ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Ingeniería Mecatrónica

DEPARTAMENTO DE ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA EN MECATRÓNICA

"INVESTIGACIÓN E IMPLEMENTACIÓN DE UN SISTEMA DE SEGURIDAD FIJO Y MÓVIL MEDIANTE UN DRONE, USANDO VISIÓN ARTIFICIAL PARA DETECCIÓN Y SEGUIMIENTO DE PERSONAS EN UN AMBIENTE EXTERNO ESPECIFICO, DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE-L."

LILIA MARISOL MORALES CACPATA
CARLOS MANUEL PAUCAR TIPAN

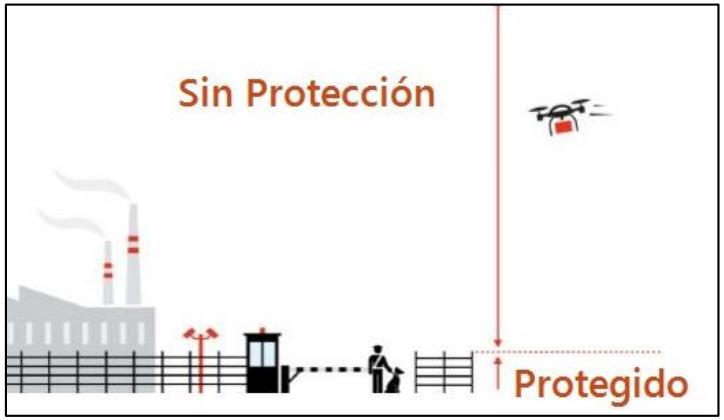
TUTOR: ING. DARIO MENDOZA CHIPANTASI

INTRODUCCIÓN

La tecnología ha ido evolucionando en varias ramas, entre estas es la visión artificial y la robótica, para lo cual se ha realizado varias investigaciones para que trabajen en conjunto. En este proyecto se trabajará en un sistema de seguridad, que detecte personas y realice el seguimiento con un vehículo aéreo no tripulado, cuyo parámetro más relevante es el uso de visión artificial, la cual está encargada de la detección de personas con la integración de las librerías de OpenCV y ROS (Robot Operating System), como interfaz de comunicación y procesamiento de control de el vehículo aéreo no tripulado.

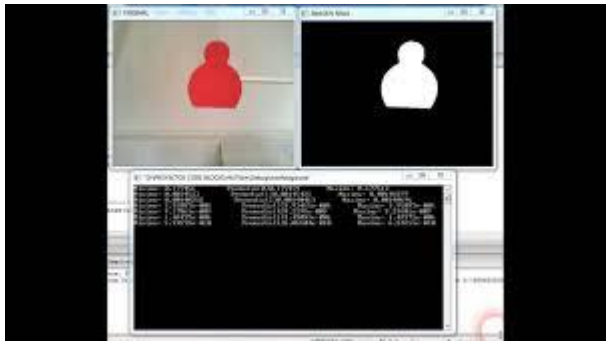
ANTECEDENTES

Actualmente el uso de visión artificial para la seguridad de lugares ya sean públicos o privados, se están afianzando como una tecnología imprescindible en los sistemas de vigilancia gracias a las posibilidades de obtener la máxima información del objeto, por medio de la adquisición y procesamiento de imágenes, recibiendo datos como la trayectoria seguida, velocidad, coordenadas de su última posición entre otros, el caso más común del uso de estos sistemas es en el control de tráfico (ITS), sistemas de seguridad en entornos militares, espacios públicos, todo esto tiene como finalidad brindar seguridad a las personas que habitan estos medios.



ANTECEDENTES

Actualmente en el país no existen empresas comerciales que ofrezcan este tipo de sistema de seguridad, sin embargo existen varios proyectos de investigación del control de vuelo de drone, pero enfocado a detección y seguimiento de colores o formas



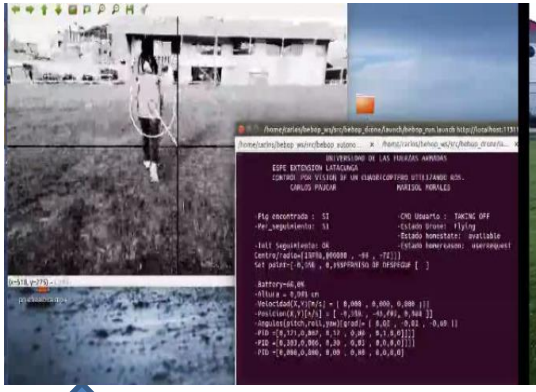
JUSTIFICACION

El presente proyecto nace debido al aumento de la inseguridad en la mayor parte de nuestro país, con lo que se busca minimizar la inseguridad en diferentes entornos, al proporcionar una herramienta tecnológica fácilmente reproducible al usar software libre. (CEDATOS, 2016)

El proyecto contempla un concepto del uso de visión artificial enfocado a la seguridad y defensa, con un especial énfasis en la investigación de detección y seguimiento de personas, que interrumpan un espacio delimitado, usando hardware y software que trabajando en conjunto se puede generar algoritmos de control, para realizar un óptimo seguimiento de personas para poder brindar seguridad.

PROPUESTA

Mejorar la seguridad de un área delimitada al implementar un sistema de visión artificial que usa en conjunto una cámara fija y otra móvil, la cual se encuentra incorporada en el drone, para detección y seguimiento de personas sospechosas.



OBJETIVO GENERAL

- Investigar e implementar un sistema de seguridad fijo y móvil mediante un drone, usando visión artificial para detección y seguimiento de personas en un ambiente externo especifico, de la Universidad de las Fuerzas Armadas ESPE-L.

OBJETIVOS ESPECÍFICOS

- Investigar técnicas para el procesamiento y análisis de imágenes, para detección de personas en movimiento en un área fija, mediante una cámara de seguridad estática.
- Indagar técnicas para el procesamiento de imágenes, detección de personas en movimiento, a una determinada altura mediante una cámara incorporada en un drone (cámara móvil).
- Desarrollar un algoritmo de control de vuelo de un drone para el seguimiento de una persona en movimiento, desde cierta altura.
- Acoplar los algoritmos para el desarrollo del sistema total de reconocimiento y seguimiento de personas.
- Realizar pruebas de funcionamiento del sistema completo, en las instalaciones de la Universidad de las Fuerzas Armadas ESPEL

INVESTIGACIONES DE SEGURIDAD CON CUADRICÓPTERO

Una de las aplicaciones otorgadas a los drones es la seguridad y vigilancia perimetral, ya que las aeronaves no tripuladas ofrecen un punto de vista único con un coste reducido en sus operaciones.

- El Ayuntamiento de Algemesí (Valencia-España) y la Universidad Politécnica de Valencia realiza desde 2016 un proyecto de vigilancia sobre los campos mediante aeronaves por control remoto

INVESTIGACIONES DE SEGURIDAD CON CUADRICÓPTERO

- La empresa americana Autonomy desarrollo un drone que esta programado para patrullar un área de forma automática, así como detectar y abordar cualquier persona que entra en un área sin autorización

- Rastreo y seguimiento de objetos basado en visión sin GPS por vehículos aéreos no tripulados

ANÁLISIS DE INVESTIGACIONES PRESENTES Y FUTURAS DE SEGURIDAD CON DRONES

- Las investigaciones dentro del ámbito comercial o académico no se han hecho esperar, encontrando varios proyectos futuristas, que parten desde una plataforma móvil comercial como es el Parrot AR Drone o desde plataformas diseñadas que se equipan con sensores y placas de control.

ANÁLISIS DE INVESTIGACIONES PRESENTES Y FUTURAS DE SEGURIDAD CON DRONES

- Las características mejoradas que Parrot Bebop 2 presenta, permitirá una detección y seguimiento de personas u objetos más exacta, dada su resolución de imagen, principalmente para investigaciones de visión, por lo que sería conveniente investigar y trabajar en dicha plataforma. Al acoplar con un sistema operativo de buenas prestaciones se creará un sistema de seguridad que tiende a caracterizarse por su velocidad de procesamiento de imagen, detección en tiempo real y robustez ante perturbaciones de luz y viento

DESCRIPCION Y SELECCIÓN DE COMPONENTES

DESCRIPCION Y SELECCIÓN DE COMPONENTES

- **Sistemas de seguridad**

Determinados como el conjunto de elementos, dispositivos e instalaciones encaminadas a dar resguardo y preservar la integridad tanto de personas como áreas, ante amenazas externas

- **Cuadricóptero**

Es un vehículo aéreo no tripulado que pueden elevarse y desplazarse por medio de cuatro rotores cooplanares

El cuadricóptero flota o ajusta su altura mediante la aplicación del mismo empuje a los cuatro rotores

El cuadricóptero ajusta su orientación mediante la aplicación de más empuje a los rotores que giran en una misma dirección

El cuadricóptero ajusta su giro aplicando más empuje a un rotor y disminuyendo a su opuesto

SELECCIÓN DE COMPONENTES

- La tabla indica los parámetros de selección del cuadricóptero, para este proyecto.

Criterios de selección	Phanto 2	Ardrone	Bebop 2
Tiempo de uso	+	-	+
Velocidad de vuelo	+	-	+
Resolucion de cámara	0	-	0
Resistencia al viento	+	-	+
Comunicación	+	-	+
Sdk	-	+	+
Robustes estructural	+	0	+
Tamaño	-	-	+
Peso	-	+	+
+	5	2	8
-	3	6	0
Valoración final	1	-3	7

SELECCIÓN DE COMPONENTES

- **El cuadricóptero: Parrot Bebop 2**

Presenta características referentes a velocidad de vuelo tanto lineal como de ascenso, robustez estructural (fibra de carbono), con un peso de 500 gramos y una autonomía de 25 minutos, sus prestaciones le permiten volar en interiores y exteriores, filmar y tomar fotos al mismo tiempo, todo esto gracias a la cámara a bordo de 14 megapíxeles

SELECCIÓN DE COMPONENTES

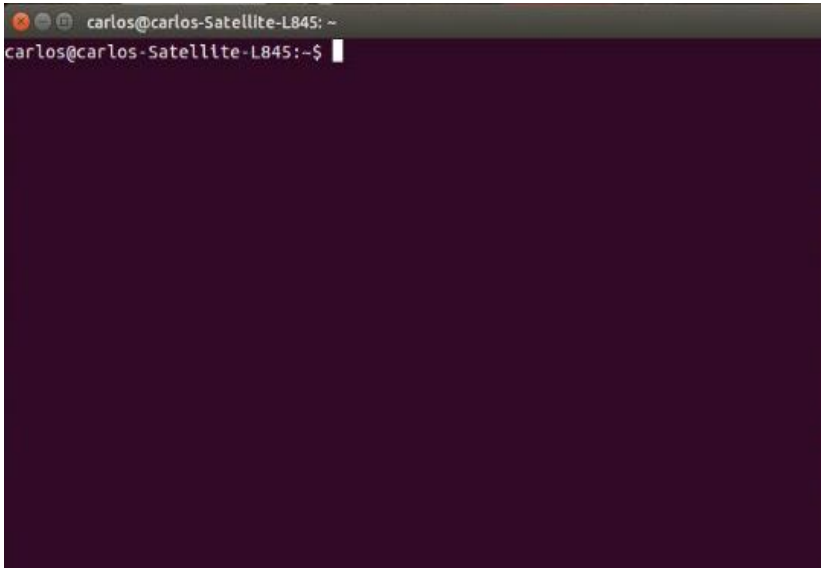
Sistema operativo

Ubuntu 14.04 LTS por ser una actualización estable y disponible con licencia libre

Criterios de selección	Windows	Linux/ ubuntu
Fiabilidad	-	+
Rendimiento	-	+
Seguridad	-	+
Drivers	+	0
Aplicaciones	+	0
Costo de licencia	-	+
+	4	5
-	3	1
Valoración final	1	4

SELECCIÓN DE COMPONENTES

- El uso del terminal es de gran ayuda para el desarrollo de este proyecto, esto es gracias a una de las características de este sistema operativo.



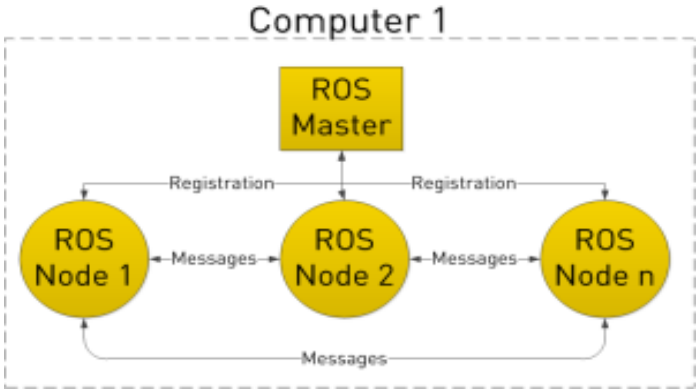
DESCRIPCION DE COMPONENTES

- **ROS (Robot Operating System)**

Sistema operativo de código abierto para desarrollar aplicaciones para robots o plataformas de investigación. Similar a un sistema operativo, permite manipular hardware, controlar de dispositivos de bajo nivel, implementación de la funcionalidad de uso común, mediante el uso de herramientas y bibliotecas para la obtención, construcción, escritura y ejecución de código en varios equipos.

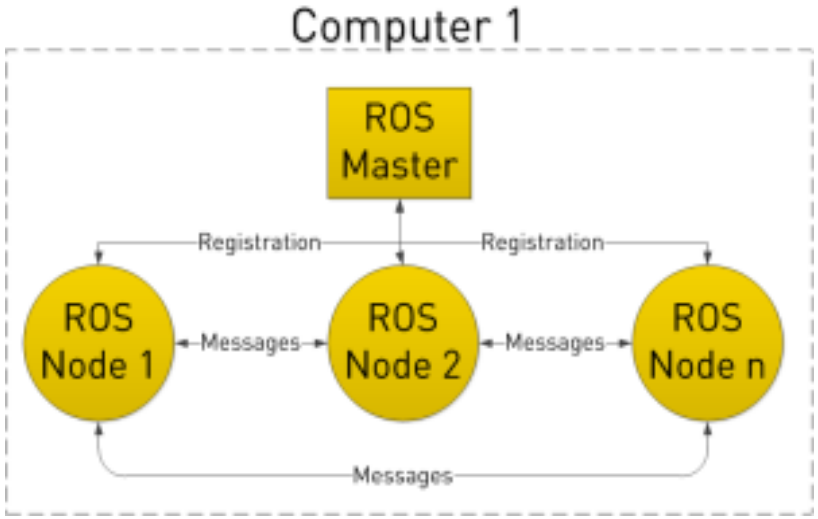
DESCRIPCION DE COMPONENTES

ROS basado en una arquitectura de grafos donde el procesamiento toma lugar en los nodos que pueden recibir, mandar y multiplexar mensajes de sensores, control, estados, planificaciones y actuadores, entre otros. La librería está orientada para un sistema Linux.



DESCRIPCION DE COMPONENTES

El procesamiento se da en los nodos que reciben, mandan y multiplexan mensajes de sensores, control, estados, planificaciones y actuadores, entre otros.



DESCRIPCION DE ELEMENTOS ROS

Conceptos de ROS

- **Paquetes (*Package*):** unidad principal para la organización del software en ROS, puede contener procesos de ejecución llamados nodos (1 o más), bibliotecas ROS, bases de datos, archivos de configuración, entre otros. Los paquetes son el elemento de construcción más importante en ROS.
- **Maestro (*Master*):** Permite el registro de nombres para que el resto de los archivos puedan encontrarlos, es decir, sin el Maestro, los nodos no podrían encontrar cada uno de los mensajes de otro o invocar servicios.
- **Nodos (*Node*):** Proceso de ejecución dentro del sistema; comparten información entre ellos para realizar una ejecución compleja, pudiendo haber varios nodos en el sistema de control de un robot
- **Temas (*Topics*):** Son canales de información entre nodos, que contienen los mensajes y los enrutan a través de un sistema de transporte con publicación/suscripción semántica.

DESCRIPCION DE ELEMENTOS ROS

- **Mensajes**: Estructura de datos, que permiten la comunicación entre nodos y contienen tanto el tipo de datos y el mensaje que se envía. Los tipos estándar: enteros, punto flotante, booleanos, etc., así como arreglos, se pueden usar, se pueden incluir estructuras y arreglos C arbitrariamente anidados.
- **Message** (**msg**) **types**: Definen la estructura de datos de los mensajes que se enviaran en ROS, se almacenan en my_package/msg/MyMessageType.msg.
- **Paquete manifiest**: Contiene la información del paquete creado tales como: nombre, versión, descripción, información de la licencia, dependencias, y otra información. Dicha información se encuentra en el archivo package.xml.

DESCRIPCION DE ELEMENTOS

- **Espacio de trabajo (*Workspace*)**

Espacio para los paquetes que se usen, sean creados, modificados o instalados.

BENEFICIOS DE ROS

- Creado y diseñado para facilitar el intercambio de software entre los aficionados y profesionales de robótica en todo el mundo debido a su enfoque didáctico y abierto, lo que ha permitido la construcción de una gran comunidad de colaboradores, así como una plataforma de investigación para grandes proyectos.
- ROS permite el uso de distintos lenguajes de programación: forma oficial soportan Python, C++.
- ROS puede ser ejecutado sobre máquinas Ubuntu y Windows, y otras plataformas como Fedora, Gentoo, etc

DESCRIPCION DE COMPONENTES

- **Paquete Bebop Drone autonomy**

controlador ROS creado específicamente para el Parrot Bebop en sus versiones 1.0 y 2.0, por desarrolladores de la Universidad Simon Fraser, basado en Parrot ARDronSDK3. Este controlador contiene el nodo bebop_driver, los mensajes personalizados creados por este nodo y los servidores para los servicios ofrecidos

DESCRIPCION DE COMPONENTES

- **Opencv**

Es una biblioteca de código abierto más popular y avanzada para aplicaciones relacionadas con el procesamiento de imágenes en tiempo real y la visión por ordenador

Abarca muchas tareas muy básicas como: captura y pre-tratamiento de los datos de imagen, y en los algoritmos de alto nivel tareas como: extracción de características, seguimiento de movimiento, aprendizaje automático.

DESCRIPCION DE COMPONENTES

- **Estructura de OpenCv 2.4.8**

Tiene estructura modular, es decir el paquete incluye varias bibliotecas compartidas o estáticas, entre las principales para este proyecto tenemos:

Funcionalidad básica. Un módulo compacto que define las estructuras de datos básicas, incluyendo las Mat

Procesamiento de imágenes. módulo de procesamiento de imágenes que incluye el filtrado de imágenes, transformaciones geométricas de imágenes, la conversión de espacio de color, histogramas, etc.

Procesamiento de vídeo. módulo de análisis de vídeo que incluye la estimación de movimiento, la sustracción del fondo, y los algoritmos de seguimiento de objetos

features2d. Detectores de características sobresalientes, descriptores y comparadores de descriptores.

objdetect. Detección de objetos e instancias de las clases predefinidas (por ejemplo, caras, ojos, tazas, personas, vehículos, etc.).

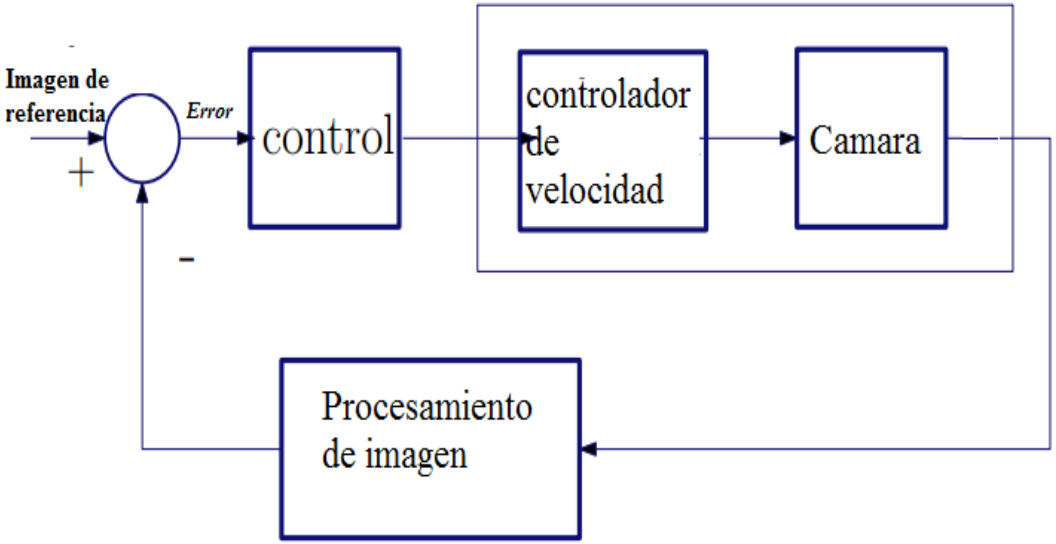
videoio. Una interfaz fácil de usar para la captura de vídeo y codecs de vídeo

DESARROLLO DEL ALGORITMO PARA PROCESAR LA IMAGEN

VISUAL SERVOING

- **Visual servoing**

Control servo visual, hace el uso de sistemas de visión por computador para controlar movimientos o acciones de un robot o plataforma



VISUAL SERVOING

IBVS (Control servo visual basado en imagen)

valores de control se obtienen de características encontradas en una imagen. Los valores se usa para estimar el movimiento del robot respecto al objetivo.

PBVS (Control servo visual basado en posición)

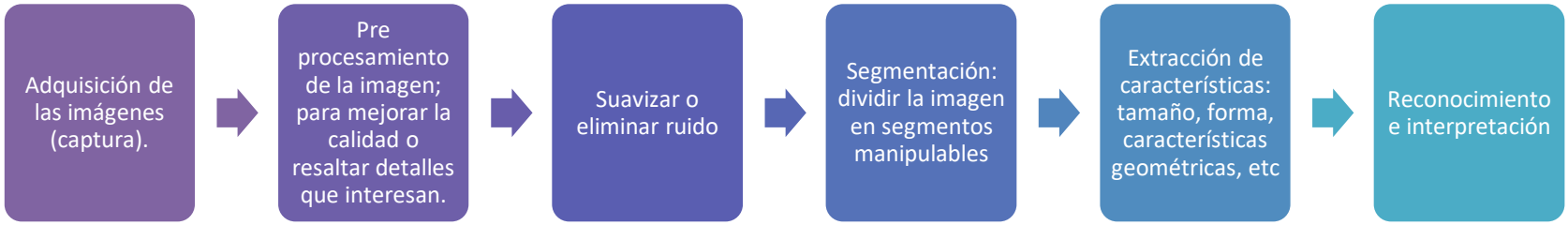
Las características son extraídas de una imagen y usadas en conjunto con un modelo geométrico del objetivo.

Recuperar la información 3D de la escena

IBS, usa únicamente imágenes en 2D y características que benefician al proyecto actual, por ejemplo: reducción del tiempo computacional

VISUAL SERVOING

Se consideran las siguientes etapas para conseguir el reconocimiento preciso del objetivo que será sometido a un proceso de seguimiento, con la manipulación de un cuadricóptero



ADQUISICIÓN DE LA IMAGEN

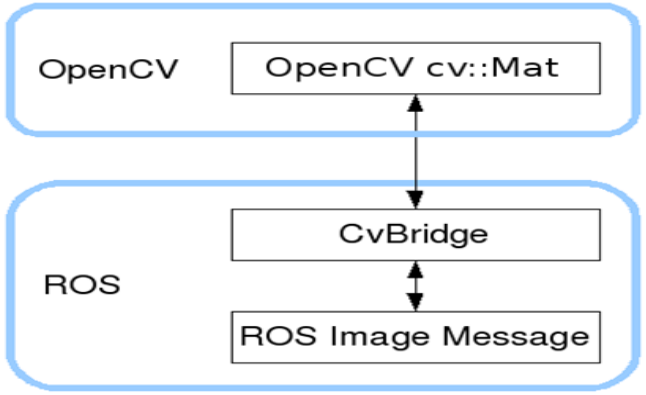
Captura de imágenes

Desde una cámara web o desde la cámara del Bebop2, inicialmente se manejan con ROS y por ende el video está en un formato de mensajes de ROS.

Conversión de imágenes ROS a OpenCV

ROS trasporta bajo su formato de mensaje sensor_msgs/image.

CvBridge que es una biblioteca de ROS que proporciona una interfaz entre ROS y OpenCv, permitiendo convertir las imágenes de ROS en OpenCv (cv:: Mat).



RECONOCIMIENTO DE OBJETOS EN IMAGEN

Visión artificial

Importante para identificación de objetos

Formas o colores

Método de identificación

Formas simples o complejas

Sistemas avanzados

Reconocer personas por sus caras o aspectos físicos

OpenCV

Librerías especializadas

Discriminar objetos o personas

Algoritmo de HOG

Robustez a cambios de luz

Definir mejor el contorno de una persona

SVM (Máquina de Soporte Vectorial)

Clasificador pre-entrenado

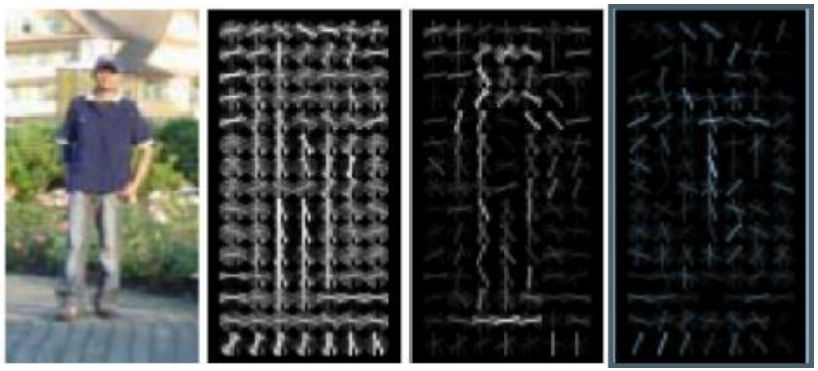
Detección de personas óptima y más aproximada

Reducir falsos positivos

DESCRIPTORES HOG (HISTOGRAMAS DE GRADIENTES ORIENTADOS)

Descriptores de HOG → Basados en la orientación del gradiente → Divide en varias celdas

Direcciones del histograma de gradiente u orientaciones de los bordes de los pixeles de las celdas



DESCRIPTORES HOG (HISTOGRAMAS DE GRADIENTES ORIENTADOS)

OBJETIVO

conseguir la máxima invariancia posible ante cambios de posición, iluminación, etc

Minimizar diferencias

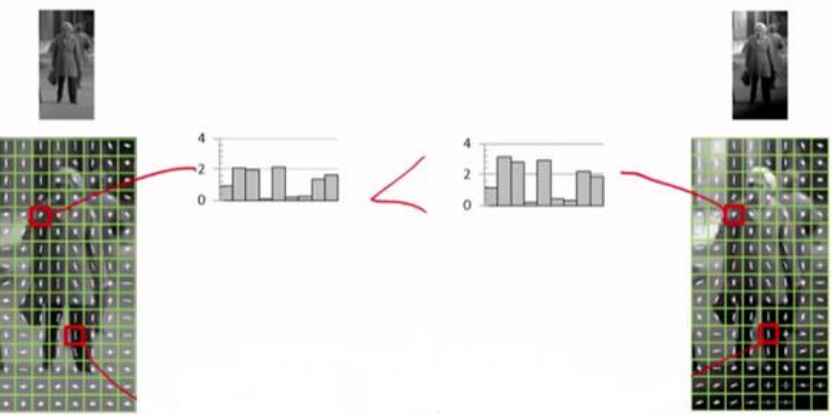
Normalizar los valores de histogramas

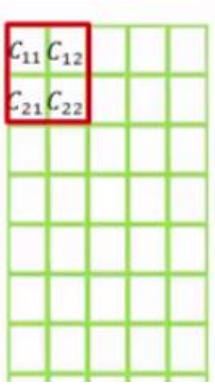
Realizado en bloques

Valores globales (gradiente) de histogramas son iguales

Descriptor final robusto

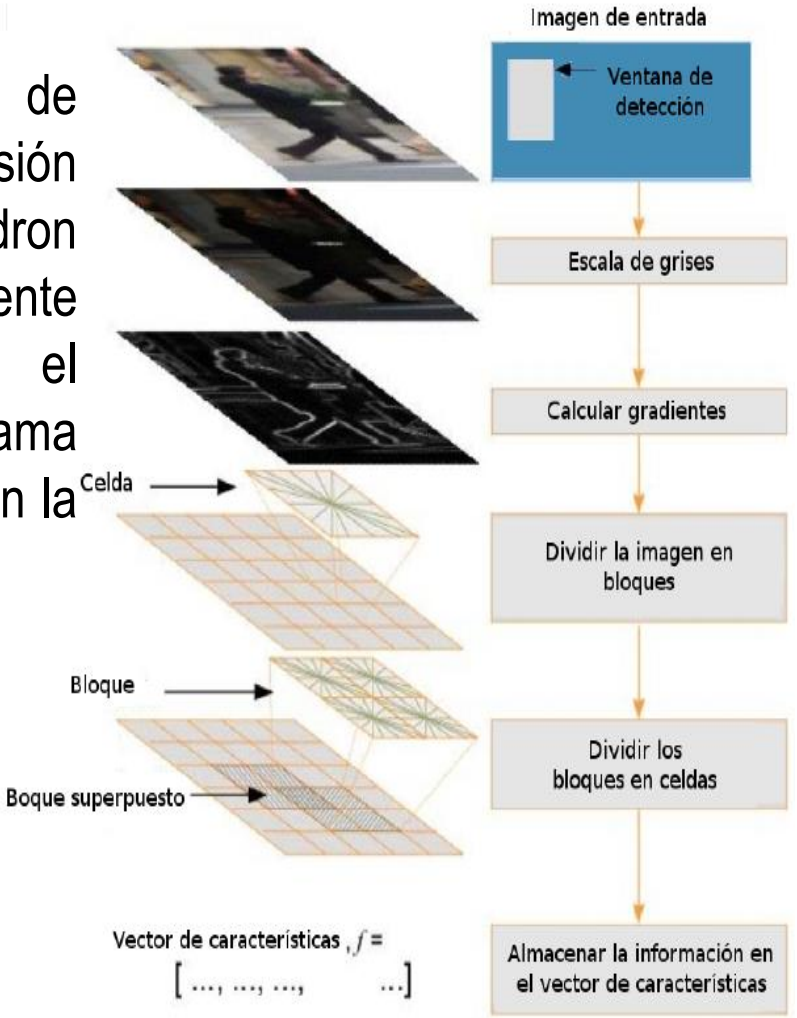
HOG proporciona los cambios de intensidad luminosa, debido a los contornos de una imagen, reconociendo la frontera de cada objeto, permitiendo aislar uno de otro





DESCRIPCIÓN BÁSICA DE HOG

HOG es capaz detectar la silueta de personas presentes en el campo de visión de la cámara. Detectado un sujeto, el dron puede ser alertado con la suficiente antelación para despegar y realizar el seguimiento, es por ello que el programa debe procesar las imágenes tomadas con la mayor brevedad posible



DESCRIPCIÓN MATEMÁTICA DE HOG

El histograma de gradientes orientados tiene como rango de valores posibles las distintas orientaciones que pueden tomar los gradientes de los píxeles, es decir los distintos grados que pueden tomar sus ángulos de gradiente. (Por ejemplo, [-90°, 90°], [0°, 180°], [0°, 360°]...). Este rango se divide en sub clases del mismo tamaño o distintos (para el rango [0°, 180°], dividiendo en nueve sub rangos: [0°, 20°), [20°, 40°)..... [160°, 180°]), y se almacena en cada uno de ellas la suma de las magnitudes de gradiente de los píxeles, cuyo ángulo de gradiente se encuentra comprendido entre esos valores.

DESCRIPCIÓN MATEMÁTICA DE HOG

- Dada una imagen A de tamaño W×H; un tamaño de celda $C_W \times C_H$ con W mod $C_W = 0$ y H mod $C_H = 0$; un tamaño de bloque en celdas $B_W \times B_H$; el ancho y alto de la imagen en celdas, W_C y H_C; y el número de bloques distribuidos horizontalmente y verticalmente, N_{BW} y N_{BH}, se calculan de la siguiente manera:

$$Wc = \frac{W}{C_W}$$
$$Hc = \frac{H}{C_H}$$

$$N_{BW} = 1 + Wc - Bw$$
$$N_{BH} = 1 + H_C - B_H$$

Y, por tanto, el número total de celdas N_C y el número total de bloques N_B resultantes de la imagen A será igual a:

$$N_C = W_C * H_C$$
$$N_B = N_{BW} * NBH$$

DESCRIPCIÓN MATEMÁTICA DE HOG

La distribución de celdas (C) y bloques (B) es la siguiente

$$A = \begin{pmatrix} a_{00} & a_{10} & \cdots & a_{(W-1)0} \\ a_{01} & a_{11} & \cdots & a_{(W-1)1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{0(H-1)} & a_{1(H-1)} & \cdots & a_{(W-1)(H-1)} \end{pmatrix}$$

$$C = \begin{pmatrix} c_{00} & c_{10} & \cdots & c_{(W_C-1)0} \\ c_{01} & c_{11} & \cdots & c_{(W_C-1)1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{0(H_C-1)} & c_{1(H_C-1)} & \cdots & c_{(W_C-1)(H_C-1)} \end{pmatrix}$$

$$B = \begin{pmatrix} b_{00} & b_{10} & \cdots & b_{(N_B W-1)0} \\ b_{01} & b_{11} & \cdots & b_{(N_B W-1)1} \\ \vdots & \vdots & \ddots & \vdots \\ b_{0(B_H-1)} & b_{1(B_H-1)} & \cdots & b_{(N_B W-1)(N_B H-1)} \end{pmatrix}$$

El descriptor de HOG calcula de forma independiente el HOG de cada celda y cada bloque agrupa los HOGs de sus celdas correspondientes.

DESCRIPCIÓN MATEMÁTICA DE HOG

El número de HOG que contiene un descriptor será

$$N_{HOG} = B_W * B_H * N_B$$

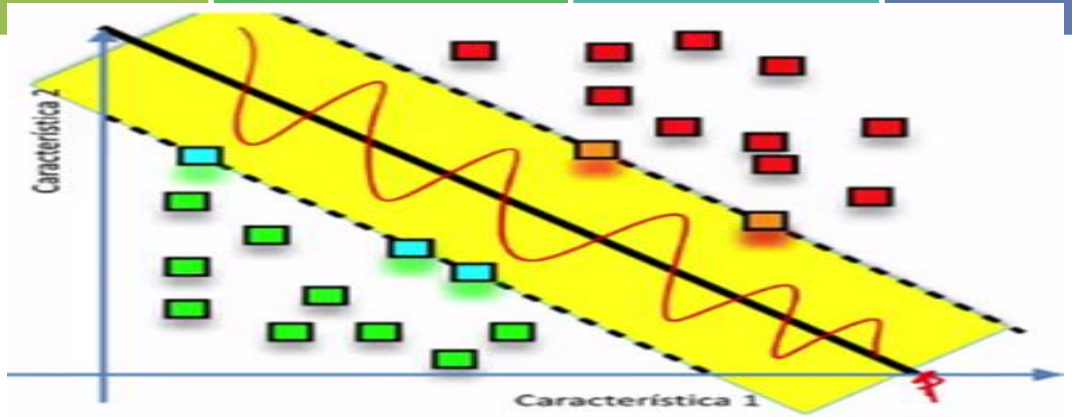
Y si se divide el HOG en n clases, dado que cada bloque contiene $B_W * B_H$ descriptores HOG, entonces el número total de valores N_V que se tomará de la imagen A será

$$N_V = n * N_{HOG}$$

Para el cálculo del modelo de detección, sobre la colección de imágenes se debe calcular el descriptor de HOG de cada imagen, etiquetando cada descriptor como positivo si es una imagen de persona (+1) o negativo si no lo es (-1).

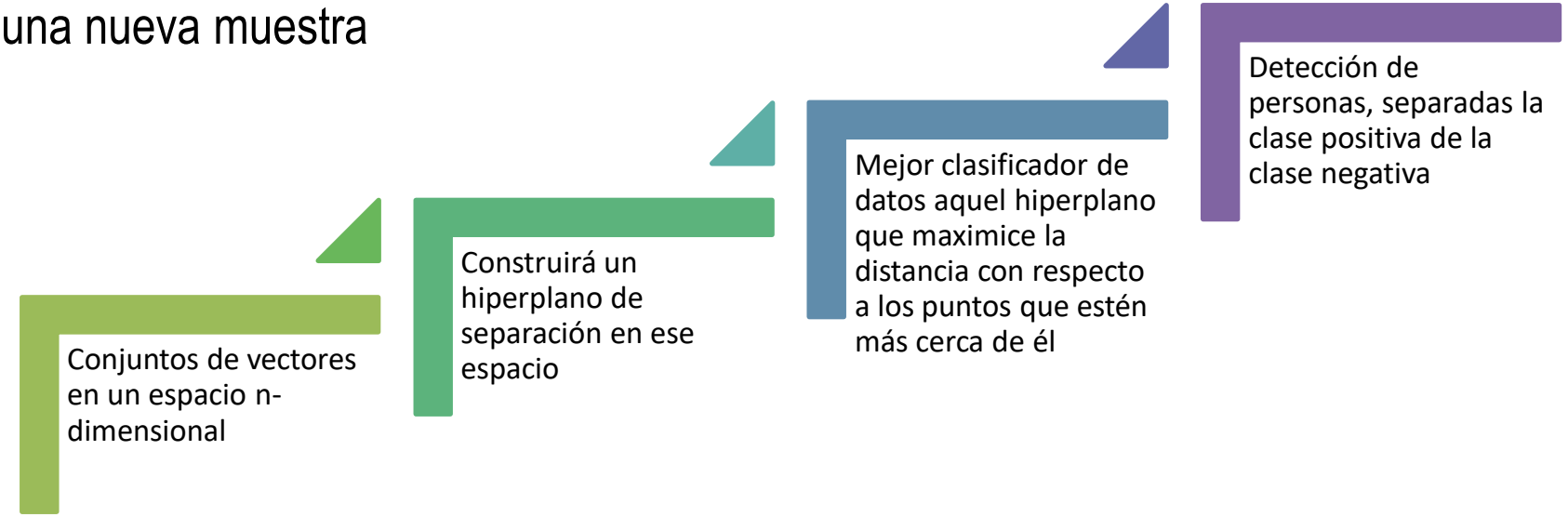
MÁQUINA DE VECTORES SOPORTE SVM (SUPORT VECTOR MACHINE)

Estructuras de aprendizaje	Transforma el espacio de entrada en otro de dimensión superior para resolver el problema mediante un hiperplano óptimo (de máximo margen).	Clasificador lineal basado en el concepto de margen máximo a partir de vectores de soporte	Margen es la región más amplia del espacio de características en el cual no existen muestras.	El plano intermedio (hiperplano) que se obtiene dentro del espacio de características, es la solución de la máquina de vectores soporte



DESCRIPCIÓN BÁSICA DE SVM

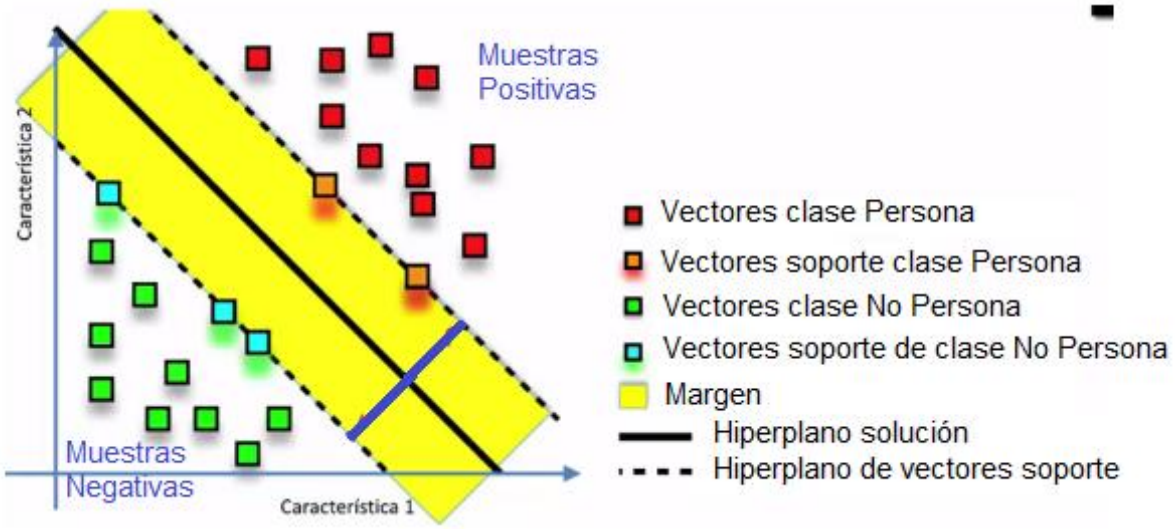
Conjunto de algoritmos de aprendizaje supervisado para la clasificación. Usando un conjunto de ejemplos de entrenamiento se puede etiquetar las clases y entrenar una SVM para construir un modelo que prediga la clase de una nueva muestra



DESCRIPCIÓN BÁSICA DE SVM

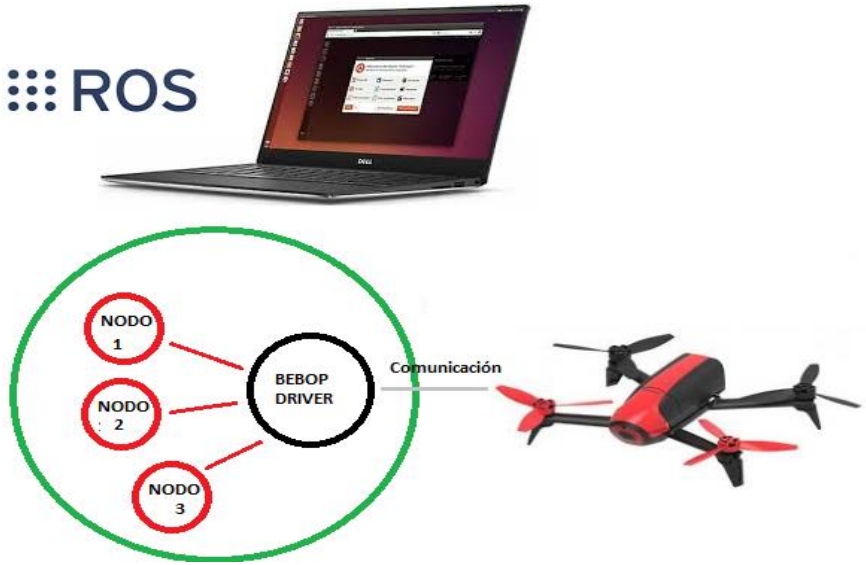
Uso de SVM

- Para detección de personas, permitirá separar la clase positiva de la clase negativa, es decir los datos que corresponden a la clase **Persona** son muestras positivas, en cambio la clase **no Persona**, representará muestras negativas



PAQUETE BEBOP AUTONOMY

El paquete bebop_autonomy es parte esencial en este proyecto porque cuenta con los controladores para la comunicación entre la plataforma ROS y el cuadricóptero Parrot Bebop2.



CONFIGURACIÓN DE PAQUETES Y LIBRERIAS

- Esto se lo hace con el fin de agregar dependencias, librerías y paquetes para el correcto funcionamiento de la aplicación, se lo hace modificando el archivo CMakeList.txt

*CMakeLists.txt ×

```
cmake_minimum_required(VERSION 2.8.3)
project(tesis_seg)
find_package(catkin REQUIRED COMPONENTS
  bebop_autonomy
  cv_bridge
  geometry_msgs
  image_transport
  opencv2
  roscpp
  sensor_msgs
  std_msgs
)
find_package(OpenCV REQUIRED)

catkin_package(
include_directories(
  ${OpenCV_INCLUDE_DIRS}
  ${catkin_INCLUDE_DIRS}
)

add_library(${PROJECT_NAME}  src/${PROJECT_NAME}/tesis_seg.cpp)

target_link_libraries(${PROJECT_NAME}_node ${catkin_LIBRARIES})
```

*package.xml ×

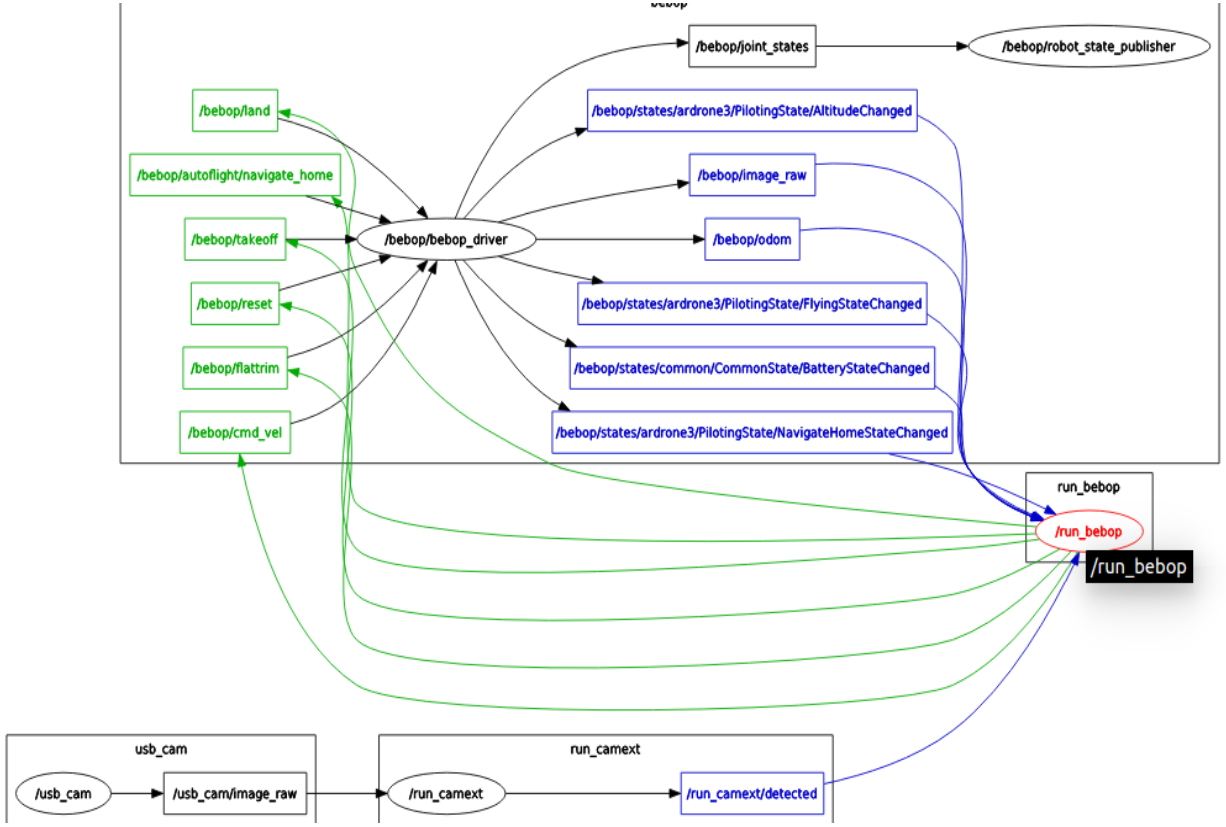
```
<?xml version="1.0"?>
<package>
  <name>tesis_seg</name>
  <version>0.0.0</version>
  <description>The tesis_seg package</description>
  <maintainer email="carlos@todo.todo">carlos</maintainer>
  <license>TODO</license>

  <buildtool_depend>catkin</buildtool_depend>
  <build_depend>bebop_autonomy</build_depend>
  <build_depend>cv_bridge</build_depend>
  <build_depend>geometry_msgs</build_depend>
  <build_depend>image_transport</build_depend>
  <build_depend>opencv2</build_depend>
  <build_depend>roscpp</build_depend>
  <build_depend>sensor_msgs</build_depend>
  <build_depend>std_msgs</build_depend>
  <run_depend>bebop_autonomy</run_depend>
  <run_depend>cv_bridge</run_depend>
  <run_depend>geometry_msgs</run_depend>
  <run_depend>image_transport</run_depend>
  <run_depend>opencv2</run_depend>
  <run_depend>roscpp</run_depend>
  <run_depend>sensor_msgs</run_depend>
  <run_depend>std_msgs</run_depend>
  <export>
  </export>
</package>
```

CONFIGURACIÓN ARCHIVO LAUNCH

- Los archivos launch son configurados para ingresar parámetros iniciales o a su vez para dar inicio a la aplicación, en este caso se configuro 3 archivos launch de inicio y uno para dar los parámetros iniciales del control PID

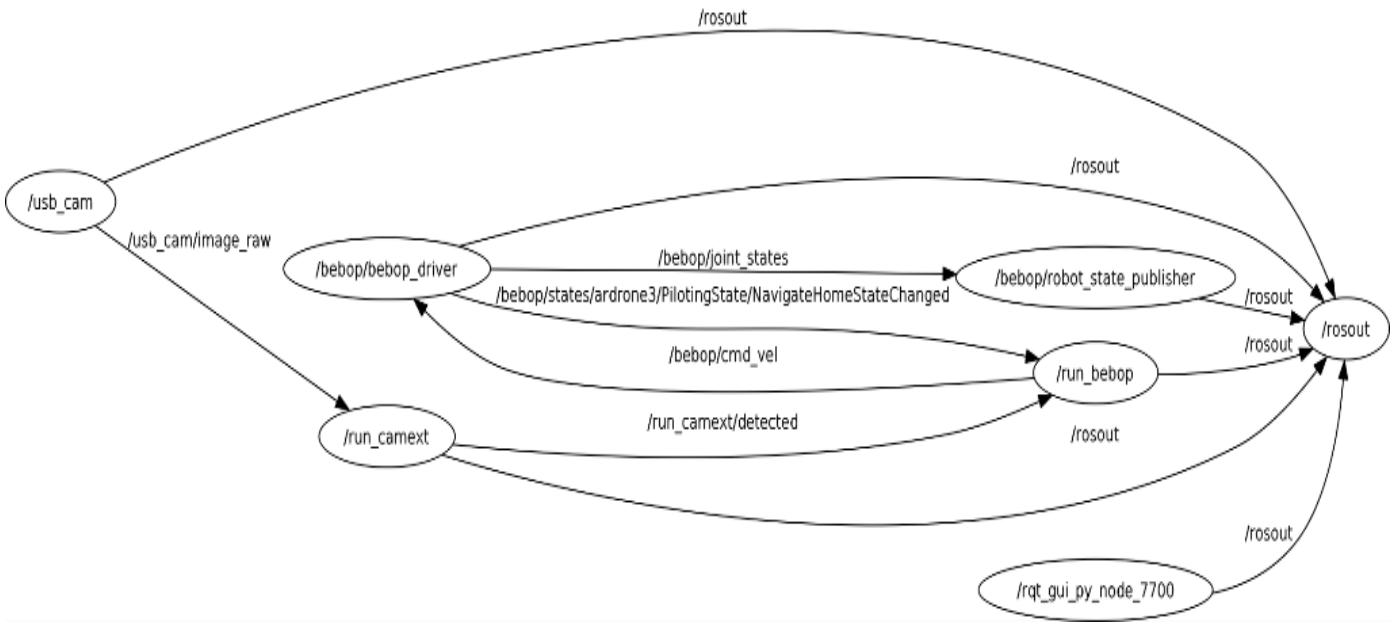
```
<!--xml-->
<launch>
      <node name ="run_bebop" pkg="bebop_drone"
type="run_bebop"  output="screen">
        <param name="pGain_roll" value="0.62" />
        <param name="iGain_roll" value="0.006" />
        <param name="dGain_roll" value="2" />
        <param name="iMax_roll" value="0.2" />
        <param name="pid_max_roll" value="0.8" />

        <param name="pGain_pitch" value="0.62" />
        <param name="iGain_pitch" value="0.006" />
        <param name="dGain_pitch" value="2" />
        <param name="iMax_pitch" value="0.2" />
        <param name="pid_max_pitch" value="0.5" />

        <param name="pGain_altura" value="0.645" />
        <param name="iGain_altura" value="0.006" />
        <param name="dGain_altura" value="0.2" />
        <param name="iMax_altura" value="0.2" />
        <param name="pid_max_altura" value="0.8" />
      </node>
</launch>
```

CONFIGURACIÓN ARCHIVO LAUNCH



EJECUCIÓN DE LA APLICACIÓN EN ROS

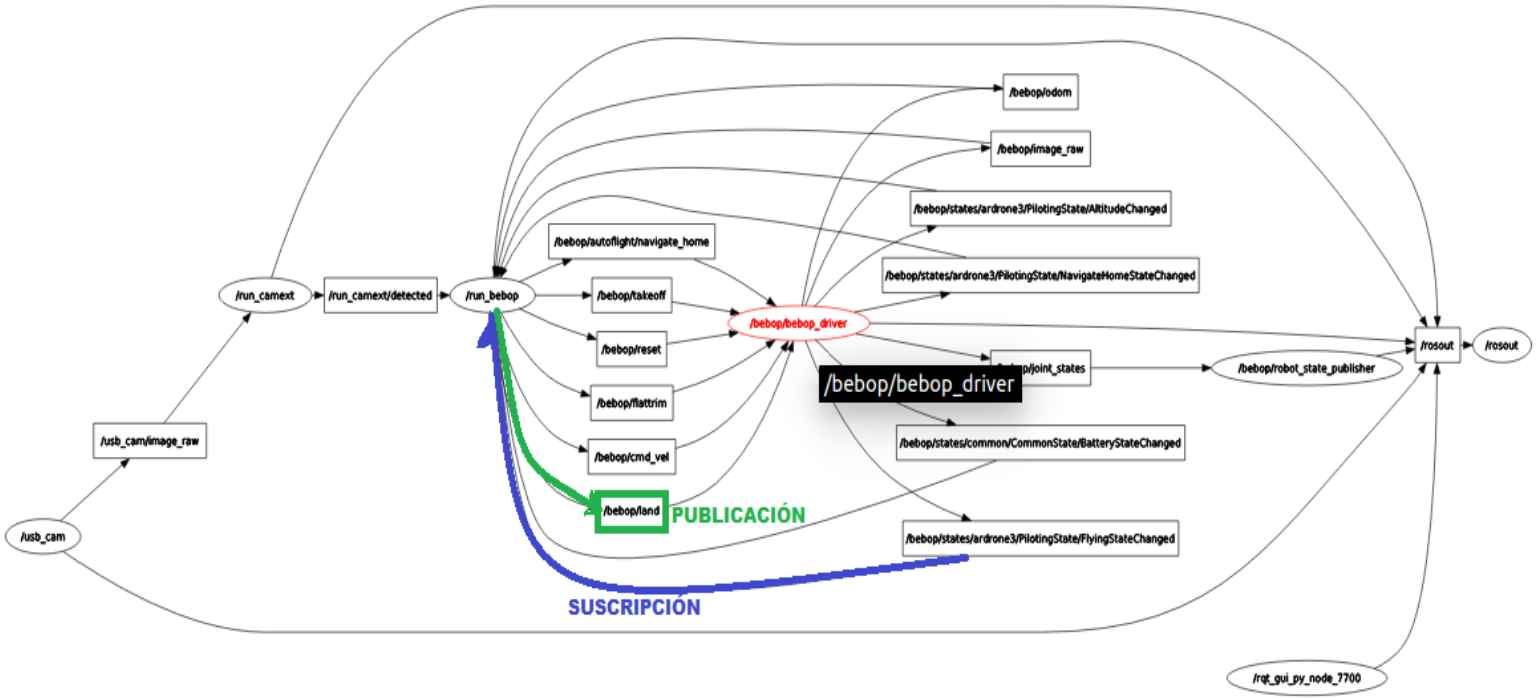
- Una vez ejecutada la aplicación se puede observar la comunicación entre nodos que son usados en esta aplicación.



EJECUCIÓN DE LA APLICACIÓN EN ROS

TÓPICOS DE SUSCRIPCIÓN Y PUBLICACIÓN

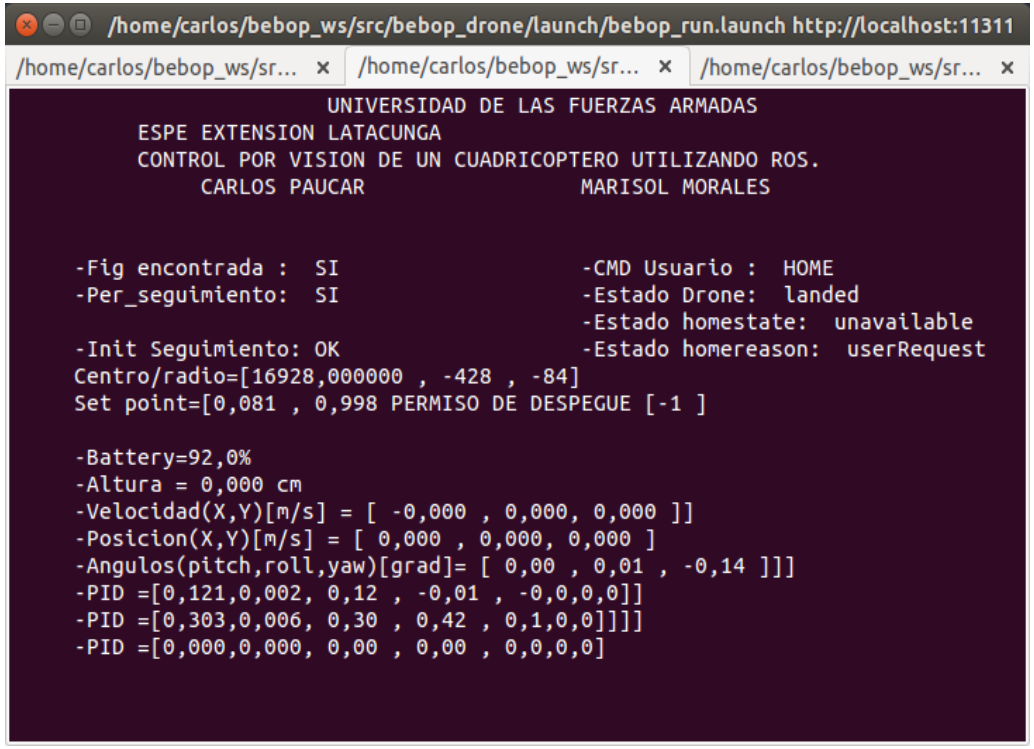
- Suscripciones sirven para recibir información que envía el nodo.
- Publicación son usados para enviar información desde el nodo



EJECUCIÓN DE LA APLICACIÓN EN ROS

INTERFAZ DE USUARIO

- Es la ventana principal donde se aprecian los datos que son enviados para el control de cuadricóptero, y se tiene una mejor idea de como se da el proceso



EJECUCIÓN DE LA APLICACIÓN EN ROS

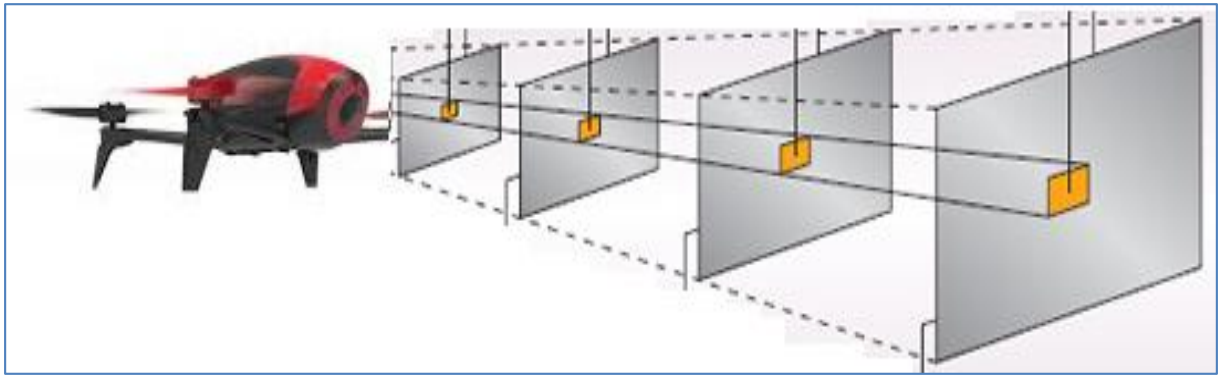
ORDENES DEL USUARIO POR TECLADO

- Se puede enviar órdenes de control al cuadricóptero por medio del teclado, mismas que pueden interferir o facilitar dicho control

Comandos desde teclado		
Tecla	Comando	Descripcion
T	Take off	Envía la orden de despegue al cuadricóptero
L	Landed	Envía la orden de aterrizaje al cuadricóptero
R	Reset	Resetea los parámetros del Bebop2
F	Flattrim	Calibra la cámara a la posición frontal, en función de la orientación del Bebop 2
P	Permiso de Seguimiento	Permite iniciar el seguimiento de modo manual enviando la orden por medio del teclado
H	Home	Envía la orden de regreso a casa

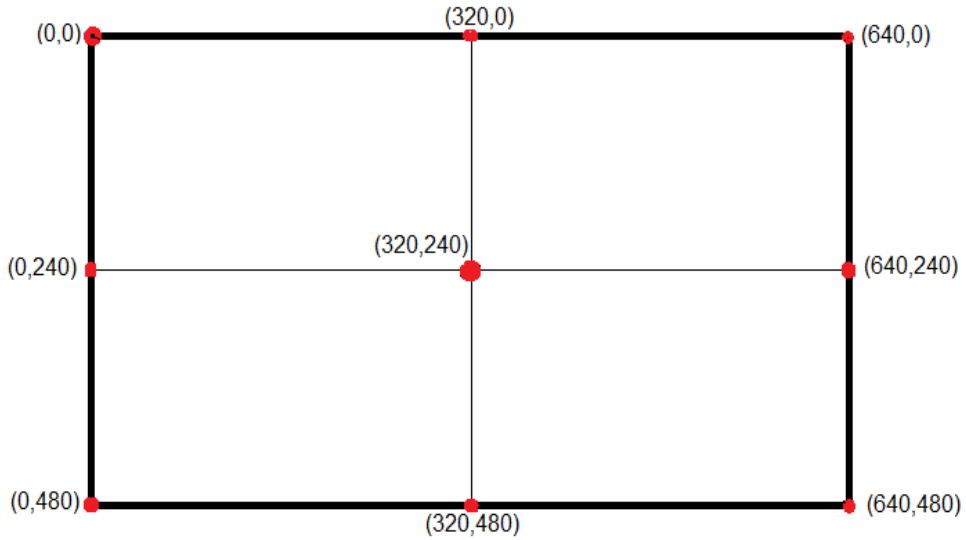
ESTIMACIÓN DE POSE

- Una vez en el seguimiento el vuelo debe ser en tiempo real o lo más aproximado posible, bajo estas circunstancias se usa la resolución de 640 x 480 pixeles.



ESTIMACIÓN DE POSE

- En el campo visual del cuadricóptero, se debe determinar la posición del objeto en movimiento, por lo tanto, la trayectoria de la persona en cualquier momento ocupará una posición (y, z)



ESTIMACIÓN DE POSE

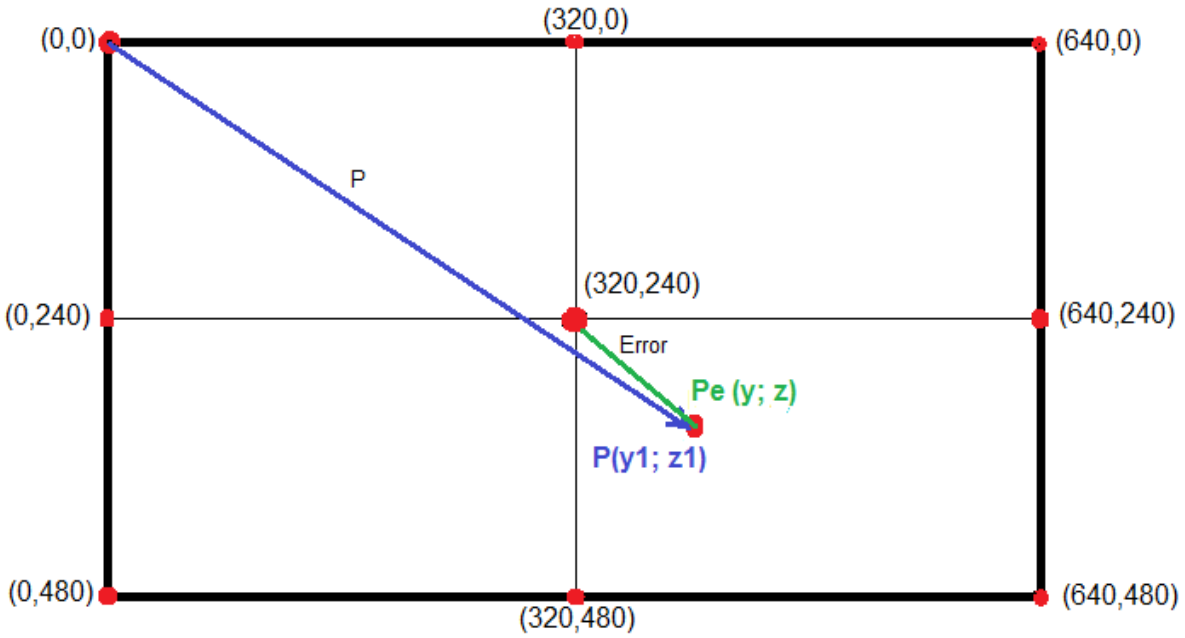
DETERMINACION DEL ERROR DE POSE

- El objetivo es mantener a la persona en el centro del campo visual, (640x480), por lo que se establece un nuevo punto de centro de la imagen recibida que es (320;240), que será el centro, durante el seguimiento.
- El centro del campo visual será establecido como el set point, para mantener la ubicación requerida y este se compara con el centro del objeto y así se determina el error que se debe corregir, esto se lo realiza por medio de una diferencia entre el punto centro del objeto y el centro establecido

$$Pe(y,z) = P(y1,z1) - P\,(320{,}240)$$

ESTIMACIÓN DE POSE

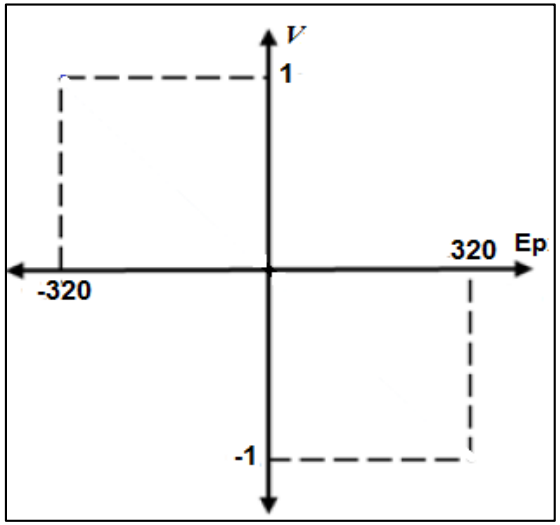
DETERMINACION DEL ERROR DE POSE



ESTIMACIÓN DE POSE

CONVERSIÓN DE ERROR DE POSICIÓN A VELOCIDAD

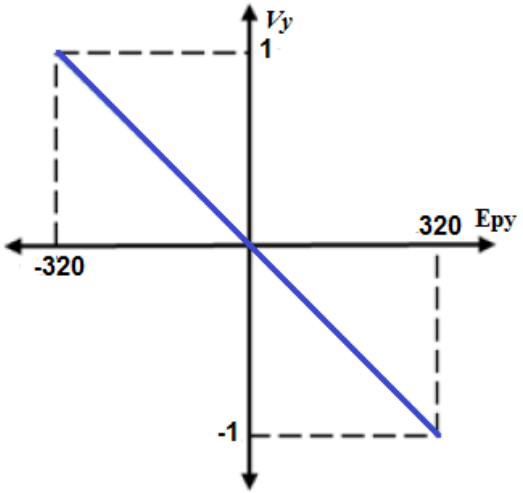
- Los valores máximos y mínimos que puede tomar el comando de velocidad lineal (± 1), en este caso se usa cmd_vel, mismo que adquirirá valores dentro del rango de ±1, siendo 1 el 100%.



ESTIMACIÓN DE POSE

POSICIÓN A VELOCIDAD EN Y (ROLL)

- Para este eje los valores oscilaran entre -320 a 320 siendo estos los valores máximos tanto para la izquierda como para la derecha

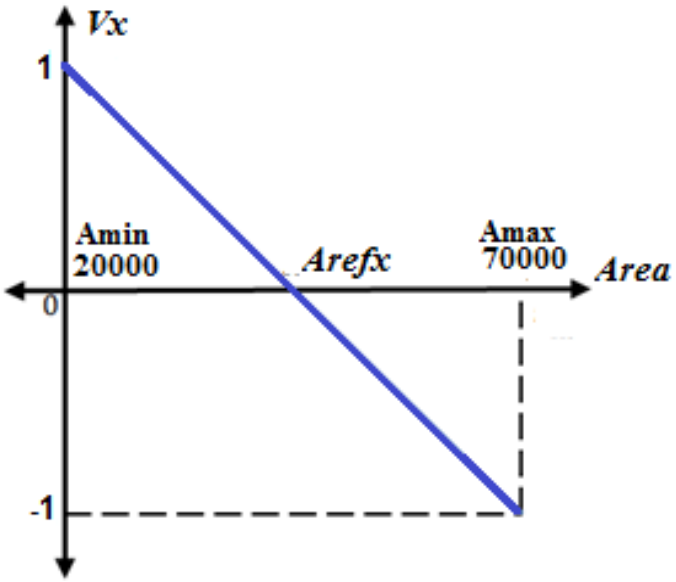


$$Vy = -\frac{Epy}{320}$$

ESTIMACIÓN DE POSE

POSICIÓN A VELOCIDAD EN X (CABECEO)

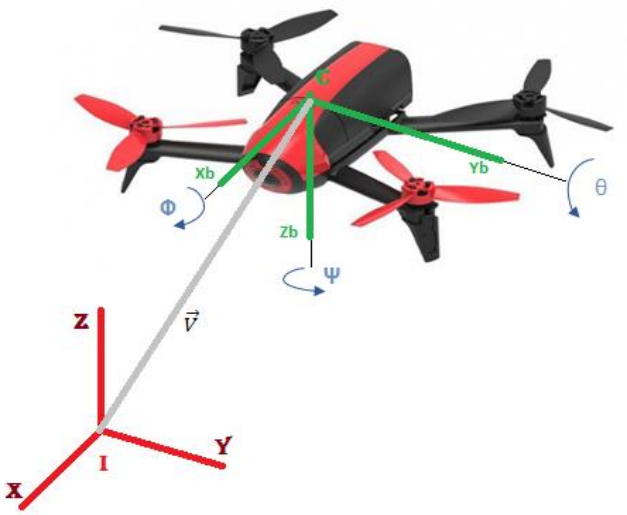
- En este eje se realiza un nuevo análisis, que consiste en trabajar con las áreas del objeto en seguimiento



$$Vx = (\left(1 - \frac{Area}{Arefx}\right) * K\right)$$

MODELADO MATEMÁTICO

- Para establecer la ubicación del cuadricóptero en cualquier punto del espacio, se debe conocer la posición de éste con respecto a un marco de referencia fijo y su orientación respecto a un marco inercial



MODELADO MATEMÁTICO

CÁLCULO DE INERCIA

- El cálculo de la matriz de inercia del cuadricóptero, se lo realizo considerando que este es simétrico.

$$\mathrm{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

MODELADO MATEMÁTICO

CÁLCULO DE INCERCIA

Inercia en el eje X

$$I_{x1} = I_{x3} = \frac{1}{12} m_m \left(I_y^2 + I_z^2\right)$$

$$I_{x2} = I_{x4} = \frac{1}{12} m_m (I_y^2 + I_z^2) + m_m \, d_{cg}^2$$

$$I_{xx} = 2I_{x1} + 2I_{x2}$$

Inercia en el eje Y

$$I_{y2} = I_{y4} = \frac{1}{12} m_m (I_x^2 + I_z^2)$$

$$I_{y1} = I_{y3} = \frac{1}{12} m_m (I_x^2 + I_z^2) + m_m \, d_{cg}^2$$

$$I_{yy} = 2I_{y1} + 2I_{y2}$$

MODELADO MATEMÁTICO

CÁLCULO DE INERCIA

Inercia en el eje Z

$$I_{z1} = I_{z2} = I_{z3} = I_{z4} = \frac{1}{12} m_m (I_x^2 + I_y^2) + m_m \, d_{cg}^2$$

$$I_{zz} = 4 I_{z1}$$

MODELADO MATEMÁTICO

CÁLCULO DE INERCIA

Variables	Valores
Masa del Dron	0.536 kg
Masa del Motor	0.04275 Kg
Longitud del rotor en X	0.023 m
Longitud del rotor en Y	0.023 m
Longitud del rotor en Z	0.0113 m
Distancia del motor al centro de gravedad del cuadricóptero	0.143 m

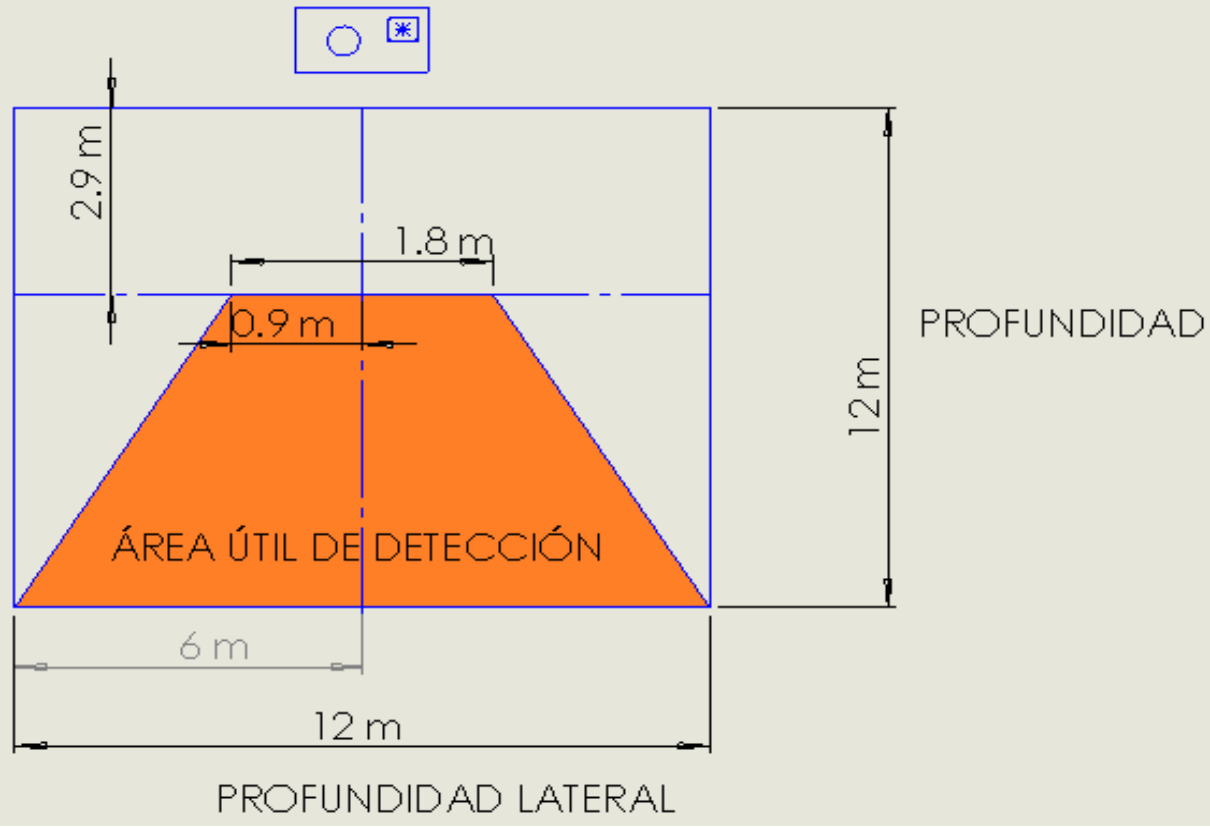
MODELADO MATEMÁTICO

POSICIÓN A VELOCIDAD EN Y (ROLL)

- Para este eje los valores oscilaran entre -320 a 320 siendo estos los valores máximos tanto para la izquierda como para la derecha

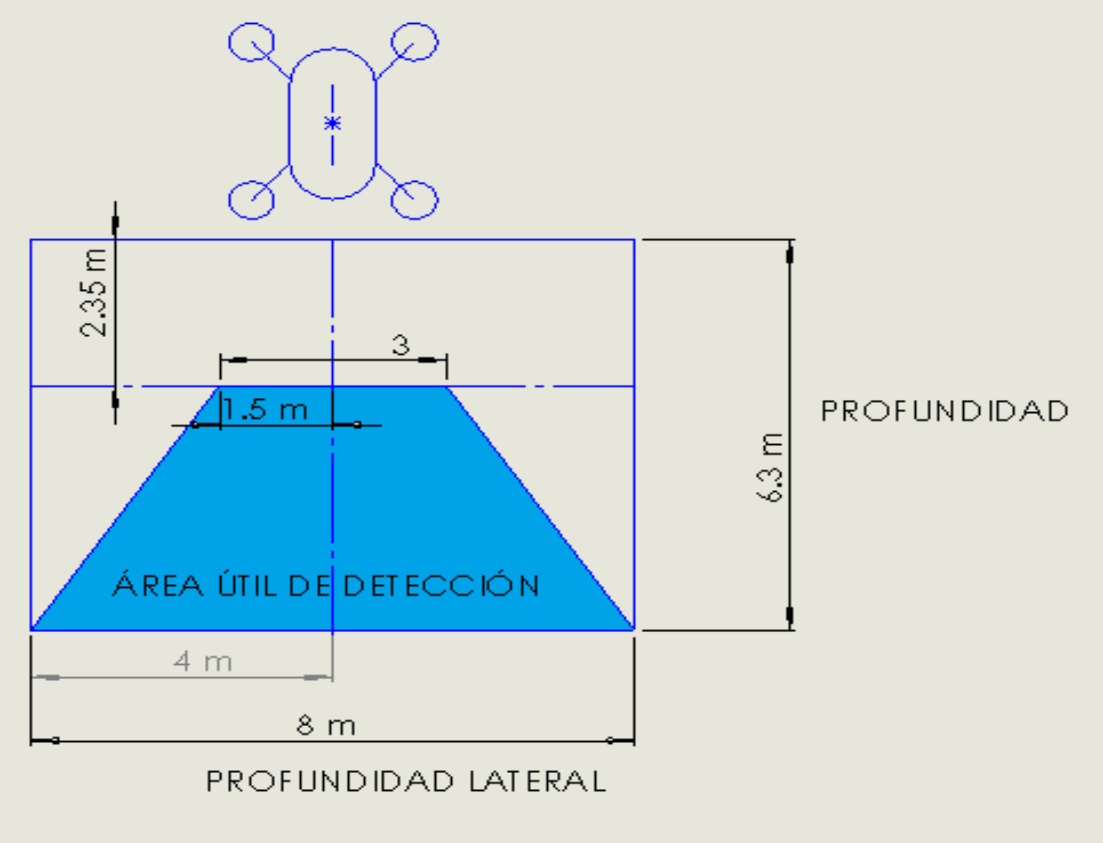
PRUEBAS Y RESULTADOS

CÁMARA ESTÁTICA



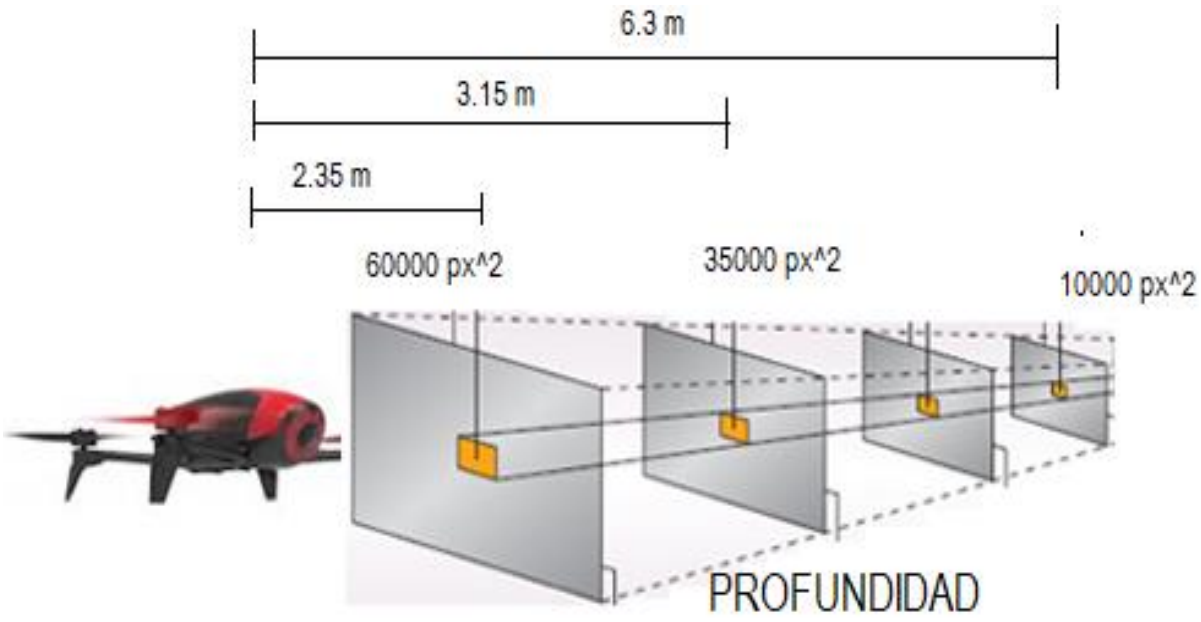
PRUEBAS Y RESULTADOS

CÁMARA DRONE



PRUEBAS Y RESULTADOS

CAMPO VISUAL



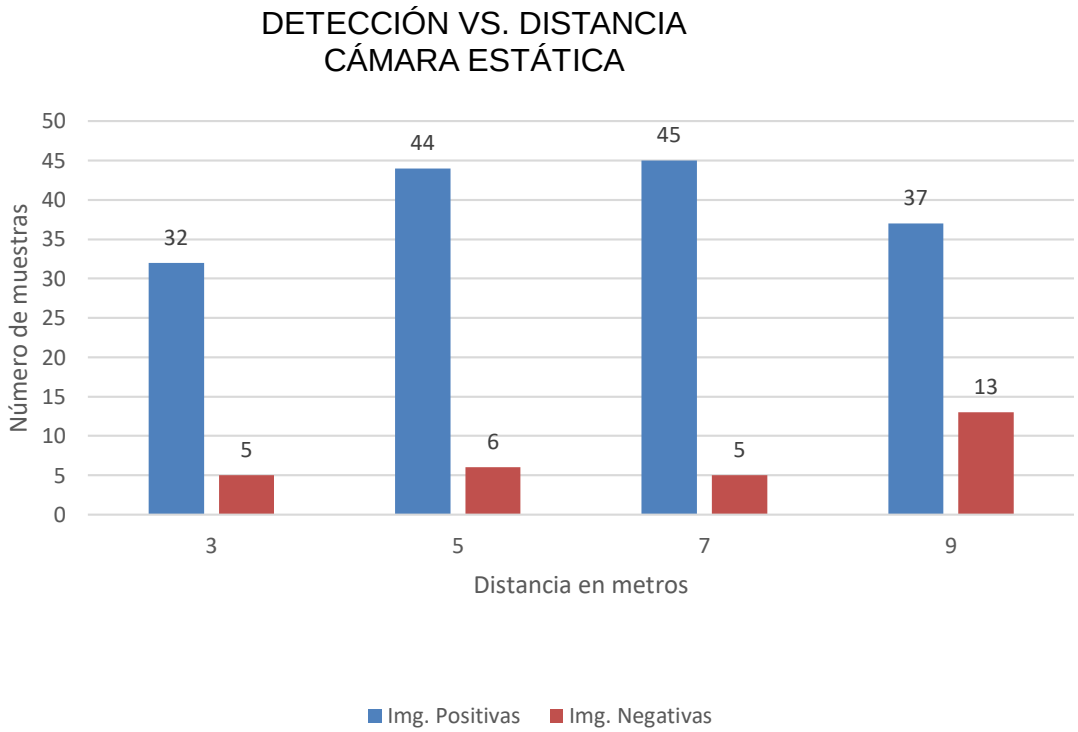
PRUEBAS Y RESULTADOS

COMANDOS DE VELOCIDAD

Área de detección en pixeles ^2	Valor driver cmd_x; cmd_y	Velocidad (m/s)	Acciones de control
10000	0,96	0,96	Adelante
20000	0,57	0,57	Adelante
30000	0,19	0,19	Adelante
35000	0	0	
40000	- 0,19	- 0,19	Atrás
50000	- 0,57	- 0,57	Atrás
60000	- 0,96	- 0,96	Atrás

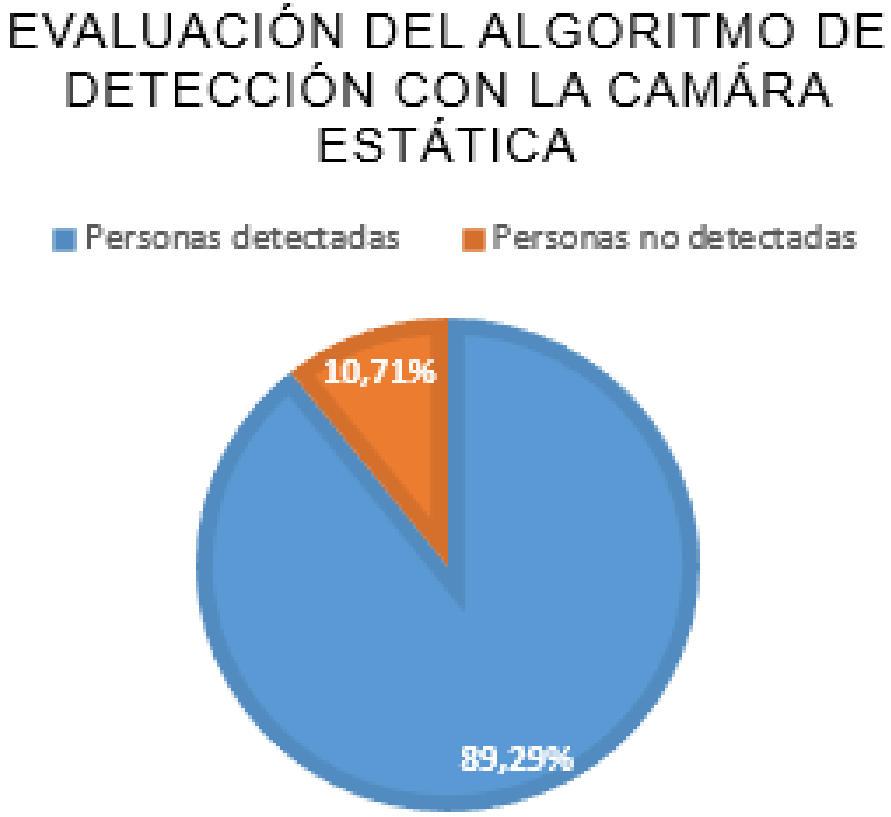
PRUEBAS Y RESULTADOS

EVALUACIÓN DEL ALGORITMO DE DETECCIÓN USANDO PARA LA CÁMARA ESTÁTICA



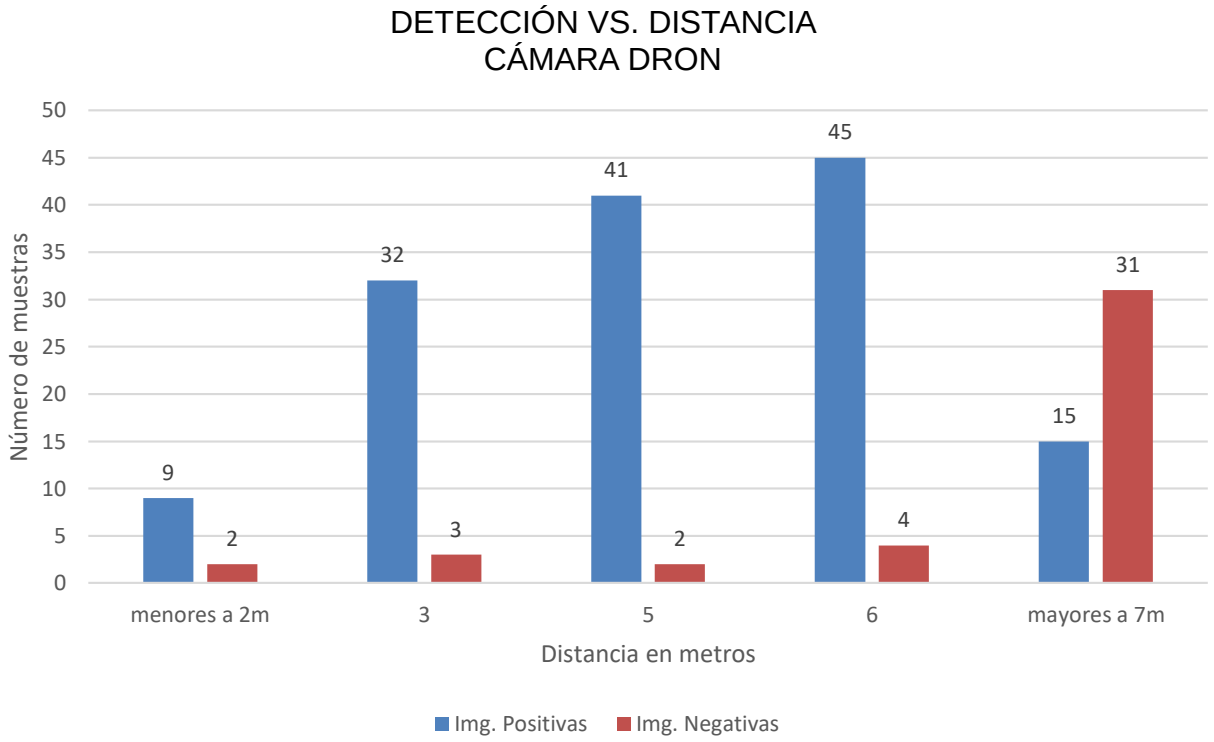
PRUEBAS Y RESULTADOS

EVALUACIÓN DEL ALGORITMO DE DETECCIÓN USANDO PARA LA CÁMARA ESTÁTICA



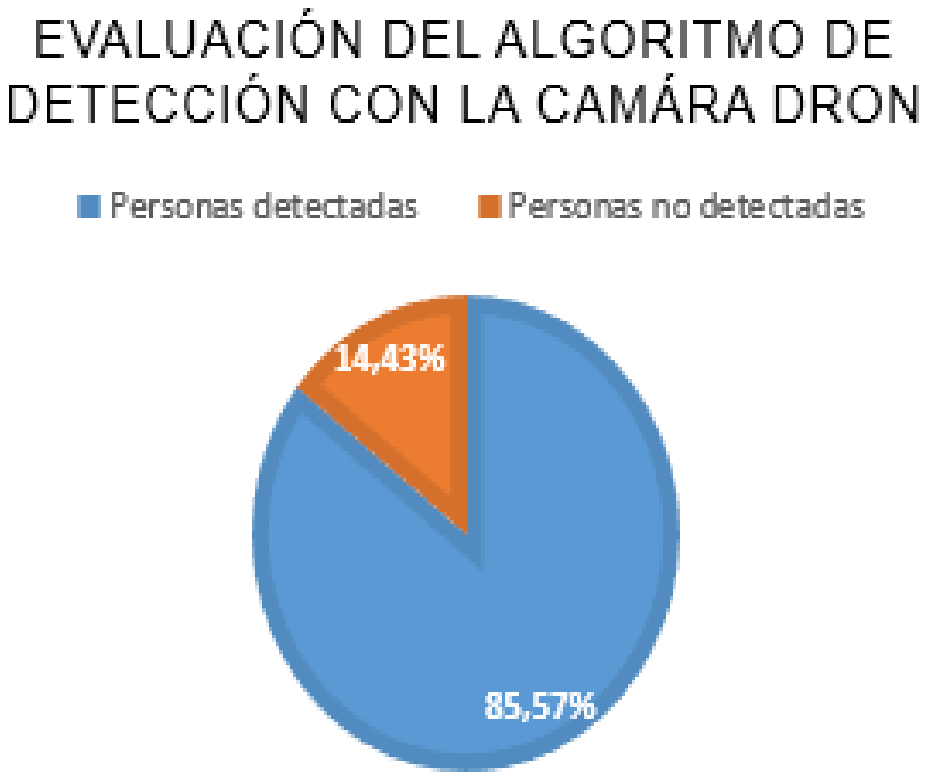
PRUEBAS Y RESULTADOS

EVALUACIÓN DEL ALGORITMO DE DETECCIÓN USANDO PARA LA CÁMARA DEL DRON



PRUEBAS Y RESULTADOS

EVALUACIÓN DEL ALGORITMO DE DETECCIÓN USANDO PARA LA CÁMARA DEL DRON



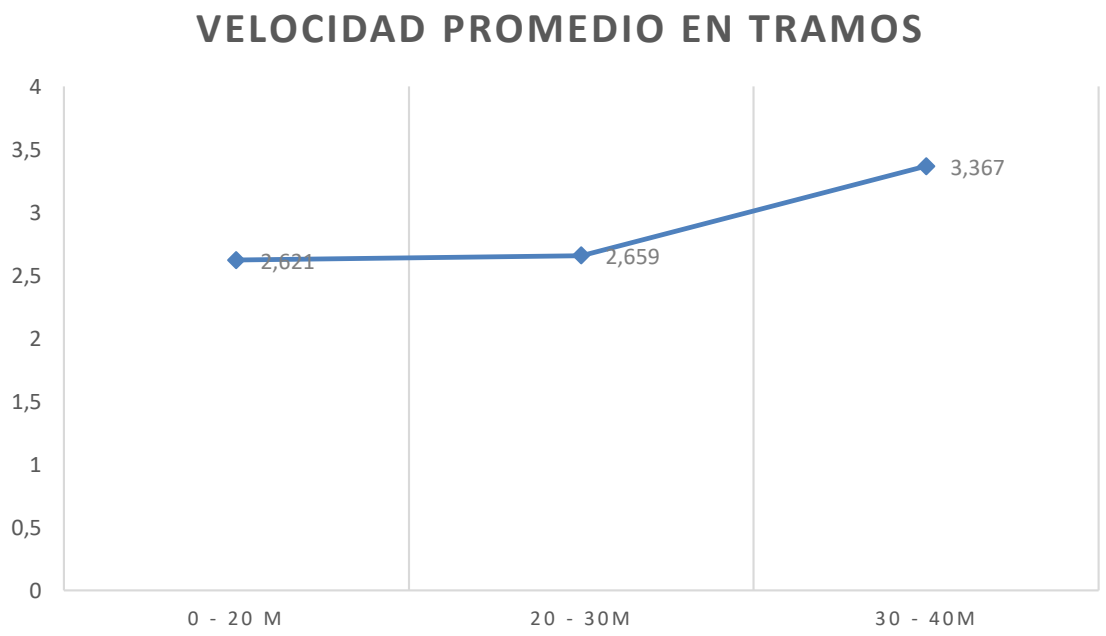
PRUEBAS Y RESULTADOS

PRUEBAS DE SEGUIMIENTO

	5m	10m	15m	20m	25m	30m	35m	40m	Total
Positivo	50	50	50	46	43	39	36	33	347
Negativo	0	0	0	4	7	11	14	17	53
Total	50	50	50	50	50	50	50	50	400

PRUEBAS Y RESULTADOS

PRUEBAS DE SEGUIMIENTO



PRUEBAS Y RESULTADOS

PRUEBAS DEL SISTEMA COMPLETO

CONCLUSIONES

- Trabajar con el cuadricóptero Parrot Bebop 2, fue una ventaja muy considerable debido a que se pude tener acceso a los datos de vuelo que son enviados por los diferentes sensores. Gracias a esto se puede desarrollar un sinnúmero de aplicaciones, y al tener su propia señal Wi-Fi hace posible que se puedan correr programas directamente desde la computadora, enviando comandos y receptando los diferentes estados de vuelo.
- En la configuración de la cámara del cuadricóptero Parrot Bebop 2, existe la posibilidad de cambiar la resolución nativa, haciendo que esto sea una ayuda al momento de procesar las imágenes.

CONCLUSIONES

- El uso de las diferentes herramientas que contiene el sistema operativo robótico (ROS), posibilita el desarrollo de aplicaciones para el control, procesamiento de imágenes, usadas al momento de realizar aplicaciones para el vuelo autónomo de cuadricópteros o plataformas robóticas.
- El sistema operativo robótico (ROS), entre sus características tiene la ventaja de ser código abierto, permitiendo tener acceso a diferentes repositorios e integrando códigos ya probados con anterioridad en sistemas robóticos complejos.

CONCLUSIONES

- Para tener un control adecuado del cuadricóptero, se debe obtener de manera adecuada la planta que representará a este, porque de acuerdo a esta se puede realizar un correcto cálculo de las ganancias para el control PID.
- Cuando se implementa el control PID, este debe estar calibrado de la forma adecuada para poder tener un seguimiento óptimo, ya que este es el que tiene el control total del cuadricóptero al momento de estar en modo autónomo.
- El uso de Matlab fue una gran ayuda al momento de implementar el control PID, ya que nos entrega la función de transferencia con sus respectivas ganancias Kp, Ki, Kd, de cada eje a controlar, pero al momento de realizar pruebas se vio la necesidad de calibrar estas usando el método oscilación de Ziegler-Nichols.

CONCLUSIONES

- Mediante el cambio de la imagen original a imagen en escala de grises, se mejoró el tiempo de procesamiento, dado que este último solo necesita un canal a diferencia de las imágenes RGB que necesitan 3.
- El algoritmo de reconocimiento de personas debe ser robusto para que pueda soportar los diferentes cambios de luz en ambientes no controlados, por tal motivo se usa el algoritmo de HOG, el cual brinda estas características.
- Mediante el uso del clasificador SVM, para separar las clases personas y no persona, desde un conjunto de entrenamiento predeterminado, se logró obtener una detección eficiente, al aplicarlo en los algoritmos de control la cámara estática y del dron.

CONCLUSIONES

- La correcta detección de personas, es fundamental, ya que gracias a esto se enviará comandos que rigen el control de vuelo del dron, tales como su despegue, regreso a casa y el principalmente el seguimiento en profundidad y lateralmente.
- La detección en vuelo, permite un adecuado seguimiento en lugares amplios y cerrados, así como en lugares abiertos, en los cuales no haya presencia de viento fuerte, dado que este influye, modificando la altura adecuada en vuelo del dron, por lo cual el cuadricóptero perdería al objetivo a seguir.

CONCLUSIONES

- La correcta detección de personas, es fundamental, ya que gracias a esto se enviará comandos que rigen el control de vuelo del dron, tales como su despegue, regreso a casa y el principalmente el seguimiento en profundidad y lateralmente.
- La detección en vuelo, permite un adecuado seguimiento en lugares amplios y cerrados, así como en lugares abiertos, en los cuales no haya presencia de viento fuerte, dado que este influye, modificando la altura adecuada en vuelo del dron, por lo cual el cuadricóptero perdería al objetivo a seguir.