Learning Objectives

- In this module…
  - Define the 2D object detection problem
  - Apply ConvNets to 2D object detection
  - Challenges of object detection
  - 2D object tracking problem

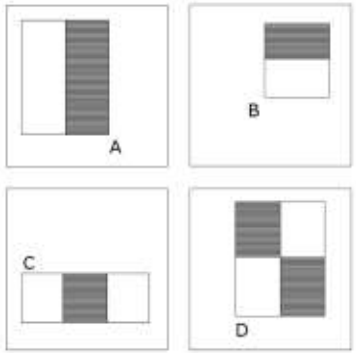
Learning Objectives

- Learn the 2D object detection task problem formulation.

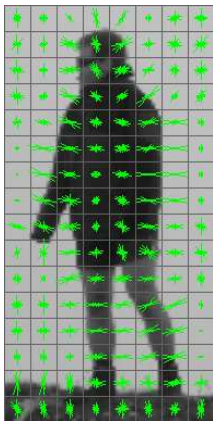
- Learn to determine how good a 2D object detector is through evaluating performance measures.

Brief History of Object Detection

- 2001 – Viola, Jones – Viola Jones Object Detection Framework



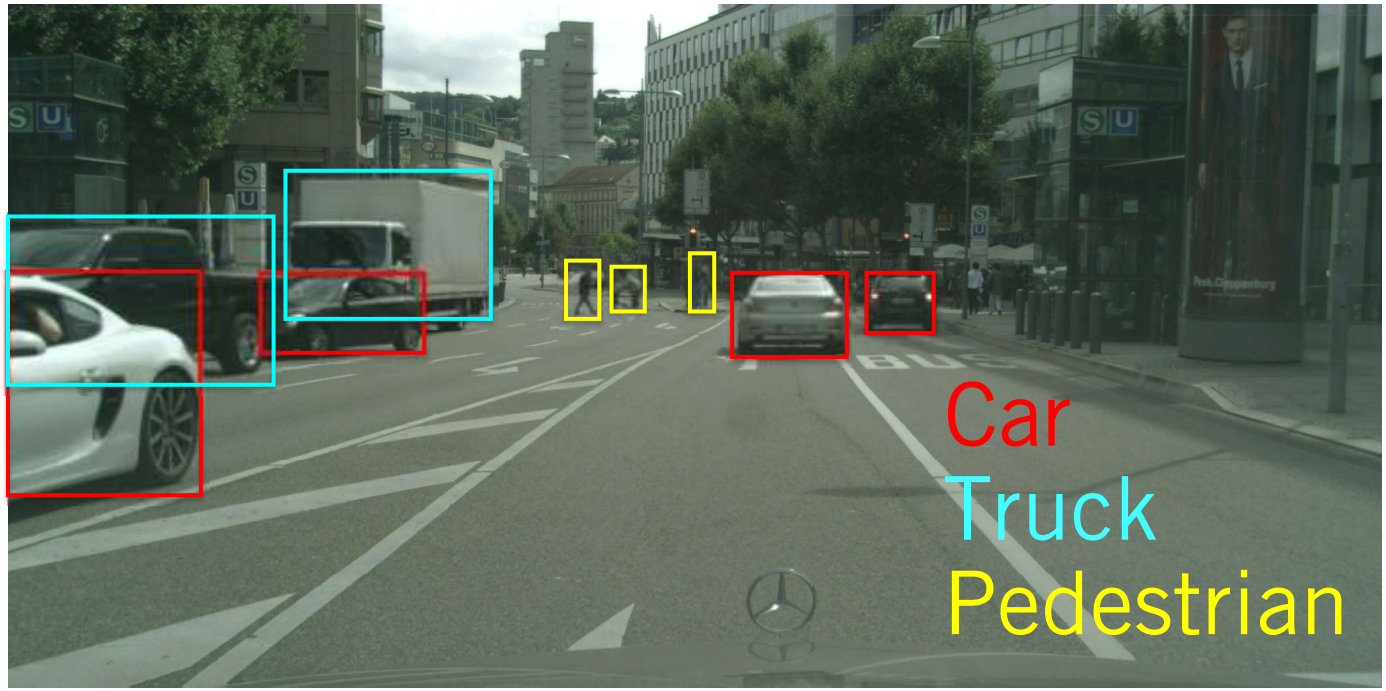
- 2005 – Dalal, Triggs – Histogram of Oriented Gradients



- 2012 – Krizhevsky, Sutskever, Hinton - Alexnet

Imagene

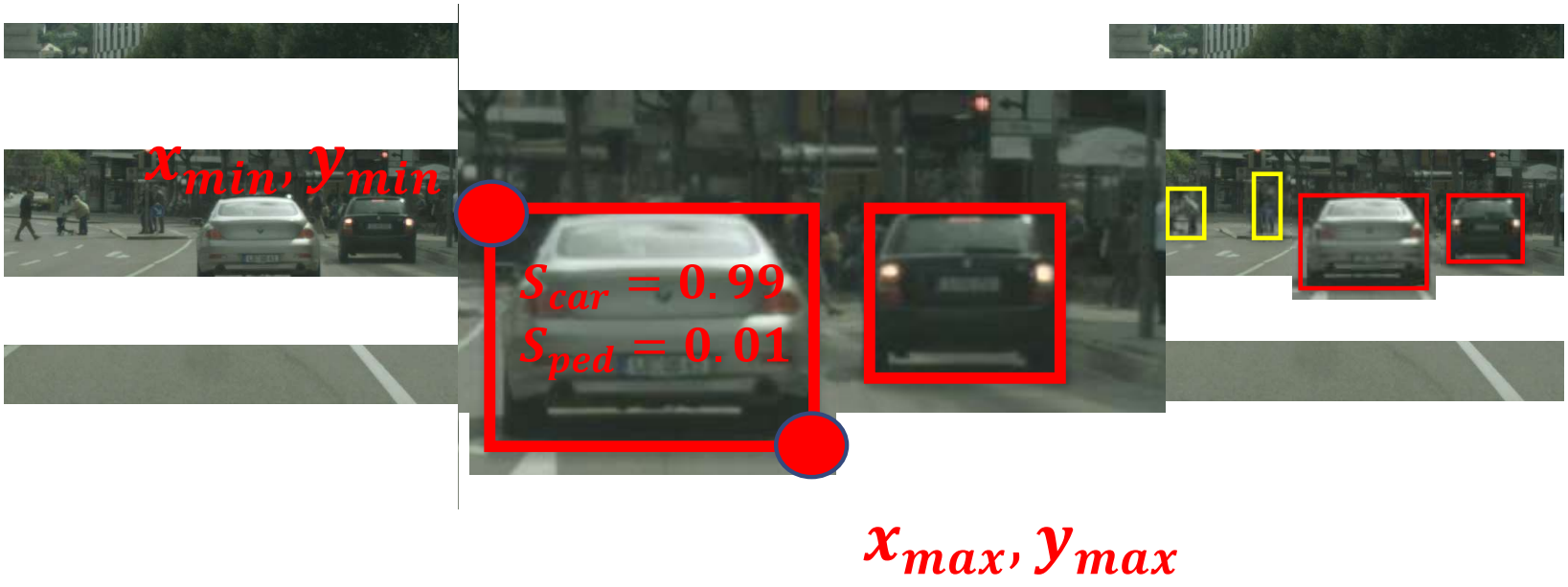
The Object Detection Problem



Object Detection Is Not Trivial !

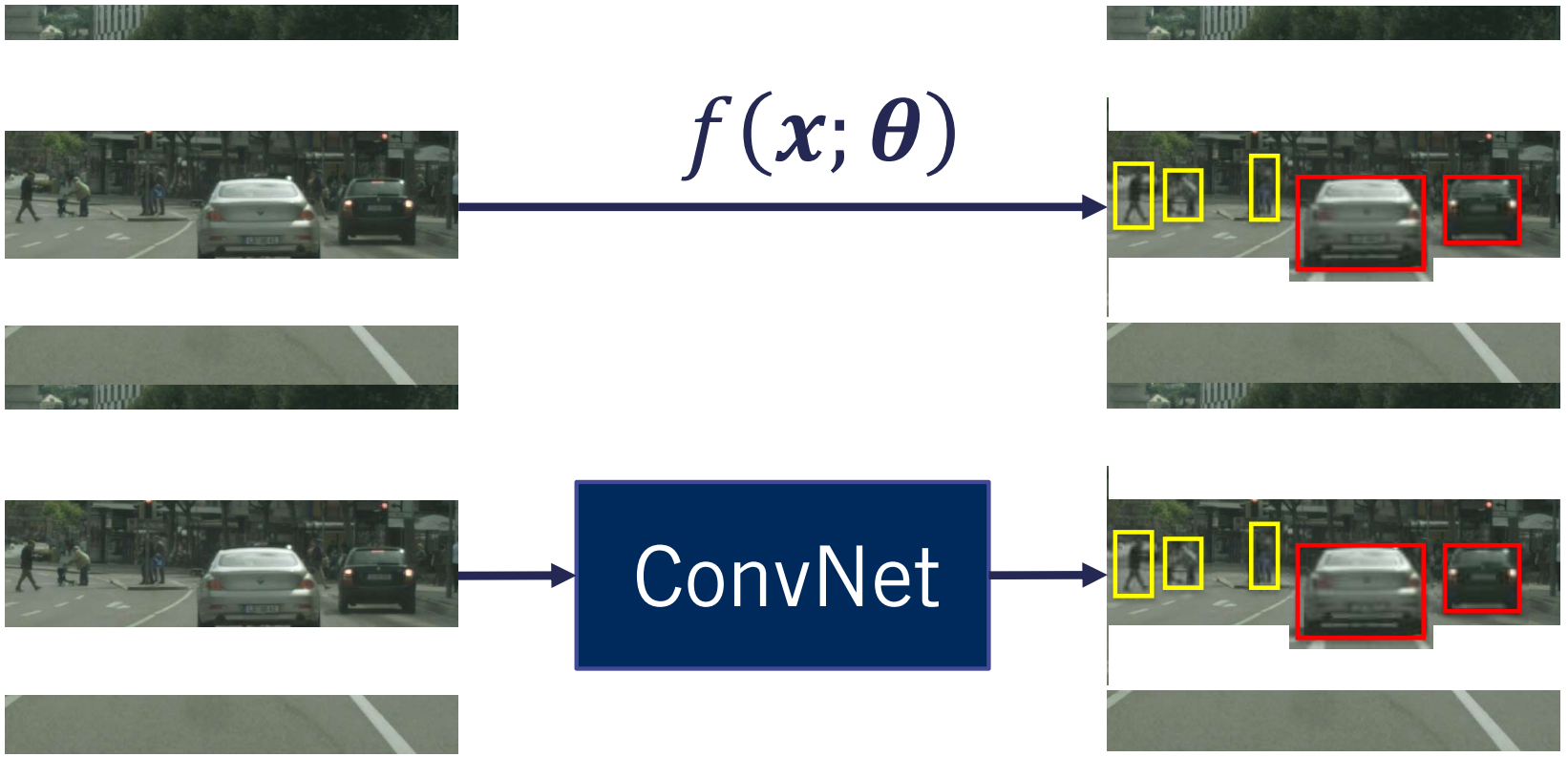
- **Extent of objects is not fully observed!**
  - **Occlusion:** Background objects covered by foreground objects
  - **Truncation:** Objects are out of image boundaries
- **Scale:** Object size gets smaller as the object moves farther away
- Illumination Changes:
  - **Too bright**
  - **Too dark**

Mathematical



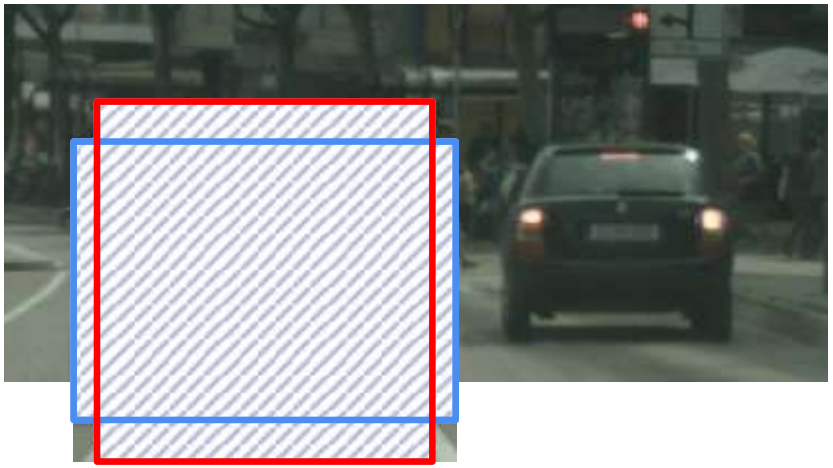
$$f(x;\theta) = [x_{min}, y_{min}, x_{max}, y_{max}, S_{class_1} \dots, S_{class_k}]$$

ConvNets For Object Detection



Evaluation Metrics

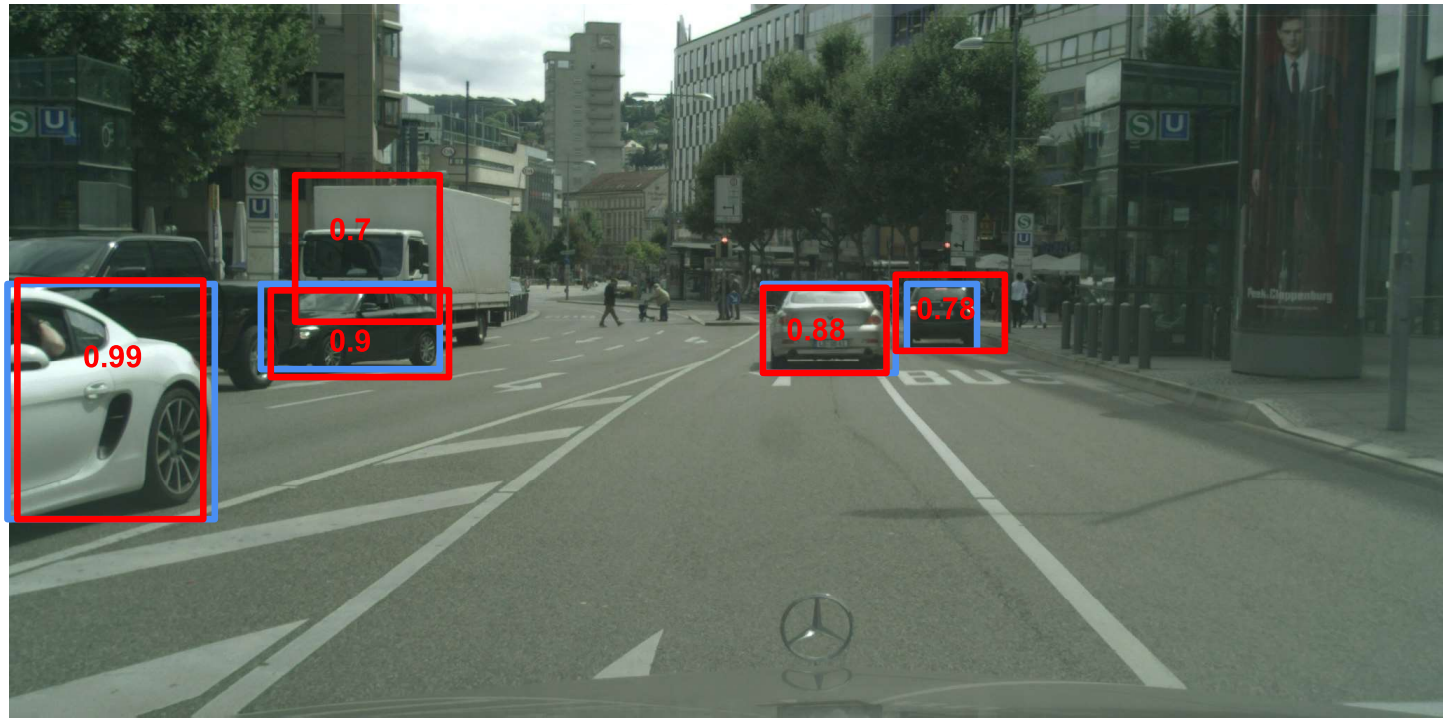
- **Intersection-Over-Union (IOU):** area of intersection of **predicted box** with a **ground truth box**, divided by the area of their union



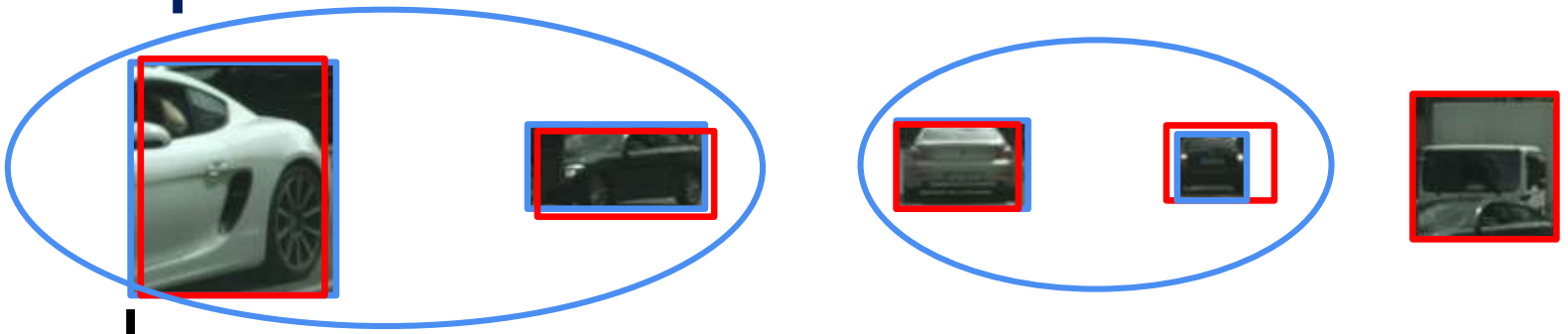
Evaluation Metrics

- **True Positive (TP):** Object class score > score threshold, and IOU > IOU threshold
- **False Positive (FP):** Object class score > score threshold, and IOU < IOU threshold
- **False Negative (FN):** Number of ground truth objects not detected by the algorithm
- **Precision:** TP/ (TP + FP)
- **Recall:** TP / (TP + FN)
- **Precision Recall Curve (PR-Curve):** Use multiple object class score thresholds to compute precision and recall. Plot the values with precision on y-axis, and recall on x-axis
- **Average Precision (AP):** Area under PR-Curve for a single class. Usually approximated using **11** recall points

Example



Example

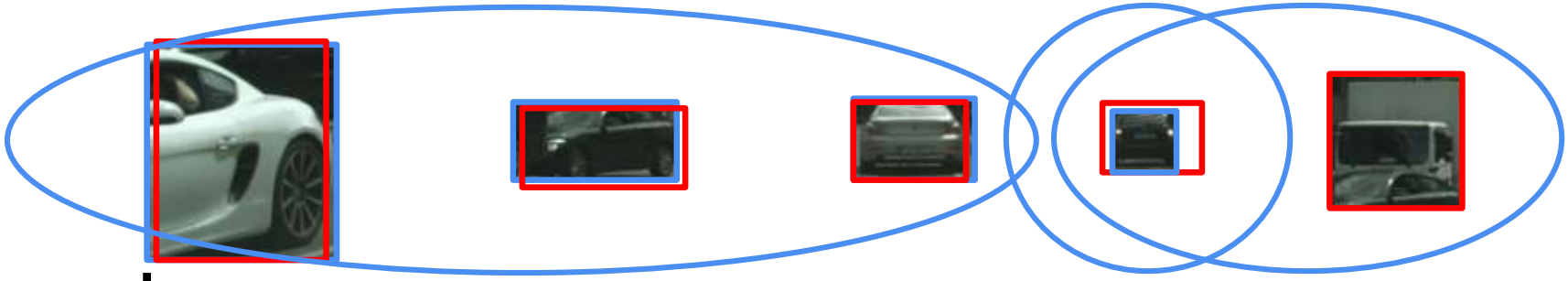


S_{car}	0.99	0.9	0.88	0.78	0.7
IOU	0.95	0.9	0.8	0.3	0

- Score Threshold: 0.9
- IOU Threshold: 0.7

- TP = 2
- FP = 0
- FN = 2
- Precision = 2/2 = 1
- Recall = 2/4 = 0.5

Example



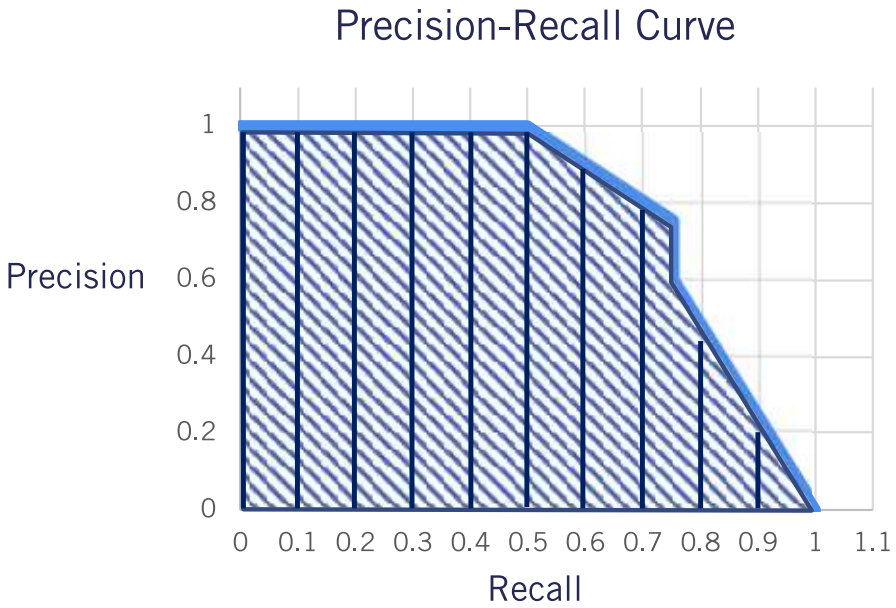
S_{car}	0.99	0.9	0.88	0.78	0.7
IOU	0.95	0.9	0.8	0.3	0

- Score Threshold: 0.7
- IOU Threshold: 0.7

- TP = 3
- FP = 2
- FN = 1
- Precision = 3/5 = 0.6
- Recall = 3/4 = 0.75

Example

Score threshold	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
Precision	1	0.75	0.6	0.6	0.6	0.6	0.6	0.6	0.6
Recall	0.5	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75



$$AP = \frac{1}{11}\sum_{r=0}^{11} p_r \approx 0.75$$

Summary

- 2D object detection comprises of localizing an object and determining what the object is
- 2D object detectors are evaluated using the Average Precision metric, at a specific IOU threshold

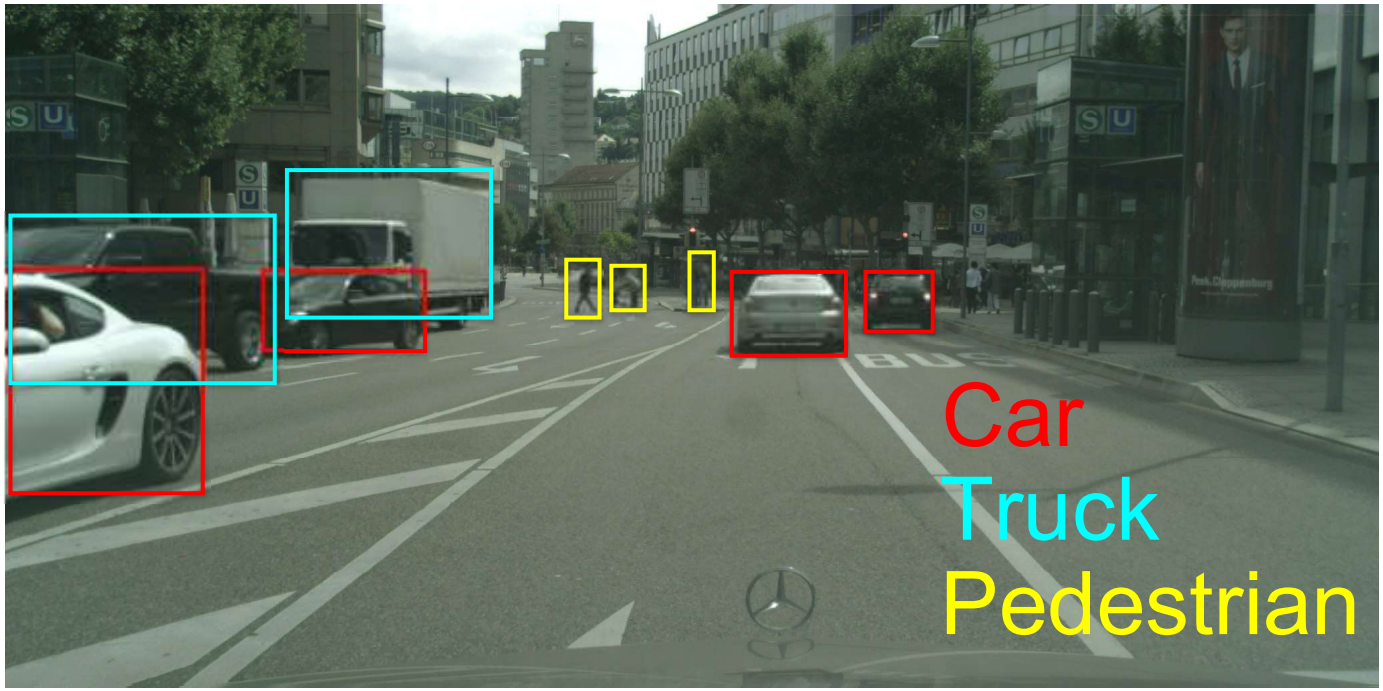
- **Next: 2D object detection using ConvNets**

Learning Objectives

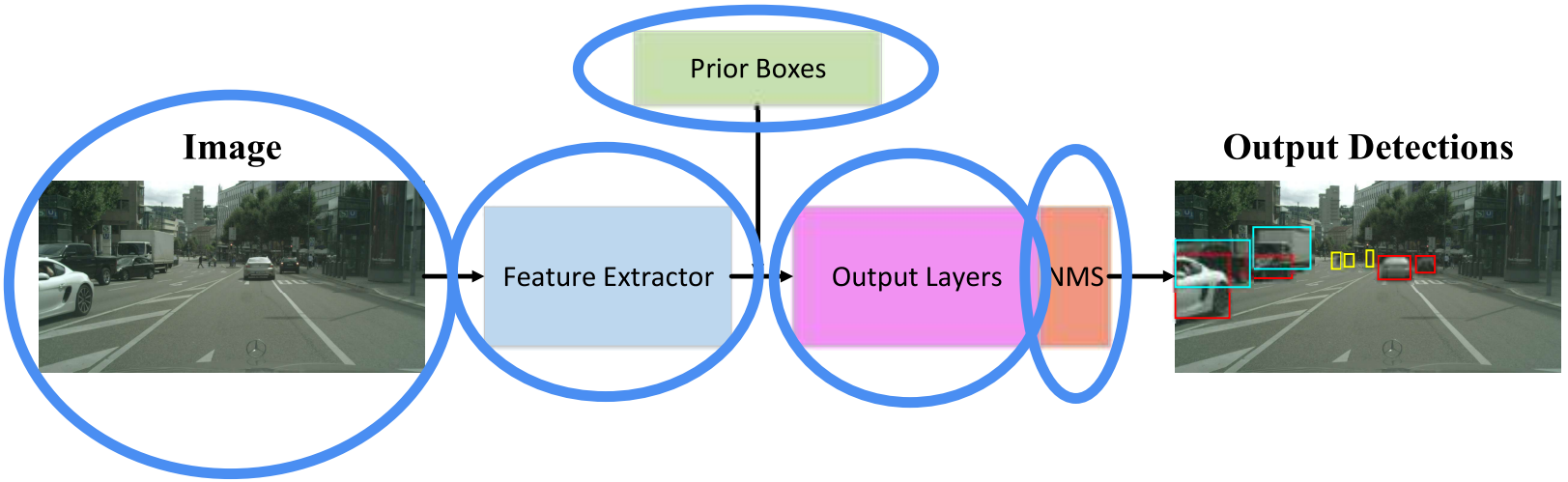
- Learn to build standard single stage architecture for 2D object detection

- Learn common neural network design choices for performing 2D object detection using the proposed architecture

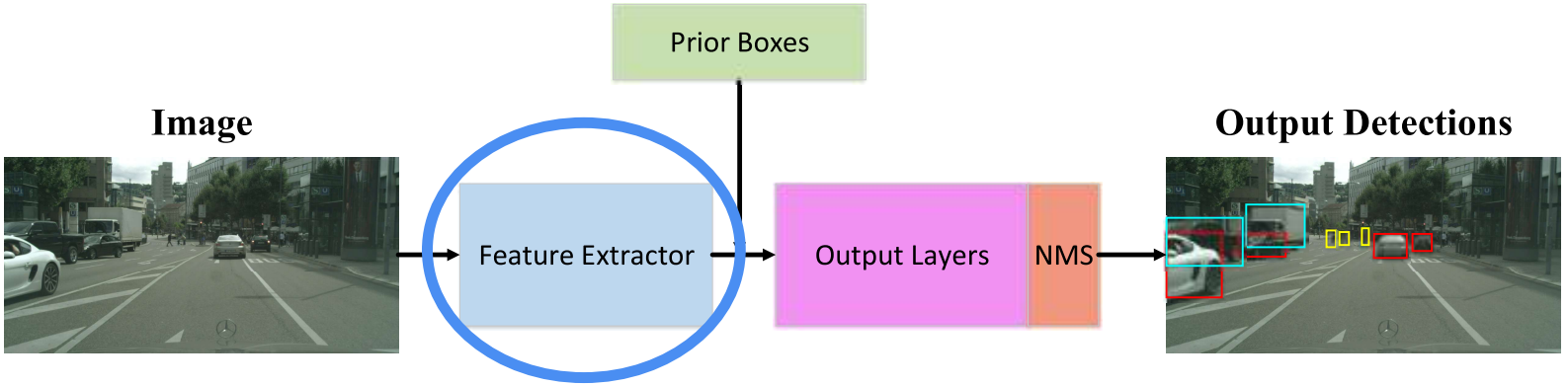
The Object Detection Problem



ConvNets For 2D Object Detection



The Feature Extractor

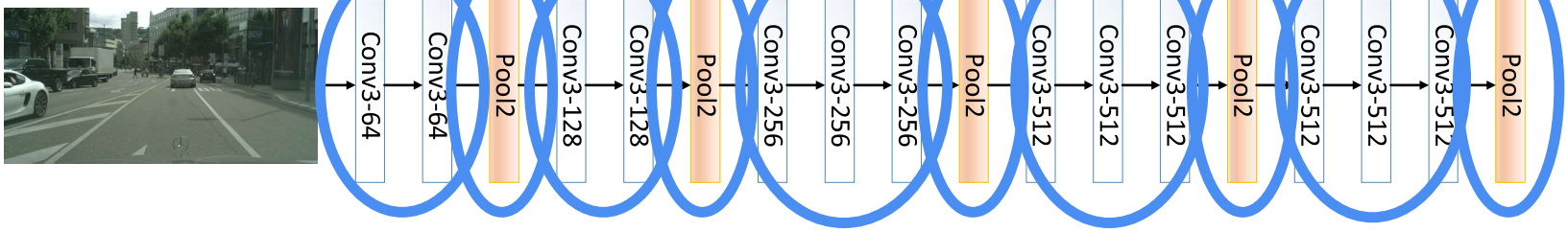


The Feature Extractor

- Feature extractors are the most computationally expensive component of the 2D object detector
- The output of feature extractors usually has much **lower width and height** than those of the input image, but much **greater depth**
- Very active area of research, with new extractors proposed on regular basis
- **Most common extractors are**: VGG, ResNet, and Inception

VGG Feature Extractor

- Alternating convolutional and pooling layers
- All convolutional layers are of size 3x3xK, with stride 1 and 1 zero-padding
- All pooling layers use the **max** function, and are of size 2x2, with stride 2 and no padding



VGG Feature Extractor

- All convolutional layers are of size 3x3, with stride 1 and 1 zero-padding
  - $W_{out} = \frac{W_{in}-m+2\times P}{S}+1$
  - $W_{out} = \frac{W_{in}-m+2\times P}{S}+1 = \frac{H_{in}-3+2\times 1}{1}+1 = W_{in}$
  - $H_{out} = \frac{H_{in}-m+2\times P}{S}+1 = \frac{H_{in}-3+2\times 1}{1}+1 = H_{in}$
  - $D_{out} = K$
- All pooling layers use the max function, and are of size 2x2, with stride 2 and no padding.
  - $W_{out} = \frac{W_{in}-m}{S}+1 = \frac{W_{in}-2}{2}+1 = \frac{W_{in}}{2}$
  - $H_{out} = \frac{H_{in}-m}{S}+1 = \frac{H_{in}-2}{2}+1 = \frac{H_{in}}{2}$
  - $D_{out} = D_{in}$

The Feature Extractor

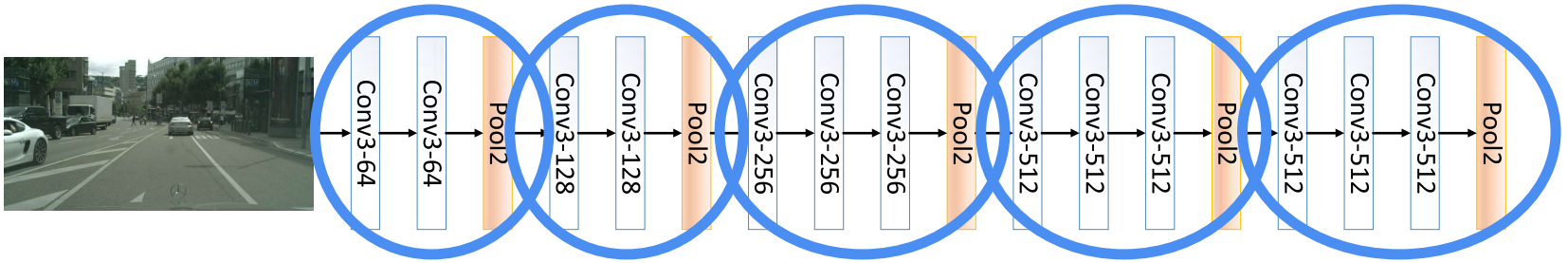
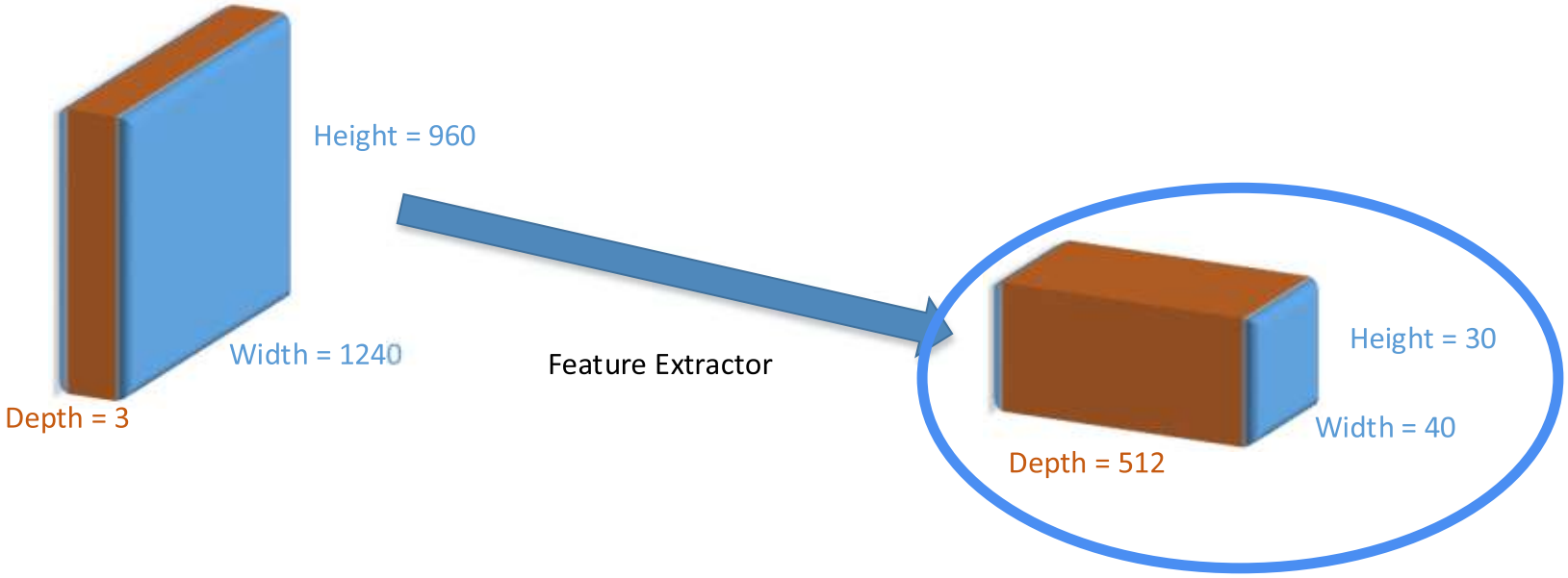
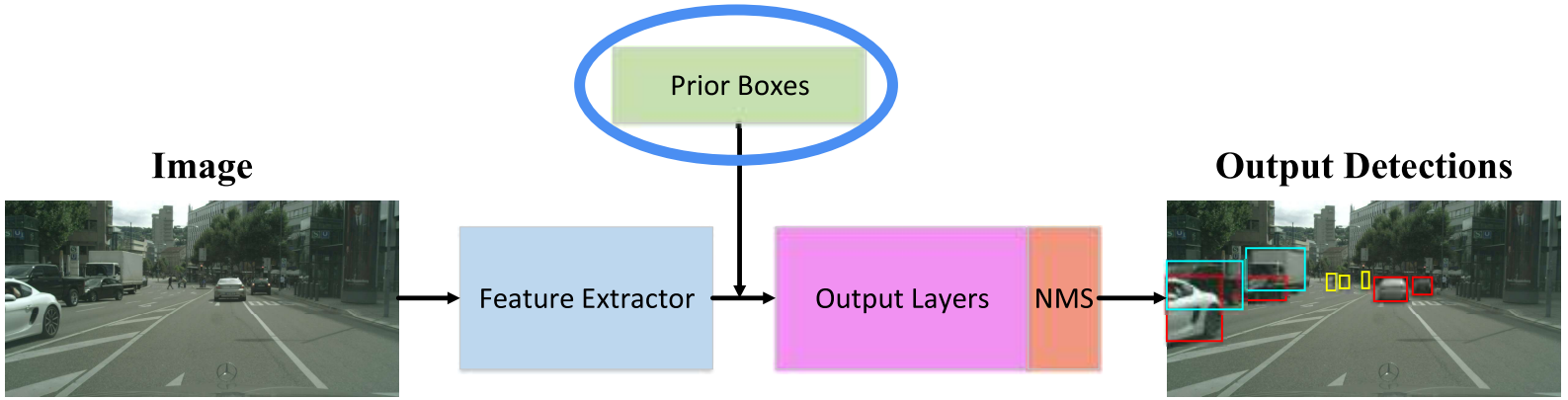


	Image	Conv1	Conv2	Conv3	Conv4	Conv5
Width	M	M/2	M/4	M/8	M/16	M/32
Height	N	N/2	N/4	N/8	N/16	N/32
Depth	3	64	128	256	512	512

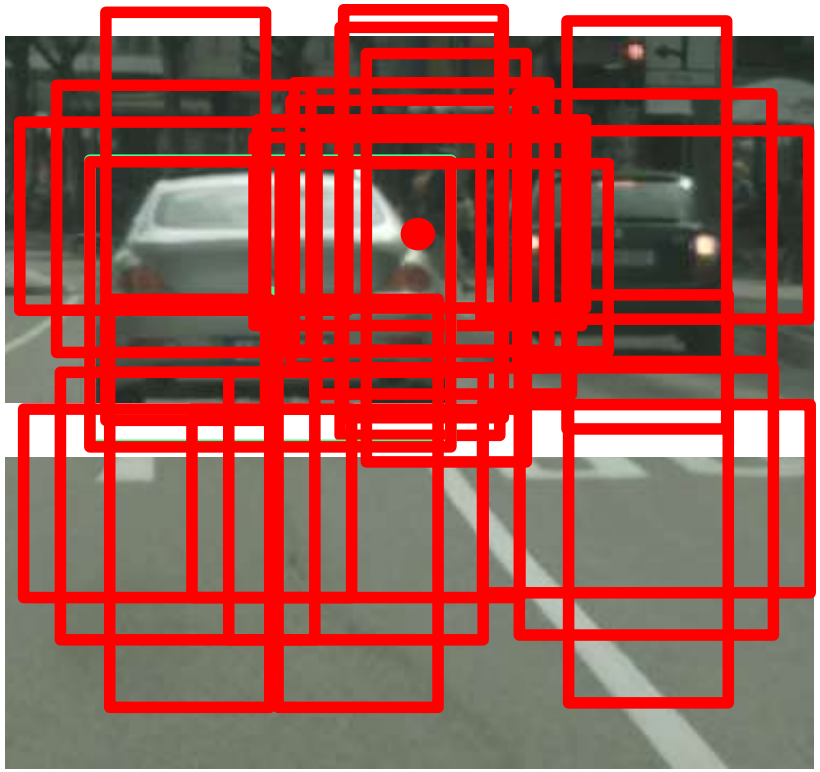
Output Volume Shape



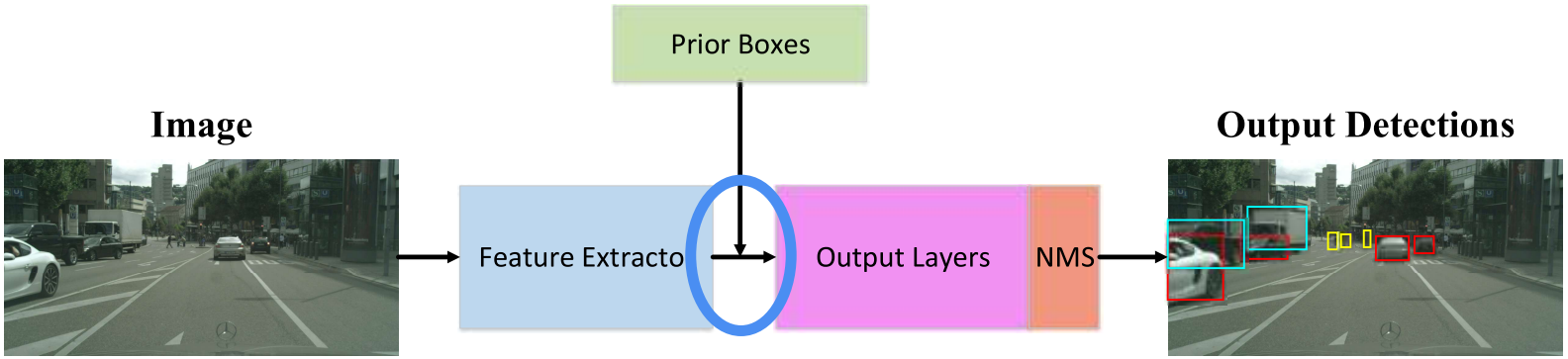
Prior/Anchor Bounding Boxes



Prior/Anchor Bounding Boxes

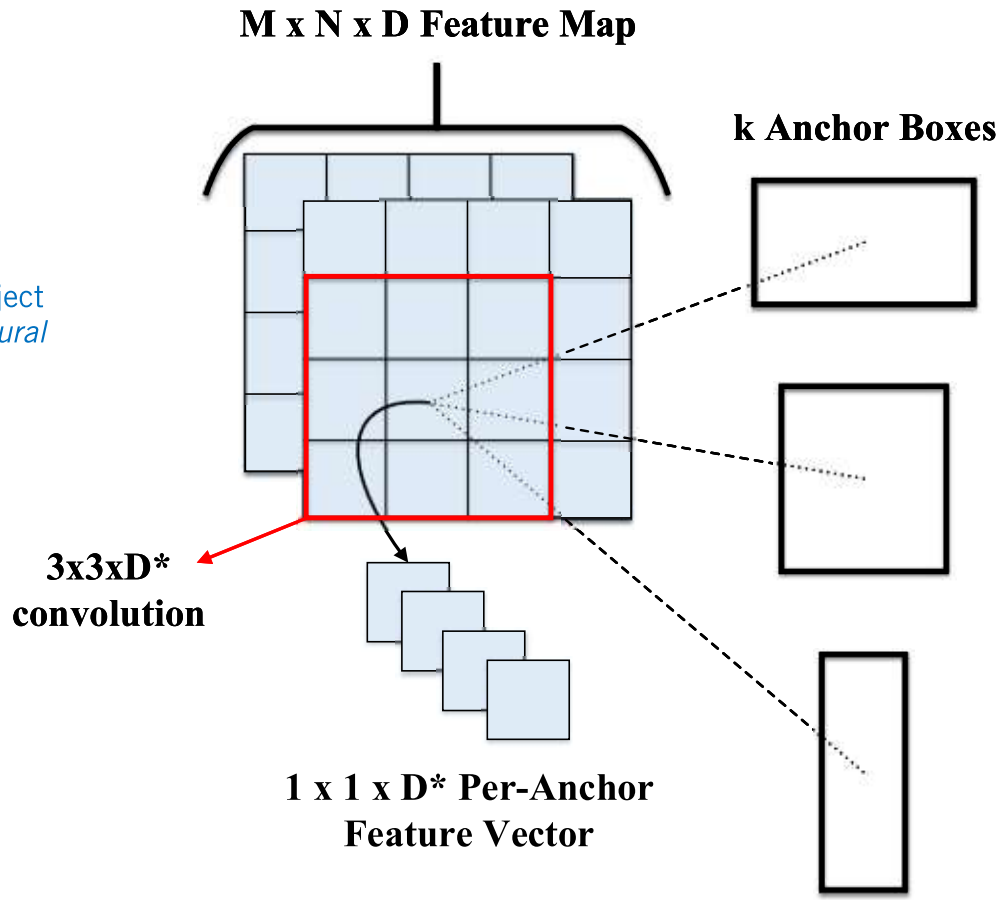


Prior/Anchor Bounding Boxes

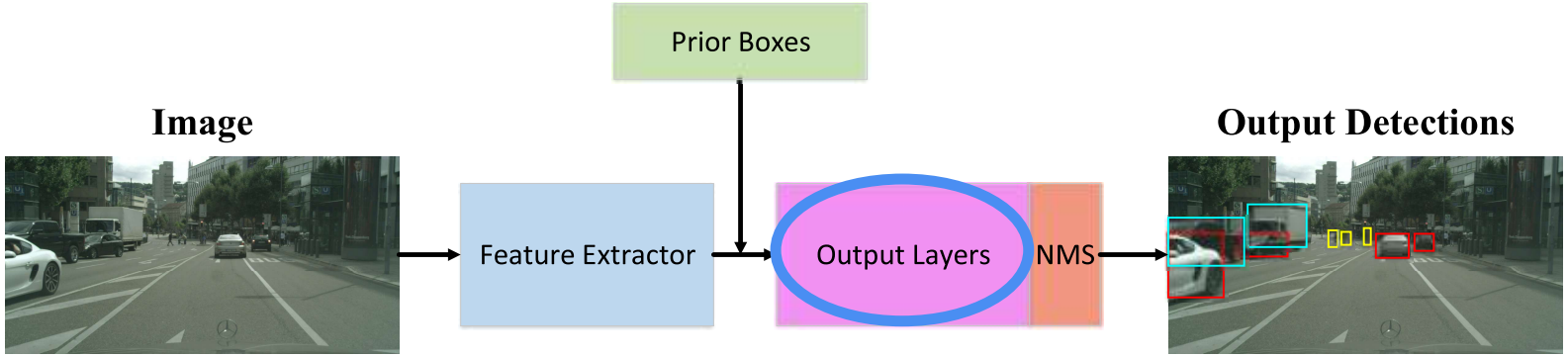


Using Anchor Boxes

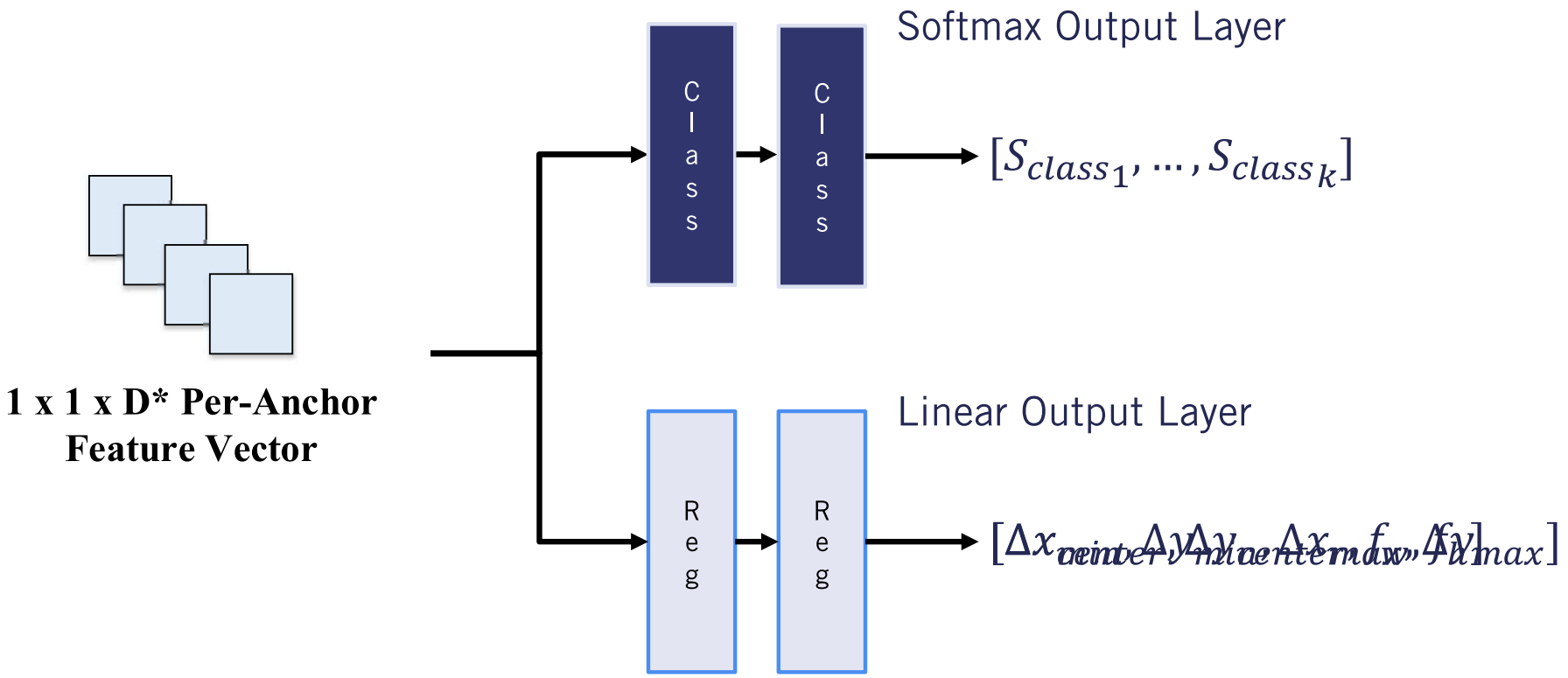
Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems*. 2015



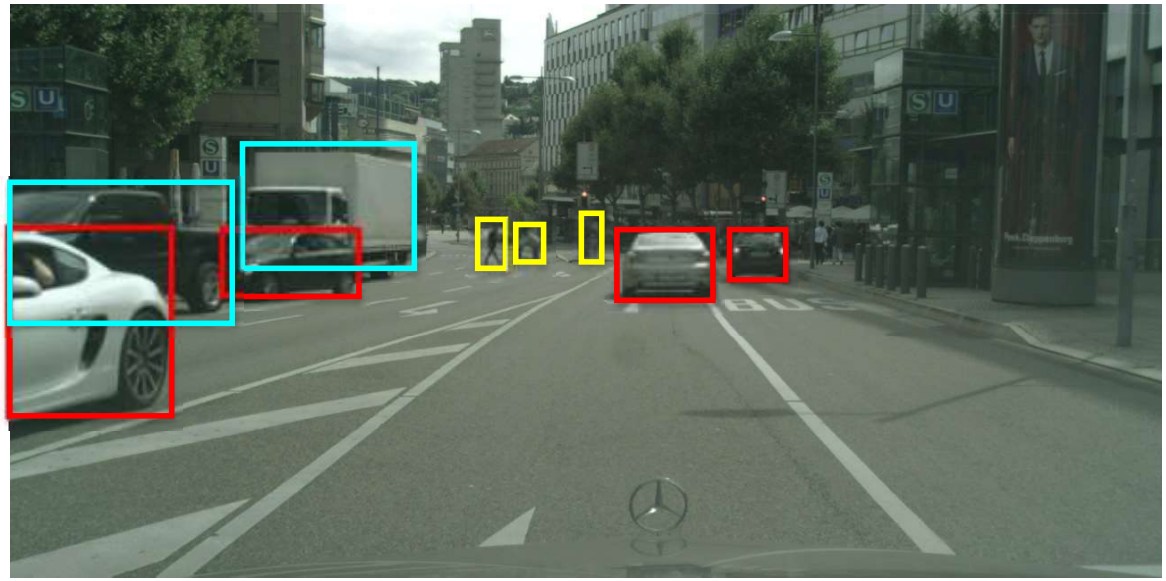
Output Layers



Classification VS Regression Heads



Output handling

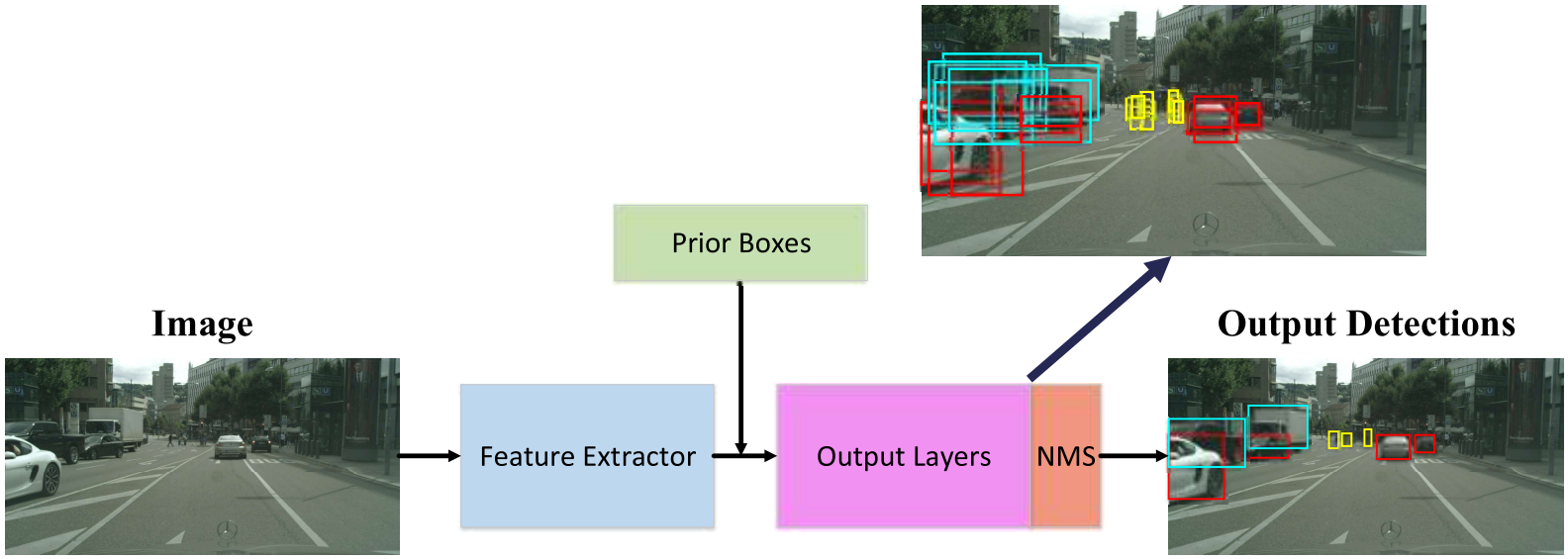


Summary

- 2D object detectors can be performed using convolutional neural networks
- Usually, anchor boxes are used as priors for the neural network to shift around to achieve object classification and localization

- **Next: Training vs Inference**

ConvNets For 2D Object Detection

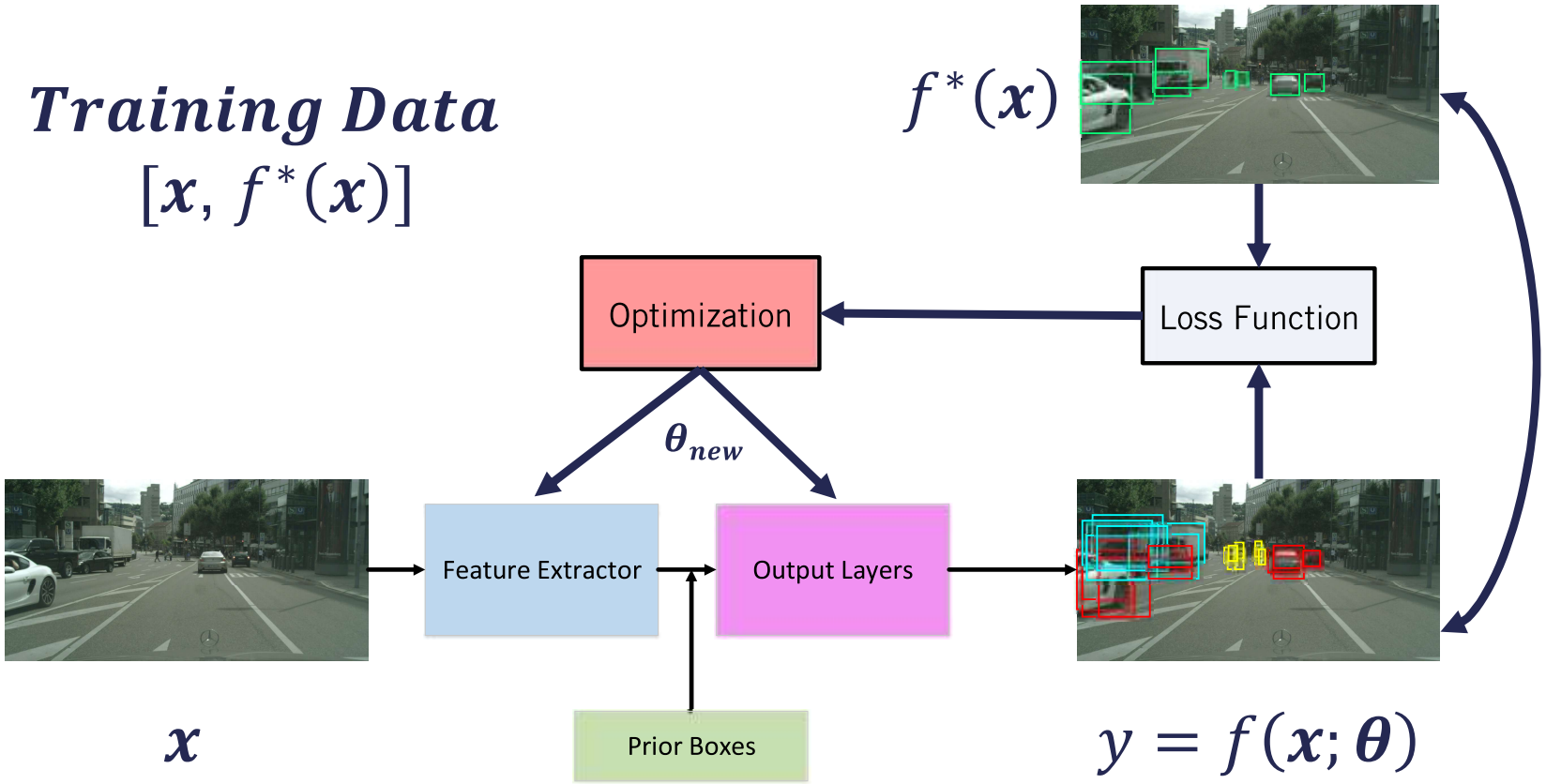


Learning objectives

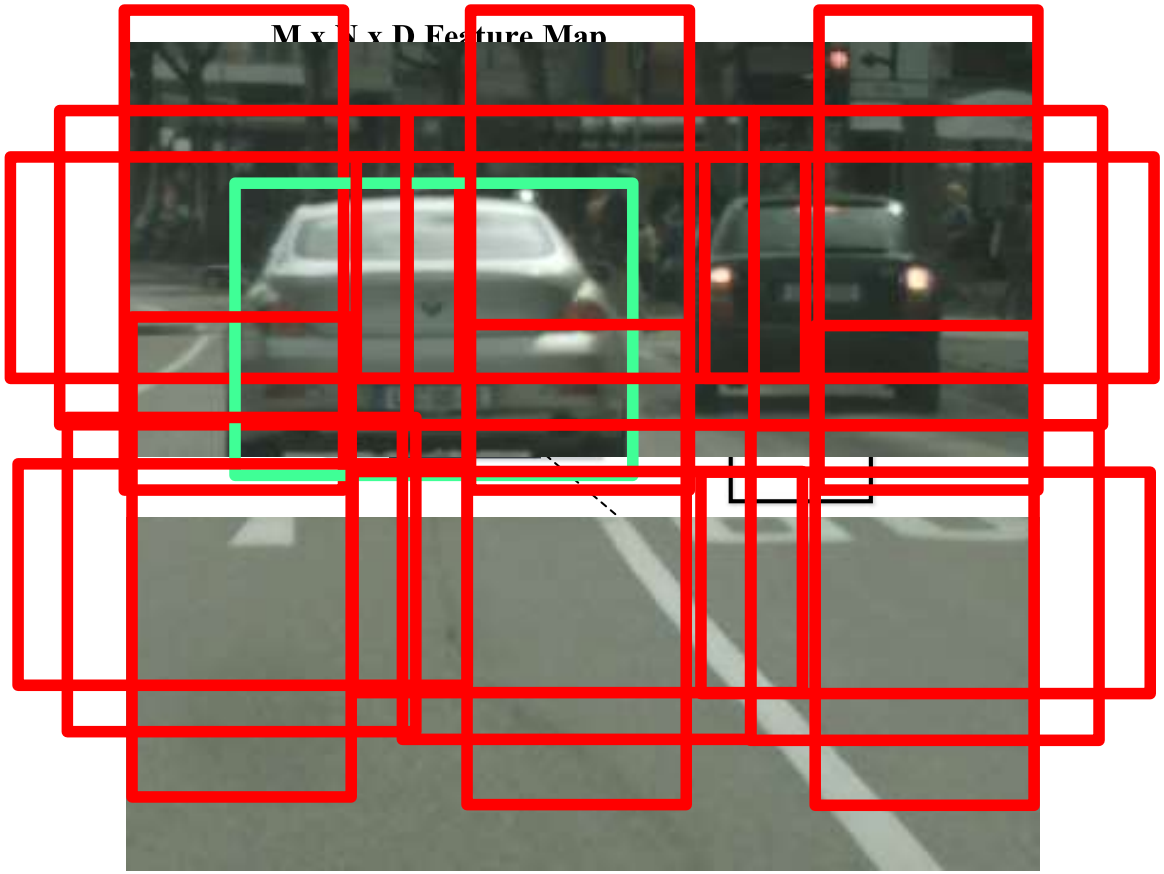
- Learn how to handle multiple detections per object during training through **minibatch selection**

- Learn how to handle multiple detections per object during inference, through **non-maximum suppression** (NMS)

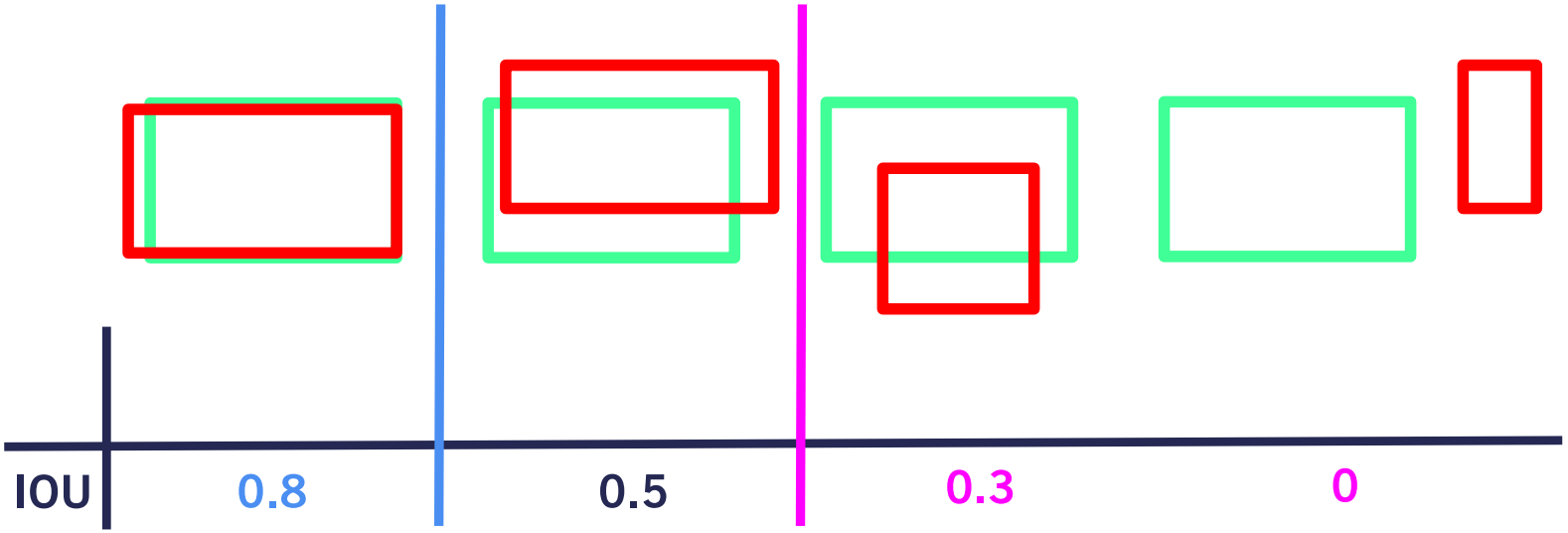
2D Object Detector Training



MiniBatch Selection



MiniBatch Selection



- Negative Member Threshold: < 0.4
- Positive Member Threshold: > 0.6

Minibatch Selection

- Negative anchors target:
  - **Classification:** Background
  - **Regression:** None
- Positive anchors target:
  - **Classification:** Category of the ground truth bounding box
  - **Regression:** Align box parameters with highest IOU ground truth bounding box

Minibatch Selection

- **Problem:** Majority of anchors are negatives results in neural network will label all detections as background
- **Solution:** Sample a chosen **minibatch size**, with 3:1 ratio of negative to positive anchors to eliminate bias towards the negative class
- Choose negatives with **highest classification loss** (online hard negative mining) to be included in the minibatch

- Example: minibatch size is 64→ 48 hardest negatives and 16 positives

Classification Loss

$$L_{cls} = \frac{1}{N_{total}} \sum_{i} CrossEntropy(s_i^*, s_i)$$

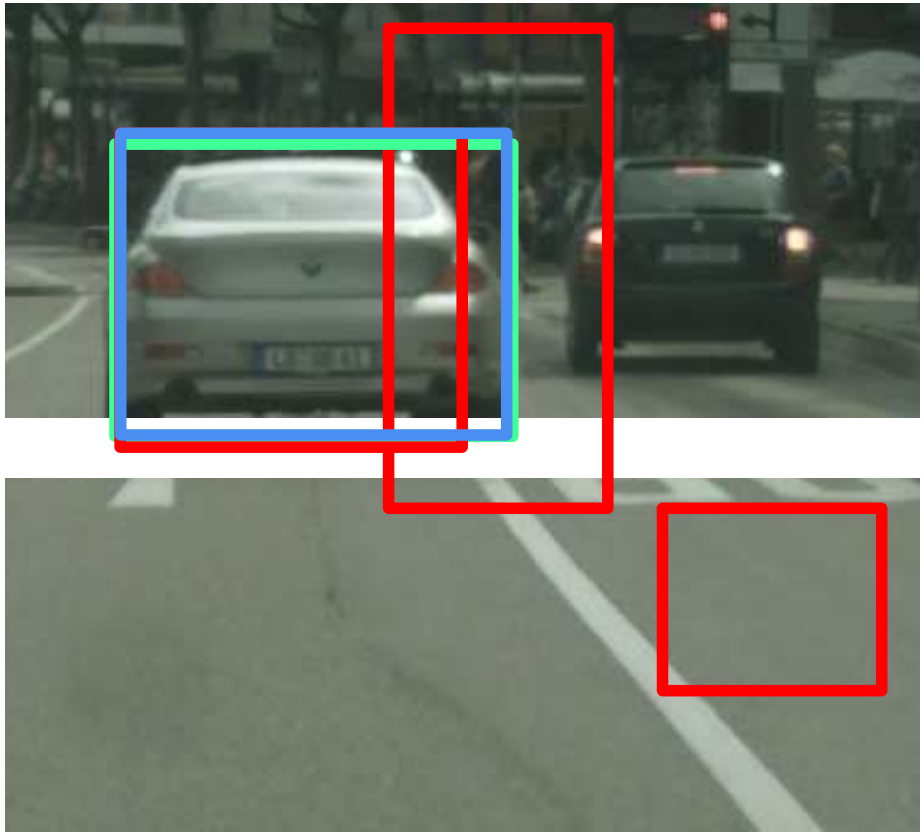
- N_{total} is the size of our minibatch
- s_i is the output of the neural network
- s_i^* is the anchor classification target:
  - **Background** if anchor is negative
  - **Ground truth box class** if anchor is positive

Regression Loss

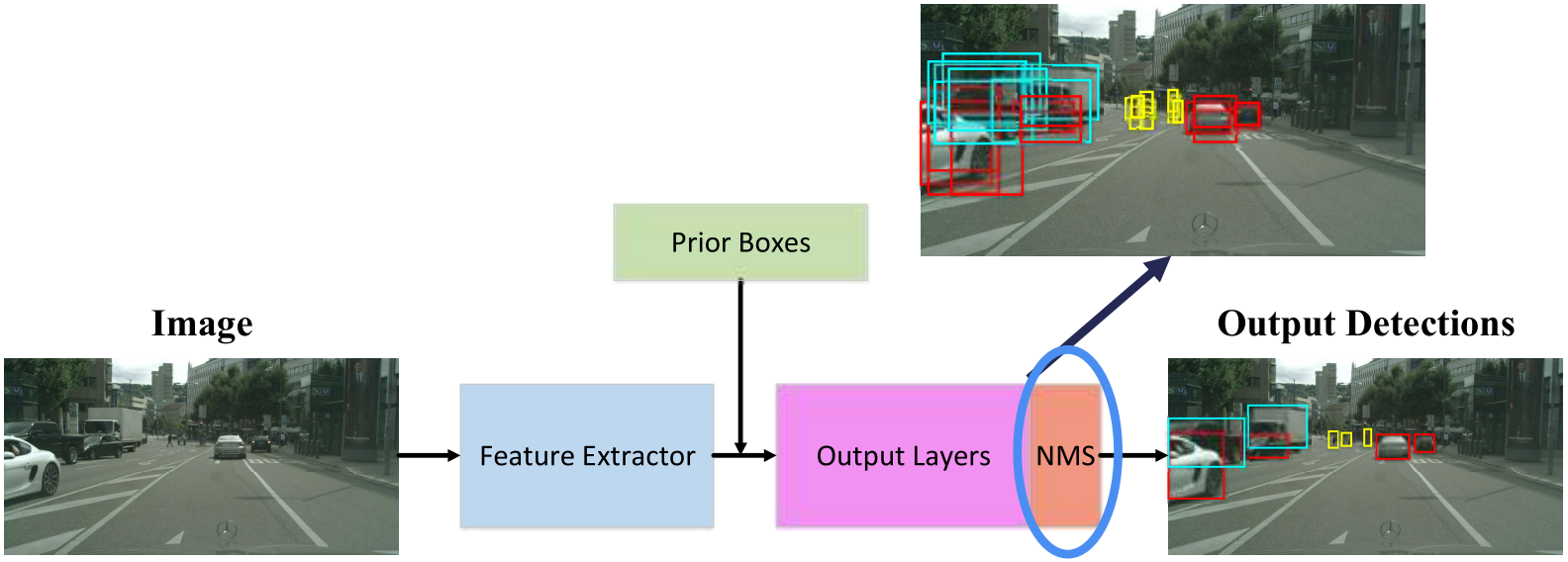
$$L_{reg} = \frac{1}{N_p} \sum_i p_i L_2(b_i^*, b_i)$$

- p_i is 0 if anchor is negative and 1 if anchor is positive
- N_p is the number of positive anchors in the minibatch
- b_i^* is the ground truth bounding box
- b_i is the estimated bounding box, applying the regressed residuals to the anchor box parameters

Visual Representation Of Training



Inference Time

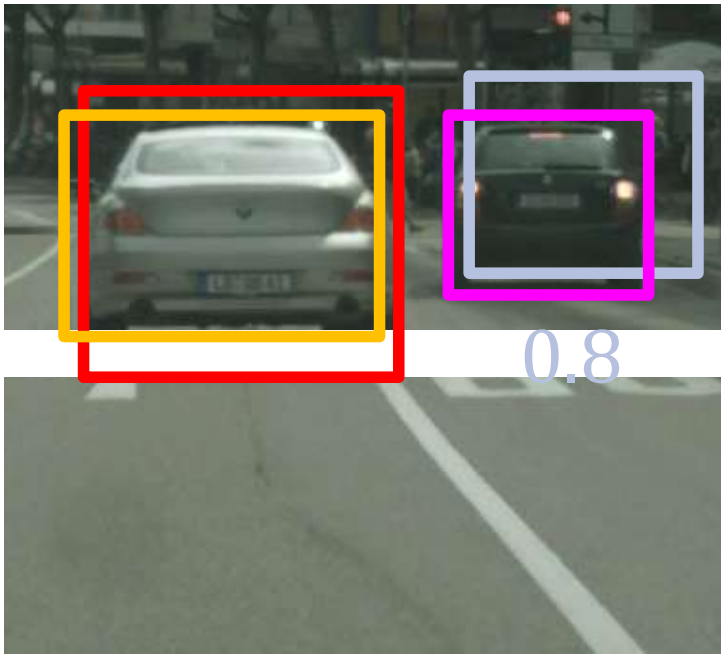


Non-Maximum Suppression

Input:
$B = \{B_1 ... B_n\} | B_i = (x_i, y_i, w_i, h_i, s_i) \forall i \in [1, n]$
IOU threshold η
begin
    $\bar{B} = Sort(B, s, \downarrow)$
    $D = \emptyset$
    for $b \in \bar{B}$ *and* $\bar{B}$ *not* $\emptyset$ **do**
        $b_{max} = b$
        $\bar{B} \leftarrow \bar{B} \backslash b_{max}$
        $D \leftarrow D \cup b_{max}$
        for $b_i \in \bar{B} \backslash b_{max}$ **do**
            if $IOU(b_{max}, b_i) \geq \eta$ **then**
                $\bar{B} \leftarrow \bar{B} \backslash b_i$
            end
        end
    end
    Output: D
end

Non-Maximum Suppression

$\geq \quad \eta = 0.7$



$$\boldsymbol{B} = \{B_1, B_2, B_3, B_4\}$$
$$\overline{\boldsymbol{B}} = \{B_1, B_2, B_3, B_4\}$$
$$\boldsymbol{S} = \{0.98, 0.94, 0.6, 0.45\}$$
$$\boldsymbol{D} = \{\,\}$$

$$b_{max} = \{B_1\}$$

Non-Maximum Suppression

$\geq \eta = 0.7$

$\boldsymbol{B} = \{B_1, B_2, B_3, B_4\}$

$\overline{\boldsymbol{B}} = \{B_2, B_4\}$

$\boldsymbol{S} = \{0.94, 0.45\}$

$\boldsymbol{D} = \{B_1\}$

$b_{max} = \{B_2\}$

Non-Maximum Suppression

$\eta = 0.7$

$\boldsymbol{B} = \{B_1, B_2, B_3, B_4\}$

$\overline{\boldsymbol{B}} = \{\}$

$\boldsymbol{S} = \{\}$

$\boldsymbol{D} = \{B_1, B_2\}$

$b_{max} = \{B_2\}$

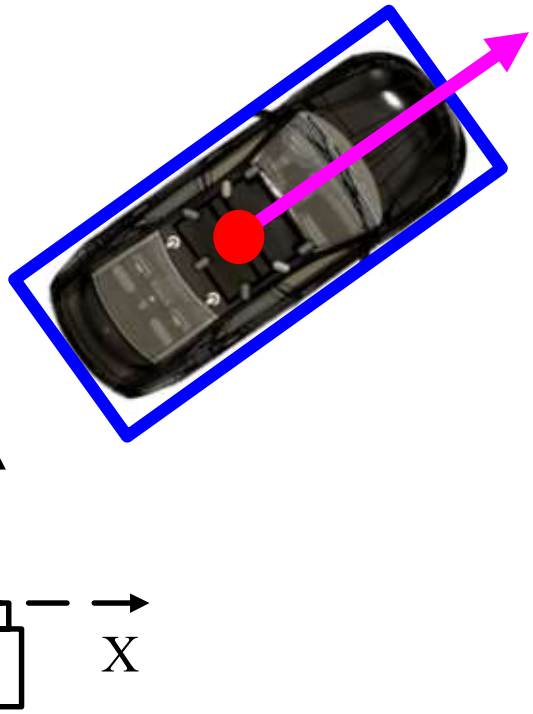
Summary

- To train a neural network for 2D object detection, use minibatch selection on anchors
- For inference, use Non-Maximum Suppression to get a single output bounding box per object
- **Next: Using 2D object detectors for autonomous driving**

Learning Objectives

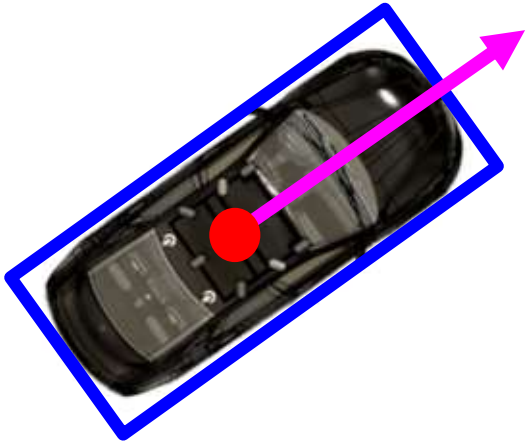
- Learn how to extend 2D object detection output to 3D
- Learn how to use 2D object detection output to track objects in images and in 3D
- Learn how to use 2D object detection to detect traffic signs and signals

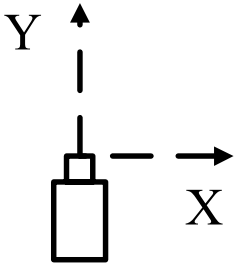
3D Object Detection



- Estimating the:
  - Category Classification
  - Position of the centroid in 3D
  - Extent in 3D
  - Orientation in 3D

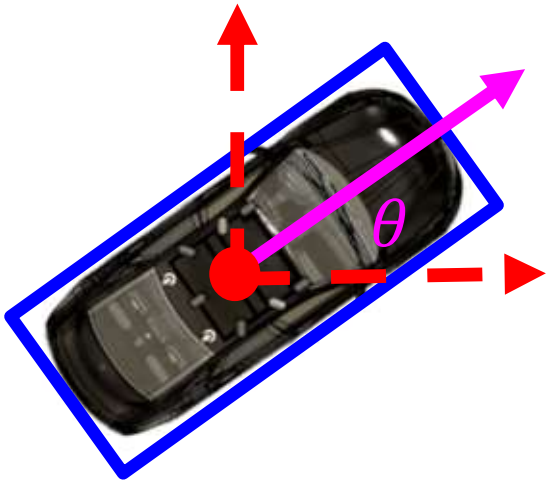
3D Object Detection

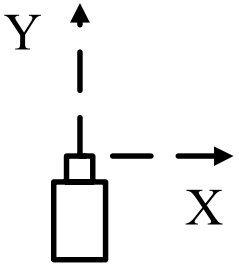




- Estimate the:
  - Car, pedestrian, cyclist
  - $[x, y, z]$
  - $[l, w, h]$
  - $[\phi, \psi, \theta]$

3D Object Detection

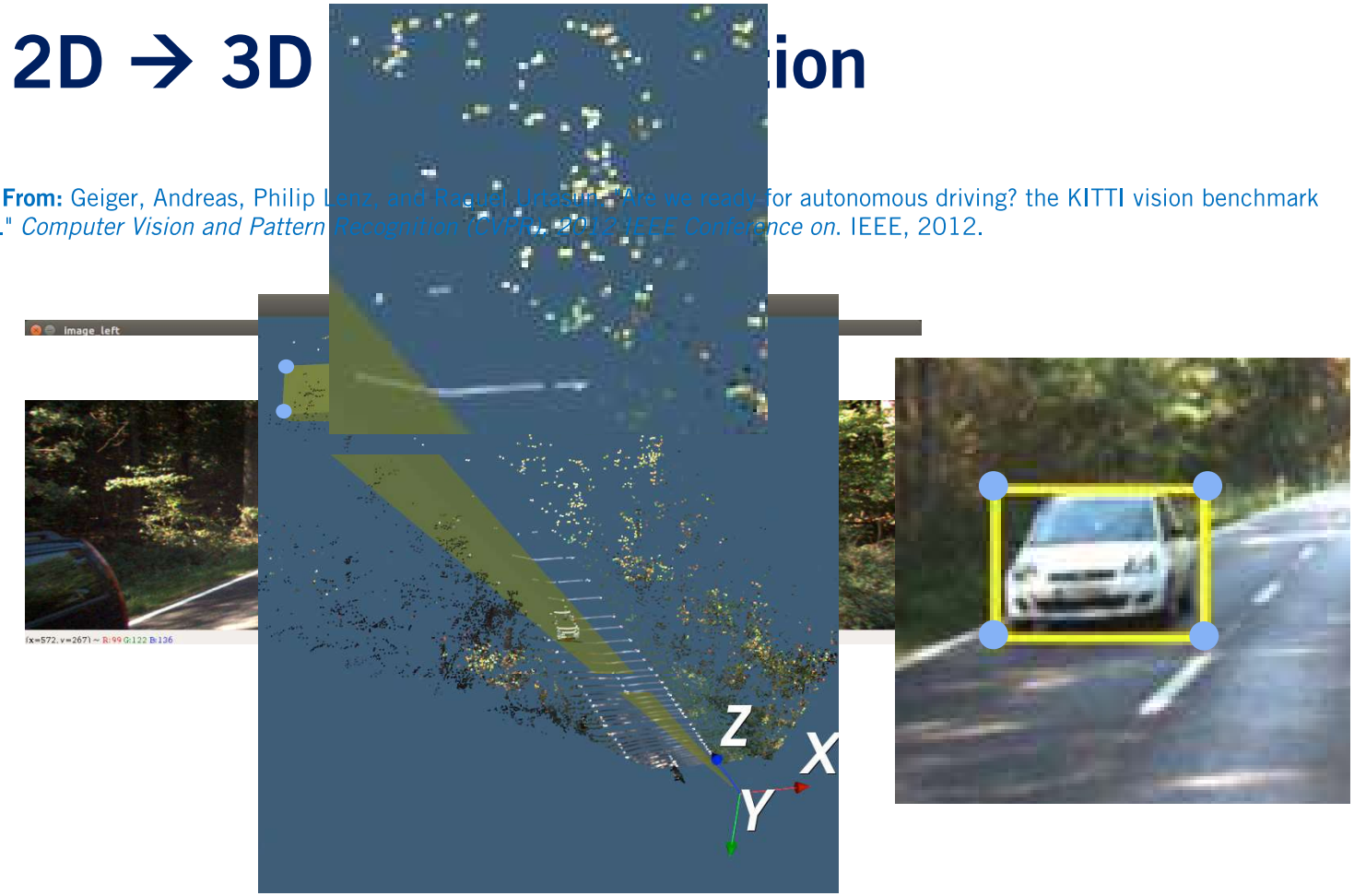




- Estimate the:
  - Car, pedestrian, cyclist
  - $[x, y, z]$
  - $[l, w, h]$
  - $[\phi, \psi, \theta]$

From 2D → 3D ion

Data From: Geiger, Andreas, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the KITTI vision benchmark suite." *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012.



Fro bject Detection

Histogram of
X Y Z

Neural
Network

Typical 3D object detection results

Code at: https://github.com/kujason/avod

From 2D → 3D Object Detection

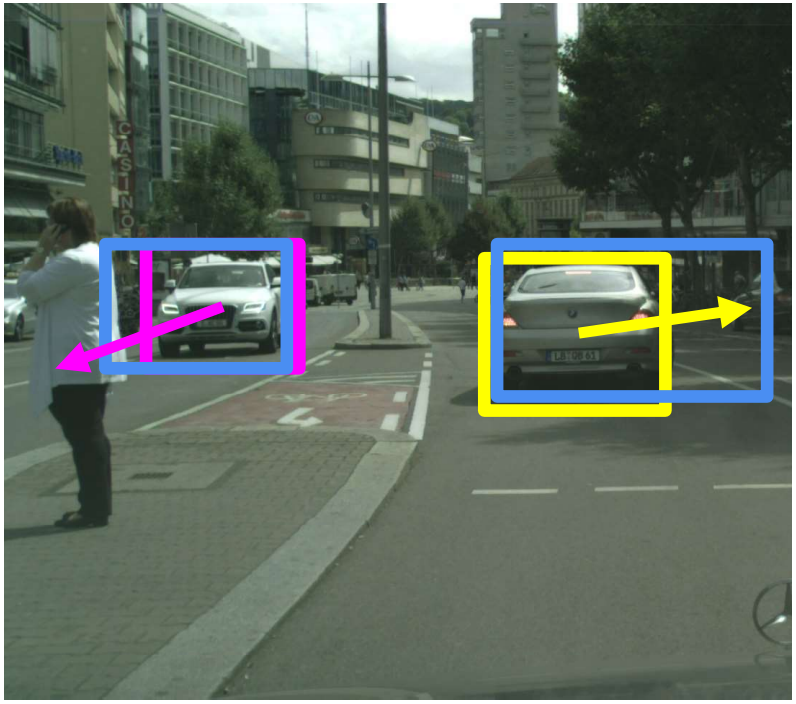
- Advantages:
  - Allows exploitation of mature 2D object detectors, with high precision and recall
  - Class already determined from 2D detection
  - Does not require prior scene knowledge, such as ground plane location

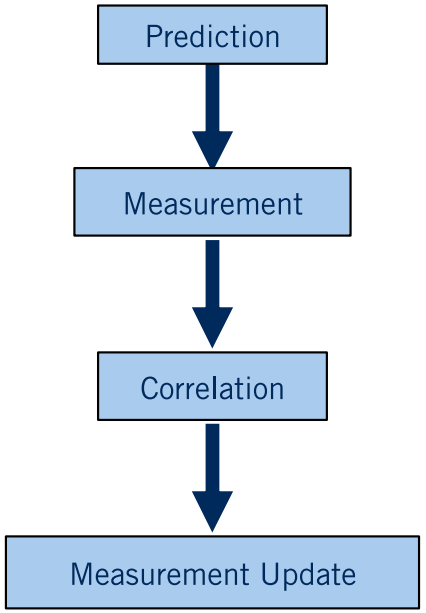
- Disadvantages:
  - The performance of the 3D estimator is bounded by the performance of the 2D detector
  - Occlusion and truncation are hard to handle from 2D only
  - 3D estimator needs to wait for 2D detector, inducing **latency** in our system

2D Object Tracking

- **Detection**: We detect the object independently in each frame and can record its position over time
- **Tracking**: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of object's motion pattern
- **Tracking Assumptions:**
  - Camera is not moving instantly to new viewpoint
  - Objects do not disappear and reappear in different places in the scene
  - If the camera is moving, there is a gradual change in pose between camera and scene

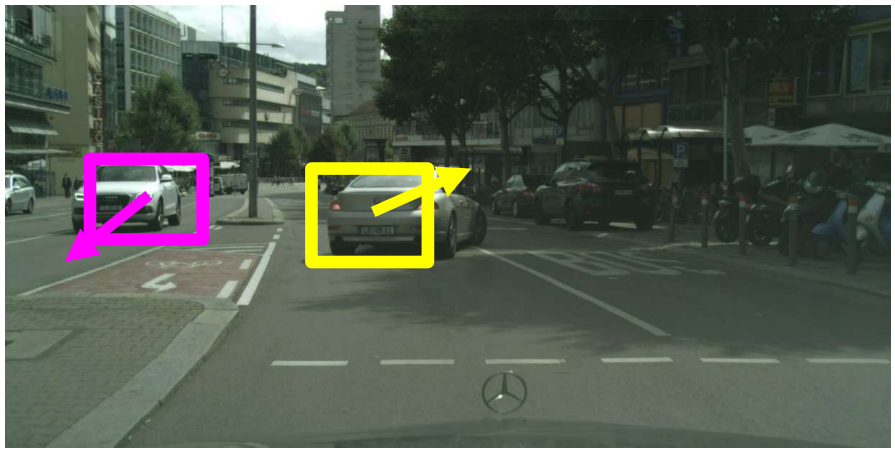
2D Object Tracking





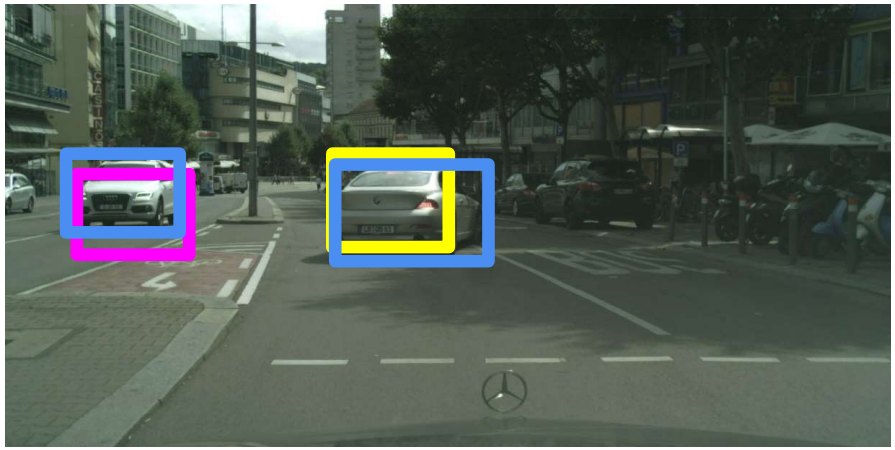
Object Tracking: Prediction

- Each object will have a predefined **motion model** in image space
- **Example:** $p_k = p_{k-1} + v_k \Delta t + \mathcal{N}(0, \Sigma)$



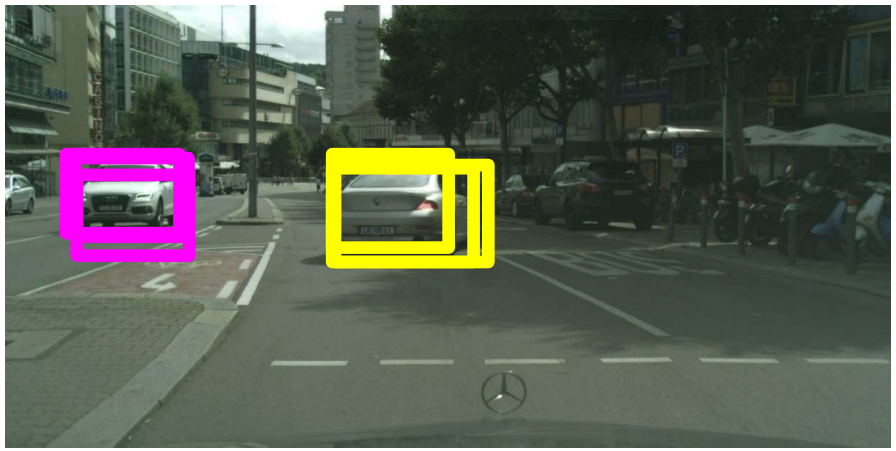
Object Tracking: Correlation

- Get **Measurement Bounding Boxes** from 2D detector.
- Correlate prediction with the **highest** IOU measurement



Object Tracking: Update

- The prediction and measurement are fused as part of the **Kalman Filter Framework**



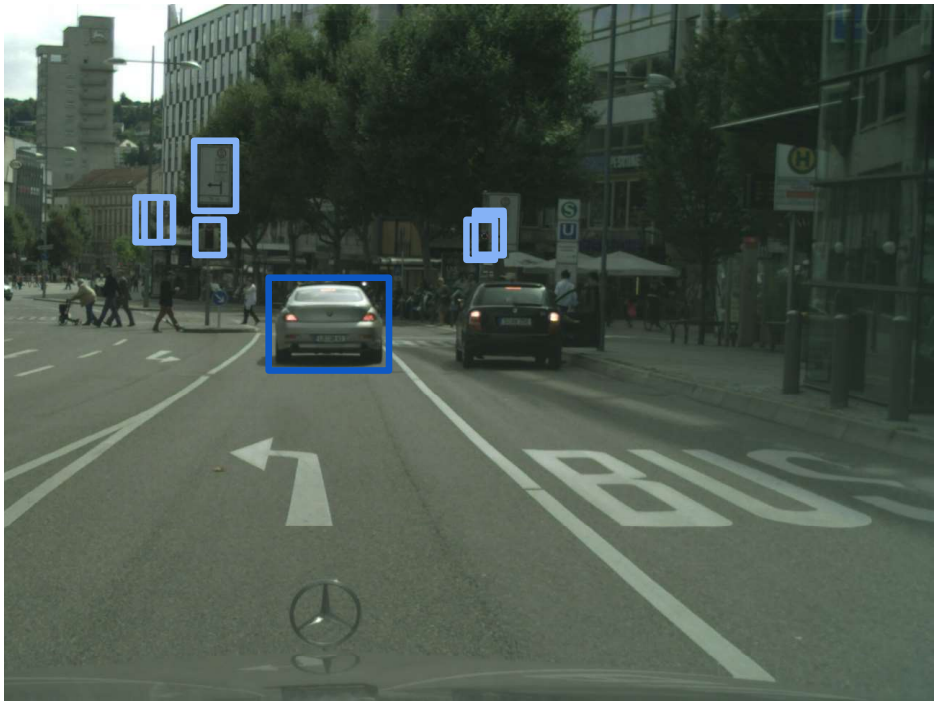
Object Tracking

- For each frame, we start new track if a measurement has no correlated prediction
- We also terminate inconsistent tracks, if a predicted object does not correlate with a measurement for a **preset** number of frames
- The same methodology can be used to track objects in 3D!

3D Object Tracking

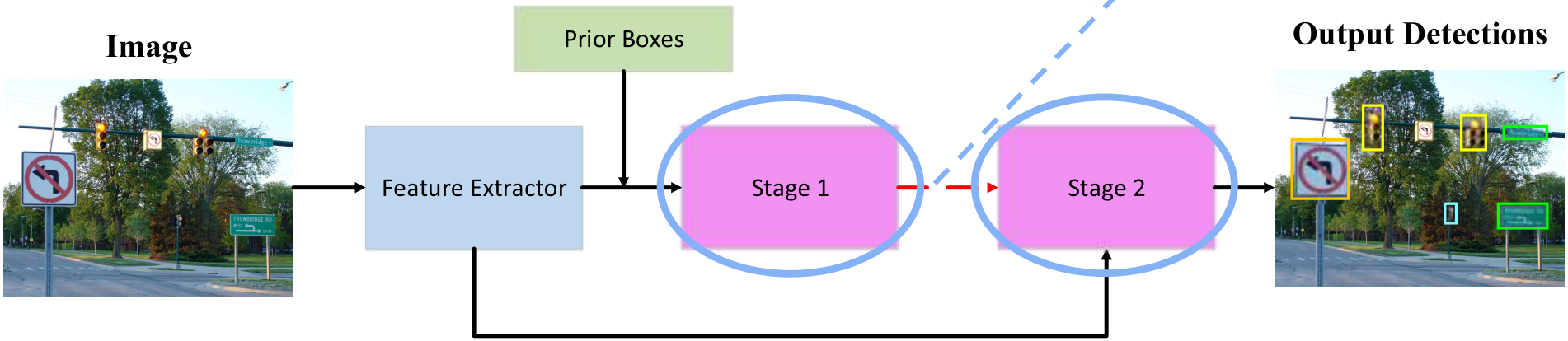
Traffic Sign and Traffic Signal Detection

- Traffic signs and signals appear smaller in size compared to cars, two-wheelers, and pedestrians.
- **Traffic signs** are highly variable with many classes to be trained on.
- **Traffic signals** have different states that are required to be detected.
- In addition, traffic signals change state as the car drives!



Traffic sign and signal detection

- 2D object detectors can be used to perform traffic sign and traffic signal detection without any modifications
- However, **multi-stage hierarchical models** have been shown to outperform the standard single stage object detectors



Summary

- The output of 2D object detectors can be extended to produce 3D object location and dimensions
- The output of 2D object detectors in consecutive frames can be used to track objects in 2D and in 3D
- The output of 2D object detectors can be used to detect traffic signs and signals, and determine the state of traffic signals