**¿Cómo implementar la detección de objetos?**

**Objetivo:**

Implementar la detección de objetos.

**Introducción:**

Antes de iniciar con las formas de implementar la detección es necesario entender en qué consiste, por lo que dedique un tiempo a definirla básicamente podemos definir la detección de objetos como una tecnología asociada a la visión artificial y en la cual nosotros hacemos un procesamiento de imágenes con el propósito de detectar casos de objetos de cierta clase. (humanos, edificios, autos).

**¿Cómo funciona?**

Para identificar un objeto la imagen recibida se transforma a escala de grises y se procede a detectar los contornos. A lo largo de los contornos se trazan las normales que intersectarán en el centro.

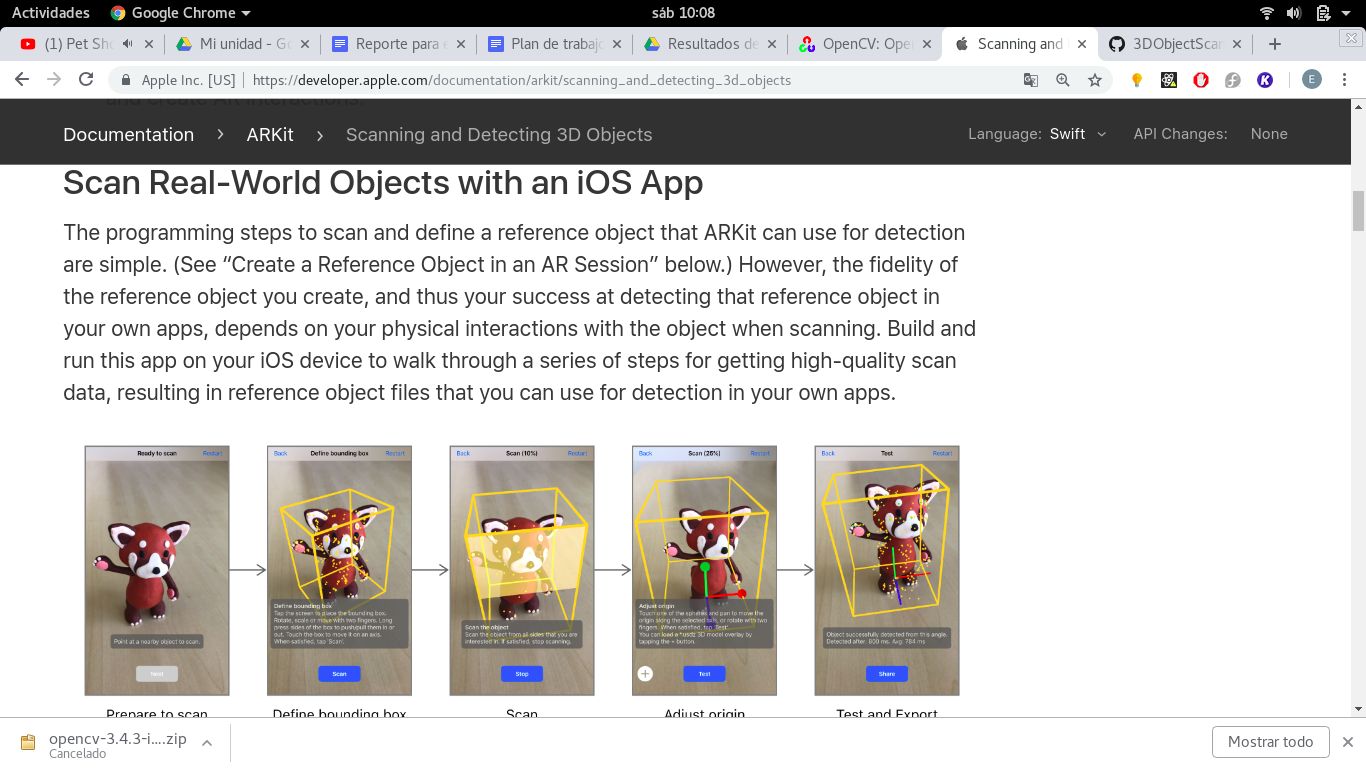
**Desarrollo:**

Investigando formas de implementar la detección de objetos encontré diversas formas de hacerlo algunas de ellas permiten mezclar la Inteligencia Artificial y la Realidad Aumentada.

A continuación hago una breve descripción de las opciones y mencionó algunas de las ventajas de dichas herramientas:

**ARKit 2:**

Existe una parte en la documentación de ARKIT en la cual específicamente nos dice

escaneo y detección de objetos 3D

<https://developer.apple.com/documentation/arkit>

**Escáner que detecta objetos de la vida real:**

<https://www.youtube.com/watch?v=FEqBW3cKF2k>

**Link del repositorio:**

<https://github.com/brianadvent/3DObjectScanningAndDetection/blob/master/FlowerShop.xcodeproj/project.pbxproj>

**Ventajas:**

Es una librería nativa para el desarrollo de realidad aumentada en IOS por lo que ya no sería necesario buscar una forma de implementar nuestro código.

**OpenCV**

Es una biblioteca destinada a la visión artificial, dentro de sus funciones se encuentra el reconocimiento de objetos, se puede usar C++, Java y Python.

**Documentación:**

<https://docs.opencv.org/3.4.3/d5/d54/group__objdetect.html>

**Demostración:**

[**https://www.youtube.com/watch?v=CppgV8inf7g**](https://www.youtube.com/watch?v=CppgV8inf7g)

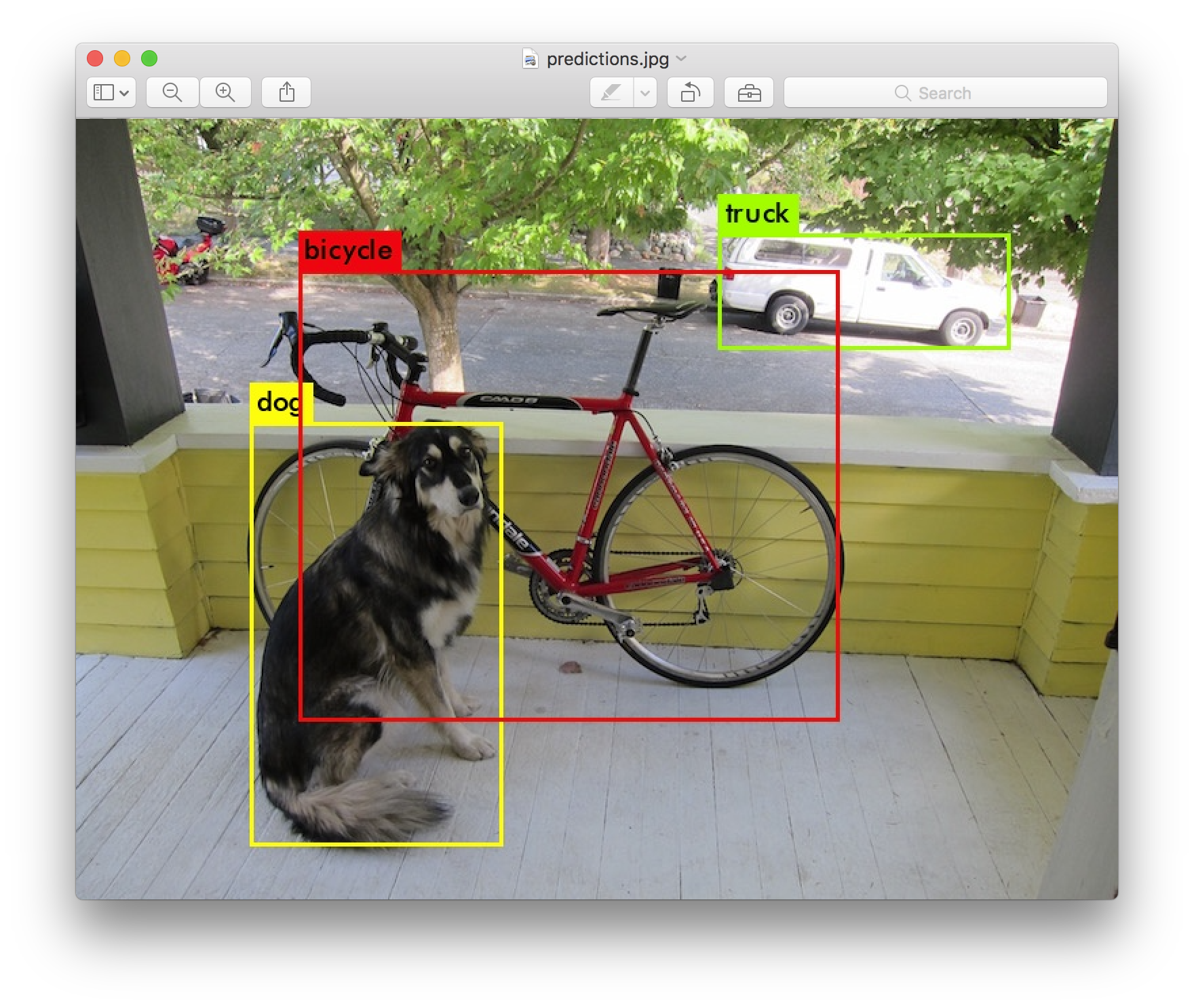
[**https://www.youtube.com/watch?v=HBwLlf-6UDQ&t=453s**](https://www.youtube.com/watch?v=HBwLlf-6UDQ&t=453s)

**Ventajas:**

Compatibilidad con linux, Mac OS y Windows

**Yolo:**

Es un sistemas que se encarga de la detección objetos en tiempo real:



**Documentación:**

[**https://pjreddie.com/darknet/yolo/**](https://pjreddie.com/darknet/yolo/)

**Código implementación:**

[**https://github.com/llSourcell/YOLO\_Object\_Detection**](https://github.com/llSourcell/YOLO_Object_Detection)

**Ventaja:**

Multiplataforma, python

**COCO: Common Objects in Context**

Es el más interesante debido a que cuando detecta un objeto encierra una superficie sobre ella como se puede ver a continuación:

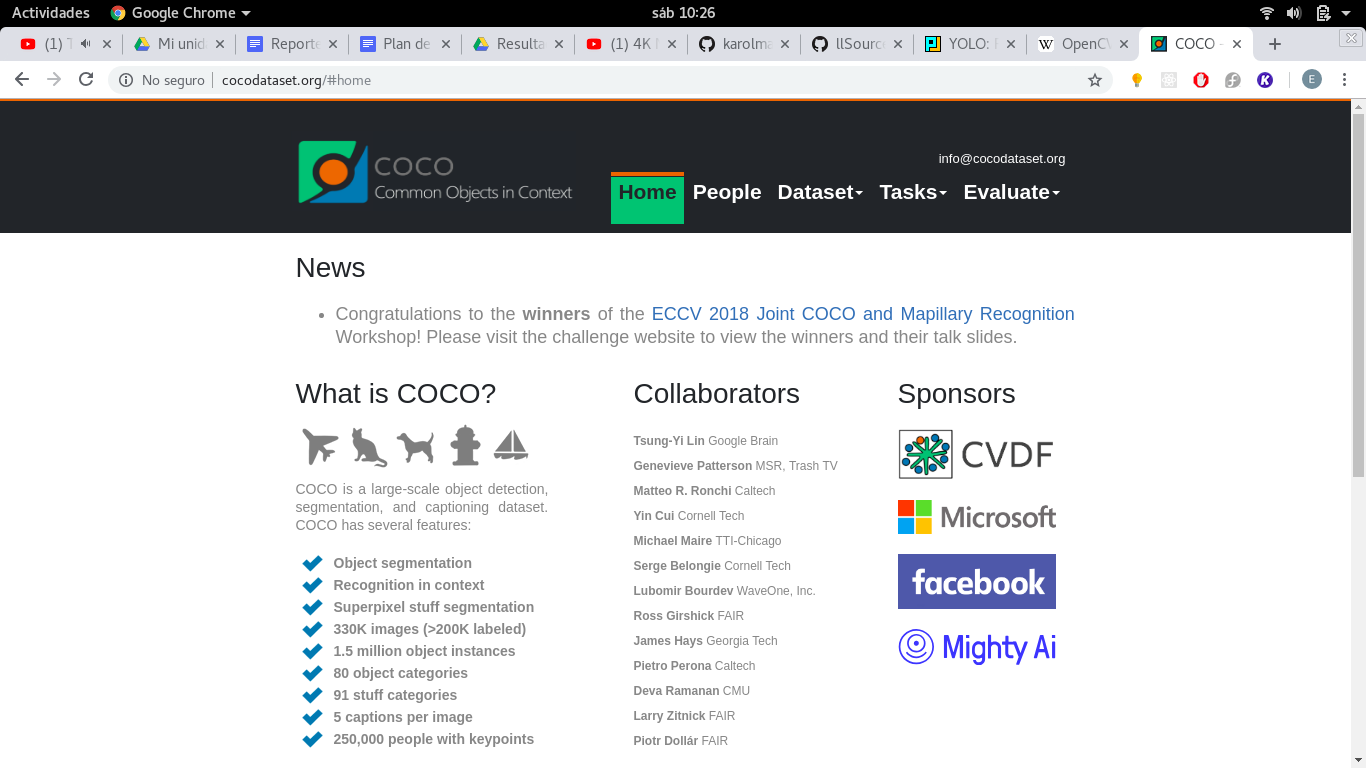


**Demostración**

[**https://www.youtube.com/watch?v=OOT3UIXZztE&t=1005s**](https://www.youtube.com/watch?v=OOT3UIXZztE&t=1005s)

**Código:**

[**https://github.com/cocodataset/cocoapi**](https://github.com/cocodataset/cocoapi)

****

**Ventajas:**

Api de Python

Fácil instalación

**Desventajas:**

No existe una documentación como tal.

**Object Detection con TensorFlow**

****

**Código del repositorio:**

[**https://github.com/tensorflow/models/tree/master/research/object\_detection**](https://github.com/tensorflow/models/tree/master/research/object_detection)

**Ventajas:**

Existe un amplia documentación al respecto.