Problem Statement

Let's learn how a self-driving car can "see" its surroundings using semantic image segmentation.

We'll cover the following

Introduction

The definition of a self-driving car is a vehicle that drives itself, with little or no human intervention. Its system uses several sensory receptors to perceive the environment. For instance, it identifies the drivable area, weather conditions, obstacles ahead and plans the next move for the vehicle accordingly.

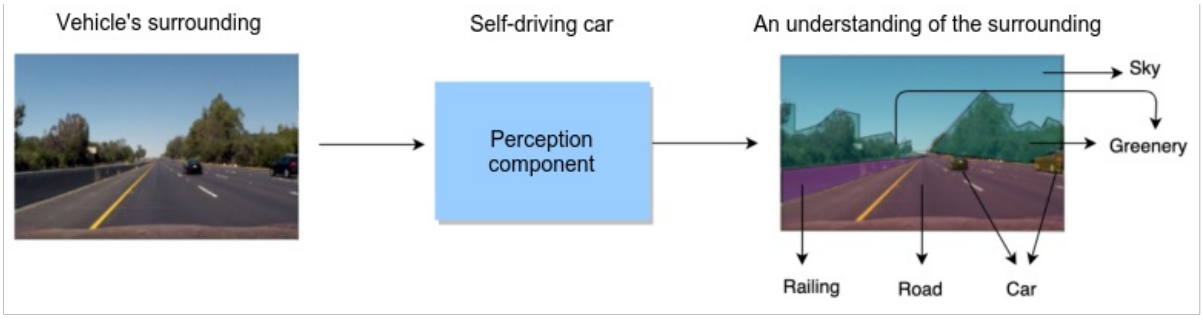
There are different levels of autonomy in such vehicles. Tesla currently implements assisted driving, where the driving is autonomous, but someone is behind the wheel. Waymo (Google's self-driving car), in contrast, is aiming for complete autonomy under all driving conditions (no driver required). Beyond human transportation, self-driving vehicles can also be utilized as a service for various purposes, e.g., Nuro is building self-driving vehicles for local goods transportation.

> Self-driving vehicles are perfect real-world systems for handling multi-sensory inputs that focus primarily on computer vision-based problems (e.g., object classification/ detection/ segmentation), using machine learning.

Now that you have some context, let's look at the problem statement.

Problem statement

The interviewer has asked you to design a self-driving car system focusing on its perception component (*semantic image segmentation in particular*). This component will allow the vehicle to perceive its environment and make informed driving decisions. Don't worry we will describe semantic image segmentation shortly.



Perform semantic image segmentation on the driving scene

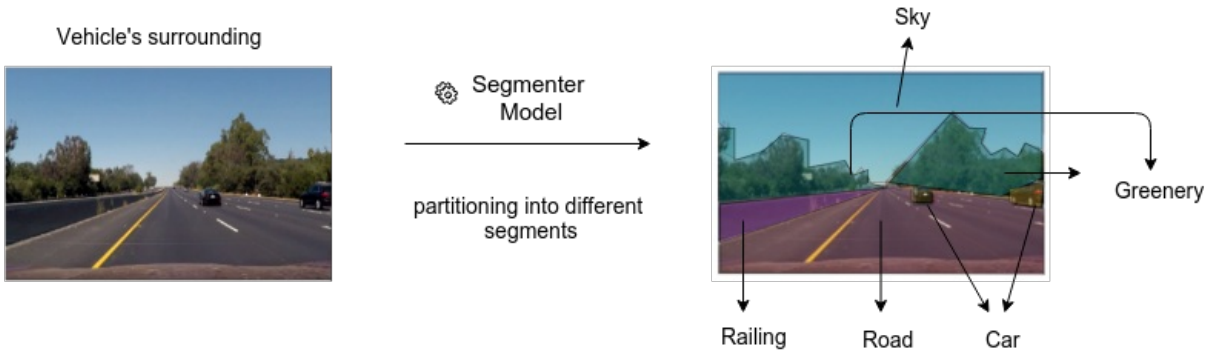
Interviewer's questions

The interviewer might ask the following questions about this problem, narrowing the scope of the question each time.

1. How would you approach a computer vision-based problem in terms of the self-driving car?
2. How would you train a semantic image segmentation model for autonomous driving?
3. How will the segmentation model fit in the overall autonomous driving system architecture?
4. How would you deal with data scarcity in imaging dataset?
5. How would you best apply data augmentation on images?
6. What are the best model architectures for image segmentation tasks?
7. Your optimized deep learning model gives a high performance on the validation set, but it fails when you take the self-driving car on the road. Why? How would you solve this issue?

> Answers
> `Q1` will be answered shortly when we talk about *subtasks*. `Q2` and `Q3` will be answered in the *architectural components* lesson. `Q4`, `Q5` and `Q7` will be answered in the *training data generation* lesson. `Q6` will be answered in the *modelling* lesson.
>
> Any other similar (multi-sensory) computer vision problem asked during the interview can be solved in a similar manner to the self-driving car problem.

Let's have a quick look at some of the major software and hardware components of the self-driving vehicle system before you attempt to answer the first question.



Semantic image segmentation of vehicle surroundings

Hardware support

Let's look at the *sensory receptors* that allow the vehicle to "see" and "hear" its environment and plan its course of action accordingly.

Camera

The camera provides the system with high-resolution visual information of its surrounding. The visual input from a frame-based camera can also be used to get an estimate of the depth/distance of objects from the self-driving car. However, the usefulness of the camera depends heavily on the lighting conditions and may not work optimally in the night scenarios.

Radar

The radar uses radio waves' reflections from solid objects to detect their presence. It adds depth on top of the visual information so that the system can accurately sense the distance between various objects. It works well in varied conditions, like low light, dirt and cloudy weather, as well as long operating distances.

Lidar

Lidar, in contrast, uses laser light reflecting off objects to create a *precise* 3D image of the environment, while measuring the distance at which objects are positioned. They are good at even detecting *small objects*. However, they do not work well during *rainy/foggy weather* and are quite *expensive*.

> One approach that exists in the industry is to get input for the system using a fusion of these sources to overcome their shortcomings.

The following diagram provides a good comparison of each type of hardware support.

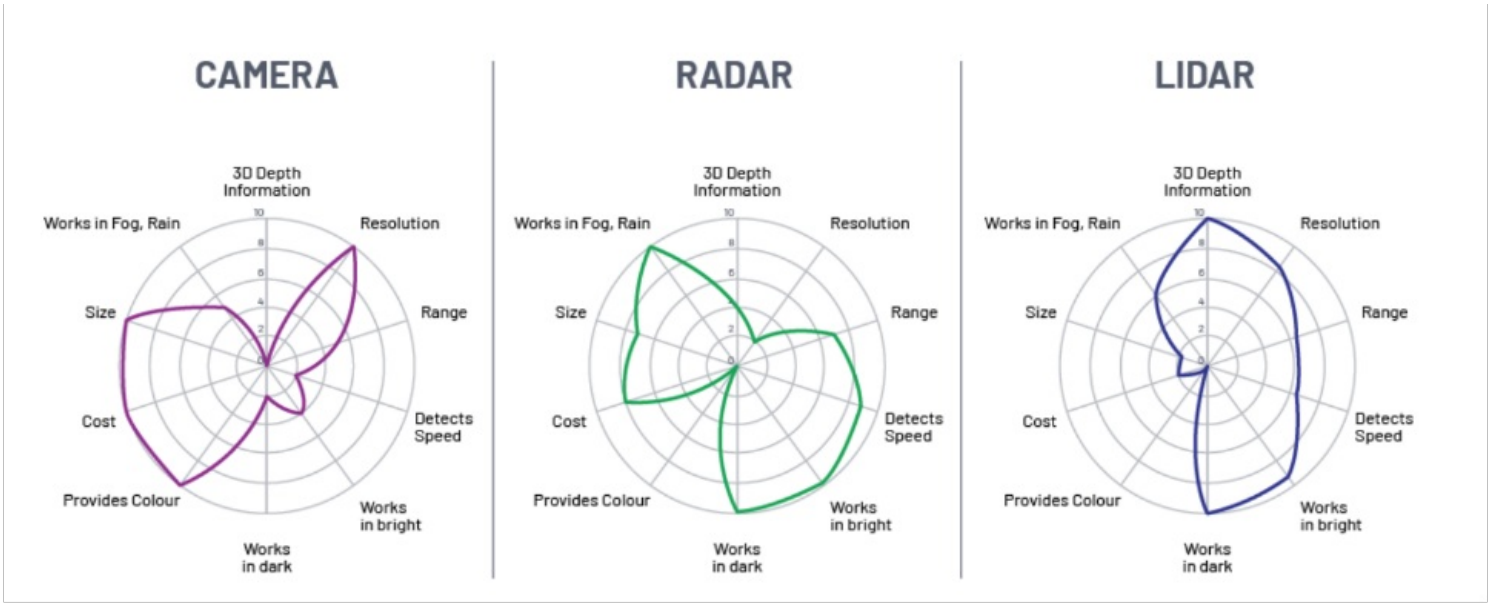


Image source: https://www.outsight.tech/technology/fused-sensing-vs.-sensor-fusion

Microphones

You have seen the devices that serve as the eyes for the self-driving system. Now, let's look at the ears of the system.

When a person hears an ambulance siren, they can detect the source of the sound, as well as the speed and direction at which the source is moving. Similarly, the self-driving vehicle uses microphones to gather audio information from the surroundings.

> Auditory information can also help the system learn about weather conditions. For instance, whether the road is wet or not (rainy weather) can be judged by the sound of the vehicle on the road. As a result, the system may decide to decrease the speed of the vehicle.

Now, you have seen the tools that provide input data to the self-driving vehicle so that it can perceive its environment.

Subtasks

The pipeline from environment perception to planning the movement of the vehicle can be split into several machine learning *subtasks* which we will describe shortly.

In the field of computer vision, the first three subtasks represent approaches that can be adapted to deal with imaging datasets (*driving images in your case*), e.g., classification, localization, segmentation, etc. Using an example, let's break this down to understand how to approach a problem statement of this scale.

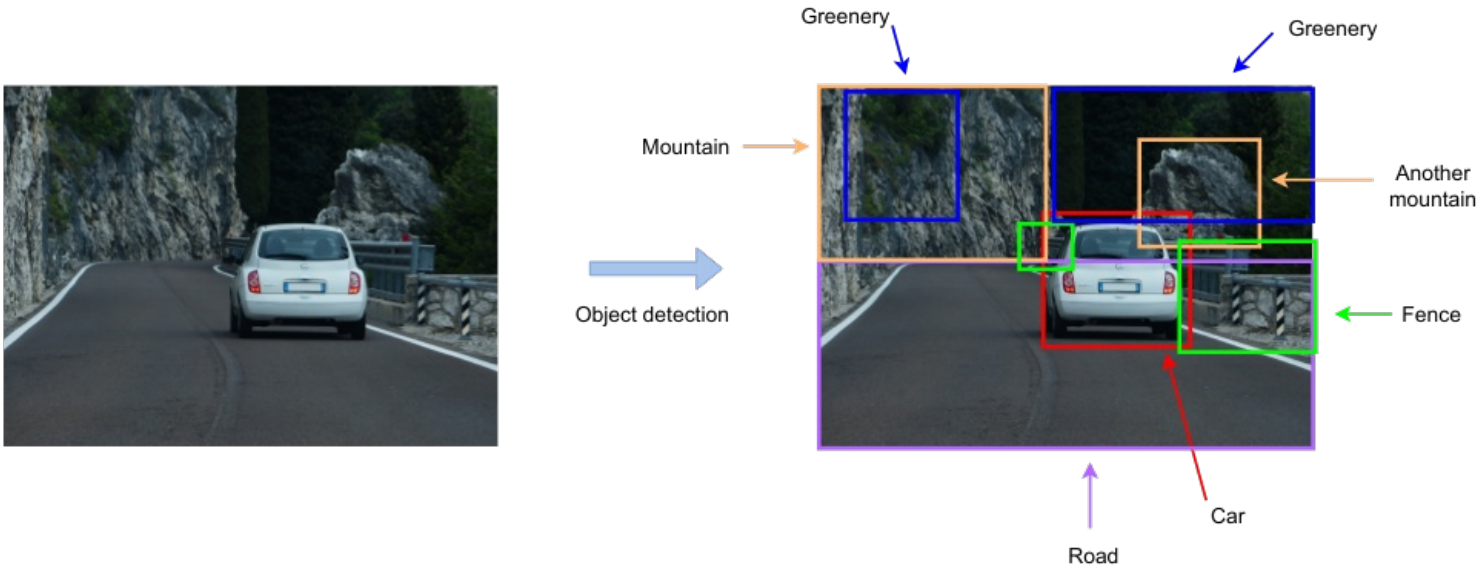
You start in the morning, and your job is to take the self-driving vehicle from Boston to New York City. Imagine a self-driving vehicle on a highway in daylight with a clear sky and less traffic. If you train an *image classier* on the labeled data, then feeding the input camera frames in real-time would probably generate the average classification scores like this: {"car":0.1, "road":0.5, "sky":0.3, "trees":0.05, "misc":0.05}.

This score provides a decent estimate of the overall categories of objects present in the surroundings. However, it does not help locate these objects. This can be solved through an *image localizer*, which will generate bounding boxes, i.e., locations on top of the predicted objects. This leads you to your first subtask, which is *object detection*.

---

1. Object detection

   In object detection, we detect instances of different class objects (e.g., greenery, fences, cars, roads, and mountains) in the surroundings and *localize* them by drawing bounding boxes.



Object detection: Different objects are identified and localized with the help of bounding boxes, as indicated in different colors

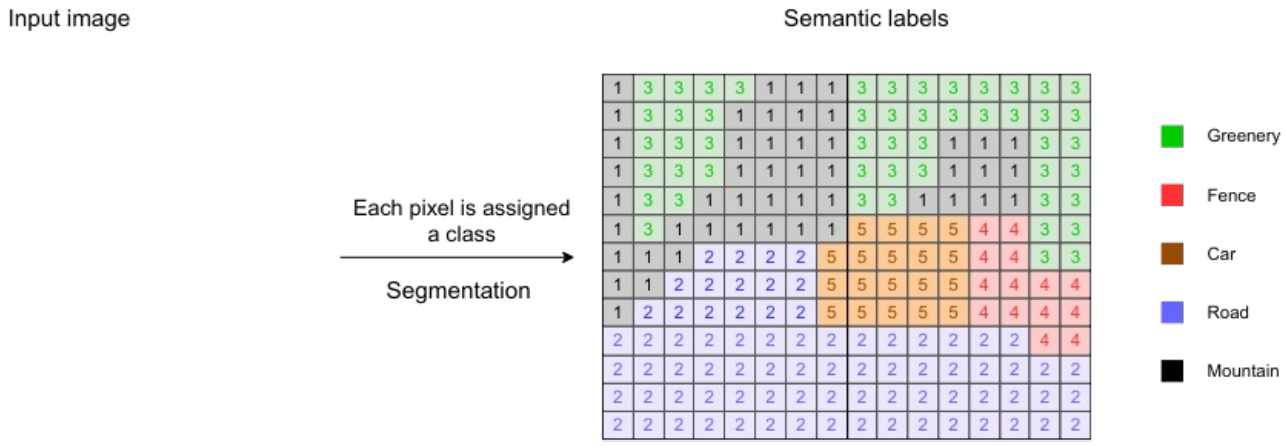
---

Continuing with the example, let's complicate the problem. You enter New York City around noon and encounter medium traffic. Your image localizer will generate a lot of overlapping bounding boxes. To resolve this, you need to upgrade the system to segment the object categories to find the optimal available path for the vehicle. You build an *image segmenter* that can generate the predicted binary mask/category on top of bounding boxes and classifications through semantic segmentation. This leads to the second subtask, i.e., semantic segmentation.

---

2. Semantic segmentation

   Semantic segmentation can be viewed as a pixel-wise classification of an image. Objects of the same class are *assigned the same label* as shown in the following diagram.



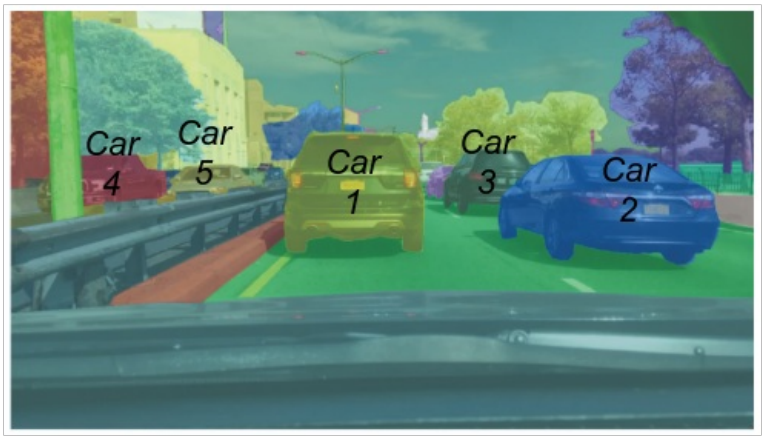
Semantic segmentation

Let's further complicate the problem. Now, you need to navigate a self-driving vehicle in a very busy street (e.g., Times Square) in Manhattan. In this case, you need to segment individual objects in real-time with high precision: You upgrade the system from semantic to instance-based segmentation, which will generate a predictive binary mask/object on top of the bounding boxes and classifications. This leads us to the third subtask, i.e., *instance segmentation*.

---

3. Instance segmentation

   Semantic segmentation does not differentiate between different instances of the same class. Instance segmentation, however, combines object detection and segmentation to classify the pixels of each instance of an object. It first detects an object (a particular instance) and then classifies its pixels. Its output is as follows:



Instance segmentation: Makes a distinction between the boundary of different instances of the same class

4. Scene understanding

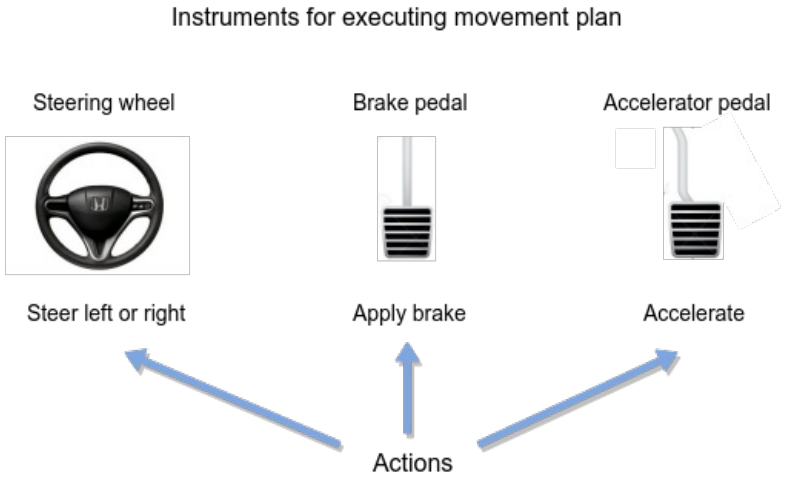
   Here, you try to understand what is happening in the surroundings. For instance, you may find out that a person is walking towards us, and you need to apply brakes. Or, as in the following diagram, we see that a car is going ahead of us on the highway.

Car ahead on the highway

Scene understanding: given an input image, the output is a text-based command: "Car ahead on the highway"

5. Movement plan

   After you have identified the objects, segmented unique objects in the image and understood the scenario, it's time to decide the movement plan for the self-driving vehicle. For example, you might decide to slow down (apply brakes) due to a car ahead.



Instruments for executing movement plan for the self-driving car

In this chapter, the focus will be the first two subtasks, i.e., the following machine learning problem:

"Perform semantic segmentation of the self-driving vehicle's surrounding environment."

← Back

Ranking

Next →

Metrics

Mark as Completed
