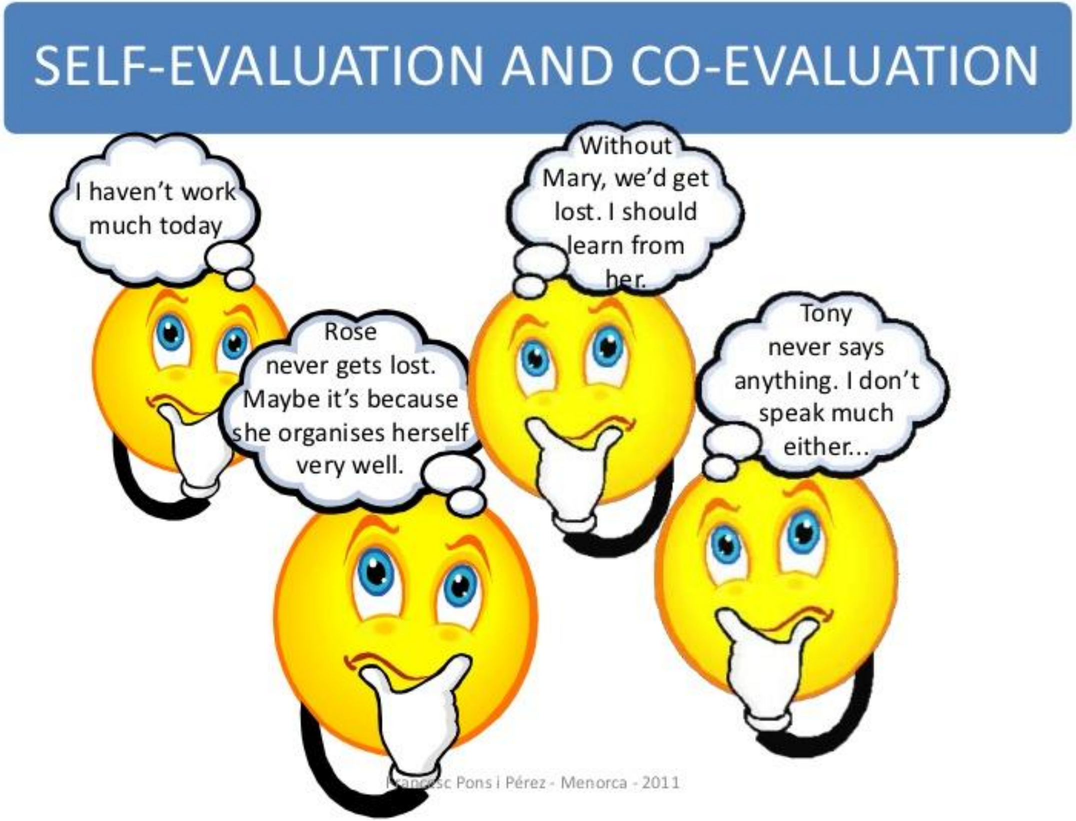


- Note: The color wheel represents the direction and color saturation for arrow magnitude.

Scoring Rubric

Task	Description	Max. Score
T1.1	IoU & mAP for ground truth + noise	2
T1.2	mAp over detections	2
T2	IoU over time	2
T3.1	MSEN	1
T3.2	PEPN	1
T3.3	Analysis & Visualizations	1
T4	Optical Flow Plot	1

Intra-team Evaluation



Grade each team member (yourself included) as:

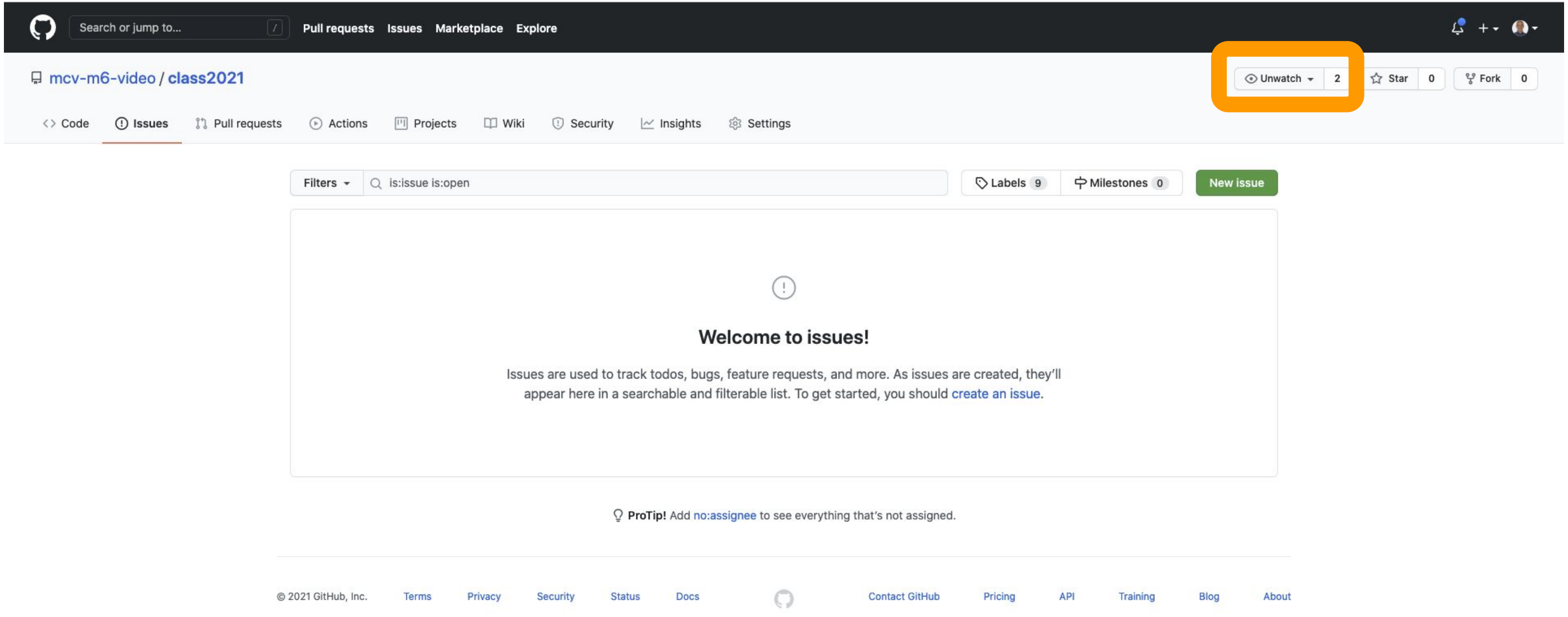
- Far below average
- Below average
- Average
- Above average
- Far above average

Goal: AVERAGE.

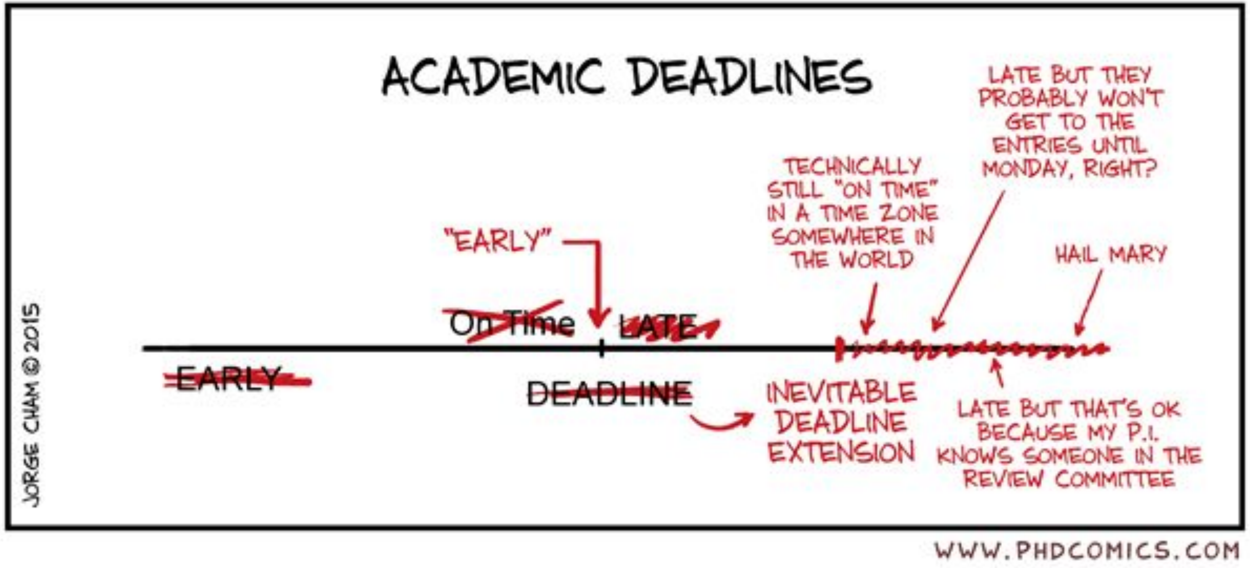
Communicate among team members if contributions are biased.

Support

- Pose questions on the [“issues”](#) tool from the GitHub class2022 repository.
- Recommended: “Watch” to these issues to be aware of all the activity.



Submission



- Deadline: **Wednesday March 16th at 3pm**.
- Deliverables:
  - Submit your report by editing these slides.
  - Provide feedback regarding the teamwork on this evaluation form.

Before you leave...

Set up a team repository in the course page on GitHub:

- Join the [MCV M6 GitHub page] with a personal account (if you haven't done it before)
- Get your team ID by writing down the team members (4) in [this spreadsheet].
- Instructors will create a github repo based on your provided usernames.
  - Create a team with the ID assigned on the spreadsheet.
  - Create a repository with your ID to store your code.
- (optional) Set a recent photo in your public Github profile.

https://github.com/orgs/mcv-m6-video/