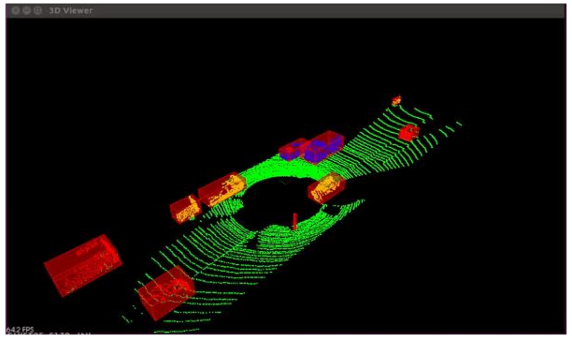
Sensors and object clustering

Nacho Sañudo
University of Modena and Reggio Emilia
Ignacio.sanudoolmedo@unimore.it

Euclidean clustering detection

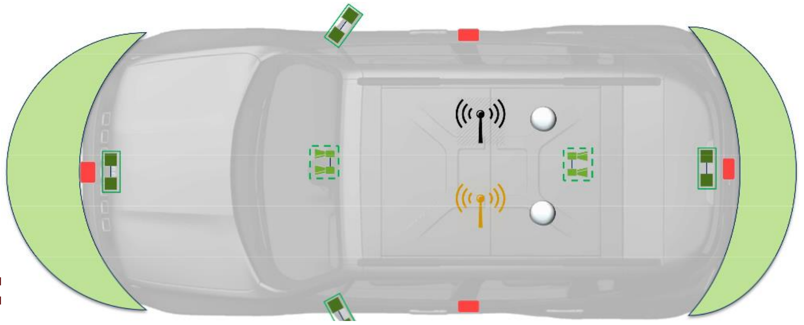


› **Objective**: Find and segment the individual object point clusters lying on the plane

1. Down sample the point cloud
2. Segment the ground plane
3. Create a KD-tree representation for the input point cloud dataset
4. Compute Euclidean distance to build the cluster list
5. The algorithm terminates when all points have been processed and are now part of the list of point clusters

```
sudo apt-get install pcl-tools
sudo apt install libpcl-dev
```

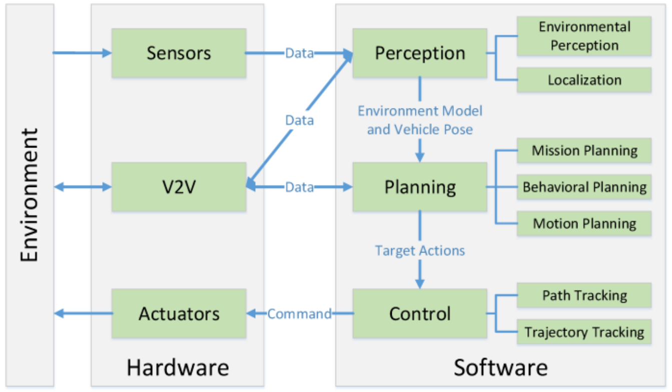
HiPeRT Car

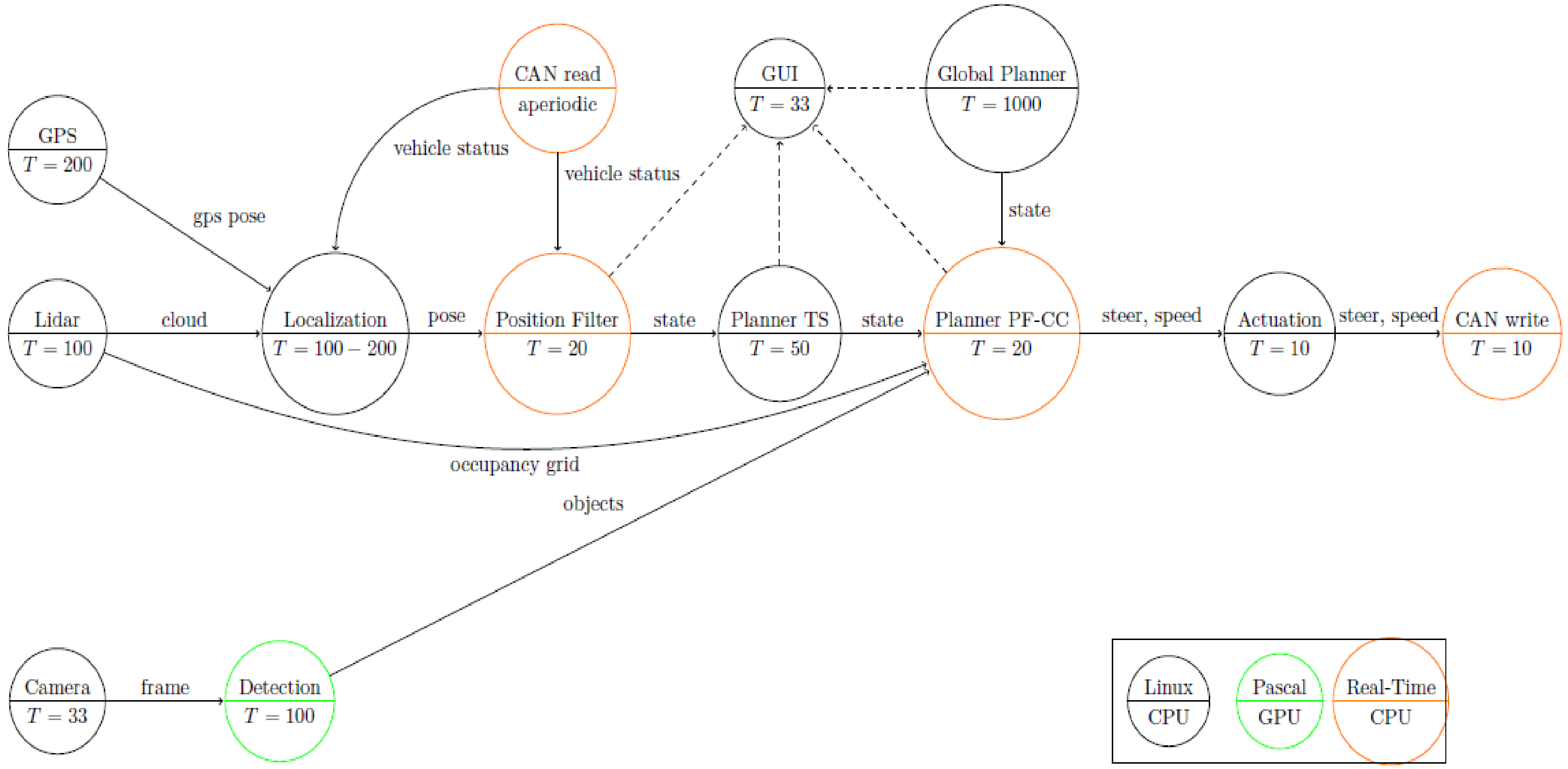


- **Maserati quattroporte**
  - 5 sekonix cameras (GMSL)
  - Ouster OS1 64/128-120m (Ethernet)
  - GPS (CAN)
  - Radar ARS301
  - We have different configurations
    - Pegasus
    - Drive PX2
    - Xavier
    - TX2

https://www.youtube.com/watch?v=qkWqRSgUFmw

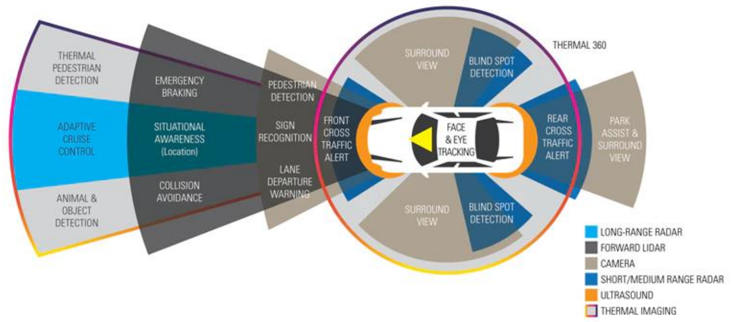
AD stack


Environment
Sensors
V2V
Actuators
Hardware
Data
Data
Data
Command
Perception
Environmental Perception
Localization
Environment Model and Vehicle Pose
Planning
Mission Planning
Behavioral Planning
Motion Planning
Target Actions
Control
Path Tracking
Trajectory Tracking
Software

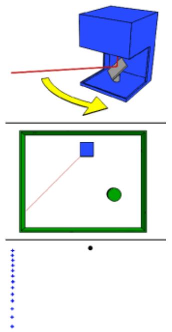

GPS
T = 200
gps pose
Lidar
T = 100
cloud
Localization
T = 100 – 200
pose
CAN read
aperiodic
vehicle status
vehicle status
Position Filter
T = 20
state
GUI
T = 33
Global Planner
T = 1000
state
Planner TS
T = 50
state
Planner PF-CC
T = 20
steer, speed
Actuation
T = 10
steer, speed
CAN write
T = 10
occupancy grid
objects
Camera
T = 33
frame
Detection
T = 100
Linux
CPU
Pascal
GPU
Real-Time
CPU

Sensors

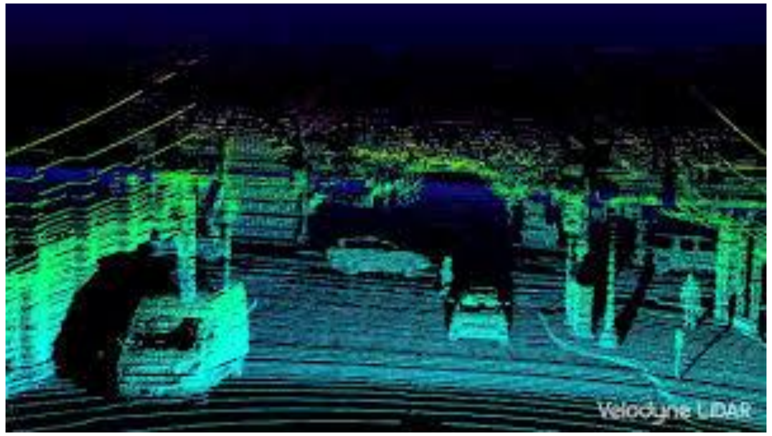
- Sensor device that measures a property of the environment
  - **Exteroceptive**: measure the propierties of the external environment
  - **Propioceptive**: measure the properties of the ego vehicle
- The number and complexity of the sensors has increased drastically over the past few years



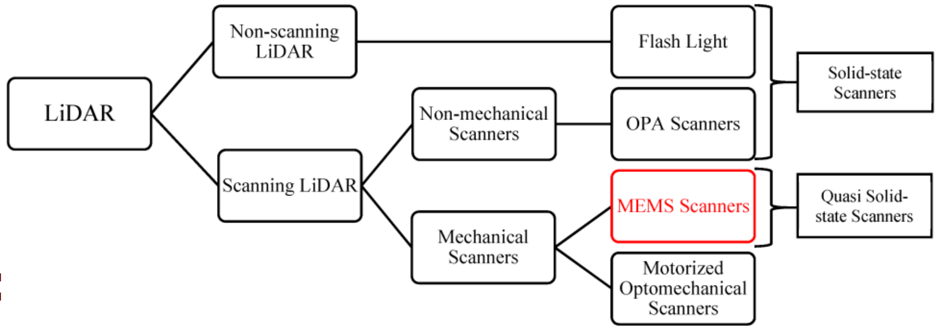
LiDAR



- Light Detection And Ranging (LiDAR) is a technology used to scan objects, **measure distances and height**
- A lidar sensor transmits **laser beams** that bounce off objects and return to the sensor
  - The distance of the objects is determined based on how long it takes the pulses of infrared light to travel back
- Using the information received, a lidar system produces a **point cloud** that looks like a shadow and reflects the object’s shape and size
  - HDL-32E generates up to 1.4 million points per second
  - They are affected by bad weather conditions
  - Very expensive
- They are generally used to make high-resolution maps and localize the car in a map but also for **object classification**
  - Computer vision can help in the classification (semantic/classification)



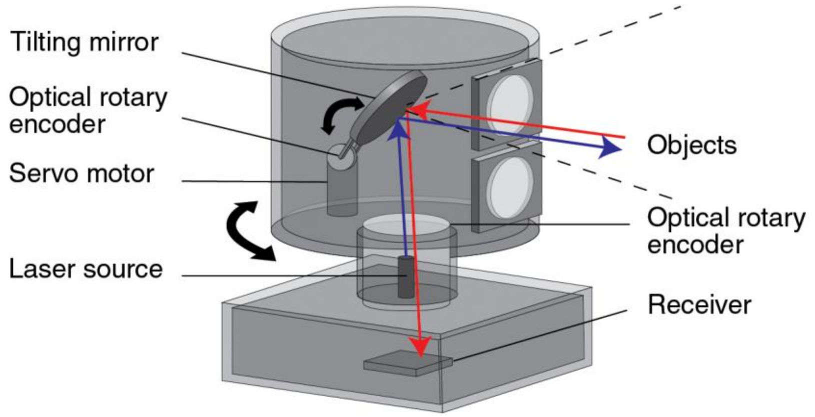
Source: Velodyne



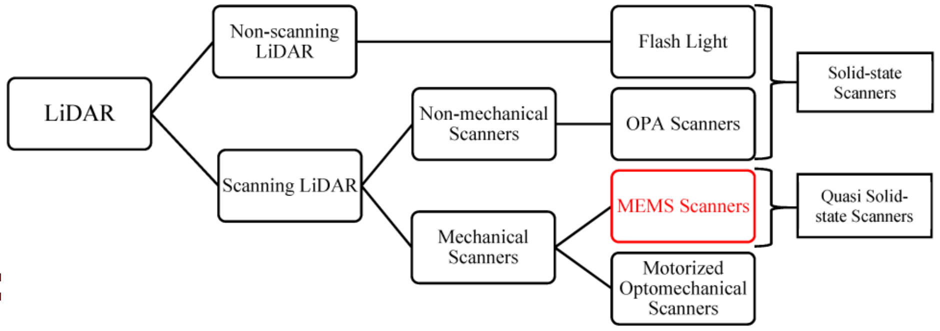
Source: https://www.mdpi.com

Type of LiDARs

› **Motorized optomechanical scanners** are the most common type of LiDAR scanners
  - Uses optics and a rotation motor to create a 360º FOV
  - High cost of mechanical parts
  - Mapping

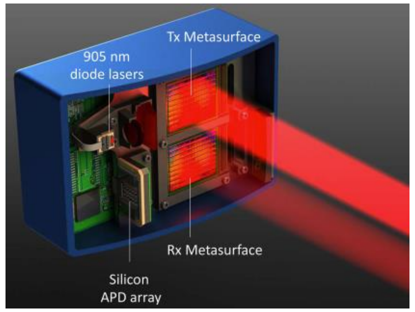


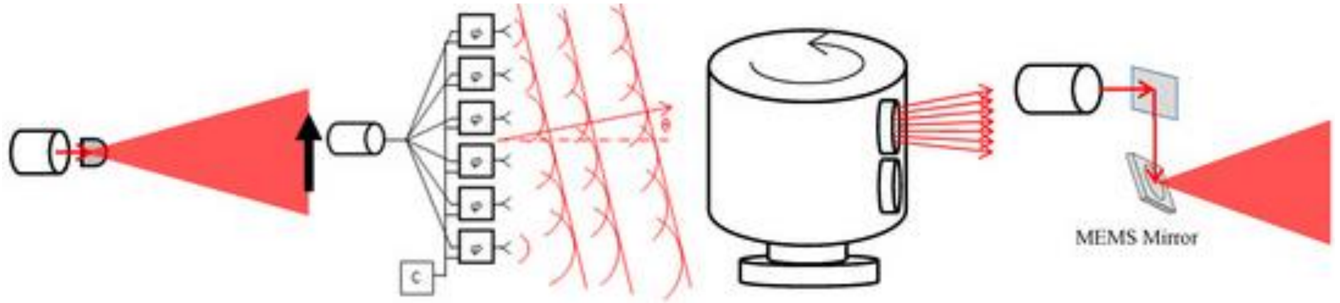
Type of LiDARs



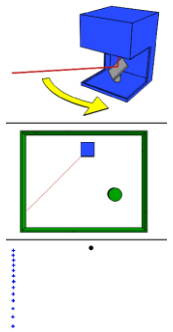
Source: https://www.mdpi.com

- **Solid state LiDAR** do not require spinning mechanical components (consequently they have a reduced FOV)
- Uses all solid-state components, which are built using no moving parts (they are resistant to vibrations, a compact size, and low price)
  - Non-scanning LiDAR (also called **Flash LiDAR**) -- No mechanical parts
    - 2D FoV of interest is entirely illuminated by the laser source (like a camera with a flashlight)
  - Optical phase array (**OPA**) – Multiple lens
    - an optical phase modulator controls the speed of light passing through the multiple lens
      - This enables to control the direction of the laser beam
  - Mirror-Based Quasi Solid-State LiDAR (**MEMS)**
    - MEMS mirrors can steer, modulate, and switch light, as well as control phase

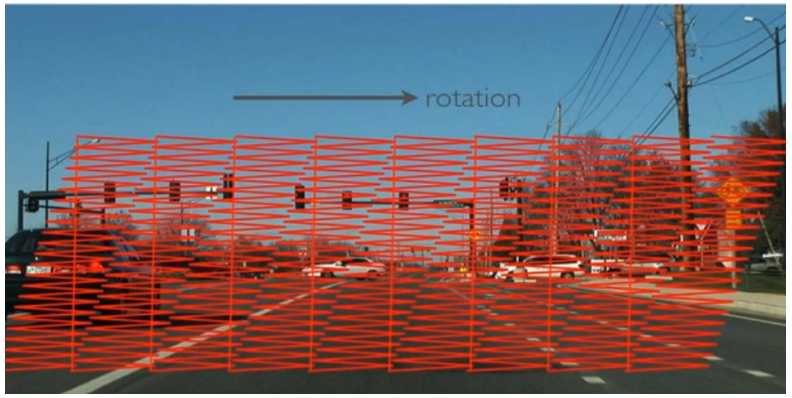




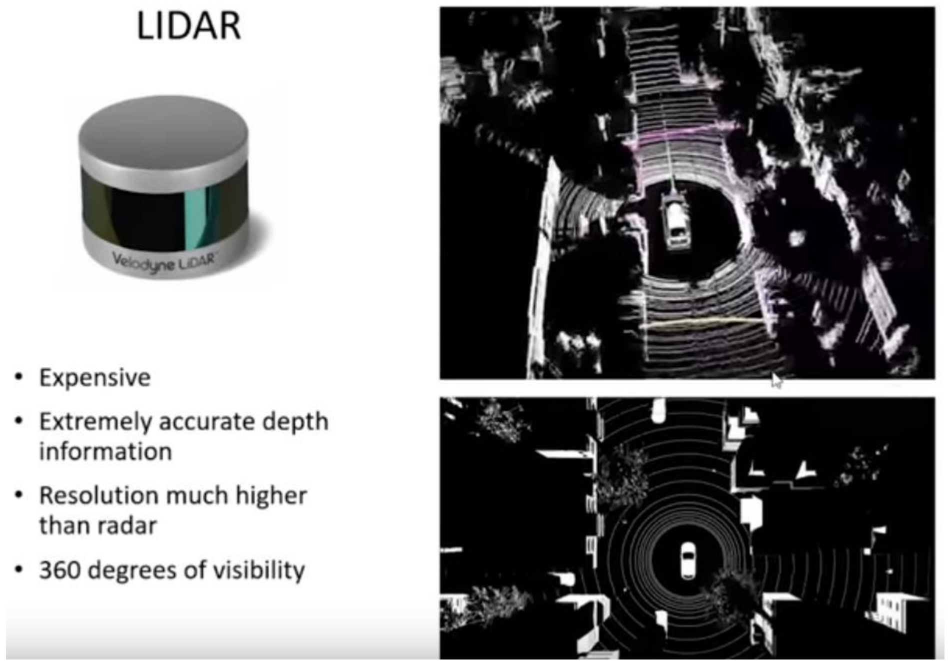
LiDAR



- Metrics:
  - Number of beams
  - Points per second (the more points the more detailed map can be built)
  - Rotation rate (hz)
  - Detection range
  - Field of view (angle)

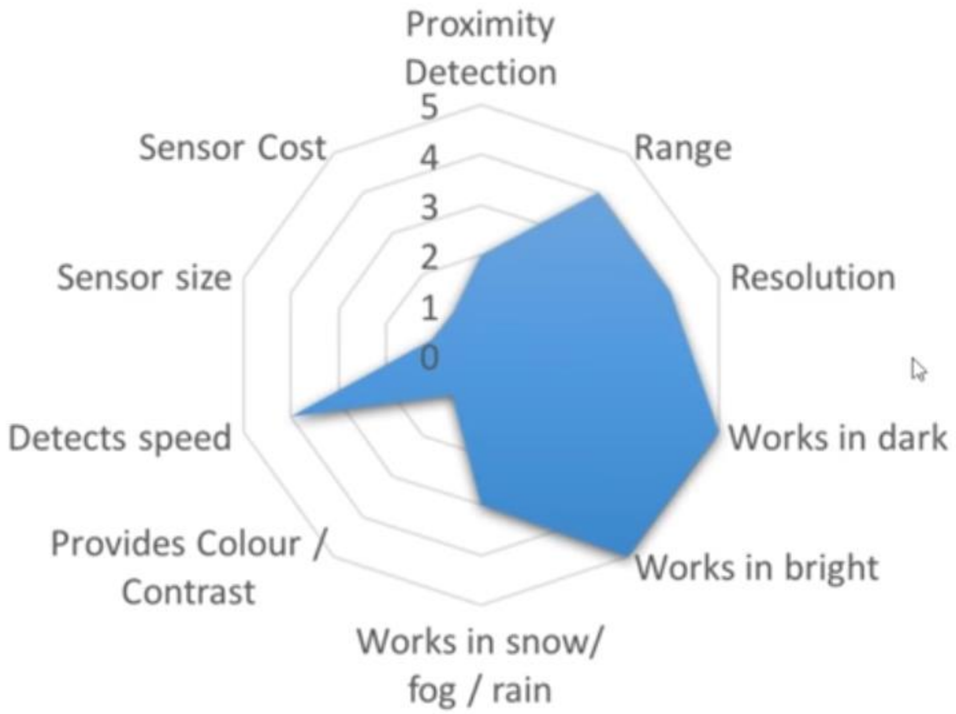


LiDAR



LIDAR

- Expensive
- Extremely accurate depth information
- Resolution much higher than radar
- 360 degrees of visibility



LiDAR players

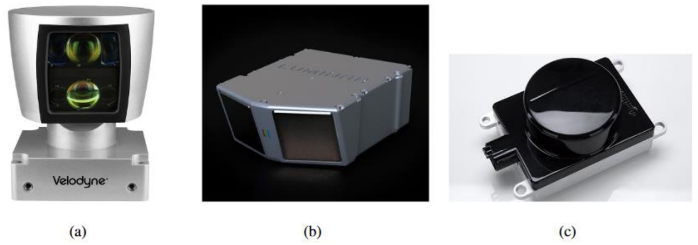
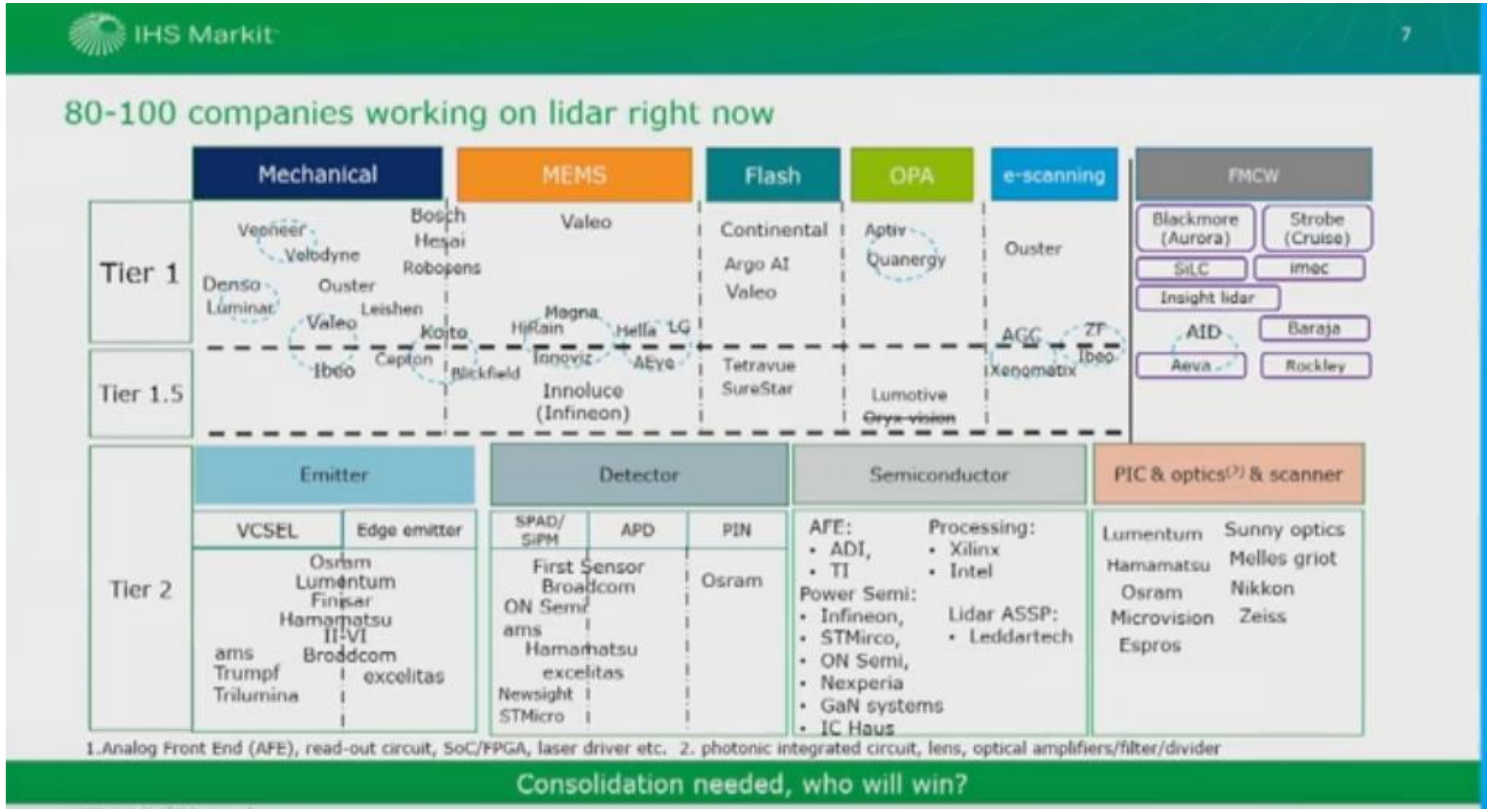


Fig. 5: LiDAR product examples: (a) mechanical spinning 905nm LiDAR from Velodyne, (b) 1550nm MEMS LiDAR from Luminar (c) Flash LiDAR from Continental



IHS Markit 7

80-100 companies working on lidar right now

	Mechanical	MEMS	Flash	OPA	e-scanning	FMCW
Tier 1	Veoneer, Velodyne, Denso, Luminar, Ouster, Leishen, Valeo, Bosch, Hesai, Robosens, Koito	Valeo, Magna, HiRain, Hella, LG	Continental, Argo AI, Valeo	Aptiv, Quanergy	Ouster, AGC, ZF	Blackmore (Aurora), Strobe (Cruise), SiLC, imec, Insight lidar, AID, Baraja, Aeva, Rockley
Tier 1.5	Ibeo, Cepton	Blickfeld, Innoviz, AEye, Innoluce (Infineon)	Tetravue, SureStar	Lumotive, ~~Oryx vision~~	Xenomatix, Ibeo	

	Emitter		Detector			Semiconductor	PIC & optics(2) & scanner
	VCSEL	Edge emitter	SPAD/ SiPM	APD	PIN		
Tier 2	Osram, Lumentum, Finisar, Hamamatsu, II-VI, ams, Broadcom, Trumpf, Trilumina	Osram, Lumentum, Finisar, Hamamatsu, II-VI, Broadcom, excelitas	First Sensor, Broadcom, ON Semi, ams, Hamamatsu, excelitas, Newsight, STMicro	First Sensor, Broadcom, Hamamatsu, excelitas	Osram	AFE: • ADI, • TI; Power Semi: • Infineon, • STMirco, • ON Semi, • Nexperia • GaN systems • IC Haus; Processing: • Xilinx • Intel; Lidar ASSP: • Leddartech	Lumentum, Hamamatsu, Osram, Microvision, Espros, Sunny optics, Melles griot, Nikkon, Zeiss

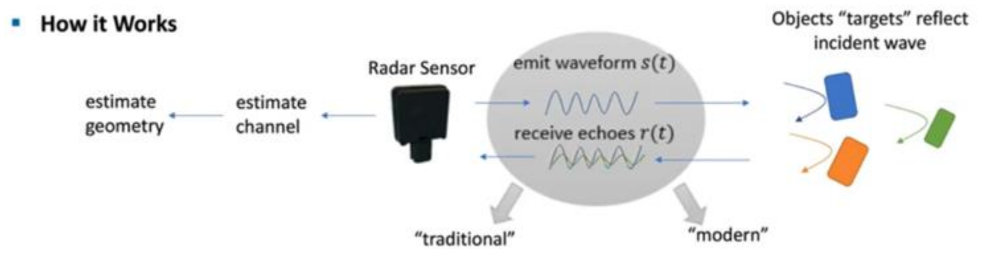
1.Analog Front End (AFE), read-out circuit, SoC/FPGA, laser driver etc. 2. photonic integrated circuit, lens, optical amplifiers/filter/divider

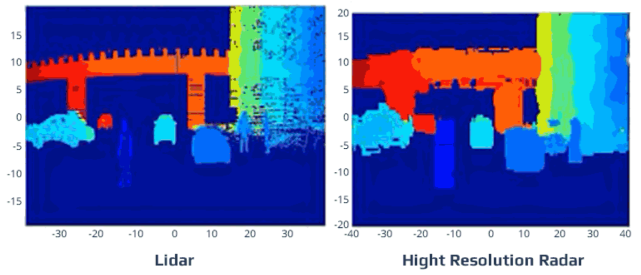
Consolidation needed, who will win?

Source: AutoSens

Radar

- Radio Detection and Ranging (Radar) is a technology system that uses **radio waves** to detect objects, their distance, angle and speed in relation to the ego-vehicle
  - FMCW sends out a chirp (a pulse whose frequency rises during its transmission)
  - The difference between the frequency of the chirp coming out of the transmitter and the frequency of the received reflection) is related to the distance from the transmitter to the object
- Compared to lidar, radar point clouds have lower accuracy and worst resolution
  - They are robust and reliable, unlike lidar radar isn't affected by weather conditions
  - Cheap and power efficent
- Radars are currently deployed in many commercial vehicles to provide different ADAS features (parking assistance, cruise control or collision avoidance)

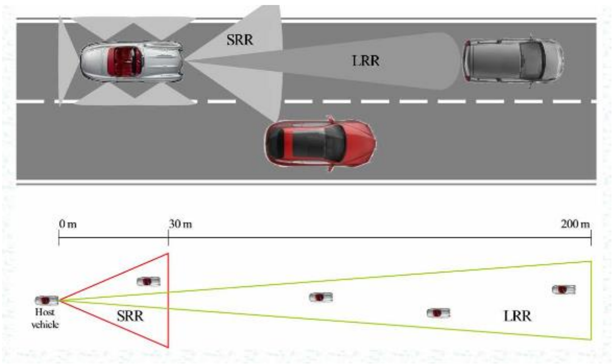


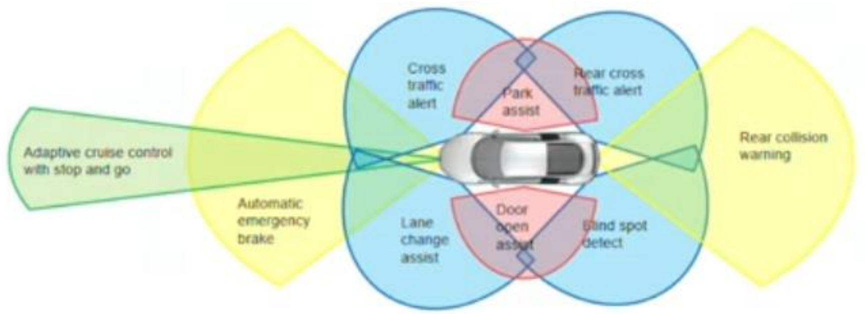


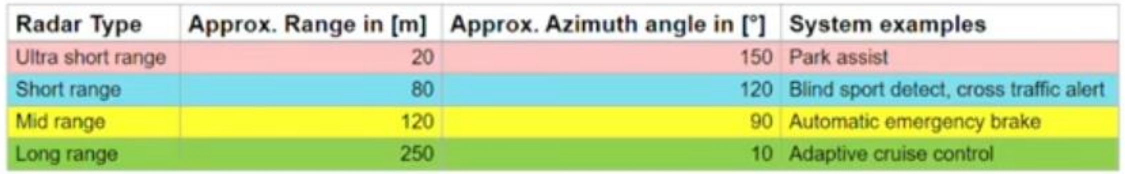
Source: Automotive Radar. Maria S. Greco & IEEE

Radar

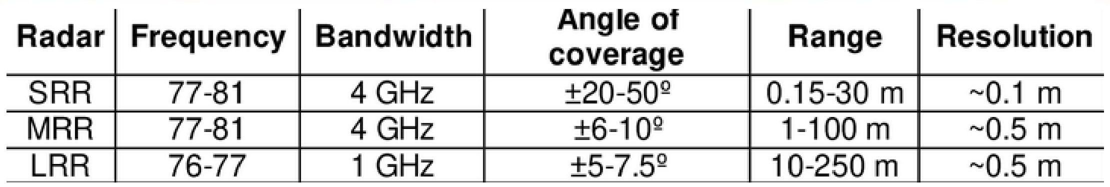
- Metrics
  - Range
    - WFOV, short range
    - NFOV, long range
  - Field of view
  - Position and speed accuracy
- Different types of radars
  - **SRR** – Short Range Radars
    - **Range** – approximately 30m
  - **MRR** – Mid Range Radars
    - **Range** – approximately 60m
  - **LRR** – Long Range Radars
    - **Range** – approximately 250m







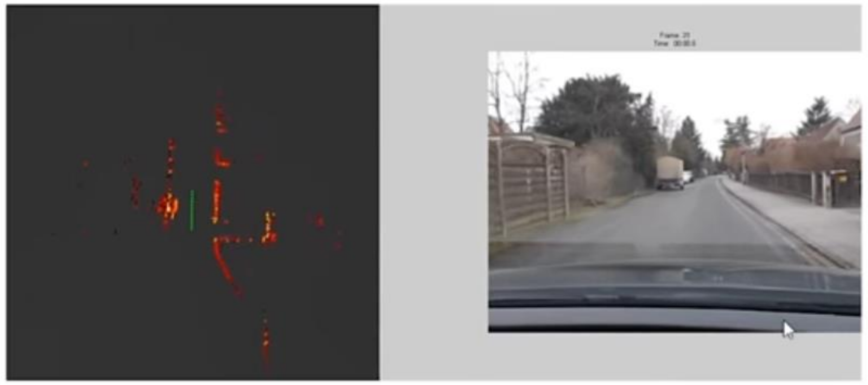
Radar Type	Approx. Range in [m]	Approx. Azimuth angle in [°]	System examples
Ultra short range	20	150	Park assist
Short range	80	120	Blind sport detect, cross traffic alert
Mid range	120	90	Automatic emergency brake
Long range	250	10	Adaptive cruise control



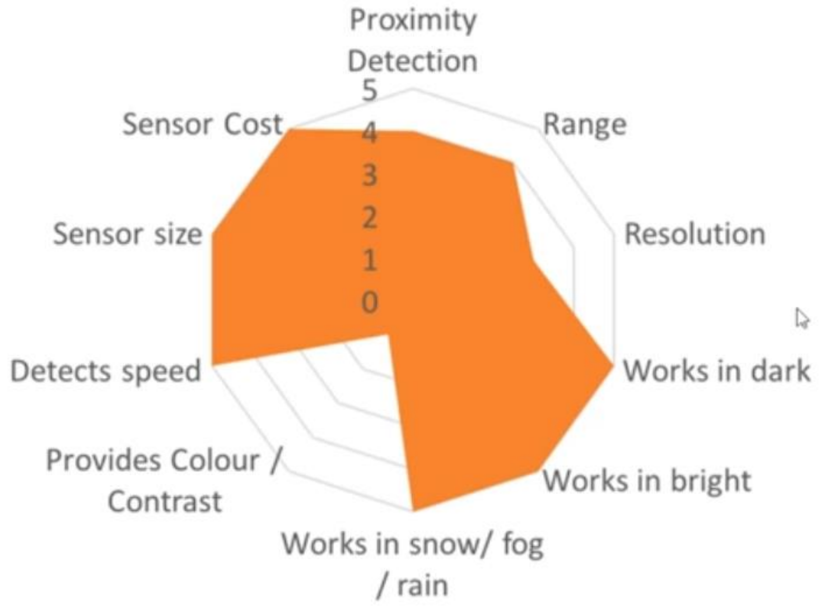
Radar	Frequency	Bandwidth	Angle of coverage	Range	Resolution
SRR	77-81	4 GHz	±20-50º	0.15-30 m	~0.1 m
MRR	77-81	4 GHz	±6-10º	1-100 m	~0.5 m
LRR	76-77	1 GHz	±5-7.5º	10-250 m	~0.5 m

Source: Automotive Radar. Maria S. Greco & AutoSens

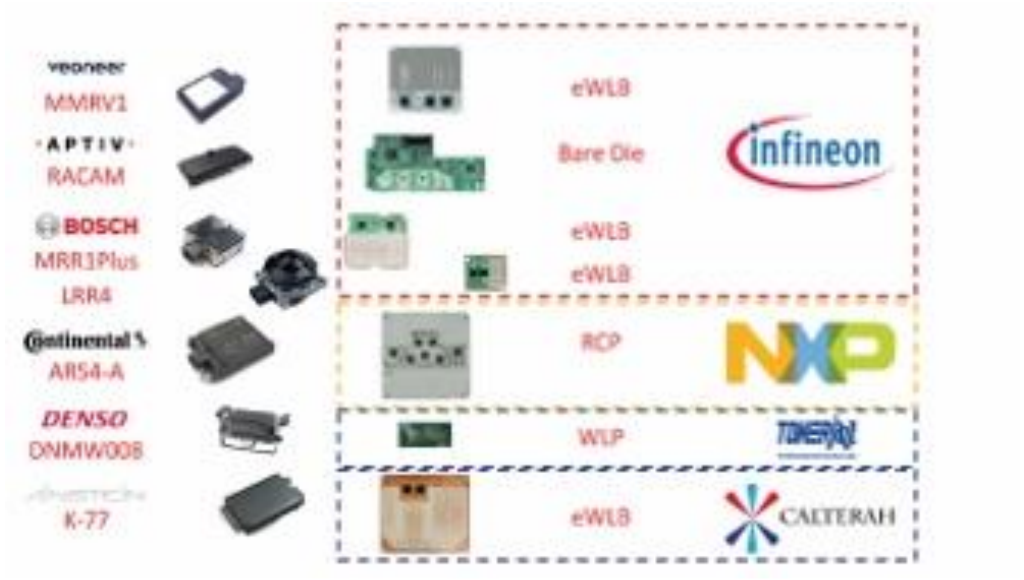
Radar

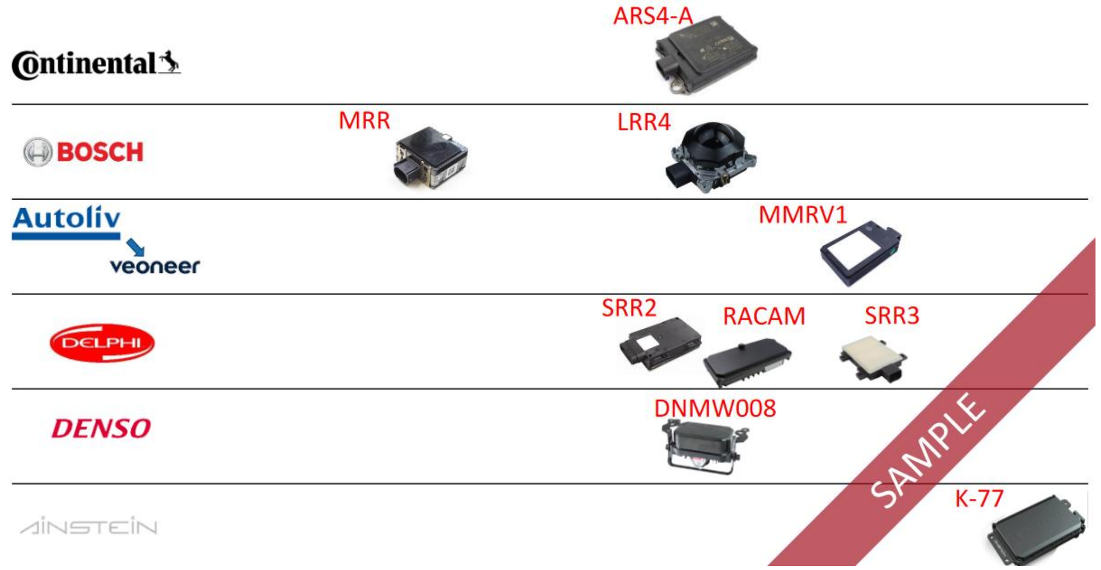


- Cheap
- Does well in extreme weather
- Low resolution
- Most used automotive sensor for object detection and tracking



Radar Players

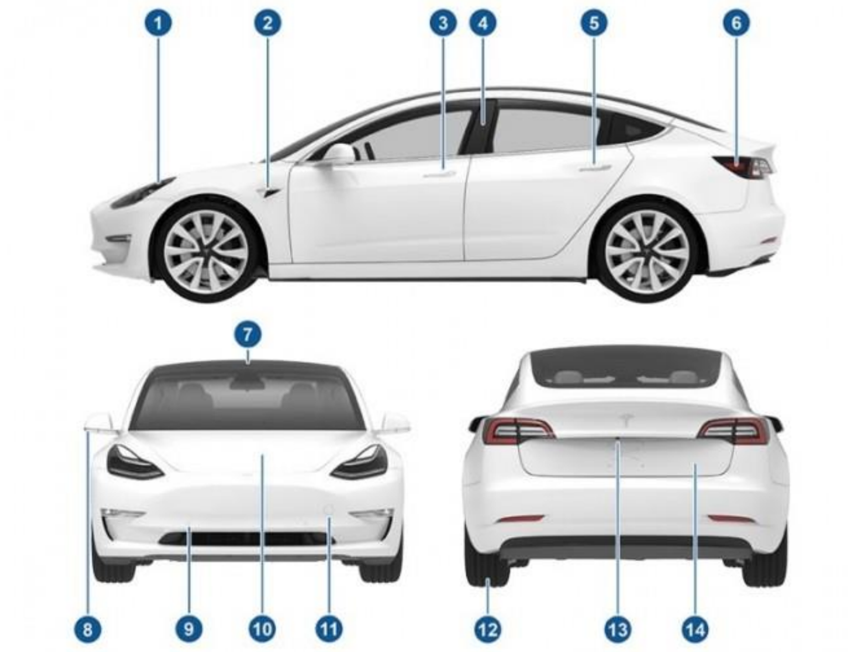




Source: SystemPlus

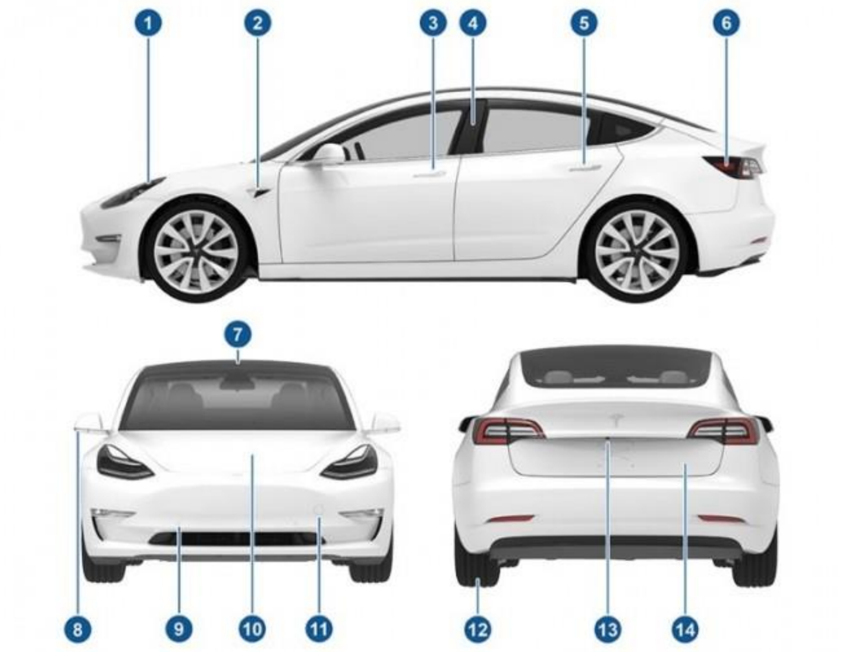
Monocular camera

- A monocular camera is a device with **one lenses**
- It is designed to capture images for enhancing the safety of the driver
- They are generally used to **detect and track objects** or **detect lane boundaries**
- Some companies use omni directional cameras to **localize** the vehicle
- Elon Musk: *"Anyone relying on lidar is doomed.“*



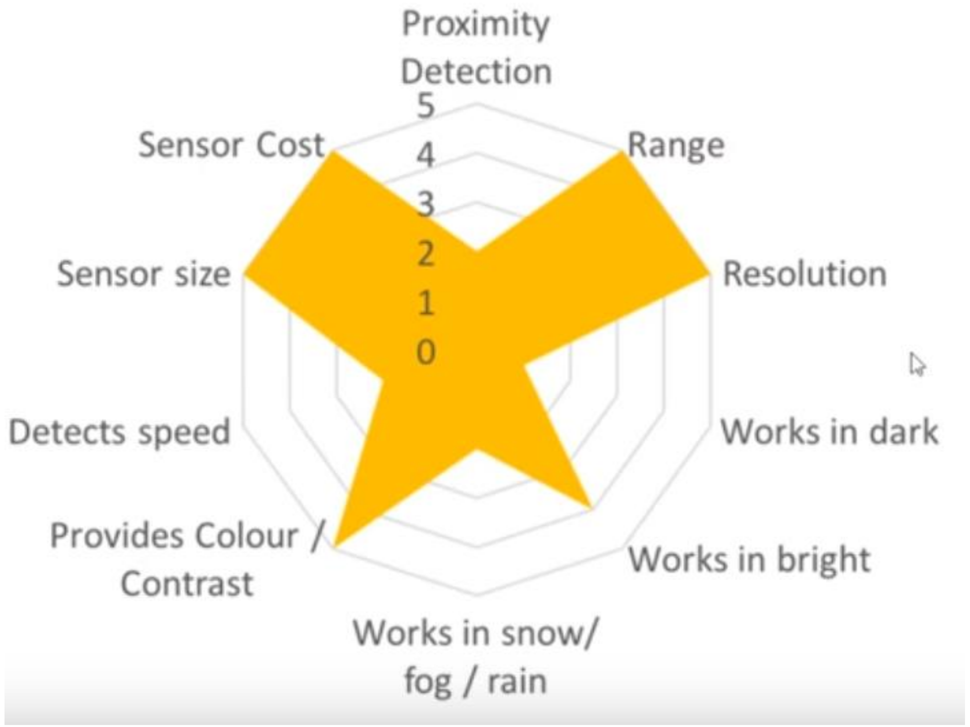
Monocular camera

- Metrics:
  - Resolution (number of pixels that create an image)
  - Field of view (horizontal and vertical angle extent that is visible to the camera)
  - Dynamic range
  - Frame rate (FPS)



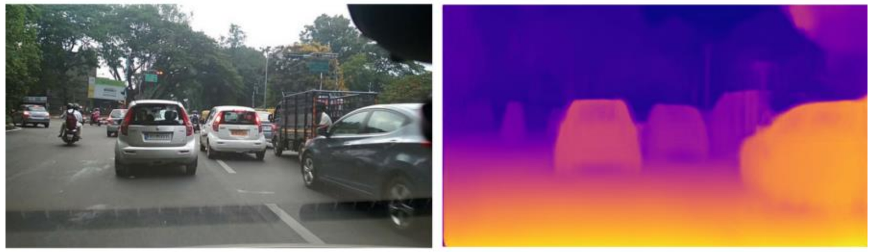
Monocular camera

- Cheap
- Highest resolution
- Huge data = deep learning
- Human brains use similar sensor technology for driving
- Bad at depth estimation
- Not good in extreme weather

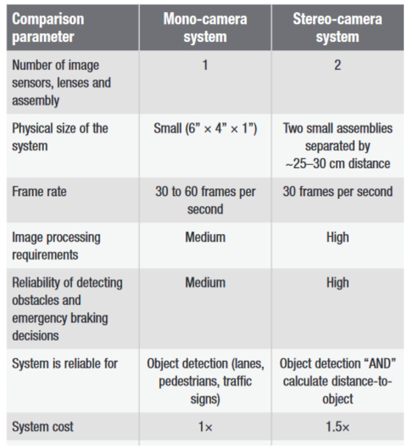


Stereo camera

- A stereo camera is a camera device with two or more lenses
  - This allows the camera to simulate human binocular vision, and therefore gives it the ability to capture three-dimensional images
  - Generally cars use just two lenses on top of the windshield
  - The lenses are at some distance from each other to see objects from different angles
- By comparing the two images produced by the lenses, **the relative depth information can be obtained in the form of a <u>disparity map</u>**, which encodes the difference in horizontal coordinates of corresponding image points
- Hence, stereo cameras are predominantly used to estimate the distance of objects in an image → **depth map**
  - A depth map is an image where each pixel has depth information



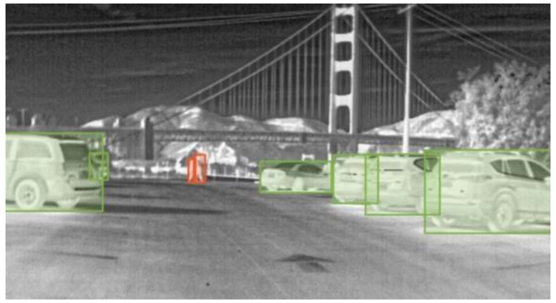
Source: Wikipedia & Bosch



Comparison parameter	Mono-camera system	Stereo-camera system
Number of image sensors, lenses and assembly	1	2
Physical size of the system	Small (6” × 4” × 1”)	Two small assemblies separated by ~25–30 cm distance
Frame rate	30 to 60 frames per second	30 frames per second
Image processing requirements	Medium	High
Reliability of detecting obstacles and emergency braking decisions	Medium	High
System is reliable for	Object detection (lanes, pedestrians, traffic signs)	Object detection “AND” calculate distance-to-object
System cost	1×	1.5×

Thermal camera

- A **thermographic camera** (also called an infrared camera or thermal camera) is a device that creates an image using infrared radiation, as a common camera that forms an image using visible light
- Thermal Cameras make pictures **from heat, not visible light**. Heat and light are both parts of the electromagnetic spectrum, but a camera that can detect visible light won't see thermal energy, and vice versa
- **Used to implement night vision systems**
  - An automotive night vision system uses a thermographic camera to **increase a driver's perception and seeing distance in darkness or poor weather beyond the reach of the vehicle's headlights**
  - Such systems are offered as optional equipment on certain premium vehicles

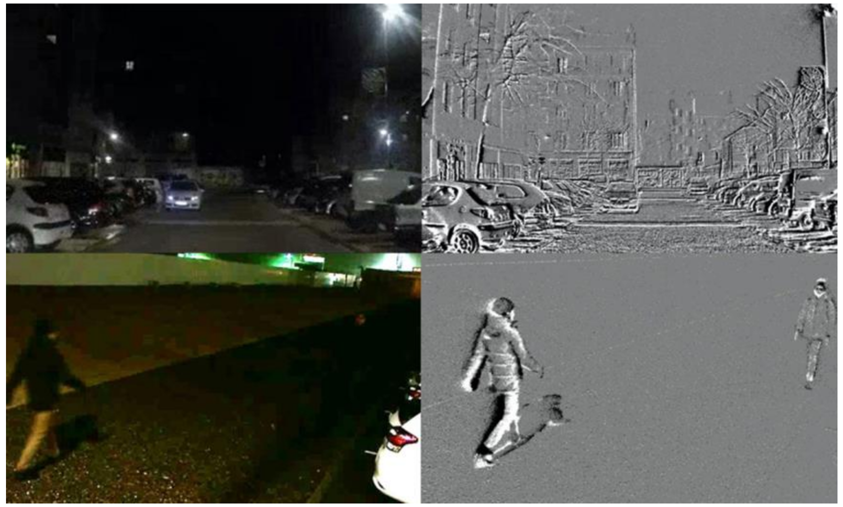


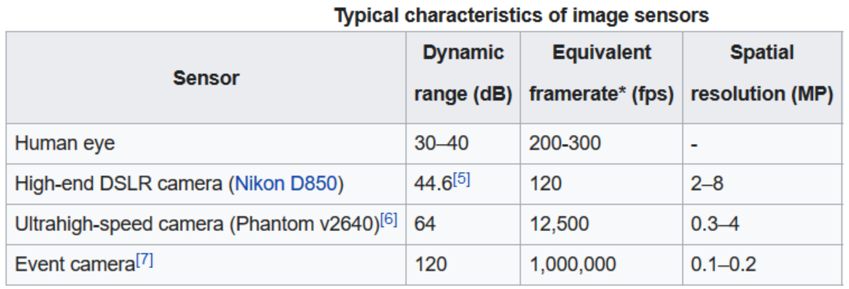
DASHCAM VIEW
VIPER VIEW
VIPER

Source: FLIR

Event based camera

- Our eyes function completely different than today’s cameras
- A camera-based system works taking and **processing multiple snapshots of the environment** (**you can loose information between frames**)
- Event based vision uses pixels to capture only relevant information from a dynamic scene reporting changes in brightness as they occur, and staying silent otherwise
- Efficient in terms of power and computing resources needed





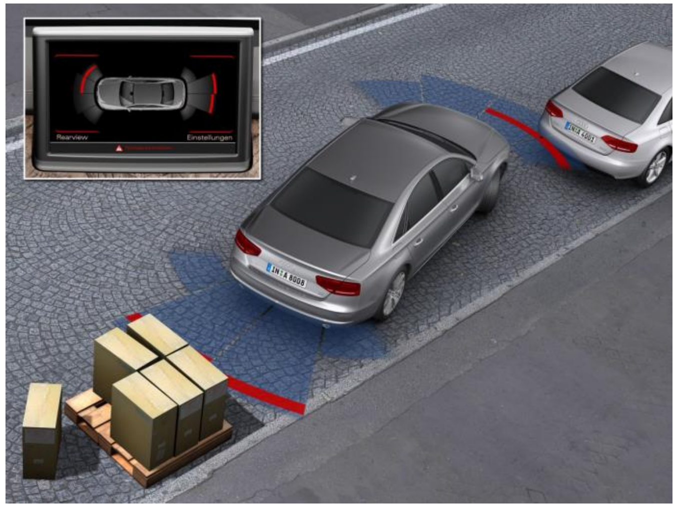
Typical characteristics of image sensors

Sensor	Dynamic range (dB)	Equivalent framerate* (fps)	Spatial resolution (MP)
Human eye	30–40	200-300	-
High-end DSLR camera (Nikon D850)	44.6[5]	120	2–8
Ultrahigh-speed camera (Phantom v2640)[6]	64	12,500	0.3–4
Event camera[7]	120	1,000,000	0.1–0.2

Source: Prophesee & Chronocam & Wikipedia

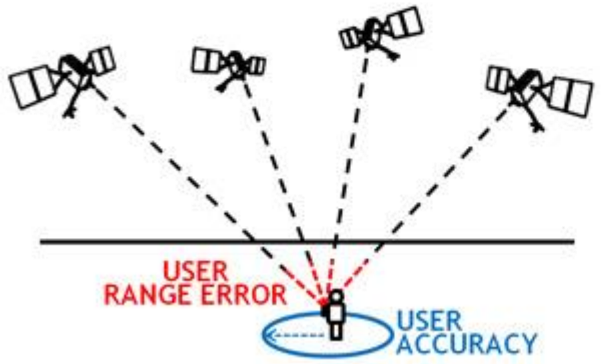
Ultrasonic sensor

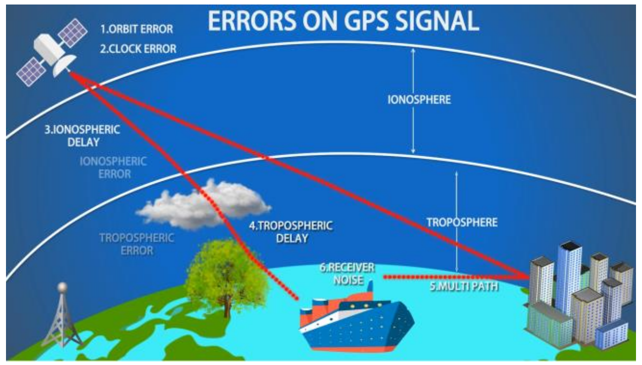
- **Ultrasonic sensor** is a sorround sensor that measures the distance to obstacles using ultrasonic sound waves
  - Distance compute based on the time the sound reflect back
  - Low cost
  - **Pro**: Unnafected by lighting and precipitation
  - **Cons**: unreliable if the material absorbs sound
  - Good solution for parking scenarios
- Metrics
  - Range
  - Field of view
  - Cost



GPS

- The Global Positioning System (GPS) is a satellite-based radio navigation system
  - To provide a position, the GPS satellites broadcast two pieces of information: their **location** and **time**
  - The distance is computed by calculating the difference of time between the signal being sent and received (**distance=speed*time**)
- One satellite is not enough to estimate the distance, to solve this, it is used **trilateration**
- GPS based solutions has many sources of errors
- Different variants of GPS: **RTK**, **PPP**, **DGPS**

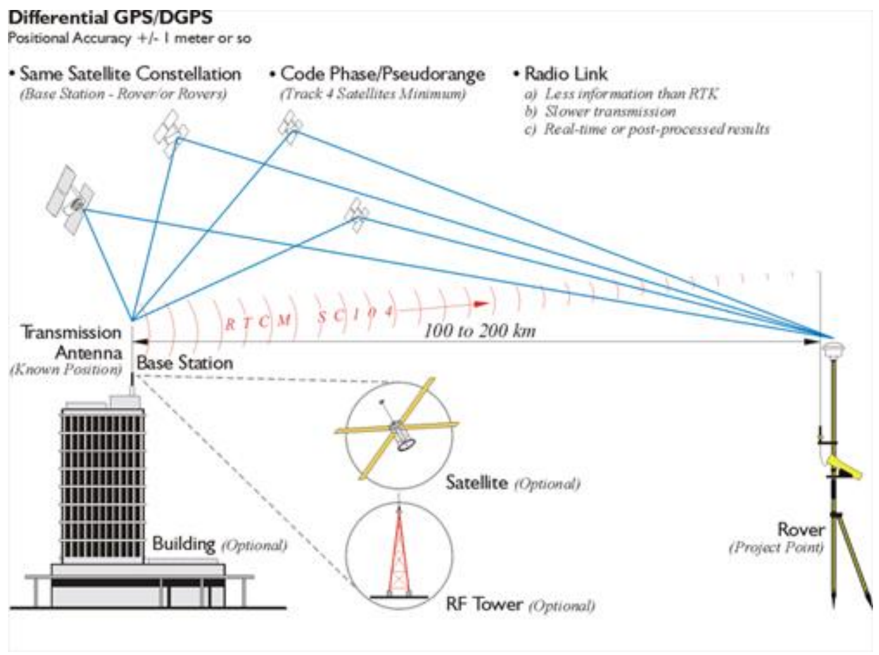




Source: Wikipedia

DGPS

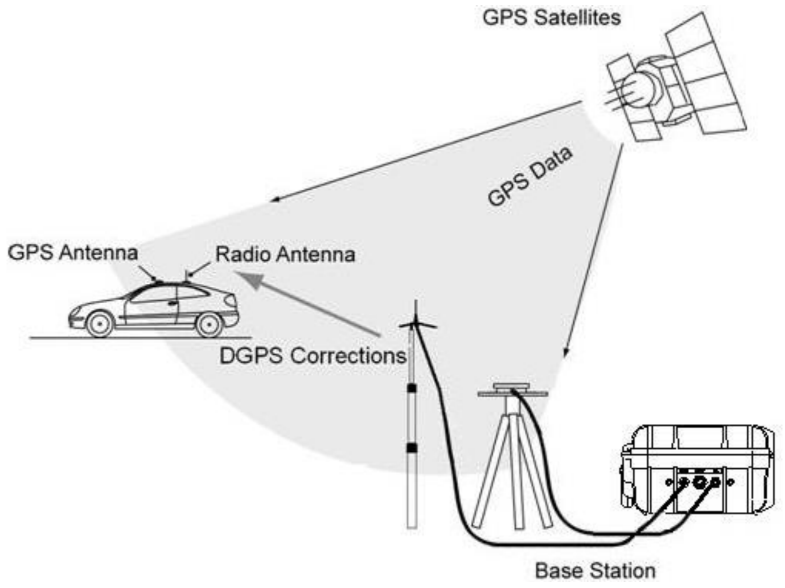
- A Differential Global Positioning System (DGPS) is an enhancement GPS which provides improved location accuracy
- Each DGPS uses a network of fixed ground-based reference stations **to broadcast the difference between the positions indicated by the GPS satellite system and known fixed positions**
  - **This enables the rover to apply the error to the signal and determine an accurate position**
  - The position accuracy depends on the distance between the base station and the rover
- From the 15-meter nominal GPS accuracy to about 1–3 cm in case of the best implementations


Differential GPS/DGPS
Positional Accuracy +/- 1 meter or so
• Same Satellite Constellation
(Base Station - Rover/or Rovers)
• Code Phase/Pseudorange
(Track 4 Satellites Minimum)
• Radio Link
a) Less information than RTK
b) Slower transmission
c) Real-time or post-processed results
RTCM SC104
100 to 200 km
Transmission Antenna (Known Position)
Base Station
Building (Optional)
Satellite (Optional)
RF Tower (Optional)
Rover (Project Point)

Source: Wikipedia

GPS RTK

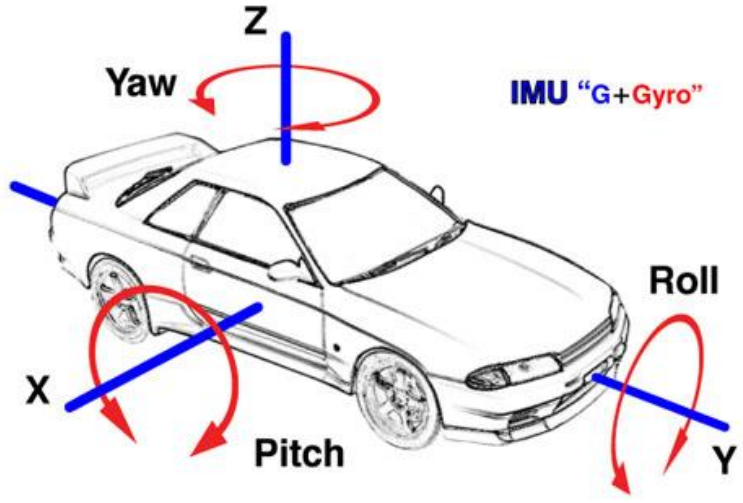
› Real Time Kinematic (RTK) is a satellite navigation technique used to enhance the precision of GPS signals
  - It uses a fixed base station (**which is generally portable**) which sends out corrections to a moving receiver
  - The moving object also receives the position from the satellite, this position is compared with the one received in the base station
  - By utilizing these corrections, the GPS engine can fix the position of the antenna to within 1 - 2cm
  - **It is mostly used to test and verify ADAS performance localization**



Source: Wikipedia

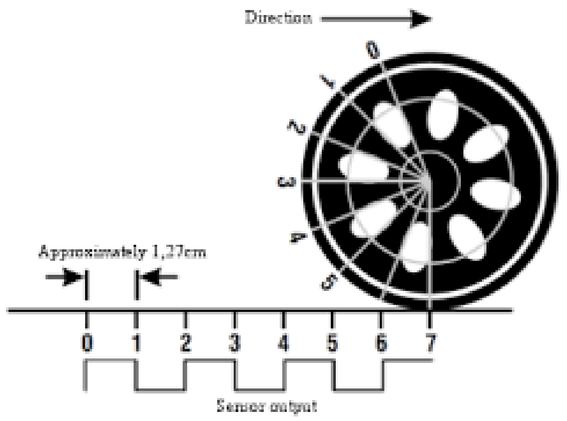
IMU

- The inertial measurement unit (IMU) measures and reports the **acceleration, angular rate, and the orientation of the ego vehicle**, using a combination of accelerometers, gyroscopes, and magnetometers
  - An IMU allows a GPS receiver to work when GPS-signals are unavailable, such as in tunnels, inside buildings, or when electronic interference is present (through sensor fusion)



Source: Wikipedia

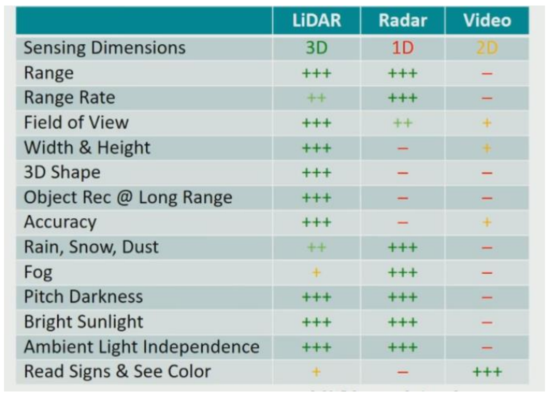
Wheel odometry



- **Tracks wheel velocities and orientation**
- Uses these to calculate overall speed and orientation of the car
  - Speed accuracy
  - Position drift
- Generally wheel odometry, GPS and IMU are used to correct the localization of the vehicle
  - Odometry installs wheel encoder to track the wheel movement and estimate the vehicle location
  - IMU uses accelerators and gyroscopes

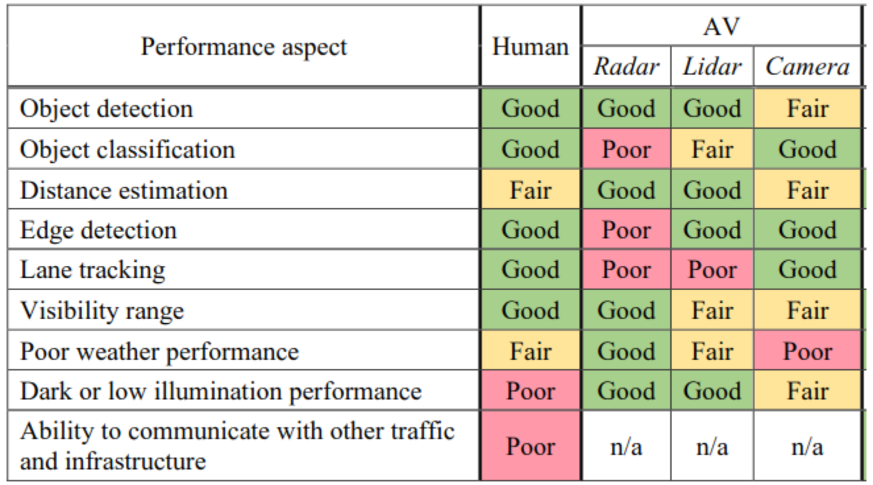
Comparison (1)

- **No single sensor can provide accurate data information for all weather conditions**
  - For instance, the performance of the cameras is poor when the weather conditions are not favorable (night or rain)
  - Unlike camera, radar works well in poor weather conditions, however, the resolution of radar is not good enough to perform object classification



	LiDAR	Radar	Video
Sensing Dimensions	3D	1D	2D
Range	+++	+++	–
Range Rate	++	+++	–
Field of View	+++	++	+
Width & Height	+++	–	+
3D Shape	+++	–	–
Object Rec @ Long Range	+++	–	–
Accuracy	+++	–	+
Rain, Snow, Dust	++	+++	–
Fog	+	+++	–
Pitch Darkness	+++	+++	–
Bright Sunlight	+++	+++	–
Ambient Light Independence	+++	+++	–
Read Signs & See Color	+	–	+++

Source: Quanergy

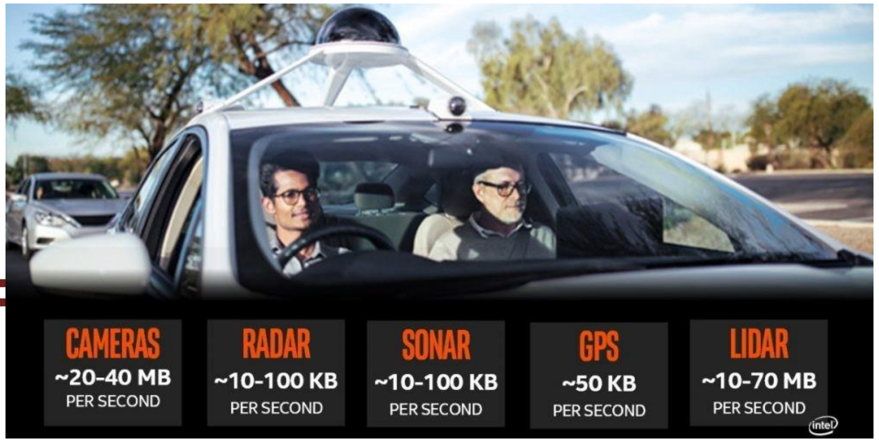


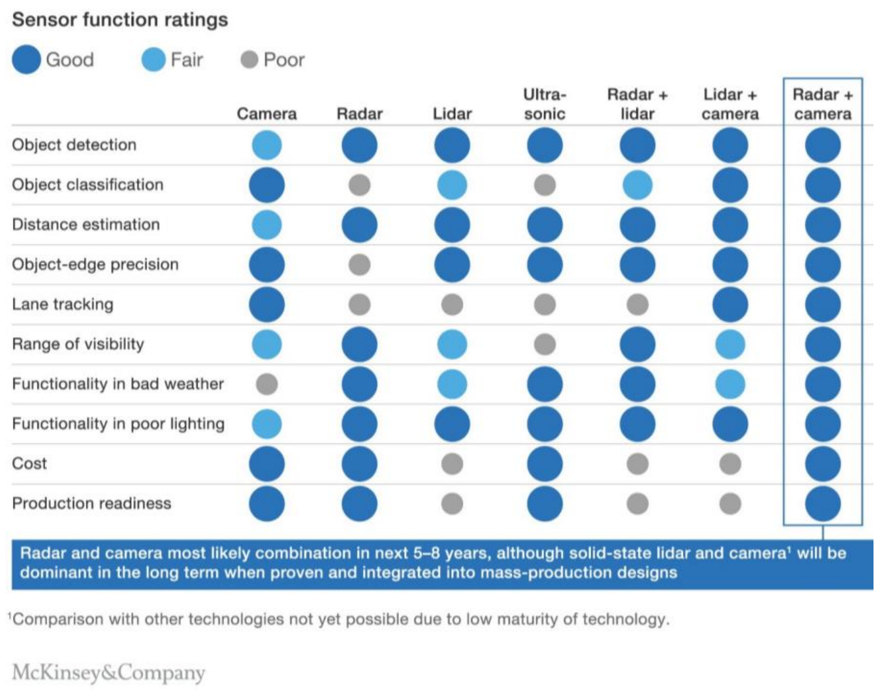
Performance aspect	Human	AV		
		Radar	*Lidar*	*Camera*
Object detection	Good	Good	Good	Fair
Object classification	Good	Poor	Fair	Good
Distance estimation	Fair	Good	Good	Fair
Edge detection	Good	Poor	Good	Good
Lane tracking	Good	Poor	Poor	Good
Visibility range	Good	Good	Fair	Fair
Poor weather performance	Fair	Good	Fair	Poor
Dark or low illumination performance	Poor	Good	Good	Fair
Ability to communicate with other traffic and infrastructure	Poor	n/a	n/a	n/a

Source: Schoettle, Brandon. Sensor Fusion: A Comparison of Sensing Capabilities of Human Drivers and Highly Automated Vehicles

Comparison (2)

- Each sensor **generates a different** amount of RAW data
- **No single sensor can provide accurate** data information for all weather conditions
- AV software stack is normally integrated **considering different sensors** which allows to build a better representation of the environment





Sensor function ratings

● Good ◍ Fair • Poor

	Camera	Radar	Lidar	Ultra-sonic	Radar + lidar	Lidar + camera	Radar + camera
Object detection	◍	●	●	●	●	●	●
Object classification	●	•	◍	•	◍	●	●
Distance estimation	◍	●	●	●	●	●	●
Object-edge precision	●	•	●	●	●	●	●
Lane tracking	●	•	•	•	•	●	●
Range of visibility	◍	●	◍	•	●	◍	●
Functionality in bad weather	•	●	◍	●	●	◍	●
Functionality in poor lighting	◍	●	●	●	●	●	●
Cost	●	●	•	●	•	•	●
Production readiness	●	●	•	●	•	•	●

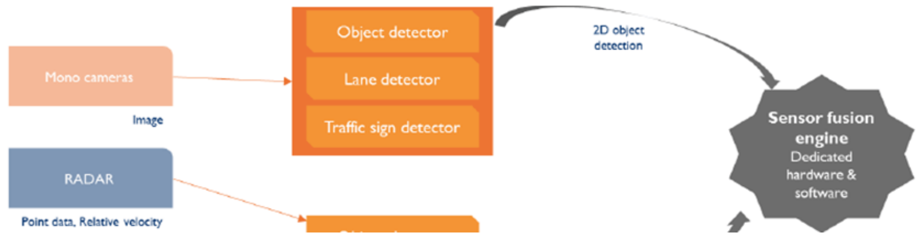
Radar and camera most likely combination in next 5–8 years, although solid-state lidar and camera[1] will be dominant in the long term when proven and integrated into mass-production designs

[1]Comparison with other technologies not yet possible due to low maturity of technology.

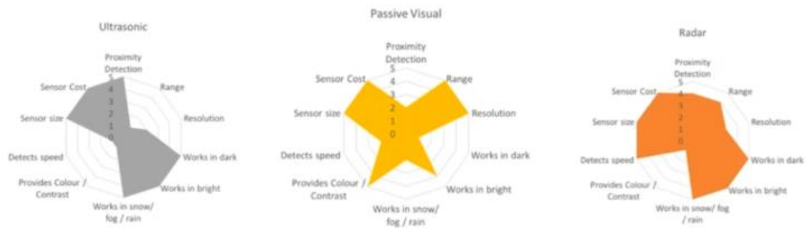
McKinsey&Company

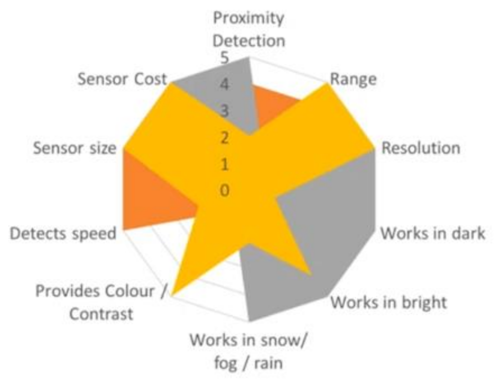
Sensor fusion ("Unity makes strength")

› To overcome the limitations of the sensors, sensor fusion can be applied
  - **Sensor fusion provides redundancy** for self-driving vehicles
  - For instance, **LiDAR and Radar data can be fused to obtain an improved 3D object detection** independently of the weather conditions
  - **We will use sensor fusion (radar+lidar) to improve the localization system of the car**



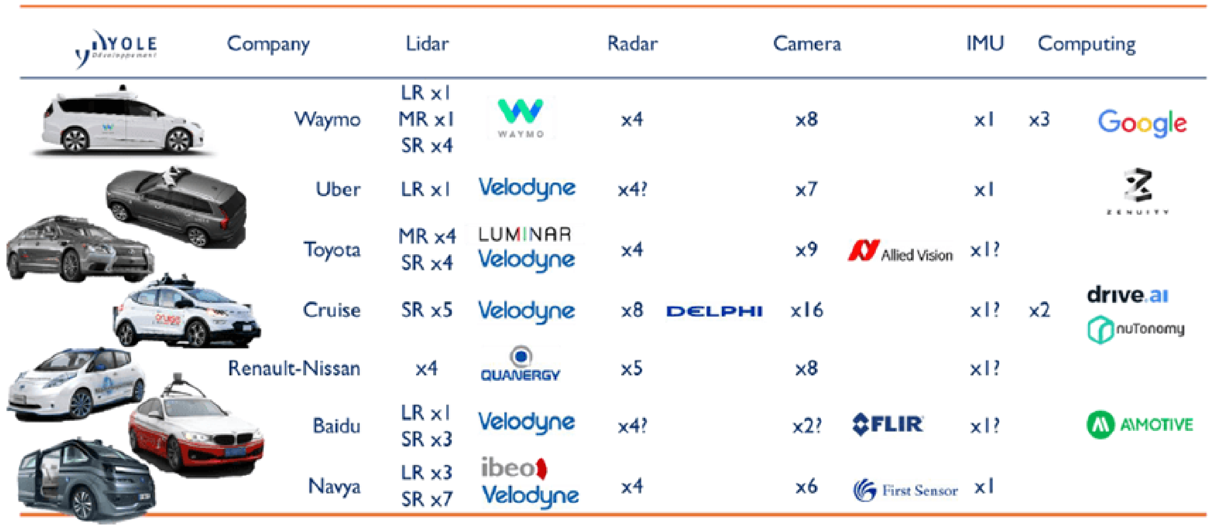
Source: Autonomous Vehicle Sensors Conference, June 26 2018, San Jose





State of the art

- Almost all automotive players are designing the future L2 to L5 features relying on LiDARs
  - Mostly thanks to the lidar's ability to perceive in poor weather and light conditions (better than all commercial cameras)
- Elon Musk (Tesla) states that: *"Anyone relying on lidar is doomed."*
  - *"Whatever a normal human can navigate, so can a camera-based system"*



YOLE Développement	Company	Lidar		Radar		Camera		IMU	Computing	
	Waymo	LR x1 MR x1 SR x4	WAYMO	x4		x8		x1	x3	Google
	Uber	LR x1	Velodyne	x4?		x7		x1		ZENUITY
	Toyota	MR x4 SR x4	LUMINAR Velodyne	x4		x9	Allied Vision	x1?		
	Cruise	SR x5	Velodyne	x8	DELPHI	x16		x1?	x2	drive.ai nuTonomy
	Renault-Nissan	x4	QUANERGY	x5		x8		x1?		
	Baidu	LR x1 SR x3	Velodyne	x4?		x2?	FLIR	x1?		AIMOTIVE
	Navya	LR x3 SR x7	ibeo Velodyne	x4		x6	First Sensor	x1		

Case study

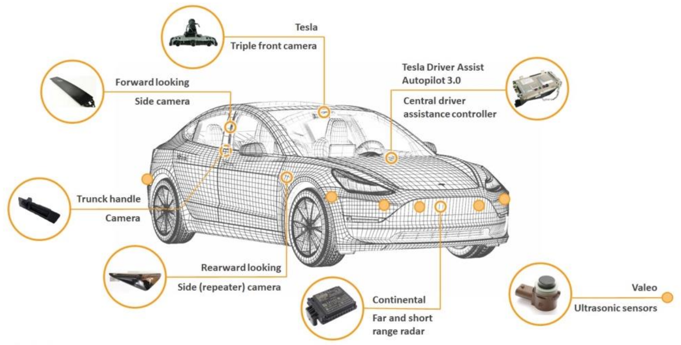
Tesla

Waymo

Case study – Tesla Model 3

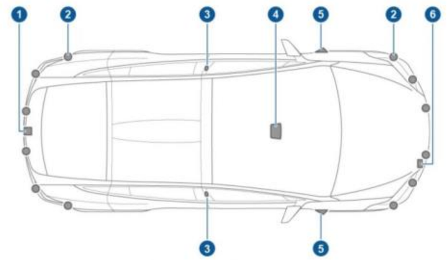
› **Autopilot**
- **Traffic-Aware Cruise Control**: Matches the speed of your car to that of the surrounding traffic
- **Autosteer**: Assists in steering within a clearly marked lane, and uses traffic-aware cruise control

› **Full Self-Driving Capability**
- **Navigate on Autopilot (Beta)**: Actively guides your car from a highway's on-ramp to off-ramp, including suggesting lane changes, navigating interchanges, automatically engaging the turn signal and taking the correct exit
- **Auto Lane Change**: Assists in moving to an adjacent lane on the highway when Autosteer is engaged
- **Autopark**: Helps automatically parallel or perpendicular park your car, with a single touch
- **Summon**: Moves your car in and out of a tight space using the mobile app or key
- **Smart Summon**: Your car will navigate more complex environments and parking spaces, maneuvering around objects as necessary to come find you in a parking lot.
- **Traffic and Stop Sign Control (Beta)**: Identifies stop signs and traffic lights and automatically slows your car to a stop on approach, with your active supervision



https://www.tesla.com/support/autopilot

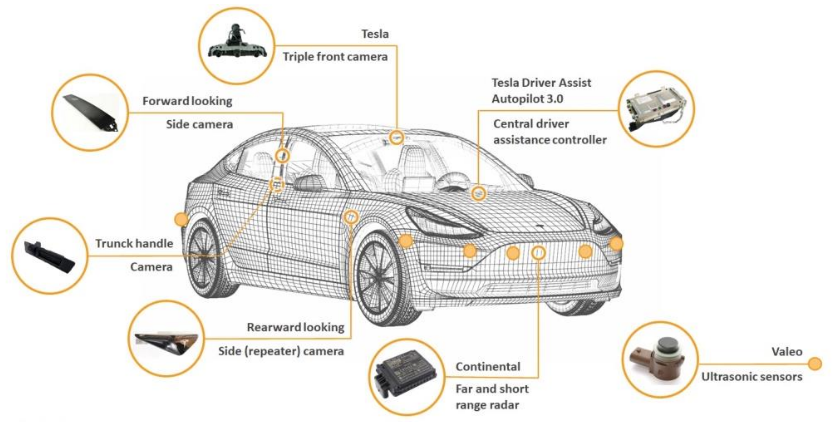
Case study – Tesla Model 3

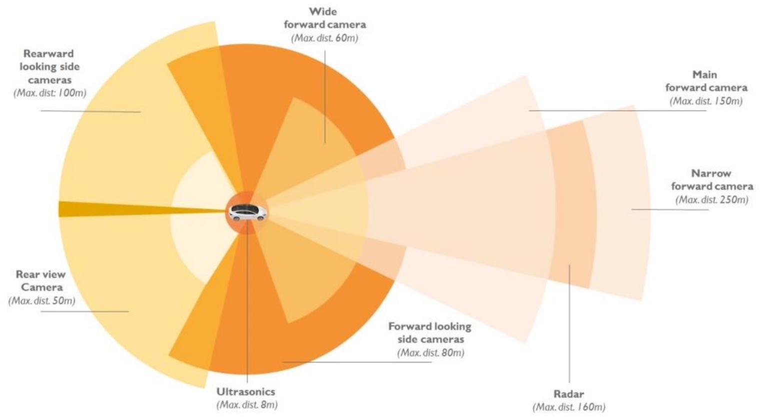


1. A camera is mounted above the rear license plate.
2. Ultrasonic sensors are located in the front and rear bumpers.
3. A camera is mounted in each door pillar.
4. Three cameras are mounted to the windshield above the rear view mirror.
5. A camera is mounted to each front fender.
6. Radar is mounted behind the front bumper on the right side of the vehicle.

Model X is also equipped with high precision electrically-assisted braking and steering systems.

- Model 3 is a smaller and more affordable EV, first produced in mid-2017
  - **Eght cameras** which provide 360-degree visibility around the car within a radius of 250 meters; 12 ultrasonic sensors
  - **Four cameras facing the front**. The main one, covers 250 meters but with a very narrow-angle of view, and there are others that cover shorter distances. The other four cameras face the sides and rear of the car and can see up to 100 meters away
  - **Sonar is used to detect obstacles** within a radius of 8 meters around the car.
  - **GPS is used to detect the position of the car** concerning the road
  - The Continental **ARS4-A radar system is used for forwarding collision warning, emergency brake assist, collision mitigation or adaptive cruise control (ACC)**

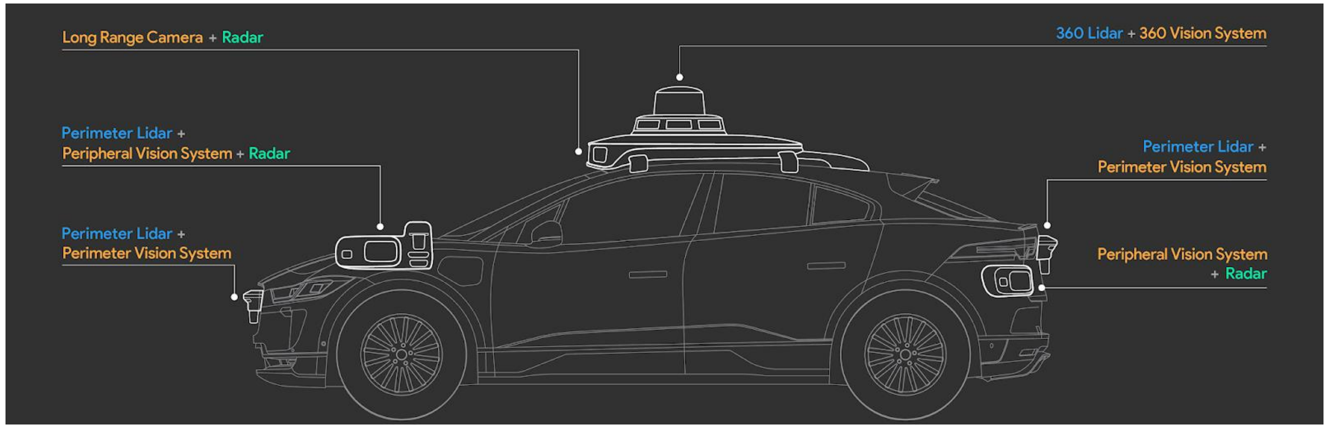




Source: System Plus Consulting & EETimes

Case study – Waymo 5th generation

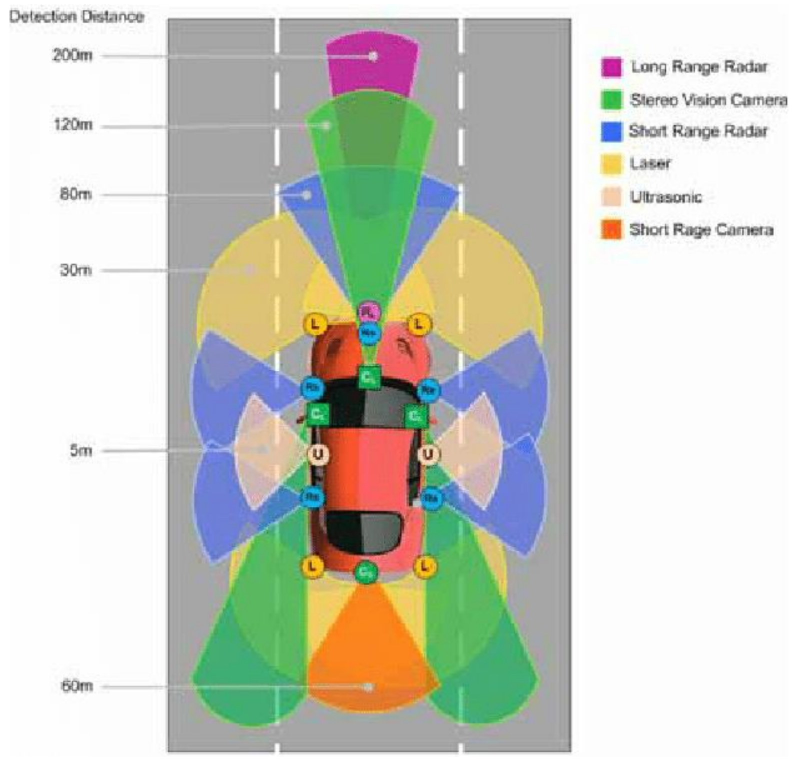
- The "5th-generation Waymo Driver sensor suite" has been tested in the Jaguar's electric I-Paces
  - All the hardware is built in house
  - A perimeter vision system works in conjunction with perimeter LiDAR for up-close perspectives
  - **LiDARs detect obstacles directly in front of the vehicle**
  - The perimeter cameras provide the machine learning algorithms additional details to identify objects
  - There's also a peripheral vision system to reduce "blind spots caused by parked cars or large vehicles."
  - Waymo's car uses radar to "instantaneously see and measure an object's velocity."


Long Range Camera + Radar
360 Lidar + 360 Vision System
Perimeter Lidar + Peripheral Vision System + Radar
Perimeter Lidar + Perimeter Vision System
Perimeter Lidar + Perimeter Vision System
Perimeter Lidar + Perimeter Vision System
Peripheral Vision System + Radar

Source: Waymo

Sensor configuration design

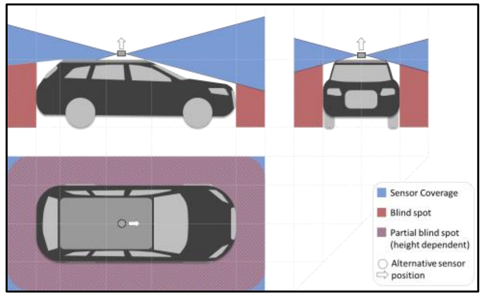
- **Sensor positioning is important**
- **Operational Design Domain (ODD):** “operating conditions under which a given driving automation system or feature thereof is specifically designed to function”
- The ODD is something to consider when designing the sensor setup
  - For **lane keeping** in a highway **one radar in front of the car should be sufficient**
  - For AD in **urban environments**, 360° horizontal coverage across all three major sensor modalities (**camera, lidar and radar**) is required in order to reliably detect traffic

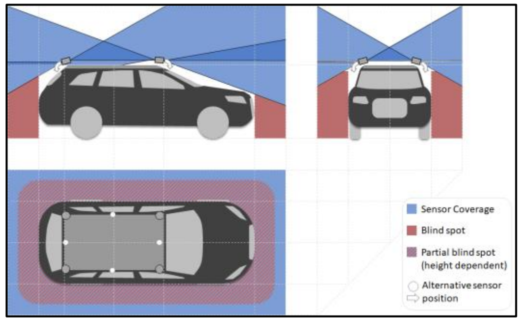


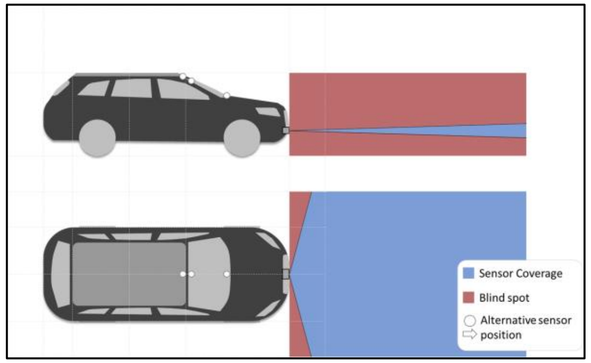
Source: https://autonomous-driving.org

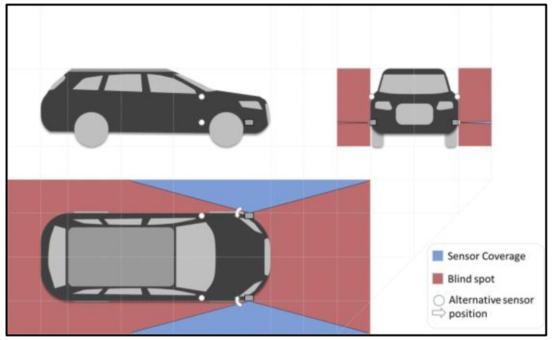
LiDAR

- **LiDAR on roof center**
  - Simple setup, no effort to synchronize and align multiple point clouds, 360° coverage with one sensor. But multiple blind spots (**localization/detection**)
- **Multiple LiDARs (tilted) on the roof**
  - Tilted sensors reach optimal coverage among roof mounted configurations. Higher complexity in terms of integration and point cloud fusion compared to single sensor (**mapping**)
- **Front LiDAR**
  - Good and simple solution to detect vehicles in front. It doesn't work well in slopes (**detection**)
- **Side view LiDAR**
  - Allows full coverage of vehicle's sides. Again, it doesn't work well in slopes (**detection**)


Sensor Coverage
Blind spot
Partial blind spot (height dependent)
Alternative sensor position

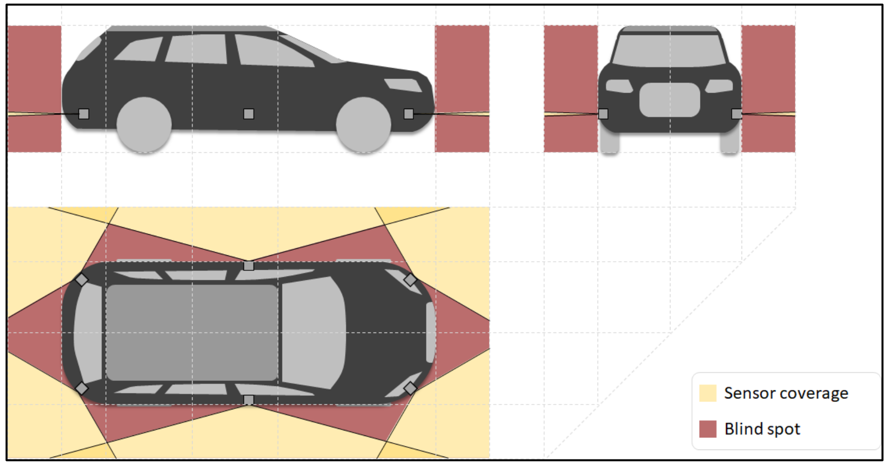

Sensor Coverage
Blind spot
Partial blind spot (height dependent)
Alternative sensor position

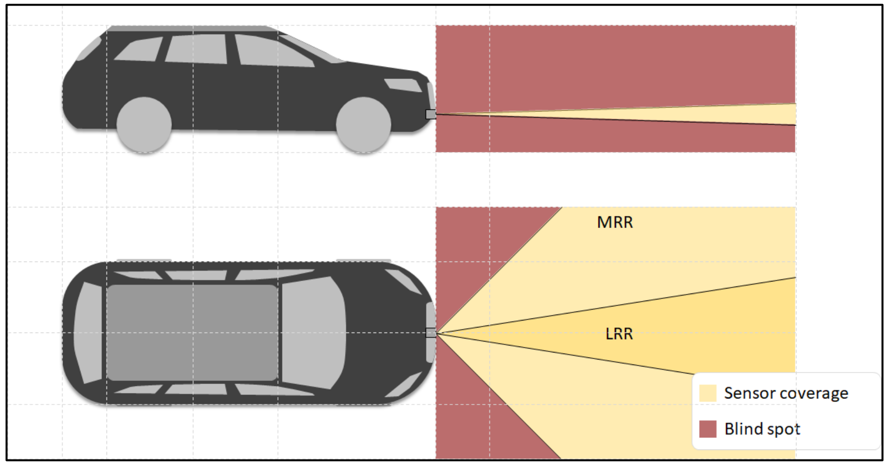




Radar

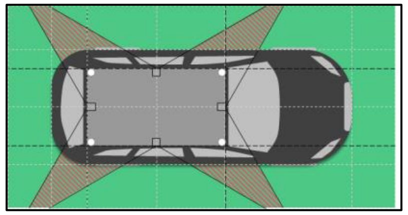
- **360° coverage by short- and mid-range radars**
  - Allows a good coverage of the environment. But the vertical FOV pose different challenges when facing non flat terrain
- **Front radar**
  - Different ranges; mid range (MRR, ~100m range) and long-range radars (LRR, ~250m range) enable ADAS features like **Adaptive Cruise Control (ACC) and Autonomous Emergency Breaking (AEB)**
    - **However the sensor is not sufficient for urban AD**

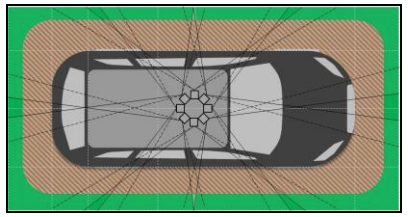


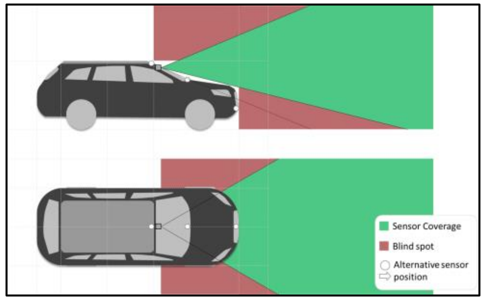


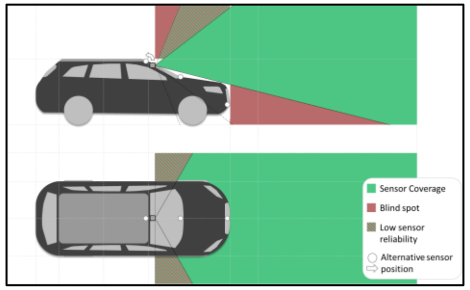
Camera

- **360º coverage by roof-mounted wide-angle cameras**
  - With this configuration it is possible to build a bird view of the environment, since it provides a 360º coverage
- **Central camera tower**
  - Mapping (feature extraction)
- **Front camera**
  - Mounted in the windshield enables ADAS features like departure warning or lane change assist. The vertical FOV is limited by the vehicle's hood
- **Wide angle camera for traffic light detection**
  - Alternative configuration mounted in the roof to detect traffic lights



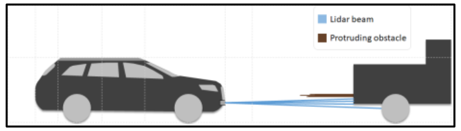


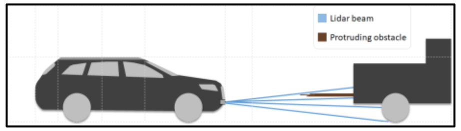


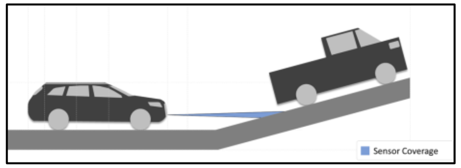


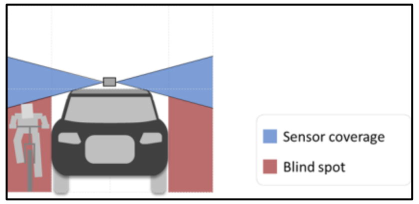
Classic problems

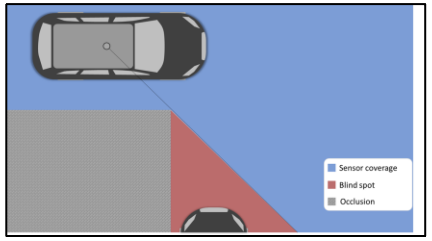
- **Mid-air and ceiling obstacles**
  - open truck doors, protruding truck freight, tree branches, boom gates or low bridges
- **Slope**
  - A ramp in front of the vehicle can be interpreted as an obstacle
- **Cyclists**
- **Crossing with occlusion**
  - A sensor facing sideways could help

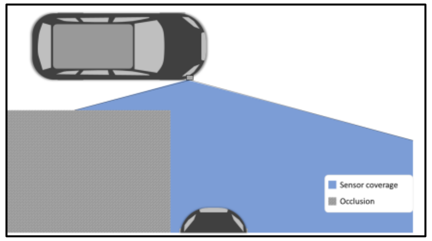










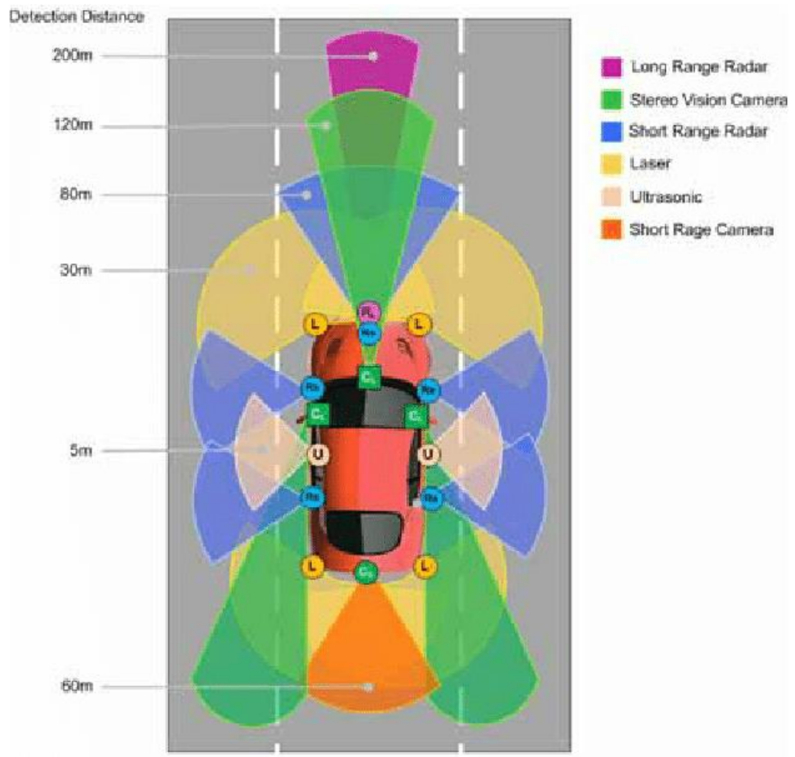


Sensor configuration design

› Propose a sensor configuration design for an autonomous vehicle that has to drive in two different scenarios
  – **Parking assistance system**
  – **Lane changing assistance system**

› Recall that we have different sensors which in turn have different metric and possible configurations

› **Objective:** design a low cost solution

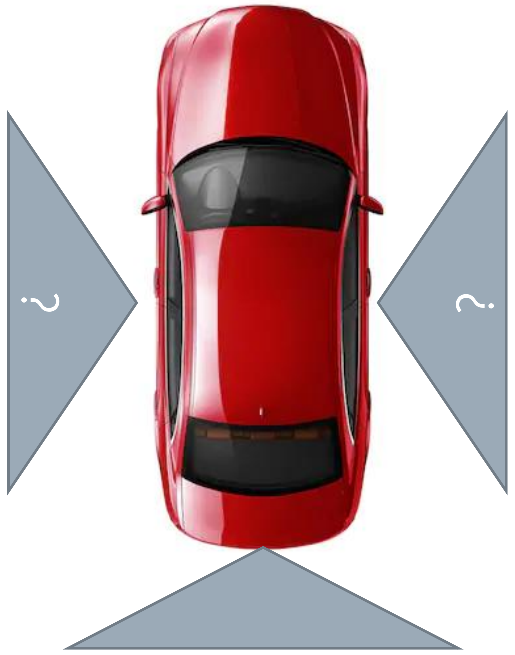


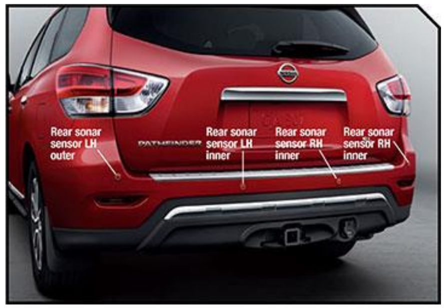
Sensor configuration design (Parking scenario)

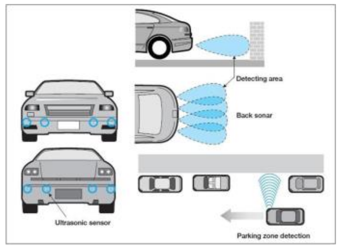
- Let’s start by defining the Operational Design Domain
  - Traffic speed:
    - **Low**
  - Traffic volume:
    - **Low**
  - Any other considerations?

Sensor configuration design (Parking scenario)

- Field of view:
  - **wide**
- Distance:
  - **short range**
- Which sensor could we use?
  - **Sonar could be a good option**
  - **Cameras**?
- In which part of the vehicle should we put the sonar?

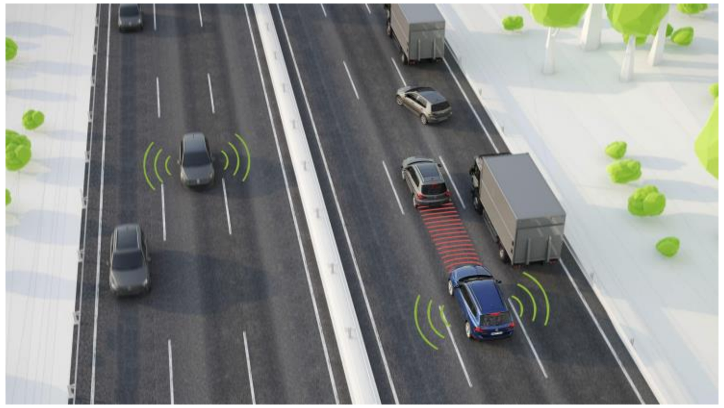



Rear sonar sensor LH outer
PATHFINDER
Rear sonar sensor LH inner
Rear sonar sensor RH inner
Rear sonar sensor RH inner



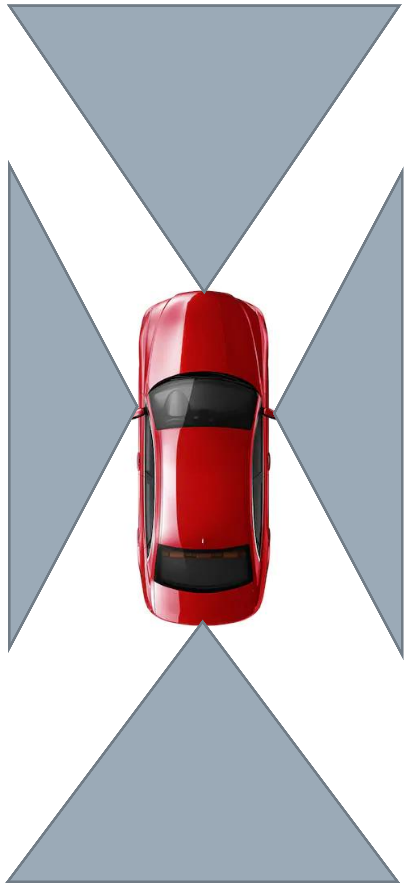
Sensor configuration design (Highway)

- Let’s start by defining the Operational Design Domain
  - Traffic speed:
    - **High**
  - Traffic volume:
    - **Low & High**
  - Any other considerations?



Sensor configuration design (Highway)

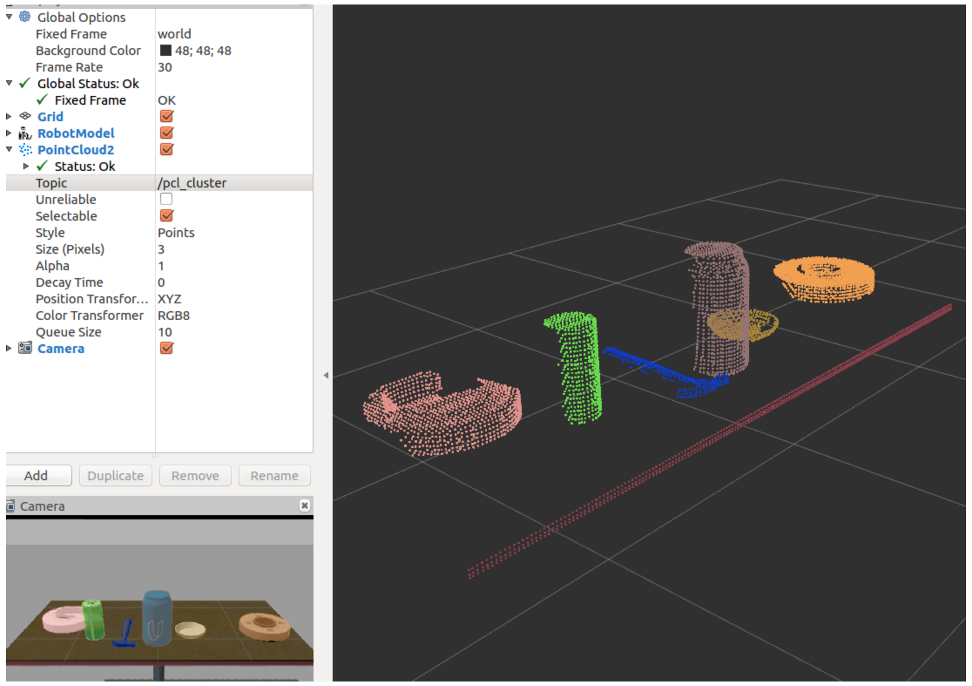
- We could configure a **radar sensor** on each side of the rear of the car to detect road users
- **Front and rear: long range view**
  - Security distance is needed before performing a lane change
- **Sides: short range but wide FOV** (it depends on the road)
  - We need to check not only adjacent lanes
- **We could also use cameras, but they are prone to errors** when detecting objects and they are generally expensive in terms of computational power needed
- **Ideally, we should use cameras and radars fused together to achieve the maximum safety performance**



Source: Valeo

Point cloud exercise

- ***Objective***: Find and segment the individual object point clusters lying on the plane (**Euclidean clustering**)
- A clustering method is used to organize an unorganized point cloud into smaller parts
  - identify which points in a point cloud belong to the same object
  1. Down sample the point cloud
  2. Segment the ground plane
  3. Create a KD-tree representation for the input point cloud dataset
  4. Compute Euclidean distance to build the cluster list
  5. the algorithm terminates when all points have been processed and are now part of the list of point clusters



Global Options	
Fixed Frame	world
Background Color	■ 48; 48; 48
Frame Rate	30
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	✓
RobotModel	✓
PointCloud2	✓
✓ Status: Ok	
Topic	/pcl_cluster
Unreliable	☐
Selectable	✓
Style	Points
Size (Pixels)	3
Alpha	1
Decay Time	0
Position Transfor…	XYZ
Color Transformer	RGB8
Queue Size	10
Camera	✓

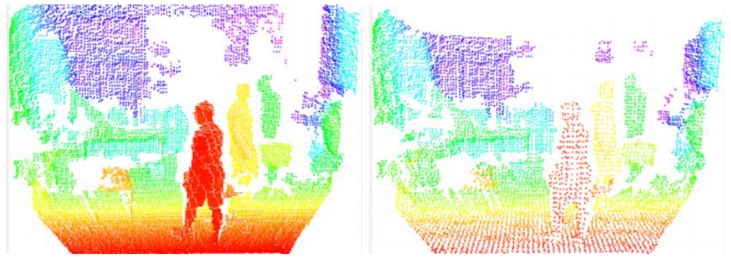
Source: https://pcl.readthedocs.io/projects/tutorials/

PCD data

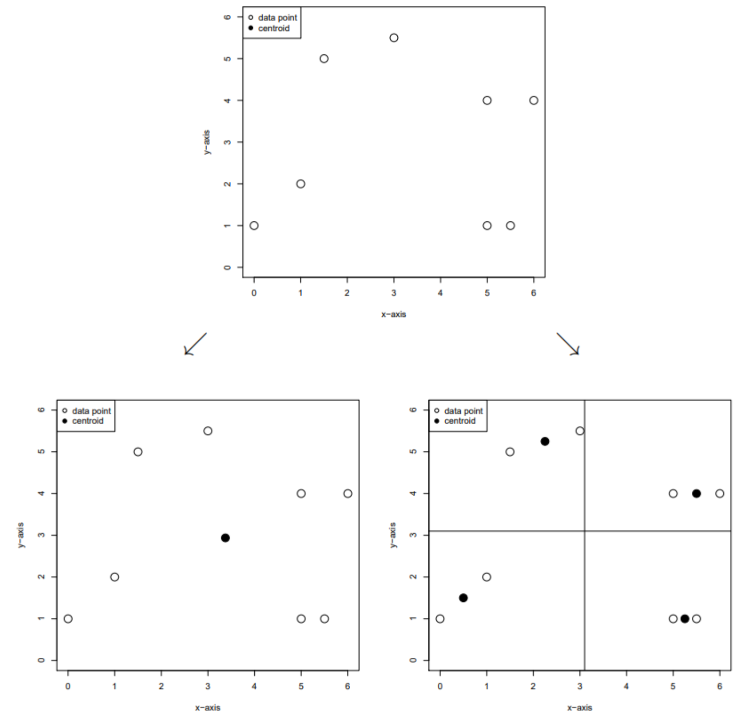
- PCD (Point Cloud Data) is a file format that supports 3D point cloud data
  - **FIELDS** - specifies the name of each dimension/field that a point can have
  - **SIZE** - specifies the size of each dimension in bytes
  - **POINTS** - specifies the total number of points in the cloud
  - **DATA** - specifies the data type that the point cloud data is stored in (ascii or binary)

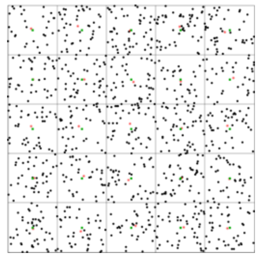
```
# .PCD v.7 - Point Cloud Data file format
VERSION .7
FIELDS x y z rgb
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 213
HEIGHT 1
VIEWPOINT 0 0 0 1 0 0 0
POINTS 213
DATA ascii
0.93773 0.33763 0 4.2108e+06
0.90805 0.35641 0 4.2108e+06
0.81915 0.32 0 4.2108e+06
0.97192 0.278 0 4.2108e+06
0.944 0.29474 0 4.2108e+06
0.98111 0.24247 0 4.2108e+06
```

Voxel filtering



- Point clouds store a lot of detailed information about physical space
  - Not all the information is required
- **Voxel filtering**: technique used to reduce the number of points (also called downsample)
  - The voxel grid filter down-samples the data by taking a spatial average of the points in the cloud (i.e., using the centroid)

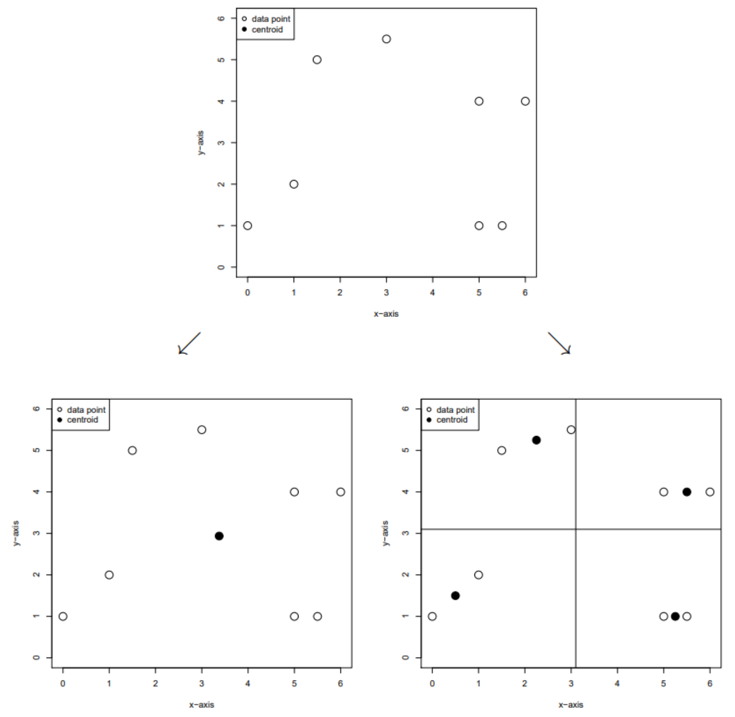




https://libpointmatcher.readthedocs.io/en/latest/DataPointsFilterDev/

Voxel filtering

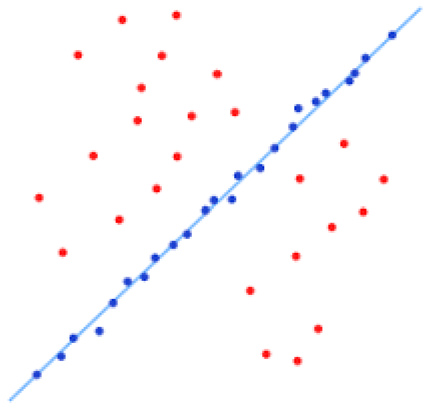
- Black dots are the centroids of each voxel
  - Left → big voxel size
  - Right → smaller voxel size
- A voxel grid filter down samples the data by averaging the points for each voxel grid in the point cloud,
- **setLeafSize(x,y,z) set the voxel grid leaf size**
  - **The bigger the voxel grid size the <u>more</u> the points filtered**



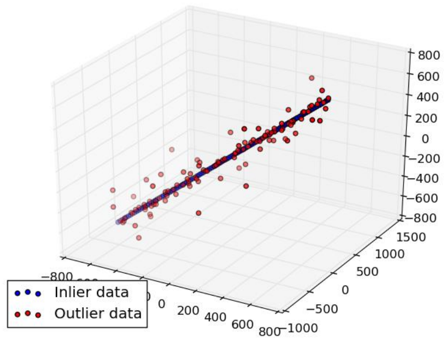
```
pcl::VoxelGrid<pcl::PCLPointCloud2> sor;
sor.setInputCloud (cloud);
sor.setLeafSize (0.01f, 0.01f, 0.01f);
sor.filter (*cloud_filtered);
```

RANSAC

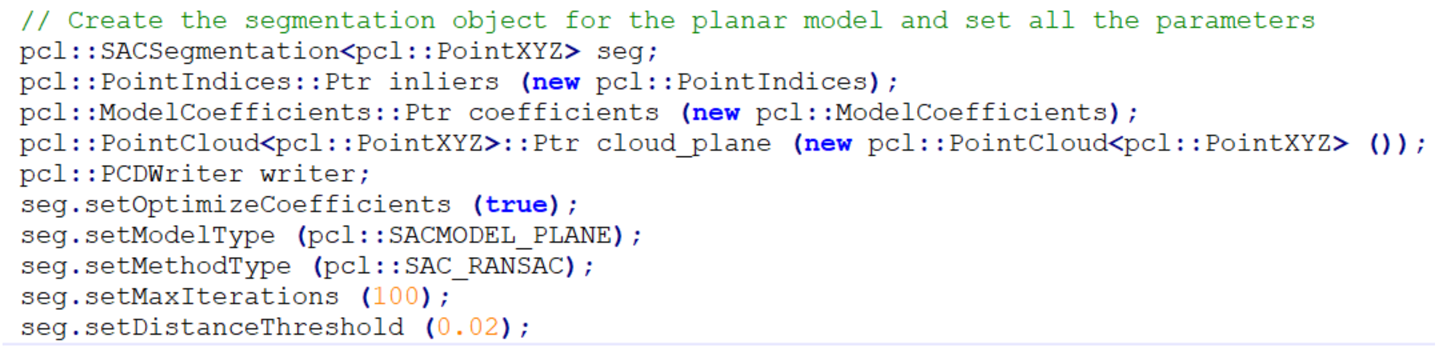
- **<u>We need to separate the ground from the other objects</u>**
- **Random sample consensus** (**RANSAC**) is an iterative method to estimate parameters of a mathematical model from a set of observed data that contains outliers
  - The algorithm groups the data into inliers and outliers
    - Inliers are the points that represent the table
    - Outliers are the points that represent the objects lying on the table
- 2D Ransac
  - Select two random points
  - Fit a line for those two points
  - Compute the points that are in/close to the line (score)
  - Repeat the process with other two random points
  - Choose the model which has the best "score"



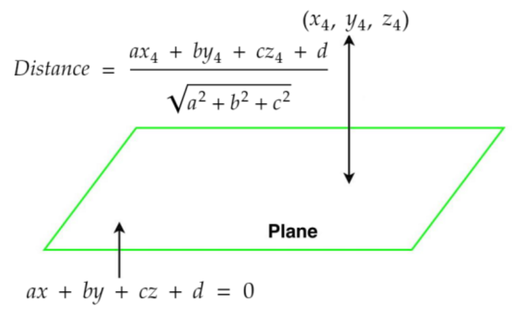
3D RANSAC



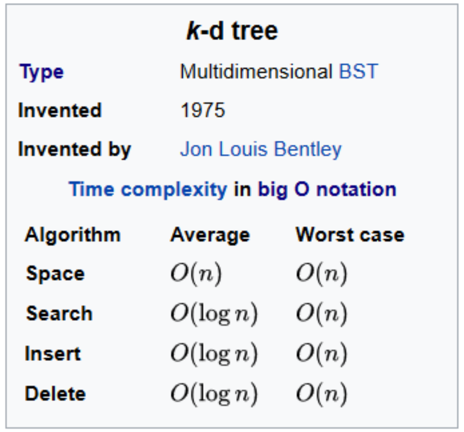
- Let us suppose that we want to get the objects (pedestrian, vehicles etc...) that are above the plane
- 3D RANSAC
  - Select three random points
  - Form a plane with these points (plane model)
  - Compute the number of points that fit into the plane model (we consider as **inliers** the points that lies close to the plane)
  - **Repeat** the process N times
  - Select the plane model that “collects” more points



```
// Create the segmentation object for the planar model and set all the parameters
pcl::SACSegmentation<pcl::PointXYZ> seg;
pcl::PointIndices::Ptr inliers (new pcl::PointIndices);
pcl::ModelCoefficients::Ptr coefficients (new pcl::ModelCoefficients);
pcl::PointCloud<pcl::PointXYZ>::Ptr cloud_plane (new pcl::PointCloud<pcl::PointXYZ> ());
pcl::PCDWriter writer;
seg.setOptimizeCoefficients (true);
seg.setModelType (pcl::SACMODEL_PLANE);
seg.setMethodType (pcl::SAC_RANSAC);
seg.setMaxIterations (100);
seg.setDistanceThreshold (0.02);
```

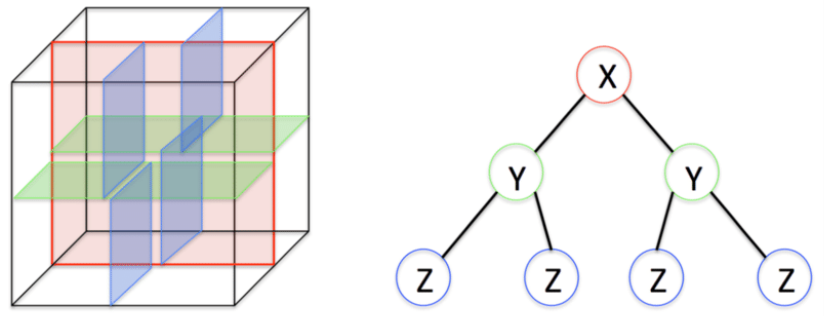

(x_4, y_4, z_4)
$Distance = \frac{ax_4 + by_4 + cz_4 + d}{\sqrt{a^2+b^2+c^2}}$
Plane
$ax + by + cz + d = 0$

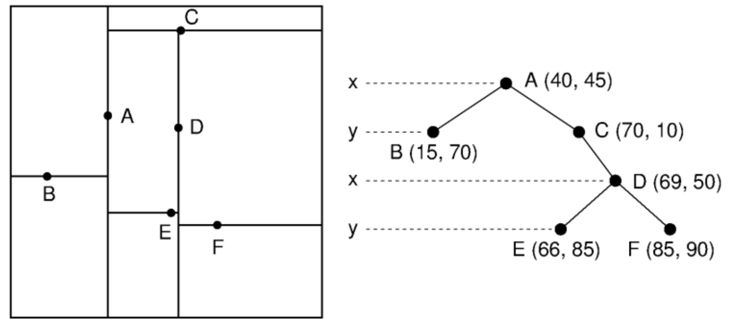
K-D Tree



***k*-d tree**		
Type	Multidimensional BST	
Invented	1975	
Invented by	Jon Louis Bentley	
Time complexity in big O notation		
Algorithm	Average	Worst case
Space	$O(n)$	$O(n)$
Search	$O(\log n)$	$O(n)$
Insert	$O(\log n)$	$O(n)$
Delete	$O(\log n)$	$O(n)$

- How to store the point cloud?
  - Arrays? → complex to compute nearest neighbor
- In computer science, a K-D tree (short for k-dimensional tree) is a space-partitioning data structure for organizing points in a k-dimensional space
  - **It is very useful to compute nearest neighbor points**





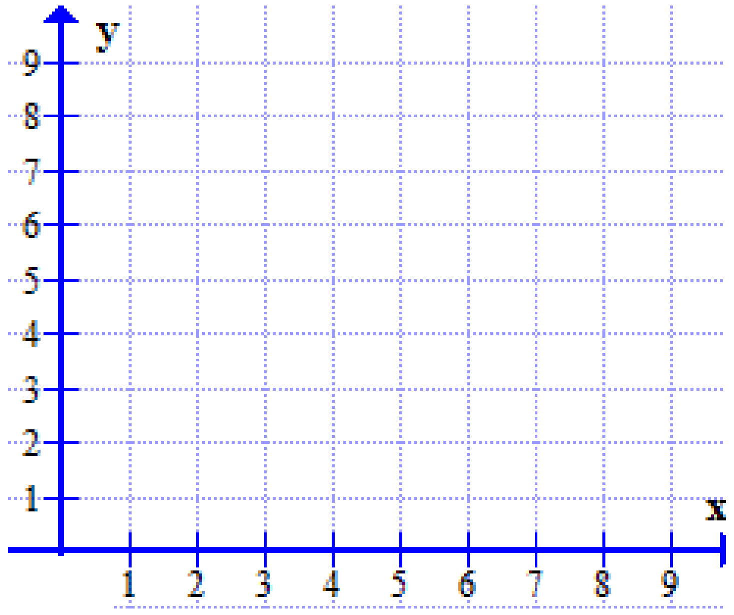
(a) (b)

X
y
X

K-D Tree search example

[2,3], [8,1], [4,5], [3,9], [1,4], [6,2], [7,7]

Build a K-D tree!



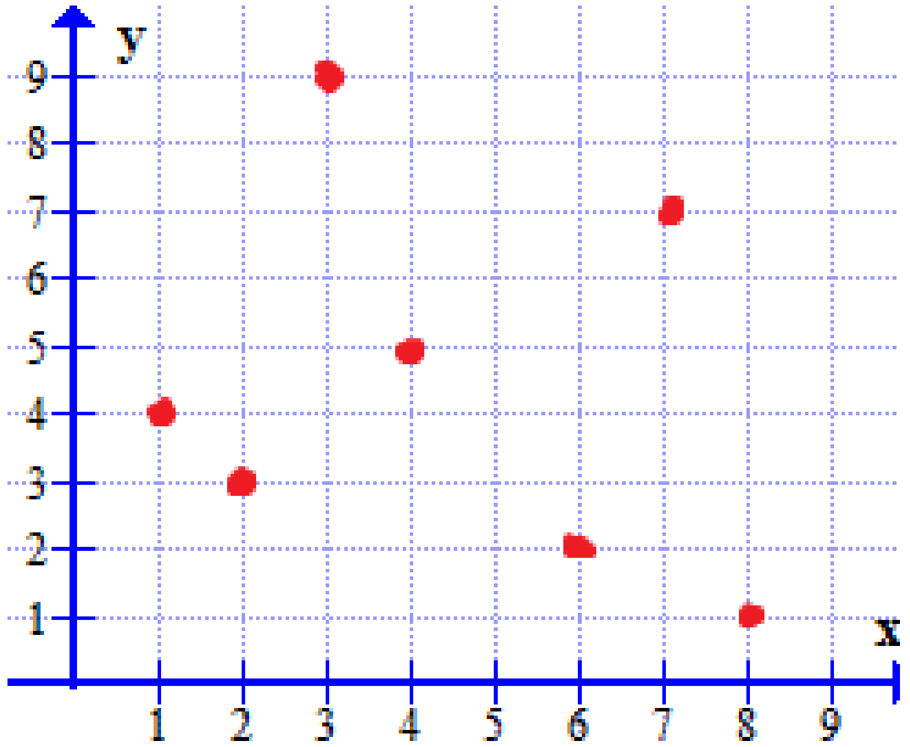
K-D Tree search example

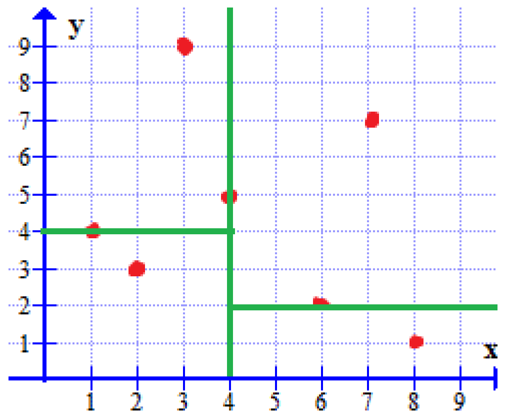
[2,3], [8,1], [4,5], [3,9], [1,4], [6,2], [7,7]

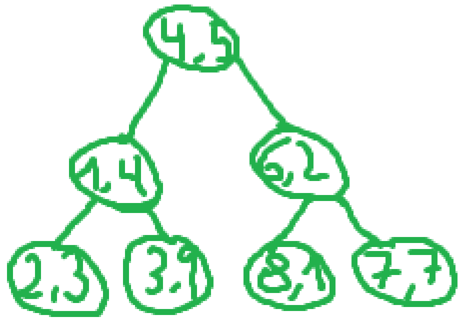
↓

[1,4], [2,3], [3,9], [4,5], [6,2], [7,7], [8,1]

1) Find the median in x and start partitioning the tree with the median as a root in the tree
2) Repeat iteratively using the median in y then in x and so on…



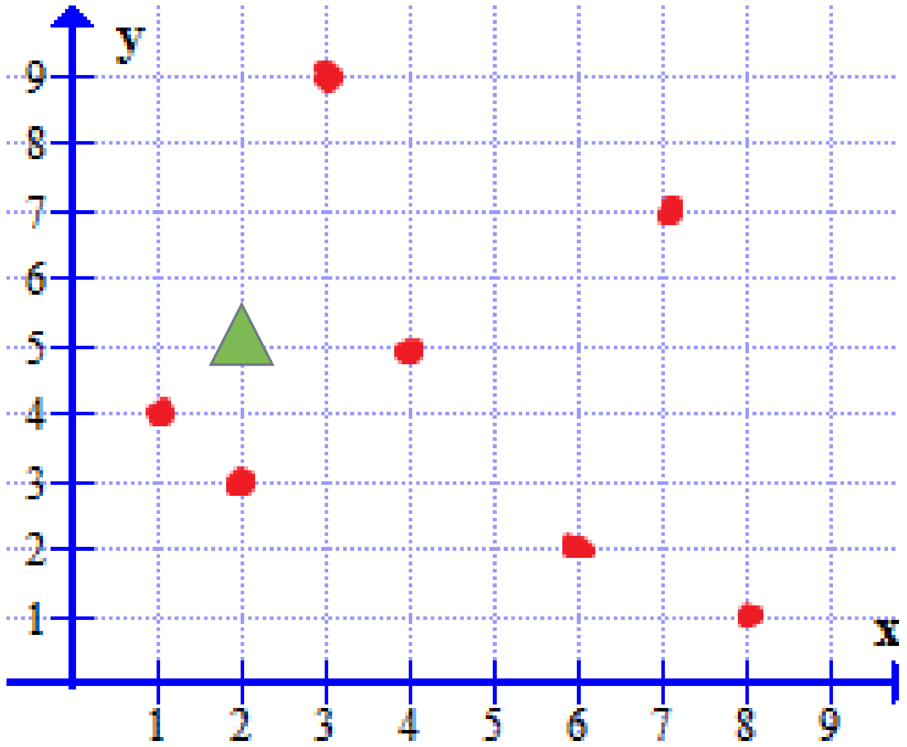




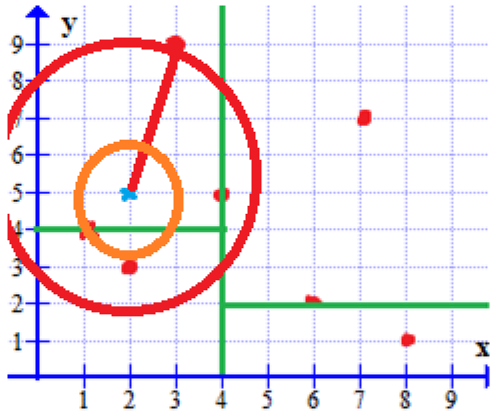
Example:
http://andrewd.ces.clemson.edu/courses/cpsc805/references/nearest_search.pdf

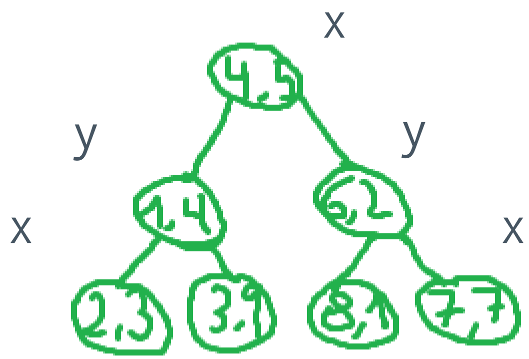
K-D Tree search example

Find the closest point to [2,5]

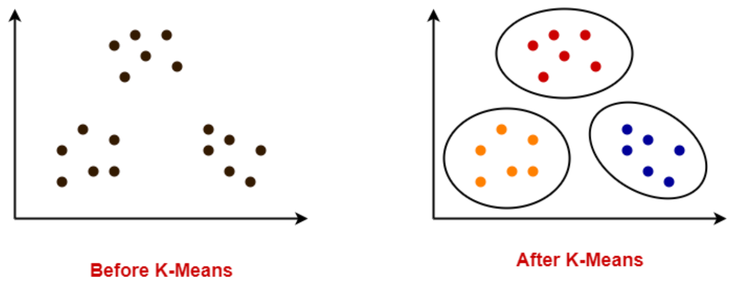


1) Find in the tree the closest point
2) Unwind the recursion and check if it is the closest point
    - it is possible that exploring the tree we do not find the closest point
    - We need to explore also the other branches if the sphere with the radius intersects with other regions
3) Repeat iteratively until finding the closest point





Clustering

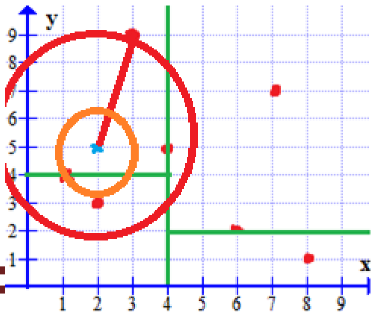


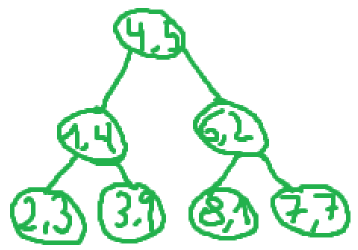
- We need a method to establish the points belong to a certain cluster
- **Euclidean clustering**: we use a K-D tree structure for finding the nearest neighbors
  - We use the Euclidean distance and **a search threshold**
    - Straight line distance between two points

$$searchpoint(x) + threshold > x > searchpoint(x) - threshold$$
$$searchpoint(y) + threshold > y > searchpoint(y) - threshold$$
$$searchpoint(z) + threshold > z > searchpoint(z) - threshold$$

$$Distance_{(searchpoint\,,\,currentpoint)} = \sqrt{(x - searchpoint(x))^2 + (y - searchpoint(y))^2 + (z - searchpoint(z))^2}$$

Clustering





- We use the Euclidean distance and the KD-tree to determine clusters
- **A search radius is defined**
  - *setClusterTolerance* **defines the maximum distance between points when defining the clusters**

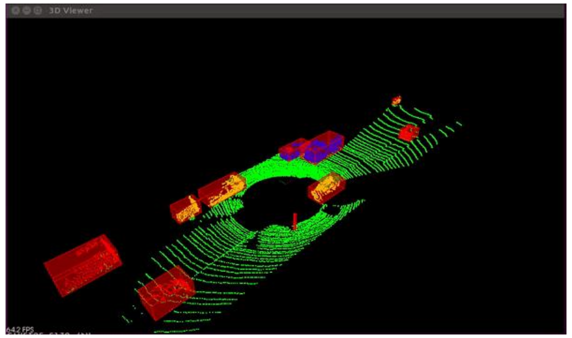
```
// Creating the KdTree object for the search method of the extraction
pcl::search::KdTree<pcl::PointXYZ>::Ptr tree (new pcl::search::KdTree<pcl::PointXYZ>);
tree->setInputCloud (cloud_filtered);

// Here we are creating a vector of PointIndices, which contain the actual index information in a vector<int>. The indices of each detected
cluster are saved here.
std::vector<pcl::PointIndices> cluster_indices;
pcl::EuclideanClusterExtraction<pcl::PointXYZ> ec;

//Set the spatial tolerance for new cluster candidates
//If you take a very small value, it can happen that an actual object can be seen as multiple clusters. On the other hand, if you set the
value too high, it could happen, that multiple objects are seen as one cluster
ec.setClusterTolerance (0.02); // 2cm

//We impose that the clusters found must have at least setMinClusterSize() points and maximum setMaxClusterSize() points
ec.setMinClusterSize (100);
ec.setMaxClusterSize (25000);
ec.setSearchMethod (tree);
ec.setInputCloud (cloud_filtered);
ec.extract (cluster_indices);
```

Euclidean clustering detection



› **Objective**: Find and segment the individual object point clusters lying on the plane

1. Down sample the point cloud
2. Segment the ground plane
3. Create a KD-tree representation for the input point cloud dataset
4. Compute Euclidean distance to build the cluster list
5. the algorithm terminates when all points have been processed and are now part of the list of point clusters

```
sudo apt-get install pcl-tools
sudo apt install libpcl-dev
```