ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"SISTEMA DE VISIÓN EN DRONES PARA LA DETECCIÓN Y SEGUIMIENTO DE BUQUES EN UN AMBIENTE SIMULADO"

INFORME DE MATERIA INTEGRADORA

Previo a la obtención del Título de:

INGENIERO EN CIENCIAS COMPUTACIONALES
ORIENTACIÓN SISTEMAS MULTIMEDIA

CARLOS LUIS CEREZO ROMERO

DIEGO ARMANDO BRIONES BRIONES

GUAYAQUIL – ECUADOR

AÑO: 2018

AGRADECIMIENTOS

A mis padres y mi Tía por apoyarme incondicionalmente en mi formación profesional, y ser mi ejemplo de vida.

A mis amigos con los cuales compartí momentos muy agradables en esta etapa de mi vida.

A mis profesores que supieron guiarme en este camino hacia el éxito.

Diego Briones

Agradezco a Dios y a mis padres por el apoyo brindado a los largo de la carrera universitaria.

También agradezco a mi tío Gabriel Cerezo por haberme recibido en su hogar durante gran parte de mi vida universitaria.

Carlos Cerezo

DEDICATORIA

El presente trabajo quiero dedicárselo a las personas que me dieron la vida, mis padres, quienes siempre se han preocupado por mis estudios y me han brindado todo su apoyo para cumplir esta meta.

Diego Briones

A mis Padres y hermanos por estar siempre conmigo a pesar de la distancia.
También este trabajo va dedicado a un compañero y amigo Fernando Chica y a toda su Familia.

Carlos Cerezo

TRIBUNAL DE EVALUACIÓN

PhD. Miguel Realpe	**PhD. Federico Dominguez**
PROFESOR EVALUADOR	PROFESOR EVALUADOR

DECLARACIÓN EXPRESA

"La responsabilidad y la autoría del contenido de este Trabajo de Titulación, nos corresponde exclusivamente; y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

.....

Diego Armando Briones Briones

.....

Carlos Luis Cerezo Romero

RESUMEN

El presente trabajo consiste en la detección y seguimiento de buques, que se encuentren navegando en rutas no permitidas (anómalas) por la Fuerza Naval del Ecuador. Este trabajo es una necesidad que tiene la Fuerza Naval del Ecuador, de que sus aviones no tripulados operen de forma autónoma y vigilen el mar territorial, así no demandar de altos costos operativos como es tener un operador en tierra que controle el dron de vigilancia, rotar turnos de operadores, comunicaciones del dron a base.

Para este proyecto se utiliza el software de simulación VREP, en donde se creará una escena simulando el mar territorial del Ecuador con barcos navegando y el dron que hará la detección y seguimiento del buque. Para la detección y seguimientos se realizará una implementación mediante un programa escrito en el lenguaje de programación Python y la librería OpenCV, el cual lee los datos de una cámara (sensor de visión) en un avión no tripulado o dron. Si existe una ruta anómala se guardará sus coordenadas de posicionamiento GPS sin necesidad de tener un operador en tierra.

Como antecedente La Fuerza Naval del Ecuador ya tiene establecidas las rutas de navegación permitidas, donde se establecen puntos de partida y llegada desde donde navegan los buques, por lo tanto el dron debe estar en la capacidad de detectar cuando un buque no cumple con las trayectorias establecidas, para ello se le proporcionan coordenadas GPS al dron de las rutas que son autorizada para la navegación.

ÍNDICE GENERAL

CAPÍTULO 1

1. INTRODUCIÓN

En años recientes el uso de drones se ha extendido en diversos territorios del mundo con aplicaciones variadas, tanto en áreas militares como civiles. Los drones pueden ser programados o controlados para muchas tareas.

Ecuador no ha estado ajeno a esta realidad y ha comenzado el desarrollo y uso de aviones no tripulados, las aplicaciones comunes de este tipo de vehículo se centran en tareas donde tienen entornos de acceso difícil o que representan algún peligro, como por ejemplo: tareas de vigilancia y seguimiento, reconocimiento geográfico, etc.

La detección y seguimiento de los buques con drones, actualmente es una necesidad de la Fuerza Naval del Ecuador, lo cual como primera etapa del proyecto consiste en simular rutas de navegación de buques que son vigiladas por los aviones no tripulados. En este proyecto nos basados en la información proporcionada por la Fuerza Naval del Ecuador la cual como entidad de control y vigilancia del mar territorial ya tienen rutas definidas en las que la navegación es permitida.

Para crear la escena de simulación hacemos uso del software VREP que nos permite agregar los elementos necesarios en la simulación como son: dron, textura del mar, buques, crear caminos (rutas de navegación), el proceso comienza con un dron que explorar una zona determinada hasta que se detecte un buque, y realice el seguimiento (tracking) del buque que está navegando.

El dron tiene incorporado una cámara y sensores como giroscopio y acelerómetro, que dentro de la simulación se controlan y manipulan mediante scripts escritos en el lenguaje de programación Pyhton. Para detectar el buque realizamos el procesamiento digital de imágenes con la librería OpenCV, y para el seguimiento (tracking) utilizamos el algoritmo Hungarian.[1]

1.1 Objetivo

En el presente proyecto se plantea realizar la detección y seguimiento de buques de forma autónoma, por medios de algoritmos de visión por computadora en un ambiente simulado.

CAPÍTULO 2

2. ESTADO DEL ARTE

En esta sección se presentan las técnicas que existen de detección y seguimiento de objetos, filtros y algoritmos, también analizaremos los diferentes simuladores robóticos que existen. Además, hacemos referencia a los trabajamos realizados por otras universidad en el campo de Visión por Computadora para la detección y seguimiento de objetos. Por último, en este capítulo hacemos referencia a las regulaciones que el estado Ecuatoriano tiene respecto al uso de drones.

2.1 Técnicas de Detección y Seguimiento

Existen varias técnicas de detección de objetos a continuación se analizan, la técnica de detección de bordes [2], lo cual se puede definir como transiciones entre dos regiones de niveles de gris significativamente distintos o también que en una imagen digital se identifica los valores de cambio de brillo drásticos, el algoritmo de Canny [3] está diseñado para la identificación de bordes de una imagen. El proceso de detección empieza transformando la imagen de entrada a una imagen en escala de grises, a la que es aplicado un filtro Gaussiano para reducir el ruido existente en la imagen, finalmente la imagen es procesada por el algoritmo de Canny al cual se le debe decir cual son los valores del umbral máximo y mínimo que nos permite decidir si un pixel pertenece al fondo o al objeto para la detección de bordes ver Figura 2.1. Manipulando los valores del umbral se puede llegar a tener en la imagen de escala de grises solo los bordes del objeto a detectar.

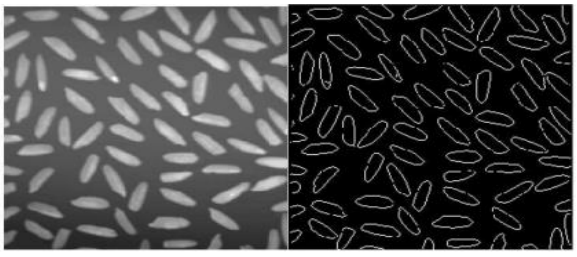


Figura 2.1: Resultado del Algoritmo de Canny

Otra técnica es la sustracción de fondo ampliamente utilizada para detectar objetos en movimiento a partir de cámaras estáticas [4], la señal de entrada para esta técnica debe ser secuencia de imágenes o video. Para detectar el objeto se debe inicialmente tener una imagen de referencia donde no haya objetos en movimiento, se deja pasar un tiempo para que existe variación entre la imagen de referencia y los actuales fotogramas. Una desventaja, como menciona Luis del Valle en su blog de Visión Artificial es que si el objeto en movimiento se queda quieto, no va se detecta [5].

El seguimiento de objetos puede ser un proceso lento debido a la gran cantidad de datos que contiene un video. Las dificultades que se encuentran en el seguimiento de objetos son: cambios de posición, la intensidad de luz y ruido en la imagen que se está procesando entre otros. La representación de un objeto en escena puede ser por puntos, formas geométricas, silueta del objeto, y modelos esqueléticos. A continuación en la Figura 2.2 se presenta una clasificación de las diferentes alternativas que existen para realizar el seguimiento de un objeto o también conocido como tracking [6].

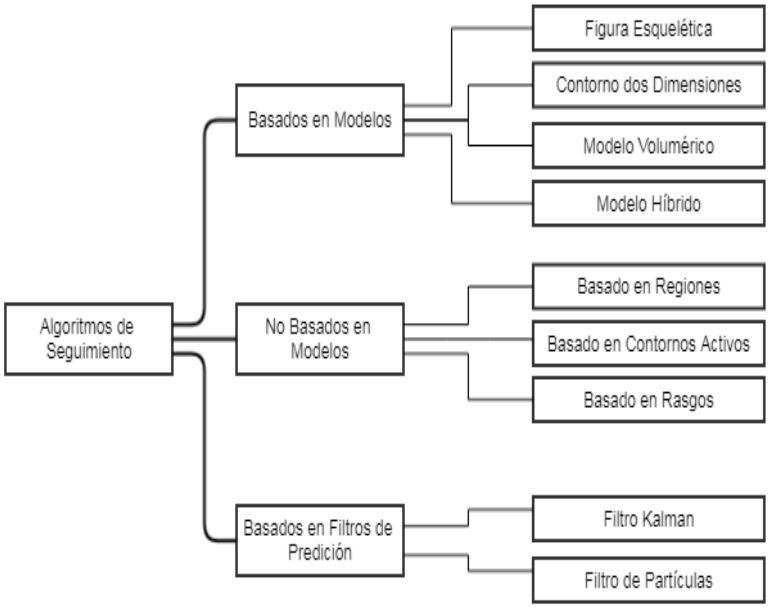


Figura 2.2 Clasificación algoritmos de seguimiento

Los algoritmos basados en modelos tienen un patrón predeterminado del sujeto a seguir. Por ejemplo algoritmo para seguimiento de personas tenemos un modelo basado en Figura Geométrica [6] que ha sido desarrollado para el cuerpo humano en su totalidad. En la otra clasificación tenemos los algoritmos basados en filtros de predicción usan información información recopilada en frames anteriores para determinar la nueva posición del objeto. Por ejemplo, el Filtro de Kalman es una solución recursiva, que hace uso de la información previa para predecir la nueva posición del objeto.

2.2 Simuladores Robóticos

La simulación tiene por objetivo interpretar el funcionamiento de un sistema sin la necesidad de construir uno real, ya que experimentar en la vida real sería muy costoso y en ciertas ocasiones representa un peligro para la integridad de las personas involucradas. Existen varios simuladores de robots que ofrecen distintas posibilidades en cuanto a funcionamiento o control de robots desde los lenguajes de programación. A continuación presentamos un análisis de algunos simuladores de robots más utilizados.

- ARGoS, simulador de código abierto, orientado a la simulación de enjambres de robots, permite utilizar diferentes motores físicos simultáneamente.
- Webots, simulador destinado a la enseñanza e investigación de robots móviles haciendo uso de una librería llamada ODE(Open Dynamics Engine), software comercial, permite controla los movimientos de un robot mediante scripts escritos en C, C++, Python y Java.
- V-REP (Virtual Robot Experimentation Platform), permite trabajar con robots móviles y no móviles, controlas las partes que conforma un robot median scripts de programación, es un software libre mientras se use con fines educativos e investigación.
- OpenRave, simulador de código abierto, multiplataforma, todos los robots se almacenan en ficheros XML por lo que es muy sencillo agregar nuevos entornos al robot.

2.3 Proyectos desarrollados

En la Universidad Nacional de Trujillo se realizó el proyecto de “Detección de bordes mediante el algoritmo de Canny”. Este algoritmo está catalogado como uno de los mejores métodos de detección de contornos basados en:

- **Primera Derivada**, es usada por que toma el valor de cero en todas las regiones donde no varía la intensidad (valores de sus pixeles), y cuando el resultado es diferente a cero significa que hubo un cambio brusco.

- **Obtención del gradiente**, para la obtención del gradiente se debe aplicar un filtro Gaussiano a la imagen original con la finalidad de suavizar la imagen y eliminar los posibles ruidos. Una vez aplicado el filtro Gaussiano se obtiene la magnitud y modulo (orientación).

Como resultado de estos métodos se obtuvo una imagen con bordes adelgazados con máximos locales creados por el ruido. Para eliminar dicho ruido se aplicó Histéresis del umbral.

Histéresis del umbral [7], la mayoría de umbrales usan un límite, lo cual da como resultado líneas entrecortada de acuerdo al valor del umbral establecido. La histéresis se encarga de este problema, estableciendo dos umbrales: umbral máximo y un umbral mínimo. En donde si consideramos un segmento de línea, si el valor está por debajo del umbral mínimo, se rechaza automáticamente. Si el valor está por encima del umbral superior se aprueba automáticamente, pero si el valor esta entre los 2 umbrales, será aceptado solo si está conectado a los pixeles.

Con métodos aplicados en este proyecto quedan contornos abiertos para lo cual se aplicó algoritmo de Deriche y Cocquerez.

Los resultados obtenidos en este proyecto fueron muy satisfactorios, pero como desventaja menciona el suavizado, puesto que, si aumentamos σ de la mascarilla lograron reducir el ruido pero difuminaron los bordes y perdieron calidad al momento de calcular la orientación.

CAPÍTULO 3

3. DISEÑO E IMPLEMENTACIÓN

Con el software VREP se comienza creando un proyecto nuevo con una escena vacía (ver Anexo 2), que es la representación gráfica inicial de la simulación, a la cual se le van agregando los elementos que se necesitan para operar sobre estos. Lo primero que se agrega a la escena es el dron predeterminado que tiene VREP que pertenece a la categoría de robots móviles. Luego se procede a cambiar el tablero a cuadros por una de las propiedades que tiene el software VREP para simular el mar. El siguiente paso fue agregar el buque, dado que éste no estaba dentro del simulador descargamos [8] el archivo que representan al buque con extensión 3d, que finalmente es importado a la escena. Después de agregar el dron, el buque y el mar, se procede a configurar las propiedades de la escena; las dimensiones del mar se trabajó con una escala de 1:100 metros por lo tanto el área del mar es de 4Km^2, al dron se le agregan los sensores de Acelerómetro, Giroscopio y GPS, con ayuda del simulador se crean los path (rutas) que son la unión de varios puntos con sus coordenadas (x, y) y que se grafica en la simulación como una línea continua suavizada y representa la ruta de navegación que tendrá el buque. La figura 3.1 muestra el resultado final de la construcción de la escena.

Figura 3.1: Escena de la simulación en V-REP

Para controlar el desplazamiento del dron y sus sensores realizamos la conexión de VREP con el lenguaje de programación Python (ver Anexo 3), que es donde escribimos los scripts para realizar la lectura de las imágenes que captura el dron con su cámara, además de las coordenadas GPS que son almacenadas en un archivo de texto plano.

El método que se propone en este proyecto para la detección y seguimiento de buques en el mar es un proceso cíclico. El dron tiene incorporado una cámara la cual será utilizada para que en primera instancia realice la detección del buque con la técnica de sustracción de fondo.

En caso de que uno o más buques entran en el rango de visión de la cámara incorporada en el dron, se procesa la información y se determina si el buque debe ser seguido por el dron. Para realizar el seguimiento el dron se desplaza en (x, y) y mantiene el objeto identificado en su rango de visión de acuerdo al movimiento del buque. Cuando el buque es identificado como un peligro por su ruta anómala el dron debe ir guardando su coordenada GPS, para posteriormente obtener el tracking del buque. Para determinar si un barco está en ruta anómala, en este proyecto se lo está comparando con coordenadas de rutas permitidas, previamente almacenadas en archivo plano.

3.1 Implementación

La arquitectura del proyecto está conformada por tres componentes principales como se muestra el diagrama de bloque en la figura 3.2. A continuación se explica el funcionamiento de cada uno de ellos.

Simulación V-REP, Virtual Robot Experimentation Platform [9] software de simulación de robots, está basado en una arquitectura distribuida lo que permite controlar cada objeto de forma individual mediante scripts de programación en lenguajes como Python, Java, Matlab, C/C++ [10]. Cuando se crea un nuevo proyecto por defecto aparece una escena que es el elemento principal de la simulación, y que ésta a su vez está conformada por subelementos llamados modelos. Para este proyecto se considera una escena con los siguientes objetos:

1. El Mar en movimiento, se considera corriente marina.
2. Buque “B1”, archivo con extensión 3d importado dentro de la escena.
3. Buque “B2”, archivo con extensión 3d importado dentro de la escena.
4. Path “P1”. Representa la ruta de navegación permitida del buque.
5. Path Anómalo “PAnomal2”, ruta no permitida del buque.
6. Quadricopter “Dron”, agregado a la escena desde la sección de robots móviles dentro de VREP, que a su vez posee los siguientes elementos.
    6.1. Cámara del Dron con Resolución 256/256 y ángulo de visión 60 grados.
    6.2. GPS. sensor que nos permite obtener coordenadas de posición del dron.
    6.3. Acelerómetro sensor destinado para controlar la potencia de los motores.
    6.4. Giroscopio, sensor para determinar la dirección en la que debe avanzar el dron.

Visión, el sistema comienza con el procesamiento continuo de las imágenes recibidas de la cámara del dron, para ello se utilizó la librería OpenCV [11] que nos permiten realizar la detección del buque con la técnica de sustracción de fondo y filtro Gausseano, y luego hacer el seguimiento del buque obteniendo las coordenadas de sensor GPS del dron respecto al buque.

Control, cuando el dron está ubicado en las mismas coordenadas (x, y) del buque, sólo cambia el desplazamiento positivo en el eje Z para dar la altura de vuelo al dron para realizar el seguimiento, se implimentó el Algoritmo de Hungarian con el menor costo de desplazamiento de los barcos comparando con t-1. Para obtener el tracking del buque. Se obtienen las coordenadas de la posición actual a través de los sensores del dron.

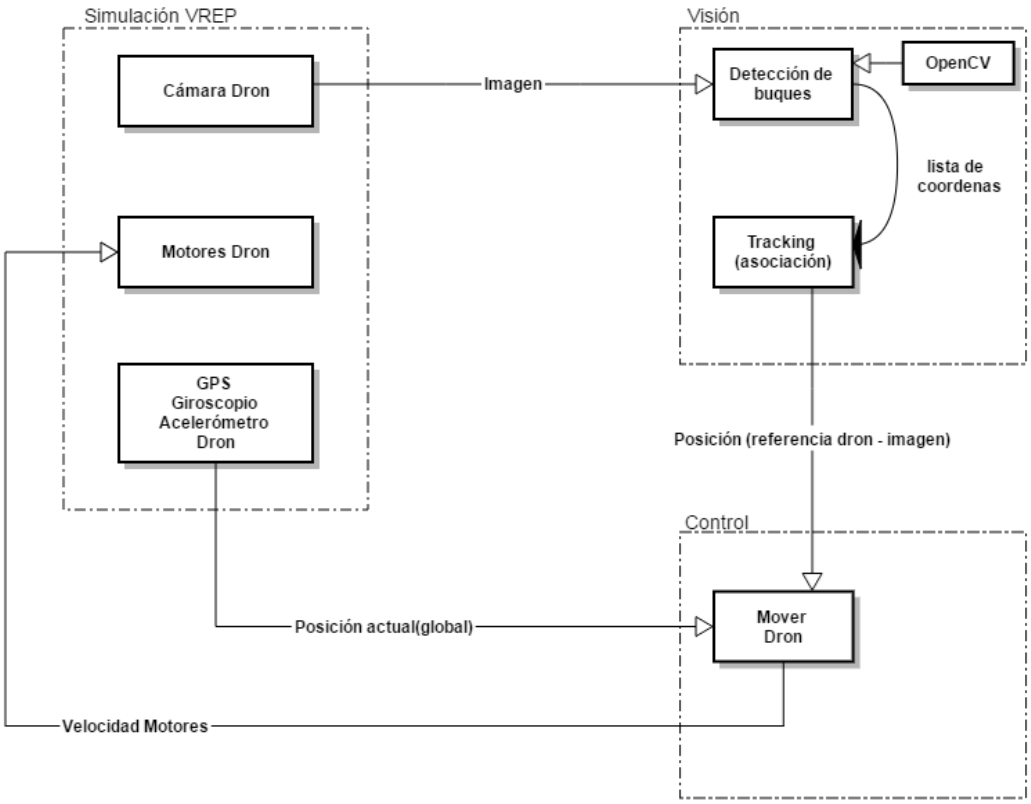


Figura 3.2: Diagrama de Bloques.

Luego que la escena ha sido creada y se ha establecido la conexión entre Python y V-REP, el sistema comienza un proceso continuo de captura de imagen de la cámara del dron cada ‘n>5’ frame con el objetivo de ahorrar costo computacional en el procesamiento de imágenes.

La escena inicia en el tiempo t_0 con un buque en una posición de coordenadas (x, y) y el buque en otro posición (w, z). Ya en el tiempo t_1 el buque y el dron están en movimiento, cuando el buque entra en su totalidad en el ángulo de visión de la cámara del dron mediante la técnica de sustración de fondo y filtro Gaussiano se lo detecta y se lo enmarca en un recuadro de color verde como se muestra en la figura 3.3, donde dos buques fueron detectados.

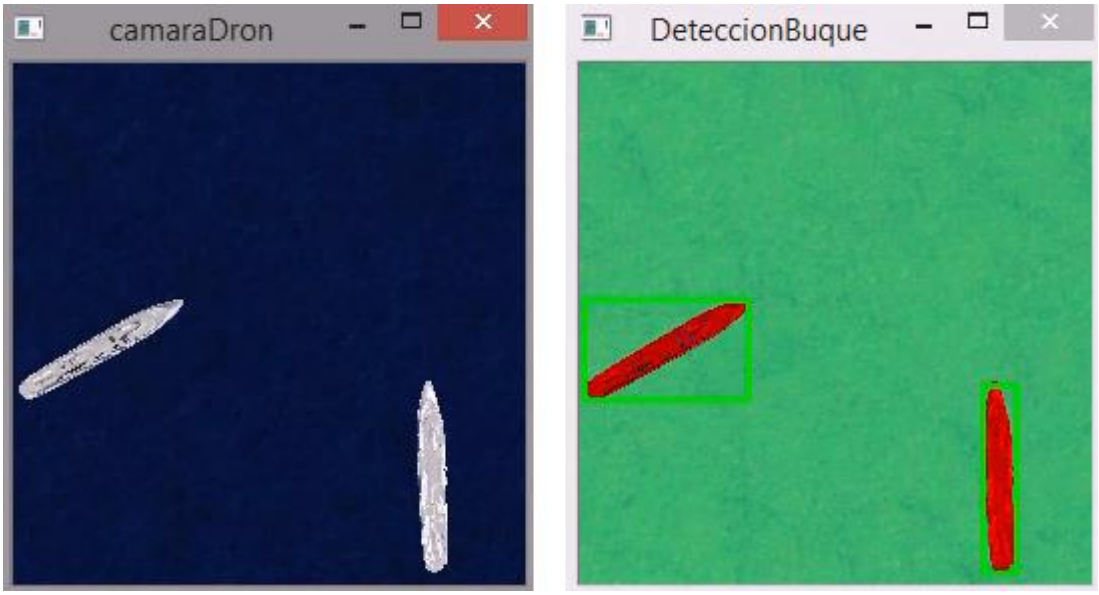


Figura 3.3: Cámara del Dron y Detección de los Buques

El Seguimiento es un proceso que se ejecuta una vez se detecta un buque. Para determinar si el buque se está saliendo del ángulo de visión de la cámara, se procede con el desplazamiento autónomo del dron para volver a ubicar al buque en el centro del ángulo de visión de la cámara del dron, para ello capturamos la posición en pixeles en el tiempo t_n del buque respecto a la cámara que tiene una resolución de 256 x 256 pixeles del dron, luego en el tiempo t_{n+1} realizamos una nueva captura de la posición del buque en pixeles respecto a la cámara del dron lo que nos permite dar el seguimiento al buque con el dron.

3.3 Seguimiento a las rutas Anómalas

En este proyecto al detectar que el buque sigue una ruta “path” anómala, inmediatamente el dron recibe la señal que tiene como prioridad seguir al buque con la ruta anómala. Para definir si una ruta es anómala o no, previamente hemos almacenado coordenadas GPS con las rutas permitidas.

Como resultado de todo el proceso se obtiene 3 archivos:

1. Tracking → contiene las coordenadas GPS del dron en su recorrido.
2. Acelerómetro → contiene los valores del acelerómetro del dron en su recorrido.

3. Giro Sensor → contiene los valores del Giro Sensor del dron en su recorrido.

En la escena se crearon cinco rutas diferentes de navegación para los buques, en las cuales siempre iniciamos con el dron sobre volando el mar y a través del sensor de visión realizamos el procesamiento digital de imágenes cada 10 frames/segundo, para que los buques sean detectados deben estar dentro del ángulo del sensor de visión.

Para poder realizar el seguimiento de los busques el dron tiene previamente cargadas las coordenadas GPS de las rutas permitidas, lo cual le permite decidir que buque es el que debe seguir.

CAPÍTULO 4

4. DISCUSIÓN DE RESULTADOS

Para la detección y seguimiento de un buque se realizaron cinco pruebas en una misma ruta (Path) con una distancia total de 400 metros. La lectura de los frames de la cámara era de 30 por segundo. En la prueba uno P1 no se detectó el buque, entonces procedió a cambiar el valor del umbral de las imágenes capturadas y procesadas en escala de grises para dejar pasar los pixeles con mayor cambio en la intensidad de los pixeles vecinos, y así aplicar mejor la sustracción de fondo. En la prueba dos P2 con el filtro Gaussiano mejorado se detectó el buque, pero no se consiguió realizar el seguimiento. En las siguientes pruebas P3, P4 y P5 se comenzó a seguir el buque pero debido a la gran cantidad de procesamiento que se realizaba con el dron iba perdiendo el seguimiento del buque debido a que se procesó 30 frames por segundo. Mientras el barco sigue con su trayectoria de navegación, el dron procesa cada frame y no se traslada hasta procesar las imágenes, esto ocasiona que el buque salga del Angulo de visón del dron por lo tanto el seguimiento era parcial y no completo toda la ruta de navegación. En la Tabla 1 se muestra el resultado de estas pruebas.

# Prueba	*Detección*	*Distancia*	*Seguimiento*
P1	No	0	No
P2	Si	0	No
P3	Si	100	Parcial
P4	Si	100	Parcial
P5	Si	100	Parcial

Tabla 1. Resultados de pruebas realizadas $(30\ frame/seg)$ con una distancia de 400m.

Para controlar que el buque no se salga del ángulo de visión del dron se cambió la tasa de procesamiento de los frames bajando a 10 frames por segundo. En la siguiente tabla se muestran los resultados de las mismas pruebas pero con el variante del procesamiento de frames.

# Prueba	*Detección*	*Distancia*	*Seguimiento*
P1	Si	400	Si
P2	Si	400	Si
P3	Si	400	Si
P4	Si	400	Si
P5	Si	400	Si

Tabla 2. Resultados de pruebas realizadas $(10\ frame/seg)$ con una distancia de 400m.

En la Tabla 3 se muestran los resultados de la detección y seguimiento de dos buques con diferentes rutas de navegación.

En las rutas se realizaron 10 pruebas donde en el tiempo t_n en el ángulo de visión del dron son detectados dos buques, apartir de ese momento se empiezan a comparar las coordenadas de GPS de las rutas permitidas con las coordenadas de los buques navegando el seguimiento se dio con una efectividad del 40%. Se obtuvieron mejores resultados cuando los buques dentro del ángulo de visión son detectados y van en sentido perpendicular. Además se obtuvieron mejores resultados cuando la velocidad de navegación es la misma para los buques. La distancia no influye en el resultado de las pruebas es sólo la representación de la navegación de un punto a otro dentro de la simulación.

Rutas	*#Pruebas*	*Buques*	*Detección*	*Distancia*	*Seguimiento*
R1	10	2	SI	100	4/10
R2	10	2	SI	200	3/10
R3	10	2	Si	300	5/10
R4	10	2	Si	400	6/10
R5	10	2	Si	500	7/10

Tabla 3. Resultados de pruebas realizadas cada $(10\ frame/seg)$ con una distancia a recorrer de 400m aproximadamente.

A continuación se presentan como entran al ángulo del visón los buques y el sentido en el que estos están navegando, como referencia a los resultados obtenidos en la Tabla 3, el sentido en el que los buques ingresan al ángulo de visión del dron si influye en el seguimiento.

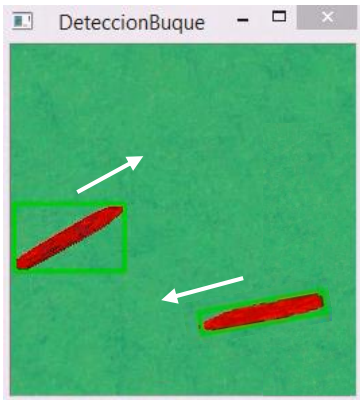


Figura 4.1 Pruebas en R1 sentido semiopuesto

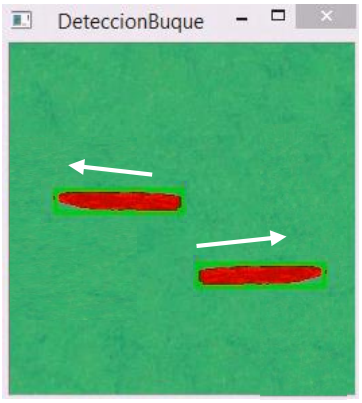


Figura 4.2 Pruebas en R2 sentido opuesto

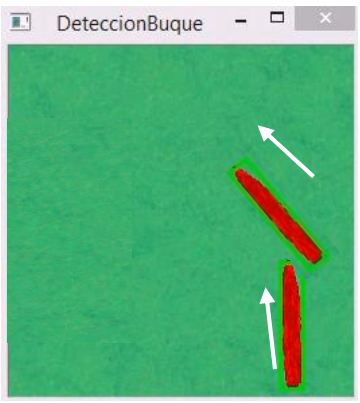


Figura 4.3 Pruebas en R3 sentido semi perpendicular

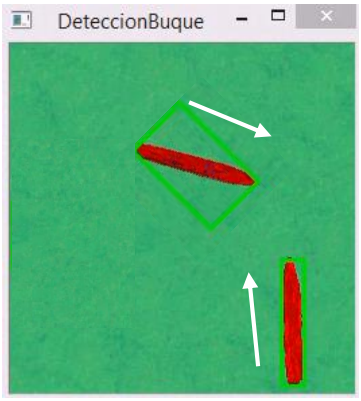


Figura 4.4 Pruebas en R4 sentido perpendicular

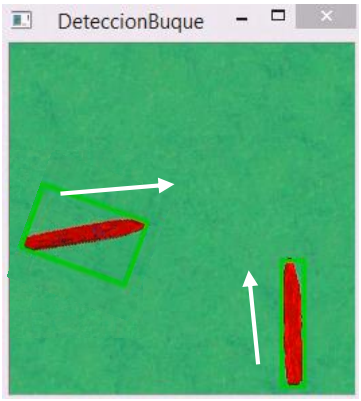


Figura 4.5 Pruebas en R5 sentido perpendicular

Las rutas fueron creadas de tal forma que podemos tener dos buques en el ángulo de visión.

CONCLUSIONES Y RECOMENDACIONES

Para poder determinar la detección de un buque a través de filtro Gaussiano, realizar la técnica de sustracción de fondo, se definió un ROI que es un área dentro del ángulo de visión para que este sea enmarcado en el recuadro de detección.

Para establecer el seguimiento del buque se trabaja con los pixeles que da el ángulo de visión del dron para así establecer cuánto se movió el buque, pero como los dos objetos están en movimiento el marco de referencia no es el mismo en los diferentes tiempos, por lo que se actualizan las coordenadas del buque mientras se alejaba del marco de referencia.

Los mejores resultados en cuanto a seguimiento se obtuvieron cuando los buques navegan por rutas que van en sentido perpendicular, ya que cuando van en un sentido paralelo el dron no siempre siguió al buque que estaba en ruta anómala.

De acuerdo con la Tabla 3 en las pruebas de la R2 fue el peor de los casos ya que los buques navegan en sentido contrario y cuando entran en el ángulo de visión del dron, éste procesa los datos de ambos, pero mientras lo hace los buques salen muy rápido del ángulo de visón por lo tanto no se realiza el seguimiento.

De acuerdo con la Tabla 3 en las pruebas de la R5 se obtuvieron los mejores resultados ya que los buques navegan en sentido perpendicular si tiempo dentro del ángulo de visón es mayor, por lo que permite procesar los frames y determinar que buque debe seguir.

El costo computacional de procesar todos los frames del video de la cámara del dron hace que el seguimiento sea parcial, debido que el buque siempre está en movimiento. Esto ocasiona un desfase mientras que el dron procesa información de la cámara, luego este se desplaza de forma autónoma, por lo tanto llega un momento que el buque sale del ángulo de visión del dron y lo pierde. Por esta razón se recomienda disminuir la captura de números de frames por segundo.

Como recomendaciones para un trabajo futuro se debe tomar en cuenta para una implementación real que el dron hace uso de una batería la cual tiene un tiempo duración para realizar el vuelo, considerando esto se puede terminar cual sería el radio de operación de un dron para que pueda volver a base luego de cumplir con su tarea.

Como trabajo futuro, se deben contemplar pruebas con un mayor número de escenas diferentes para analizar el comportamiento del sistema. Otro aspecto a considerar sería auto aprendizaje de las rutas permitidas por el espacio marítimo con redes neuronales para facilitar la detección de las rutas anómalas.

BIBLIOGRAFÍA

[1] Gábor Szűcs, Dávid Papp, Dániel Lovas, “SVM classification of moving objects tracked by Kalman filter and Hungarian method”, Budapest University of Technology and Economics, Budapest, Hungary. 2015

[2] Morales Guzmán Saúl, Acosta Vázquez Sergio “Detección de Bordes mediante Inteligencia Artificial”, Tema de Tesis, Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Profesional “Adolfo López Mateos”, México 2013. Disponible en http://tesis.ipn.mx/bitstream/handle/123456789/13778/deteccion%20de%20bordes%20mediante%20inteligencia%20artificial.pdf?sequence=1, visitado 2017, Septiembre 10.

[3] Jorge Valverde, “Detección de bordes mediante el algoritmo de Canny”, Escuela Académico Profesional de Informática, Universidad Nacional de Trujillo, Perú. 2008.

[4] L. Gervasoni, J.P. D’amato, R. Barbuzza, M. Vérene, “Método eficiente para la sustracción de fondo en videos usando GPU”, Facultad de Ciencias Exactas, Universidad Nacional de la Prov. De Buenos Aires. Argentina, Septiembre 2014.

[5] L. Del Valle, “Detección de movimiento con OpenCv y Python”, Visión Artificial [Online], Disponible en: https://programarfacil.com/blog/vision-artificial/deteccion-de-movimiento-con-opencv-python/, Visitado 2017, septiembre 10.

[6] J. Yáñez, “Tracking de personas a partir de visión artificial”, Proyecto de fin de Carrera, Universidad Carlos III de Madrid, España, 2010, Disponible en: https://orff.uc3m.es/bitstream/handle/10016/10118/PFC_Javier_Yanez_Garcia.pdf;jsessionid=D2E5B104DF62F5251A23C7A9A3D2E06A?sequence=4, visitado 2017, Septiembre15.

[7] Gonzalo Andres Andrade Moreira, “Correspondencia Multiespectral en el espacio de HOUGH”, Proyecto de fin de Carrera, Escuela Superior Politécnica del Litoral, Ecuador, 2015, Disponible en: https://www.dspace.espol.edu.ec/retrieve/90395/D-84729.pdf visitado 2017, septiembre 20.

[8] Modelos de buques con extensión 3d. (2016) Disponible en: https://3dlancer.net/es/models?keyword=buque, visitado 2017, septiembre 15.

[9] VREP User Manual Version 3.3.2, (2016, agosto 29), [Online] Disponible en: http://www.coppeliarobotics.com/helpFiles/ , visitado 2017, septiembre 19.

[10] Carlos Martínez, “Simulación en entornos con robots manipuladores móviles”, Tesis de grado, Universidad Politécnica de Valencia, (2015), [Online] Disponible en: https://riunet.upv.es/bitstream/handle/10251/55467/MART%C3%8DNEZ%20-%20Simulaci%C3%B3n%20de%20entornos%20con%20robots%20manipuladores%20m%C3%B3viles.pdf?sequence=2, visitado 2017, septiembre 19.

[11] OpenCV Version 3.0, ”Getting Start with image”, API Docs, Disponible en: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_image_display/py_image_display.html, visitado 2017, septiembre 19.

ANEXOS

Anexo 1

Instalación Software y Librerías

1. **Descargar V-REP PRO EDU V3.3.2 rev3 desde la url:**

   http://www.coppeliarobotics.com/downloads.html

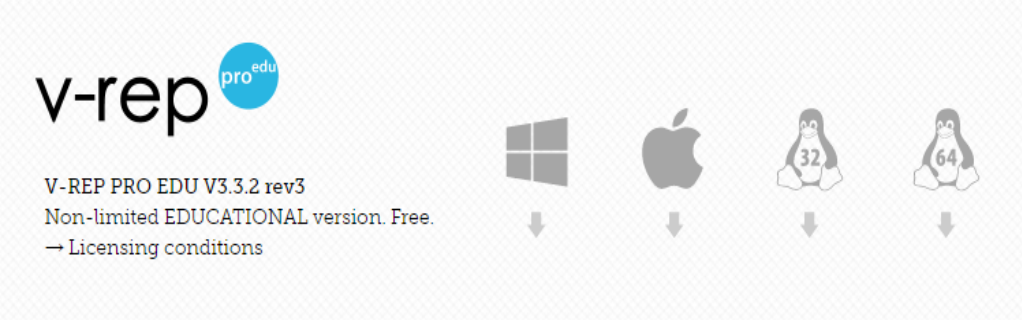
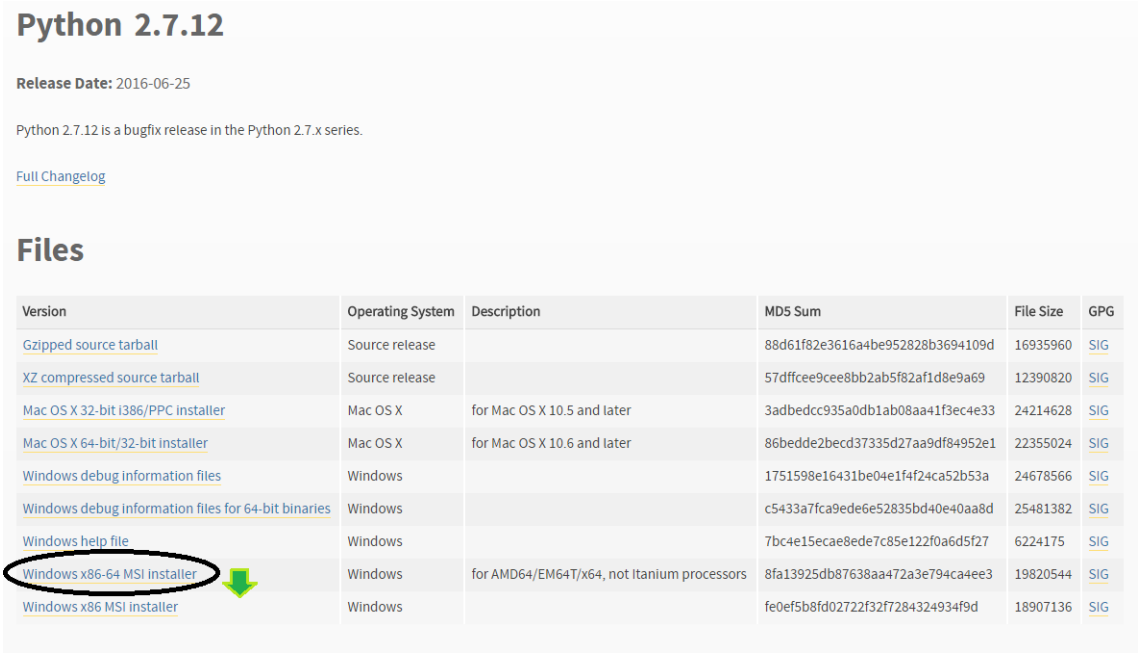


Figura Anexo 1.1

Una vez descargado proceder con la instalación.

2. **Descargar Python 2.7.12 desde la url:**
   https://www.python.org/downloads/release/python-2712/



Python 2.7.12

Release Date: 2016-06-25

Python 2.7.12 is a bugfix release in the Python 2.7.x series.

Full Changelog

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		88d61f82e3616a4be952828b3694109d	16935960	SIG
XZ compressed source tarball	Source release		57dffcee9cee8bb2ab5f82af1d8e9a69	12390820	SIG
Mac OS X 32-bit i386/PPC installer	Mac OS X	for Mac OS X 10.5 and later	3adbedcc935a0db1ab08aa41f3ec4e33	24214628	SIG
Mac OS X 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	86bedde2becd37335d27aa9df84952e1	22355024	SIG
Windows debug information files	Windows		1751598e16431be04e1f4f24ca52b53a	24678566	SIG
Windows debug information files for 64-bit binaries	Windows		c5433a7fca9ede6e52835bd40e40aa8d	25481382	SIG
Windows help file	Windows		7bc4e15ecae8ede7c85e122f0a6d5f27	6224175	SIG
Windows x86-64 MSI installer	Windows	for AMD64/EM64T/x64, not Itanium processors	8fa13925db87638aa472a3e794ca4ee3	19820544	SIG
Windows x86 MSI installer	Windows		fe0ef5b8fd02722f32f728432493 4f9d	18907136	SIG

Figura Anexo 1.2

Una vez descargado proceder a instalar.

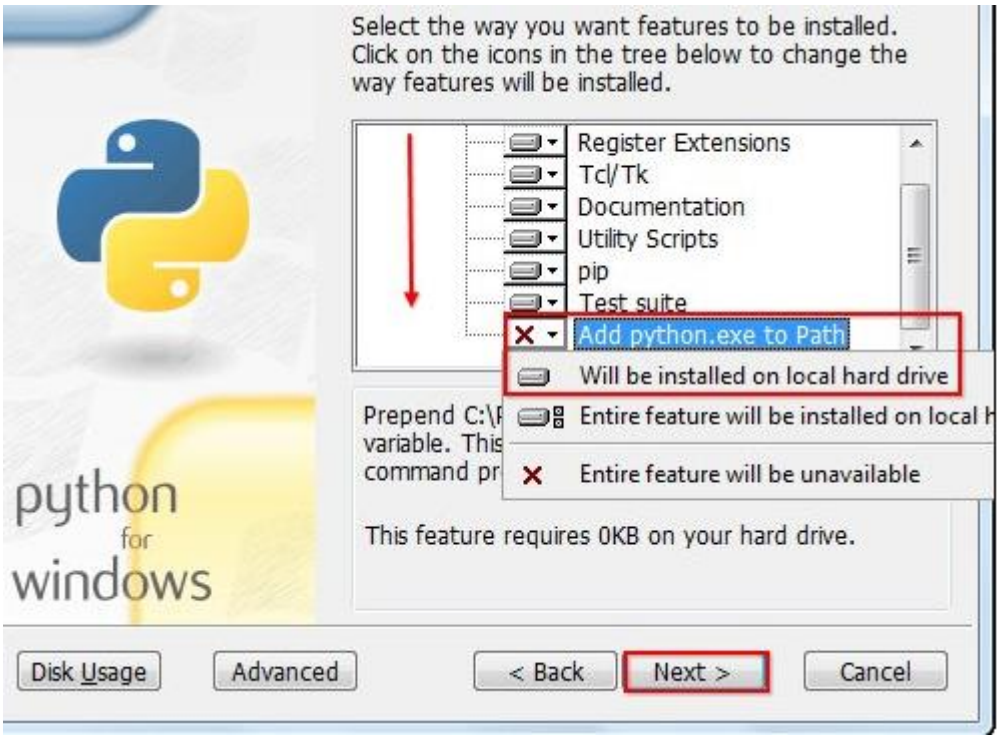


Figura Anexo 1.3

Añadir variables del sistema.

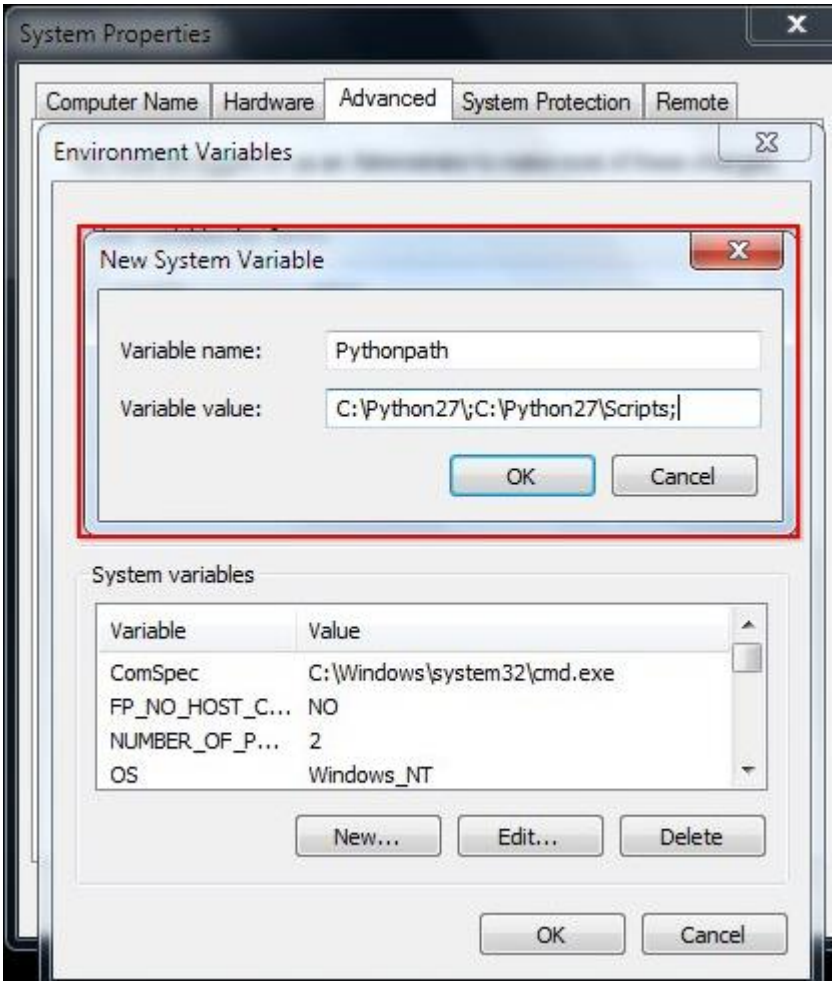


Figura Anexo 1.4

3. **Descargar la librería OpenCV 3.1.0 desde la url:**

   http://ufpr.dl.sourceforge.net/project/opencvlibrary/opencv-win/3.1.0/opencv-3.1.0.exe

   Una vez descargado proceder con la instalación de la librería OpenCV.
   Despues de haber terminado la instalación ir a la ruta opencv/build/python/2.7, copiar cv2.pyd y pegarlo en la ruta C:/Python27/lib/site-packages.

4. **Descargar Numpy 1.12.0 desde la url:**

   https://pypi.python.org/pypi/numpy

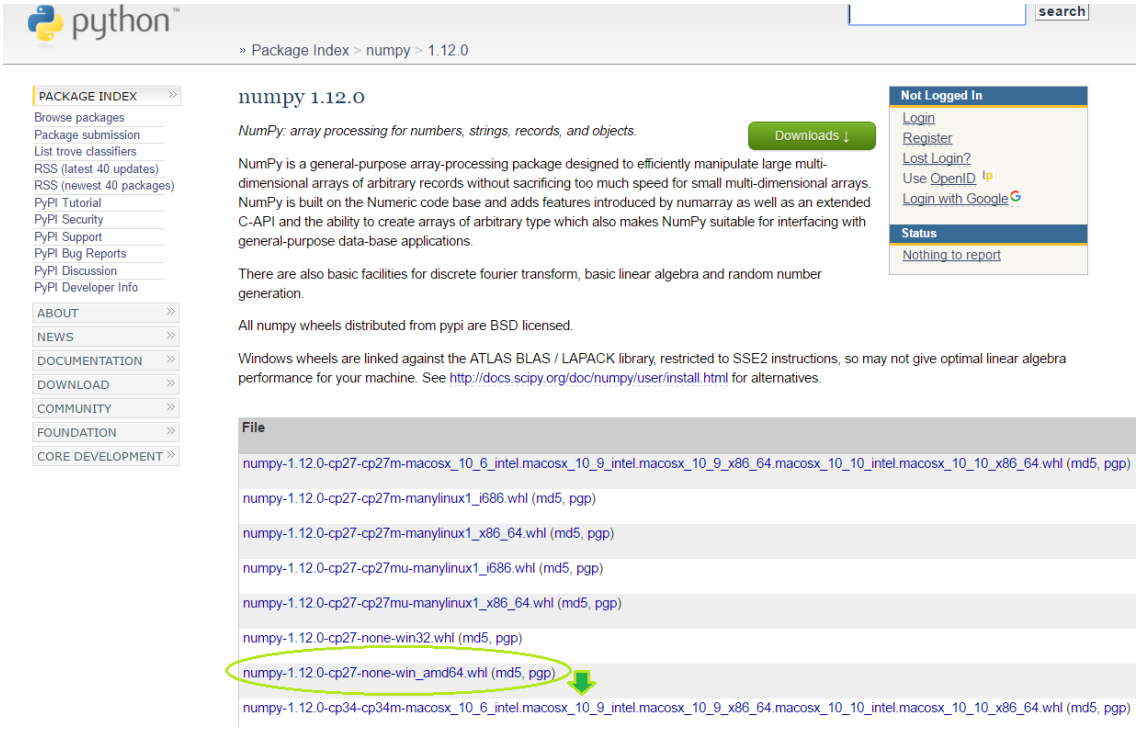


Figura Anexo 1.5

Copiar numpy-1.12.0b1-cp27-none-win_amd64.whl y pegarlo en el directorio (C:\Python27\Scripts)

Abrir cmd ir a C:\Python27\Scripts

Ejecutar pip2.7.exe install "numpy-1.12.0b1-cp27-none-win_amd64.whl"

Anexo 2

Elaboración de la Escena

Para la creación de una escena abrir el software V-REP PRO EDU**.**

Crear escena

En la creación de la escena para este proyecto nosotros trabajaremos con los siguientes elementos:

- El entorno de trabajo (Escena).
- Robot Quadricopter
- El script principal (Objetos - buques).

Al abrir el software V-REP nos presenta una escena básica con el nombre “New Scene”, procedemos a guardar la escena (Menu Bar → File → Save scene) para poder cambiarle el nombre a “Espacio Marítimo”.

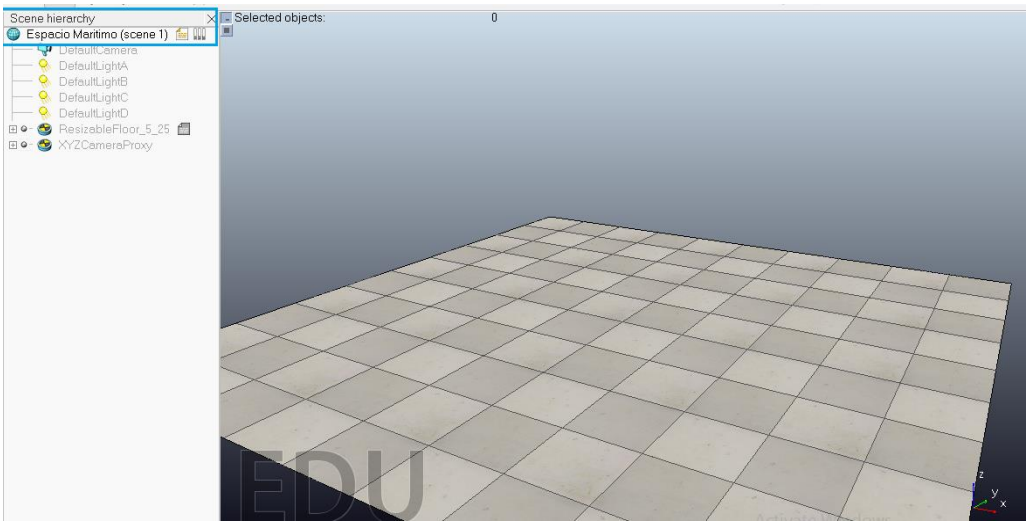


Figura Anexo 2.1

Añadir Robot Quadricopter.

Nos colocamos en la ventana (model browser → robots), arrastramos el robot Quadricopter añ entorno de trabajo.

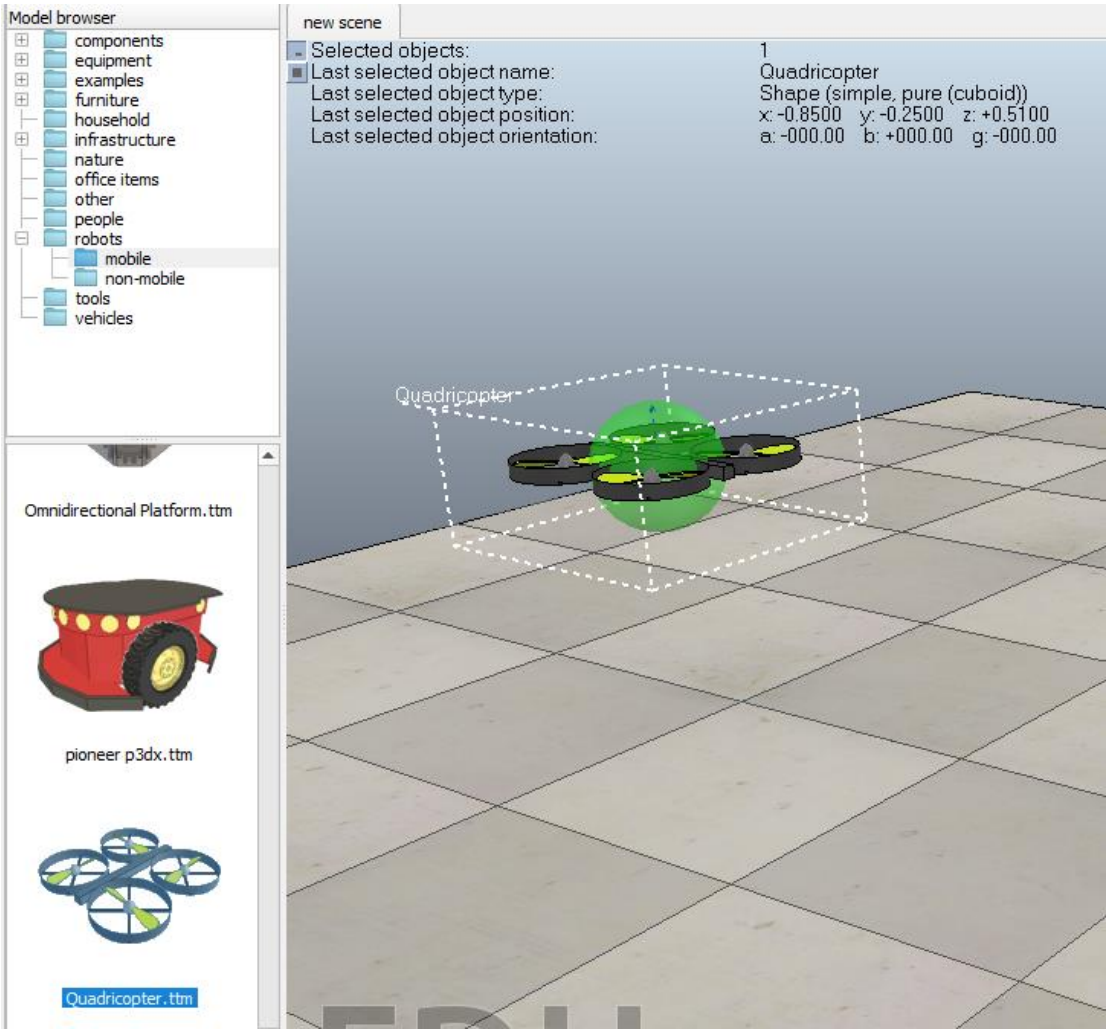


Figura Anexo 2.2

Añadir Sensores a Robot.

Se colocan los distintos sensores a usar en la ubicación de nuestra preferencia.

- Sensor de Visión.

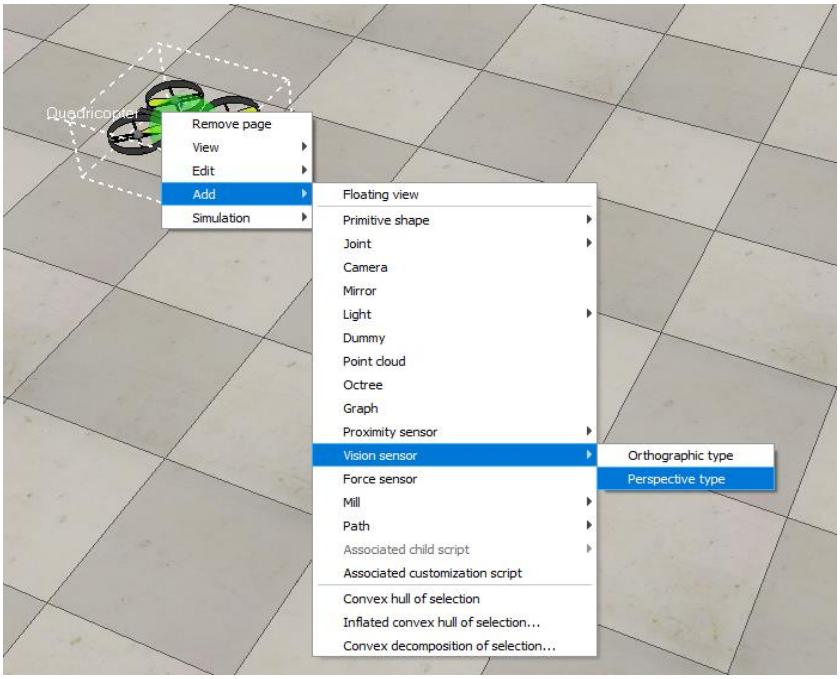


Figura Anexo 2.3

- Sensor Acelerómetro

  Para encontrar los sensores, vamos a la ventana (model browser → componets → sensors), seleccionamos el acelerómetro y lo arrastramos a la escena, estos pasos lo volvemos a repetir para agregar otros sensores como GPS y GyroSensor.

Figura Anexo 2.4

Para ubicar el acelerómetro en el Quadricopter debemos seleccionar el objeto Accelerometer agregado a la escena, y arrastrarlo dentro del objeto Quadricopter.

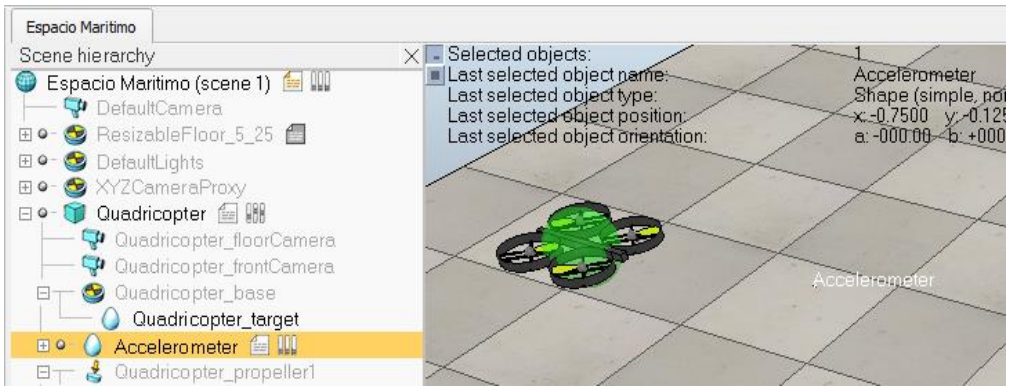


Figura Anexo 2.5

Y posteriormente se procede a colocarlos en la misma posición, seleccionando los dos objetos Quadricopter y Accelerometer presionamos el botón  Object/Item/Position/Orientation y “Apply to Selection”.

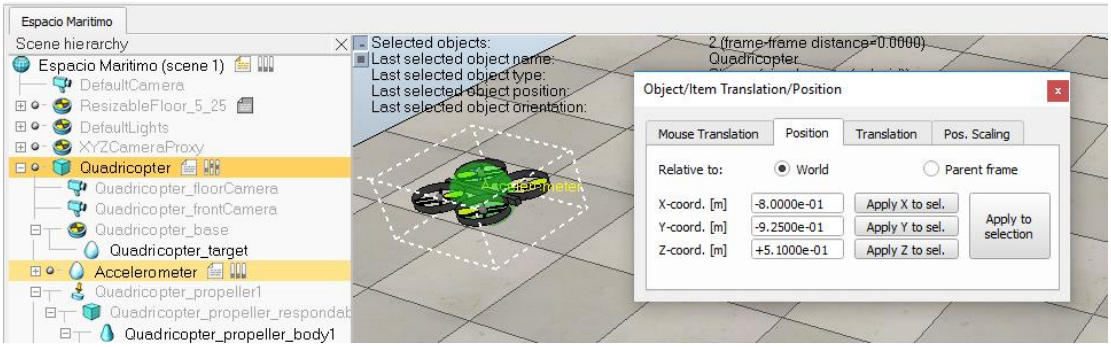


Figura Anexo 2.6

Importar/Copiar objeto (shapes) en V-REP

Para importar objetos lo haremos por la opción (Menu Bar → File → Import → Mesh…) o podemos copiar objetos de otra escena (Ctrl+c) y pegar en nuestra escena (Ctrl+v).

V-REP soporta las siguientes extensiones de archivos (OBJ, DXF, 3DS, STL, COLLADA, URDF).

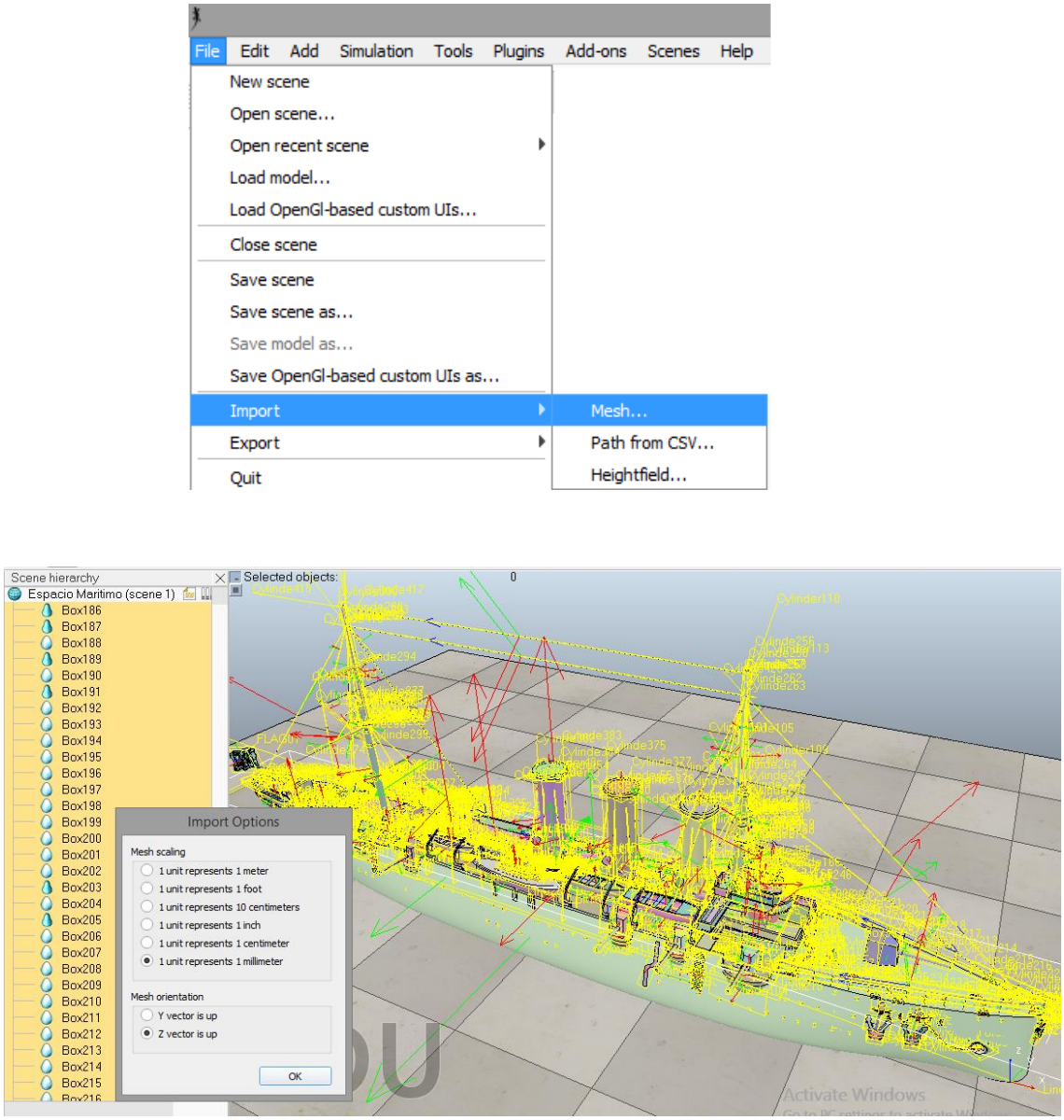


Figura Anexo 2.7

Una vez importado el objeto, por defecto aparecerá seleccionadas todas sus figuras geométricas, para mayor visibilidad en la escena la agrupamos como indica en a la Figura.

Clic derecho sobre una figura geométrica seleccionada (Edit → Grouping/Merging → Group selected shapes).

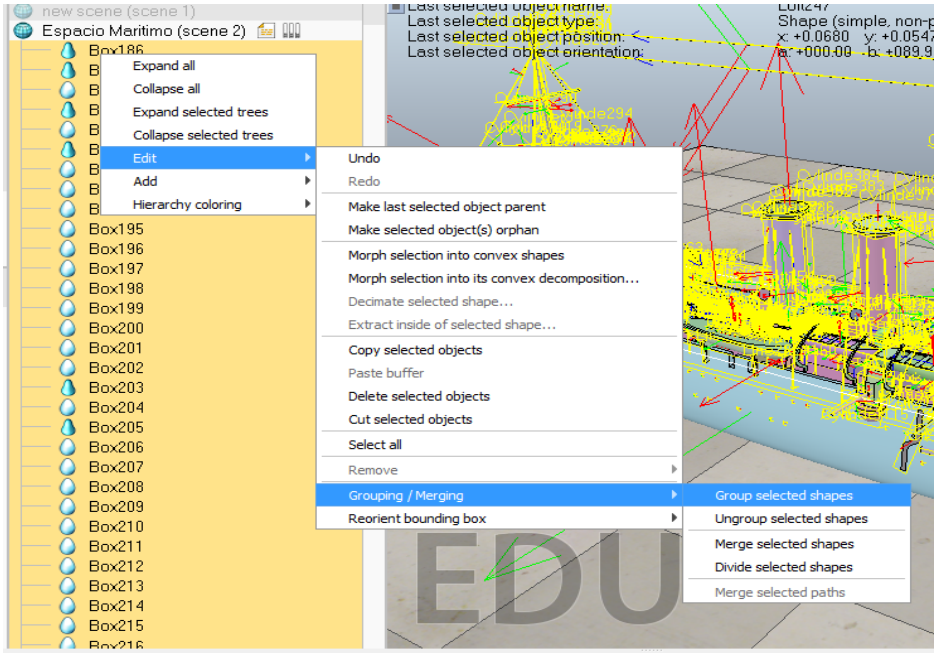


Figura Anexo 2.8

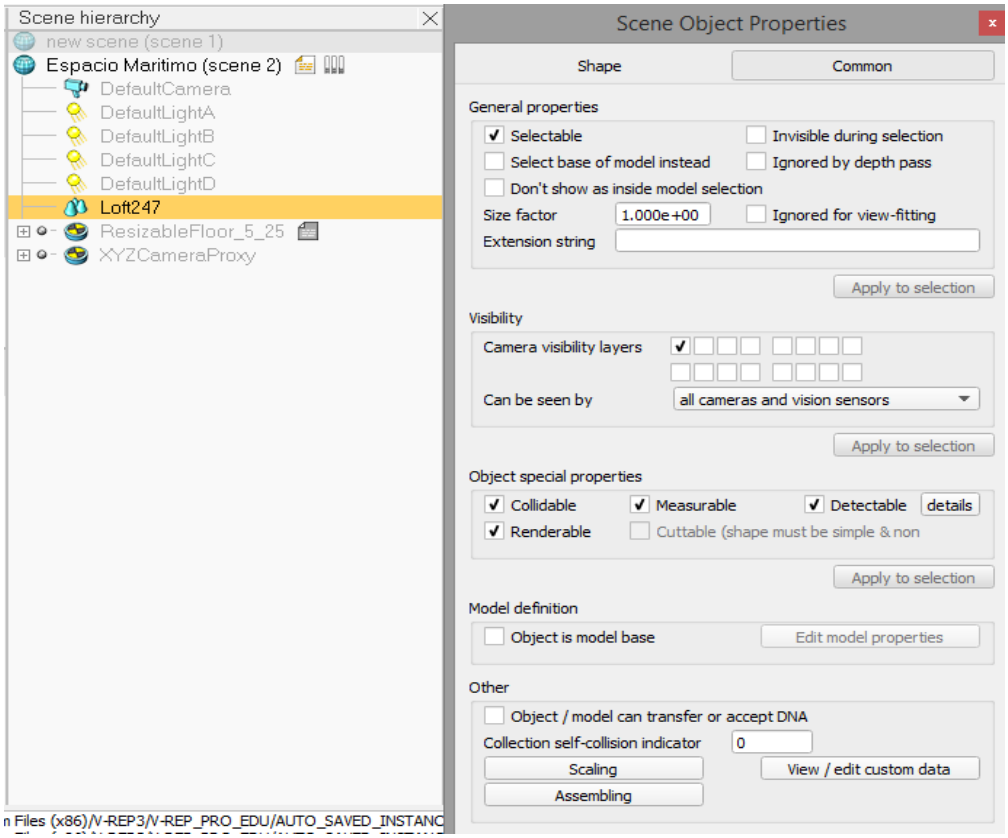
Modificar Escala del objeto

Una vez posesionado en el objeto a modificar su escala, se accede a las propiedades Común que es parte del Dialogo “Scene Object Properties”, que se encuentra en:

- (Menu Bar → Tools → Scene Object Properties).
- (Scene hierarchy) Doble clic en el icono del objeto.

- Clic en el botón  ToolBar.

En el dialogo “Scene Object Properties” clic en la pestaña Common el dialogo muestra los parámetros y ajuste del objeto seleccionado.



. **Figura Anexo 2.9**

En la parte inferior del dialogo clic en el botón “Scaling”, presentara el dialogo “Object / Model Scaling”. El parámetro “Scaling factor” indica que factor de escala se aplicara al objeto una vez presionado el botón “Ok” se aplicara los cambios a todas las dimensiones del objeto.

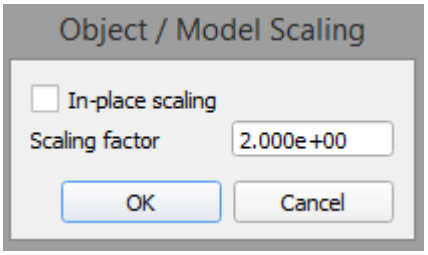


Figura Anexo 2.10

Crear Path

Para la crear un “Path” clic derecho sobre la escena (Add → Path → Semegment type).

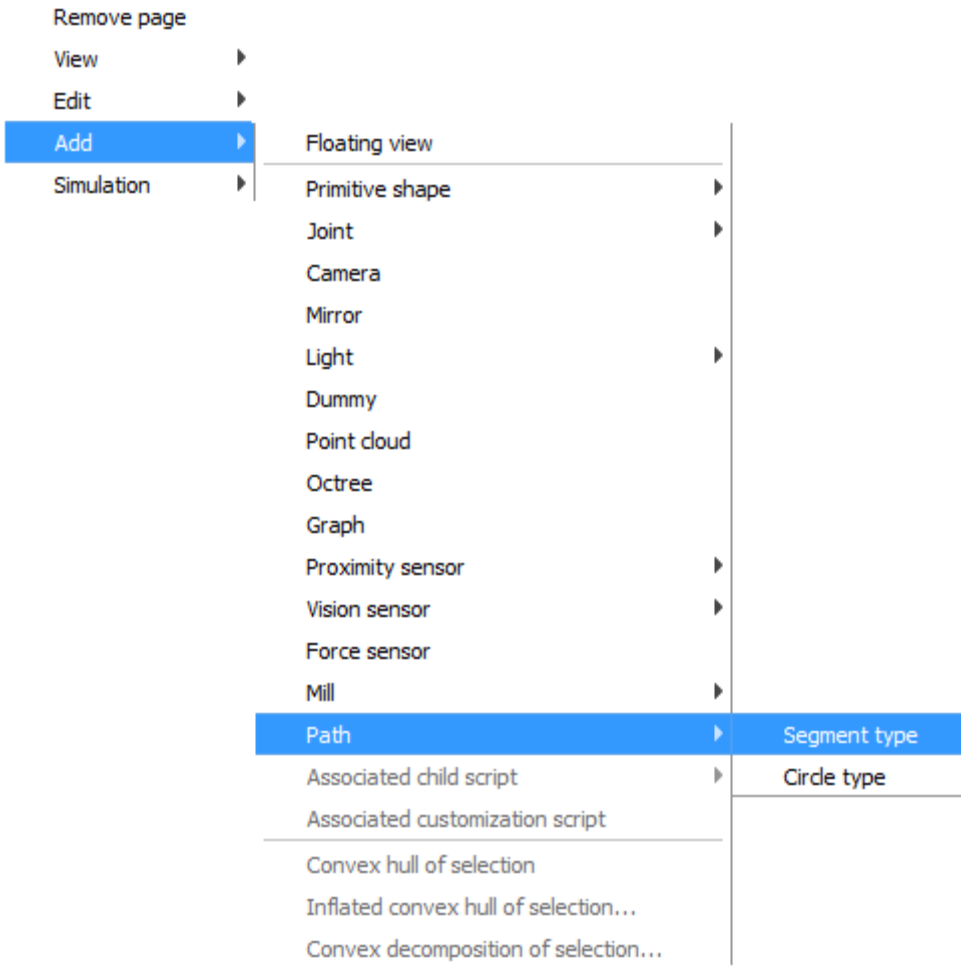


Figura Anexo 2.11

En la escena se crea un Path con 2 puntos, para editar el Path abrir el dialogo “Path edition” donde se puede agregar puntos y dar curvas en el path.

Para agregar un nuevo punto, seleccionar un punto del path en (Scene hierarchy) el cual se pondrá blanco en la escena, clic derecho sobre el punto seleccionado en (Scene hierarchy) y dar clic en la opción (Insert new control point after selection), se agregará el punto en la misma coordenada del punto seleccionado, para mover el punto y dar curva al path seleccionar el nuevo punto, ir a la escena dar clic sobre el punto blanco tener presionado el clic y mover con el mouse a la nueva coordenada.

Nota: Para poder mover un punto a una nueva coordenada deben tener abierto el dialogo “Object/Item/Position/Orientation”.

Para cerrar el path debemos cambiar los parámetros del dialogo “Path Edition” ver.

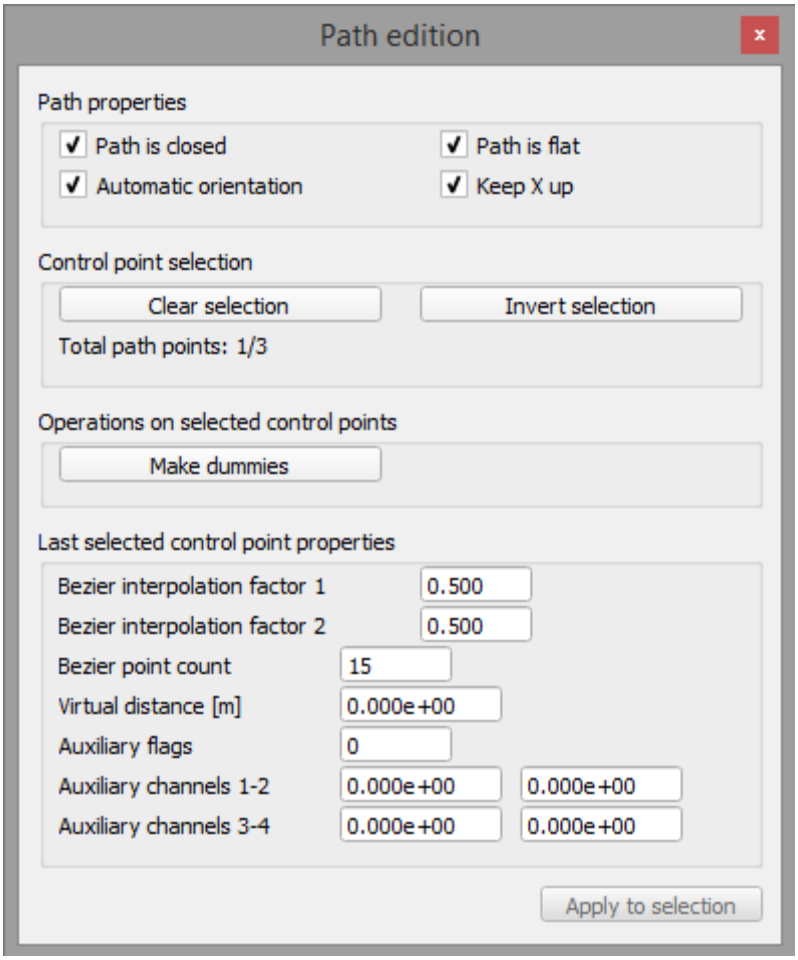


Figura Anexo 2.12

Anexo 3

Conexión entre Python y V-REP.

1. Instalar el software PyScripter.
2. Crear carpeta de trabajo del Proyecto.
3. Ir a la ruta “C:\Program Files\V-REP3\V-REP_PRO_EDU\programming \remoteApiBindings\python\python”, se copia el contenido de la carpeta python a la carpeta de trabajo del proyecto.

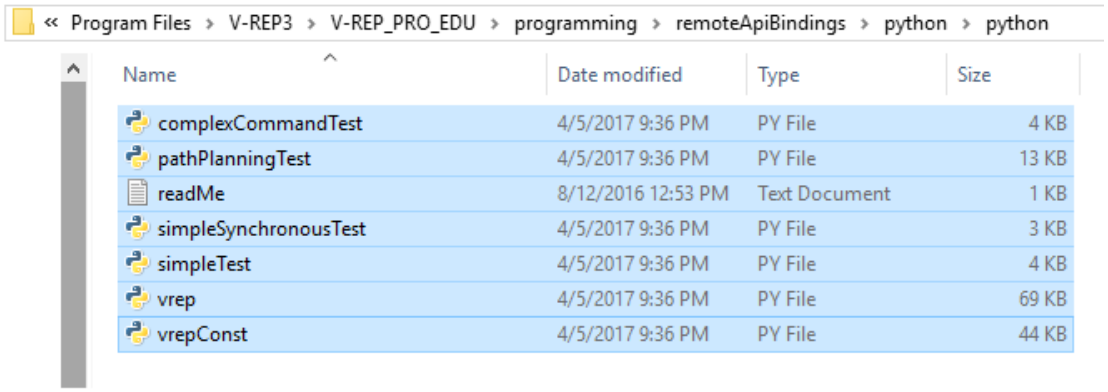


Figura Anexo 3.1

4. Ir a la ruta “C:\Program Files\V-REP3\V-REP_PRO_EDU\programming \remoteApiBindings\lib\lib”, se copia el “remoteApi.dll” de la carpeta 32bits o 64bits según el Sistema Operativo y se lo coloca en la carpeta de trabajo del proyecto.

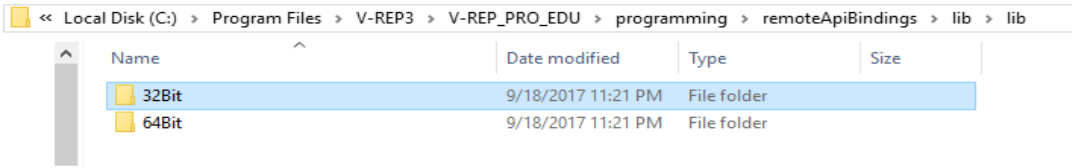


Figura Anexo 3.2

5. Para conectar el programa a desarrollar con V-REP se necesita conocer el valor del puerto que se encuentra en el archivo “remoteApiConnections” de la ruta “C:\Program Files\V-REP3\V-REP_PRO_EDU”.

   portIndex1_port = 19997
   portIndex1_debug = false
   portIndex1_syncSimTrigger = true

6. El valor definido de “portIndex1_port” será el puerto el puerto de conexión V-REP, el cual, debe ser cambiado en la línea de código.

```
clientID=vrep.simxStart('127.0.0.1',19997,True,True,5000,5) # Connect to V-REP
```

   para su correcto funcionamiento.

7. Se procede a correr la simulación en V-REP y después en Python.