Lecture 3 | Sensors & Attacks

Secure Autonomous and Cyber-Physical Systems

CS 599 001/ECE 599 004

Winter 2022

Prof. Sibin Mohan

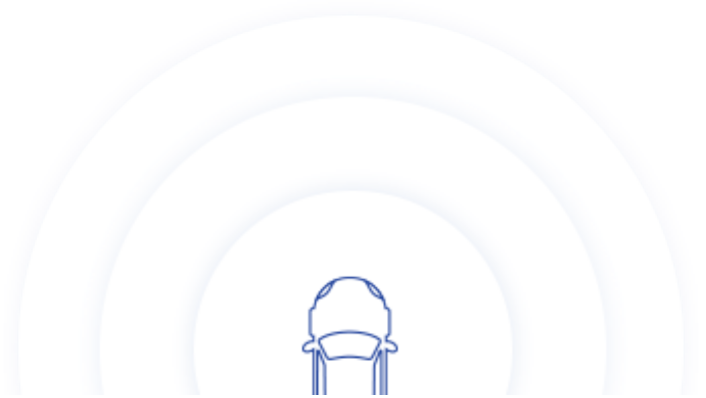
https://bit.ly/secureauto2022

Common Sensor Types

- **LiDAR/Millimeter Rada**
- **GPS**
- **Cameras:** lane, road, traffic light, stop line, surroundings
- **Stereo Vision**
- **Sonic**

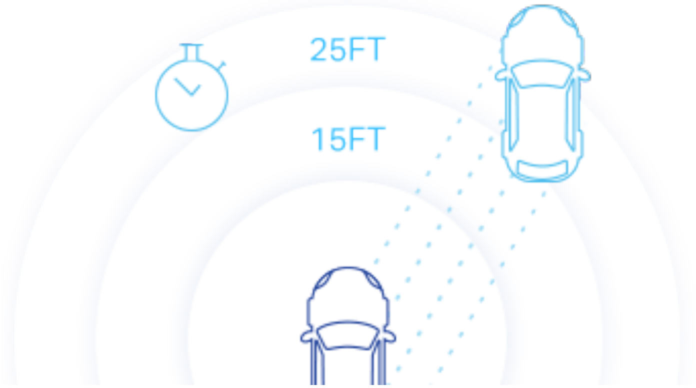
LiDAR

- **Li**ght **D**etection **A**nd **R**anging
- Laser scanning/3D scanning
- Uses eye-safe laser beams ➔ create 3D representation of environment



A typical lidar sensor emits pulsed light waves into the surrounding environment

These pulses bounce off surrounding objects and return to the sensor



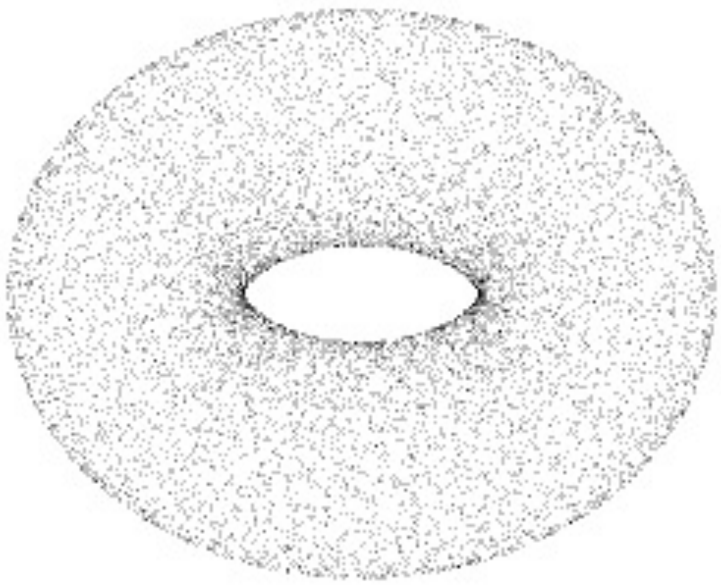
The sensor uses the time it took for each pulse to return to the sensor to calculate the distance it traveled

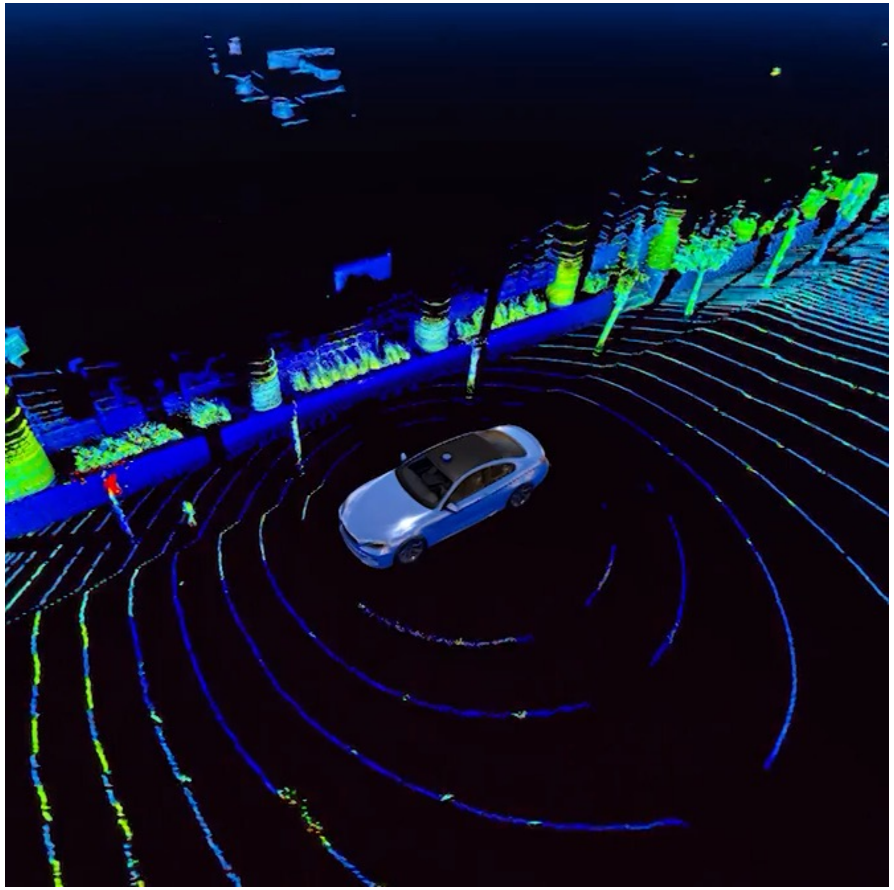
LiDAR Output

Point clouds In 3D

Range: **70-100** m

View: **360 degrees**





References

- SAE J3016 Standard:

https://sibin.github.io/teaching/cs599-osu-secure-autonomous-cps/winter_2022/other_docs/J3016_201609.pdf

- A better explanation of the standard and its components:

https://www.atlantis-press.com/journals/jase/125934832/view

Millimeter Wave Radar [mmWave]

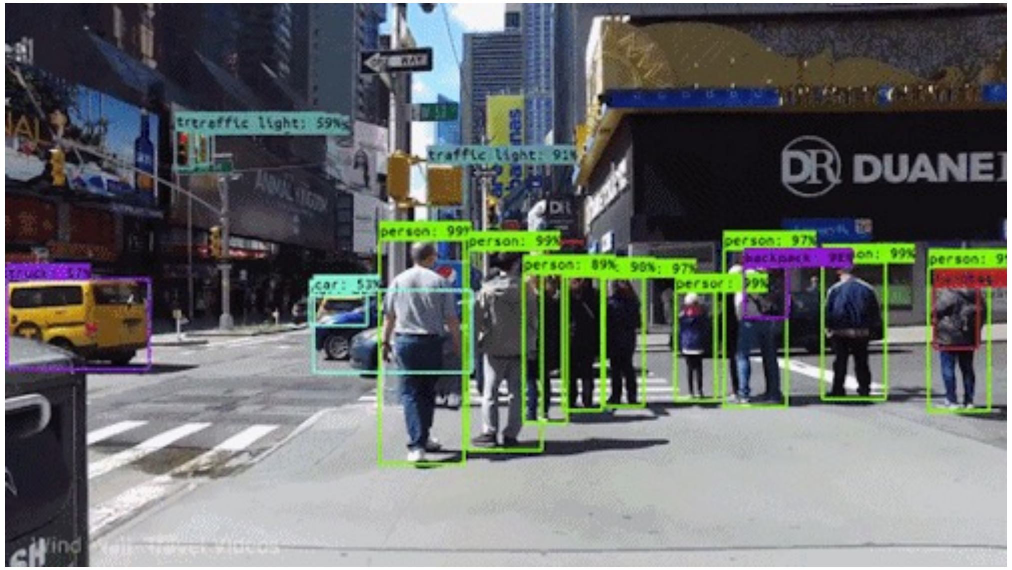
- Radar technology
- short-wavelength electromagnetic waves
- Measures reflected radar signals

- **High accuracy**
- 76-81 GHz ➔ detect movements in a **fraction of a millimeter**!
- Limited distance [< 80m]

- Also, used for in-cabin monitoring of drivers

Cameras

- Accurate way to create visual representations
- Front, left, right, rear cameras
  - to create **a 360-degree view**
- Main focus ➔ **object detection**

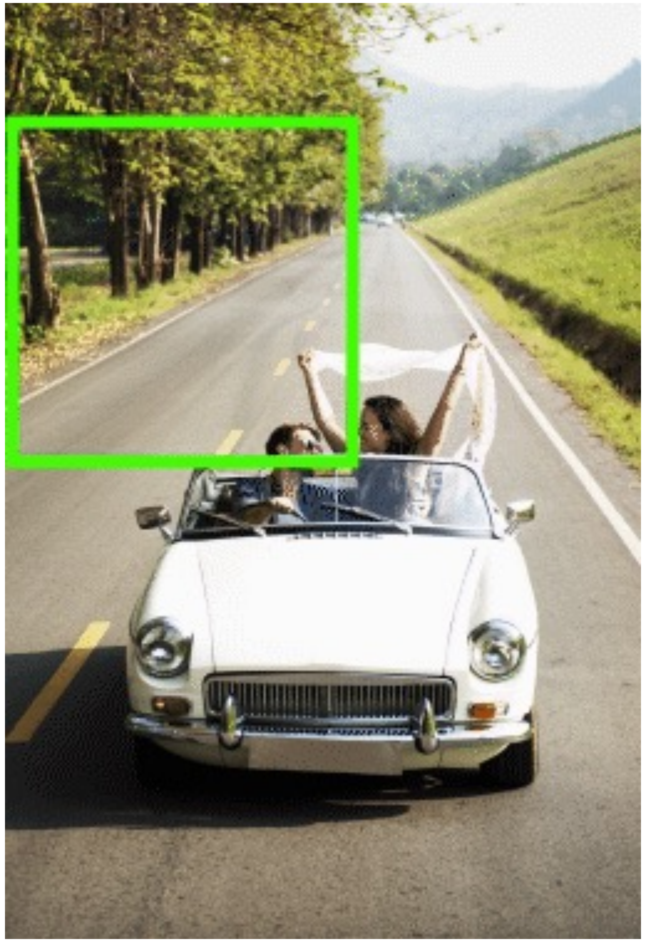


Cameras | Computer Vision

- Computer Vision algorithms for object detection
  1. Image classification → determine objects in an image are
  2. Image localizations → providing specific locations of image [bounding box]

Cameras | Image Classification

- Convolutional Neural Networks (CNNs)
  - **trained** to recognize objects like cars, pedestrians, etc.
  - performs **convolution operations at runtime**
    - to classify images from camera
- CNNs limited to single objects taking up entire image
- **Sliding Windows**!

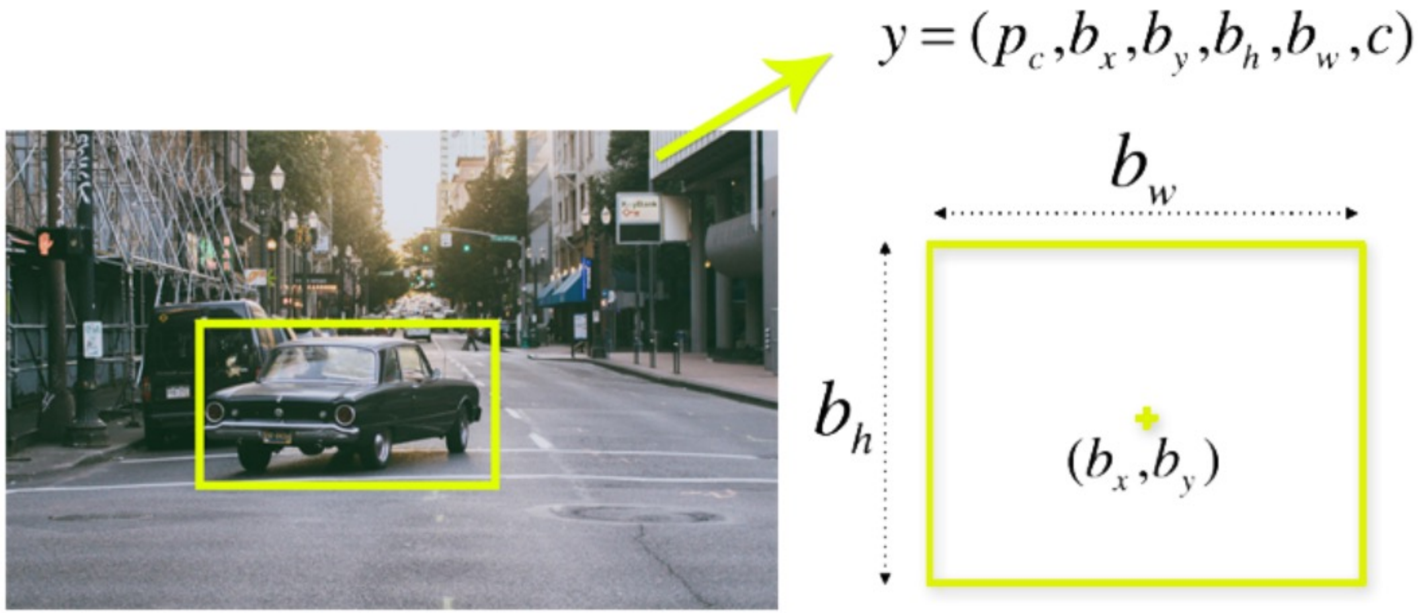


Cameras | Sliding Window Algorithm

What about objects much larger or much smaller than window size?

YOLO!

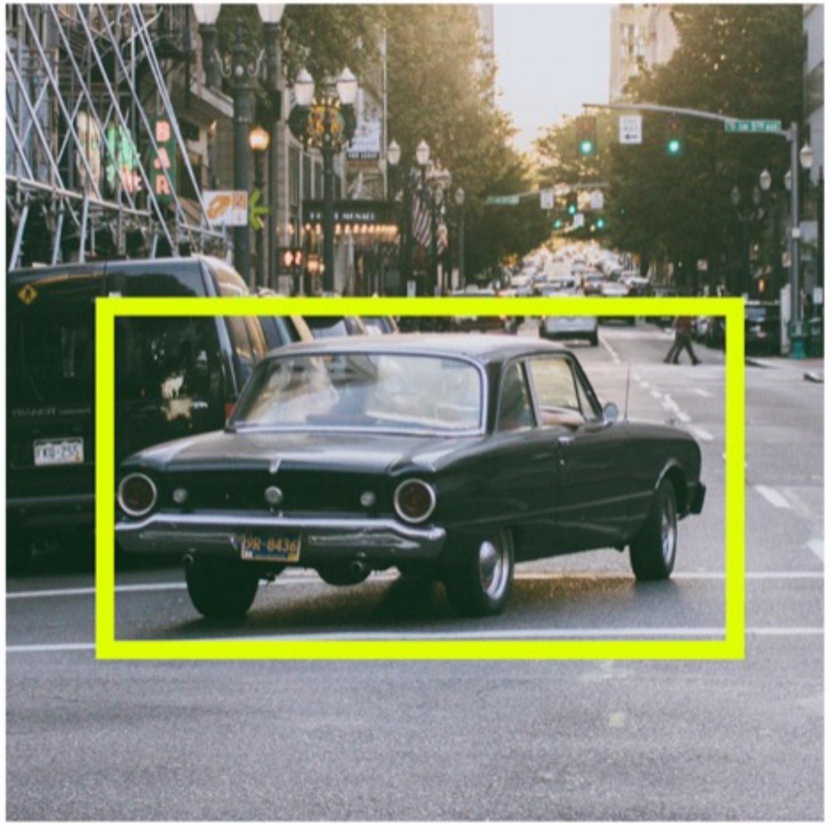
Cameras | YOLO



- "**you only look once**"
- Image split up into grid ➔ run once through CNN

Cameras | YOLO [contd.]

preprocessed image
(608, 608, 3)

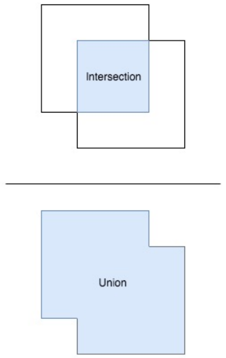


Cameras | YOLO | Image Localization

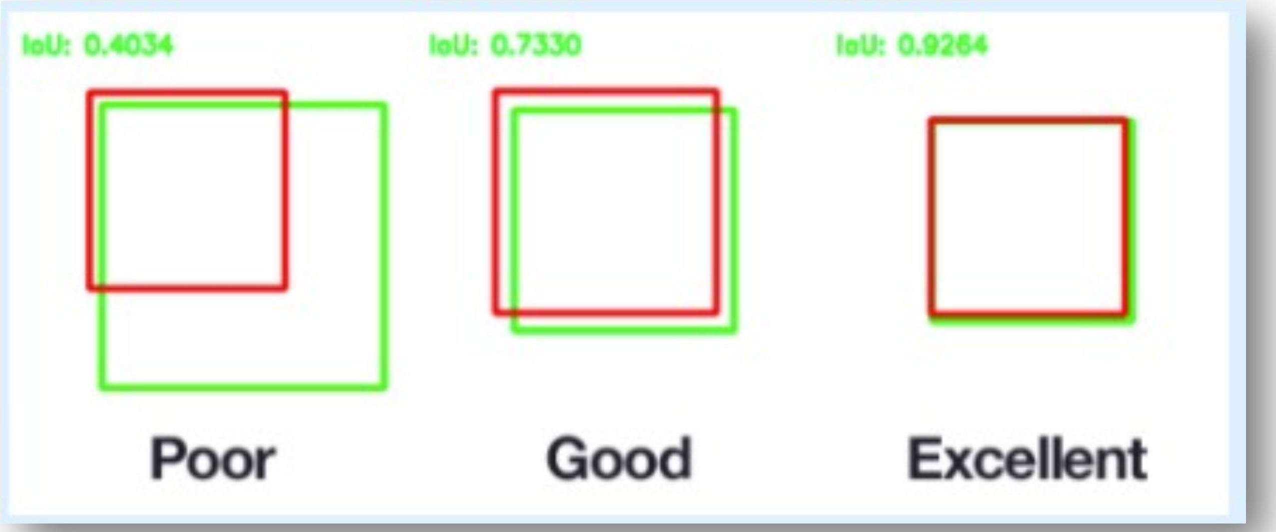
Cameras | YOLO | IoUs

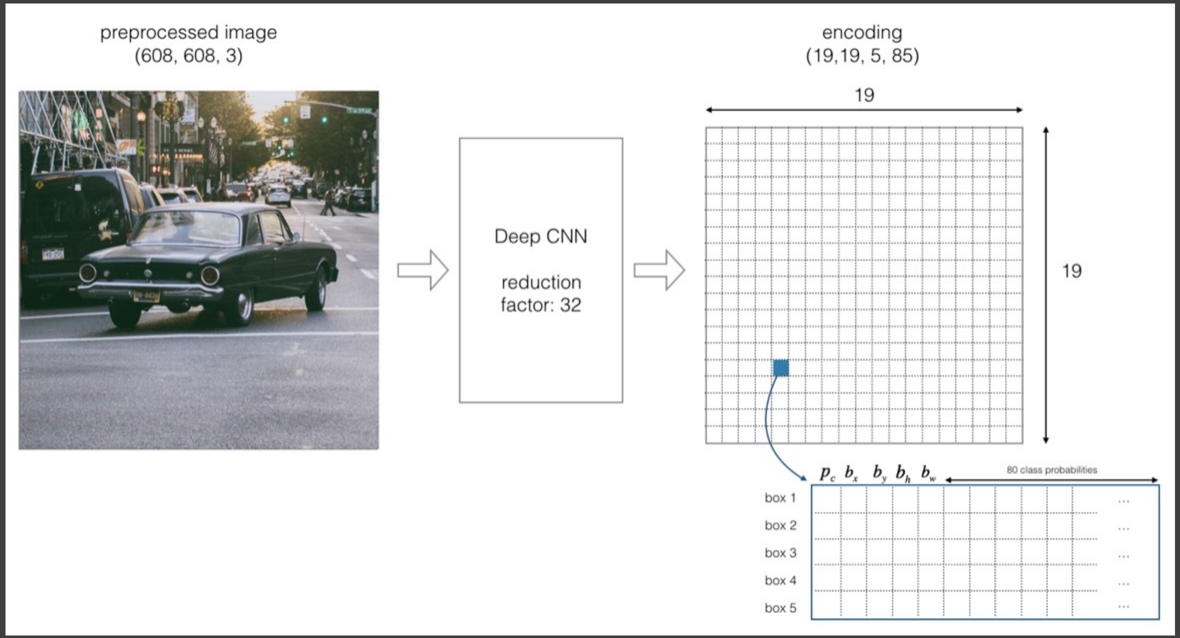
- During training ➔ compare CNN bounding box to **actual** ones
- Cost function, "**intersection over union**" (**IoU**)

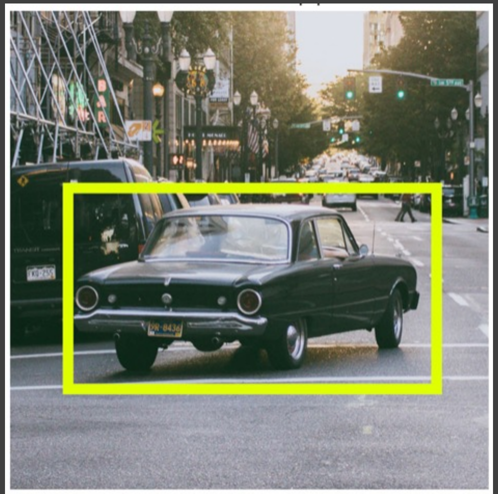
$$\boldsymbol{IoU} = \frac{area\ of\ \boldsymbol{intersection}\ of\ bounding\ boxes}{area\ of\ \boldsymbol{union}\ of\ bounding\ boxes}$$



- If IoU is closer to **1** ➔ better the bounding box







Cameras | YOLO | Non-Max Suppression

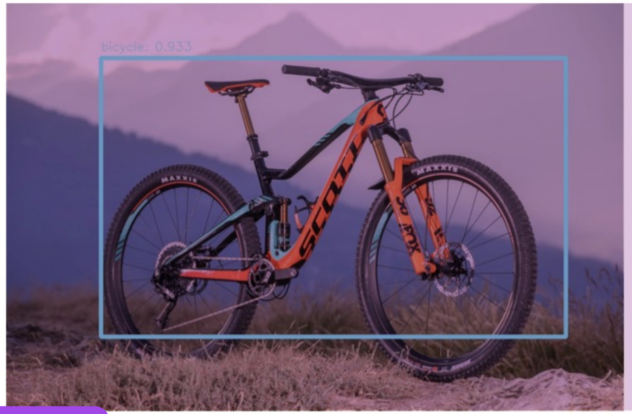
- Majority of the cells won't have bounding boxes
- Remove boxes with
  - low object probability
  - highest shared area
- **non-max suppression**
  - discard bounding boxes with probability less than threshold
    i.e. $p < 0.5$ or 0.6
  - take box with highest prediction value
  - discard/suppress boxes with IoU > threshold with that box
    i.e. 0.5 or 0.6
- suppress boxes that don't have maximum probability

Attacking Object Detectors?

- Falsify the training set
  - Larger impact
  - Harder to do – less public access
- Modify objects being detected
  - Add paint/tape/appendages to cars to that it presents differently
- Attack the inputs
  - Add stickers to objects
  - Add extraneous pixels/data to the camera inputsß

Attacking Object Detectors | Example

- Maximize loss of CNN classifier
- Maximize loss of object detector



40x40 patch

Cameras

- Additional cameras
  - Lane following
  - Traffic signal monitoring

Stereo Vision

- Problem with regular cameras+YOLO is **2D vision**
- “Fuse” camera data with LiDAR ➔ expensive

- Align **two cameras** and use **geometry**
- **Pseudo-LiDAR**

Stereo Vision

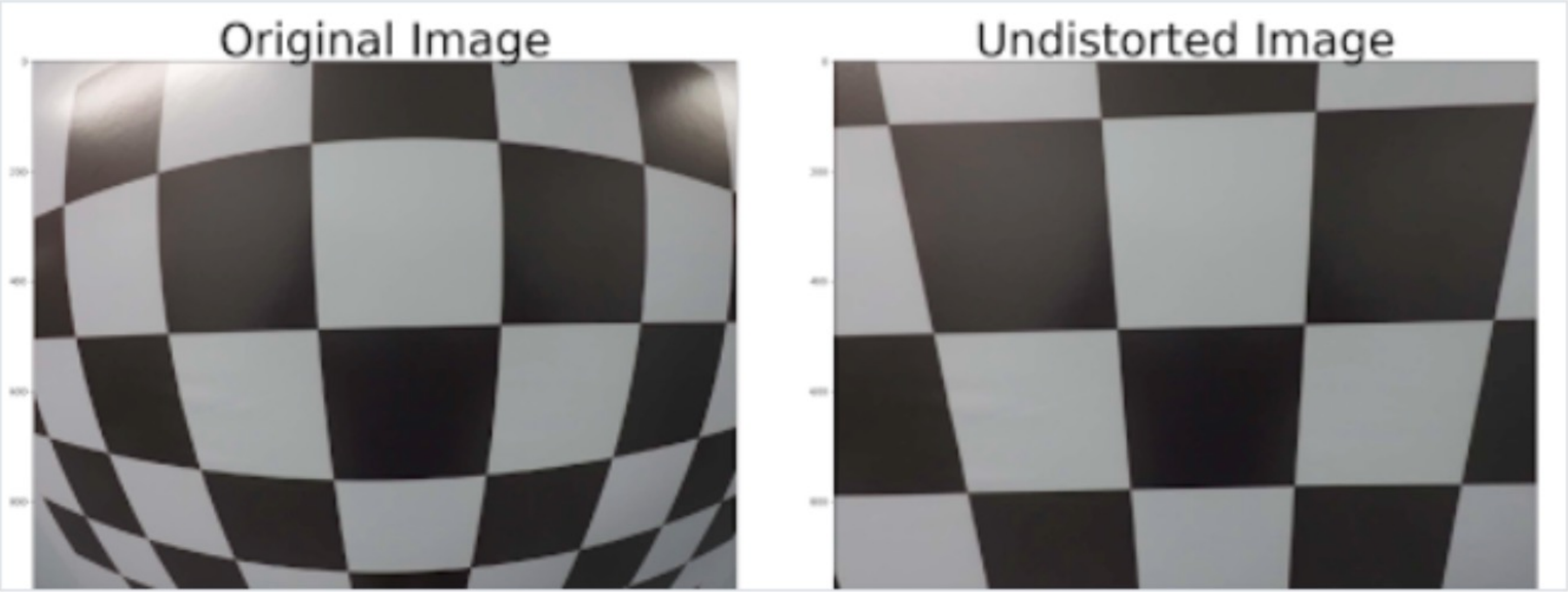
- Retrieve distance of an object using **two cameras** and **triangulation**

Steps

- Stereo calibration
- Epipolar geometry
- Disparity mapping
- Depth mapping
- Obstacle detection estimation

Stereo Vision | Calibration

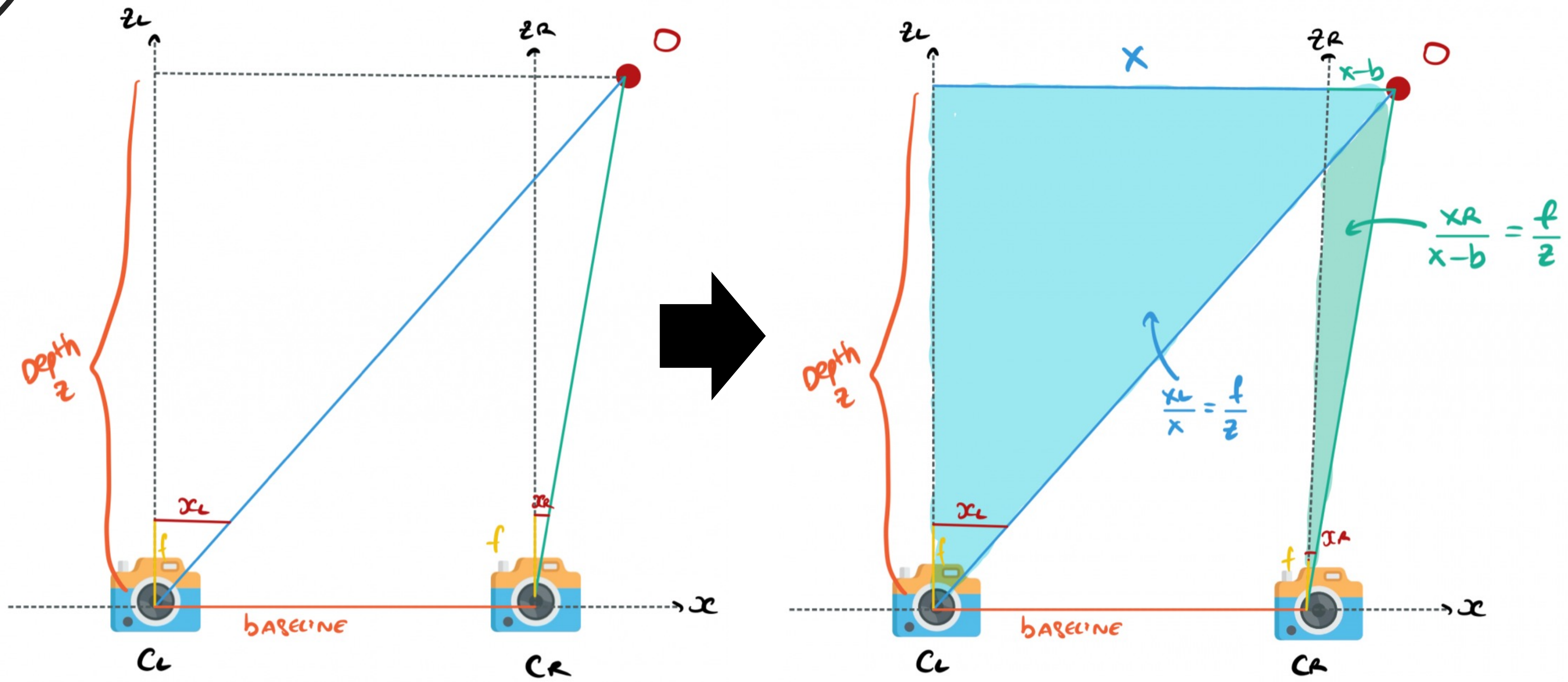
Create undistorted images from original camera ones



Stereo Vision | Calibration + Epipolar Geometry

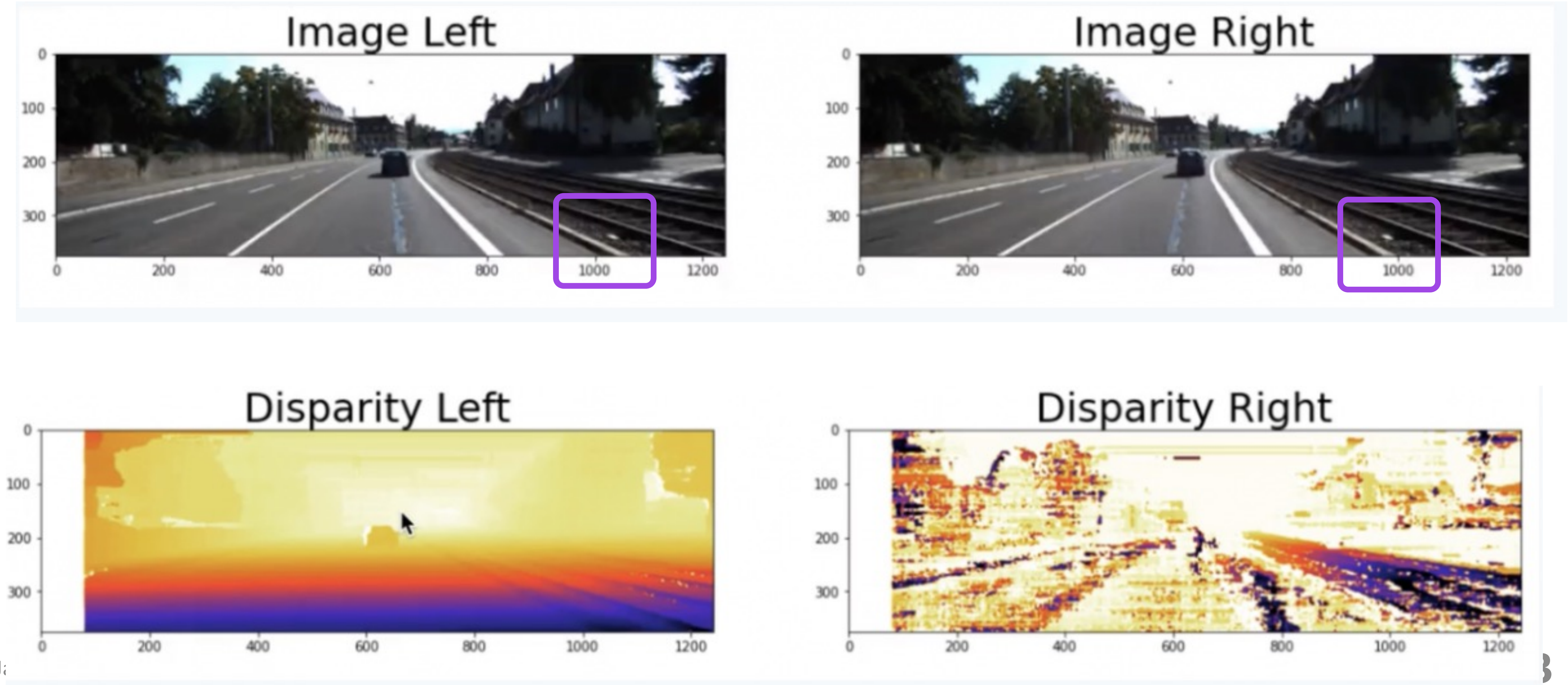
$$Z = \frac{f \cdot b}{x_L - x_R} = \frac{f \cdot b}{d}$$

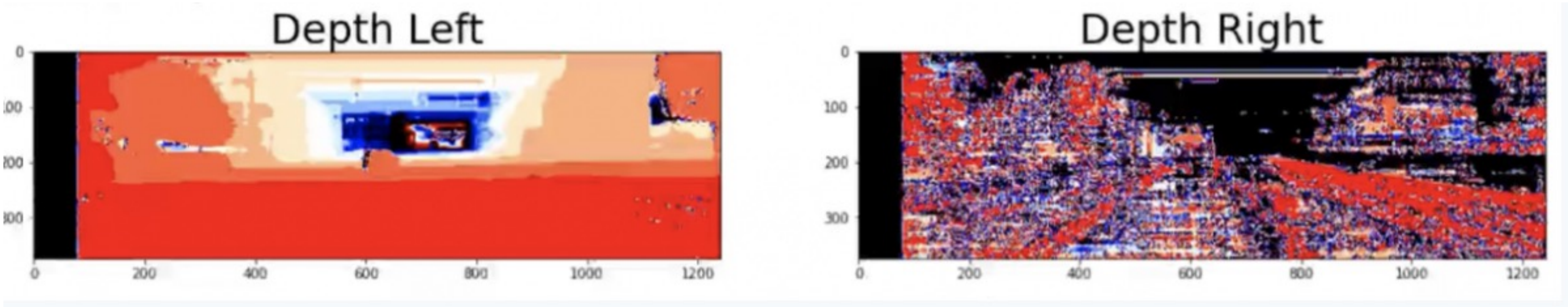
Geometry Calculations



Stereo Vision | Disparity Mapping

- Difference in image location of same 3D point from 2 camera angles





Stereo Vision | Depth Map

- **Distance of each pixel** in an image
  - Using other image+disparity map

Pseudo-LiDAR

Stereo Vision | Estimate Depth

- Using depth map, combine with YOLO
- E.g. run YOLO on left image and then use depth map
- In bounding box from YOLO, closest point can be taken

References

- mmWave

https://www.ti.com/lit/wp/spyy005a/spyy005a.pdf?ts=1641417836995&ref_url=https%253A%252F%252Fwww.google.com%252F

- Computer Vision/YOLO

https://medium.com/@albertlai631/how-do-self-driving-cars-see-13054aee2503

https://www.kdnuggets.com/2018/09/object-detection-image-classification-yolo.html

- Attack on YOLO paper

https://arxiv.org/pdf/1806.02299.pdf

- Stereo Vision/Pseudo LiDAR

https://www.thinkautonomous.ai/blog/?p=pseudo-lidar-stereo-vision-for-self-driving-cars