

Conducción autónoma sobre plataforma real y simulada con seguimiento de carril e identificación de señales de tráfico y peatones mediante redes neuronales

Álvaro Mariscal Ávila

a.mariscal.2018@alumnos.urjc.es



Trabajo fin de grado

5 de julio de 2022



(CC) Julio Vega

Este trabajo se entrega bajo licencia CC BY-NC-SA. Usted es libre de (a) compartir: copiar y redistribuir el material en cualquier medio o formato; y (b) adaptar: remezclar, transformar y crear a partir del material. El licenciador no puede revocar estas libertades mientras cumpla con los términos de la licencia.

Contenidos

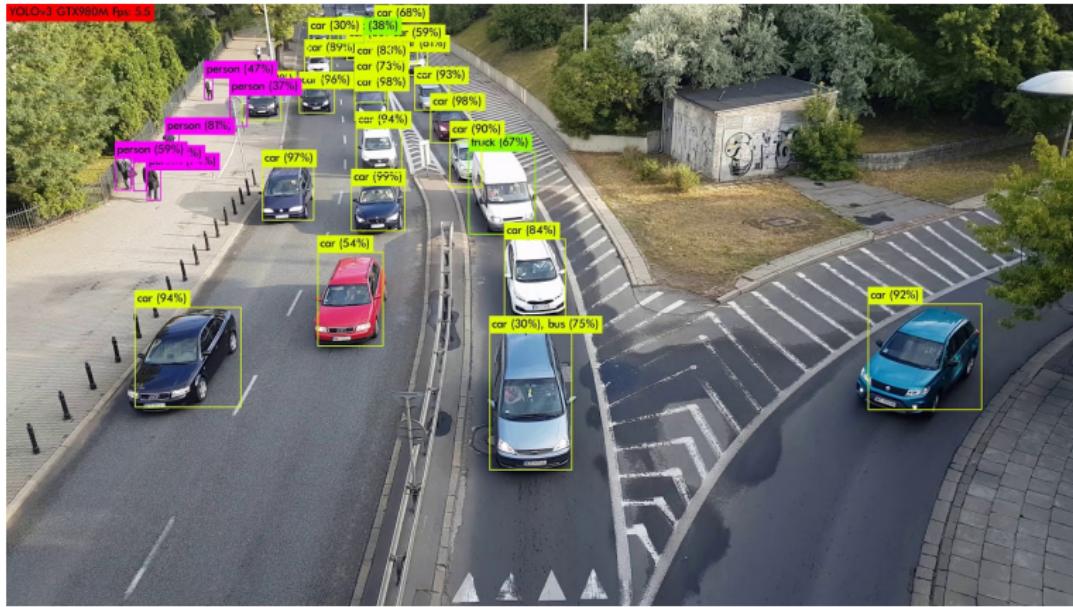
- 1 Introducción
- 2 Objetivos
- 3 Plataforma de desarrollo
- 4 Sistema de conducción autónoma
- 5 Conclusiones

Introducción

Inteligencia artificial

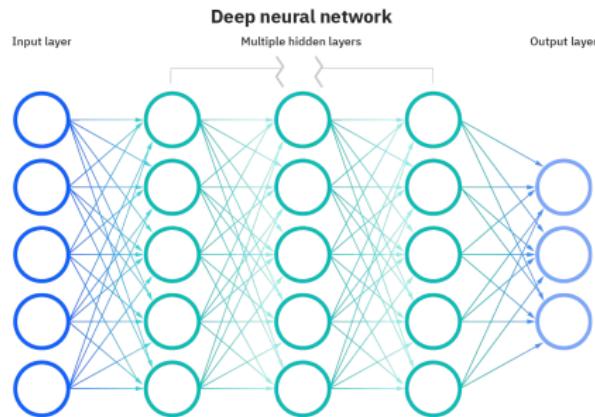
- Entender y emular el comportamiento humano.
- Dotar a sistemas de cierta inteligencia y de la capacidad de aprender.
- Visión artificial, aprendizaje automático o aprendizaje profundo.

Visión artificial



Deep Learning

- Neocognitrón (1979): Red neuronal con 5 o 6 capas para reconocer caracteres japoneses.



Vehículos autónomos



AMRs

- Sucesores de los AGVs.



Objetivos

Descripción del problema

- Desarrollar un **coche autónomo** capaz de circular por un **circuito** en un entorno dinámico, interactuando con objetos propios de una ciudad en dos entornos distintos:
 - Entorno **simulado** con *Gazebo*.
 - Entorno **real** usando un robot con *Jetson Nano*.
- En ambos entornos se han de cumplir dos subobjetivos:
 - Seguimiento de **carril**
 - Detección de **objetos**

Requisitos

- El sistema operativo utilizado será *GNU/Linux*, concretamente la distribución *Ubuntu 18.04 LTS*.
- El entorno simulado requerirá la presencia de una tarjeta gráfica dedicada: *NVIDIA* y *CUDA*.
- El entorno real requerirá un robot con la placa de desarrollo *NVIDIA Jetson Nano*, ya que esta es una de las placas con *GPU* más económicas.
- El lenguaje de programación utilizado será *Python*.

Plataforma de desarrollo

NVIDIA Jetson Nano

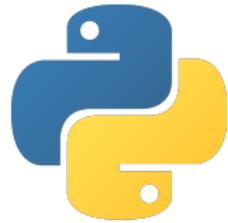
- Placa de desarrollo de **bajo coste** con **GPU** dedicada.
- Arquitectura **Aarch64**, soporte para **GNU/Linux** y puertos **GPIO**.



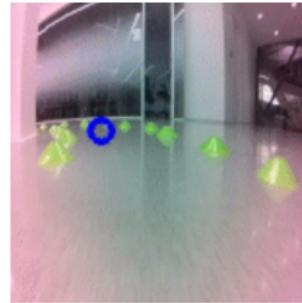
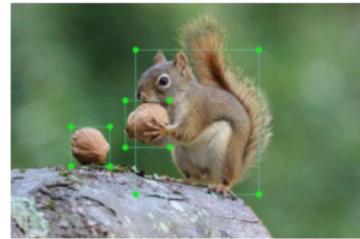
Componentes



Software



Software relacionado con visión



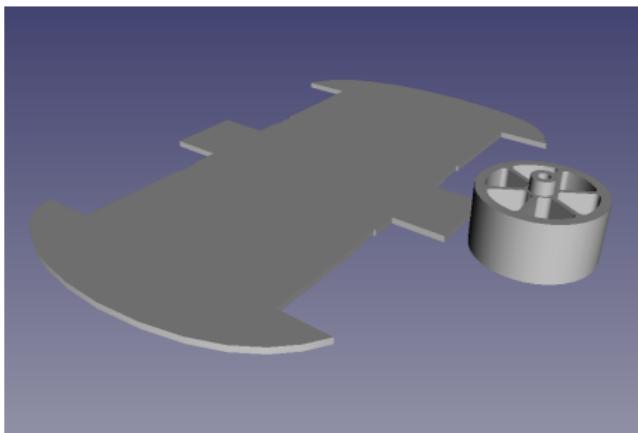
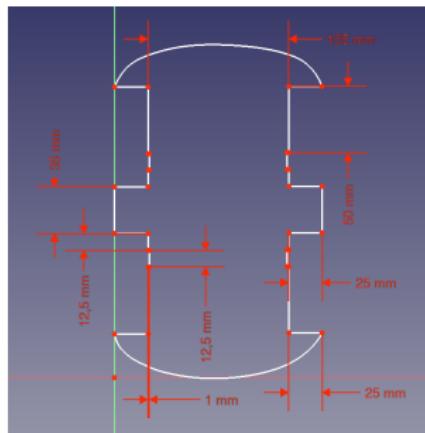
Sistema de conducción autónoma

Modelo de la ciudad en Gazebo

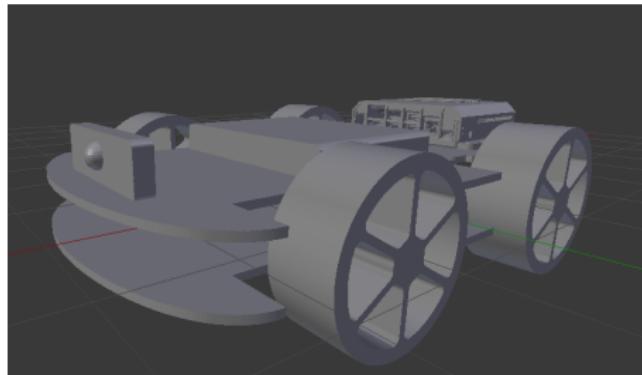
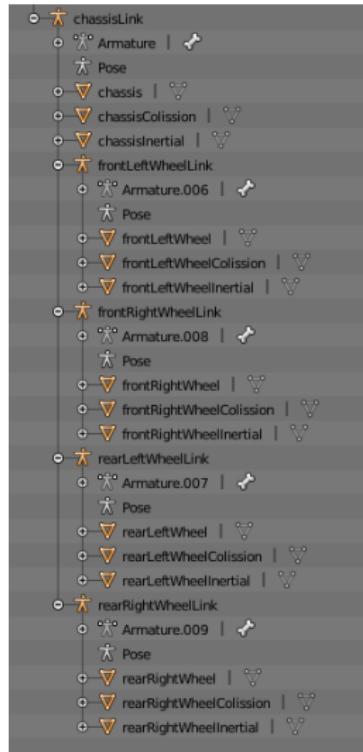


- Reproducible en el entorno real
- Semáforo y peatón dinámico

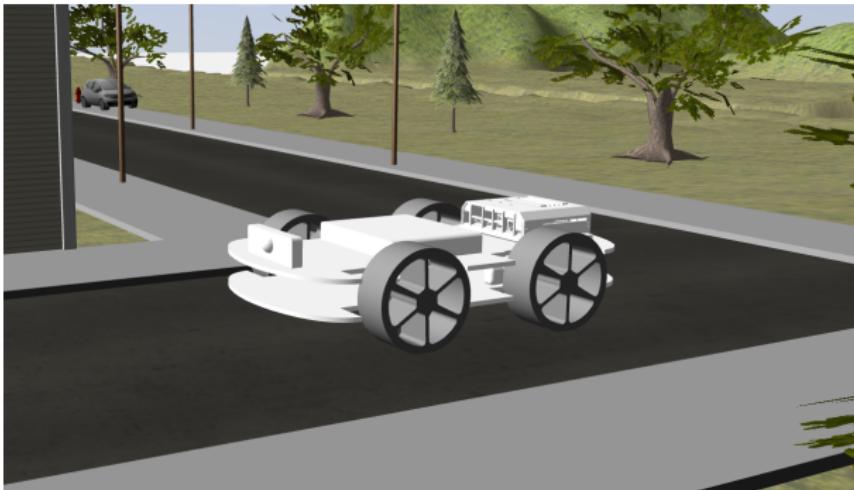
Modelo del vehículo en *FreeCAD*



Modelo y jerarquía del vehículo en *Blender*

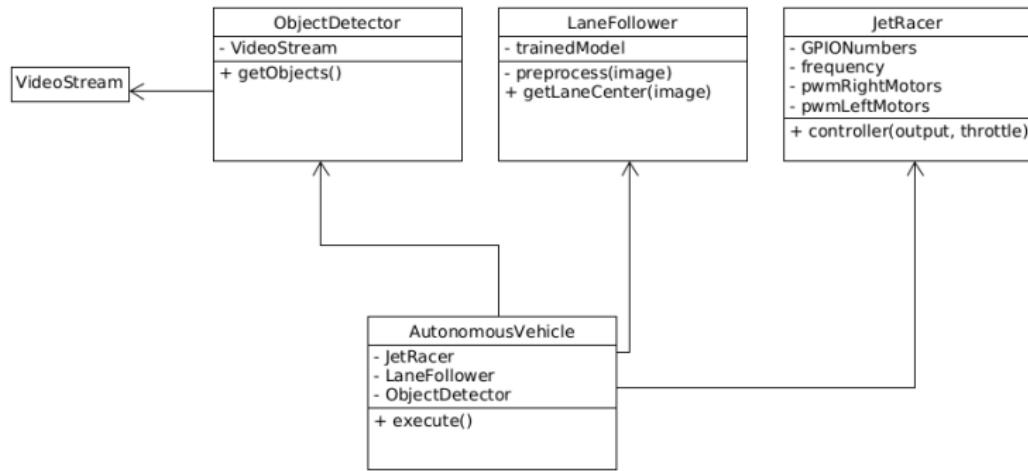


Modelo del vehículo en Gazebo



- *Plugins* para movimiento y cámara conectados con *ROS*

Diagrama de clases

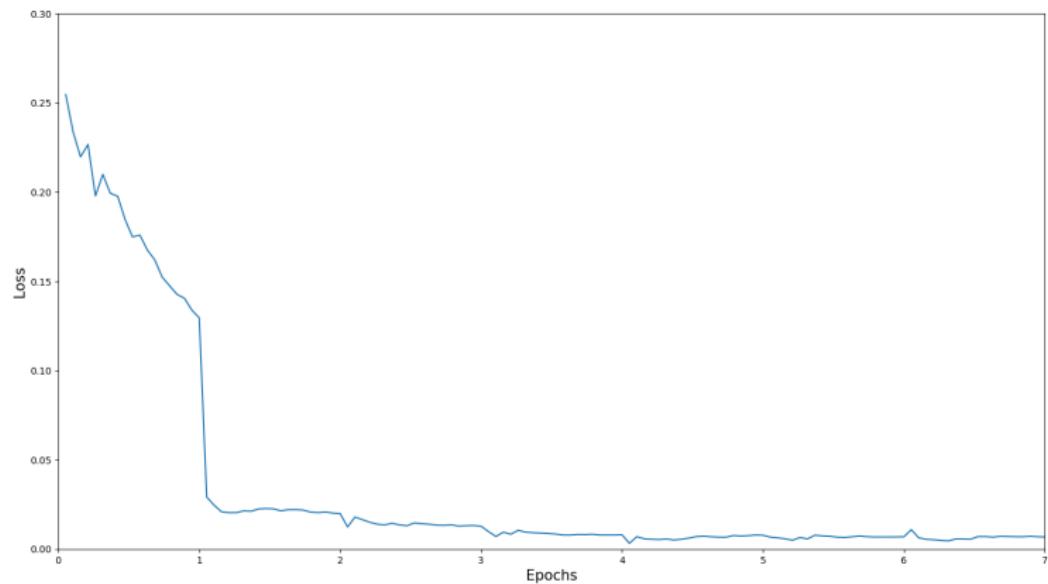


Seguimiento de carril



- Red *ResNet-18* preentrenada: convolucional de 18 capas con **bloques residuales**.
- Notebook de *Jupyter*.
- Guardado de cada imagen: *x_y_identificador_unico.jpg*.
- Dataset con situaciones **difíciles**.

Entrenamiento red seguimiento de carril



- En cada *epoch* se calcula el error cuadrático medio (*loss*).
- Propagación hacia atrás de los errores desde la capa de salida hasta la primera capa.

Salida red seguimiento de carril



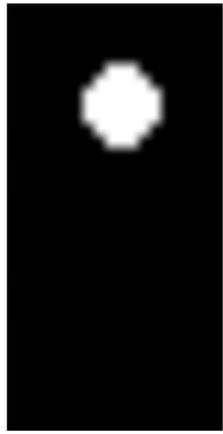
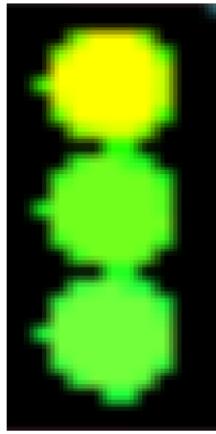
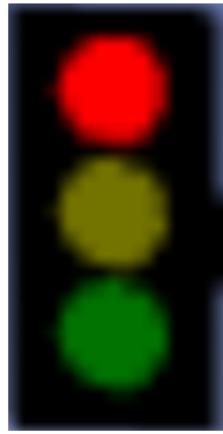
- 224 píxeles.
- Entorno experimental: **autómata**.

Detección de objetos

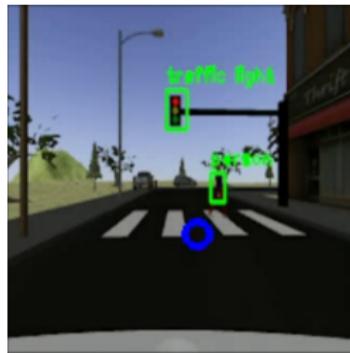


- *YOLO V3 Tiny*.
- Objetos muy *genéricos*: alta probabilidad de detección.
- *Bounding Box*: clase y probabilidad.

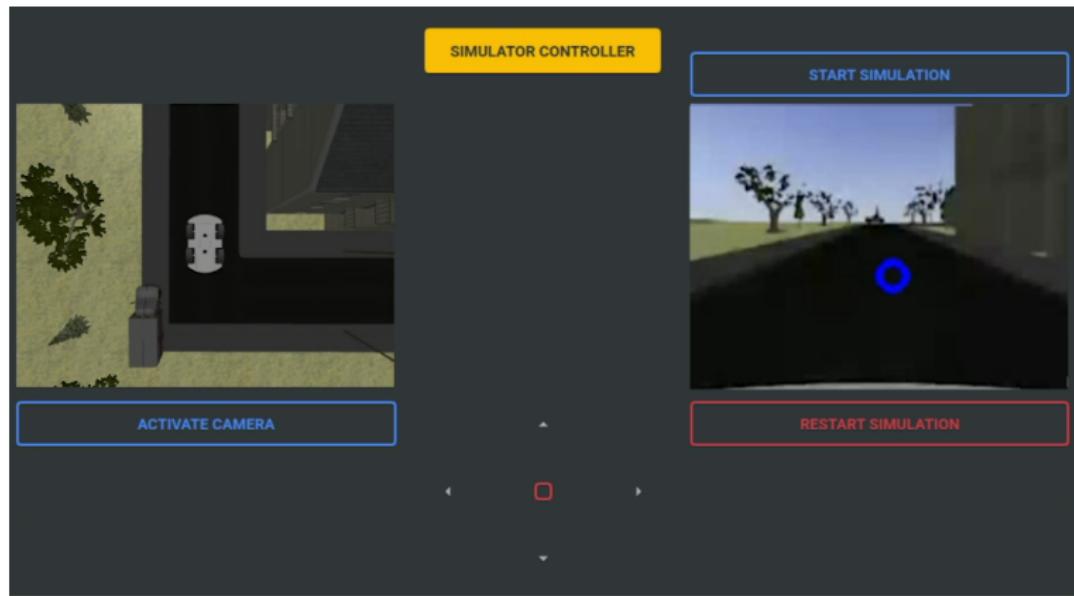
Detección de semáforo



Ejecución en el simulador



Interfaz de usuario



Entorno real

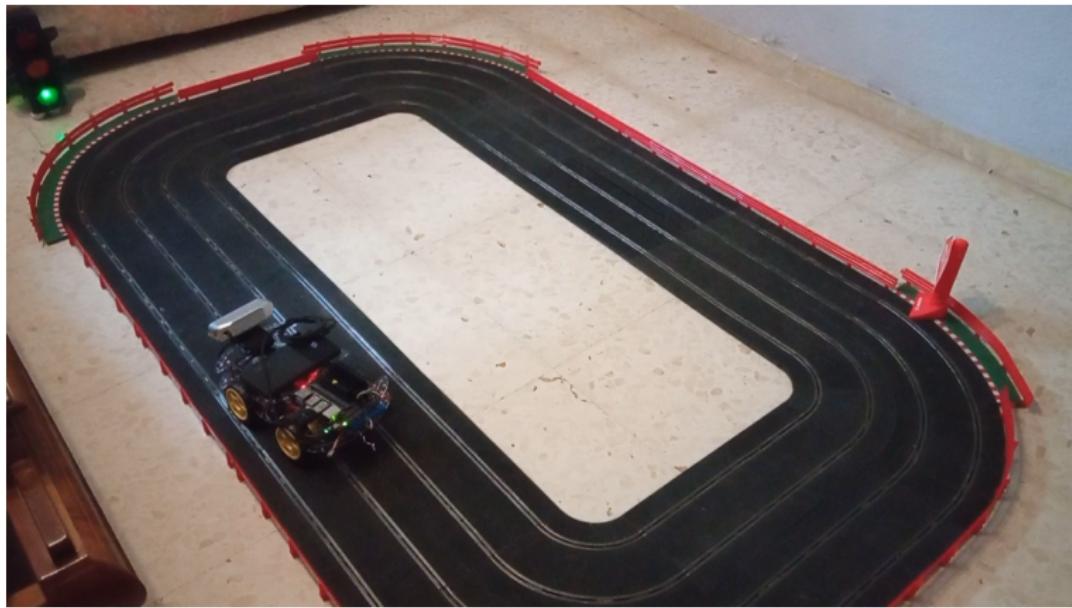


Círculo inicial



- Entrenamiento con luz *artificial*.

Circuito con objetos



Dataset objetos reales

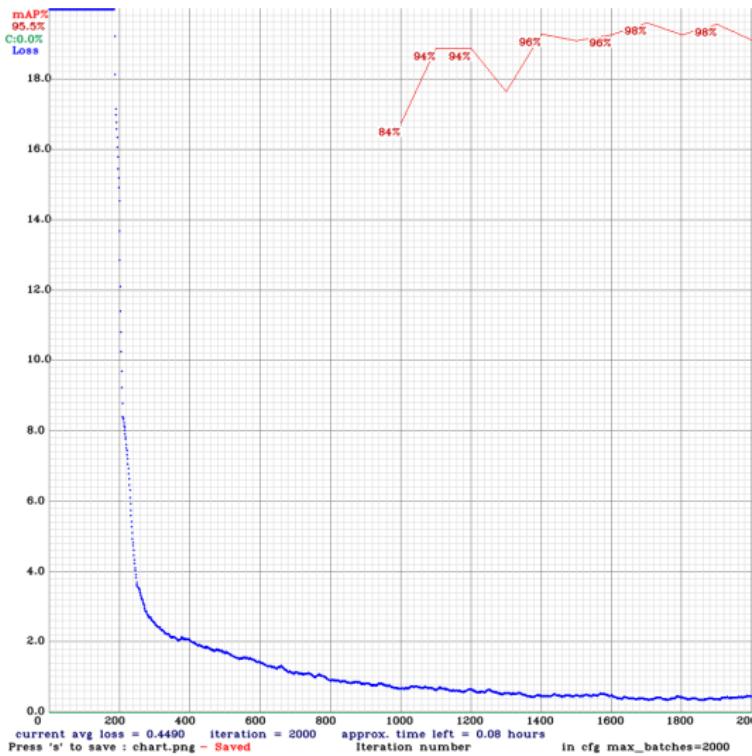


Entrenamiento red de detección de objetos



- Portátil con *NVIDIA MX330* $\simeq 95^\circ C$.
- 2000 iteraciones $\simeq 3$ horas.
- Únicamente con *dataset* propio.

Entrenamiento red de detección de objetos

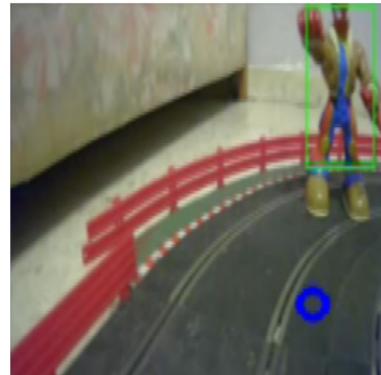
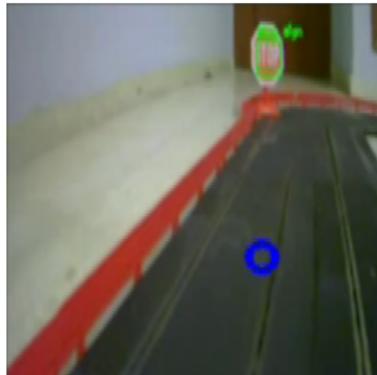


Comparación probabilidad de detección

- Imagen nueva para ambas redes.



Ejecución en el entorno real



Conclusiones

Conclusiones

- Para lograr los objetivos se han utilizado dos **redes neuronales**:
 - Seguimiento de carril: librería *JetRacer* que implementa una red *residual ResNet-18* combinada con un **controlador**.
 - Detección de objetos: *framework Darknet* que permite ejecutar una red *convolucional YOLO V3 Tiny* previamente entrenada mediante un **dataset** propio con los objetos reales para aumentar la fiabilidad.
- Todo ello ha sido combinado en dos paquetes **ROS**, diferenciando entorno simulado y real.
- Limitaciones: **ángulo** de la cámara y **resolución** de imagen en las redes neuronales.

Líneas futuras



Conducción autónoma sobre plataforma real y simulada con seguimiento de carril e identificación de señales de tráfico y peatones mediante redes neuronales

Álvaro Mariscal Ávila

a.mariscal.2018@alumnos.urjc.es



Trabajo fin de grado

5 de julio de 2022