Proyecto AffluenceCounter

Documento de Diseño

Autores: Israel Peñalver
Alex Darío Cevallos
David Valladares

Índice

1 Introducción

En esta sección se proporcionará una descripción sobre el desarrollo realizado para construir la aplicación de AffluenceCounter y se mostrará su funcionamiento utilizando como referencia los test desarrollados. Por último se señala donde se encuentra el repositorio que aloja todo el desarrollo.

1.1 Desarrollo

La aplicación irá leyendo los frames del vídeo que se le pasa como entrada. Para cada frame del vídeo se realiza una primera etapa que consiste en detectar personas en la imagen, utilizando una red yolo pre-entrenada, por lo que podemos obtener los bounding box de las personas detectadas. Y existe una segunda etapa que es la encargada de seguir a las personas, identificándolas con un identificador único. Para contabilizar las personas que entran a la tienda o pasan de largo, utilizamos tres zonas de referencia, una zona en la izquierda de la imagen, otra en la zona derecha y otra en la entrada de la tienda.

1.2 Descripción

El funcionamiento general de la aplicación consistirá en pasar un vídeo como entrada y se devolverá las personas que han entrado en la tienda y las que han pasado de largo. Las figuras 1 y 2 reflejan un test que contempla este funcionamiento general, dónde se contabiliza a una persona que entra en la tienda. Para ello se hace uso de las referencias mencionadas en el apartado anterior:

- Si una persona pasa por la zona de referencia izquierda o derecha y después por la zona de referencia de la puerta, esto quiere decir que la persona ha entrado en la tienda.
- Si una persona pasa por la zona de referencia izquierda o derecha y posteriormente vuelve a pasar por una de estas zonas, eso indica que la persona ha pasado de largo.

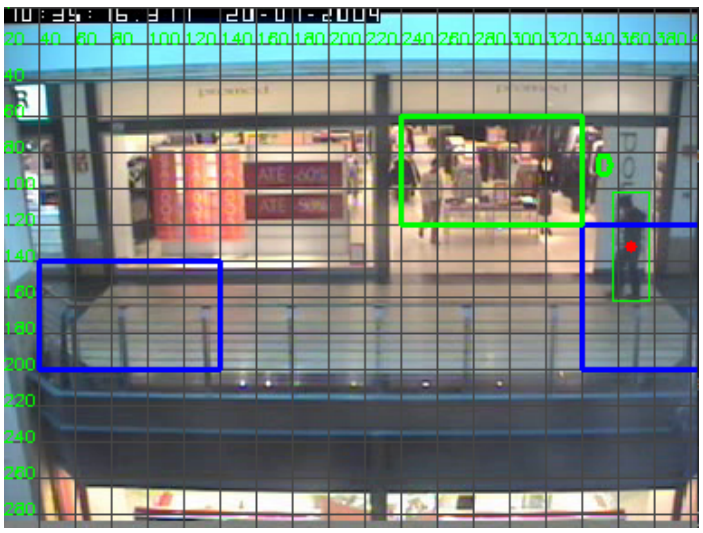

Figura 1. Persona detectada (id0) que pasa por la zona de referencia derecha.

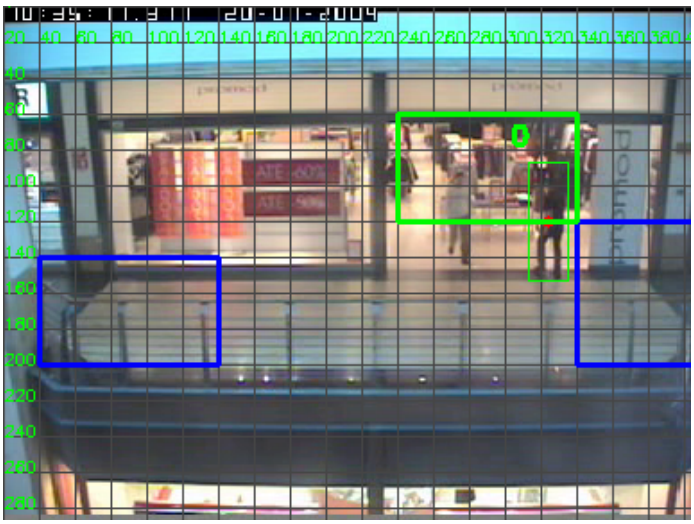

Figura 2. Persona detectada (id0) que pasa por la zona de referencia de la puerta.

En cuánto al funcionamiento interno de la aplicación, para cada frame obtenido primero se detectan las personas que aparecen en el frame. En la figura 3 se muestra un ejemplo en el que se detectan a dos personas en la imagen.

Figura 3. Detección de dos personas en un frame.

Una vez obtenidos los bounding box de las personas detectadas, el siguiente paso es realizar un seguimiento de ellas, para ello se las identifica con un identificador único. En la figura 4 se muestra un ejemplo de seguimiento de dos personas.

Figura 4. Seguimiento de dos personas.

Mediante este seguimiento de las personas se puede contabilizar aquellas que pasan de largo o entran a la tienda, utilizando los puntos de referencia para ello. En la figura 5 se muestra un ejemplo de test donde se detecta que una persona ha pasado por la zona de referencia izquierda.

Figura 5. Persona pasando por la zona de referencia izquierda.

1.3 Repositorio

Tanto el desarrollo realizado como la planificación del proyecto se encuentran alojadas en el repositorio de GitHub:

https://github.com/dvalladaresv/AIVA_2021_Deteccion_de_actividad_grupo_F

2 Diseño de la solución

En esta sección se muestran y detallan los diagramas de clases, secuencias y actividad que han permitido el desarrollo de la aplicación.

2.1 Diagrama de clases

A continuación, en la figura 6 se muestra el diagrama de todas las clases necesarias para el funcionamiento de la aplicación AffluenceCounter.

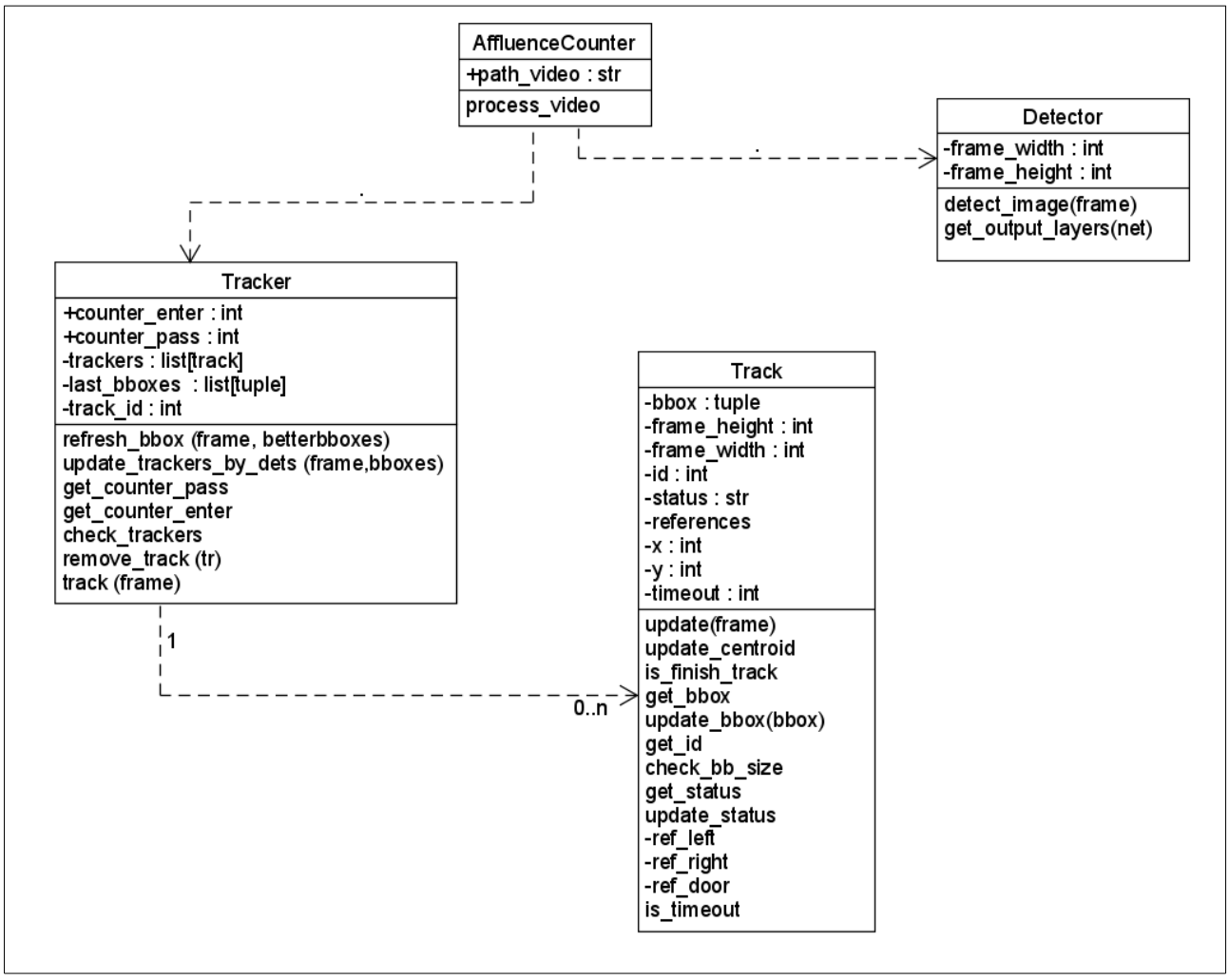

Figura 6. Diagrama de clases

Como puede observarse en el diagrama, las cuatro clases necesarias son: AffluenceCounter, Detector, Tracker y Track. AffluenceCounter es la aplicación principal, y utiliza principalmente dos clases, que son la clase Detector y la clase Tracker. Resumidamente se puede decir que la clase Detector va a servir para detectar a las personas, y la clase Tracker para seguir a las personas

detectadas. Además, la clase Tracker puede tener 0..n Tracks, ya que en Tracker se gestionan los seguimientos, y cada seguimiento se encuentra en la clase Track.

AffluenceCounter: Es la clase principal, lo que necesita para funcionar es la ruta del video que va a procesar, y esta clase se encarga de ir enviando los frames al detector y las detecciones al tracker para poder realizar el conteo, y finaliza la ejecución realizando un conteo final de todos los clientes que han pasado por delante de la tienda y todos los que han entrado.

Detector: Esta clase recibe frames y devuelve las bboxes de las personas detectadas, para realizar esta detección de personas se utiliza una red yolo pre-entrenada.

Tracker: Es la clase encargada de gestionar los seguimientos, ya que cada seguimiento individualizado se gestiona en la clase Track, su misión principal es la de intentar emparejar los bboxes propuestos por el detector con los Tracks que ya estaba gestionando, en caso de poder emparejar correctamente debe actualizar las posiciones, y en caso de que no se pueda emparejar debe encargarse de crear un nuevo Track, que pasará a ser gestionado. Además, es capaz de eliminar Tracks cuando ha pasado cierto tiempo sin actividad, asumiendo que se había detectado un falso positivo.

Track: Es necesaria para realizar el seguimiento de una persona desde que aparece en la escena hasta que desaparece para analizar por que lugares ha pasado y así determinar si ha pasado a la tienda o si ha pasado de largo.

2.2 Diagrama de secuencias

Para el diagrama de secuencias (Figura 7) se han utilizado las cuatro clases, AffluenceCounter, Detector, Tracker y Track. Para la detección, que se ejecuta desde “AffluenceCounter hasta Detector”, se ha utilizado el método “detect_image”. Para actualizar los bounding box de los trackers existentes o la creación de uno nuevo, se ejecuta desde “AffluenceCounter hasta Tracker”, se ha utilizado el método “update_trackers_by_dets”. Para actualizar el bbox de cada persona de forma individual, se ejecuta desde “Tracker hasta track”, se utiliza el método “refresh_bbox”. Para actualizar el seguimiento de las personas, se ejecuta desde “AffluenceCounter hasta Tracker”, se utiliza el método “track”. Para poder comprobar el estado del seguimiento de las personas, se ejecuta desde “AffluenceCounter hasta Tracker”, se utiliza el método “check_trackers”. Para obtener los resultados una vez procesado el vídeo, se ejecuta desde “Affluence counter hasta Tracker”, se utilizan los métodos “get_counter_pass” y “get_counter_enter”.

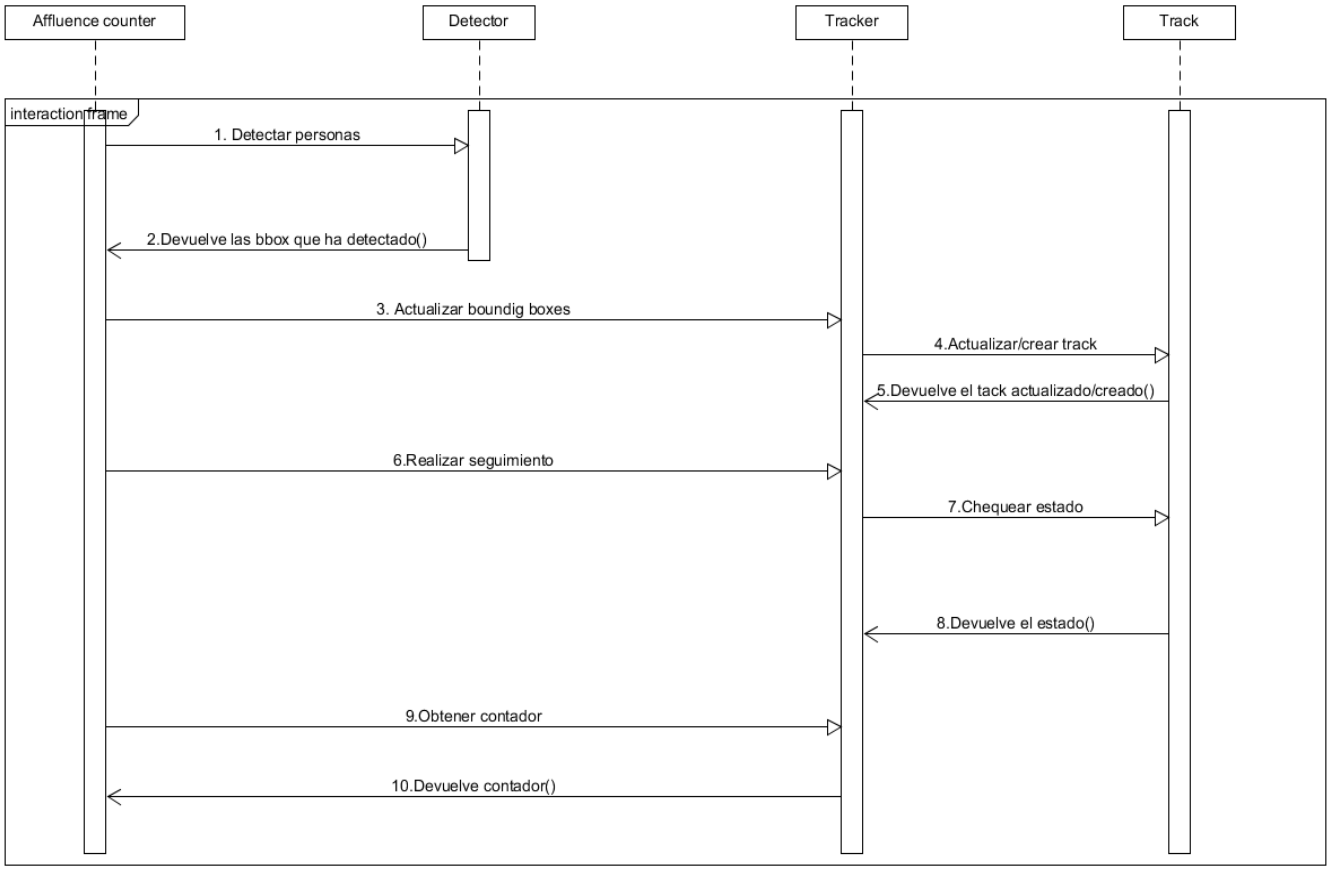

Figura 7. Diagrama de secuencias.

2.3 Diagrama de actividad

Se ha realizado un diagrama de actividad para los principales algoritmos desarrollados.

Diagrama de Actividad – Detector

El diagrama de actividad consiste en la detección de personas. Al instanciar la clase se le pasa las rutas de los pesos y los parámetros, inicializando así la red pre-entrenada, que esta basada en yolo. Al llamar al método “detect_image” pasandole una imagen, la red devolverá los bounding box que ha detectado y la confianza de que sea una persona. Si esta confianza supera un umbral establecido ese bbox se añade a la lista de bbox detectados, si no lo supera se descarta. Una vez finalizado el proceso, se devuelve la lista de los bbox detectados. Este proceso esta reflejado en la figura 8.

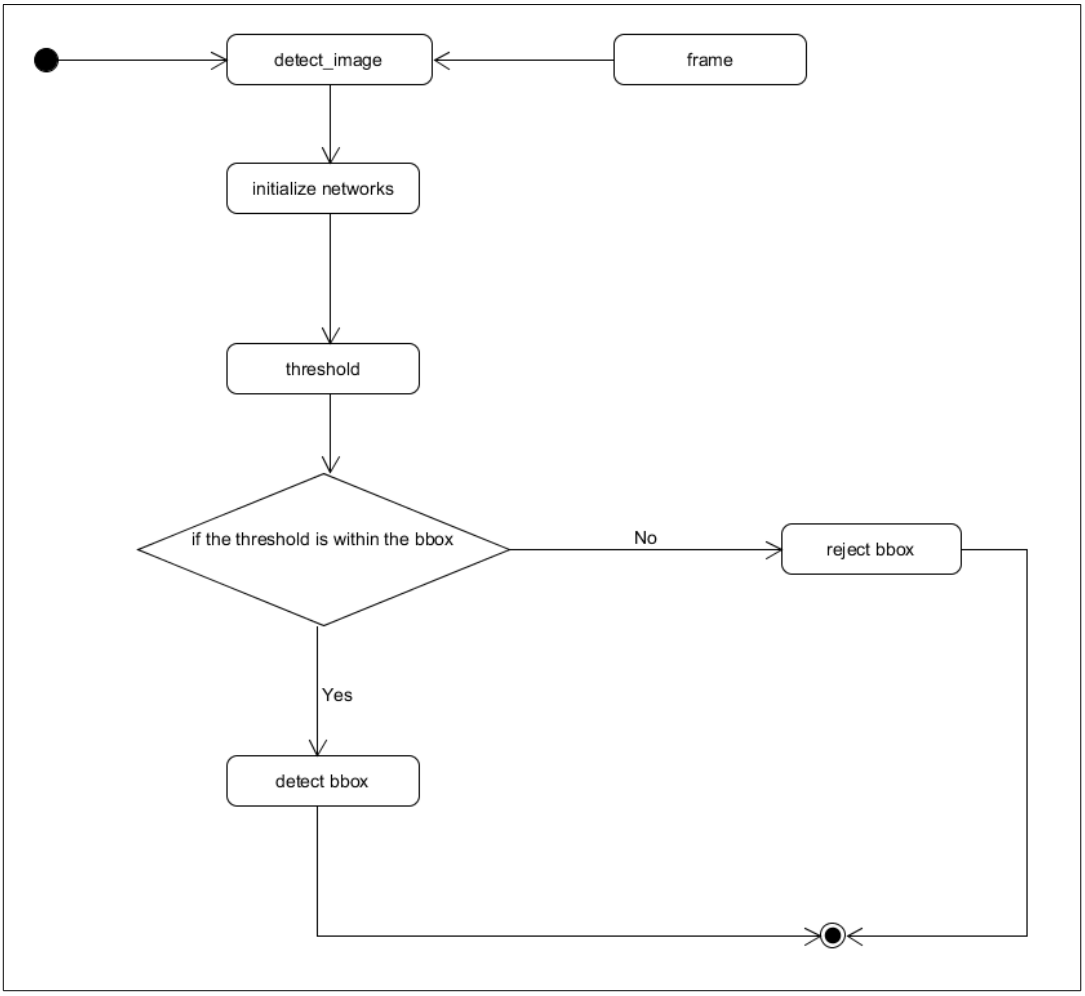

Figura 8. Diagrama de actividad de “detect_image”.

Diagrama de Actividad – Tracker

Se van a realizar 2 diagramas de actividades relacionadas con el seguimiento, ya que tiene dos funciones muy importantes claramente diferenciables.

El primer diagrama de actividad es sobre el método “update_trackers_by_dets” (Figura 9). Este método realiza la actualización (modificación de las posiciones) de los Tracks que tiene que gestionar el Tracker. Como puede verse en el diagrama, por cada bbox propuesto por el detector, se compara con todos los tracks anteriores y se comprueba si esa bbox puede emparejarse con uno de los tracks. En caso de que se pueda, se empareja y se pone de posición de ese track la propuesta por el detector. En caso de que no pueda emparejarse con ningún track, se crea uno nuevo con las

posiciones propuestas, y este track nuevo creado se añade a la lista de tracks a gestionar si cumple lo siguiente: que no es un track finalizado y que tiene un tamaño correcto.

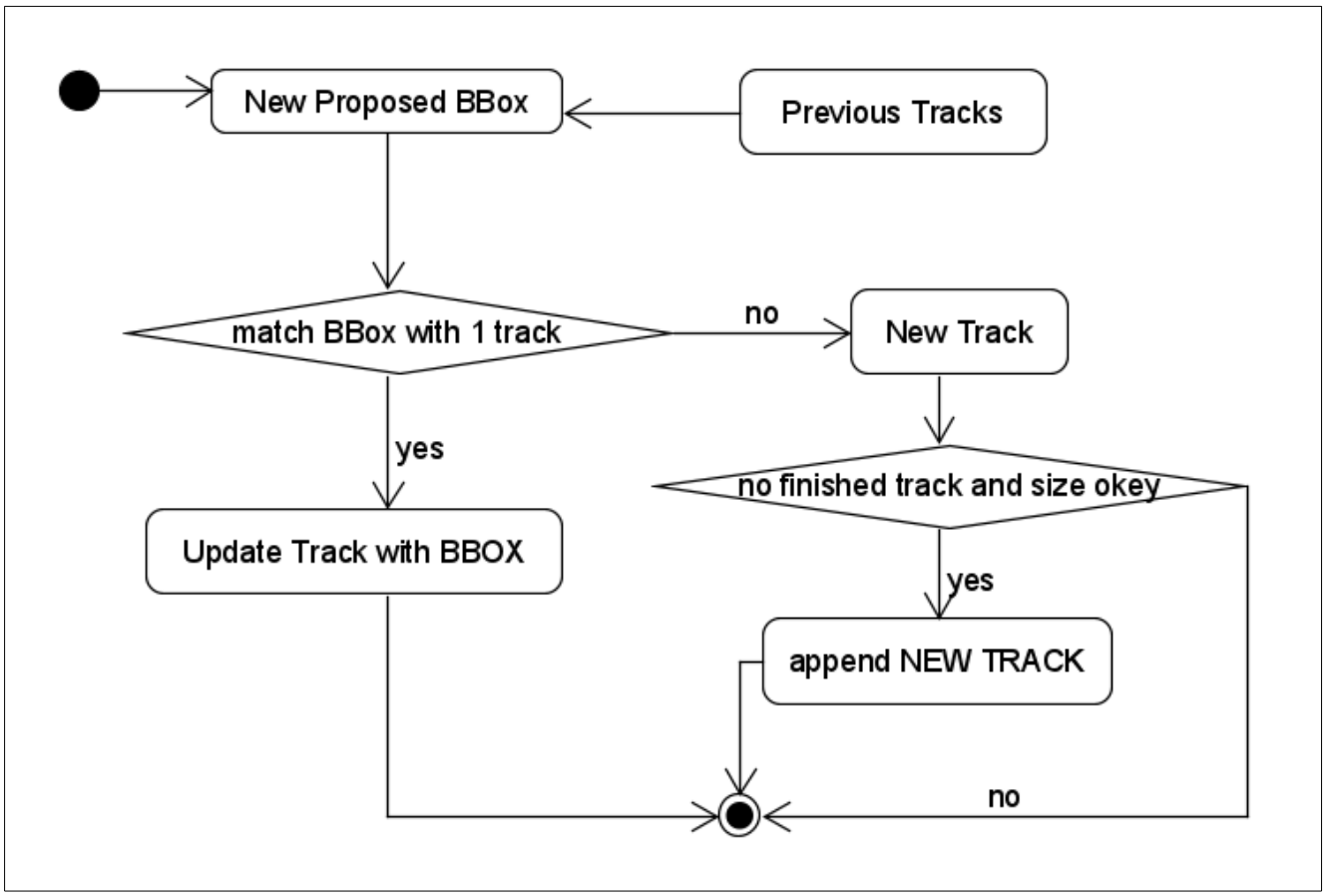

Figura 9. Diagrama de actividad de “update_trackers_by_dets”.

El siguiente diagrama de actividad es sobre el método “track” (Figura 10). El método track se encarga de recorrer todos los tracks para verificar su estado y actualizarlo en caso de ser necesario, es decir se encarga de comprobar por que posiciones ha pasado esa persona detectada desde el momento que se detectó en la escena, y también se encarga de eliminar los tracks que llevan cierto tiempo sin actividad y se consideran en “timeout”. Al analizar todos ellos, lista todos ellos en una lista.

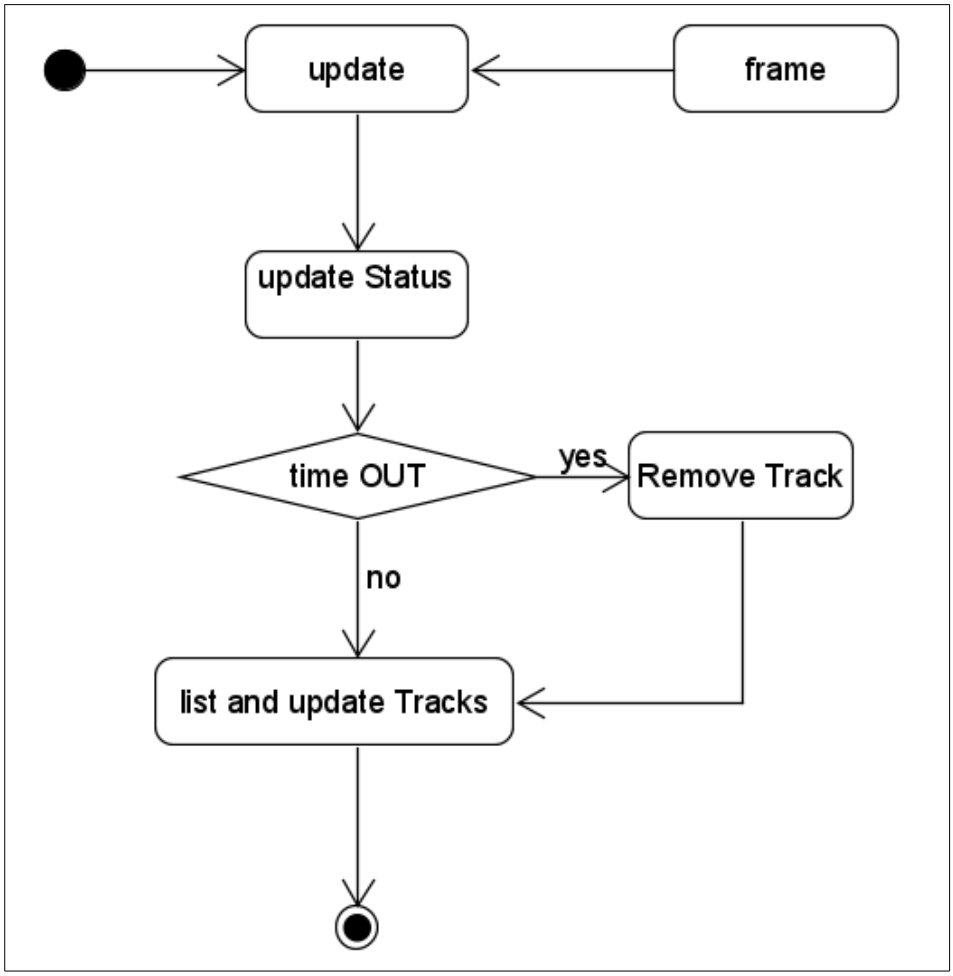

Figura 10. Diagrama de actividad de “track”.