

ANEXO 1. Manuales

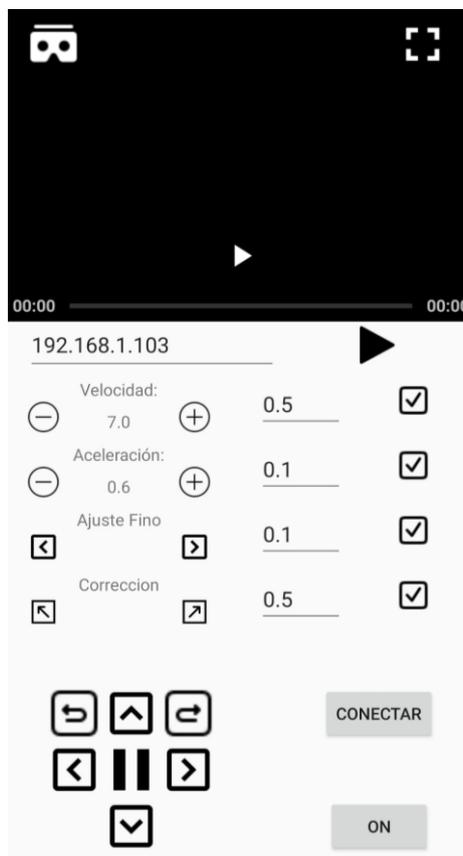
ANEXO 1.1. Manual de instalación

Para instalar y configurar correctamente el dispositivo móvil para el correcto uso de la aplicación y el control del sistema móvil se deben seguir los siguientes pasos:

1. Instalar la aplicación lo cual se puede hacer instalando el fichero app-debug.apk en la carpeta apk del proyecto Para ello, deberemos aceptar la opción de “Orígenes desconocidos” que encontraremos en ajustes del dispositivo móvil.
2. Una vez instalada la aplicación, para el control del sistema móvil se debe entrar en ajustes de Bluetooth del dispositivo móvil, buscar el equipo “VideoCoche” y pulsar sobre él para guardarlo y así que la aplicación pueda buscarlo.
3. Tras esto, se debe encender el mando VRBOX pulsando un par de segundos el botón ON, tras esto, se buscará el dispositivo VRBOX y se emparejará el dispositivo móvil con él para poder usarlo.

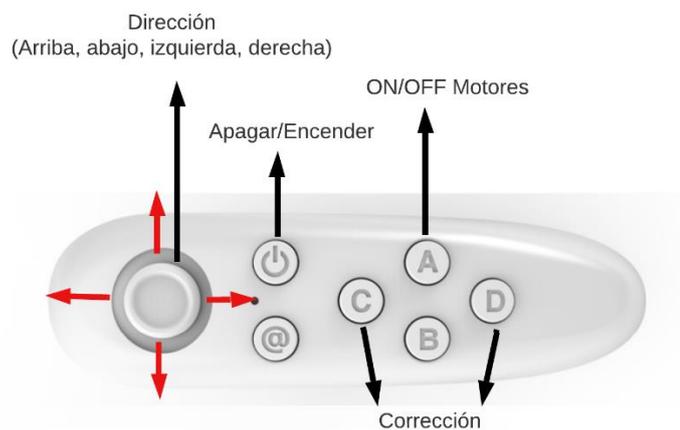
ANEXO 1.2. Guía de uso

Al instalar la aplicación, si no se ha activado el Bluetooth, la aplicación pedirá activar el Bluetooth. Al aceptar, se debe pulsar en el botón “Conectar” para que el móvil establezca una sesión Bluetooth con el sistema móvil. Tras esto se tienen varios controles que se pueden visualizar en la siguiente imagen:



- El botón On/Off para encender o apagar los motores de las ruedas.
- También tenemos las flechas direccionales para indicarle al sistema móvil qué dirección debe seguir. Si el sistema móvil está parado, hasta que no se pulsa una de las flechas no se pondrá en movimiento.
- El botón Start/Stop, que es el botón con el icono “Play” situado en el medio de los botones direccionales, que permite iniciar o bien parar el sistema móvil (en este caso, la parada se realiza de manera progresiva).
- Los selectores que permiten ajustar tanto la velocidad a la que queremos que vaya el sistema móvil como la aceleración, es decir, lo rápido que realizan los cambios progresivos de velocidad.
- El selector de Ajuste Fino para realizar los pequeños ajustes de trayectoria cuando el sistema móvil se desvía.
- El selector de Corrección para realizar cambios en la trayectoria del robot y así redirigirlo a que tome la dirección adecuada
- Los editores de texto que permiten modificar diversas variables, con su botón “Aceptar” al lado (botones con un icono de) , que sirven para modificar las siguientes variables:
 - *Ajuste_fino*
 - *Tiempo_giro*
 - El salto que dan los botones + y – de los selectores

Además, también se puede controlar al sistema móvil con un mando VRBOX, con el cual debe usar el Joystick para manejarlo, si se mantiene mueve el joystick hacia una dirección, el sistema móvil se moverá en esa dirección. Al dejar de mantener el joystick en esa dirección, el sistema móvil frenará. También el mando tiene un botón para encender o apagar los motores. Además, los botones C y D del manado, sirven para aplicar una corrección a la trayectoria del sistema móvil para poder enderezarlo y que tome la dirección deseada por el usuario (equivalen a las flechas direccionales de la aplicación táctil):

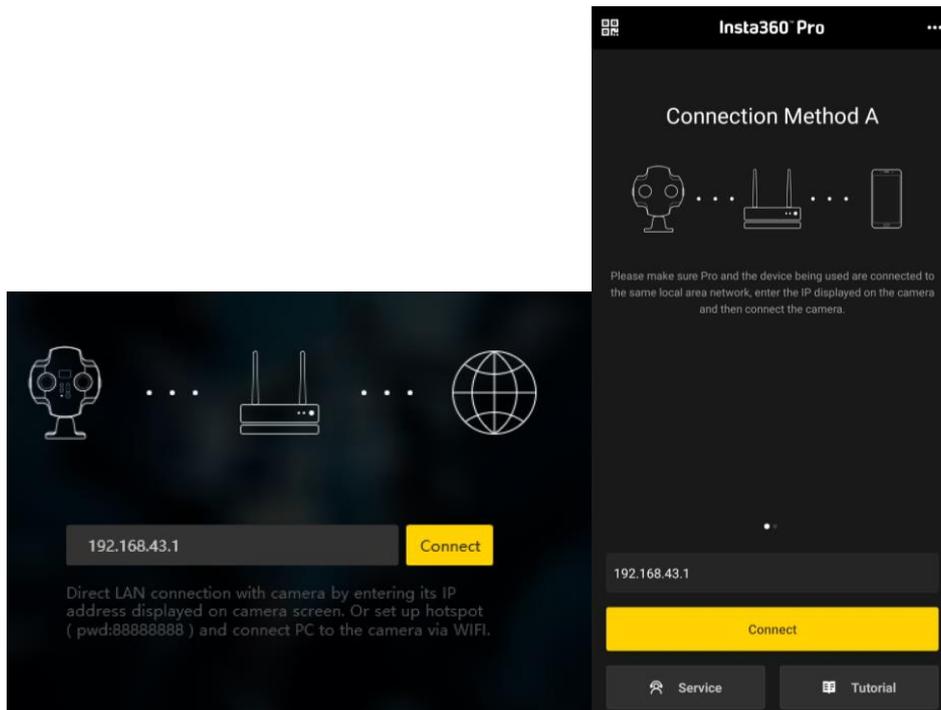


ANEXO 1.3. Guía para grabar vídeo 360

Para grabar el contenido en directo de la cámara Insta 360 Pro utilizando la aplicación móvil o la aplicación para PC, seguiremos los siguientes pasos.

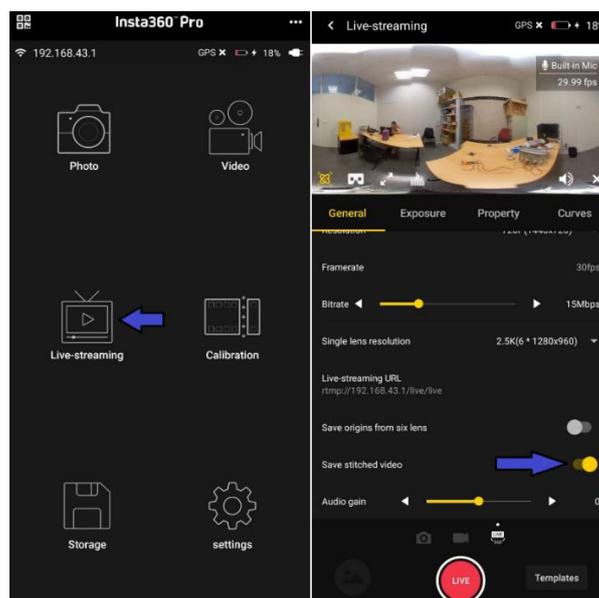
Para acceder a la cámara desde la aplicación móvil o el programa de PC, debemos conectarnos a la misma red WiFi que la cámara, o en el caso que ésta haga de punto de acceso debemos

conectarnos a su red WiFi Insta360-Pro-YCGQM8-OSC. Tras ello, en la pantalla, introduciremos la dirección IP de la cámara, la cual se muestra en la pantalla de la propia cámara.

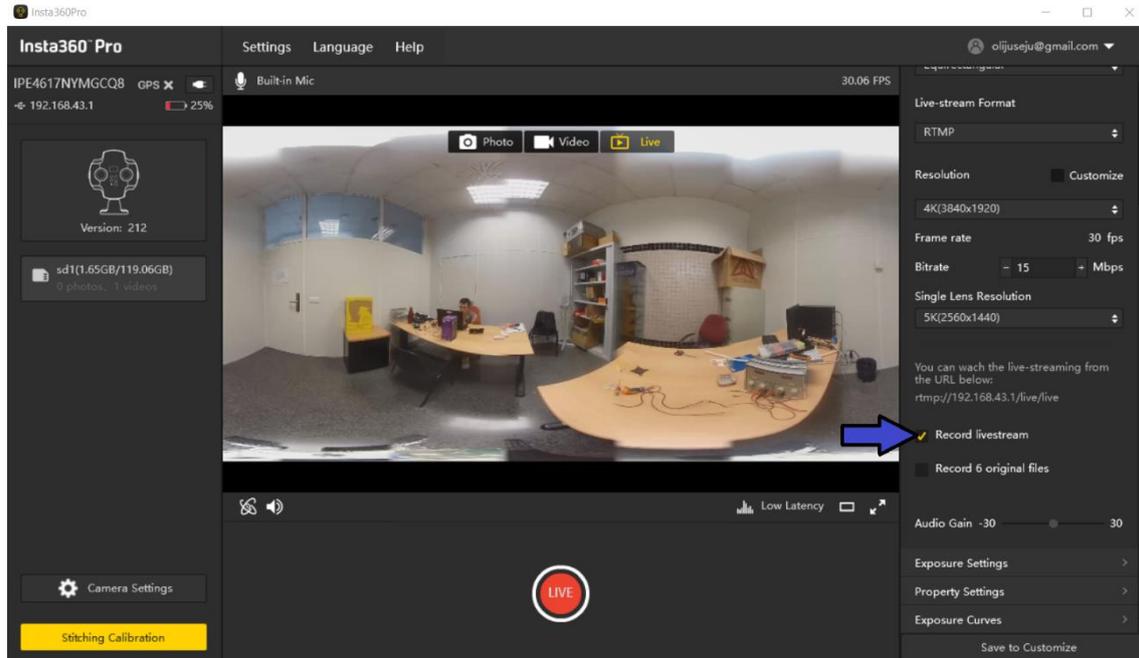


Si estamos utilizando la aplicación móvil, descargaremos la aplicación desde Play Store o desde el enlace proporcionado en la página oficial de Insta 360 [Ref 23] e instalaremos la aplicación en nuestro dispositivo móvil. Desde esta página también podemos descargar el programa para PC, si así lo deseamos

En el caso de la aplicación móvil, una vez dentro de la aplicación, en la opción “Live”, encontraremos la opción "Save Stitched video" en la pantalla de la aplicación. Pulsaremos en esta opción y después pulsaremos el botón “Live” para guardar el video grabado en nuestro dispositivo móvil. De esta manera, el video grabado se guardará en la memoria SD de la cámara para que podamos acceder a él más tarde.



Si estamos utilizando la aplicación para PC, una vez dentro del programa, seleccionaremos arriba la opción "Live" y en el menú de la derecha encontraremos la opción "Record Livestream". Haremos clic en esta opción y después pulsaremos el botón "Live" para comenzar a grabar el video en directo. La aplicación guardará el video grabado en la memoria SD de la cámara para que podamos acceder a él más tarde.



Además, desde la aplicación o programa para PC podemos configurar opciones como la calidad a la que se emite el vídeo o el Bitrate. Es importante que la calidad seleccionada no sea la máxima ya que el vídeo en directo no se verá fluido en el reproductor Exoplayer.

Es importante tener en cuenta que la opción para grabar el video puede variar dependiendo de si estamos utilizando la aplicación móvil o la aplicación para PC. La opción "Save Stitched video" está disponible en la aplicación móvil, mientras que la opción "Record Livestream" está disponible en la aplicación para PC.

Recordemos que las grabaciones se guardarán en la memoria SD de nuestra cámara Insta 360 Pro, y podremos acceder a estas grabaciones a través de la aplicación móvil o el programa para PC según nuestra elección.

Si solamente queremos transmitir el vídeo en directo para visualizarlo en la aplicación móvil, quitaremos la opción "Save Stitched video" o "Record Livestream" del programa o aplicación móvil y pulsaremos en el botón Live

ANEXO 2. Problemas de implementación resueltos durante el TFG

Primer problema: Cambio de dirección atrás-adelante

El primer problema encontrado fue el de perfeccionar el código de Arduino relacionado con el controlador implementado en la función *ActualizaMot()* para que la aceleración de las ruedas se diera de forma progresiva. También ha sido un problema el hecho de que los 4 motores deban cambiar su aceleración y velocidad a la vez, pero de manera independiente, por lo que, al cambiar el movimiento de atrás hacia delante, si el sistema móvil estaba realizando un giro, las velocidades de las 4 ruedas no son exactamente iguales, por lo que unas empezaban a girar marcha atrás antes que las otras, lo cual producía un giro del sistema móvil sobre su eje que no estaba controlado. Se ha podido arreglar controlando cuándo las 4 ruedas del sistema móvil están paradas. De esta forma, por ejemplo, si el sistema móvil está girando a la derecha y el usuario pulsa el botón de marcha atrás, el sistema móvil lo primero que hará es frenar sus ruedas de forma progresiva y sólo cuando todas las ruedas están paradas, realizará el movimiento marcha atrás.

Segundo problema: Nuevo cableado del sistema móvil

El segundo problema está relacionado con la parte física del sistema móvil. Cuando se me entregó el sistema móvil, el cableado estaba a medio conectar, con cables rotos y estropeados, que se han tenido que cambiar. Además, en los documentos que se habían creado anteriormente para el desarrollo de este sistema móvil había diversos errores en cuanto a correspondencia de pines y voltajes. Ahora el cableado del sistema móvil es mucho más limpio, con cables nuevos y crimpados con materiales nuevos, además de un cambio de microcontrolador para que sea más eficiente. Sumado a esto se ha creado una PCB para que el cableado resulte mucho más sencillo y limpio, la cual ha sido desarrollada con el apoyo de profesores y técnicos del Campus utilizando el programa Altium.

Tercer problema: Desconexión del Bluetooth al cambiar la orientación

Otro problema destacable es referente al uso de Bluetooth en la aplicación, ya que al cambiar la orientación de la pantalla, el socket Bluetooth se cierra debido a que se llama a la función *onDestroy()*. Para solucionarlo, se controla si la conexión estaba establecida antes de cambiar la orientación, por medio de un booleano que se pasa por medio de la función *onSaveInstanceState()*. Por lo tanto, si en *onCreate()* se recibe este booleano como un true, significa que se ha cambiado la orientación de la aplicación en medio de la conexión, por lo que se vuelve a reconectar automáticamente.

Cuarto problema: Visor estereoscópico

Otro problema que me llevó varios días fue intentar utilizar el SDK de Google VR para reproducir el video 360 por medio de código, como *VideoView* o *PanoramaView*, lo cual no ha sido posible ya que esta SDK ya no se utiliza. La solución parecía ser la nueva NDK de Google Cardboard, pero la implementación en una aplicación para capturar un flujo de video era demasiado compleja. También se optó por utilizar un WebView para ello, creando una web en local que disponga de un reproductor Javascript llamado *Video.js*, que reciba el flujo RTMP de la cámara Insta360 y reproduzca el vídeo en 360, sin embargo, el reproductor de vídeo utilizado no admitía transmisión RTMP, tan solo HLS, el cual provocaba una alta latencia de 15 a 20 segundos en la transmisión de vídeo en directo. Se probaron diversas librerías como Clappr, FlowPlayer o Flashponer, pero ninguna de éstas admitía flujo RTMP.

También se utilizó la plataforma de YouTube, subiendo los streamings de la cámara a un canal de YouTube, el cual era accesible desde la aplicación móvil, de esta forma se podía visualizar el flujo RTMP retransmitido en la cámara por medio del reproductor de YouTube de Android. Sin embargo, este reproductor no cuenta con funcionalidades de pantalla completa o de visor estereoscópico a menos que el usuario salga de la aplicación y entre en la aplicación de YouTube, lo cual impide al usuario controlar el sistema móvil. Además, se necesita de conexión a internet en todo momento para visualizar el video en directo a través de YouTube, por lo tanto, cuando el usuario se conectase a la cámara a través de la red WiFi de ésta, no podría visualizar el streaming.

Para crear el visor estereoscópico y poder usar el dispositivo HMD en para ver el video, se intentó también utilizar un reproductor Exoplayer que ocupase media pantalla y un objeto TextureView que replicase el Bitmap de éste, es decir, lo que se muestra en el reproductor. Sin embargo, copiar el Bitmap del reproductor en cada instante y mostrarlo en el TextureView era demasiada carga computacional y la aplicación no podía hacerlo. El resultado es que se copiaba un frame del video al TextureView muy de vez en cuando y muchas veces solo se copiaba una pantalla en negro.

Quinto problema: Uso del mando VRBOX

El tema del control del sistema móvil mediante un mando VRBOX también supuso un problema, ya que, para ello, tuve que hacer que la app detectase inputs de entrada, es decir, mensajes del tipo *KeyEvent*, recibidos en las funciones *OnKeyUp* y *OnKeyDown*. Al mantener pulsado un botón o desplazar el joystick hacia delante durante más de 1 segundo, se mandan muchos mensajes del tipo *OnKeyDown* que saturaban la ESP32, la solución fue aplicar un retraso (*postDelayed*) a un manejador (*handler*), en la función *EsperaYMensaje*, para que no pueda enviar más de un mensaje del tipo que sea cada 1.5 segundos.

Como la comunicación vía Bluetooth entre la app y el ESP32 era algo lenta, se tuvo que limitar los mensajes que se mandaban cada periodo de tiempo, por lo que el usuario no puede mandar más de un mensaje cada 1500 milisegundos para no saturar el ESP32, esto se consiguió implementando un contador de tiempo que se ejecuta cada vez que se envía cualquier mensaje desde la aplicación móvil. Este contador de tiempo activa un flag que permite el intercambio de mensajes. Cada vez que se envía un mensaje, este *flag* se desactiva

Sexto problema: Detectar desconexiones desde la aplicación móvil

Otro problema fue cómo parar el sistema móvil si se pierde la conexión bluetooth con el dispositivo del usuario. Para ello, se detecta en la aplicación cuándo se pierde la conexión y se reinician las variables a su estado inicial. Esto se hace con un *BroadcastReceiver* llamado *BLReceiver*, que recibe un mensaje cuando se pierde la conexión.

En el código de Arduino se tuvo que implementar un *callback* en el *SerialBT* para que reciba un mensaje cuando se pierde la conexión, lo que actualiza la aceleración al valor por defecto y reinicia todas las variables poniendo *Stop=true*.

Séptimo problema: Desviaciones en la trayectoria del sistema móvil

Uno de los problemas más grandes con respecto al sistema móvil tiene que ver con mantener la dirección de movimiento de éste. Pese a que los 4 motores alcanzaban una misma velocidad (medida gracias al encoder) de manera progresiva gracias a los lazos control implementados en

el código de Arduino, el sistema móvil se desvía hacia la derecha cuando se le indicaba el seguir una línea recta.

Lo primero que se hizo fue intentar ajustar las ruedas, se cambiaron las correas defectuosas y se apretaron los tornillos de los motores y engranajes, que estaban algo sueltos. Tras eso, al ver que el resultado era el mismo, se optó por cambiar a unos motores nuevos, debido a que se detectó que el encoder de uno de los motores fallaba, mediante el uso de un osciloscopio.

Tras cambiar a los motores nuevos, hubo que rehacer el cableado debido a que el orden de los pines era distinto. Al probarlos se comprobó que la desviación era un poco menor, pero se seguía desviando.

Se separó la alimentación de los motores de la alimentación de la ESP32 para reducir el ruido, y, sin embargo, el resultado seguía siendo el mismo.

Se probó a cambiar el código de Arduino a uno más simple, enviando la misma señal a los 4 motores, obtenida gracias a hacer la media de velocidades de los 4 encoder. Sin embargo, al enviar la misma potencia a los 4 motores, cada uno alcanzaba una velocidad diferente, por lo que no era una alternativa válida y se descartó.

Se hicieron pruebas con un sensor (MPU-9250), haciendo uso tanto del acelerómetro como del giroscopio que lleva incorporado, pero los resultados no fueron favorables, debido a que el sistema móvil tiembla mucho en su base y eso afectaba al acelerómetro y no se podía colocar el giroscopio en el medio de la base del sistema móvil por lo que daba medidas incorrectas.

También se optó por la opción de añadir un sesgo a uno de los lados de las ruedas del sistema móvil, es decir multiplicar por un valor configurable un poco inferior a 1 la velocidad de referencia de los motores 2 y 4, para compensar la desviación de la dirección. Sin embargo, esto no dio buenos resultados, ya que la desviación no era continua, sino que de vez en cuando, el robot desviaba su trayectoria hacia uno de los lados, por lo que frenar un poco la rueda de uno de los lados no ayudaba, por lo tanto, esta opción se descartó.

También se descubrió que al tensar demasiado los cables metálicos necesarios para reducir los temblores que pueden llegar a la cámara, las ruedas de un lado u otro se descuadran, es decir, que se suben por encima de las demás y no llegan a tocar el suelo, lo que no permitía que el movimiento fuese completamente rectilíneo.

Finalmente se arregló este problema añadiendo el selector de ajuste_fino para que el usuario pueda enderezar el sistema móvil hacia la izquierda o derecha pulsando el botón correspondiente del selector, solucionando el problema de las desviaciones que sufre el sistema móvil en su trayectoria

ANEXO 3. Listado de materiales para el montaje del proyecto

A continuación, se listan los materiales necesarios para el montaje del sistema móvil, junto a un datasheet o una página con información de cada material para su mejor comprensión:

- 1 cámara 360 Insta360Pro
 - <https://www.insta360.com/es/product/insta360-pro>
- 4 conectores, PH, Receptáculo, 6 Vías, 2 mm
 - <https://www.farnell.com/datasheets/3283086.pdf>
- 40 contactos terminales para crimpado de cables
 - https://www.googleadservices.com/pagead/aclk?sa=L&ai=DChcSEwihlerJ55OAAxXYptUKHds5BjsYABAFGgJ3cw&ohost=www.google.com&cid=CAESa-D2u-k1AIncGrLlozeynwfwKc-QuLbFpxoWdK0qM4rOdpYJnHGwuGqppJMaroedD0PfvGnlTxDw27u5W7MNLxAZbld5nPXrvCmB94XJV6GPIBYInIFZ0IRBBMp1AP018UvEM4ieivuWC8S2&sig=AUD64_2GwTBLrb_7vA5eNhSU2361cj6y7w&ctype=5&q=&ved=2ahUKEwiFtuPJ55OAAxVqVqQEHU8PAVoQ9aACKAB6BAGDEBw&adurl=
- 10 conectores de 10 contactos para cables usados en la PCB
 - <https://docs.rs-online.com/cb14/0900766b816fa49d.pdf>
- 10 conectores de 3 contactos para cables usados en la PCB
 - <https://www.mouser.es/datasheet/2/445/691352710003-1719656.pdf>
- 1 convertidor DC-DC RS Pro K78xxM-1000R3
 - <https://docs.rs-online.com/dc16/A700000006631882.pdf>
- 1 microcontrolador ESP32-Feather
 - <https://cdn-learn.adafruit.com/downloads/pdf/adafruit-huzzah32-esp32-feather.pdf>
- 4 motores CC de 6V 12V 24V con codificador
 - <https://usermanual.wiki/Pdf/lt25Ga370.987610479.pdf>
- 1 batería de litio de 12V
 - https://www.amazon.es/NASTIMA-Inal%C3%A1mbricas-Videoc%C3%A1maras-Reproductores-Electr%C3%B3nico/dp/B09FK4ZDSL/ref=sr_1_18_sspa?adgrpid=56505394256&hvadid=275592403137&hvdev=c&hvlocphy=1005545&hvnetw=g&hvqmt=b&hvrnd=11501133479706317054&hvtargid=kwd-296729968242&keywords=bater%C3%ADa+de+l%C3%ADtio+12v&qid=1689531753&sr=8-18-spons&sp_csd=d2lkZ2V0TmFtZT1zcF9tdGY&psc=1
- 1 condensador de 1µF y 6.3V para soldar en PCB
 - <https://docs.rs-online.com/ef6c/A700000008614803.pdf>
- 2 drivers en H de Placa Compatible Con L298N
 - <https://components101.com/modules/l293n-motor-driver-module>
- Chasis de tanque de Metal RC 4wd Robot oruga cadena coche vehículo plataforma móvil Tractor juguete
 - <https://es.aliexpress.com/i/32804987771.html>

En la siguiente tabla se listan todos los materiales necesarios para el montaje del sistema móvil junto a su precio para estimar un precio total del proyecto:

Producto	Precio (en euros)
1 cámara 360 Insta360Pro	3500

4 conectores, PH, Receptáculo, 6 Vías, 2 mm	0,26
40 contactos terminales para crimpado de cables	1,12
20 metros de cable multifilar amarillo, rosa, blanco, marrón, gris, verde	2,88
10 conectores de 10 contactos para cables usados en la PCB	34,3
10 conectores de 3 contactos para cables usados en la PCB	12,6
1 convertidor DC-DC RS Pro K78xxM-1000R3	4,04
1 microcontrolador ESP32-Feather Adafruit	30,04
Pack 100 condensadores de 1µF y 6.3V para soldar en PCB	0,03
4 motores CC de 6V 12V 24V con codificador	45,36
1 batería de litio de 12V	55,98
2 drivers en H de Placa Compatible Con L298N	11,98
Chasis de tanque de Metal RC 4wd Robot oruga	164,09
Horas de programación y montaje (20 euros por hora)	1500,00
TOTAL	5362,68

Por lo tanto, para el montaje del sistema móvil con cámara Insta360 Pro, el presupuesto llega hasta los 5362,68 euros. Sin contar el dinero que requieren las horas de programación y montaje del proyecto, el presupuesto es de 3862,68 euros, considerando unas 75 horas de trabajo para montar y programar todo el sistema móvil. Sin tener en cuenta el costo de la cámara, el sistema móvil en sí llega a costar 362,68 euros

ANEXO 4. Tablas de variables

A continuación, se muestra una tabla con todas las variables utilizadas en el código programado en el IDE de Arduino para controlar el microcontrolador ESP32.

NOMBRE	TIPO	DESCRIPCIÓN
MX_encoder_Y	INT	Número de pin del canal A o B del encoder de un motor
MX_CM_pin_dir	INT	Número del pin del canal directo de un motor
MX_CM_pin_inv	INT	Número del pin del canal inverso de un motor
Btin	STRING	Orden Bluetooth recibida vía Bluetooth Serial
Ord	STRING	Btin procesado y eliminando texto basura
Bton	STRING	Inicio de la orden anterior
conect	BOOL	Indica si el sistema móvil está conectado al bluetooth
Ref_iz	FLOAT	Referencia de velocidad de los motores 1 y 3
Ref_der	FLOAT	Referencia de velocidad de los motores 2 y 4
Ref_des	FLOAT	Referencia de velocidad a alcanzar en este instante
Ref_des_ant	FLOAT	Referencia de velocidad a alcanzar en el instante anterior
T	INT	Periodo de muestreo en milisegundos
C2RS	FLOAT	Sirve para convertir la velocidad de pulsos a rad/s
DutyMax	INT	Cantidad máxima de energía que se puede suministrar a un motor
Stop	BOOL	Indica si los motores están encendidos o apagados
MXik_1	FLOAT	Error en el instante anterior de uno de los motores
MX_ref	FLOAT	Referencia de velocidad a alcanzar de uno de los motores
MX_vel	FLOAT	Velocidad actual de uno de los motores
timer_d	INT	Timer que cuenta cuánto tiempo lleva desconectado el sistema móvil
FLAG_DISCONNECT	BOOL	Indica que el sistema móvil se ha desconectado vía Bluetooth
FLAG_CONNECT	BOOL	Indica que el sistema móvil se ha vuelto a conectar vía Bluetooth
tiempo_impulso	INT	Tiempo en milisegundos en el que el sistema móvil está ajustando
timer_co	INT	Timer que cuenta cuánto tiempo lleva ajustando el sistema móvil
MX_ready	BOOL	Indica que el motor ya ha alcanzado la velocidad de referencia
tiempo_giro	INT	Tiempo en milisegundos en el que el sistema móvil está corrigiendo su trayectoria
giro	INT	Dirección en la que corregirá el sistema móvil
timer_gi	INT	Timer que cuenta cuánto tiempo lleva corrigiendo el sistema móvil
kp	FLOAT	Constante proporcional del lazo control
ki	FLOAT	Constante integral del lazo control
Uk	FLOAT	Parte proporcional del controlador

Uik	FLOAT	Parte integral del controlador
Ik	FLOAT	Integral del error del controlador
acc	FLOAT	Suma de Uk + Uik
duty	INT	la señal que se envía al pin correspondiente
Desv	INT	Dirección que debe tomar por el sistema móvil
AntDesv		Dirección anterior
isCambioDesv	BOOL	Detecta si se ha cambiado de una <i>desv</i> positiva a una negativa
er	FLOAT	Error entre la entrada y la referencia de la velocidad
Cm_ref	FLOAT	Cambio de referencia por cada ciclo
ajuste_fino	FLOAT	Valor utilizado para corregir la velocidad de las ruedas de uno de los lados
stoplento	BOOL	Detecta si el sistema está parando de forma progresiva
sesgoLento	FLOAT	Número por el que se multiplican las ruedas que deben ir lentas para girar
sesgoRapido	FLOAT	Número por el que se multiplican las ruedas que deben ir rápidas para girar

A continuación, se muestra una tabla que recoge todas las variables utilizadas en el código de la aplicación móvil en Android Studio.

NOMBRE	TIPO	DESCRIPCION
dir	INT	Dirección a tomar por el sistema móvil
ac	FLOAT	Aceleración o cambio de referencia de velocidad por cada ciclo
vel	FLOAT	Velocidad que debe tomar el sistema móvil
sesgo_delante	FLOAT	Valor utilizado para corregir la velocidad de las ruedas de uno de los lados
correccion	FLOAT	Valor que se suma o resta a <i>sesgo_delante</i> para girar a un lado u otro
correccionV	FLOAT	Valor que se suma o resta a <i>vel</i> para aumentar o disminuir la velocidad
correccionA	FLOAT	Valor que se suma o resta a <i>ac</i> para aumentar o disminuir la aceleración
tiempo_giro	INT	Tiempo en milisegundos en el que el sistema móvil está corrigiendo su trayectoria
stop	BOOL	Indica si los motores están encendidos o apagados
esperarFin	BOOL	Indica si los botones pueden enviar mensajes o no
onkeydown	BOOL	Indica si se ha pulsado uno de los botones del mando
onKeyup	BOOL	Indica si se ha dejado de pulsar uno de los botones del mando
pantallaGrande	BOOL	Indica si se va a visualizar el video 360 en pantalla grande o no
forw_back	INT	Indica si el sistema móvil se está moviendo hacia detrás o hacia delante
tiempoEsperaMando	INT	Indica el tiempo que pasa desde que el mando manda una orden hasta que puede mandar otra

tiempoEspera	INT	Indica el tiempo que pasa desde que el teléfono manda una orden hasta que puede mandar otra
tiempoImpulso	INT	Timer que cuenta cuanto tiempo debe estar ajustando el sistema móvil
videoUrl	STRING	Url del vídeo en directo de la cámara
connection	BOOL	Indica si el dispositivo móvil está conectado por bluetooth al sistema móvil
btSocket	BLUETOOTHSOCKET	Se utiliza para establecer una conexión de comunicación inalámbrica entre dispositivos a través de Bluetooth
btAdapter	BLUETOOTHADAPTER	Se utiliza para administrar y controlar la funcionalidad Bluetooth de un dispositivo
exoplayer	EXOPLAYER	Instancia del reproductor de vídeo utilizado
context	CONTEXT	Contexto de la pantalla en la que se encuentra la aplicación
isModoJoystick	BOOL	Indica si se está utilizando el mando o no