|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | |  | |  | |
| SELECTOR CLAMP  Brazo mecánico que seleccionará las figuras que correspondan a la imagen o descripción introducida por el usuario a través de una aplicación móvil. |
| PROJECT SPRINT #4. DATE: 13 May 2020  Ridouan Elbachiri Irene González Fernández Javi Ledesma Pardo Pol Magre Moya |

Table of Contents

[**Project description**](#_heading=h.gjdgxs) **1**

[**Electronic components**](#_heading=h.30j0zll) **1**

[**Software Architecture**](#_heading=h.vnrdnv59gxnm) **2**

[Aplicación móvil](#_heading=h.c901r7qb4x5r) 3

[Módulo de detección de objetos](#_heading=h.vrijg2omp4ex) 4

[**Amazing contributions**](#_heading=h.l8fpj8t7rsi7) **5**

[**Extra components and 3D pieces**](#_heading=h.3znysh7) **5**

[**Simulation**](#_heading=h.mebe7h715bfj) **9**

[**Strategy for validation, testing and simulation (COMPULSORY, only the strategy, you do not need the tests to be done yet at this stage)**](#_heading=h.faauexmrhc94) **10**

[**References**](#_heading=h.q66uvppj8fuh) **11**

SELECTOR CLAMP

Brazo mecánico que seleccionará las figuras que correspondan a la imagen o descripción introducida por el usuario a través de una aplicación móvil.

Project description

Nuestro grupo ha desarrollado el proyecto llamado Selector Clamp. Este consiste en un brazo mecánico que cuenta con 4 grados de libertad para realizar sus movimientos de búsqueda de objetos y una cámara con la cual poder detectar su entorno.

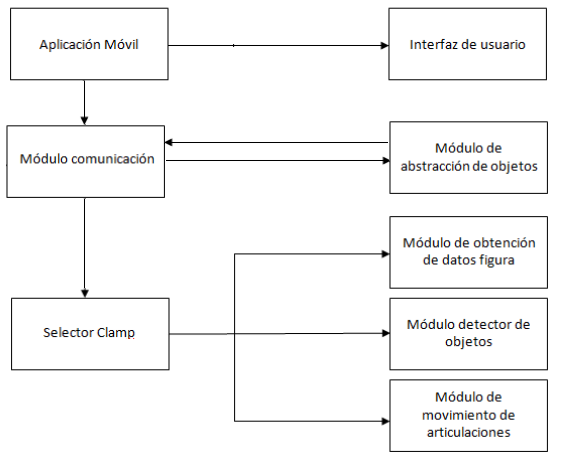
El usuario a través de una aplicación móvil podrá seleccionar los detalles de la búsqueda. La aplicación permitirá hacer búsquedas de forma y/o color de los objetos. A más a más, hay la opción de hacer varias fotografías vía smartphone, a continuación, el usuario seleccionará aquellas imágenes que mejor describan la forma del objeto que desea que el brazo busque. Entre la comunicación del usuario y Selector Clamp tendremos un servidor donde se almacenarán las órdenes dadas por el usuario.

Electronic Components

Esta es la lista de los componentes propuestos (98,68€):

* [RaspBerry PI 3 Model A+](https://es.farnell.com/raspberry-pi/rpi3-modap/placa-sbc-raspberry-pi-3-modelo/dp/2946269) (22,57€)
* [WebCam RPI](https://es.farnell.com/raspberry-pi/rpi-8mp-camera-board/raspberry-pi-camera-board-v2/dp/2510728) (22,57€) + [cable](https://tienda.bricogeek.com/accesorios-raspberry-pi/972-cable-para-camara-raspberry-pi-30cm.html) (2,50€)
* [Servomotor 360º](https://diotronic.com/servo-360%C2%A7-6v11kg-cm-4-8-7-2v_31875/) (10,41€)
* [3 Mini servos 180º](https://tienda.bricogeek.com/servomotores/1320-mini-servo-feetech-3-5kg-ft1117m-fb-con-feedback.html) (26,70€ (precio unitario 8,90€))
* [Controlador I2C](https://tienda.bricogeek.com/adafruit-feather-wings/1337-adafruit-prop-maker-featherwing-con-pines.html) (11,95€)
* [Bateria 9V](https://es.farnell.com/panasonic/6f22rel-1bp/battery-zinc-carbon-pp3/dp/1018880) (1,98€)

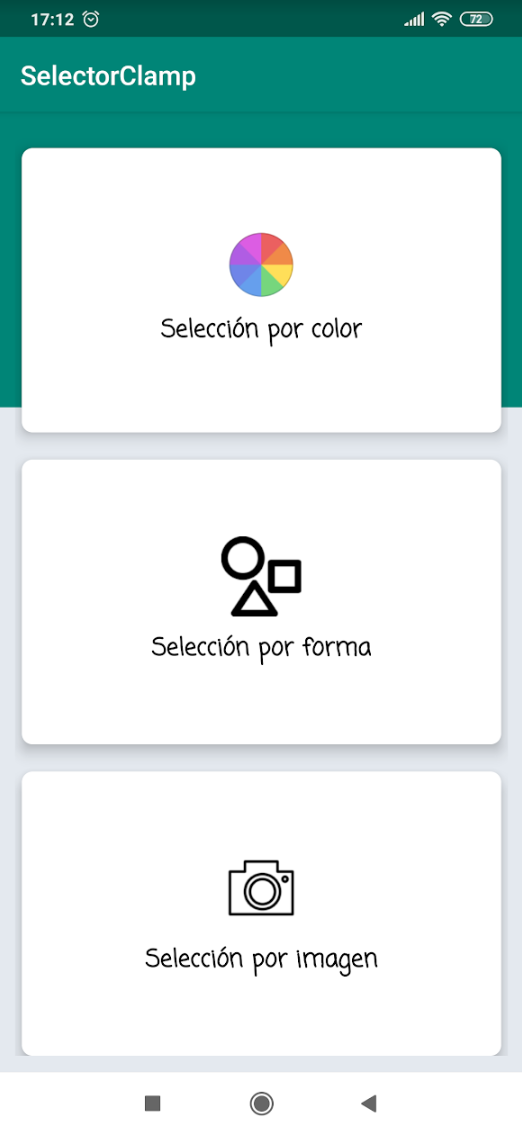
Software Architecture



Los módulos de nuestro proyecto se diferenciarán en:

* Módulos de la “Aplicación Móvil”.
  + *Interfaz de usuario*: diseñar una interfaz android en la que definamos las características de los objetos que seleccionará nuestro robot: color y/o forma.
* Módulo *de abstracción de objetos*: Clasificación de los objetos que contienen las imágenes captadas para enviarlos al módulo del robot.
* Módulo de comunicación: recibe de la aplicación, módulo de abstracción de objetos y enviar al módulo del robot.
* Módulos del robot “Selector Clamp”.
  + *Módulo de obtención de datos*: recibir los datos del usuario y tratarlos. De esta forma, el robot sabrá cuál es su nueva búsqueda.
  + *Módulo detector de objetos*: tratar los datos de nuestra RPI Camera para detectar los colores y/o formas descritos en la búsqueda.
  + *Módulo de movimiento de las articulaciones*: controlar el movimiento de todos los servos para recoger y almacenar en cajas cada tipo de objeto (ordenamiento según color y forma).

## Aplicación móvil



Uno de los módulos realizados hasta el momento es la aplicación mòbil que le servirá al usuario para comunicarse con nuestro robot. En la figura anterior se muestra la pantalla principal donde el usuario podrá introducir información sobre la figura que desea seleccionar ya sea indicando la forma, el color o directamente capturando la figura con una fotografía desde su dispositivo móvil. A continuación se muestra un ejemplo de cómo el usuario podría seleccionar una figura segun su color y/o forma.



Usamos un servicio de Google llamado Firebase el cual nos proporciona entre muchas otras cosas una base de datos a tiempo real que la usamos para guardar información de la figura introducida por el usuario. A continuación la placa del robot leerá la información de la base de datos para conocer las figuras a seleccionar.



Por otra parte el usuario desde la aplicación móvil puede capturar una imagen con la cámara y comunicar la al microcontrolador a través de Firebase.

## Módulo de detección de objetos

El módulo de detección de objetos utiliza una de momento una red neuronal entrenada con una base de datos con objetos aleatorios. Posteriormente cuando tengamos una base abundante de imágenes de los objetos que queremos que coja nuestro robot. La base de datos usada de momento es la COCO (Common Objects in Context).

Con las distintas redes que se han usado la que mejores resultados proporciona es la YOLO que está basada en una Faster R-CNN. Este método de detección de objetos nos funciona muy bien para imágenes en el entorno real, no funciona tan bien en el entorno simulado. Esto probablemente es debido a que las texturas y iluminaciones son menos realistas.

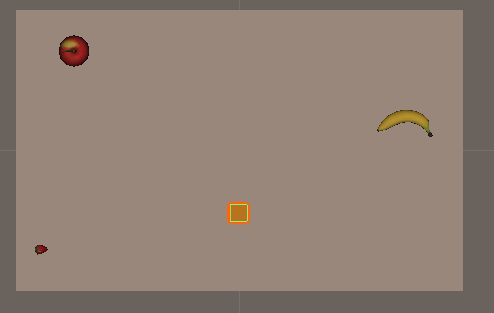
Los resultados que nos da igualmente son muy comprensibles ya que los fallos que nos da son entendibles con el ojo humano. Es decir, por ejemplo confunde una manzana con un balón de deporte según la posición pero realmente lo parece.

Esta red neuronal que obtenemos pre-entrenada, nos servirá tanto para la abstracción de objetos de los objetos obtenidos por la aplicación mòbil, como para detectar los objetos en la simulación.

Adjuntamos algunos resultados a continuación de imágenes reales y de la simulación:









En algunas nos detecta cuál es el objetos y en otras pasandole el nombre del objeto a buscar nos encuentra las distintas posiciones.

Amazing contributions

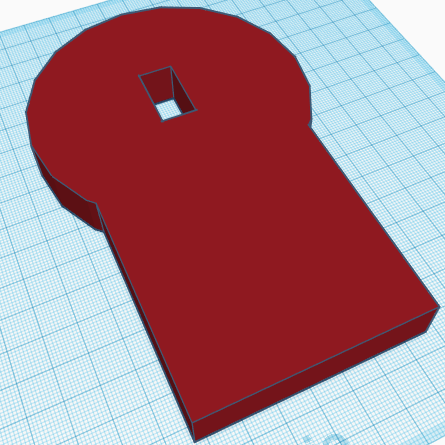
Nuestro proyecto implementa módulos bastante diferentes entre sí, por una banda existe la implementación de una aplicación en Android, con su correspondiente interfaz de usuario y una conexión a una base de datos en tiempo real. Por otra banda módulos como el de detección de objetos que usa visión por computador que permiten que el robot compare la imagen que ha tomado el usuario con la figura que detecta por su cámara y por lo tanto seleccione la figura correcta.

El proyecto se inspira a una nota superior a 7.5, ya que cumple con la funcionalidad básica de un robot autónomo. Por otra parte se implementa con la visión por computador, además se ha implementado una aplicación móvil propia, para interactuar con el robo desde cualquier lugar.

Extra components and 3D pieces

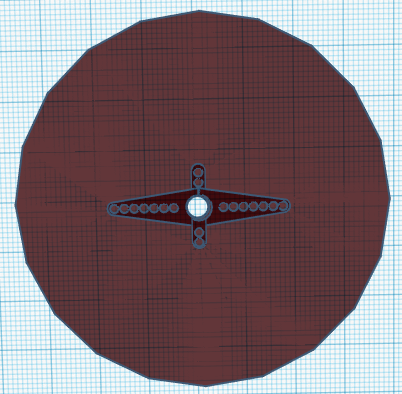
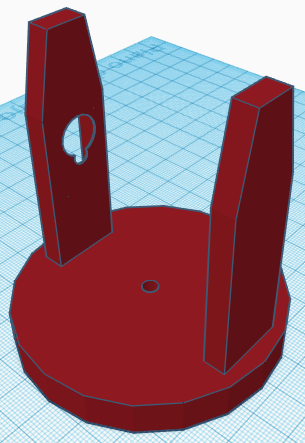
* Base

La funcionalidad de esta parte del robot es la de soporte del brazo y será donde guardaremos nuestra placa. Además, cuenta con un agujero en el que poner nuestro servo motor de 360º. Así pues, definimos esta parte como la base u hombro del brazo mecánico. El movimiento que realiza es horizontal.



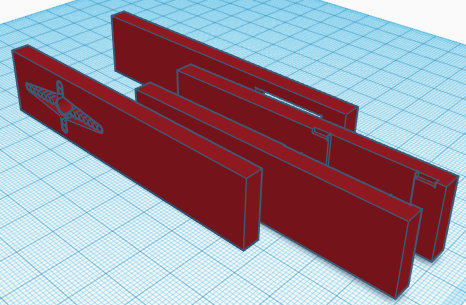
* Brazo I

En esta parte hemos desarrollado la primera parte del brazo. La parte inferior cuenta con un agujero que se utiliza para atornillar esta parte con el servomotor de la base. En la parte superior contamos con un agujero para poner nuestro segundo servo, su movimiento es vertical.



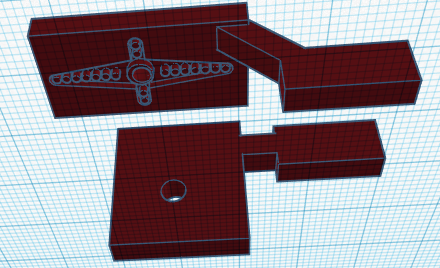
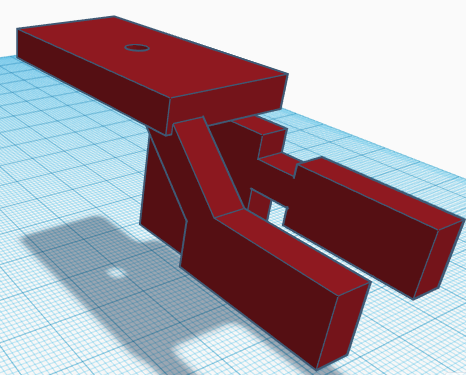
* Brazo II y antebrazo

La funcionalidad de esta parte consiste en la unión del brazo y el antebrazo. En la siguiente imagen tenemos en la parte más alejada de la imagen dos elementos que forman la segunda parte del brazo y a la parte frontal otros dos que forman la parte del antebrazo. La segunda parte del brazo está atornillada a la primera del brazo y tenemos un servomotor que está atornillado con el antebrazo (Codo). Su movimiento es vertical.

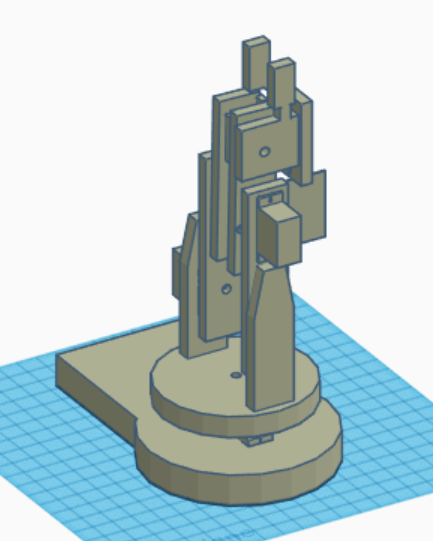
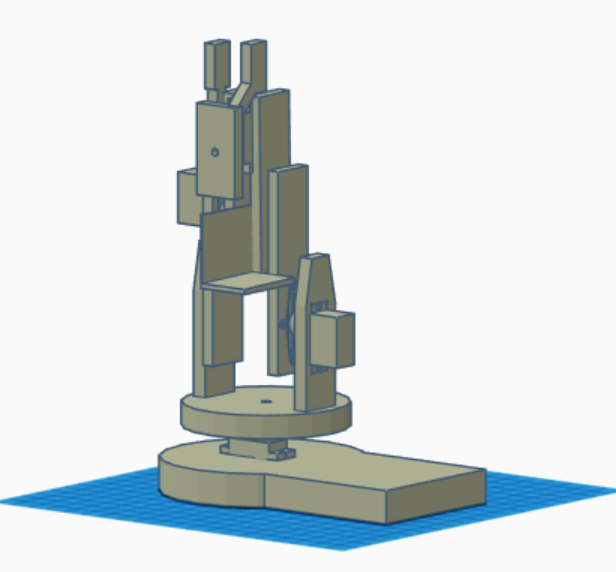


* Pinza

Es la última componente diseñado en 3D que tiene nuestro robot. Cuenta con dos elementos que juntos forman la pinza con la que cogeremos los objetos. El componente de la izquierda irá atornillado a un servo y será la parte de la pinza que se moverá. Mientras que el componente de la derecha es una parte fija atornillada al antebrazo. Su movimiento es horizontal.

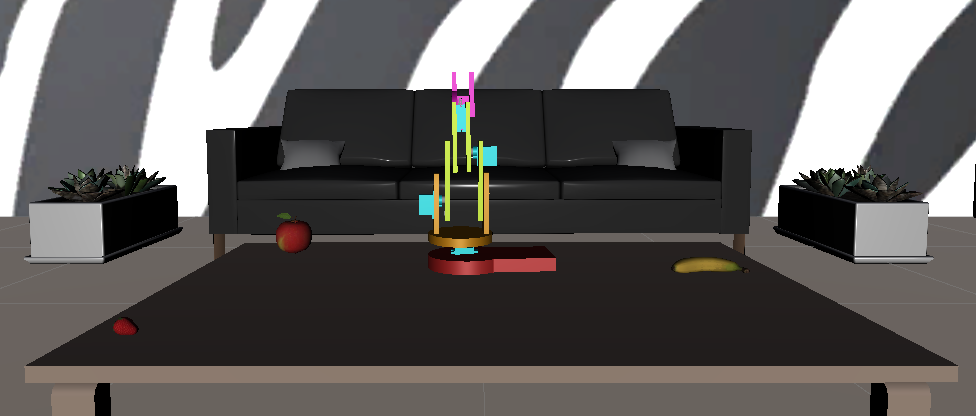


* Robot entero



Simulation

La simulación se ha llevado a cabo en Unity, creando un entorno lo más realista posible. Donde encontramos el Selector Clamp en el centro y está rodeado de objetos que deberá ser capaz de detectar y agarrar.



El modelo 3D del robot ha sido importado del modelo que ya tenemos hecho en Tinkercad, utilizando la extensión .obj que es la única compatible con ambas aplicaciones.

El entorno debe tener distintas texturas y formas, además también contará con la problemática de la luz (muy distinta a la natural).

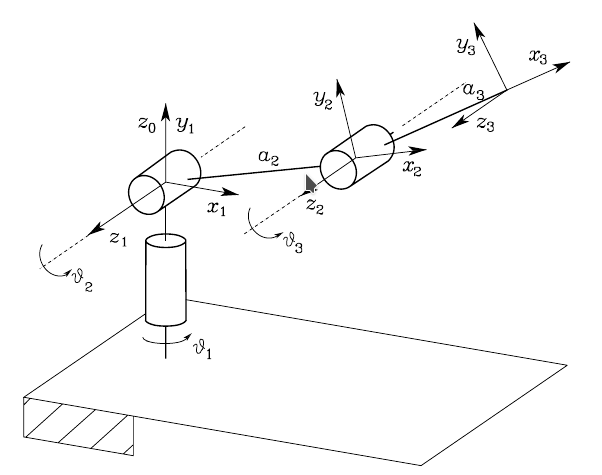
Se ha desarrollado una aplicación móvil en Java, en el entorno de Android Studio para que el usuario pueda interactuar con el robot. Cuando el usuario introduce información sobre el objeto que quiere buscar, ya sea indicando la forma, el color o subiendo una imagen del mismo, la aplicación sube esta información a una base de datos de Firebase.

Para comunicar el módulo de la aplicación con los módulos de detección y del robot, utilizamos una comunicación por base de datos, de esta forma facilita la integración en la parte de simulación en Unity. Nos basamos en una comunicación cliente servidor entre la aplicación y el módulo de comunicación, este módulo está implementado en python, donde la aplicación envía una petición a la base de datos, la misma será comunicada al módulo de detección para clasificar en en caso de que sea una imagen, de la misma manera se comunica la respuesta con el fin de ser leída por el módulo de robot para realizar la simulación.

El robot cuenta con una cámara que detecta mediante un algoritmo Faster R-CNN aquellos objetos parecidos a los que el usuario ha definido en su búsqueda.

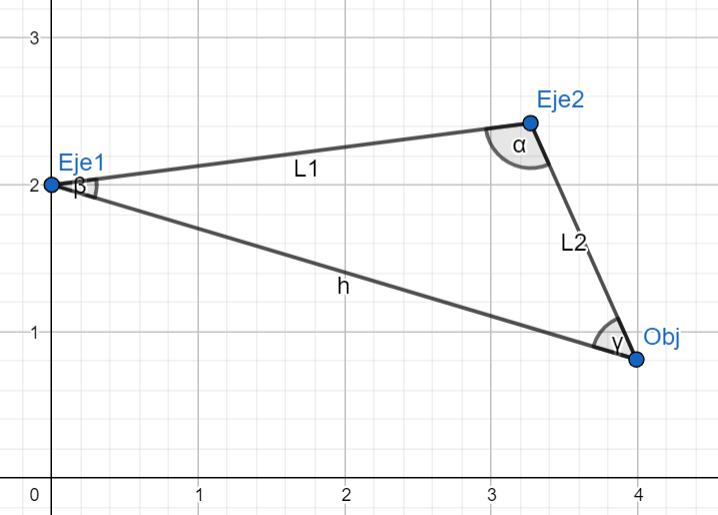
Cuando el robot esté encendido dará una vuelta de reconocimiento para mapear los objetos de su entorno. Debemos recordar que el servo motor de la base es el único que puede girar 360º así que podrá realizar el giro sobre sí mismo sin ningún problema. Durante este giro realizará mediante la red neuronal Yolo la detección de los objetos del tipo seleccionado y los irá añadiendo a una lista para su posterior clasificación.

Para coger los objetos con sus coordenadas en la lista antes mencionada, hemos hecho una función para calcular la cinemática inversa de nuestro robot. La representación funcional sin tener en cuenta la pinza de nuestro robot es la siguiente:



Para realiza la cinemática inversa, primeramente determinamos el . Donde el representa el argumento principal del número complejo entre .

Una vez definido este ángulo, formamos un triángulo con nuestro objeto a coger como un vértice, las coordenadas del eje 1 otro vértice y las coordenadas del eje 2 el otro.



Donde , ya que en Unity tenemos los ejes y girados. La distáncia es la longitud del antebrazo del robot, es la longitud del brazo del robot y es la altura de la base. La distancia del objeto al eje inferior sería en la coordenada .

Las coordenadas del objeto son relativas al robot.

Y el cálculo de los ángulos se hace con el teorema del coseno. Y se obtiene el siguiente resultado:

Y así definimos todos los ejes relativos al movimiento, ya que la pinza solo se abre o cierra no define la posición del robot para coger un objeto.

Una vez en la posición del objeto, lo sujetamos y lo desplazamos hasta la posición donde los depositamos. Esto se repite hasta que no haya objetos que buscar en la lista. En tal caso, el robot volverá a su posición inicial.



Strategy for validation, testing and simulation (COMPULSORY, only the strategy, you do not need the tests to be done yet at this stage)

La estrategia utilizada se basa en mitigar los posibles riesgos que pueden aparecer durante el desarrollo del proyecto. Para ello establecimos un plan de contingencias.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk #** | **Description** | **Probability** (High/Medium/Low) | **Impact** (High/Medium/Low**)** | **Contingency plan** |
| 1 | No implementar la aplicación. | Low | Medium | Pasar las imagen de forma manual. |
| 2 | No disponer de los componentes hardware. | Low | High | Realizar la simulación de la arquitectura Software de manera separada, utilizando Unity y tinkerCad. |
| 3 | Dificultad para distinguir figuras con la información dada. | Medium | High | Siempre cabe la posibilidad de hacer las figuras muy diferentes entre sí y colores muy llamativos. |
| 4 | Límite computación de la placa. | Medium | Medium | Establecer límite de imágenes que se le pasa a la placa. Y usar servicios del Cloud. |
| 5 | No detectar objetos en el entorno de la simulación | Medium | Medium | Se podría probar de reentrenar la red con imágenes virtuales también. Y si no solo se simulará el movimiento del robot en el entorno virtual. |

References

This project has been inspired by the following Internet projects:

Brazo mecánico de 4 ejes:

<https://www.instructables.com/id/4-Axis-Robot-Arm-DIY/>

Base de datos en tiempo real, Firebase en Android:

<https://firebase.google.com/docs/android/setup?hl=es-419>

Desarrollo Aplicaciones con Android Studio:

<https://developer.android.com/studio/intro?hl=es-419>

Subir imagenes al Storage de Firebase:

<https://www.youtube.com/watch?v=MfCiiTEwt3g&list=PLrnPJCHvNZuB_7nB5QD-4bNg6tpdEUImQ>

Integración Unity con Firebase:

<https://firebase.google.com/docs/admin/setup?hl=es-419#c>

Interactuar con Firebase desde Unity

<https://firebase.google.com/docs/database/web/read-and-write>