



UNIVERSIDAD DE MÁLAGA



Trabajo fin de grado

Sistema de tracking y reidentificación de jugadores en fútbol amateur

Realizado por
Soriano Muñoz Juan Ignacio

Profesor encargado:
Luque Baena Rafael Marcos
Jerez Aragonés Jose Manuel
Departamento
Lenguajes y Ciencias de la Computación

MÁLAGA, DICIEMBRE de 2024



UNIVERSIDAD
DE MÁLAGA



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
ESTUDIANTES DE INGENIERÍA BIOINFORMÁTICA

Sistema de tracking y reidentificación de jugadores en fútbol amateur

Trabajo fin de grado

Realizado por
Soriano Muñoz Juan Ignacio

Profesor encargado:
Luque Baena Rafael Marcos
Jerez Aragonés Jose Manuel

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, DICIEMBRE DE 2024

Contents

1	Introducción	3
2	Diario de avances	3
2.1	Semana 3-16 de marzo)	3
2.2	Semana 17-31 de marzo [1]	3

1 Introducción

2 Diario de avances

2.1 Semana 3-16 de marzo)

Por ahora lo que llevamos es un dataset hecho en roboflow con un partido de España contra Suiza. Realizamos capturas y dividimos el conjunto en training, validation y test.

Entrené el modelo de YOLO con este dataset revisado y el modelo no supo detectar bien el balón debido a la poca cantidad de imágenes donde se pueda ver bien la bola. El árbitro y los jugadores fueron bien detectados.

Se replanteó el objetivo del TFG. Se focalizará en la reidentificación de jugadores cuando salen fuera de plano y en el desarrollo de una aplicación que permita al usuario decidir si cuando se produce un cambio de identificador, mantenerlo o cambiarlo, creando un dataset revisado.

2.2 Semana 17-31 de marzo [1]

A la hora de medir resultados como estas gráficas:

Results on Datasets

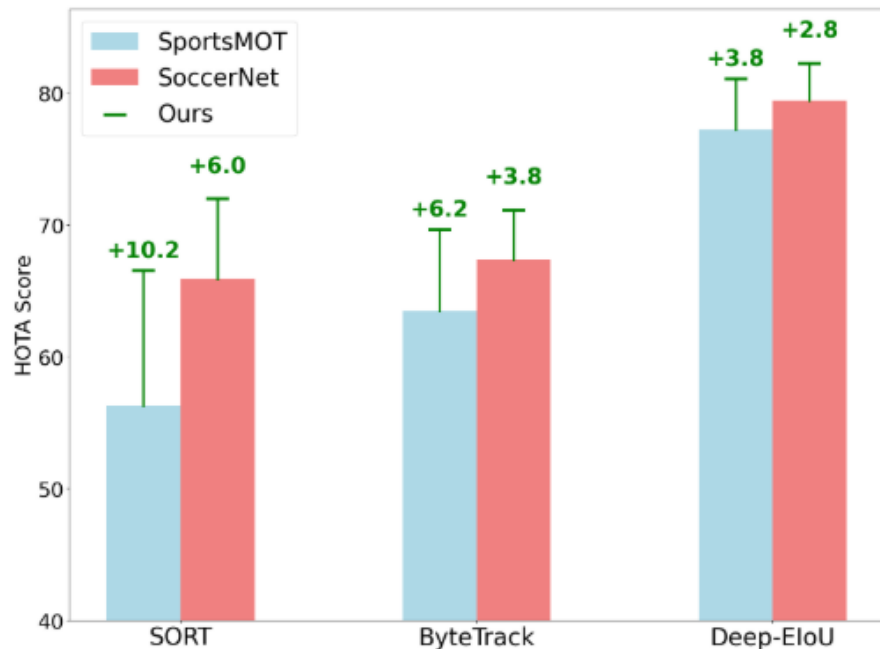


Figure 1: Métricas [gta_link](#)

Nos centraremos en la medida de los ID's intentando minimizarlos, lo máximo posible, ya que la métrica significa número de ids generados.

Me he puesto a mirar lo que hace exactamente el código del repositorio de [gta-link](#). Y en resumidas cuentas necesitamos un dataset con una pred, ya realizada,

Table 1: Tracking performance on SportsMOT before and after applying our Global Tracklet Association (GTA) method.

Method	HOTA↑	AssA↑	IDF1↑	DetA↑	MOTA↑	IDs↓
SORT [26]	56.28	42.67	58.83	74.30	85.11	5180
SORT + GTA	66.52 (+10.24)	59.59 (+16.92)	77.37 (+18.54)	74.29	85.27	3547 (-1633)
ByteTrack [31]	63.46	51.81	70.76	77.81	94.91	3147
ByteTrack + GTA	69.74 (+6.28)	62.61 (+10.80)	83.16 (+12.40)	77.72	95.01	2107 (-1040)
Deep-Elou [19]	77.21	67.63	79.81	88.22	96.30	2909
Deep-Elou + GTA	81.04 (+3.83)	74.51 (+6.88)	86.51 (+6.70)	88.21	96.32	2737 (-172)

Figure 2: **Objetivo**

como es el caso de SoccerNet. Podemos descargar el dataset con el tracking realizado en los archivos .txt que se encuentran dentro de cada clip.

He utilizado las siguientes líneas de comando para realizar la descarga del dataset.

```
from SoccerNet.Downloader import SoccerNetDownloader
mySoccerNetDownloader = SoccerNetDownloader(LocalDirectory="path/to/SoccerNet")
mySoccerNetDownloader.downloadDataTask(task="tracking",
    split=["train", "test", "challenge"])
```

Posteriormente hice unos ajustes en el código para que cogiera los archivos `gt.txt` para que se tomara como referencia **—pred_dir tracking results directory**, ya que estos son archivos **MOT**, que guardan información sobre los **bounding box** de cada objeto de cada frame, conteniendo información sobre **frame**, **id**, **bb_left**, **bb_top**, **bb_width**, **bb_height**, **conf**, **x**, **y**, **z**, en este orden.

Estos son esenciales para que se pueda ejecutar el archivo `generate_tracklets.py` para generar los **tracklets**.

Un **tracklet** se genera al seguir a un objeto desde su detección en un fotograma hasta el siguiente. En este proceso, el código asocia las detecciones de objetos de cada fotograma con un identificador único (**ID**) para cada objeto, y los agrupa en una **”trayectoria”** que sigue ese objeto a lo largo del tiempo.

En el código:

- **Extracción de características:** El código utiliza un modelo de reidentificación de personas (**FeatureExtractor**) para extraer características visuales de cada objeto detectado en los fotogramas. Esto es útil para seguir objetos entre fotogramas, incluso cuando se producen cambios de apariencia debido a variaciones en la vista o el movimiento.
- **Cálculo de tracklets:** Para cada fotograma, el código agrupa las detecciones de objetos utilizando el identificador único (**track_id**). Si un objeto se detecta en un fotograma y luego aparece en otro, se agrega a un tracklet, que es una colección de todas las detecciones del mismo objeto a lo largo de varios fotogramas. Además, se guarda información como el puntaje de la detección y las características extraídas del modelo de reidentificación.
- **Salvado de tracklets:** Al final de cada secuencia de video o serie de fotogramas, los tracklets generados se guardan en un archivo **pickle** para su posterior uso, permitiendo analizar y trabajar con las trayectorias de los objetos en el futuro.

El programa `generate_tracklets.py`, lo que hará será generar unos ficheros `.pkl` que guardan los datos de las trayectorias de los objetos detectados a lo largo del tiempo. Esto incluye, por ejemplo, las **detecciones de objetos en cada fotograma**, las **características extraídas por el modelo de reidentificación**, y los **identificadores únicos (ID)** asignados a cada objeto (en formato binario).

Una vez que se tenga esos ficheros se realiza una **refinación de los tracklets** para evitar que se cambien los **ids con frecuencia**. Mejora los **tracklets** (trayectorias de objetos) generados por un **tracker en tareas de seguimiento de múltiples objetos (MOT)**. Utiliza dos componentes principales: el **Tracklet Splitter**, que divide **tracklets impuros** (con múltiples identidades) en subtracklets más precisos mediante **clustering (DBSCAN)**, y el **Tracklet Connector**, que fusiona **tracklets fragmentados** que pertenecen al mismo objeto basándose en **similitudes visuales y restricciones espaciales**. Los resultados refinados se guardan en archivos `.txt` en formato **MOT**, listos para su evaluación, visualización o análisis posterior. Este proceso optimiza la **precisión del seguimiento**, corrigiendo errores como **cambios de identidad y fragmentaciones**.

Ahora que tenemos un dataset etiquetado, tendremos que:

- **Encontrar una combinación de hiperparámetros óptima:** Con el objetivo de que el cambio de ids sea el mínimo posible.
- **Desarrollar una aplicación:** Con el onjetivo de que avise al usuario sobre los cambios de id en los frames y confirme si está bien cambiado o debería conservarse el anterior.

References

- [1] sjc042. *gta-link*. 2025.



UNIVERSIDAD
DE MÁLAGA

| **uma.es**

E.T.S. DE INGENIERÍA INFORMÁTICA

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga