Universitat de les Illes Balears

Final Practice

Subject: Computer vision and 3D Reconstruction

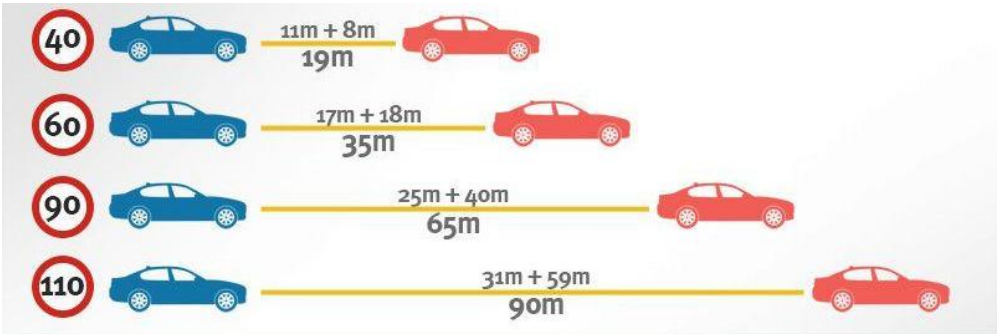
Year: 2025-2026

Context

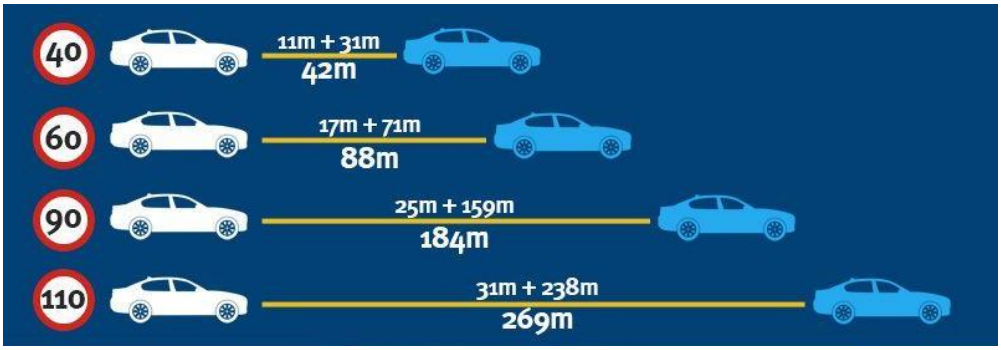
We have a self-driving car that detects an object obstructing the road, calculates its distance and reduces its speed as it approaches.

See the different braking distances, when the pavement is dry or wet:

- *Dry pavement, optimal weather*



- *Wet pavement, suboptimal weather*



Objects that you can find obstructing the road

- Pedestrians.
- Cyclists and Motorcyclists.
- Vehicles.

- Animals.
- Fallen cargo (boxes, pallets), tires or furniture dropped by other vehicles.
- Traffic cones, barrels, vms (Variable Message Signs), and temporary barriers.
- Fallen tree branches, rocks in landslide-prone areas, or deep flooding (puddles).
- Deep holes in the asphalt that can damage the vehicle.
- Etc.

Work

1. Choose one or more objects you want to detect and find a dataset that contains them.
2. Detect the object by applying YOLO or another CNN. You can use the results from your YOLO practice.
3. Decide which camera you will use and how many.
    a. The camera can be RGB-D or only RGB.
    b. You can use a single camera, or multiple cameras to capture images from different points of view.
4. If you choose to use only RGB images:
    a. You need images with different points of views.
    b. You can detect and match the key points and use Epipolar Geometry to find the 3D coordinates of the object.
5. If you choose to use RGB-D images:
    a. You need the original RGB image to detect the bounding box of the object and the RGB-D image to obtain the depth distance in that position.
6. Test your program. Take one or more photos with **your camera** of the chosen object. If you have an RGB-D camera you only need one photo to calculate the distance. Otherwise, you need a minimum of two RGB images..

    Example: Place a ball in the middle of a road, take a photo, and use your program to determine the distance to the object. Repeat this step at different distances.

7. Display the distance (in meters) on the output image, along with the recommended driving speed.
8. Verify your results. Measure the distance between the camera and the object using a measuring tape (in the real world) and compare it with your program's result.
9. What method do you think is better? Why? You can implement the two methods and compare them. You will get a higher grade if you implement both methods.

10. Conclusions. Advantages and disadvantages of the chosen method.

Rules

- Cite all used bibliography or webpages.
- Use your own images. Are not allowed to share images between students.
- All practices must be solved individually.
- Source code must be clean and commented when required.
- Answers should relate to what has been learned in theory.
- Too similar practices will be considered copy and will be rated by 0.
- Longer doesn't mean better.
- Presentation will have a strong weight in the final mark.

The submission must be a PDF document with the results and answers to the questions above. You should also submit all the source code used to solve the problem with instructions to execute it.

Respuesta punto 5 (RGB-D) - Uso de RGB y depth en img/24_1_2026

Analisis de los ficheros de profundidad (Polycam keyframes):
- Hay 3 keyframes. Las imagenes RGB estan en keyframes/images (1024x768) y los mapas de profundidad en keyframes/depth (PNG 16-bit, 256x192). Los nombres coinciden (timestamp), por lo que cada RGB tiene su depth correspondiente.
- Los mapas de profundidad estan a 1/4 de resolucion. Para convertir coordenadas de un bbox detectado en RGB a depth: x_d = x_rgb * 256/1024, y_d = y_rgb * 192/768.
- Los valores del depth estan en milimetros. Ejemplo real (id 238019426873): center depth = 1053 (mm) y center_depth = 1.052 m en el JSON de camara. Por tanto distancia_m = depth_mm / 1000.
- Existe keyframes/confidence (0-255) para filtrar pixeles con poca confianza.

Flujo RGB-D aplicado:
1) Detectar el bbox en la imagen RGB (YOLO).
2) Llevar el centro (o una ventana dentro del bbox) al depth, usando la escala 1/4.
3) Leer la profundidad (mediana de ventana) y convertir a metros.
4) (Opcional) Filtrar usando el mapa de confianza.

Con esto se cumple el requisito: RGB para localizar el objeto (bbox) y RGB-D para obtener la distancia en esa posicion.