UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE

DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO MECATRÓNICO

TEMA: "DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE VISIÓN ARTIFICIAL PARA PLANEACIÓN DE TRAYECTORIA DE UN VEHÍCULO AÉREO NO TRIPULADO EN UN ENTORNO CERRADO UTILIZANDO ROS"

AUTORES:

GUILLÉN ABAD, JOAN ANDRE

MANOSALVAS BANCHÓN, JOSÉ FELIPE

DIRECTOR:

Dr. AGUILAR CASTILLO, WILBERT GEOVANNY

SANGOLQUÍ

2018

CERTIFICADO TUTOR

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

CERTIFICACIÓN

Certifico que el trabajo de titulación **"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE VISIÓN ARTIFICIAL PARA PLANEACIÓN DE TRAYECTORIA DE UN VEHÍCULO AÉREO NO TRIPULADO EN UN ENTORNO CERRADO UTILIZANDO ROS"**, realizado por los señores **GUILLEN ABAD JOAN ANDRE** y **MANOSALVAS BANCHÓN JOSÉ FELIPE**, ha sido revisado en su totalidad y analizado por software anti-plagio, el mismo que cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de las Fuerzas Armadas ESPE, por lo tanto me permito acreditarlos y autorizar a los señores **MANOSALVAS BANCHÓN JOSÉ FELIPE** y **GUILLÉN ABAD JOAN ANDRE** para que lo sustenten públicamente.

Sangolquí, 30 de enero de 2018

Atentamente,

Dr. Wilbert Geovanny Aguilar Castillo

DIRECTOR

AUTORÍA DE RESPONSABILIDAD

DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

AUTORÍA DE RESPONSABILIDAD

Nosotros, **MANOSALVAS BANCHÓN JOSÉ FELIPE** con cédula de identidad No. 1722480090 y **GUILLÉN ABAD JOAN ANDRE** con cédula de identidad No. 1724235567, declaramos que este trabajo de titulación **"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE VISIÓN ARTIFICIAL PARA PLANEACIÓN DE TRAYECTORIA DE UN VEHÍCULO AÉREO NO TRIPULADO EN UN ENTORNO CERRADO UTILIZANDO ROS"** ha sido desarrollada considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaramos que este trabajo es de nuestra autoría, en virtud de ello nos declaramos responsables del contenido, veracidad y alcance del trabajo mencionado.

Sangolquí, 30 de enero de 2018

MANOSALVAS BANCHÓN JOSÉ FELIPE
CI. 1722480090

GUILLÉN ABAD JOAN ANDRE
CI. 1724235567

AUTORIZACIÓN

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

AUTORIZACIÓN

Nosotros, **MANOSALVAS BANCHÓN JOSÉ FELIPE** con cédula de identidad No. 1722480090 y **GUILLÉN ABAD JOAN ANDRE** con cédula de identidad No. 1724235567, autorizamos a la Universidad de las Fuerzas Armadas ESPE la publicación en la biblioteca virtual de la institución el presente trabajo con el tema: **"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE VISIÓN ARTIFICIAL PARA PLANEACIÓN DE TRAYECTORIA DE UN VEHÍCULO AÉREO NO TRIPULADO EN UN ENTORNO CERRADO UTILIZANDO ROS"**, cuyos contenidos son de nuestra responsabilidad y autoría.

Sangolquí, 30 de enero de 2018

MANOSALVAS BANCHÓN JOSÉ FELIPE
CI. 1722480090

GUILLÉN ABAD JOAN ANDRE
CI. 1724235567

DEDICATORIA

A mis padres Celmira y Rolando, quienes son mi fuerza y por quienes batallo.
A mi hermano Christian por compartir buenos y malos momentos conmigo.

A Elizabeth mi novia por su apoyo condicional cada momento.

A mis familiares y amigos que siempre estuvieron conmigo brindándome palabras de aliento.

Joan Guillén

Dedico este gran paso de mi vida a mi familia, amigos y personas que siempre ocuparan un lugar en mi corazón ya que son el motor que me impulsa a continuar en el camino y a mí mismo para seguir centrado y a continuar alcanzando mis metas

Felipe Manosalvas

AGRADECIMIENTO

A Dios por el valor, la fortaleza y perseverancia recibida, que me permitió culminar con éxito mi carrera universitaria.

A mis padres quienes son y serán el pilar fundamental en mi vida, que con su amor y apoyo han sabido guiarme en cada momento.

A mi novia por su apoyo y amor incondicional, por brindarme su cariño y compartir su vida conmigo.

A mis profesores a quienes les debo mis conocimientos y formación profesional

Al Doctor Wilbert Aguilar por su apoyo, amistad y asesoría dedicada en la realización de este proyecto

A Felipe por la amistad compartida durante mucho tiempo

Finalmente agradezco a esta prestigiosa universidad por permitirme formarme profesionalmente y como persona

Joan Guillen

Gracias a todos mis familiares que siempre estuvieron presentes en este largo proceso de la universidad que de alguna u otra manera se hicieron sentir con sus palabras de apoyo, actos de presencia y ayuda.

Gracias a mis maestros que me enseñaron tanto de lo que debían como de la vida.

Gracias a mis amigos que me acompañaron dándome ánimos para avanzar en un camino que cada vez se volvía más duro.

Gracias a todas esas personas que en momentos buenos y malos estuvieron a mi lado y también a los que me pusieron a prueba para darme cuenta que la victoria es más satisfactoria con un camino difícil

Felipe Manosalvas

ÍNDICE DE CONTENIDOS

CERTIFICADO TUTOR i

AUTORÍA DE RESPONSABILIDAD ii

AUTORIZACIÓN iii

DEDICATORIA iv

AGRADECIMIENTO v

ÍNDICE DE CONTENIDOS vii

ÍNDICE DE FIGURAS xiii

ÍNDICE DE TABLAS xix

RESUMEN xxi

ABSTRACT xxii

CAPÍTULO I

DESCRIPCIÓN GENERAL DEL PROYECTO 1

1.1. Introducción 1

1.2. Antecedentes 2

1.3. Justificación e importancia 4

1.4. Área de influencia 5

1.5. Alcance del proyecto 5

1.6. Objetivos 6

1.6.1. Objetivo general 6

1.6.2. Objetivos específicos 6

1.7. Estructura del documento 7

CAPÍTULO II

ESTADO DEL ARTE 9

2.1. Vehículos aéreos no tripulados UAVs 9

2.1.1 Definición 9

2.1.2 Clasificación....10
2.2. Quadrotor o cuatrirotor....13
2.2.1 Componentes....15
2.2.1.1 Controlador de vuelo....15
2.2.1.2 Motores....16
2.2.1.3 Electronic Speed Controller ESC....17
2.2.1.4 Hélices....18
2.2.1.5 Batería LiPo....19
2.2.1.6 Frame o armazón....20
2.2.2. Funcionamiento....20
2.2.2.1 Movimiento Pitch....21
2.2.2.2 Movimiento Roll....21
2.2.2.3 Movimiento Yaw....22
2.2.2.4 Movimiento Gaz o Throttle....22
2.3. Marcadores....23
2.3.1 Marcadores Pasivos....23
2.3.2 Marcadores Activos....23
2.4. Quadrotor: Parrot Mambo....25
2.5. Fundamentos de sistemas de visión por computador....26
2.6. OpenCV....32
2.6.1 Concepto....32
2.6.2 Introducción a OpenCV....33
2.6.3 Funcionalidades del núcleo....33
2.6.4 Procesamiento de imágenes....34
2.6.5 Calibración de cámaras....35
2.6.6 Entrada y salida de video....36

2.6.7 Característica 2D 37
2.6.8 Análisis de video 38
2.6.9 Detección de objetos....39
2.6.10 Unión de imágenes 39
2.7. Filtros de discriminación para procesamiento de imágenes 40
2.8. Algoritmos de control para planeación de trayectoria de vuelo 45
2.9. Software 49
2.6.1 Sistema Operativo: Ubuntu 16.04 LTS 49
2.6.2 Robot Operating System ROS 50
2.6.2.1 ¿Qué es ROS? 50
2.6.2.2 Objetivos de ROS 51
2.6.2.3 Áreas de aplicación de ROS 51
2.6.2.4 Funcionamiento del sistema 52
2.6.2.4.1 Sistemas de archivos....52
2.6.2.4.2 Computación a nivel gráfico....52
2.6.2.5 Virtualización de ROS: RVIZ....53
2.6.2.6 Estructura de ROS 54
2.6.2.6.1 Repositorio 54
2.6.2.6.2 Pila (Stack) 55
2.6.2.6.3 Paquete 55
2.6.2.6.4 Nodo....55
2.6.3 Lenguaje de programación: Python....55
2.10. Controlador PID 57

CAPÍTULO III

DISEÑO DE BASTIDOR PARA ARREGLO DE CÁMARAS 62

3.1 Diseño CAD....62

3.1.1 Bastidor....62
3.1.2 Cámaras....62
3.2 Selección de materiales....63
3.2.1 Bastidor....63
3.2.2 Cámaras....64
CAPÍTULO IV
DISEÑO Y SIMULACIÓN DE VEHÍCULO AÉREO NO TRIPULADO....66
4.1 Diseño CAD....66
4.1.1 Cuatrirotor de ala rotativa....66
4.1.2 Base para interfaz de comunicación PC – aeronave....66
4.2 Simulación de vehículo aéreo no tripulado....67
4.2.1 Simulación cuatrirotor de ala rotativa....67
4.3 Selección de materiales del vehículo aéreo no tripulado....70
4.3.1 Frame o armazón....71
4.3.2 Motores....71
4.3.3 Hélices....73
4.3.4 Microcontrolador....73
4.3.5 Módulo de comunicación por radio frecuencia....77
4.3.6 Unidad de medición inercial (IMU)....78
4.3.7 Electronic Speed Control (ESC)....80
4.3.8 Batería Li-PO....82
4.4 Diseño circuito de alimentación para marcadores activos....84
4.5 Diseño circuito de comunicación PC – aeronave....85
CAPÍTULO V
SISTEMA DE CONTROL....89
5.1 Diseño del controladores PID....89

5.2 Diseño de algoritmos de control para generación de trayectoria.....94
5.2.1 Algoritmo RRT 94
5.2.2 Optimización de trayectoria.....96
CAPÍTULO VI
CONSTRUCCIÓN E IMPLEMENTACIÓN 99
6.1 Construcción e implementación de los componentes mecánicos 99
6.1.1 Construcción de vehículo aéreo no tripulado 99
6.1.2 Construcción de bastidor para arreglo de cámaras 100
6.2 Implementación de los componentes eléctricos y/o electrónicos 102
6.2.1 Circuito de interface PC – aeronave.....102
6.2.2 Circuito de alimentación para marcadores activos 103
6.3 Desarrollo e implementación de software 104
6.3.1 Implementación de filtros físicos y digitales para procesamiento de imágenes 104
6.3.2 Implementación de sistema de visión por computador.....106
6.3.3 Implementación en software del HMI 113
6.3.4 Implementación de algoritmo de planificación de trayectoria RRT 118
6.3.3.1 Algoritmo RRT.....118
6.3.3.2 Optimización de trayectoria algoritmo RRT.....119
6.4 Implementación de la plataforma de soporte ROS 120
6.5 Implementación del sistema de control.....121
6.4.1 Implementación de controladores PID 122
CAPÍTULO VII
PRUEBAS Y RESULTADOS.....124
7.1 Pruebas y resultados posicionamiento (X e Y) y altitud (Z).....124
7.2 Pruebas y resultados costo computacional en cálculo de trayectoria 128

7.2.1 Pruebas trayectoria RRT estándar 128
7.2.2 Pruebas trayectoria RRT optimizada 128
7.2.3 Resultados trayectoria RRT estándar y trayectoria RRT optimizada 128
7.3 Pruebas y resultados seguimiento de trayectoria 129
7.4 Interpretación de resultados 137
CAPÍTULO VIII
CONCLUSIONES, RECOMENDACIONES Y TRABAJOS FUTUROS 141
8.1 Conclusiones 141
8.2 Recomendaciones 143
8.3 Trabajos Futuros 143
8.4 Bibliografía 144
8.5 Referencias 148

ÍNDICE DE FIGURAS

Figura 1 Vehículo aéreo no tripulado UAV.....9
Figura 2 Aplicación UAV sector agrícola.....10
Figura 3 Algunos tipos de aeronaves utilizadas en los UAVs 11
Figura 4 Quadrotor 12
Figura 5 Helicóptero autónomo 12
Figura 6 Aeroplano autónomo 12
Figura 7 Dirigible autónomo.....12
Figura 8 Momentos de pitch, yaw y roll 14
Figura 9 Esquema de fuerzas de un cuatrirotor y momentos de cada motor 14
Figura 10 Controlador de vuelo CC3D.....16
Figura 11 (a) Motor Brushed; (b) Motor Brushless.....17
Figura 12 Controlador de velocidad de motor, Electronic Speed Controller ESC.....18
Figura 13 Hélices para cuatrirotor.....19
Figura 14 Baterías tipo LiPo 19
Figura 15 Frame o estructura de un cuatrirotor.....19
Figura 16 (a) Rotaciones de hélices de un cuatrirotor; (b) Rotación del cuatrirotor en relación a sus ejes 20
Figura 17 Movimiento Pitch 21
Figura 18 Movimiento Roll.....21
Figura 19 Movimiento Yaw 22
Figura 20 (a) Aumento de velocidad y altura; (b) Reducción de velocidad y altura.....22
Figura 21 Marcador pasivo colocado en un vehículo aéreo no tripulado 23
Figura 22 Marcador activo colocado en un vehículo aéreo no tripulado 24
Figura 23 Cuatrirotor Parrot Mambo.....25
Figura 24 Cámara térmica.....26
Figura 25 Obtención de posición 27
Figura 26 Determinación de caracteres.....27
Figura 27 Pixeles en una imagen 28
Figura 28 Matriz de imagen en escala de grises.....28
Figura 29 Imágenes binarias 29

Figura 30 Ejemplo de imágenes en baja y alta resolución 29
Figura 31 Efecto ojo de pez 31
Figura 32 Proceso de calibración de la cámara 31
Figura 33 Proceso de visión por computador 32
Figura 34 Imagen de espacio de memoria OpenCV 33
Figura 35 Imagen cambiada a escala de grises OpenCV 33
Figura 36 Dibujos básicos OpenCV 34
Figura 37 Inserción de texto OpenCV 34
Figura 38 Erosionado de imágenes OpenCV 35
Figura 39 Cámara no calibrada OpenCV 35
Figura 40 Cámara calibrada OpenCV 36
Figura 41 Captura desde un archivo OpenCV 36
Figura 42 Video guardado OpenCV 37
Figura 43 Detección de esquinas OpenCV 37
Figura 44 Esquinas obtenidas de imagen OpenCV 38
Figura 45 Comparación de características desde 2 imágenes OpenCV 38
Figura 46 Sustracción de fondo OpenCV 38
Figura 47 Detección de rostros con método de cascada OpenCV 39
Figura 48 Unión de imágenes OpenCV 40
Figura 49 Ruido captado por la cámara OpenCV 41
Figura 50 Cambio de espacio de color OpenCV 42
Figura 51 Detección de contornos OpenCV 42
Figura 52 Llenado de contornos OpenCV 43
Figura 53 Uso de filtros threshold 43
Figura 54 Detección de color 44
Figura 55 Filtros físicos para cámaras 45
Figura 56 Crecimiento del árbol RRT 46
Figura 57 Crecimiento uniforme RRT 46
Figura 58 Crecimiento evadiendo obstáculos 46
Figura 59 Post procesado de RRT 48
Figura 60 Entorno de trabajo Ubuntu 16.04 LTS 49
Figura 61 Logo Ubuntu 16.04 LTS 50

Figura 62 Nivel de computación gráfica de ROS 53
Figura 63 Entorno de trabajo de RVIZ 54
Figura 64 Jerarquización de ROS 55
Figura 65 Logo Python 3.0 56
Figura 66 Controlador PID de una planta 57
Figura 67 Controlador proporcional P. 58
Figura 68 Controlador Proporcional Derivativo PD. 59
Figura 69 Controlador Proporcional Integral PI. 60
Figura 70 Controlador Proporcional Integral Derivativo PID. 61
Figura 71 Diseño en Software CAD Bastidor 62
Figura 72 Diseño en Software CAD cámara PlayStation 3 Eye 62
Figura 73 Diseño Software CAD soporte filtro físico para cámara PlayStation 3 Eye 63
Figura 74 Cámara PlayStation 3 Eye 65
Figura 75 Hub USB 2.0 65
Figura 76 Cuatrirotor de ala rotativa 66
Figura 77 Base para interfaz de comunicación PC – aeronave 67
Figura 78 Medición experimental de peso generado al arrancar un motor a su máxima potencia 67
Figura 79 Distribución de cargas sobre el elemento para análisis estático 68
Figura 80 Análisis de esfuerzo de Von Mises cuatrirotor 69
Figura 81 Análisis de factor de seguridad cuatrirotor 70
Figura 82 Piezas de corte con laser 71
Figura 83 Motor Brushless Turnigy 1806-2280KV 72
Figura 84 Hélices seleccionadas 73
Figura 85 OpenPilot CC3D Flight Controlller 75
Figura 86 Arduino Nano 75
Figura 87 NanoPi Neo Plus 2 76
Figura 88 Módulo de comunicación FRSKY 2.4 GHz 78
Figura 89 OpenPilot CC3D Flight Controller, localización IMU Invensense MPU-6000 Gyro/Accel 79
Figura 90 MPU-6000 Gyro/Accel Invensense 80

Figura 91 ESC BLHeli-12A 81
Figura 92 Batería seleccionada Multi Star 1400 mAh 83
Figura 93 1W High Power LED 84
Figura 94 D – SUN 3A Step Down Switching Voltage Regulator 85
Figura 95 Circuito de alimentación para marcadores activos 85
Figura 96 Arduino Micro 86
Figura 97 5 mm Round LED 86
Figura 98 Botón paro de emergencia de medio giro G & V Electric 87
Figura 99 Circuito de comunicación PC - aeronave 88
Figura 100 Datos obtenidos modelo experimental pitch (a) y roll (b) de la aeronave 89
Figura 101 Planta pitch en lazo cerrado 90
Figura 102 Comportamiento planta pitch en lazo cerrado ante entrada escalón 90
Figura 103 Planta roll en lazo cerrado 90
Figura 104 Comportamiento planta roll en lazo cerrado ante entrada escalón 91
Figura 105 Plantas pitch y roll en lazo cerrado con control PID. 91
Figura 106 Comportamiento de la planta ante entrada escalón unitario 92
Figura 107 Cálculo de la constante de tiempo T1, método de Ziegler-Nichols 92
Figura 108 Cálculo de la constante de tiempo T2, método de Ziegler-Nichols 93
Figura 109 Algoritmo RRT unidireccional 95
Figura 110 Funciones del algoritmo RRT 96
Figura 111 Algoritmo RRT generación de trayectoria 96
Figura 112 Algoritmo optimización trayectoria RRT 97
Figura 113 Ejemplo de aplicación de algoritmo de optimización de trayectoria RRT 98
Figura 114 Armazón cuatrirotor 99
Figura 115 Cuatrirotor con elementos eléctricos y electrónicos 100
Figura 116 Prisma rectangular con perfiles cuadrado de aluminio 100
Figura 117 Implementación de red de seguridad a la estructura 101
Figura 118 Implementación de estructuras para soporte de arreglo de cámaras 101
Figura 119 Bastidor completo implementado 102
Figura 120 Circuito de comunicación PC – aeronave 102

Figura 121 Circuito de alimentación para marcadores activos 103
Figura 122 Funcionamiento circuito de alimentación para marcadores activos 103
Figura 123 Filtro físico en cámara 104
Figura 124 Aplicación de escala de grises sobre imagen de cuatrirotor 105
Figura 125 Aplicación de filtro threshold sobre imagen aplicada escala de grises 105
Figura 126 Captura de imágenes por cámara 106
Figura 127 Primera interpolación y ajuste de curvas 109
Figura 128 Segunda interpolación y ajuste de curvas 110
Figura 129 Curva y ecuación general para la cámara 0 111
Figura 130 Posición y altitud calculados y presentados en la HMI 113
Figura 131 Arquitectura HMI 114
Figura 132 Navegación sección Monitoreo 115
Figura 133 Navegación sección Control Manual 115
Figura 134 Navegación sección Control Automático 116
Figura 135 HMI sección portada, monitoreo 1 y control manual 116
Figura 136 HMI sección control automático 117
Figura 137 HMI sección monitoreo 2 118
Figura 138 Ejemplo de generación de trayectoria por algoritmo RRT 119
Figura 139 Puntos remitidos al sistema de parte del algoritmo RRT 119
Figura 140 Optimización de trayectoria 120
Figura 141 Plataforma de comunicación ROS implementada 121
Figura 142 Planta en lazo cerrado implementado un controlador PID con primeras constantes 122
Figura 143 Comportamiento de la planta ante entrada escalón primeras constantes 122
Figura 144 Planta en lazo cerrado implementado un controlador PID con constantes definitivas 123
Figura 145 Comportamiento de la planta ante entrada escalón constantes definitivas 123
Figura 146 Prueba 1 posicionamiento real, coordenadas X y Y 124
Figura 147 Prueba 1 altitud real, coordenada Z 125

Figura 148 Prueba 1 posicionamiento y altitud calculadas, coordenadas X, Y y Z 125
Figura 149 Prueba 2 posicionamiento real, coordenadas X y Y 126
Figura 150 Prueba 2 altitud real, coordenada Z 126
Figura 151 Prueba 2 posicionamiento y altitud calculadas, coordenadas X, Y y Z. 127
Figura 152 Prueba 1 seguimiento de trayectoria HMI 130
Figura 153 Prueba 1 ajuste de curvas 130
Figura 154 Prueba 2 seguimiento de trayectoria HMI 130
Figura 155 Prueba 2 ajuste de curvas 131
Figura 156 Prueba 3 seguimiento de trayectoria HMI 131
Figura 157 Prueba 3 ajuste de curvas 131
Figura 158 Prueba 4 seguimiento de trayectoria HMI 132
Figura 159 Prueba 4 ajuste de curvas 132
Figura 160 Prueba 5 seguimiento de trayectoria HMI 132
Figura 161 Prueba 5 ajuste de curvas 133
Figura 162 Prueba 6 seguimiento de trayectoria HMI 133
Figura 163 Prueba 6 ajuste de curvas 133
Figura 164 Valores reales vs valores calculados posición en X 137
Figura 165 Valores reales vs valores calculados posición en Y 138
Figura 166 Valores reales vs valores calculados posición en Z 138

ÍNDICE DE TABLAS

Tabla 1 *Características de los principales tipos de aeronaves para UAVs* 11
Tabla 2 *Clasificación de principales UAVs de acuerdo al nivel de autonomía* 12
Tabla 3 *Comparativo entre marcadores activos y pasivos disponibles* 24
Tabla 4 *Materiales analizados* 63
Tabla 5 *Cámaras analizadas* 64
Tabla 6 *Detalle de cargas aplicadas al elemento para análisis estático* 68
Tabla 7 *Resultados de estudio de Esfuerzo de Von Mises sobre cuatrirotor* 69
Tabla 8 *Detalle de peso en la aeronave* 72
Tabla 9 *Especificaciones técnicas motor Brushless Turnigy 1806-2280KV* 72
Tabla 10 *Especificaciones hélices seleccionadas* 73
Tabla 11 *Comparativa entre OpenPilot CC3D Flight Controller, Arduino Nano y FriendlyARM NanoPi Neo Plus 2* 76
Tabla 12 *Especificaciones técnicas módulo de comunicación FRSKY* 78
Tabla 13 *Especificaciones MPU-6000 Gyro/Accel Invensense* 80
Tabla 14 *Comparativa ESCs BLHeli* 81
Tabla 15 *Comparativa batería seleccionada vs. Cálculos realizados* 83
Tabla 16 *Características técnicas 1W High Power LED* 84
Tabla 17 *Características técnicas D – SUN 3 A Step Down Switching Voltage Regulator* 85
Tabla 18 *Características Arduino Micro* 86
Tabla 19 *Características técnicas 5 mm Round LED* 87
Tabla 20 *Ecuaciones de cálculo de parámetros del controlador PID en base al Método de Ziegler-Nichols* 93
Tabla 21 *Datos adquiridos desde cámara 0* 107
Tabla 22 *Coeficientes obtenidos de la primera interpolación* 110
Tabla 23 *Ejemplo de cálculo final de posición X e Y* 112
Tabla 24 *Resultados pruebas de posicionamiento coordenadas X, Y y Z* 127
Tabla 25 *Tiempos de ejecución y número de nodos algoritmo RRT estándar* 128
Tabla 26 *Tiempos de ejecución y número de nodos algoritmo RRT optimizado* 128

Tabla 27 *Tiempo computacional de ejecución de algoritmos* 128
Tabla 28 *Comparativa de eficiencia entre algoritmos implementados* 129
Tabla 29 *Prueba 1 seguimiento de trayectoria, error calculado* 134
Tabla 30: *Prueba 2 seguimiento de trayectoria, error calculado* 134
Tabla 31 *Prueba 3 seguimiento de trayectoria, error calculado* 135
Tabla 32 *Prueba 4 seguimiento de trayectoria, error calculado* 135
Tabla 33 *Prueba 5 seguimiento de trayectoria, error calculado* 136
Tabla 34 *Prueba 6 seguimiento de trayectoria, error calculado* 136
Tabla 35 *Errores promedio punto a punto para seguimiento de trayectoria* 137
Tabla 36 *Cálculo de error en posicionamiento valor real vs valor calculado* 137
Tabla 37 *Resumen de error promedio entre valores reales vs valores calculados* . 138
Tabla 38 *Porcentaje de coste computacional algoritmo RRT estándar vs algoritmo RRT optimizado* 139
Tabla 39 *Resumen de error de seguimiento de trayectoria* 140
Tabla 40 *Cálculo de error promedio general punto a punto para seguimiento de trayectoria* 140

RESUMEN

El presente proyecto muestra el diseño, construcción e implementación de un sistema de visión por computador para la generación de trayectoria de vuelo de un vehículo aéreo no tripulado específicamente para entornos cerrados utilizando la plataforma ROS, cuyo objetivo es la estimación de posición de la aeronave utilizando marcadores visuales siendo aplicados criterios de visión por computador, además de la generación de trayectoria en base a algoritmos para la evasión de obstáculos y posteriormente la realización de seguimiento de trayectoria por parte de la aeronave. En el capítulo uno se detalla la introducción, antecedentes y objetivos del proyecto. El capítulo dos muestra conceptos de visión por computador, robótica aérea particularmente de los cuatrirotores, algoritmos para planeación de rutas de vuelo, definición de la plataforma ROS y controlador PID. En el capítulo siguiente se muestra el diseño y selección de materiales para la construcción del bastidor donde será montado el arreglo de cámaras para el sistema de visión por computador. En el capítulo cuatro se detalla el diseño, simulación y selección de materiales para la posterior construcción del vehículo aéreo no tripulado, incluyendo el diseño del circuito para la implementación de marcadores visuales y el circuito para comunicación inalámbrica a 2.4 GHz de la aeronave con el software de control. A continuación en el capítulo cinco se muestra el diseño del controlador PID para la aeronave y el diseño de algoritmos para la generación de trayectoria. En el capítulo seis se detalla la construcción de componentes mecánicos del proyecto, implementación de componentes eléctricos, electrónicos y software en general incluyendo la realización de una HMI de supervisión y monitoreo. En el capítulo siguiente se muestran las pruebas realizadas en cuanto a posicionamiento, seguimiento de trayectoria y coste computacional. Finalmente en el capítulo ocho se presentan conclusiones, recomendaciones para mejorar el funcionamiento del sistema y propuestas de trabajos futuros en base al trabajo de titulación actual.

Palabras Clave:

- **CUATRIROTOR**
- **VISIÓN POR COMPUTADOR**
- **ROS**
- **ESTIMACIÓN DE POSICIÓN**
- **GENERACIÓN DE TRAYECTORIA**

ABSTRACT

The present project shows the design, construction and implementation of a computer vision system for the generation of the flight path of an unmanned aerial vehicle specifically for closed environments using the ROS platform, whose objective is to estimate the position of the aircraft using visual markers being applied criteria of vision by computer, in addition to the generation of trajectory on the basis of algorithms for the evasion of obstacles and later the accomplishment of pursuit of trajectory on the part of the aircraft. Chapter one details the introduction, background and objectives of the project. Chapter two shows concepts of computer vision, aerial robotics, particularly of quadrotor, algorithms for planning flight routes, definition of the ROS platform and PID controller. The following chapter shows the design and selection of materials for the construction of the frame where the array of cameras for the computer vision system will be mounted. Chapter four details the design, simulation and selection of materials for the subsequent construction of the unmanned aerial vehicle, including the design of the circuit for the implementation of visual markers and the circuit for 2.4 GHz wireless communication of the aircraft with the software of control. Next, in chapter five, the design of the PID controller for the aircraft and the design of algorithms for trajectory generation are shown. Chapter six details the construction of mechanical components of the project, the implementation of electrical and electronic components and software in general, including the implementation of an HMI for supervision and monitoring. The following chapter shows the tests performed in terms of positioning, trajectory tracking and computational cost. Finally, chapter eight presents conclusions, recommendations to improve the functioning of the system and proposals for future work based on the current title work.

Keywords:

- **QUADROTOR**
- **COMPUTER VISIÓN**
- **ROS**
- **POSITION ESTIMATION**
- **GENERATION OF TRAJECTORY**

CAPÍTULO I

DESCRIPCIÓN GENERAL DEL PROYECTO

1.1. Introducción

En el ámbito de la robótica, la utilización de elementos y sistemas de visión por computador para seguimiento visual de un objeto nos provee de información necesaria para que los vehículos aéreos no tripulados puedan ejecutar varias tareas sin necesidad de disponer de un sistema de posicionamiento global (GPS) el cual se basa en satélites. Por esta razón es imprescindible el uso de arreglo de cámaras dentro de un entorno cerrado para la localización de un robot ya sea terrestre o aéreo.

La finalidad de este proyecto es diseñar, desarrollar, implementar y evaluar un sistema de visión por computador para un vehículo aéreo no tripulado en un entorno cerrado utilizando bases de software de ROS (Robot Operating System). Para conseguir los objetivos de este proyecto es necesario aplicar técnicas de visión por computador y de control automático.

El presente documento registra la utilización para el sistema de visión por computador paquetes computacionales como ROS, virtualización de ROS como RVIZ, paquetes y programación basada en OPENCV, filtros digitales y físicos, calibración y adecuación al usar un arreglo de cámaras. Como parte de diseño mecánico presenta un desarrollo CAD del sistema tanto para el sistema de cámaras como para el robot aéreo, además de simulaciones y estudios computacionales realizados sobre el sistema.

Haciendo referencia a la parte de programación y creación de software, se diseña una interfaz gráfica capaz de realizar el control manual o automático y monitoreo del sistema en general, para el caso de programación se basa en lenguaje de código Python, creación de nodos, publicadores y suscriptores sobre ROS, algoritmo Rapidly Exploring Random Trees (RRT) para generación de trayectoria y diseño e implementación de controladores de vuelo digitales (controladores PID).

1.2. Antecedentes

Los vehículos aéreos no tripulados (UAVs) por sus siglas en inglés Unmanned Aerial Vehicles, han generado un área de investigación muy activa durante los últimos años, pudiendo ser controlados remotamente o volar de forma autónoma basándose en planes de vuelo pre programados mediante el uso de sistemas de control complejos.

Este término se originó luego que el pionero en aviación francés Etienne Oehmichen comprobó que era posible la construcción de un cuadricóptero, llamados así por sus cuatro hélices que lo propulsan, con la fabricación del Oehmichen No.2 en 1922. Siendo el primer antecedente histórico de un cuadricóptero practico.

El dotar a los vehículos aéreos no tripulados de capacidad propia para reconocer un obstáculo es un requisito indispensable para aumentar la independencia de estos vehículos y asegurar así la seguridad de las operaciones. De modo que los “sense and avoid drones” fueron considerados por el World Economic Forum como una de las tecnologías emergentes del 2015.

Algunos de los proyectos desarrollados anteriormente en relación a este tema se describen a continuación, destacando como tema principal el control inteligente de un vehículo aéreo no tripulado basado en visión por computador:

A nivel internacional en la ciudad de San Cristóbal De La Laguna, España (Rodríguez, 2015) en la Universidad de La Laguna se desarrolló el proyecto denominado “Sistema de posicionamiento para un drone”, su autor es: Elías Rodríguez. El proyecto presenta el desarrollo de un sistema informático de posicionamiento basado en odometría visual para un drone que vaya a utilizarse en interiores o en lugares donde no se dispone señal GPS.

Además, en la ciudad de Madrid, España (Mejías, 2006) en la Universidad Politécnica de Madrid se desarrolló el trabajo de Tesis Doctoral llamado “Control Visual de un Vehículo Aéreo Autónomo Usando Detección y Seguimiento de

Características en Espacios Exteriores", su autor es: Luis Mejías. En el proyecto se presenta el diseño de robots aéreos guiados por visión, con base en el uso de sensores visuales para realizar tareas como evasión de obstáculos, seguimiento de objetivos o detección de características salientes. Donde se propone técnicas de visión por computador que permitan a un vehículo aéreo realizar maniobras mientras se hace seguimiento de características en entornos donde el GPS tiene fallos de recepción.

De igual forma en la ciudad de Tamaulipas, México (Soto, 2012) en el Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional se desarrolló un proyecto de Tesis para obtener el título de Maestro en Ciencias en Computación denominado "Interacción hombre-robot con vehículos aéreos no tripulados basada en visión", su autor es: Daniel Soto. En la Tesis de Master se presenta el desarrollo de una interfaz entre un vehículo aéreo no tripulado y una persona. Donde la interfaz realiza el rastreo de una persona y el reconocimiento de sus órdenes gestuales, por medio de visión de computadora.

A nivel nacional, en la ciudad de Quito, en la Escuela Politécnica Nacional (María & Maldonado, 2014), se desarrolló el trabajo de titulación "Control por visión de un cuadricóptero utilizando ROS", sus autores fueron María Calderón y Diego Maldonado. El proyecto corresponde a la implementación de un concepto de visión artificial, específicamente reconocimiento y seguimiento de objetos, con un hardware en particular: un cuadricóptero.

Conjuntamente, en la ciudad de Sangolquí, en la Universidad de las Fuerzas Armadas ESPE [11-13], (Aguilar, Casaliglla, & Pólit, 2017), se desarrolló el trabajo de titulación denominado "Obstacle Avoindance Based-Visual Navigation for Micro Aerial Vehicles", sus autores fueron Wilbert Aguilar, Verónica Casaliglla y José Pólit. Este trabajo muestra el diseño e implementación de un sistema de prevención de obstáculos para vehículos aéreos no tripulados de bajo costo UAVs usando visión por computador como la principal fuente de información a través de la cámara monocular a bordo [10,14].

1.3. Justificación e importancia

En la actualidad las aeronaves no tripuladas o drones se encuentran en casi cualquier aplicación imaginable, desde su uso en campos de ingeniería como la realización de fotogrametría para levantamiento topográficos o sobrevuelos en campos para la determinación de la salud de los cultivos por medio de cámaras multi espectrales, hasta en la industria cinematográfica para la filmación de escenas desde nuevos y emocionantes ángulos. Todo esto se logra gracias a complejos algoritmos y sensores los cuales le permiten saber a la aeronave en qué punto del espacio se encuentra exactamente, pero esto tiene una desventaja, ya que el principal sensor que utiliza para su posicionamiento es el GPS, este sensor no funciona dentro de zonas techadas lo cual convierte a un sofisticado drone capaz de realizar tareas autónomas en una simple nave a control remoto.

El desarrollo de este proyecto de investigación, permite realizar el control de rutas de vuelo para drones por medio de cámaras, las cuales capturan el movimiento de marcadores especiales ubicados en la aeronave obteniendo su posición exacta en el espacio varias veces por segundo y por medio de algoritmos matemáticos calcular en tiempo real los movimientos óptimos a ser realizados por el drone para mantenerse dentro de la ruta deseada.

Este proyecto permite abrir campo en el uso de drones en lugares que no tengan señal GPS (indoor), también es muy beneficioso ya que a futuro será posible utilizar más de una aeronave al mismo tiempo sin necesidad de agregar elementos de hardware tanto a los drones como al sistema de captura de movimiento ya que todo será controlado por las mismas cámaras y computador, reduciendo drásticamente de esta manera los costos de implementación de nuevas aeronaves.

El objetivo de relacionar en un mismo conjunto a un vehículo aéreo no tripulado y a varias cámaras dentro de un lugar cerrado es debido a que mediante un sistema de visión por computador se pueda conseguir planificar trayectorias para el vehículo aéreo, logrando que este pueda establecer varias acciones dentro de este entorno, como por ejemplo aplicaciones que conlleven la evasión de obstáculos o la ejecución

de acciones sobre algún determinado objeto que se encuentre en este espacio. Por tal motivo el aporte del vehículo aéreo en el proyecto es la de un actuador en el mismo.

1.4. Área de influencia

El proyecto de investigación pretende la utilización de tecnologías de visión por computador y algoritmos de planificación de trayectoria para la obtención de rutas de vuelo dentro de un entorno cerrado, permitiendo posteriormente la realización de diferentes trabajos investigativos en el área de la Robótica, Automatización y Control.

Adicionalmente, representa una ayuda académica tanto para estudiantes como para docentes, para desarrollar y contribuir en el conocimiento y aprendizaje de técnicas de automatización de procesos como lo es la implementación de sistemas de control inteligentes y la obtención de datos a través de cámaras en general. Considerando que permite fortalecer y ratificar los conceptos y principios adquiridos en el transcurso de la carrera y su relación con los procesos de tipo industrial, revolucionando en la calidad de vida del hombre.

1.5. Alcance del proyecto

En el campo de investigación de este proyecto tiene como alcance el mejorar e implementar los algoritmos y métodos de captura de movimiento actuales, tanto para la determinación de las coordenadas de posicionamiento y orientación de la aeronave en el espacio, como para la planeación autónoma de la ruta de vuelo y su posterior control.

Adicionalmente se deja abierto el proyecto para que a futuro se retome el mismo como un proyecto de titulación en el que puede abarcar el diseño e implementación de un prototipo de vehículo aéreo no tripulado de modo que pueda dimensionarse el mismo para las especificaciones de trabajo sobre el área determinada.

1.6. Objetivos

1.6.1. Objetivo general

Diseñar e implementar un sistema de visión artificial para planeación de trayectoria de un vehículo aéreo no tripulado en un entorno cerrado utilizando ROS.

1.6.2. Objetivos específicos

- Investigar, diseñar y simular por software varios modelos de drone para la obtención de un modelo idóneo para la implementación en este proyecto.
- Diseñar y construir un prototipo de UAV de micro escala, con capacidad de comunicación por radio frecuencia que se ajuste a la aplicación propuesta.
- Construir e implementar marcadores activos en la aeronave que puedan ser detectados con facilidad por el sistema de visión por computador para la estimación de la posición y rotación yaw del vehículo en tiempo real.
- Modificar el sistema de visión de las cámaras con sistema de iluminación infrarroja con la implementación de un filtro y ángulo de grabación amplia para mejorar la captura del movimiento de los marcadores.
- Diseñar y construir un bastidor que permita la colocación fácil y rápida del sistema de captura de movimiento.
- Diseñar e implementar un circuito para la comunicación entre el software y la aeronave.
- Estimar la posición y rotación yaw en tiempo real de la aeronave aplicando procesamiento digital de imágenes.
- Implementar un algoritmo planificación de movimientos para el cálculo de la trayectoria optima a ser utilizada por el drone.
- Diseñar e implementar un sistema de control de seguimiento de ruta en base al conjunto de puntos de la trayectoria obtenidos por el algoritmo de planificación de movimientos.
- Diseñar e implementar una HMI que permita realizar la programación y monitoreo de las rutas de vuelo al drone en tiempo real.

1.7. Estructura del documento

El presente documento consta de ocho capítulos, el primer capítulo se presenta la descripción general del proyecto como definición del mismo, además del planteamiento del problema con su respectiva justificación e importancia, su alcance, área de influencia y los objetivos del proyecto.

En el capítulo dos, generalizado como “Estado del Arte”, menciona fundamentos teóricos del proyecto de investigación haciendo referencia a temas como: conceptos y tipos de vehículos aéreos no tripulados, fundamentos de sistemas de visión por computador, sistemas digitales de procesamiento de imágenes, algoritmos de control para planificación de trayectoria, software utilizado en el proyecto y teoría de controladores.

En el capítulo tres, denominado “Diseño de bastidor para montaje del sistema de cámaras y redes de seguridad”, presenta el diseño mecánico, eléctrico y electrónico, así como la selección de materiales para la implementación del sistema de montaje de cámaras.

En el capítulo cuatro, llamado “Diseño y simulación de modelos de aeronaves”, presenta el diseño mecánico, simulación y selección de materiales para un vehículo aéreo no tripulado implementado en el proyecto de investigación.

En el capítulo cinco, referenciado como “Sistema de control”, presenta el diseño, dimensionamiento y selección de dispositivos para el sistema de control, así como la definición de algoritmos de control para planificación de trayectorias de ruta de vuelo.

En el capítulo seis, denominado “Construcción e implementación”, presenta el proceso de construcción de la aeronave, el bastidor, la implementación del sistema de visión por computador, construcción del circuito de comunicación y la implementación de filtros de procesamiento de imágenes y algoritmo de control.

En el capítulo siete, llamado “Pruebas y resultados”, se describen los resultados obtenidos con respecto al sistema de visión por computador y la aplicación del algoritmo de control para la planificación de trayectoria en base a la implementación de un controlador.

Finalmente, en el capítulo ocho se detallan las conclusiones y recomendaciones alcanzadas en la ejecución y desarrollo del proyecto de investigación y propuestas de trabajos futuros basándose en este tema.

CAPÍTULO II

ESTADO DEL ARTE

2.1. Vehículos aéreos no tripulados UAVs

2.1.1 Definición

En la actualidad existe un amplio espectro de aeronaves con la capacidad de realizar misiones con cierto grado de autonomía. En el pasado este tipo de aparatos fueron denominados ROA (Remotely Piloted Aircraft) o UA (Unmmaned Aircraft o Uninhabited Aircraft), mientras que en la actualidad se designan a este tipo de aeronaves como UAV (Unmanned Aerial Vehicle) (Barrientos, y otros, 2007).

Esta denominación hace referencia a la carencia de tripulación en el vehículo aéreo. Debe entenderse que esta condición no excluye la existencia de piloto, controlador u operador que puedan realizar un trabajo sobre la aeronave desde tierra. En lo que sigue se utilizara el termino UAV de manera indiferente, considerando que esta no transporta personas y puede ser total o parcialmente controlada desde la estación de tierra.

En los últimos años, la proyección de estos sistemas desde el sector civil al militar, ha generado que diferentes grupos de investigadores ligados a la aérea de la robótica se hayan enfocado a la investigación y desarrollo de estos. Por este motivo al ser parte de un ámbito aeronáutico, se utiliza en ocasiones el término robot aéreo, entendiendo a este como un sistema físico, capaz de realizar una determinada acción en el aire de manera autónoma o semiautónoma.

Figura 1 Vehículo aéreo no tripulado UAV

Los UAVs tienen un gran potencial para realizar diversas tareas, puesto que pueden acceder a sitios donde las condiciones climatológicas o cartográficas sean complejas. Además proporcionan información muy valiosa con imágenes a vuelo de pájaro y datos en tiempo real que proporcionan los sensores incorporados en el aparato.

Entre las aplicaciones comúnmente utilizadas de los UAVs están las siguientes:

- Rastreo de personas u objetos
- Video, fotografía y cartografía aérea
- Seguridad y aplicaciones militares
- Exploración de lugares de difícil acceso
- Construcción e inspecciones
- Control y monitoreo en la agricultura, etc.

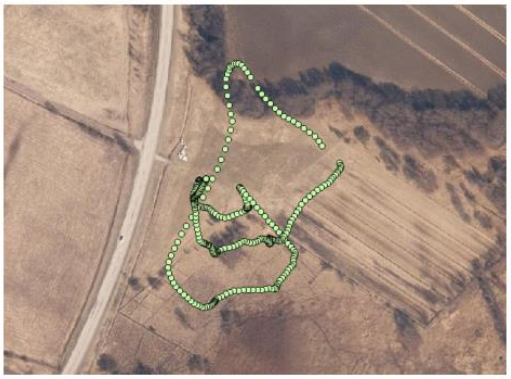


Figura 2 Aplicación UAV sector agrícola
Fuente: (Saari, y otros, 2017)

2.1.2 Clasificación

Al momento de establecer una clasificación de los UAVs se atiende a diferentes criterios. Siendo el más simple el que se basa en el tipo de aeronave. De acuerdo a esto puede distinguirse a las de despegue vertical de las que no lo son, entre las primeras se tienen a las de ala rotativa o hélices (quadrotors y helicópteros), los de ala flexible (parapentes, ala delta) y por último los auto sustentados (dirigibles y globos aerostáticos). Dentro de los no poseen despegue vertical, se encuentran los de ala fija (aeroplanos). En la Figura 3 se muestra un cuadro de clasificación de los UAVs.

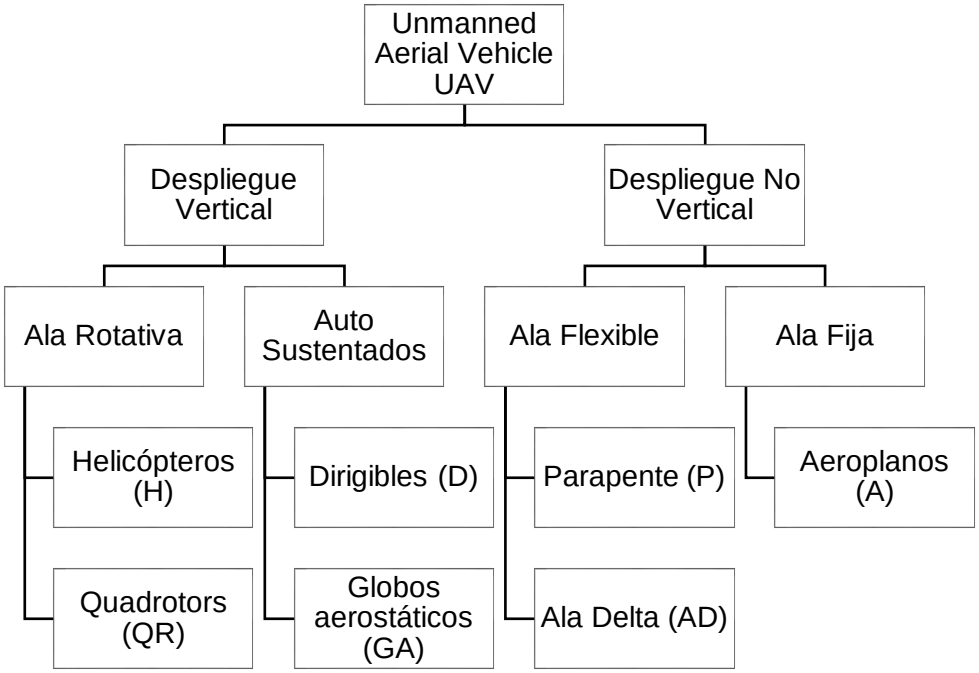


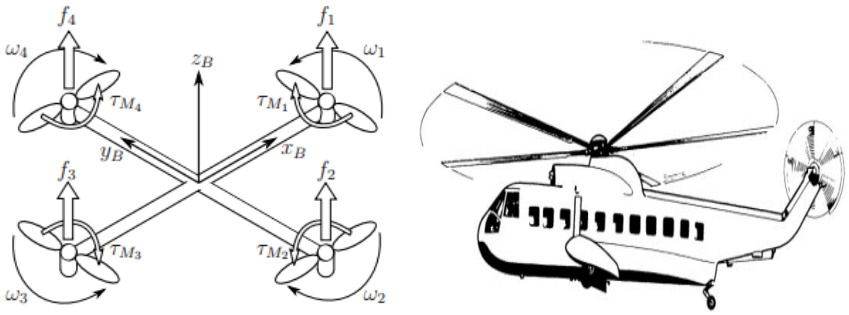
Figura 3 Algunos tipos de aeronaves utilizadas en los UAVs
Fuente: (Barrientos, y otros, 2007)

Las prestaciones y aplicaciones varían mucho de un tipo de aeronave a otra, cada una de ellas cubre un espectro de aplicabilidad diferente. La Tabla 1 recoge las características básicas de algunas principales aeronaves utilizadas para UAVs. Mientras que de la Figura 4 a Figura 7 se muestra una representación gráfica de algunos UAVs.

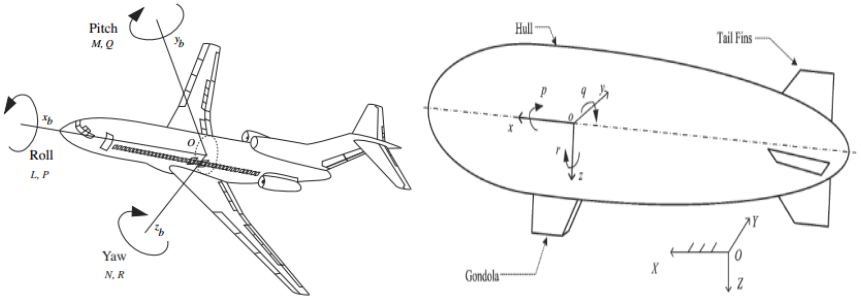
Tabla 1
Características de los principales tipos de aeronaves para UAVs

Característica	Helicópteros	Aeroplanos	Dirigibles	Quadrotors
Capacidad de vuelo estacionario	***		****	***
Velocidad de desplazamiento	***	****	*	**
Maniobrabilidad	***	*	*	****
Autonomía de vuelo (tiempo)	**	***	****	*
Resistencia a perturbaciones externas (viento)	**	****	*	**
Auto estabilidad	*	***	****	**
Capacidad de vuelos verticales	****	*	**	****
Capacidad de carga	***	****	*	**
Capacidad de vuelo en interiores	**	*	***	****
Techo de vuelo	**	****	***	*

Fuente: (Barrientos, y otros, 2007)



(a) (b)



(c) (d)

(a) *Figura 4* Quadrotor
Fuente: (Luukkonen, 2011)
(b) *Figura 5* Helicóptero autónomo
Fuente: (Johnson, 1994)
(c) *Figura 6* Aeroplano autónomo
Fuente: (Hull, 2007)
(d) *Figura 7* Dirigible autónomo
Fuente: (Li, 2008)

Por último se muestra en la Tabla 2 una clasificación de acuerdo al nivel de autonomía de la aeronave.

Tabla 2
Clasificación de principales UAVs de acuerdo al nivel de autonomía

	Blanco móvil (Drone)	Nivel 1 No autónomo	Nivel 2 Maniobra autónoma	Nivel 3 Piloto inteligente limitado	Nivel 4 Piloto inteligente completo
Trayectoria de vuelo en 4 dimensiones	Pre programada	Pre planificada y pre programada	Pre planificada y pre programada	Parcialmente autónomo, posibilidad de cálculo de ruta	Posibilidad de cálculo de ruta
Presencia de piloto	Por seguridad	Guiado y control continuo	Como Nivel 1	Supervisión continua, guiado y control ocasional	Solo como respaldo

Continua Tabla 2 →

Necesidad de actuación del piloto	Solo en FTS	En todos los casos	Cuando no haya maniobra automática	Ocasionalmente	Solo como respaldo
Piloto automático	No	No	No	Limitado	Completo
Presencia de ATC	Supervisión	Contacto continuo con el operador	Contacto continuo con el operador	Contacto continuo con el operador y con el piloto automático	Contacto continuo con el piloto automática (con el operador como respaldo)
Intervención de ATC	Petición o activación del FTS	Por petición del operador	Por petición del operador	Por petición del operador o del piloto automático	Por petición del piloto automática (del operador por respaldo)

Fuente: (Barrientos, y otros, 2007)

2.2. Quadrotor o cuatrirotor

Un Quadrotor equivale a una aeronave que dispone de cuatro brazos, estos sitúan en sus extremos motores con hélices que permite alzar el vuelo del aparato y desarrollar un empuje. Su estabilidad se consigue debido a que dos hélices giran en sentido horario y las otras dos restantes lo hacen en sentido anti horario. Con esto los efectos giroscópicos y los momentos aerodinámicos tienden a cancelarse en vuelo estacionario.

Un sistema Quadrotor está compuesto generalmente de dos partes, la estación en tierra y su parte aérea. La estación en tierra o móvil asegura la preparación de su misión, explotación, la comunicación de datos con su parte aérea y los componentes que controlan a la aeronave. El subsistema aéreo está compuesto de una o varias plataformas compuestas de un componente útil adaptado a la misión (GPS, cámara de video, varios sensores, controlador de vuelo, motores, etc.), además de un sistema de envío y recepción de información.

Estos generalmente son de talla reducida, de bajo coste y de una complejidad menor de construcción en comparación a un aeroplano. Su tamaño puede variar desde algunos centímetros hasta varios metros de longitud, al igual que su forma y tipo de propulsión (Castillo, Garcia, Lozano, & Albertos, 2007). La entrada de control principal o fuerza principal es la suma de cada fuerza producida por cada elemento rotativo, siendo de esta forma:

$$u = \sum_{i=1}^{4} fi \qquad (1)$$

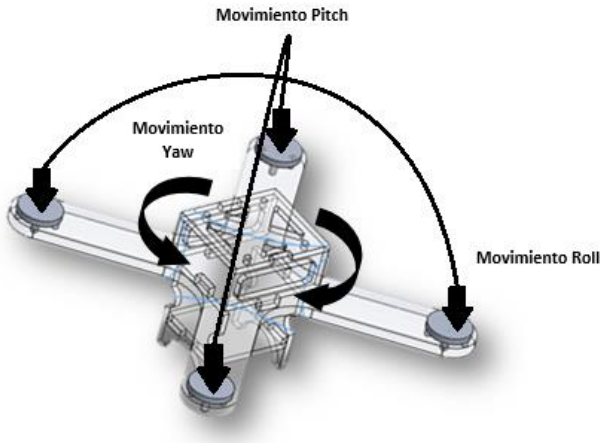


Figura 8 Momentos de pitch, yaw y roll

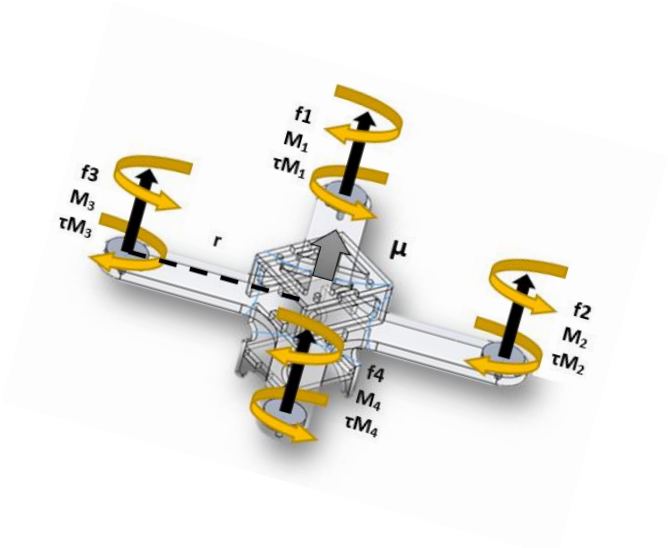


Figura 9 Esquema de fuerzas de un cuatrirotor y momentos de cada motor

En vista de esta configuración, se puede destacar que el cuatrirotor tiene muchas similitudes con el tipo de avión PVTOL (Planar Vertical TakeOff and Landing). Se puede apreciar que al anular los ángulos de roll, pitch o yaw el Quadrotor se reduce a dos PVTOLs fusionados por su eje ortogonal (Castillo, Lozano, & Dzul, Modelling and Control of Mini-Flying Machines, 2005).

El momento de pitch[1] es producido por la diferencia de $f1 - f3$, a su vez el momento en roll[2] es producido por $f2 - f4$ y finalmente el momento en yaw[3] es producido por la suma $\tau_{M1} + \tau_{M2} + \tau_{M3} + \tau_{M4}$, donde τ_{Mi} es el momento de reacción del motor i $(\text{Castillo, Garcia, Lozano, \& Albertos, 2007})$. Ver Figura 8 y Figura 9.

2.2.1 Componentes

De acuerdo al fabricante los componentes de un cuatrirotor pueden tener variaciones, pero en general contienen elementos eléctricos, electrónicos y mecánicos comunes en estos:

2.2.1.1 Controlador de vuelo

Este elemento se encarga del control de velocidad de cada motor, además del envío y recepción de datos e información desde y hacia un dispositivo externo. Este cuenta con varios elementos que combinados se consigue la suficiente información del entorno y de este modo tomar las decisiones correctas sobre los actuadores, ver Figura 10. A continuación se enlistan estos elementos:

- Giróscopo: este permite obtener la velocidad angular de la aeronave ante cambios de posición.
- Acelerómetro: permite medir la propia inercia de los movimientos del cuatrirotor.
- Magnetómetro: es un dispositivo que ejecuta la acción de una brújula, el cual envía información de la dirección en la que apunta el quadrotor.

[1] pitch: inclinación hacia adelante y hacia atrás del cuatrirotor
[2] roll: inclinación de izquierda a derecha o viceversa del cuatrirotor
[3] yaw: rotación hacia la izquierda o derecha del cuatrirotor

- Sensor barométrico: se emplea generalmente para conocer la altura real de vuelo del aparato.
- GPS: permite acceder y conocer a las coordenadas exactas en el entorno en el que se encuentre el cuatrirotor y así poder desplazarse de forma autónoma.
- Procesador: este debe ser lo suficientemente potente para recibir, procesar y reenviar información en tiempo real. (Grewal, Weill, & Andrews, 2007)

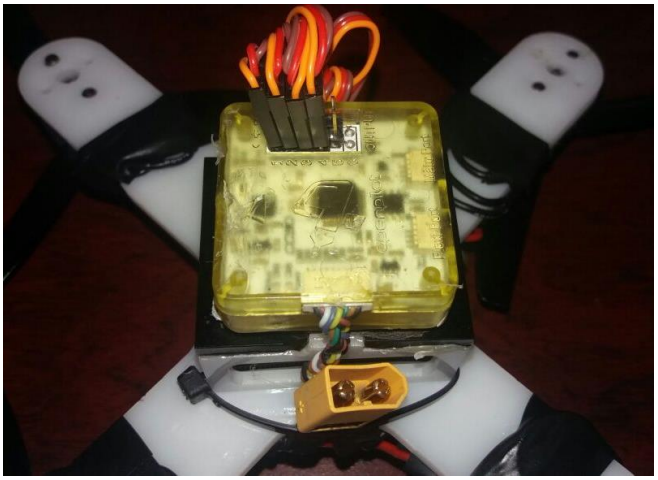


Figura 10 Controlador de vuelo CC3D

2.2.1.2 Motores

Son los elementos encargados de transmitir la energía eléctrica en energía mecánica hacia las hélices y permitir la sustentación de la aeronave. Estos pueden ser de tipo Brushed y Brushless.

- **Brushed**

Su funcionamiento se basa en un conjunto de bobinas giratorias que actúan como un electroimán con dos polos. El inducido o rotor es un electroimán, y el imán de campo es un imán permanente, ver Figura 11. El conmutador es un dispositivo envuelto alrededor de un eje que mantiene contacto físico con los cepillos y estos a su vez están conectados a la fuente de alimentación permitiendo el giro de la armadura. La dirección de rotación se puede determinar invirtiendo la polaridad de los cepillos. (Reston Condit, Microchip Technology Inc., 2004)

- **Brushless**

Este tipo de motor utiliza un rotor externo de imán permanente, bobinas de tres fases de conducción y uno o más dispositivos de efecto Hall de modo que se pueda detectar con este la posición del rotor, ver Figura 11. Las bobinas se activan, una fase después de la otra permitiendo el giro del motor con una frecuencia variable. (Padmaraja Yedamale, Microchip Technology Inc., 2003)

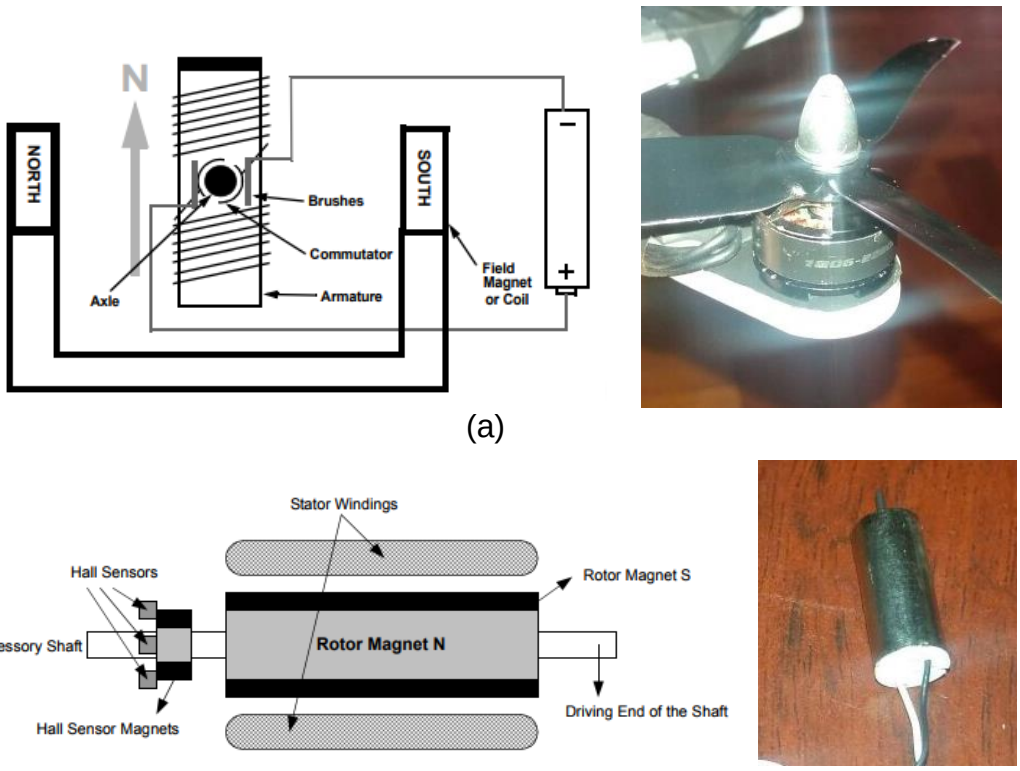


Figura 11 (a) Motor Brushless
Fuente: (Reston Condit, Microchip Technology Inc., 2004); (b) Motor Brushed
Fuente: (Padmaraja Yedamale, Microchip Technology Inc., 2003)

2.2.1.3 Electronic Speed Controller ESC

Son componentes permiten el control de cada uno de los motores convirtiendo la corriente continua de una batería a una fuente de tensión variable y de sentido reversible para cada motor, ver Figura 12. (Futaba Corporation, 2011)

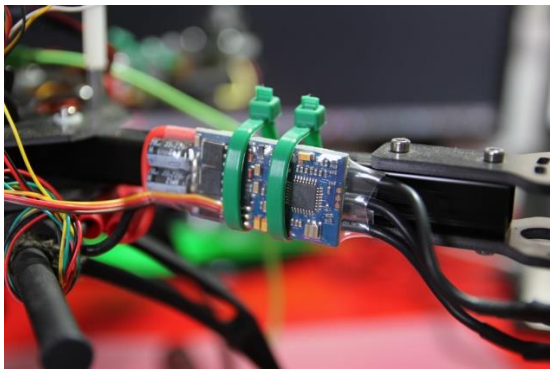


Figura 12 Controlador de velocidad de motor, Electronic Speed Controller ESC

Comercialmente son clasificados de acuerdo a la máxima corriente soportada, disponen generalmente de dos terminales para su alimentación, tres pines de conexión para 5Vcc, tierra y la entrada de señal desde el microcontrolador, además de tres pines para la conexión con los motores tipo Brushless. (Futaba Corporation, 2011)

2.2.1.4 Hélices

Son los elementos que permiten la sustentación de la aeronave al ser giradas por los motores, estas pueden estar construidas de varios materiales y diferentes tamaños, ver Figura 13. Para el caso de estos elementos se tiene varias consideraciones que se enlistan a continuación:

- Un aumento en el grado de inclinación o del tamaño supondrá un mayor empuje.
- Un grado de inclinación menor permite un giro mucho más rápido.
- Un grado de inclinación mayor consiguen mover una mayor cantidad de aire.
- Los principales materiales de fabricación de hélices son la fibra de carbono, la madera y polímeros en general. (Gettinger & Holland, 2015)

Figura 13 Hélices para cuatrirotor

2.2.1.5 Batería LiPo

La batería es uno de los elementos que se debe tener mayor cuidado a elegir, generalmente se emplean baterías tipo LiPo. Las tensiones junto con su capacidad y peso son características principales de una batería, estas se deben elegir en base a los motores y elementos electrónicos que posea la aeronave, ver Figura 14.

Figura 14 Baterías tipo LiPo

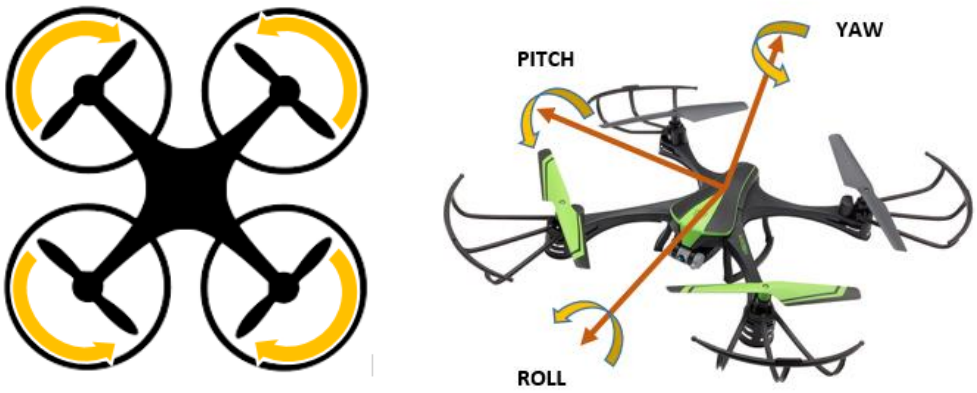
Figura 15 Frame o estructura de un cuatrirotor

2.2.1.6 Frame o armazón

Es el cuerpo donde se van a ensamblar todos los componentes de la aeronave, esta debe ser de un material resistente y liviano. Existen diferentes tamaños y modelos de frames donde su principal función es reducir las vibraciones y proteger los demás componentes del cuatrirotor, ver Figura 15.

2.2.2. Funcionamiento

El movimiento en el cuatrirotor es generado por sus hélices, las cuales al rotar dos en sentido horario y las otras dos en sentido anti horario, logra una estabilidad al equipo siempre y cuando el par generado por los motores sea el mismo. Además cuando se combinan velocidades en los motores se producen movimientos de rotación y traslación en la aeronave, produciendo cuatro principales movimientos, tres de traslación para el caso de pitch, roll y gaz o throttle[4], y uno de rotación para el caso del movimiento yaw, estos movimiento dependen de su eje, tal y como se muestra en la Figura 16.



(a) (b)

Figura 16 (a) Rotaciones de hélices de un cuatrirotor; (b) Rotación del cuatrirotor en relación a sus ejes

[4] throttle: controla el movimiento vertical hacia arriba o hacia abaja del cuatrirotor

2.2.2.1 Movimiento Pitch

Este movimiento dirige a la aeronave hacia delante (pitch +) o hacia atrás (pitch -) con referencia a su frente. Uno de sus motores aumenta su velocidad, mientras que otro disminuye con respecto a su velocidad promedio. Véase Figura 17.

Figura 17 Movimiento Pitch. Flechas de color verde muestran aumento de velocidad, flechas de color naranja velocidad promedio y flecha de color rojo representan reducción de velocidad

2.2.2.2 Movimiento Roll

Este movimiento dirige a la aeronave hacia la derecha (roll +) o hacia la izquierda (roll -) con referencia a su frente. Uno de sus motores aumenta su velocidad, mientras que otro disminuye con respecto a su velocidad promedio. Véase Figura 18.

Figura 18 Movimiento Roll. Flechas de color verde muestran aumento de velocidad, flechas de color naranja velocidad promedio y flecha de color rojo representan reducción de velocidad

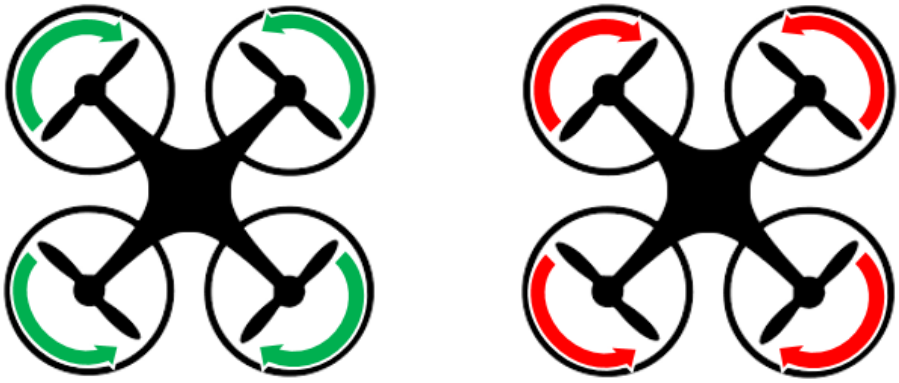
2.2.2.3 Movimiento Yaw

Este movimiento rota la aeronave en relación al eje ortogonal de pitch y roll, dos de sus motores aumentan su velocidad, mientras que otros dos mantienen su velocidad promedio. Véase Figura 19.

Figura 19 Movimiento Yaw. Flechas de color verde muestran aumento de velocidad y flechas de color rojo representan reducción de velocidad

2.2.2.4 Movimiento Gaz o Throttle

Este movimiento asciende o desciende la aeronave, para esto se debe aumentar o disminuir la velocidad de los cuatro motores simultáneamente. Véase Figura 20.



(a) (b)

Figura 20 (a) Aumento de velocidad y altura; (b) Reducción de velocidad y altura

2.3. Marcadores

Son elementos mecánicos o electrónicos que permiten la detección y registro de movimiento de un dispositivo para una evaluación posterior de esta información. Un sistema óptico de captura de movimiento detecta la presencia del marcador y puede registrar diversas variables físicas que se desee analizar pudiendo estar entre estas la posición, velocidad, etc. (Rozantsev, Lepetit, & Fua, 2014). En la Tabla 3 se presenta una comparativa entre los distintos marcadores disponibles.

2.3.1 Marcadores Pasivos

Los marcadores pasivos son elementos mecánicos que disponen de una superficie porosa reflectante, estos elementos reflejan la luz externa proveniente de un dispositivo generador de luz que se encuentra situado cercano a la cámara, de este modo se permite la localización puntual del marcador. (Rozantsev, Lepetit, & Fua, 2014). Ver Figura 21.

Figura 21 Marcador pasivo colocado en un vehículo aéreo no tripulado

2.3.2 Marcadores Activos

Los marcadores activos son elementos electrónicos poseen un elemento generador de luz en su composición, permitiendo a la cámara reconocer el marcador por medio

de la luz emitida por este elemento. Para este tipo de marcadores es necesario disponer de una fuente de alimentación para el mismo y un circuito de protección o regulación de voltaje. (Rozantsev, Lepetit, & Fua, 2014). Ver Figura 22.

Figura 22 Marcador activo colocado en un vehículo aéreo no tripulado

Tabla 3

Comparativo entre marcadores activos y pasivos disponibles

Tipo de marcador	Visibilidad	Rango	Descripción
Pasivo esférico	Todas direcciones	4 a 10 metros	Depende del tamaño del marcador y la resolución de la cámara
Pasivo esférico pintado	Todas las direcciones	3 a7 metros	Depende del tamaño del marcador y la resolución de la cámara
Activo LED	Limitada: ± 60°	10 metros	Necesita fuente de alimentación
Activo LED con esfera difusora	Limitada: ± 270°	4.5 metros	Requiere de elemento difusor, necesita fuente de alimentación
Activo esférico varios LEDs	Todas las direcciones	20 metros	Funciona en exteriores, necesita fuente de alimentación
Activo plano varios LEDs	Limitada: ± 90°	20 metros	Funciona en exteriores, necesita fuente de alimentación

2.4. Quadrotor: Parrot Mambo

Cuatrirotor que pertenece a la gama de minidrones de la empresa de producción de robots aéreos Parrot. Es considerado como uno de los minidrones más estables del mercado gracias a su pilotaje automático. Ver Figura 23.

Figura 23 Cuatrirotor Parrot Mambo

Esta aeronave dispone de un controlador de vuelo muy avanzado e incorpora sensores muy eficientes, facilitando su pilotaje en interiores y exteriores. Este dispone de las siguientes especificaciones técnicas:

- Sensores de estabilización: unidad de medida inercial para evaluar la velocidad, la inclinación y el contacto con obstáculos (acelerómetro de 3 ejes y giroscopio de 3 ejes).
- Estabilización vertical: sensor de ultrasonido y sensor de presión
- Estabilización horizontal: sensor de cámara
- Energía: batería LiPo 550mAh
- Diseño aeronáutico: software de control de vuelo y estabilización, proporción peso/velocidad optimizada
- Compatibilidad SDK OS Linux

2.5. Fundamentos de sistemas de visión por computador

Los sistemas de visión por computador son catalogados como un campo de la inteligencia artificial, mediante complejos algoritmos y procesos nos permiten obtener datos de un entorno visual sin tener que estar directamente en contacto con el objeto de medición, pero para la obtención final de la variable deseada es necesario que la imagen pase por varios procesos de discriminación de datos, estos sistemas están diseñados para funcionar exactamente como el conjunto de los ojos y el cerebro humano.

Gracias a estos sistemas es posible obtener variables del entorno como:

- La temperatura de objetos, ver Figura 24
- Posición de objetos en el espacio, ver Figura 25
- Reconocimiento de formas o caracteres, ver Figura 26
- Obtención de relaciones espaciales

Con la obtención de las variables antes descritas es posible realizar sistemas como:

- Sistemas de estabilización de imágenes
- Sistemas de detección de eventos
- Sistemas de control de calidad
- Inspección de objetos sin contacto físico

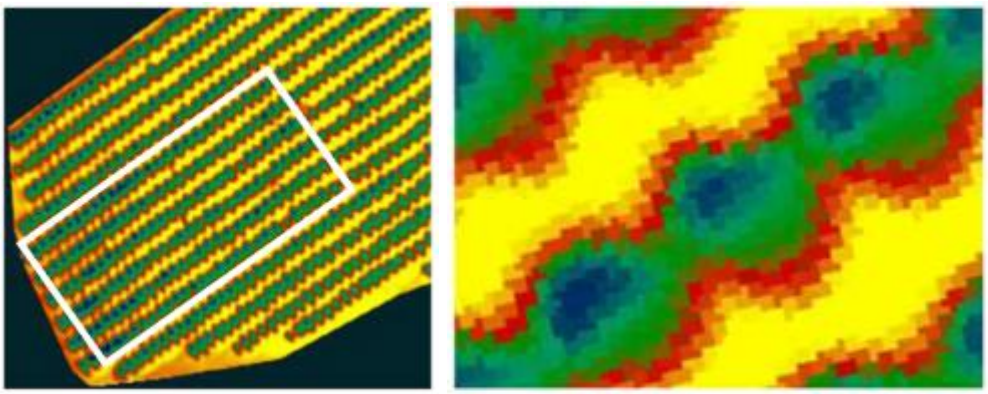


Figura 24 Cámara térmica
Fuente: (Berni, Zarco-Tejada, Suarez, & Fereres, 2009)

Figura 25 Obtención de posición
Fuente: (Preiss, Honig, Sukhatme, & Ayanian, 2017)

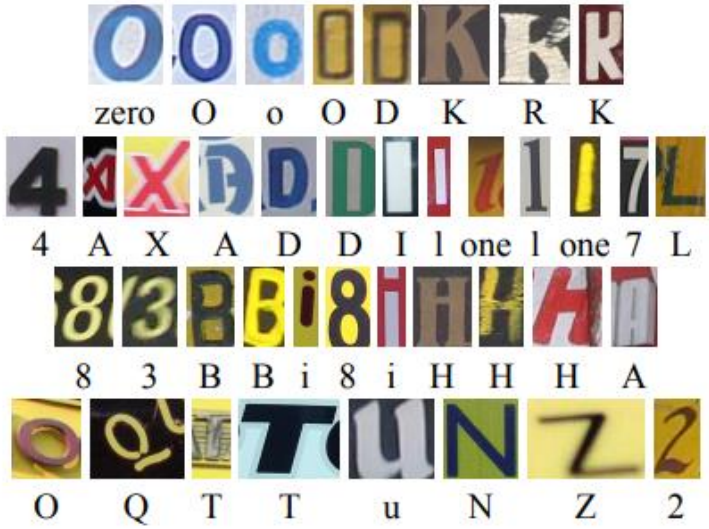


Figura 26 Determinación de caracteres
Fuente: (Kumar, Gupta, Khana, Chaudhury, & Joshi, 2007)

Para la realización un sistema de visión por computador eficiente es muy importante tener en cuenta ciertos aspectos de los sensores (en este caso las cámaras) ya que de esto depende el resultado final, a continuación, describimos algunos de estos factores.

La forma en que nosotros percibimos la información suministrada por el sensor es como una imagen, pero en realidad nos está entregando una matriz de píxeles en donde cada píxel tiene un arreglo de 3 colores RGB o rojo, verde, azul por sus siglas en inglés, ver Figura 27, estos 3 colores primarios pueden tomar valores de 0 a 256 cada uno, esto representa la saturación o intensidad de cada color y con la

combinación de estos 3 es posible generar una variedad de casi 17 millones de colores (16777216) por píxel.

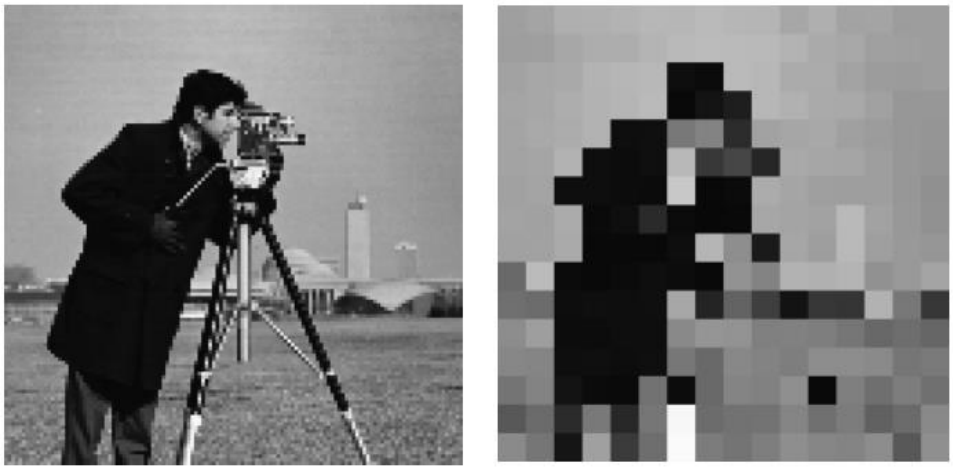


Figura 27 Pixeles en una imagen
Fuente: (Prasantha, Shashidhara, & Balasubramanya, 2009)

Es posible que la imagen se encuentre en escala de grises, es decir, que cada píxel solo tiene un elemento el cual es la saturación o intensidad del blanco, el valor del 0 equivale a negro completamente, y 256 el blanco completo, de aquí los valores intermedios se aclaran u oscurecen en función de que extremo se acerque, este concepto se aprecia mejor en la Figura 28.

Figura 28 Matriz de imagen en escala de grises
Fuente: (Fridrich, Soukal, & Lukas, 2003)

También existe las imágenes binarias en las cuales básicamente cada píxel tiene únicamente 2 valores posibles, 0 (negro) o 1 (blanco) como se observa en la Figura 29, este tipo de imágenes son útiles principalmente para la detección de contornos ya que en este caso no importa la intensidad de color, sino la diferenciación de una cosa de otra.

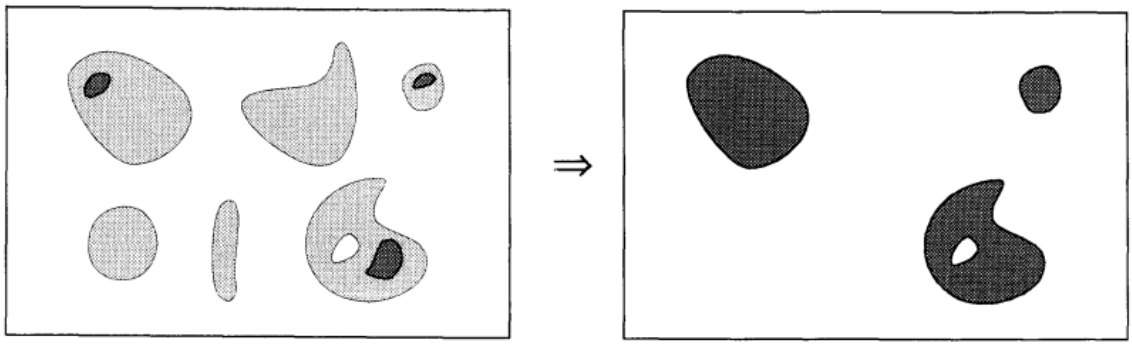


Figura 29 Imágenes binarias
Fuente: (Vincent, 1993)

La resolución de la imagen se traduce en la definición más pequeña que puede ser capturado por el sensor, de esta manera si la imagen tiene más píxeles o mayor resolución, es posible captar más detalles con el mismo campo de visión que de un sensor con menor resolución, esta diferencia es más apreciable en la Figura 30. Existen cámaras con resoluciones que van desde 300x400 píxeles hasta cámaras de resolución 4k que se traduce en 4096 × 2160 píxeles.

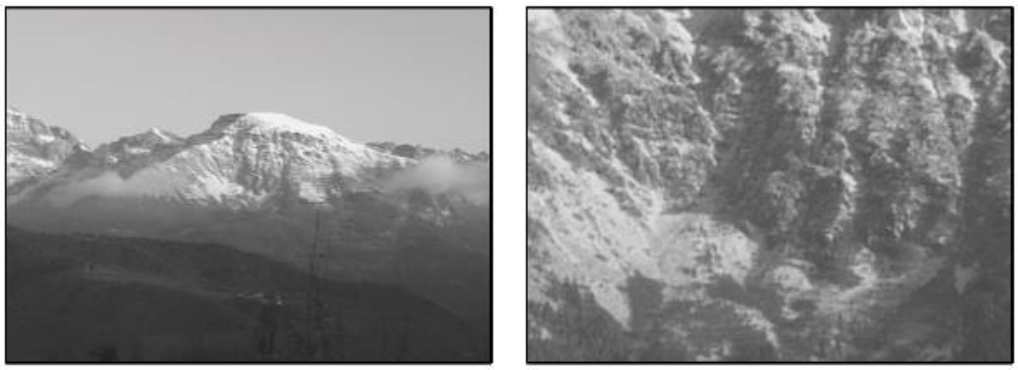


Figura 30 Ejemplo de imágenes en baja y alta resolución
Fuente: (Dufournaud, Schmid, & Horaud, 2000)

Otro factor importante es la frecuencia de trabajo del sensor, por lo general es conocido como FPS o cuadros por segundos por sus siglas en inglés, esto se refiere a la cantidad de imágenes por segundo que el sensor puede captar, para que podamos tomar a una secuencia de imágenes estáticas como una sola en movimiento mínimo se necesita de 24 imágenes por segundo para que la fluidez del movimiento no se vea afectado, hoy en día el estándar es de 30fps en las cámaras más sencillas pero existen cámaras de 60, 120 fps, incluso existen cámaras denominadas de alta

velocidad con las cuales es posible captar el desplazamiento de una bala disparada a través del aire, estas cámaras son capaces de captar hasta 32000 fps.

Por último, es necesario conocer cuál es la óptica de la cámara o mejor conocido como ángulo de visión, este factor depende de la cantidad, disposición y forma de las lentes que se encuentren frente al sensor, esto tiene varias funciones como las de generar un zoom óptico en la imagen, es decir, poder observar objetos lejanos o la de cambiar el campo visual de la cámara, esto hace que desde la misma posición la cámara sea capaz de captar una imagen más grande o más reducida. En función del tipo de óptica que se utilice es necesario un post procesamiento ya que las cámaras de stock eliminan el efecto ojo de pez que producen ciertas lentes, pero al cambiarlas este efecto cambia y las imágenes obtenidas son diferentes, por lo general la imagen tiene un aspecto esférico.

Si se utiliza este tipo de lentes, es necesario realizar un proceso llamado calibración donde es posible eliminar esta forma radial, este efecto es apreciable en la Figura 31, pero se pierde bastante tamaño de la imagen original ya que al final se obtiene una imagen plana, pero con filos faltantes y de esta forma es inservible realizar el procesamiento de imágenes. El procedo consiste en tomar una serie de imágenes con la cámara apuntando hacia una impresión parecida a un tablero de ajedrez de filos rectos, este proceso es apreciable en la Figura 32, la cámara mostrara una imagen con filos curvados por efecto de la lente, pero al correr el proceso de calibración se compara la imagen que se tiene de la cámara y la que se debería obtener y esto hace que se puedan obtener las variables intrínsecas y extrínsecas tanto de la cámara como del espacio, con estos valores es posible guerrear una calibración es decir que las imágenes provenientes por la cámara sean transformadas con estas variables a una imagen normal.

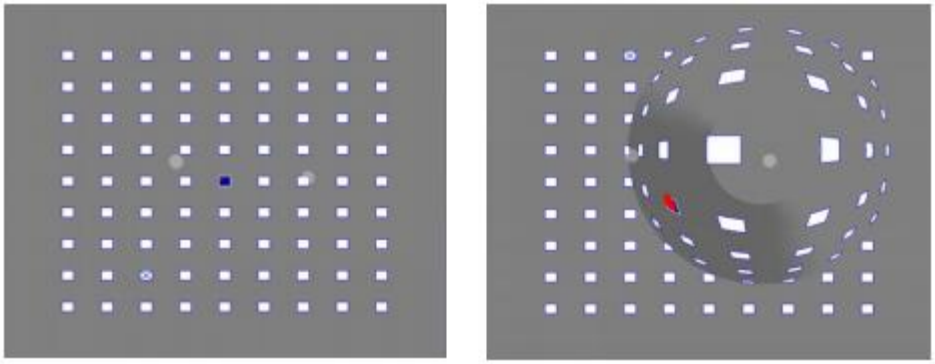


Figura 31 Efecto ojo de pez
Fuente: (Miniotas, Spakov, & MacKenzie, 2004)

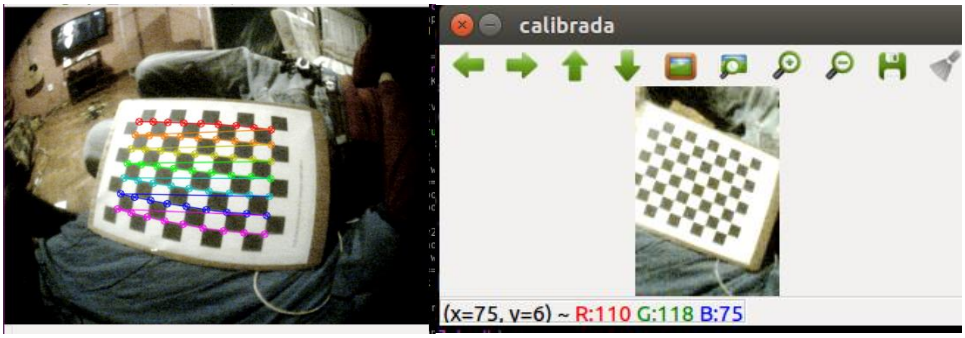


Figura 32 Proceso de calibración de la cámara

En los sistemas de visión por computador es necesario seleccionar con cuidado las características del sistema que suplan sus necesidades, esto es muy importante ya que por lo general es mal entendida la relación de “más es mejor”, en casos como estos el tener una cámara de 4k a 120 fps generaría que el sistema de visión por computador corra muy lento ya que sin un procesador adecuado, el sistema tiene millones de datos que analizar volviendo el proceso ineficiente y con sobrecalentamiento por parte del procesador. No necesariamente una mejor resolución o mayor cantidad de cuadros por segundo significa un mejor sistema.

El proceso para la obtención de datos por medio de visión por computador es un conjunto de pasos ordenados los cuales se describe en la Figura 33.

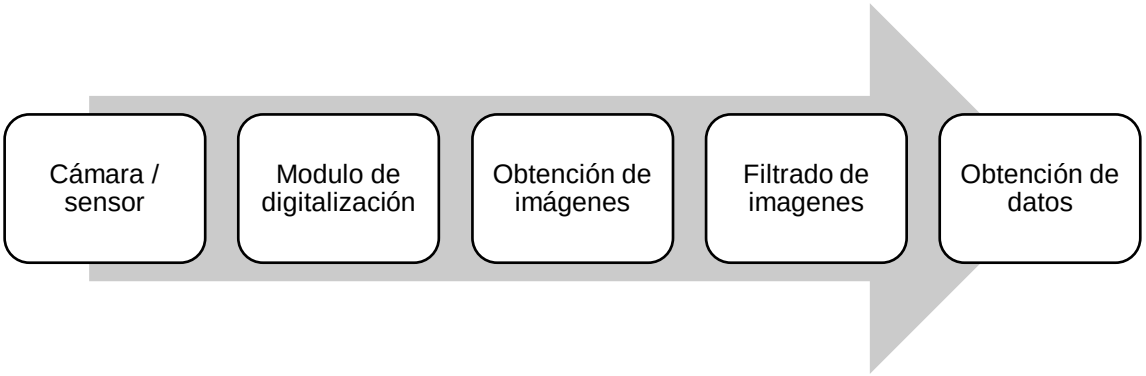


Figura 33 Proceso de visión por computador

2.6. OpenCV

2.6.1 Concepto

OpenCV es una librería open source de visión por computador la cual es capaz de correr bajo Windows, Linux, OS iOS o Android, soporta múltiples lenguajes de programación como C, C++, Python, Java y sirve tanto para fines educativos como comerciales. OpenCV está enfocado para el uso en aplicaciones en tiempo real ya que su núcleo esta optimizado, generando buena eficiencia al tomar ventaja del procesamiento de multi-core.

Esta librería contiene más de 2500 algoritmos los cuales ayudan a la detección de rostros, identificación de objetos y acciones humanas, rastrear movimientos, etc. A continuación, presentamos algunos temas importantes que se pueden realizar con OpenCV.

- Introducción a OpenCV
- Funcionalidades del núcleo
- Procesamiento de imágenes
- Calibración de cámaras
- Entrada y salida de video
- Características 2D
- Detección de objetos
- Unión de imágenes

2.6.2 Introducción a OpenCV

Entre las cosas más básicas que se pueden realizar en OpenCV es el trabajo con imágenes, es posible carga imágenes en un espacio de memoria asignado una variable para posteriormente mostrarla. Véase la Figura 34.

Figura 34 Imagen de espacio de memoria OpenCV
Fuente: (Parker, 2011)

Una vez cargada la imagen en una variable es posible hacer algunas modificaciones simples a esta como cambiarla a escala de grises, o a otro tipo de espacio de color, también se puede guardar las imágenes de la memoria en rutas especificadas. Véase la Figura 35.

Figura 35 Imagen cambiada a escala de grises OpenCV
Fuente: (Lorsakul & Suthakorn, 2007)

2.6.3 Funcionalidades del núcleo

Es posible realizar dibujos básicos sobre un espacio de memoria denotado como imagen. Se puede dibujar líneas, elipses, rectángulos, círculos, y poli líneas, todo esto dándole especificaciones como el grosor de la línea, el color, la ubicación en coordenadas de X e Y. Véase la Figura 36.

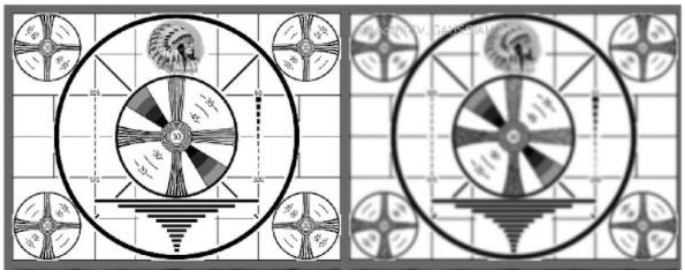


Figura 36 Dibujos básicos OpenCV
Fuente: (Bradski & Kaebler, 2011)

También es posible insertar texto dentro de la imagen basta con especificar la posición, el texto y la fuente. Véase la Figura 37.

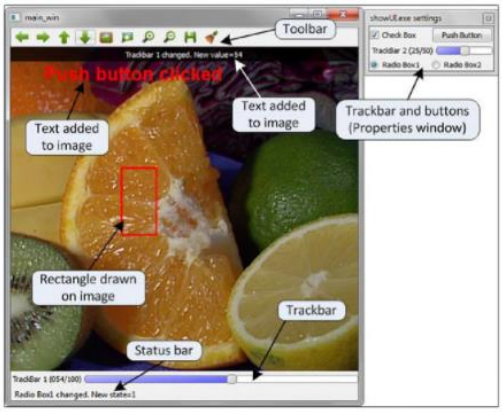


Figura 37 Inserción de texto OpenCV
Fuente: (Bueno Garcia, y otros, 2015)

2.6.4 Procesamiento de imágenes

OpenCV es capaz de realizar procesamiento de imágenes para transfórmalas en función de los requerimientos. Se tienen varios algoritmos para filtra imágenes, estos realizan suavizado de imágenes, el cual elimina ruidos y las estabiliza, erosiona o dilata imágenes, adelgaza o dilata cierta constante, también se puede hacer thresholding, el cual genera una imagen cambiando una imagen en escala de grises a imagen binaria en función de un valor umbral. Véase la Figura 38.

Figura 38 Erosionado de imágenes OpenCV
Fuente: (Oliveira & Conci)

2.6.5 Calibración de cámaras

La calibración de las cámaras es un proceso el cual es muy importante para la obtención de datos correcta, desde la aparición de las cámaras tipo pinole las cuales son muy baratas, se han vuelto una constante en nuestras vidas diarias, ya que las encontramos en todos lados, desde cámaras para circuitos cerrados de televisión (cámaras de seguridad) hasta en sistemas FPV (first person view), desafortunadamente el bajo costo de este tipo de cámaras tiene su precio, estas cámaras tienen una distorsión significativa en la imagen capturada [1-6].

Las imágenes sin calibrar se pueden observar en la Figura 39, se puede notar como los filos rectos se observan curvados en la imagen, por medio de un proceso es posible obtener una relación entre las unidades naturales de las cámaras (pixeles) y los del mundo real (milímetros), con esta relación es posible reconstruir la imagen, como se observa en la Figura 40.

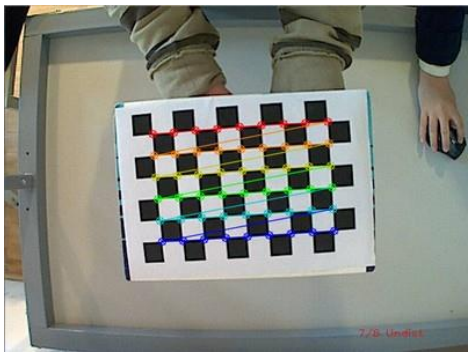


Figura 39 Cámara no calibrada OpenCV

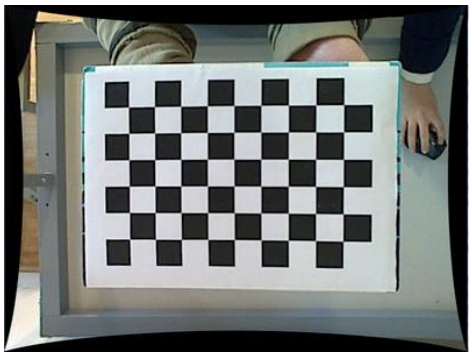


Figura 40 Cámara calibrada OpenCV

2.6.6 Entrada y salida de video

Es posible obtener grabación de video desde diferentes objetos, básicamente existen 2 métodos de grabación, el primero es con imágenes en tiempo real o imágenes pregrabadas, como puede ser una cámara, un archivo multimedia, un dispositivo de transmisión de video. Por suerte con OpenCV es posible trabajar con los estos dos métodos de la misma forma. El resultado de la captura de imágenes desde un archivo por medio de OpenCV se observa en la Figura 41.

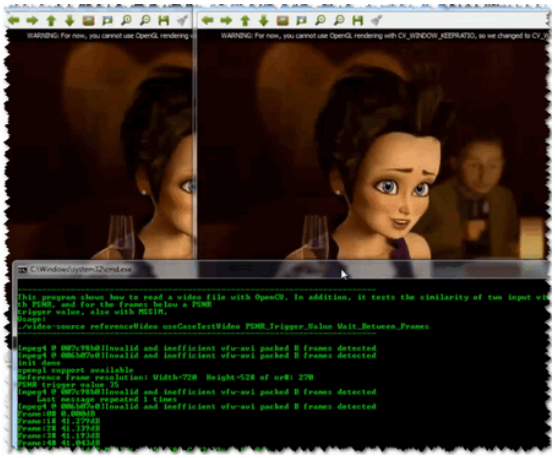


Figura 41 Captura desde un archivo OpenCV
Fuente: (Bueno Garcia, y otros, 2015)

También es posible crear un video después de haber procesado imágenes, esto es muy simple, se lo puede hacer llamando a la clase VideoWriter la cual se encarga de esta tarea, el resultado de generar un video después de un proceso de filtrado se puede observar en la Figura 42.

Figura 42 Video guardado OpenCV
Fuente: (Bueno Garcia, y otros, 2015)

2.6.7 Característica 2D

En visión por computador es muy importante el juntar puntos entre diferentes cuadros en un ambiente, ya que si conocemos como se relacionan las dos imágenes, es posible extraer información de ellas. En el procesamiento de imágenes OpenCV es capaz de determinas ciertas características la cuales son de vital ayuda en los procesos de obtención de datos de imágenes, tres de estas cualidades son la determinación de esquinas en una imagen, también como el detectar características en una imagen, esto es importante para encontrar elementos de una imagen dentro de otra que sería la 3era característica importante de OpenCV esos conceptos se pueden visualizar en la Figura 43, Figura 44 y Figura 45 respectivamente.

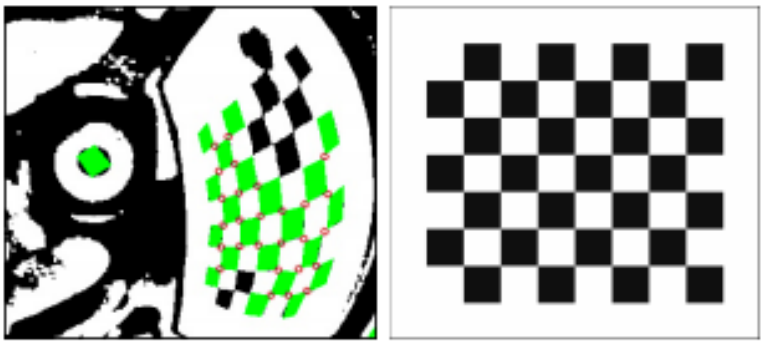


Figura 43 Detección de esquinas OpenCV
Fuente: (Rufli, Scaramuzza, & Siegwart, 2008)

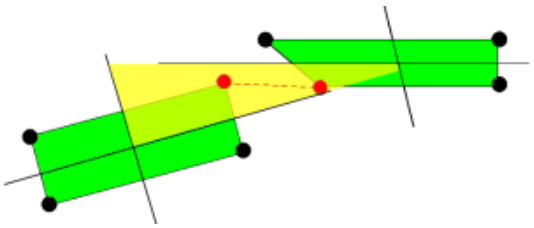


Figura 44 Esquinas obtenidas de imagen OpenCV
Fuente: (Rufli, Scaramuzza, & Siegwart, 2008)

Figura 45 Comparación de características desde 2 imágenes OpenCV
Fuente: (Bueno Garcia, y otros, 2015)

2.6.8 Análisis de video

El análisis de video es importante ya que es posible obtener características en función del análisis de los cuadros anteriores, es decir se pueden extraer características como la determinación de un objeto en movimiento solo con la comparación entre cuadros ya que si algo se mueve y la cámara se encuentra estacionaria, los pixeles no cambian, pero a comparación de en donde se encuentra moviéndose un objeto, con este principio también es posible la extracción del fondo, es decir quitar el fondeo de una imagen con movimiento, este proceso se observa en la Figura 46.

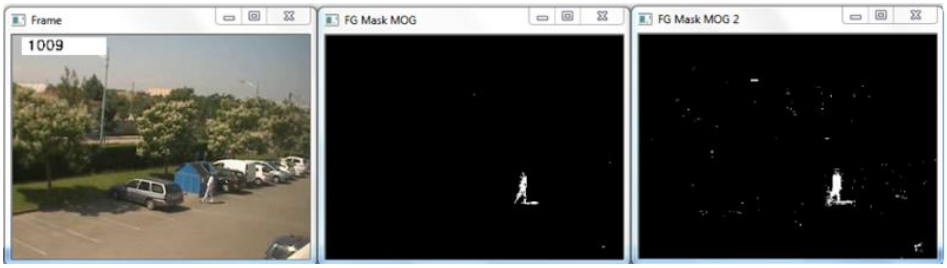


Figura 46 Sustracción de fondo OpenCV
Fuente: (Bueno Garcia, y otros, 2015)

2.6.9 Detección de objetos

Para la detección de objetos el método más usado es el de clasificador en cascada el cual es un tipo de aprendizaje de maquina [7-9], en el cual el sistema necesite imágenes tanto del objeto a identificar como imágenes que no tienen nada que ver con el objeto, esto es necesario para que el sistema aprenda a identificar un elemento de otro, este método si se lo utiliza bien se puede obtener un procesamiento bastante rápido para un sistema en tiempo real, es posible usar estos métodos para la detección de rostros como se muestra en la Figura 47.

Figura 47 Detección de rostros con método de cascada OpenCV
Fuente: (Deva Ramanan)

2.6.10 Unión de imágenes

El método de unión de imágenes es un proceso en el cual es posible generar una sola imagen compuesta por varias más pequeñas, esto se logra al analizar imagen por imagen en búsqueda de puntos de interés después se unen cada una por medio de estos puntos ya encontrados, no todas las imágenes se encuentran tomadas con el mismo ángulo, es decir que en una se veras más de un sector de la imagen que de otra, esto hace que al unir las imágenes se generen espacios negros los cuales esta vacíos de la información de cualquiera de las imágenes, esto se aprecia mejor en la Figura 48.

Figura 48 Unión de imágenes OpenCV
Fuente: (Shao & Malkani, 2008)

2.7. Filtros de discriminación para procesamiento de imágenes

Hoy en día los en los sistemas de visión por computador en tiempo real es necesario tener una rápida respuesta por parte de los serrones y una gran fiabilidad por parte de los algoritmos, esto para que los sistemas que realizan el control en base a los datos enviados por los sensores sean lo más precisos posibles, aquí entran en juego los filtros de discriminación para imágenes los cuales cumplen la tarea de resaltar o eliminar. Ciertas características que son de nuestra importancia, este puede ser un proceso tanto matemático, algorítmico o incluso físico, también puede ser una mezcla de todos estos para aumentar la fiabilidad del resultado final.

Para obtener un análisis de imágenes robusto siempre es recomendable pasar la información por filtros ya que este tipo de datos son sensibles a cambios simples como a la iluminación, la velocidad de los objetos que pasan frente al él o la estabilidad o movimiento de la cámara debido a la naturaleza del sensor y del ambiente en el que se encuentra.

En el proceso de filtrar las imágenes por varios métodos es muy importante tomar en cuenta que el orden en el que se realice este proceso va a inferir en el resultado final, por esta razón es necesario un análisis de la imagen real vs la imagen deseada para realizar una planificación de que tipos de filtros se aplicaran y en qué orden.

El proceso matemático para la discriminación de imágenes se trata de básicamente de la obtención de los datos de la imagen en forma de una matriz, posteriormente en esta se encuentran determinadas características las cuales por medio de operaciones

matriciales o algebraicas es posible eliminar o filtrar los datos que no son necesarios para nosotros, es decir de esta forma podemos detectar contornos, suavizar/eliminar ruidos, cambiar entre espacios de colores o simplificar la imagen.

El suavizado/eliminación de ruido en las imágenes es un proceso que por lo general es el primer paso para la implementación de visión por computador ya que las imágenes que llegan desde los sensores usualmente tienen ruido visualizado como uno o varios píxeles de cierto color dentro de un sitio donde no debería estar, este efecto se visualiza en la Figura 49, eso se visualiza mejor en la figura continuación, como se puede observar el proceso elimina bastante el ruido, pero se pierde nitidez en la imagen, es decir los bordes son menos cortantes. Esto se debe a que el proceso matemático toma muestras de ciertas áreas, saca un promedio del color que debería estar en esa área y después iguala a todos los píxeles dentro de dicho contorno al valor obtenido.

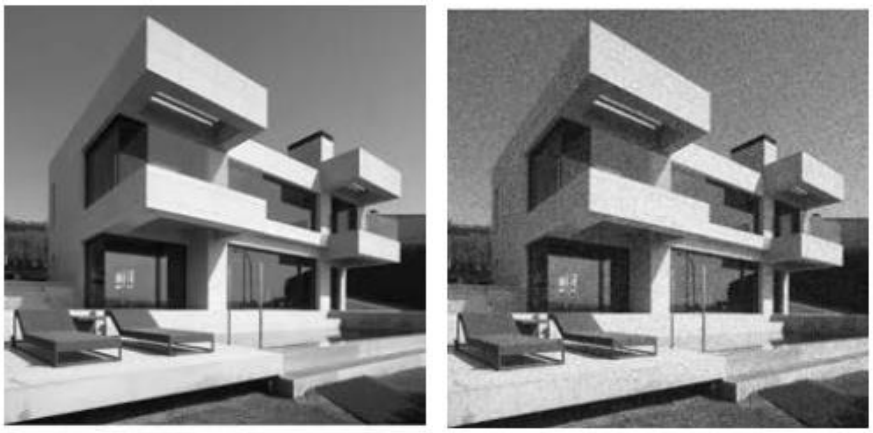


Figura 49 Ruido captado por la cámara OpenCV
Fuente: (Akinlar & Topal, 2012)

Existen varios tipos de espacios de colores, esto quiere decir que la misma imagen puede ser expresada en lo que para nosotros sería otro color, como por ejemplo lo es la imagen normal RGB, también hay la imagen conocida como en blanco y negro que en realidad es una escala de grises lo que observamos, existe otros tipos de espacios de colores como el BGR, también se puede invertir los colores o usar filtros HSV, algunos de estos espacios de colores se muestran en la Figura 50. Básicamente los contornos de las imágenes son las mismas lo que cambia es el color que son varios

tipos de interpretación diferentes lo cual puede ayudar en nuestro proceso para el procesamiento de imágenes.

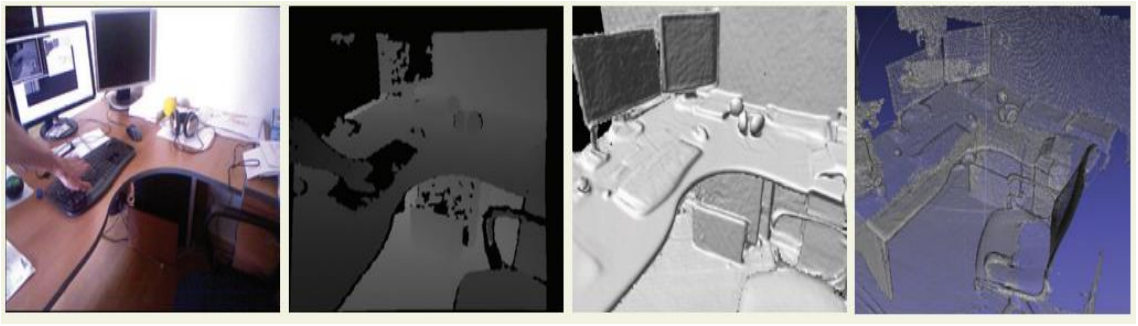


Figura 50 Cambio de espacio de color OpenCV
Fuente: (Pulli, Baksheev, Kornyakov, & Eruhimov, 2012)

Otro filtro es el de detección de contornos el cual básicamente trabaja con 2 imágenes separadas por sus coordenadas tanto en x como en y, cada imagen tiene una bandera o flag levantada donde se encuentra un punto de gradiente especifico, si al comparar las dos imágenes (x e y) las banderas levantadas coinciden quiere decir claramente que es este punto se encuentra un contorno, para la generación de las 2 imágenes a comparar se realiza una discriminación de gradiente, usualmente se sabe que existe un contorno cuando en la imagen existe un cambio de gradiente o color, como se muestra en la Figura 51, mientras mayor sea el gradiente la presencia de un objeto sobre otro es inminente, esto hace que se genere las 2 imágenes a utilizare en el algoritmo y el resultado final es una sola imagen binaria en la cual se observan solo los contornos, esto se observa en la Figura 52.

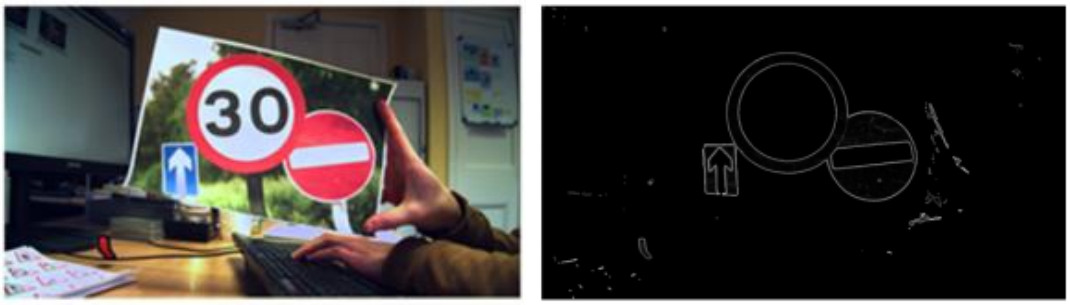


Figura 51 Detección de contornos OpenCV
Fuente: (Russell & Fischaber, 2013)

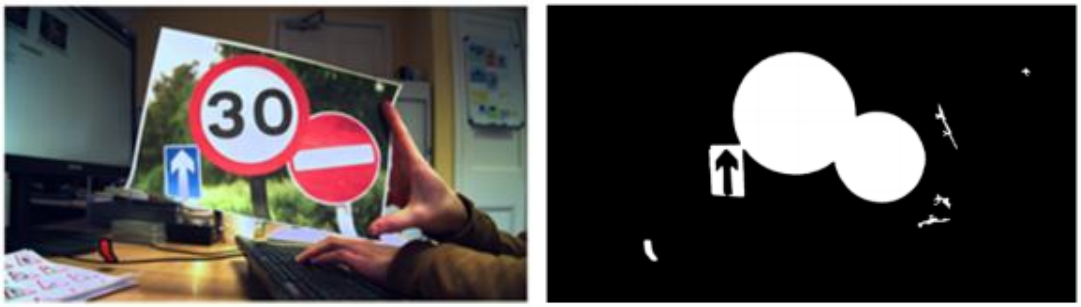


Figura 52 Llenado de contornos OpenCV
Fuente: (Russell & Fischaber, 2013)

También tenemos el filtro threshold el cual simplifica la imagen en otra de tipo binario, esto lo hace en función de una imagen la cual se encuentra en escala de grises, al pasarla por un proceso en la cual se compara cara píxel con un valor llamado umbral es posible determinar cuál píxel será blanco y cual será negro, este proceso se observa en la Figura 53. Es te proceso es importante para la discriminación final antes de un procesamiento de imagen (obtener datos de una imagen) ya que simplifica de manera extrema la imagen, reduciendo así los tiempos de procesamiento o la lógica para la obtención de datos pasos posteriores.

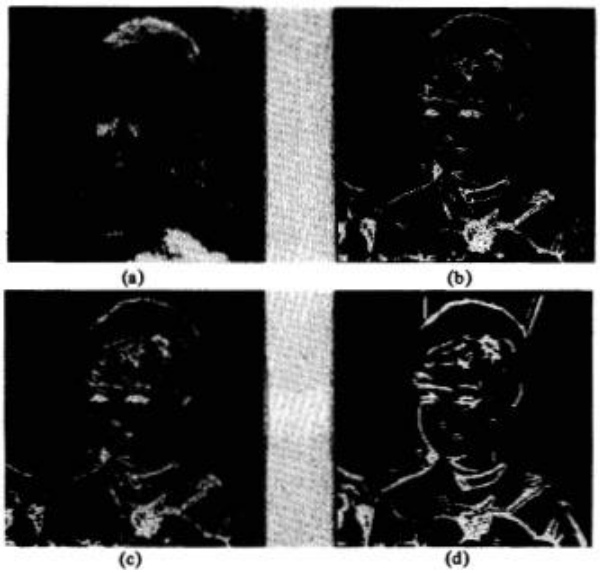


Figura 53 Uso de filtros threshold: (a) Original; (b) mascara 3 x 3 de operador diferencial; (c) mascara 3 x 3 de operador de coincidencia; (d) mascara 7 x 7 de operador de coincidencia
Fuente: (Abdou & Pratt, 1979)

Es importante aclarar que existe varias libreas con estos métodos para la discriminación de imágenes, en las cuales no es necesario pasar por un conjunto de filtros para ejecutar el filtro deseado, sino que ciertos tipos de filtrado realizan un pre-

procesamiento por otros métodos para obtener el resultado final, como lo es el ejemplo de la librería OpenCV en la cual para el método de detección de contornos no es necesario pasarlo por otros filtros antes de llamar a este, sino que con la imagen original es posible obtener la detección de contornos, internamente este llama a otros simultáneamente como el de suavizado o el de threshold a parte de su propio filtro para obtener un resultado final.

Otro tipo de filtrado es el algorítmico, el cual se trata de resaltar elementos en la imagen de las cuales se tiene conocimiento tanto en forma, color o tamaño, como se observa en la Figura 54, este tipo de procesamiento no demanda de muchos recursos informáticos ya que basta con comparar la imagen con atributos preestablecidos. Si sabemos que el objeto a detectar tiene cierto color, solo se tiene que buscar dicha característica en la imagen o si se conoce que en esta debe existir exactamente un numero establecido de elementos basta con contarlos, en este tipo de filtrado a veces se demanda de un pre- procesamiento para que la aplicación de estos no tengo falsos positivos.

Figura 54 Detección de color
Fuente: (Van den Bergh & Van Gool, 2011)

Por ultimo tenemos los filtros físicos que son básicamente elementos como lentes o materiales que se posicionan frente de la cámara para que la imagen discrimine ciertas características, como lo es generalmente el tipo de espectro, se usa varios tipos de filtros como los IR, UV o polarizados los cuales elimina la luz infrarroja, ultravioleta o el resplandor en la lente de la cámara respectivamente, algunos de estos filtros se observan en la Figura 55, haciendo referencia a los filtros utilizados en este

proyecto. Incluso existen filtros que eliminan el color, solo dejan pasar cierta cantidad de luz permitiendo observar simplemente siluetas del objeto como son los lentes de protección para observar eclipses.

Figura 55 Filtros físicos para cámaras

2.8. Algoritmos de control para planeación de trayectoria de vuelo

El campo de la planeación de rutas de vuelo ha sido muy investigado dando así varias formas de llegar a un mismo resultado, pero por maneras diferentes, el objetivo principal es el de llegar desde un punto inicial hasta un punto final navegando a través de los campos libres de colisión, esto se logra gracias al uso de algoritmos RRT (Rapidly-Exploring Random Tree) o exploración aleatoria rápida (LaValle, 1998) [15-18].

El proceso que sigue el algoritmo para determinar la ruta es esencialmente un conjunto de pasos repetitivos de generación de puntos aleatorios desde un punto previo desplazándose una distancia establecida, después se mide la distancia desde el punto de llegada hasta los nuevos puntos generados, se procede a elegir el punto que se encuentre más cerca del objetivo y posteriormente se revisa que el punto seleccionado se encuentre en un espacio libre de colisión que básicamente se refiere a que el punto no se encuentre sobre un obstáculo, de esta forma el árbol crece de a poco, partiendo desde un punto hasta tener varias ramificaciones, el proceso de generación del árbol se aprecia de mejor manera en la Figura 56 (LaValle, 1998).

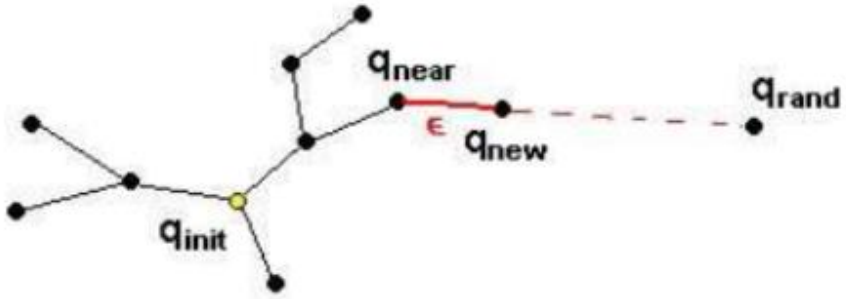


Figura 56 Crecimiento del árbol RRT
Fuente: (Gómez-Bravo, 2006)

Con estas condiciones se puede observar un crecimiento del árbol de una forma muy coherente, es decir, si el árbol se encuentra dentro de un espacio libre de colisiones, este crecerá de forma uniforme sin alguna preferencia de crecimiento como se observa en la Figura 57 a diferencia de si el árbol se encuentra dentro de una zona en la cual se observa que existe áreas de colisión, en la Figura 58 se observa un crecimiento orientado a evadir los obstáculos.

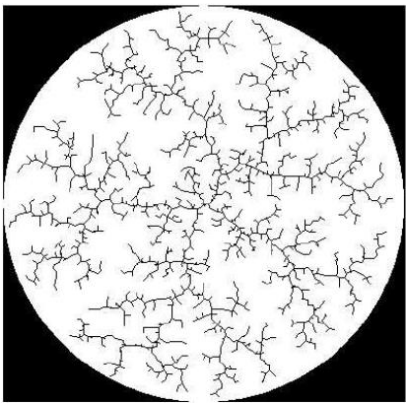


Figura 57 Crecimiento uniforme RRT
Fuente: (Gómez-Bravo, 2006)

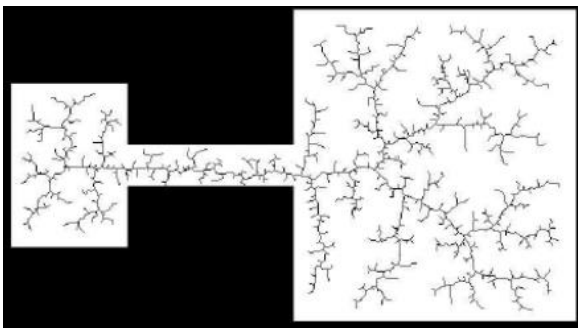


Figura 58 Crecimiento evadiendo obstáculos
Fuente: (Gómez-Bravo, 2006)

A continuación, se presentan varias extensiones y mejoras del algoritmo RRT:

- **Algoritmo RRT bidireccional básico**

En esta extensión del algoritmo RRT básico lo que se hace es iniciar el árbol tanto desde el punto de inicio como desde el punto de llegada, el algoritmo se detiene con una respuesta en el momento en que los dos árboles se cruzan, en este momento se sabe que se ha encontrado una solución al sistema.

En este método existen dos puntos importantes que resaltar, el primero, es que los puntos generados de forma aleatoria son los mismos tanto para el árbol de salida como para el árbol de llegada, eso sirve para ahorrar recursos computacionales. La segunda característica de este método bidireccional es que ninguno de los dos árboles crece si alguno de estos presenta una colisión, esto genera que el crecimiento se uniforme en ambos lados. Es necesario señalar que los dos árboles tienen que coincidir en el mismo punto para que el algoritmo lo tome como respuesta encontrada y termine el cálculo de la ruta, esto se vuelve en una restricción ya que condiciona al sistema de una forma muy específica (Kuffner & LaValle, 2000).

- **Algoritmo RRT Ext-Ext**

Para agilizar la conexión entre los dos árboles descritos en el algoritmo anterior, se creó la variante Ext-Ext, en esta variación cada uno de los arboles tiende su dirección hacia el encuentro de su homologo, esto hace que el algoritmo sea más efectivo a comparación de otros ya que no es puramente aleatorio, de esta forma cada árbol toma la mitad de tiempo de su proceso en explorar espacios libres y la otra mitad en buscar a su homónimo (LaValle & Kuffner, Rapidly-Exploring Random Trees: Progress and Prospects., 2001).

- **Algoritmo RRT Ext-Con**

En este algoritmo a diferencia de los anteriores el proceso cambia ya que se reemplaza el paso de generación de puntos aleatorios por uno de conexión o agregar

puntos consecutivos hasta llegar al objetivo o estar dentro de un área de colisión. Con este algoritmo las trayectorias generadas son más rectilíneas por su naturaleza de adición de segmentos sucesiva, pero el método de conexión toma mayor cantidad de recursos computacionales, esto se justifica ya que con mayor frecuencia las trayectorias generadas son optimizadas (Kuffner & LaValle, 2000).

Este tipo de sistemas para planeación de trayectorias es más eficiente pasa robots no omnidireccionales ya que se adecua más a la cinemática de su sistema, al no haber tantos cambios bruscos de dirección como en los sistemas RRT antes descritos.

- **Post procesado en trayectorias RRT**

Las trayectorias generadas por los algoritmos RRT suelen ser complejas de seguir ya que son generados por aleatoriedad, es decir, puede que llegue al objetivo la ruta, pero no de una forma coherente, realizando de esta manera movimientos innecesarios por parte del vehículo (Laumont, Jacobs, & Murray, 1994).

Para mejorar las trayectorias es necesario pasarlas por un post procesamiento en el cual se eliminan vértices que no son importantes, estos algoritmos son rápidos y sencillos, también tienen un coste computacional bastante bajo, haciendo que el uso de estos procesos sean casi obligatorios para la generación de trayectorias.

Básicamente el algoritmo funciona como se observa en la Figura 59.

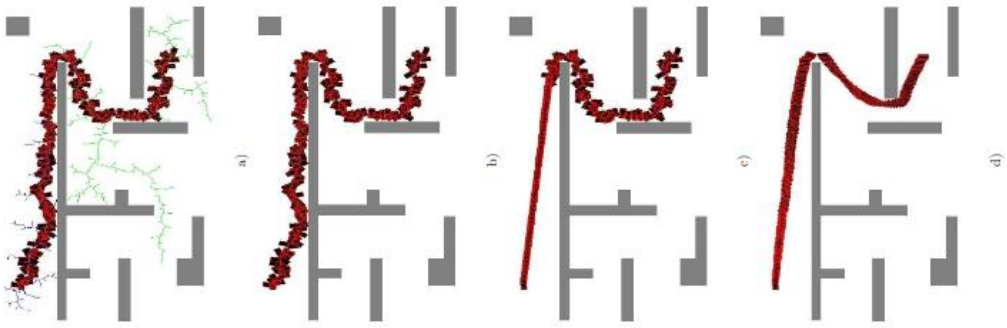


Figura 59 Post procesado de RRT
Fuente: (Gómez-Bravo, 2006)

2.9. Software

2.6.1 Sistema Operativo: Ubuntu 16.04 LTS

Es un sistema operativo basado en GNU/Linux desarrollado por Canonical Ltd. y Fundación Ubuntu distribuido como software libre, manteniendo durante algún tiempo su propio entorno de escritorio llamado Unity (interfaz de usuario creado para un entorno de escritorio GNOME (Entorno de Modelo de Objeto de Red GNU)). Está compuesto básicamente de software de licencia libre o de código abierto, el cual permite a sus usuarios participar en el desarrollado del mismo, probando versiones inestables del sistema, redactando código, solucionando bugs, etc.

Canonical Ltd. lanza cada seis meses una nueva versión de escritorio y servidor, logrando contar con esto aplicaciones recientes con nuevo código abierto. Para este proyecto fue utilizado la versión Ubuntu 16.04 LTS, en donde la numeración indica el año y mes en que fue lanzado, la versión LTS significa que dispone de soporte técnico extendido con actualizaciones de seguridad de paquetes de software, en la Figura 60 se puede observar el entorno de trabajo de Ubuntu 16.04 LTS.

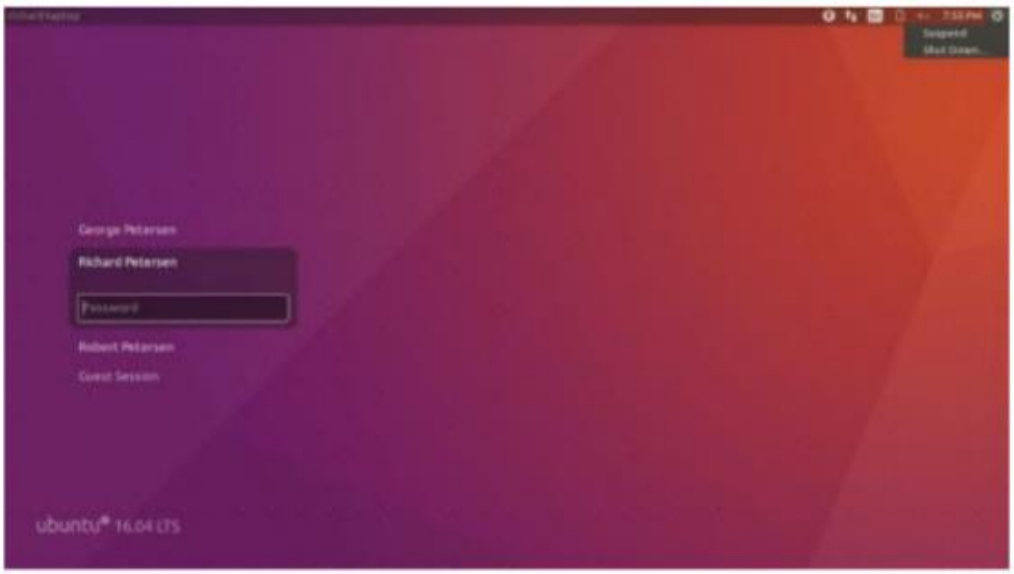


Figura 60 Entorno de trabajo Ubuntu 16.04 LTS
Fuente: (Petersen, 2016)

La razón para utilizar este sistema operativo es debido a que se adapta afinadamente a las necesidades de desarrolladores, además es totalmente compatibles con paquetes y herramientas usadas en la ejecución de este proyecto de investigación. Su logo se muestra en la Figura 61.

Figura 61 Logo Ubuntu 16.04 LTS
Fuente: (Petersen, 2016)

2.6.2 Robot Operating System ROS

2.6.2.1 ¿Qué es ROS?

ROS (Robot Operating System) es una plataforma de desarrollo para sistemas robóticos open source, la cual proporciona una serie de servicios y librerías que simplifican la creación de aplicaciones complejas para robots (Garcia, 2013).

Dispone de una similitud de otras plataformas para desarrollo robótico, pero esta aúna lo mejor de estos sistemas, creando un sistema capaz de comunicarse tanto con robot modernos y varios disponibles ya en el mercado. Desde su creación ha facilitado el intercambio de software entre aficionados y desarrolladores de la robótica, lo que ha permitido la creación de una gran comunidad en todo el mundo.

ROS permite el uso de diferentes lenguajes de programación, de manera oficial soportan lenguajes como Python, C++, Lisp, Lua y otras en fase experimental como Java. Esta plataforma puede ejecutarse en diversos sistemas operativos siendo los más óptimos y utilizados como Ubuntu y Mac OS X, habiendo la posibilidad de ser ejecutado en Fedora, Gentoo, etc. (Garcia, 2013)

2.6.2.2 Objetivos de ROS

El objetivo de ROS se basa en apoyar el código reutilizable en la investigación robótica y el desarrollo, siendo un sistema con una estructura dividida en procesos (conocidas como nodos o clases) permite la ejecución de procesos en forma individual y la utilización de estos de forma flexible en tiempo real, los mismos que se pueden agrupar en paquetes y pilas siendo fácilmente compartidos y distribuidos.

Varios de los objetivos adicionales a los mencionados se enlistan a continuación:

- Modelo de bibliotecas: es un modelo de desarrollo preferido por los usuarios, donde se puede escribir una serie de programas ligadas a interfaces funcionales y limpias que permitan una rápida compresión y modificación.
- Sencillez: ROS está diseñado para ser tan sencillo como sea posible de modo que sus programas puedan ocuparse de manera global con otros marcos de robot de software.
- Independencia de idiomas: puede ser implementado y ejecutado en base a varios lenguajes de programación modernos como Python, C++ y Lisp.
- Escala: este sistema puede ser aplicado para sistemas complejos y grandes, facilitando la ejecución del mismo.
- Modo test: dispone de un modo de prueba del sistema haciendo una orden de llamada ROSTEST (Garcia, 2013).

2.6.2.3 Áreas de aplicación de ROS

Algunas de las aéreas de aplicación de ROS se enlistan a continuación:

- Reconocimiento facial y de objetos.
- Publicación y suscripción de flujo de datos.
- Seguimiento de objetos.
- Percepción de profundidad utilizando un arreglo de cámaras [19-21].
- Planificación y seguimiento de trayectoria.

- Robots móviles (aéreos, terrestres o acuáticos).
- Testeo de sistemas, etc. (Albornoz & Calahorrano, 2016).

2.6.2.4 Funcionamiento del sistema

ROS dispone de tres niveles de conceptos: el nivel de sistemas de archivos, el nivel de computación a nivel gráfico y por último el nivel comunitario. A continuación se resumen los dos primeros niveles, siendo estos los principales (Garcia, 2013).

2.6.2.4.1 Sistemas de archivos

Representan todos los recursos que pueden encontrarse en un programa:

- Paquetes: estos representan la unidad principal en la organización de software de ROS. Este puede contener procesos ejecutables, bibliotecas independientes, archivos de configuración y conjunto de datos.
- Pilas: representan una colección de paquetes que tiene una misma función.
- Manifiestos de pilas: este recurso proporciona información acerca de las pilas, que incluyen la información de licencia y dependencias de otras pilas.
- Mensajes: definen la estructura de datos para los mensajes que serán emitidos en ROS.
- Servicios: definen la solicitud y estructura de datos requeridos por ROS (Garcia, 2013).

2.6.2.4.2 Computación a nivel gráfico

Este nivel se encarga de procesar el conjunto de todos los datos. Refiriéndose a conceptos básicos como nodos, maestro, mensajes y temas, los cuales proporcionan información de diferente índole.

- Nodos: estos representan procesos a ser ejecutados de forma simultánea llevando a cabo cálculos. Por ejemplo en el sistema de control de un robot usualmente

necesitan varios nodos, siendo un nodo para el control de las ruedas, un nodo que realice la localización, un nodo que ejecute la planificación de trayectoria y así sucesivamente.

- Maestro: el maestro proporciona el registro de nombres y la búsqueda de estos. Sin el Maestro, los nodos no podrían comunicarse entre sí para el envío y recepción de información.
- Mensajes: representan una estructura de datos, pudiendo incluir estructuras de formas anidadas y matrices. Los nodos se comunican enviando y recibiendo mensajes.
- Temas o tópicos: el tema representa el nombre para identificar el contenido del mensaje. Los mensajes se enrutan a través de un sistema de comunicación de publicación / suscripción semántica. En forma de ejemplo un nodo publica un mensaje en un determinado tema y a su vez otro nodo se suscribe a ese tema para recibir el mensaje (Garcia, 2013).

En la Figura 62 se presenta el nivel de computación gráfica de ROS

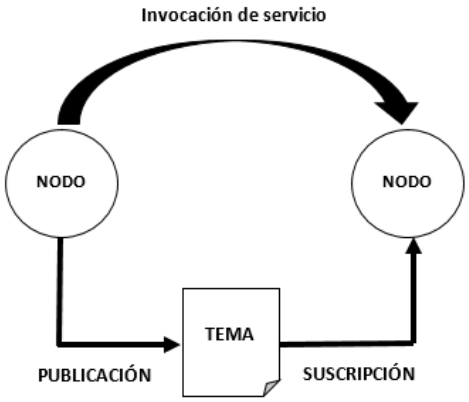


Figura 62 Nivel de computación gráfica de ROS

2.6.2.5 Virtualización de ROS: RVIZ

ROS dispone de una herramienta de visualización en 3D llamada RVIZ, que posibilita que cualquier sistema robótico pueda ser representado en una imagen en 3D, respondiendo en tiempo real a lo que ocurre en el mundo real. Para este proyecto se utilizó esta herramienta para la visualización de los obstáculos digitales y el

cuatrirotor, de modo que pueda observarse en tiempo real el movimiento de la aeronave por medio de su espacio de trabajo.

Para generar cada modelo de robot se debe editar un archivo XML y redactar en URDF (Unified Robot Description Format), en donde se especificaran las dimensiones del robot, movimientos de sus articulaciones, inserción de parámetros físicos como masa e inercia, etc. (Garcia, 2013). En la Figura 63 se puede observar el entorno de trabajo de RVIZ.

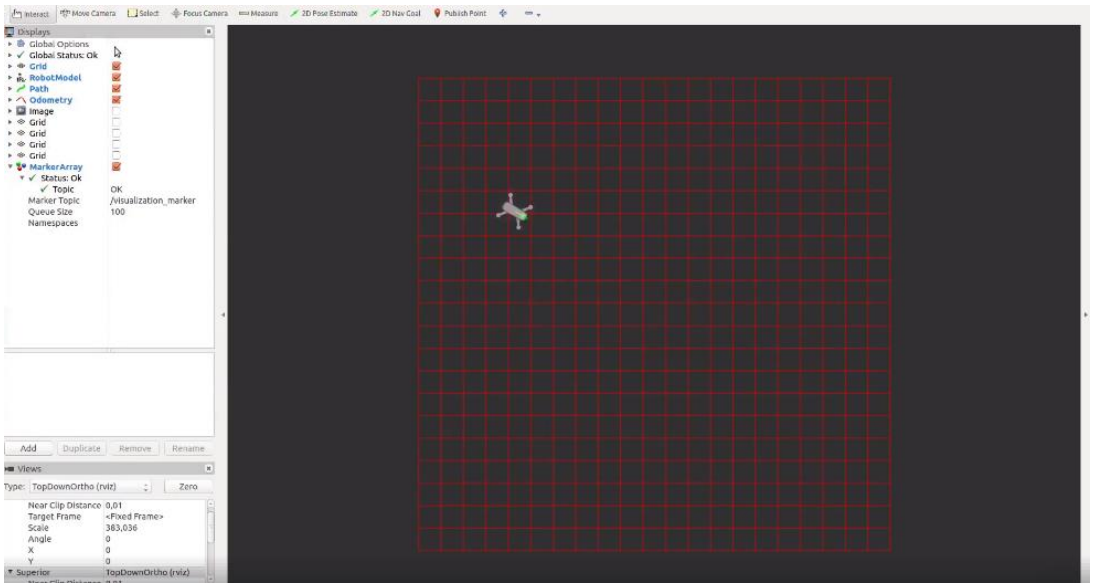


Figura 63 Entorno de trabajo de RVIZ

2.6.2.6 Estructura de ROS

Se va a presentar un resumen de la estructura de ROS como sistema operativo, desde lo más global (repositorio) a lo más específico (nodo).

2.6.2.6.1 Repositorio

Representan ficheros compuestos por uno o más paquetes y pilas, que permiten descargar los ficheros necesarios de forma rápida y cómoda (Garcia, 2013).

2.6.2.6.2 Pila (Stack)

Representan una colección de paquetes con una misma funcionalidad, es el equivalente a una librería en otros sistemas de programación, dentro de los cuales se puede encontrar ficheros con metadatos para que el sistema funcione de forma correcta (Garcia, 2013).

2.6.2.6.3 Paquete

Representa la unidad principal de la organización de ROS, esta contiene lo necesario para ser funcional y equivale a un paquete en C (Garcia, 2013).

2.6.2.6.4 Nodo

Representan los procesos ejecutables que están incluidos dentro de los paquetes, regularmente se utilizan varios nodos dentro de un mismo programa (Garcia, 2013).

En la Figura 64 se observa una imagen esquemática de jerarquía de la plataforma ROS.

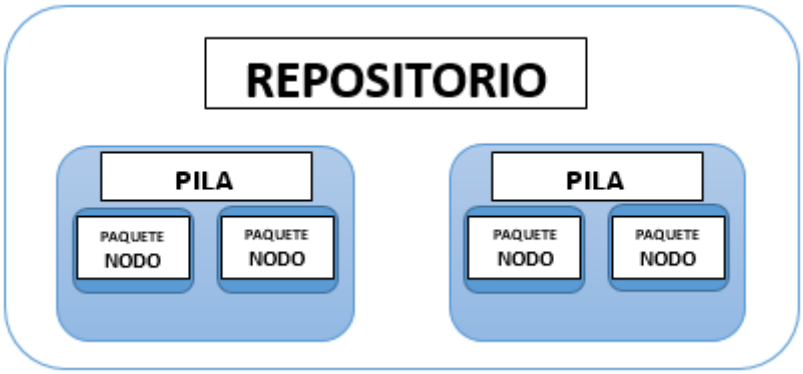


Figura 64 Jerarquización de ROS

2.6.3 Lenguaje de programación: Python

Python es un lenguaje de programación diseñado por Guido Van Rossum, el cual se encuentra en un proceso de continuo desarrollo por una gran comunidad de usuarios. Representa un lenguaje de programación totalmente libre que trabaja

prácticamente sobre cualquier plataforma por ejemplo sistemas PC bajo Linux, sistemas PC bajo Microsoft Windows, sistemas Macintosh de Apple, etc.

Este lenguaje de programación presenta una serie de ventajas que lo hacen más atractivo, tanto para un uso profesional como para un tema de aprendizaje de programación. Entre las ventajas más interesantes se enlistan las siguientes:

- Python es considerado por muchos un lenguaje de alto nivel, debido a que sus programas son muy compactos y cortos que su equivalente en lenguajes como C.
- Es un lenguaje de programación muy legible, su sintaxis es muy elegante y permite una escritura de programas cuya comprensión resulta más fácil que a comparación de otros lenguajes.
- Ofrece un entorno interactivo que facilita la realización de ensayos.
- Su entorno detecta muchos errores de programación, proporcionando con esto una información muy detallada para detectar estos errores y corregirlos.
- Puede usarse como un lenguaje de programación imperativo procedimental o como un lenguaje de programación orientado a objetos.
- Dispone de varios juegos de estructuras de datos que se pueden manipular para facilitar la programación (Marzal & Gracia, 2003).

Para este proyecto se utilizó este lenguaje de programación en su versión 3.0 en conjunto a la plataforma de ROS para la creación de todos los programas. En la Figura 65 se observa el logo de Python 3.0.

Figura 65 Logo Python 3.0
Fuente: (Seguridad Pent, 2016)

2.10. Controlador PID

Un controlador PID hace referencia a un mecanismo de control ampliamente usado en varios sistemas de control industrial con realimentación de bucle cerrado. El PID viene determinado por tres parámetros de control: el proporcional (P), el integral (I) y el derivativo (D), donde dependiendo de la modalidad del controlador cualquiera de estos valores puede ser cero (Ogata, 2010).

El controlador PID se alimenta de un error que es calculado a partir de la salida deseada menos la salida lograda. El controlador intenta minimizar el error ajustando la entrada al sistema.

Las tres acciones de control, interviene en mayor medida sobre alguna característica en la salida, como por ejemplo: el error en estado estacionario, el sobre impulso, el tiempo de estabilización, etc.). De igual forma influye en menor medida sobre otras características del sistema. Debido a esto, es imposible sintonizar un controlador que actué de forma perfecta (tiempo de estabilización en cero, error en estado estacionario igual a cero, etc.) (Ogata, 2010).

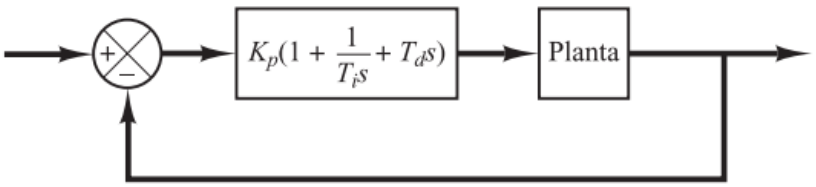


Figura 66 Controlador PID de una planta
Fuente: (Ogata, 2010)

La Figura 66 muestra un controlador PID de una planta. Determinando el modelo matemático de la planta a controlar, es posible determinar los parámetros del controlador aplicando diversas técnicas de diseño con el fin de que cumpla los requerimientos del estado estable y transitorio del sistema en lazo cerrado. A su vez si el modelamiento matemático de una planta resulta complejo determinar, tampoco será posible utilizar un método analítico para diseñar un controlador PID. Para este

caso se debe acudir a procedimientos experimentales para la sintonía de controladores PID (Ogata, 2010).

- **Acción de control proporcional (P):** el controlador responde de manera proporcional a la señal del error.

$$u(t) = Kp * e(t) \quad (2)$$

Descrita desde su función de transferencia se tiene la siguiente ecuación:

$$Cp(s) = Kp \quad (3)$$

Donde Kp es la constante de proporcionalidad ajustable.

Este controlador puede controlar cualquier planta estable, pero posee un desempeño limitado y error en estado estable (off-set). Mientras mayor sea el error mayor será la señal de control, sin embargo el aumentar en demasía la señal de control puede ocasionar que exista sobre impulso y que el sistema se vuelva oscilatorio. Véase la Figura 67.

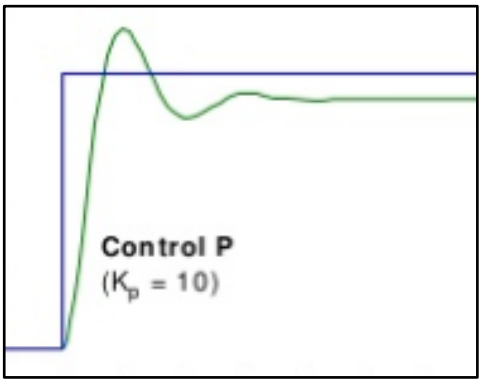


Figura 67 Controlador proporcional P.

- **Acción de control derivativa (D):** el controlador entrega una salida del controlador proporcional a la derivada del error.

$$u(t) = Kd * \frac{de(t)}{dt} \quad (4)$$

Descrita desde su función de transferencia se tiene la siguiente ecuación:

$$Cd(s) = sKd \quad (5)$$

Este controlador permite reducir el sobre impulso y atenuar la oscilación del sistema haciendo más rápida la acción de control.

- **Acción de control integral (I):** el controlador entrega una salida del controlador proporcional al error acumulado.

$$u(t) = Ki * \int_0^t e(\tau)d(\tau) \quad (6)$$

Descrita desde su función de transferencia se tiene la siguiente ecuación:

$$Ci(s) = \frac{Ki}{s} \quad (7)$$

Este controlador permite eliminar el error en estado estable aumentado el tiempo de establecimiento y pudiendo originar mayores oscilaciones al sistema. Representa una acción de control más lenta.

- **Acción de control proporcional – derivativa (PD):** el controlador PD obtiene que la salida del sistema se posicione bajo o sobre el valor deseado de salida.

$$u(t) = Kp * e(t) + Kp * Td * \frac{de(t)}{dt} \quad (8)$$

Donde Td representa a una constante de tiempo derivativo el cual ajusta la acción derivativa. Descrita desde su función de transferencia se tiene la siguiente ecuación:

$$C_{PD}(s) = Kp + sKpTd = Kp(1 + sTd) \quad (9)$$

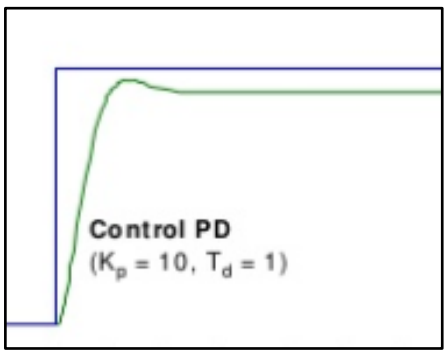


Figura 68 Controlador Proporcional Derivativo PD.

En la Figura 68 se observa la acción de un controlador PD sobre un sistema.

- **Acción de control proporcional – integral (PI):** el controlador PI logra que el error en estado estable será siempre cero.

$$u(t) = Kp * e(t) + \frac{Kp}{Ti} * \int_0^t e(\tau)d(\tau) \quad (10)$$

Donde Ti representa a una constante de tiempo integral el cual ajusta la acción integral. Descrita desde su función de transferencia se tiene la siguiente ecuación:

$$C_{PI}(s) = Kp\left(1 + \frac{1}{Tis}\right) \quad (11)$$

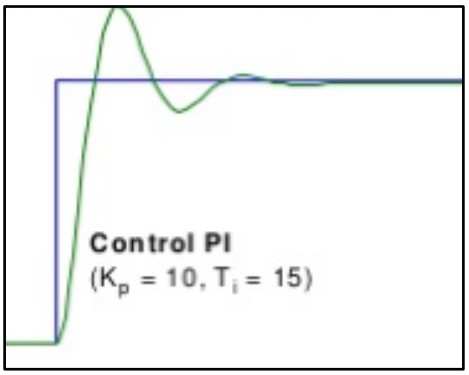


Figura 69 Controlador Proporcional Integral PI.

En la Figura 69 se observa la acción de un controlador PI sobre un sistema.

- **Acción de control proporcional – integral – derivativa (PID):** el controlador PID combina las ventajas de cada una de las tres acciones de control individuales.

$$u(t) = Kp * e(t) + \frac{Kp}{Ti} * \int_0^t e(\tau)d(\tau) + Kp * Td * \frac{de(t)}{dt} \quad (12)$$

Donde Td y Ti representan las constantes de tiempo derivativo e integral respectivamente. Descrita desde su función de transferencia se tiene la siguiente ecuación:

$$C_{PID}(s) = Kp\left(1 + \frac{1}{Tis} + Tds\right) \quad (13)$$

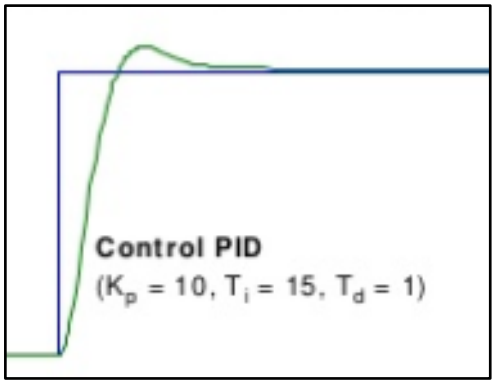


Figura 70 Controlador Proporcional Integral Derivativo PID.

En la Figura 70 se observa la acción de un controlador PID sobre un sistema.

CAPÍTULO III

DISEÑO DE BASTIDOR PARA ARREGLO DE CÁMARAS

3.1 Diseño CAD

Mediante un software de diseño CAD se bosquejo los tres principales elementos que forman parte del sistema de visión por computador, los cuales se presentan a continuación. Los planos se encuentran detallados en el Anexo D.

3.1.1 Bastidor

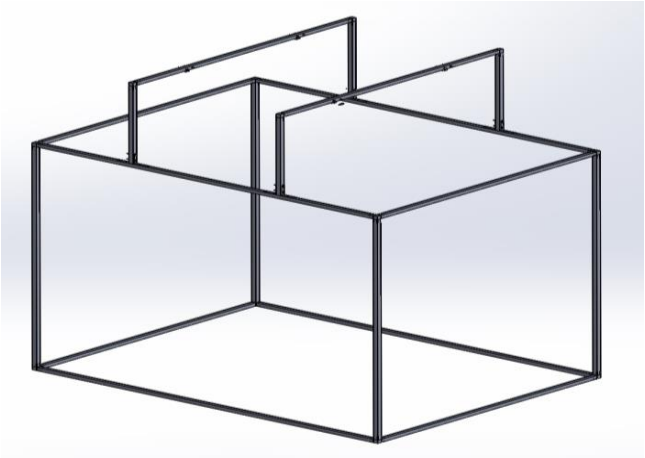


Figura 71 Diseño en Software CAD Bastidor

3.1.2 Cámaras

Figura 72 Diseño en Software CAD cámara PlayStation 3 Eye

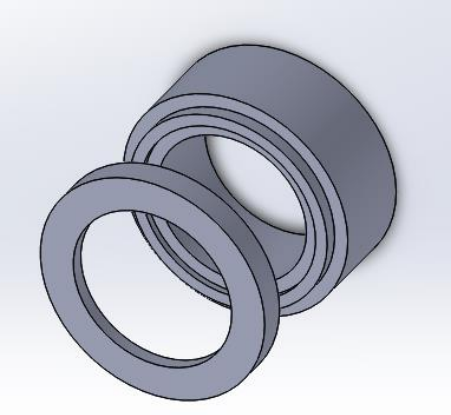


Figura 73 Diseño Software CAD soporte filtro físico para cámara PlayStation 3 Eye

3.2 Selección de materiales

3.2.1 Bastidor

A la hora de seleccionar el material para la construcción del bastidor para el montaje del arreglo de cámaras se tomaron en cuenta parámetros de materiales como son: la disponibilidad en el mercado nacional, la capacidad de ser mecanizado y el costo. De modo que se obtenga una estructura rígida que soporte el sistema de cámaras y las redes de protección.

Los materiales analizados se presentan en la Tabla 4.

Tabla 4
Materiales analizados

Material: Tubería cuadrada	Módulo de Rigidez [G]		Denominación	Espesor	Disponibilidad	Peso	Costo
	Mpsi	GPa	Pulgadas	Milímetros	Cualitativa	Kg/m	$/6m
Aluminio	3.80	26.2	1	1.10	Alta	0.27	$14
Acero al carbono	11.5	79.3	1	1.10	Alta	0.84	$6
Acero Inoxidable 304	10.6	73.1	1	1.20	Baja	0.88	$34

Fuente: (Budynas & Nisbett, 2008)

El aluminio presenta un menor módulo de rigidez por lo cual es el material adecuado, puesto que su maquinabilidad y disponibilidad es alta, además el peso y costo a comparación de los demás materiales es bajo permitiendo así mayor accesibilidad y maniobrabilidad respectivamente.

3.2.2 Cámaras

Para la selección de cámaras se utilizó criterios de selección en base a parámetros como ángulo de visión, resolución, interfaz de comunicación, fotogramas por segundo (FPS), disponibilidad en el mercado y costo. De tal manera que pueda seleccionarse la mejor opción de cámaras para este proyecto en base a sus requerimientos.

Las cámaras analizadas se presentan en la Tabla 5.

Tabla 5
Cámaras analizadas

Cámara	Angulo de visión	Fotogramas por segundo	Resolución	Interfaz de comunicación	Costo	Disponibilidad
	Grados	FPS	Pixeles	Cualitativa	$/unidad	Cualitativa
Cámara matricial JAI A-200 CL	120°	68 fps	1600x1200	CameraLink Base	_ _ _	Baja
PlayStation 3 Eye	75°	187 fps 60 fps	320x240 640x480	USB 2.0	$20	Alta
Logitech HD WebCam C270	60°	30 fps	640x480	USB 2.0	$599	Media

La cámara PlayStation 3 Eye dispone de un mayor ángulo de visión y cantidad de fotogramas por segundo a comparación de las demás cámaras acoplándose adecuadamente a la aplicación de este proyecto, además su interfaz de comunicación USB 2.0 es ampliamente usado en la actualidad permitiendo un manejo y control simple del dispositivo. Por tal razón se selecciona la cámara PlayStation 3 Eye en un arreglo de 4 cámaras implementado un filtro físico (ver Figura 55) en cada una de ellas, conectadas por dos Hub USB 2.0 (uno cada dos cámaras) para evitar interferencias de señal. En la Figura 74 y Figura 75 se observa la cámara PlayStation 3 Eye y un Hub USB 2.0 respectivamente.

Figura 74 Cámara PlayStation 3 Eye

Figura 75 Hub USB 2.0

CAPÍTULO IV

DISEÑO Y SIMULACIÓN DE VEHÍCULO AÉREO NO TRIPULADO

4.1 Diseño CAD

4.1.1 Cuatrirotor de ala rotativa

Utilizando software CAD se realizó el diseño de la aeronave, incorporando dimensiones reales y material en base a este se realizara la implementación del cuatrirotor. Véase Figura 76.

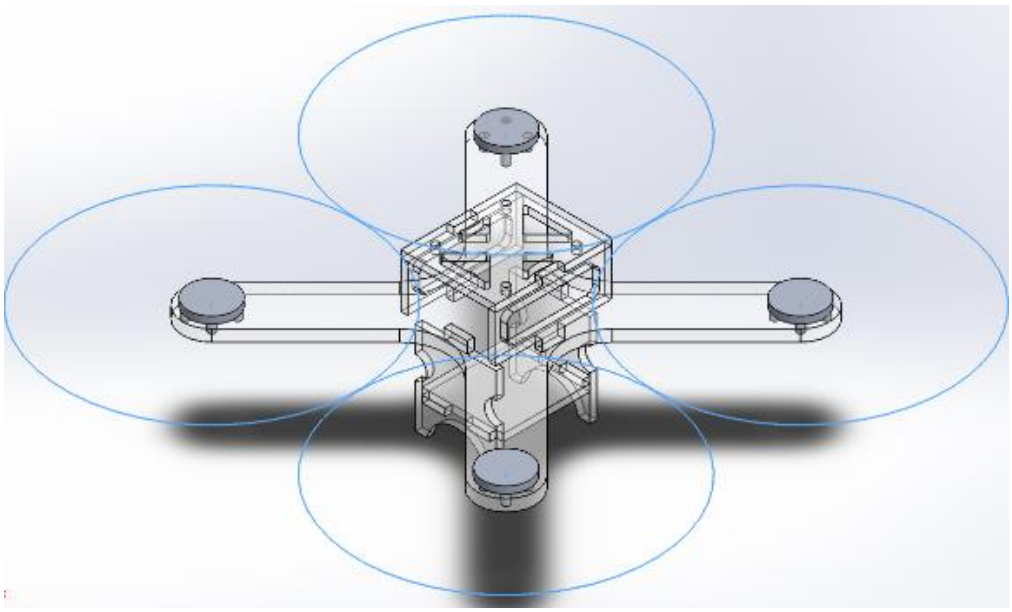


Figura 76 Cuatrirotor de ala rotativa

4.1.2 Base para interfaz de comunicación PC – aeronave

La Figura 77 representa el diseño de la base donde se montara el circuito de comunicación PC – aeronave, en este se colocara un microcontrolador contenido en una placa Arduino Nano, un módulo de comunicación por radio frecuencia y adicionalmente se incluirá un botón de emergencia que permitirá desconectar el cuatrirotor de todo el sistema ante posibles acontecimientos imprevistos.

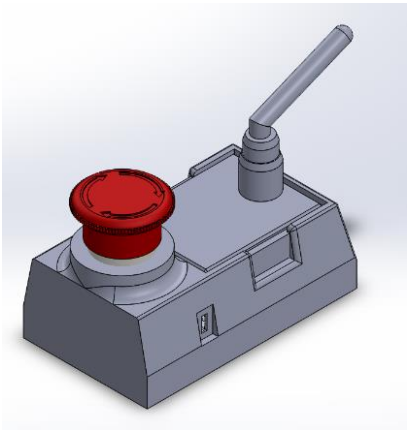


Figura 77 Base para interfaz de comunicación PC – aeronave

4.2 Simulación de vehículo aéreo no tripulado

Mediante el uso de software CAD, se realizó el estudio de análisis estático en el vehículo aéreo de modo que pueda observarse su comportamiento en reposo con cargas como su peso. Este análisis incluye estudio de esfuerzo de Von Mises y factor de seguridad en la estructura de la aeronave.

4.2.1 Simulación cuatrirotor de ala rotativa

Para el análisis estático realizado en la aeronave se aplicó las cargas que se muestran en la Figura 79. Estas cargas se basan en una medición experimental de peso generado al momento de arrancar un motor a su máxima potencia, presentadas en la Figura 78.

Figura 78 Medición experimental de peso generado al arrancar un motor a su máxima potencia

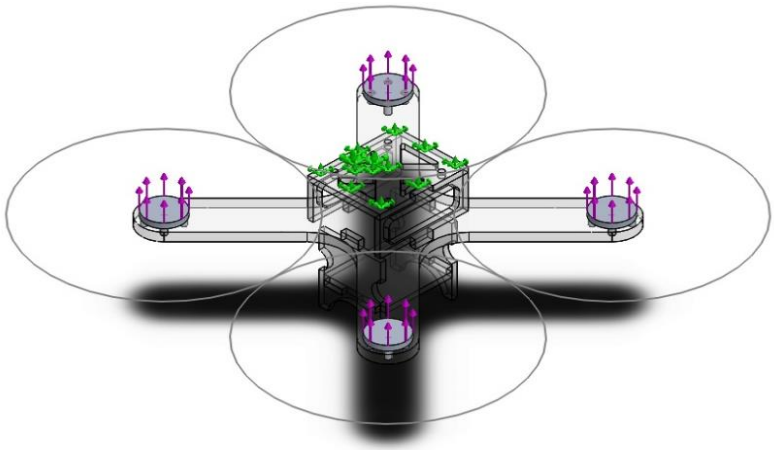
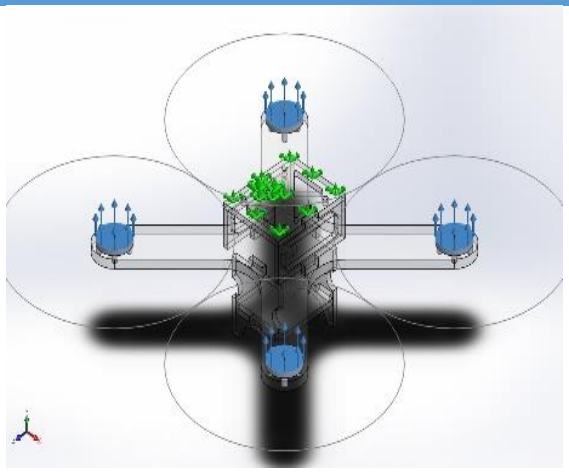


Figura 79 Distribución de cargas sobre el elemento para análisis estático

En la Tabla 6 se enlistan los valores de cargas aplicada sobre el elemento al realizar el estudio.

Tabla 6

Detalle de cargas aplicadas al elemento para análisis estático

Carga	Ilustración	Carga aplicada
Fuerza		Tipo: fuerza normal a las 4 caras Valor: 2.21 N

En base a lo presentado se realizó un mallado en el elemento y posteriormente se obtuvieron los siguientes resultados:

Tabla 7
Resultados de estudio de Esfuerzo de Von Mises sobre cuatrirotor

Tipo de estudio	Mínimo	Máximo
Esfuerzo de Von Mises	4.44199 N/m2	2.94654e+006 N/m2

De acuerdo con los datos mostrados en la Tabla 7 se tiene un valor máximo de 29.47 (MN/m2) con el que se muestran las zonas críticas donde se tiene un mayor esfuerzo al momento de arrancar los motores con el mayor empuje posible para lograr elevar la aeronave. La zona donde se tiene mayor cantidad de esfuerzo es en la saliente de los brazos del armazón, véase Figura 80.

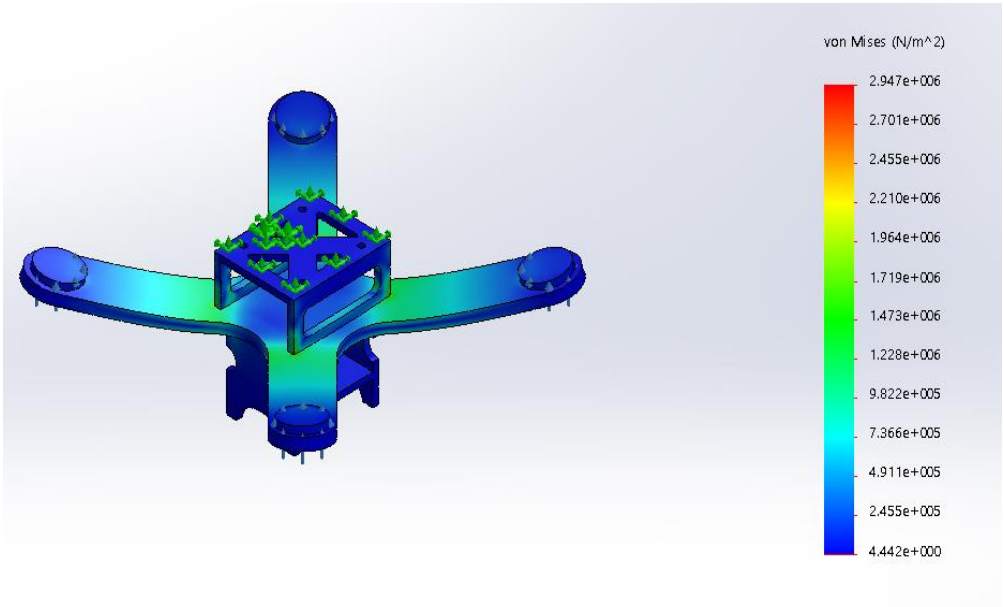


Figura 80 Análisis de esfuerzo de Von Mises cuatrirotor

Así mismo en análisis de factor de seguridad en el elemento nos demuestran que se realizó un correcto análisis en su diseño, evitando de este modo que se produzcan fallas al momento de arrancar el sistema, véase Figura 81.

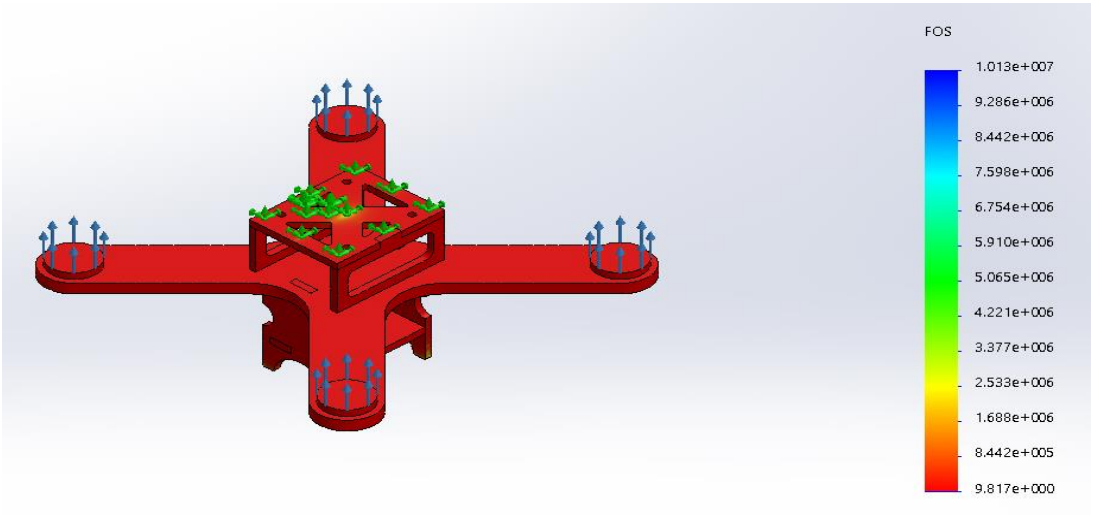


Figura 81 Análisis de factor de seguridad cuatrirotor

Con la siguiente ecuación se puede corroborar el análisis de factor de seguridad:

$$n = \frac{S_y}{\sigma_e} \qquad (13)$$

En donde:

S_y: $limite\ de\ fluencia$
σ_e: $esfuerzo\ máximo\ (von\ Mises)$

De acuerdo a esto se tiene:

$$n = \frac{55{,}57\times10^6}{2{,}947\times10^6} = 18{,}9$$

El valor obtenido es de 18.9 de factor de seguridad con lo que se comprueba que el diseño realizado resistirá a las cargas aplicadas al momento de la implementación.

4.3 Selección de materiales del vehículo aéreo no tripulado

Se seleccionó el cuatrirotor de ala rotativa II debido a que se observa mejores prestaciones, tanto en tamaño como en el análisis estático realizado en numerales anteriores. En base a esta aeronave se seleccionaron los siguientes materiales y elementos para la implementación posterior del cuatrirotor.

4.3.1 Frame o armazón

El mayor inconveniente que se encuentra al momento de construir cualquier proyecto es el coste económico que incorpora. Por tal razón en este proyecto se ha pretendido recortar costes de desarrollo de dos maneras: mecanizando las piezas a ser utilizadas con contratación de servicio de tecnología económica como lo es el corte laser sobre un material de acrílico de 6 mm; e intentando obtener materiales económicos para la implementación del prototipo. El material y espesor utilizado para el armazón está diseñado para soportar esfuerzos a los que pueda verse sometidos la aeronave.

En la Figura 82 se observa las piezas de corte con láser.

Figura 82 Piezas de corte con laser

Los planos de cada pieza se encuentran en el Anexo E.

4.3.2 Motores

Los motores representan los actuadores del sistema cuatrirotor los cuales pueden modificar su estado y generar movimiento a la aeronave. En base al análisis estático realizado anteriormente se determinó que el conjunto motor – hélice debe lograr sustentar un peso total de 366 gramos detallados en la Tabla 8. Por tal razón en base a la Tabla 9 se seleccionó el motor tipo Brushless Turnigy 1806-2280KV. Véase Figura 83.

Tabla 8

Detalle de peso en la aeronave

Elemento	Peso [gramos]
Armazón	109 gramos
Batería 3 celdas (peso promedio)	117 gramos
Motores x 4	76.8 gramos
Controladores x 4	3.2 gramos
TOTAL	306 gramos

Figura 83 Motor Brushless Turnigy 1806-2280KV

Tabla 9

Especificaciones técnicas motor Brushless Turnigy 1806-2280KV

Tamaño	1806-2280KV
Peso unitario	19.2 gramos
Voltaje	2~3 celdas
Máxima corriente [A]	10 A Máx.
Corriente Burst	15 A (10 segundos)
Potencia [W]	117.7
Eficiencia [G/W]	3.5
Velocidad [RPM]	15910
Diámetro de eje	5 milímetros
Rotación	2 CW 2 CCW
Empuje máximo (4 motores)	460 gramos

Este tipo de motor genera un empuje máximo de 115 gramos, un conjunto de cuatro motores generan 460 gramos de empuje máximo abasteciendo completamente el peso total de la aeronave con una diferencia de 154 gramos, pudiéndose implementar una sistema adicional a futuro al cuatrirotor.

4.3.3 Hélices

Las hélices dependen de dos parámetros que son forma y material los que determinan que fuerza de empuje y resistencia al avance generan. Para este proyecto se discriminara el análisis de mecánica de fluidos y estudio de flujo entorno a cuerpos para la selección de hélices, así que fundamentalmente se seleccionaran en base a la elección de los motores.

Para cuatrirotores existe una gran variedad de hélices desde 5x3 pulgadas a 9x4.5 pulgadas y en general cuanto mayor es el aparato a volar, mayores serán las hélices a usar.

Se seleccionaron las hélices especificadas en la Tabla 10 y Figura 84.

Tabla 10
Especificaciones hélices seleccionadas

Material	Plástico
Dimensión	5 pulgadas x 4 pulgadas (pitch) x 3 aspas

Figura 84 Hélices seleccionadas

4.3.4 Microcontrolador

El Microcontrolador actúa como elemento de comunicación entre los distintos elementos de la aeronave, permitiendo la interactuación con ellas.

Un microcontrolador para este tipo de aplicación debe disponer de:

- Salidas PWM necesarias para los cuatro ESC y posterior accionamiento de los motores
- Protocolo SPI (Serial Peripheral Interface), para permitir la comunicación con el módulo de radio frecuencia.
- Protocolo I2C (Inter Integrated Circuit), para permitir la recepción de información desde la unidad de medición inercial (IMU) acerca del estado de orientación de la aeronave.
- Gran variedad de repositorio y librerías; una interfaz de programación sencilla.
- Bajo consumo y compatibilidad con el elemento de alimentación.

Para la selección del microcontrolador a continuación se describen algunas opciones para su posterior elección.

- **OpenPilot CC3D Flight Controller**

Es un controlador de vuelo desarrollado por OpenPilot, este proporciona una sensación de nitidez al volar tricopteros, quads, hexas, octocopters e incluso aviones de ala fija.

Dispone un potente microcontrolador STM32 de 32 bits funcionando a 90 MIP (Million of Instructions Per Second) con una memoria flash de 128 KB y 20 KB de RAM. La placa en la que está contenido el microcontrolador dispone de una unidad de medición inercial (IMU) con giroscopio y acelerómetro de alto rendimiento de 3 ejes. Dispone de 6 canales PWM y PPM combinado; y voltaje de alimentación de 5 V.

Soporta software de configuración en Windows, Mac y Linux. Interfaz de comunicación microUSB directo de alta velocidad. Dispone una memoria EEPROM integrada de 4 Mbits para almacenamiento de configuración. Ofrece conectividad I2C mediante su puerto Flexi-port. Véase Figura 85.

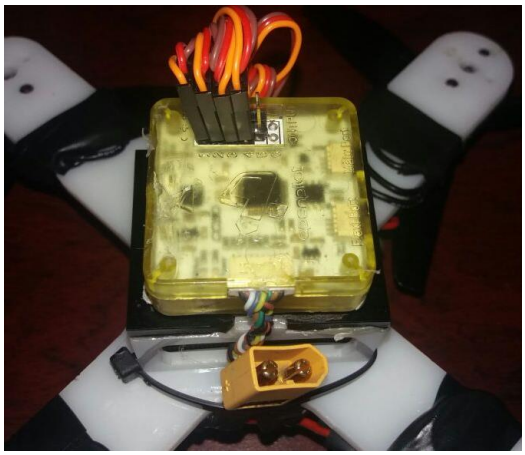


Figura 85 OpenPilot CC3D Flight Controlller

- **Arduino Nano**

El Arduino Nano es una placa programable pequeña, completa y compatible con un microcontrolador ATmega328. Dispone la misma funcionalidad del Arduino Duemilanove. (Arduino, 2017)

Su tensión de funcionamiento es de 5 V. Dispone de una memoria flash de 32 KB de los cuales 2 KB utiliza su gestor de arranque, una memoria SRAM de 2 KB y una memoria EEPROM de 1 KB. Soporta software de configuración Windows, Mac y Linux. Ofrece conectividad I2C y SPI Velocidad de reloj de 16 MHz y 6 canales PWM e interfaz de comunicación microUSB. Véase Figura 86.

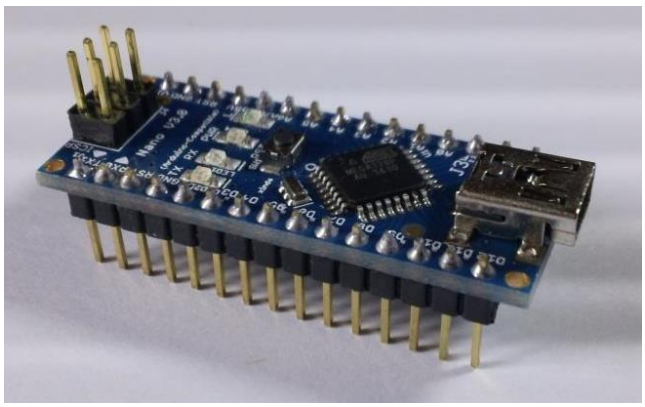


Figura 86 Arduino Nano

- **FriendlyARM NanoPi Neo Plus 2**

NanoPi Neo Plus 2 es una placa programable creada por FriendlyARM, dispone un procesador Allwinner H5 de 64 bits, una memoria RAM de 512 MB y memoria flash de 8 GB e interfaz de comunicación microUSB. Frecuencia de funcionamiento de 1.5 GHz. Voltaje de alimentación de 5 V a 2 A. Ofrece conectividad I2C y SPI. Presenta 1 canal PWM e interfaz de comunicación microUSB. Véase Figura 87.

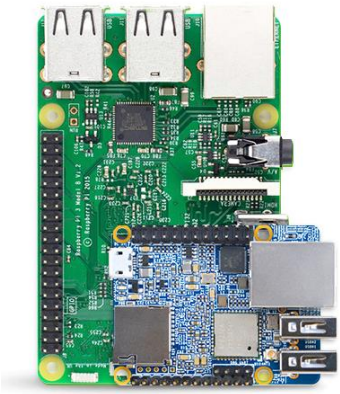


Figura 87 NanoPi Neo Plus 2
Fuente: (FriendlyElec, 2017)

En la Tabla 11 se puede observar una comparación de cada uno de las placas antes descritas de acuerdo a los parámetros y requerimientos demandados por la aeronave para la posterior elección de una de las placas.

Tabla 11
Comparativa entre OpenPilot CC3D Flight Controller, Arduino Nano y FriendlyARM NanoPi Neo Plus 2

	OpenPilot CC3D Flight Controller	Arduino Nano	FriendlyARM NanoPi Neo Plus 2
CPU	90 MIP	16 MHz	1.5 GHz
Flash Memory	128 KB	32 KB	8 GB
RAM Memory	20 KB	2 KB (SRAM)	512 MB
EEPROM	4 Mbits	1 KB	1 GB
Tensión de funcionamiento	5 V	5 V	5 V
Número de canales PWM	6 canales de entrada 10 canales de salida	6 canales	1 canal
Protocolo SPI	SI	SI	SI
Protocolo I2C	SI	SI	SI

Continua Tabla 11 →

Interfaz de comunicación	Micro USB	Micro USB	Micro USB
Sensores y componentes	Matriz de giroscopio y acelerómetro de 3 ejes Funciones MainPort: telemetría en serie, GPS, S. Bus, satélites Spektrum	No especificado	Gigabit Ethernet 802.11n WiFi Bluetooth 4.0
Costo [$/unidad]	$ 25	$ 22	$ 30
Disponibilidad en el mercado	Media	Alta	Baja

De primera mano se descarta la tarjeta FriendlyARM NanoPi Neo Plus 2, debido a que su disponibilidad es baja en el mercado nacional ocasionando inconvenientes al momento de su importación por el tiempo en el que se demora en arribar la compra.

Finalmente decidiendo entre las tarjetas OpenPilot CC3D Flight Controller y Arduino Nano, aunque la disponibilidad y el coste de la placa Arduino Nano son alta y baja respectivamente, los elementos que esta dispone no se ajustan adecuadamente a los requerimientos del proyecto por lo que se descarta.

Finalmente la tarjeta seleccionada es OpenPilot CC3D Flight Controller debido a que posee mejores prestaciones en cuanto a CPU, memoria RAM y flash, número de canales PWM y principalmente que dispone de una matriz de giroscopio y acelerómetro de 3 ejes incorporado, permitiendo la utilización de esta sin la necesidad de adquirir una externa.

4.3.5 Módulo de comunicación por radio frecuencia

Para la selección del módulo de comunicación por radio frecuencia es necesario conocer la banda de trabajo en la que se enviara y receptara información desde y hacia la aeronave la cual es de 2.4 GHz. Además se busca compatibilidad con la placa OpenPilot CC3D Flight Controller y que el mismo disponga de comunicación mediante un BUS SPI.

Por esta razón se seleccionó el módulo de comunicación FRSKY DJT 2.4 GHz, ver Figura 88, el cual dispone de las siguientes características técnicas que se ajustan a los requerimientos del proyecto. Véase Tabla 12.

Tabla 12
Especificaciones técnicas módulo de comunicación FRSKY

Marca	FRSKY
Banda	2.4 GHz
Componentes	Módulo de telemetría DJI Receptor RX de V8FR-II
Receptor RX de V8FR-II	
Voltaje de funcionamiento	3.0 a 16.0V
No. De canales	8
Resolución	3072 (11 bits)
Gama	1.5 a 2.5 Km
Módulo de telemetría DJI	
Voltaje de funcionamiento	6.0 a 13.0V
Potencia de salida	60mW
Resolución	3072 (11 bits)

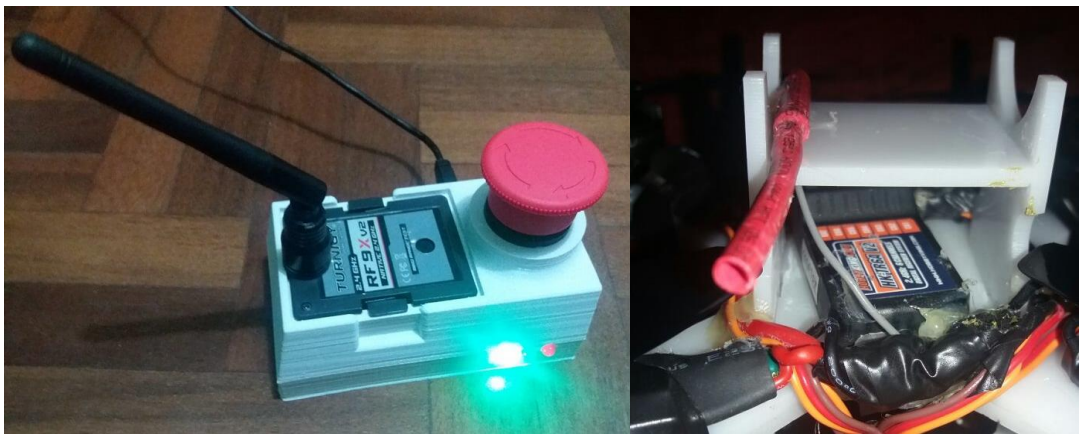


Figura 88 Módulo de comunicación FRSKY 2.4 GHz

4.3.6 Unidad de medición inercial (IMU)

La unidad de medición inercial o IMU por sus siglas en inglés Inertial Measurement Unit es necesaria para la detección de rotación del cuatrirotor en el espacio, el cual utilizada sensores con tecnología MEMS (Microelectromechanical Systems) o sistemas mecánicos y electromecánicos hechos con técnicas de microfabricación.

Los sensores que se utilizan normalmente en una IMU son: el acelerómetro, el giroscopio y el magnetómetro.

- **Acelerómetro**

Mediante el uso de sensores capacitivos se detecta el diferencial del desplazamiento causado por una aceleración de una masa colocada en el sensor.

- **Giroscopio**

Típicamente el giroscopio detecta la variación de la velocidad de rotación sobre un eje. Este es llamado también de estructura vibrante ya que al rotar el giroscopio sobre un eje, el efecto Coriolis produce una vibración que es medida por unos platos capacitivos.

- **Magnetómetro**

Es un sensor basado en el efecto Hall, mediante ese se puede conocer el valor del campo magnético aplicado.

La tarjeta de vuelo OpenPilot CC3D Flight Controller seleccionada anteriormente dispone de varios sensores y componentes, uno de ellos es el IMU Invensense MPU-6000 Gyro/Accel, el cual será nuestra unidad de medición inercial seleccionada ya que se encuentra incluida en la placa mencionada. Véase Figura 89 y Figura 90. Sus características y especificaciones técnicas se detallan en la Tabla 13.

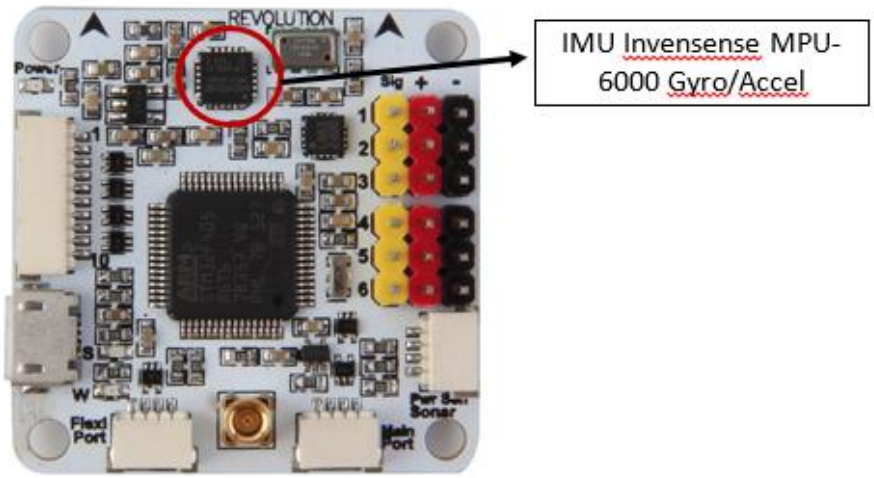


Figura 89 OpenPilot CC3D Flight Controller, localización IMU Invensense MPU-6000 Gyro/Accel

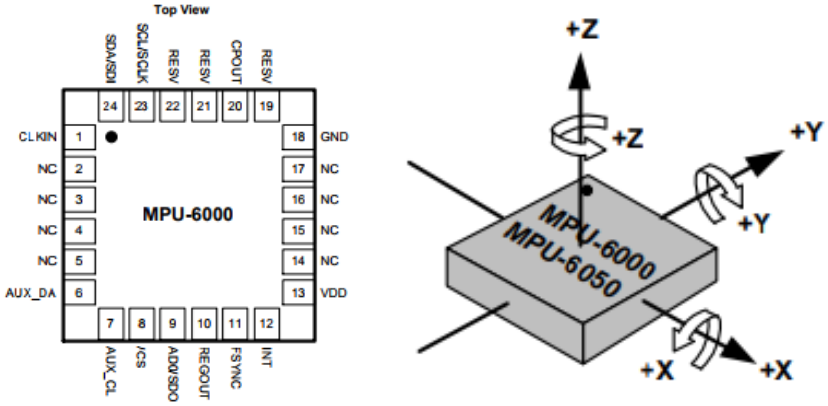


Figura 90 MPU-6000 Gyro/Accel Invensense
Fuente: (InvenSense Inc., 2013)

Tabla 13
Especificaciones MPU-6000 Gyro/Accel Invensense

Módulos a bordo	Giroscopio de 3 ejes Acelerómetro de 3 ejes DMP (Digital Motion Processor)
Beneficios clave	Giroscopio y acelerómetro programable por el usuario
Interfaz de comunicación	Serial Peripheral Interface (SPI) General Purpose Inpu/Output (GPIO) Inter Integrated Circuit (I2C)
Voltaje de alimentación	3.3 V
Compatibilidad	mikroBUS

4.3.7 Electronic Speed Control (ESC)

Es un dispositivo de electrónica de potencia que permite la conexión entre el microcontrolador y los motores, el cual realiza las siguientes funciones:

- Transforman la corriente continua de la batería a trifásica.
- Determinan el ratio de encendido y apagado en base a la recepción de señal PWM desde el microcontrolador de una red de transistores modulando la señal trifásica destinada a los motores.

El dispositivo seleccionado como ESC se basa en la corriente utilizada por los motores tipo Brushless elegidos anteriormente, los cuales consumen una corriente máxima de 10 A. Además se busca un dispositivo que esté dotado de BEC (circuito eliminador de batería) el cual actúa como un regulador de voltaje teniendo como

entrada el voltaje de la batería y como salida una señal de 5V utilizada para alimentar el microcontrolador. Para la elección del dispositivo se escogió la marca BLHeli en la cual se ofrecen varios ESCs con diversas especificaciones los cuales se muestran en la Tabla 14.

Tabla 14
Comparativa ESCs BLHeli

Item	Corriente continua [A]	Corriente Burst [10 seg.]	Batería Li-PO [celdas]	Salida BEC	Programable
BLHeli-6 A	6 A	8 A	1-2	0.8 A/ 5V	SI
BLHeli-12 A	12 A	15 A	2-4	1 A/ 5V	SI
BLHeli-20 A	20 A	25 A	2-4	2 A/ 5V	SI
BLHeli-30 A	30 A	40 A	2-4	2 A/ 5V	SI
BLHeli-40 A –UBEC	40 A	50 A	2-6	3 A/ 5V	SI
BLHeli-50 A –UBEC	50 A	60 A	2-6	3 A/ 5V	SI
BLHeli-60 A -UBEC	60 A	80 A	2-6	5 A/ 5V	SI
BLHeli-80 A –UBEC	80 A	100 A	2-6	5 A/ 5V	SI

El dispositivo seleccionado como ESC para este proyecto es el variador BLHeli-12 A. Véase Figura 91.

Figura 91 ESC BLHeli-12A

4.3.8 Batería Li-PO

Como dispositivo de alimentación se utilizara una batería de polímero de litio Li-PO. Es una batería muy utilizada para aeromodelismo ya que dispone de una alta densidad de energía y una alta tasa de descarga necesaria para alimentar los cuatro motores simultáneamente.

La elección de la batería estará influenciada por los siguientes parámetros:

- Alimentación de dispositivos de control: la alimentación del microcontrolador como máximo será de 12V de acuerdo a la elección del mismo, y el voltaje de entrada de los ESCs como mínimo deberá ser de 11.1V y máximo de 14.8V.
- Tasa de descarga: esta es necesaria para mantener la sustentación de la aeronave en el aire a través de los motores, para conocer su valor se debe conocer el máximo consumo de corriente por motor que para nuestro proyecto es de 10 A, este valor al multiplicarlo por cuatro y dividirlo por la carga eléctrica de la batería se puede calcular la tasa de descarga mínima necesaria.
- Carga eléctrica: con este parámetro se logra obtener una sustentación de la aeronave en promedio de 15 a 20 minutos. Para esto se calcula el peso aproximado del cuatrirotor, incluyendo todos los elementos inclusive la batería y a través de tablas se puede calcular el consumo de corriente en estado estacionario de la aeronave.

A continuación se presentan los cálculos para conocer la carga eléctrica y la tasa de descarga necesaria y la posterior elección de la batería. Para el caso de peso de la batería se tomara un peso promedio de baterías de 3 celdas.

$$Peso_{aeronave} = Peso_{armazon} + Peso_{motores} + Peso_{bateria} + Peso_{controladores} \quad (14)$$

$$Peso_{aeronave} = 109\ gramos + 4 * 19.2\ gramos + 117\ gramos + 4 * 0.8\ gramos$$

$$Peso_{aeronave} = 306\ gramos$$

$$\frac{Peso_{aeronave}}{No.motores} = \frac{306\ gramos}{4\ motores} = 76.5\ gramos \rightarrow I \cong 0.73\ A \quad (15)$$

$$C_{min} = I \times No.motores \times Tiempo_{vuelo} \quad (16)$$

$$C_{min} = \frac{0.73\,A}{1\,A} \times 1000mA \times 4\,motores \times \frac{20\,min}{60\,min} \times 1hora$$

$$C_{min} = 981\,mAh$$

$$Tasa_{descarga} = \frac{9.8A \times 4\,motores}{0.981\,Ah} = 40C \quad (17)$$

La carga eléctrica mínima Cmin que requiere la aeronave es de 1160 mAh, mientras que la tasa de descarga mínima deberá ser de 34 veces la carga eléctrica de la batería. Al valor calculado se deberá añadir un 10 % de margen de seguridad para evitar daños en la batería y componentes eléctricos y electrónicos.

En base a un catálogo de una tienda online se seleccionó la batería que mejor se ajusta a los requerimientos de cálculo. Véase Figura 92.

La Tabla 15 presenta la selección de la batería de acuerdo a los cálculos realizados.

Tabla 15

Comparativa batería seleccionada vs. Cálculos realizados

	Cálculos con 10 % de margen de seguridad	Batería comercial MultiStar 1400 mAh
No. Celdas	3	3
Voltaje Nominal [V]	11.1 V	11.1 V
Carga eléctrica [mAh]	1100 mAh	1400 mAh
Tasa de descarga	40C	40-80C
Peso [gramos]	117 gramos	115 gramos
Dimensiones	------	86 x 34 x 20 mm

Figura 92 Batería seleccionada Multi Star 1400 mAh

4.4 Diseño circuito de alimentación para marcadores activos

En el diseño del circuito de alimentación para los marcadores activos se plantearon algunos requerimientos. Estos se basan principalmente en mantener un alto brillo en cada LED de modo que puedan ser detectados fácilmente por el sistema de arreglo de cámaras. Por esta razón se seleccionaron tres LED de 1 watt de potencia de disipación, ver Figura 93. Las especificaciones técnicas de este elemento se describen en la Tabla 16.

Figura 93 1W High Power LED

Tabla 16

Características técnicas 1W High Power LED

Ítems	Símbolo	Máximo valor absoluto
Corriente de conducción	IF	0.35 A
Corriente máxima de conducción	IFP	0.8 A
Voltaje de conducción	VR	5 V
Potencia de disipación	PD	1 W

Fuente: (Datasheet 1 W High Power LED, s.f.)

Teniendo en consideración el voltaje máximo de conducción y teniendo una fuente de alimentación de 12 V desde la batería de la aeronave, se opta por seleccionar un regulador de voltaje de 12 a 4.6 V como dispositivo previo a la conexión de los LED de potencia, de modo que pueda lograrse la mayor cantidad de luminosidad sin dañar los elementos. Siendo estos LED dispositivos de potencia no es necesario conectar previamente resistencias para regulación de corriente.

El regulador de voltaje seleccionado según necesidad del proyecto y disponibilidad en el mercado es el D – SUN 3 A Step Down Switching Voltage Regulator, ver Figura 94. Las características de este dispositivo se citan en la Tabla 17.

Figura 94 D – SUN 3 A Step Down Switching Voltage Regulator

Tabla 17

Características técnicas D – SUN 3 A Step Down Switching Voltage Regulator

Voltaje de entrada	4.5 a 28 V
Voltaje de salida	0.8 a 20 V
Corriente de salida	3 A (máximo)

Fuente: (Datasheet D-Sun 3A Step Down Switching Voltage, s.f.)

Disponiendo de estos elementos se procedió a diseñar el circuito para su posterior implementación en la aeronave. La Figura 95 muestra el circuito a implementar.

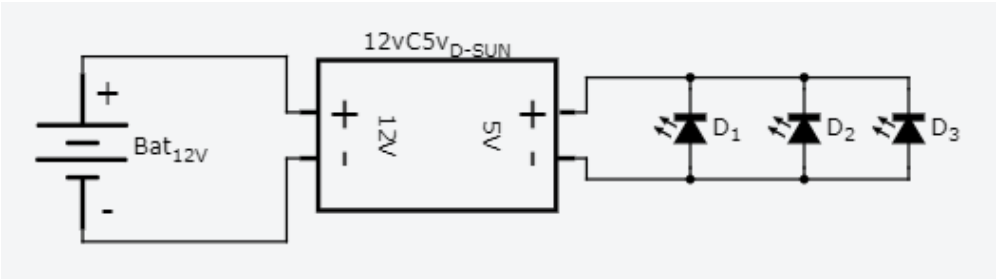


Figura 95 Circuito de alimentación para marcadores activos

4.5 Diseño circuito de comunicación PC – aeronave

Para el circuito de comunicación PC – aeronave es necesario disponer de una placa con un microcontrolador que disponga pines de entrada/salida y lectura/escritura de modo que se pueda enviar y recibir información de forma adecuada. Por este motivo

se optó por seleccionar la tarjeta Arduino Micro (ver Figura 96) de modo que esta gestione todas las acciones a realizar este circuito. Las características que dispone esta tarjeta se detallan en la Tabla 18.

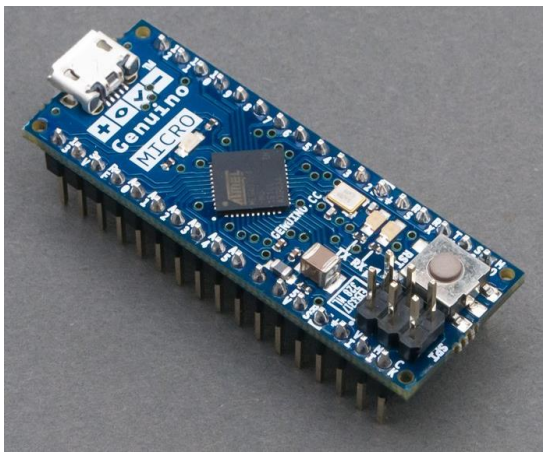


Figura 96 Arduino Micro

Tabla 18
Características Arduino Micro

Microcontrolador	ATmega32U4
Oscilador de cristal	16 MHz
Interfaz	microUSB
Pines entrada/salida	20 pines digitales 7 salidas PWM 12 entradas analógicas

Fuente: (Arduino, 2017)

Este circuito dispondrá de elementos para visualización de estados, para esto se seleccionará LED de 5 mm (ver Figura 97) con las características citadas en la Tabla 19. De este modo se procederá a calcular las resistencias necesarias para el circuito mostrado en la Figura 99. Para el cálculo se hará uso de la ecuación 18.

Figura 97 5 mm Round LED

Tabla 19

Características técnicas 5 mm Round LED

Ítems	Símbolo	Máximo valor absoluto
Corriente de conducción	IF	30 mA
Corriente máxima de conducción	IFP	100 mA
Voltaje de conducción	VR	5 V
Potencia de disipación	PD	100 mW

$$V = I \times R \tag{18}$$

$$R = \frac{V}{I} = \frac{5\ V}{0.03\ A} = 166.66\ \Omega$$

De acuerdo a la disponibilidad del mercado se seleccionó una resistencia de 330 Ω.

Además en el circuito se implementara un paro de emergencia, este paro de emergencia deberá disponer características del tipo industrial, por tal motivo se seleccionó en razón a la disponibilidad del mercado el paro de emergencia de medio giro G & V Electric normalmente cerrado, ver Figura 98.

Figura 98 Botón paro de emergencia de medio giro G & V Electric

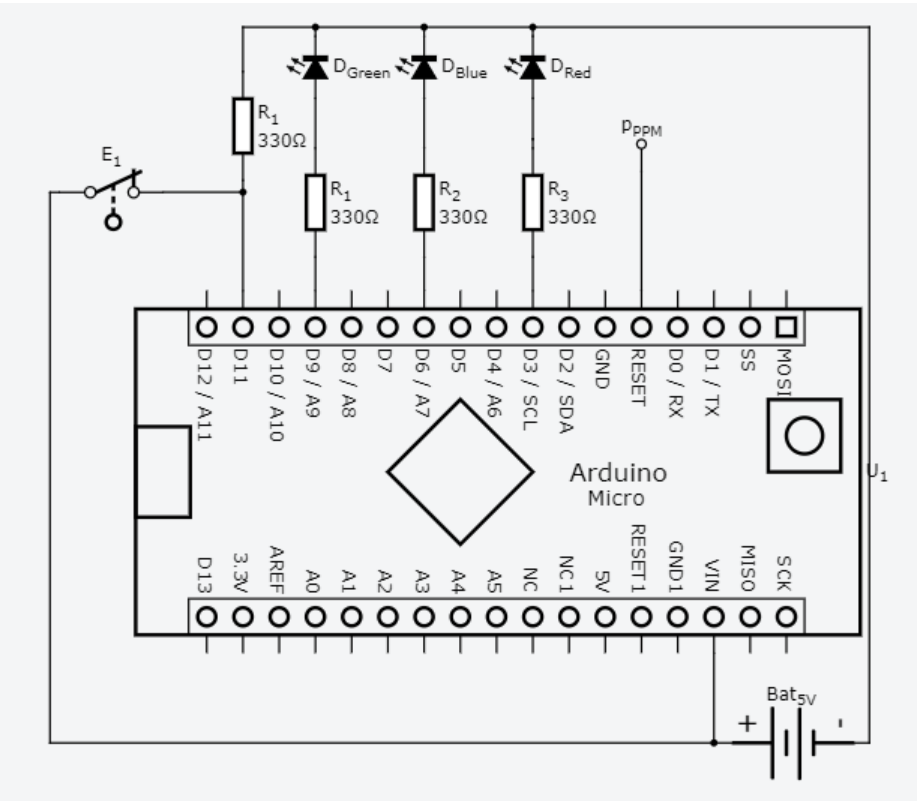


Figura 99 Circuito de comunicación PC - aeronave

CAPÍTULO V

SISTEMA DE CONTROL

5.1 Diseño del controladores PID

El modelamiento de la planta se lo realizo en base a un modelo experimental de movimiento de la aeronave, para esto se suministró una entrada al cuatrirotor de modo que se adquirió varios datos de movimiento del mismo proporcionado por el sistema de visión por computador.

	A	B	C	D	E
1	90	383,6365122	692,0418178	587,7701338	357
2	90	383,6365122	692,0418178	587,7701338	357
3	90	382,9475593	692,522028	590,3806088	357
4	90	382,9475593	692,522028	590,3806088	357
5	90	382,9475593	692,522028	590,3806088	357
6	90	382,0691777	692,5908406	592,4378268	355
7	90	382,0691777	692,5908406	592,4378268	355
8	90	382,0691777	692,5908406	592,4378268	355
9	90	382,0691777	692,5908406	592,4378268	355
10	90	381,048516	692,6399255	594,1147632	355
11	90	381,048516	692,6399255	594,1147632	355
12	90	380,2290442	692,6851586	595,4251504	355
13	90	380,2290442	692,6851586	595,4251504	355
14	90	380,2290442	692,6851586	595,4251504	355
15	90	379,5543628	691,1853754	596,480811	357
16	90	379,5543628	691,1853754	596,480811	357
17	90	379,5543628	691,1853754	596,480811	357
18	90	379,0172426	690,8883344	600,5782039	360
19	90	379,0172426	690,8883344	600,5782039	360
20	90	379,0172426	690,8883344	600,5782039	360
21	90	378,9118747	690,6507016	603,8561182	360
22	90	378,9118747	690,6507016	603,8561182	360
23	90	378,9118747	690,6507016	603,8561182	360
24	90	379,1489605	689,8299685	606,5390808	357

(a)

	A	B	C	D	E
1	70	1030,706584	290,6698976	611,3137246	360
2	70	1030,706584	290,6698976	611,3137246	360
3	70	1030,706584	290,6698976	611,3137246	360
4	70	1030,706584	290,6698976	611,3137246	360
5	70	1031,29435	290,0147722	609,8412577	360
6	70	1031,29435	290,0147722	609,8412577	360
7	70	1031,29435	290,0147722	609,8412577	360
8	70	1031,390303	289,1664836	615,0938852	360
9	70	1031,390303	289,1664836	615,0938852	360
10	70	1032,266655	288,0097731	616,0413086	360
11	70	1032,266655	288,0097731	616,0413086	360
12	70	1032,266655	288,0097731	616,0413086	360
13	70	1032,266655	288,0097731	616,0413086	360
14	70	1032,266655	288,0097731	616,0413086	360
15	70	1031,848301	286,9390614	620,0539259	360
16	70	1031,848301	286,9390614	620,0539259	360
17	70	1030,694293	285,9218694	620,0236467	360
18	70	1030,694293	285,9218694	620,0236467	360
19	70	1030,694293	285,9218694	620,0236467	360
20	70	1029,524727	285,1098671	620,0382487	360
21	70	1029,524727	285,1098671	620,0382487	360
22	70	1029,524727	285,1098671	620,0382487	360
23	70	1028,63523	284,4711898	620,2703223	360
24	70	1028,63523	284,4711898	620,2703223	360

(b)

Figura 100 Datos obtenidos modelo experimental pitch (a) y roll (b) de la aeronave

Una muestra de los datos obtenidos se presenta en la Figura 100 de los movimientos pitch y roll, respectivamente. La ecuación de modelamiento de la planta se obtuvo a través de un software de simulación, estas ecuaciones se muestran a continuación:

Planta movimiento pitch

$$G_1(s) = \frac{313.6}{s^2+7.029s+34.39} \tag{19}$$

El comportamiento de la planta en función del movimiento pitch ante una entrada escalón se visualiza en la Figura 102.

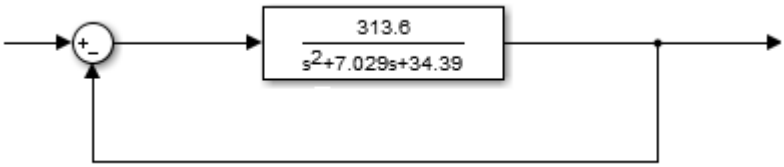


Figura 101 Planta pitch en lazo cerrado

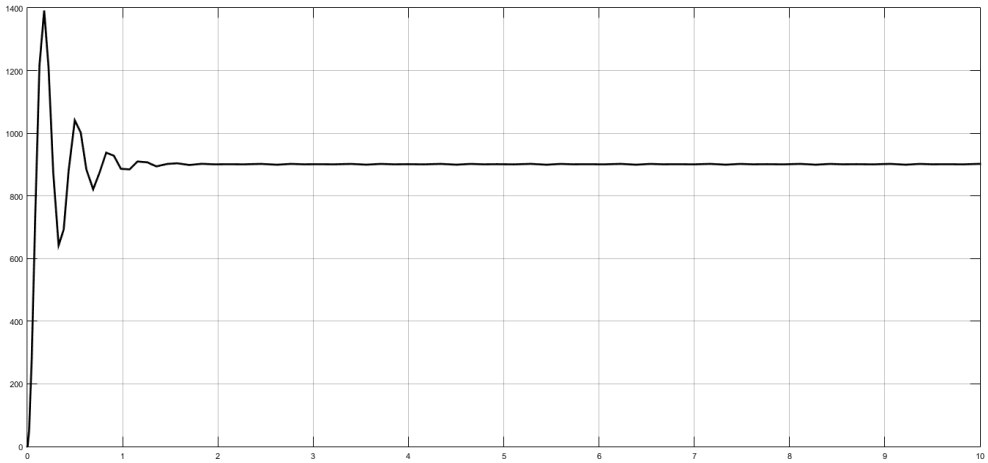


Figura 102 Comportamiento planta pitch en lazo cerrado ante entrada escalón

Planta movimiento roll

$$G_2(s) = \frac{313.6}{s^2+7.029s+34.39} \tag{20}$$

El comportamiento de la planta en función del movimiento roll ante una entrada escalón se visualiza en la Figura 104.

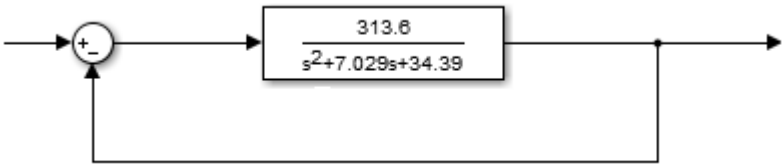


Figura 103 Planta roll en lazo cerrado

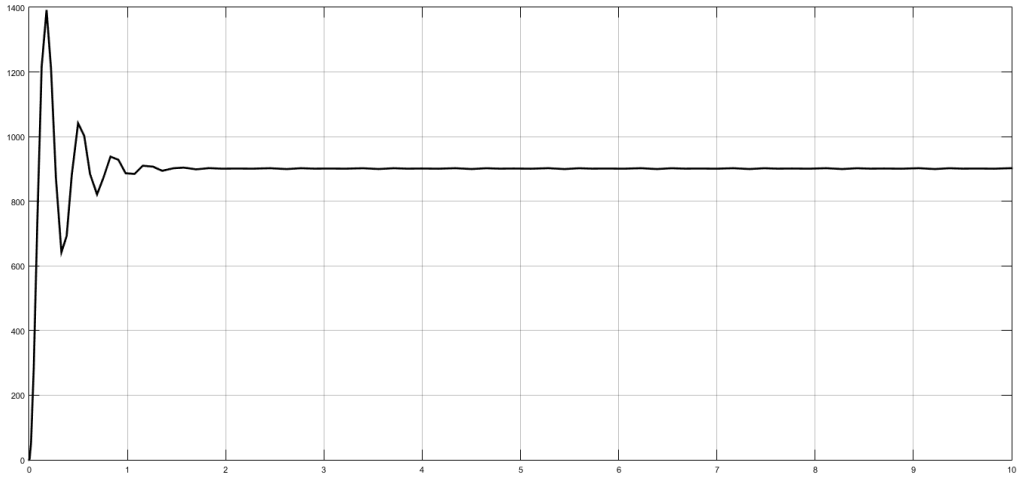


Figura 104 Comportamiento planta roll en lazo cerrado ante entrada escalón

Luego de realizado el modelamiento individual por cada tipo de movimiento se observa que tanto la planta para pitch como para roll son las mismas teniendo un comportamiento similar por lo tanto se puede diseñar un solo controlador que se ajuste a las dos plantas.

- **Controlador PID movimiento pitch y movimiento roll**

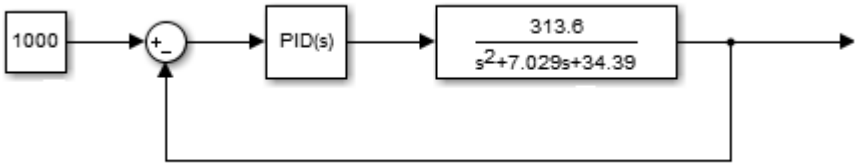


Figura 105 Plantas pitch y roll en lazo cerrado con control PID.

Para el diseño del controlador PID se tiene el modelo de ambas plantas como:

$$G_P(s) = \frac{313.6}{s^2+7.029s+34.39} \tag{21}$$

El comportamiento del sistema en lazo cerrado ante una entrada escalón se puede observar en la Figura 106.

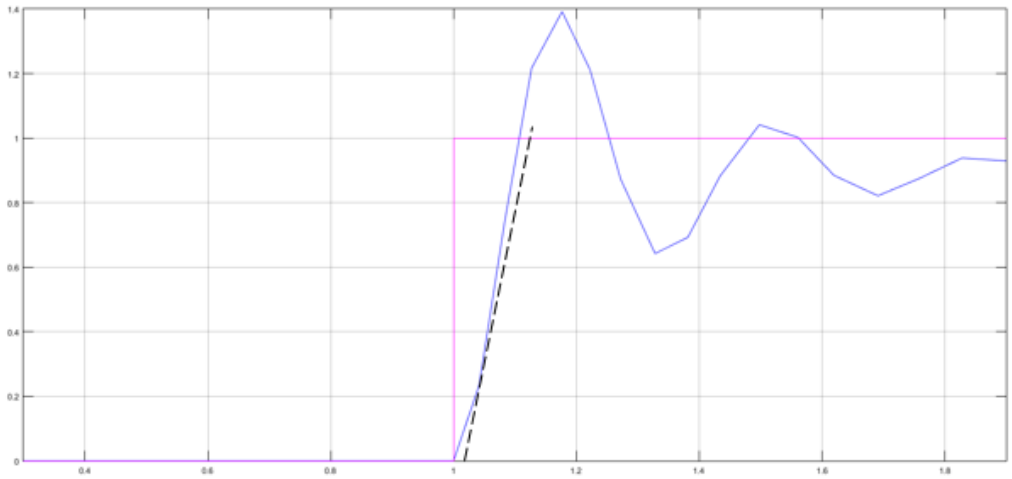


Figura 106 Comportamiento de la planta ante entrada escalón unitario

Para el cálculo de las constantes de control se utilizara el método de Ziegler-Nichols el cual nos permitirá ajustar o sintonizar el controlador PID de forma empírica. Para esto se necesita trazar un recta tangente (recta entrecortada negra) a la señal del salida del sistema (curva azul).

En base a esto se hallara dos tiempos, el tiempo T1 corresponde al tiempo muerto, el cual hace referencia al valor que tarda el sistema en comenzar a responder. Se mide desde que la señal escalón sube hasta el punto de corte de la recta tangente con el valor inicial de la curva del sistema, para este caso el valor es cero. En la Figura 107 se observa el cálculo de este tiempo.

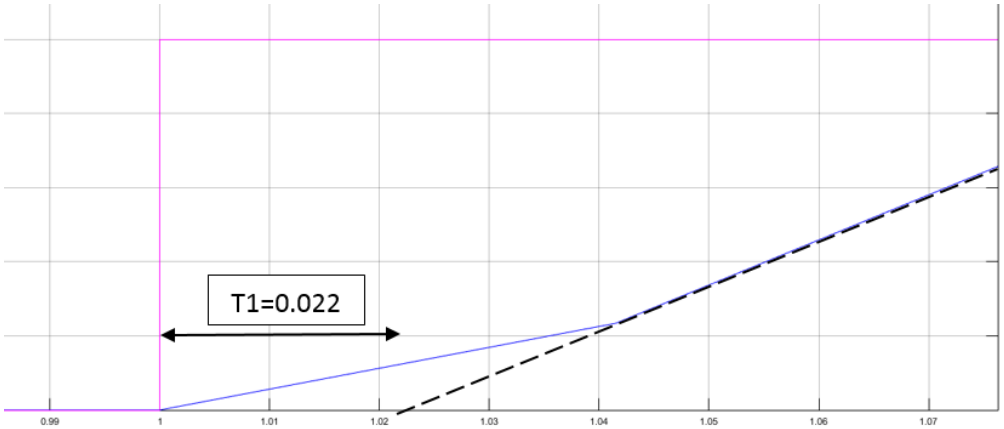


Figura 107 Cálculo de la constante de tiempo T1, método de Ziegler-Nichols

Para el caso del tiempo T2 este hace referencia al tiempo de subida, el cual se halla desde el punto de corte de la recta tangente con el valor inicial hasta el punto de intersección de la misma recta el valor final del sistema. En la Figura 108 se muestra el cálculo de este tiempo.

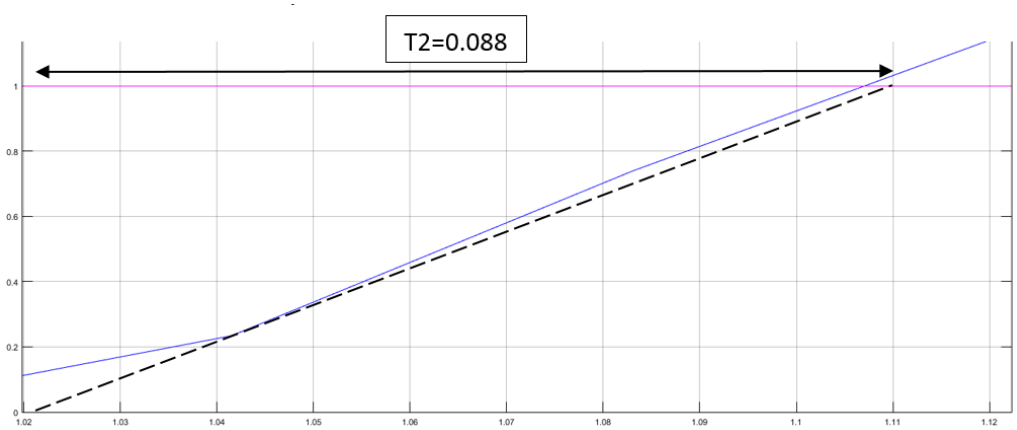


Figura 108 Cálculo de la constante de tiempo T2, método de Ziegler-Nichols

A partir de estos parámetros T1 = 0.022 y T2 = 0.088 se puede calcular la constante del sistema Ko de acuerdo a las ecuación mostrada a continuación.

$$Ko = \frac{dX * T2}{dY * T1} \tag{22}$$

$$dX : variación\ de\ la\ señal\ escalón = \ 5\ V$$
$$dY : variación\ de\ la\ respuesta\ del\ sistema = 1000\ mm$$

$$Ko = \frac{(5) * (0.088)}{(1000) * (0.022)} = 0.02\ V/mm$$

De acuerdo al valor calculado de Ko se puede hallar las constantes del controlador PID en base a las ecuaciones detalladas en la Tabla 20.

Tabla 20
Ecuaciones de cálculo de parámetros del controlador PID en base al Método de Ziegler-Nichols

	Kp	Ki	Kd
P	Ko		
PI	0.9*Ko	0.27*Ko/T1	
PID	1.2*Ko	0.60*Ko/T1	0.60*Ko*T1

Aplicando las ecuaciones, se hallaron los valores de las constantes del controlador como:

$$Kp = 1.2*Ko = 1.2*(0.02) = 0.024 \quad (23)$$

$$Ki = 0.60*Ko/T1 = 0.60*(0.02)/(0.022) = 0.54 \quad (24)$$

$$Kd = 0.60*Ko*T1 = 0.60*(0.02)*(0.022) = 0.000264 \quad (25)$$

5.2 Diseño de algoritmos de control para generación de trayectoria

5.2.1 Algoritmo RRT

El diseño del algoritmo RRT para esta aplicación se basa en la construcción de un árbol de ramificaciones el cual crece desde el punto inicial hasta lograr alcanzar el punto final. El tipo de algoritmo RRT a implementar será de tipo unidireccional este se basa en una serie de conceptos y parámetros detallados a continuación:

- C: conjunto de configuraciones posibles de la aeronave en el espacio.
- C_{libre}: subconjunto de C de las configuraciones posibles que no colisionan ningún obstáculo en el espacio.
- $p_{inicial}$: representa el punto inicial o coordenadas en el plano X e Y; y la orientación de la aeronave respecto a su eje de referencia X.
- p_{final}: representa el punto final que se desea alcanzar.
- p_{rand}: representa un punto aleatorio generado a partir del algoritmo.
- p_{prox}: representa un punto cercano al punto aleatorio generado por el algoritmo.
- p_{nuevo}: representa un nuevo punto agregado a la ramificación del árbol de trayectoria.
- Δ: longitud del segmento de crecimiento de cada ramificación del árbol.

Para la escritura del algoritmo RRT se implementara dos instrucciones. Con la primera se obtiene un punto aleatorio dentro del espacio libre de colisión; mientras que la segunda instrucción hace crecer el árbol en dirección del punto aleatorio

obtenida en la primera. Este se encuentra limitado por un número máximo de iteraciones ***kmax***.

La ampliación del árbol se obtiene con una función a la que llamaremos ***crecimiento***. La estructura de esta función está basada en el cálculo del punto próximo mediante la función ***generación*** creando el punto más cercano al punto aleatorio. Ver Figura 109.

```
algoritmoRRT (pinicial)
{
        arbol[0] = pinicial;
        para k = 1 hasta kmax
        {
                prand = configuracionAleatoria();
                crecimiento(arbol, prand);
        }
        retornar arbol;
}
```

Figura 109 Algoritmo RRT unidireccional
Fuente: (López, Gómez-Bravo, Cuesta, & Ollero, 2006)

Inmediatamente la función ***nuevopunto*** calcula el punto nuevo que será agregado a la ramificación de la trayectoria final mediante la inserción de la longitud de segmento Δ partiendo del punto próximo al punto aleatorio. Es importante mencionar que al obtener el punto nuevo se debe comprobar si este colisiona o no con algún obstáculo, para el caso de generarse colisión se debe retornar a la primera instrucción. Ver Figura 110.

Si el punto aleatorio se localiza a una distancia Δ del punto próximo se considera que el punto ha sido ***alcanzado***. Si por el contrario no se ha logrado alcanzar el punto aleatorio devuelve el valor de ***avanzado***. Si el punto aleatorio colisiona con algún obstáculo regresa el valor de ***rechazado***.

```
Función crecimiento(arbol, prand)
{
        pnuevo =  nuevopunto(prand, arbol);
        Si ( nuevaconfiguracion (prand, pprox, pnuevo) )
        {
                nuevovertice(arbol, pnuevo);
                Si ( pnuevo = prand )
                {
                        retornar alcanzado;
                }
                Caso contrario
                {
                        retornar avanzado;
                }
        Caso contrario
        {
                retornar rechazado;
        }
```

Figura 110 Funciones del algoritmo RRT
Fuente: (López, Gómez-Bravo, Cuesta, & Ollero, 2006)

La generación de trayectoria y ramificación se puede observar en la Figura 111**.**

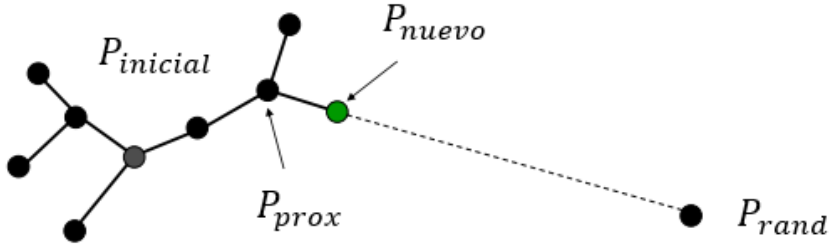


Figura 111 Algoritmo RRT generación de trayectoria

5.2.2 Optimización de trayectoria

La trayectoria generada con el algoritmo RRT resulta compleja para ser recorrida por la aeronave debido a que dispone de giros complicados, por esta razón es aconsejable realizar una optimización de la misma para evitar estas complicaciones. Para esto se adicionara al código del algoritmo RRT un conjunto de sentencias que generan de forma iterativa un camino más simple. Ver Figura 112.

```
Función optimizacion(algoritmoRRT)
{
        fin = 0;
        Mientras (no fin)
        {
                colision = 1;
                a = numerovertices;
                Mientras (colision)
                {
                        Si ( segmentovalido(1, a) )
                        {
                                eliminarruta(1, a, algoritmoRRT, trayectoriafinal);
                                colision = 0;
                        }
                        Caso contrario
                        {
                                a = a/2;
                        }
                        Si ( a ≤ 2)
                        {
                                eliminarruta(1,2,algoritmoRRT,trayectoriafinal);
                                colision = 0;
                        }
                }
                Si (numerovertices ≤ 2)
                {
                        fin = 1;
                }
        }
        retornar(trayectoriafinal);
}
```

Figura 112 Algoritmo optimización trayectoria RRT
Fuente: (López, Gómez-Bravo, Cuesta, & Ollero, 2006)

Este algoritmo de optimización se basa en la eliminación de tramos de la ramificación y la inserción de una trayectoria recta libre de colisión. La función ***numerovertices*** devuelve el número de vértices que componen la trayectoria original y el carácter ***a*** el número de vértices por el cual será reemplazado.

En cada iteración, la función ***segmentovalido(1, a)*** genera una ruta entre el vértice *1* y el vértice *a* y a su vez genera una serie de puntos comprobando de igual manera si existe colisión sobre estos. En el caso de existir un segmento de trayectoria valida da paso a la función ***eliminarruta****,* que se encarga de eliminar de la trayectoria original todos los vértices, disminuye el valor que devolverá la función ***numerovertices*** e incorporara la nueva ruta a la trayectoria final. Ver Figura 113.

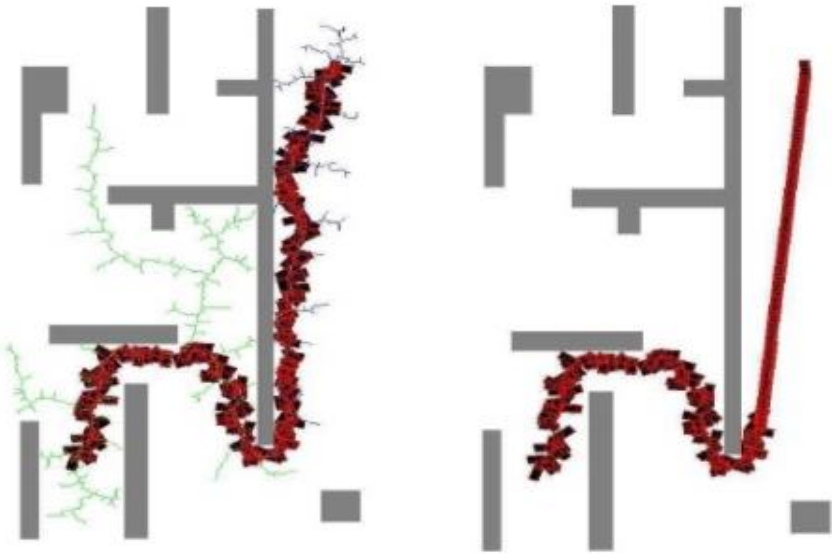


Figura 113 Ejemplo de aplicación de algoritmo de optimización de trayectoria RRT
Fuente: (López, Gómez-Bravo, Cuesta, & Ollero, 2006)

CAPÍTULO VI

CONSTRUCCIÓN E IMPLEMENTACIÓN

6.1 Construcción e implementación de los componentes mecánicos

6.1.1 Construcción de vehículo aéreo no tripulado

En la armazón diseñada en el software CAD y luego de ser ensamblada, se debe incorporar todos los elementos que fueron seleccionados en el Capítulo IV, como son los motores, ESCs, placa controladora de vuelo y batería. A continuación se presenta documentado la construcción del cuatrirotor:

En primer lugar se ensambla todos los elementos de acrílico maquinados en corte laser de modo que pueda formarse la armazón del cuatrirotor. Ver Figura 114.

Figura 114 Armazón cuatrirotor

Seguidamente se acoplara los elementos eléctricos y electrónicos como son motores, placa controladora de vuelo y ESCs. Ver Figura 115. Para la adaptación de los ESCs es necesario colocar un elemento aislante eléctrico que recubra el controlador de velocidad de modo que pueda evitarse posibles corto circuitos con otros elementos electrónicos. De igual manera cada uno de estos dispositivos se deberá mantener sujetos al armazón con correas o amarras, impidiendo que existan vibraciones y desconexión en estos elementos.

Figura 115 Cuatrirotor con elementos eléctricos y electrónicos

6.1.2 Construcción de bastidor para arreglo de cámaras

En base al diseño CAD realizado por software, se debe construir la estructura en la que se montara el arreglo de cámaras. En primera instancia se adquiere perfiles cuadrados de aluminio de 1 pulgada, adicionalmente se dispone de acoples plásticos para facilitar la unión de los perfiles de aluminio. Con estos dos elementos se construyó un prisma rectangular. Véase Figura 116.

Figura 116 Prisma rectangular con perfiles cuadrado de aluminio

Seguidamente a la estructura ya construida se adicionara una red de seguridad de nylon, la cual evitara que el cuatrirotor abandone el espacio de trabajo. La estructura en conjunto a la red de seguridad se presenta en la Figura 117.

Figura 117 Implementación de red de seguridad a la estructura

Consecutivamente se construirán dos estructura en forma de C con el perfil de aluminio en conjunto a los acoples plásticos, se montaran la mismas al prisma rectangular con ayuda de piezas estructurales de ángulo recto, adicionalmente se maquinaran agujeros para la posterior colocación del arreglo de las cuatro cámaras. Las tres estructuras en conjunto se muestran en la Figura 118.

Figura 118 Implementación de estructuras para soporte de arreglo de cámaras

Finalmente se implementara el arreglo de las cuatro cámaras en las dos estructuras en C montadas anteriormente, en conjunto a los dispositivos de conexión HUB USB 2.0. En la Figura 119 se presenta el bastidor implementado.

Figura 119 Bastidor completo implementado

6.2 Implementación de los componentes eléctricos y/o electrónicos

6.2.1 Circuito de interface PC – aeronave

En la Figura 120 se muestra la implementación del circuito de comunicación PC – aeronave sobre el soporte diseñado en el software CAD. En este se puede observar el módulo de comunicación por radio frecuencia, el microcontrolador como placa comercial Arduino Nano y el botón de emergencia.

Figura 120 Circuito de comunicación PC – aeronave

La programación de la placa Arduino se muestra en el Anexo A.

6.2.2 Circuito de alimentación para marcadores activos

El circuito presentado en la Figura 121 hace referencia al circuito de alimentación implementado en el cuatrirotor para la activación de los marcadores activos que serán detectados por el sistema de arreglo de cámaras.

Figura 121 Circuito de alimentación para marcadores activos

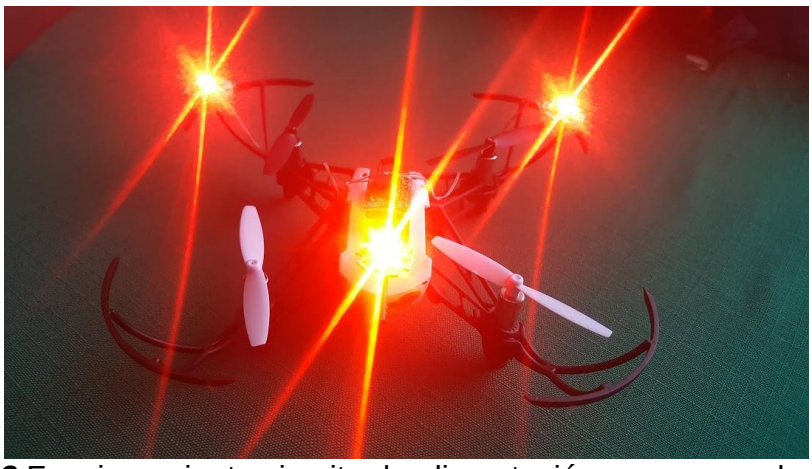


Figura 122 Funcionamiento circuito de alimentación para marcadores activos

La Figura 122 presenta el funcionamiento del circuito implementado en el cuatrirotor.

6.3 Desarrollo e implementación de software

La implementación de software incluye varios programas realizados bajo lenguaje de programación Python, la comunicación entre estos programas se da en base a ROS con la creación de nodos, tópicos, publicadores y suscriptores en cada uno de los programas. A continuación se describen cada uno de los algoritmos y su aplicación en el proyecto.

6.3.1 Implementación de filtros físicos y digitales para procesamiento de imágenes

La implementación de un filtro físico en cada una de las cámaras es necesario debido a que se evita el uso excesivo de filtros digitales y principalmente se puede obtener únicamente la luz emitida por los marcadores activos implementados en el cuatrirotor esto se logra utilizado como filtro una película opaca obtenida de los disquetes. En la Figura 123 se muestra la implementación de filtros físicos en una cámara.

Figura 123 Filtro físico en cámara

Adicionalmente al filtro físico se implementó dos filtros digitales que permiten una mejor recepción de luminiscencia emitido por los marcadores, estos filtros son: filtro escala de grises y filtro threshold, estos permiten representar el valor de cada pixel en su equivalente a una graduación de gris y segmentar la imagen en escala de grises

para obtener únicamente los tres marcadores. Las Figuras 124 y Figura 125 presentan la adición de filtro tanto en escala de grises y filtro threshold respectivamente.

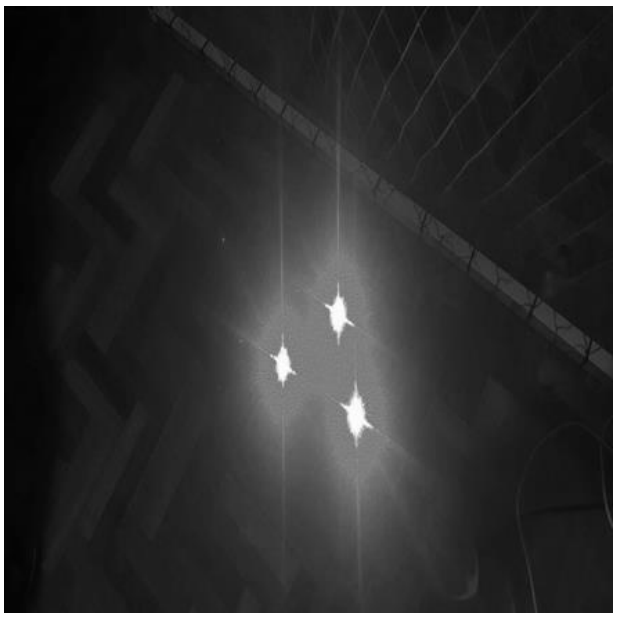


Figura 124 Aplicación de escala de grises sobre imagen de cuatrirotor

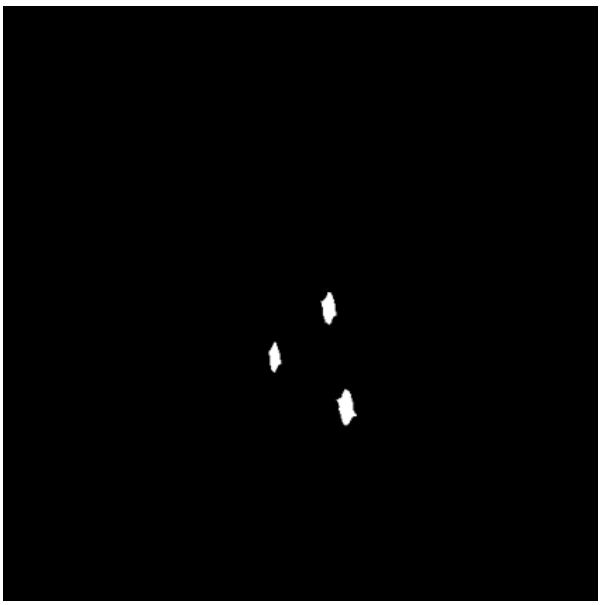


Figura 125 Aplicación de filtro threshold sobre imagen aplicada escala de grises

El código de programación para el procesamiento de imágenes así como el código de calibración de cámaras se presenta en el Anexo B.

6.3.2 Implementación de sistema de visión por computador

La implementación del sistema de visión por computador se basa en la visualización y procesamiento digital de imágenes. Para el caso de procesamiento digital de imágenes se expone su implementación en el punto 6.3, mientras que la visualización se presenta en la ejecución de una interfaz gráfica con la que interactúe el usuario.

Una vez implementado el arreglo de cámaras, se procede a realizar la unión de video captado por cada uno de ellas, generando una sola imagen de la cual se va a requerir para el cálculo de posición y altitud de la aeronave. En la Figura 126 se aprecia la captura de imágenes de cada cámara.

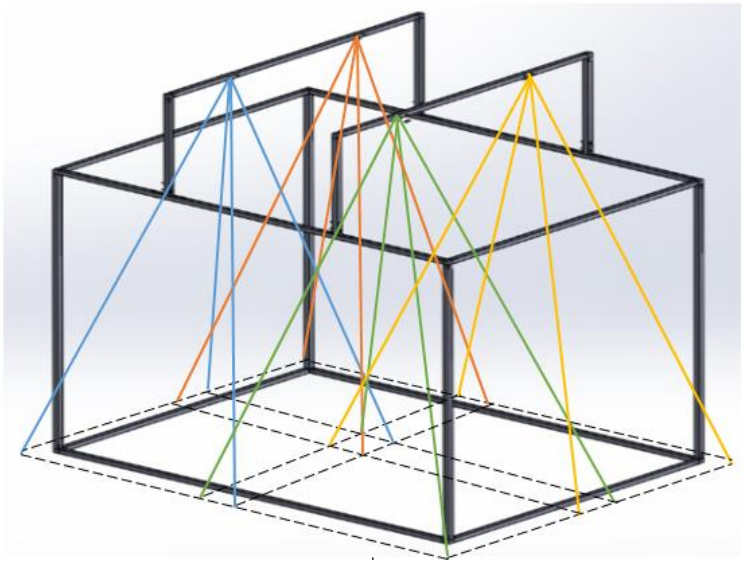


Figura 126 Captura de imágenes arreglo de cámaras

Para el cálculo de posición y altitud se adquirió datos de forma experimental de los movimientos pitch, roll y throttle en relación a los pixeles captados por las cámaras, esta información se exporto a una hoja de cálculo para su posterior procesamiento, en la Tabla 21 se aprecia los datos tomados y exportados a una hoja de cálculo con respecto a la cámara 0, en esta imagen se muestran una pequeña parte de los datos recopilados en cuanto pixeles y coordenadas X, Y y Z.

Tabla 21
Datos adquiridos desde cámara 0

PIXELES X	PIXELES Y	PIXELES Z	POSICION X	POSICION Y	POSICION Z
269	171	39,01281841	200	150	0
317	207	39	400	300	0
366	240	36,23534186	600	450	0
412	275	39,01281841	800	600	0
460	312	39,01281841	1000	750	0
508	345	37,21558813	1200	900	0
556	381	39,45883931	1400	1050	0
603	418	38,20994635	1600	1200	0
271	449	37,05401463	200	1350	0
323	415	38,20994635	400	1200	0
367	382	38	600	1050	0
413	346	39,11521443	800	900	0
460	312	39	1000	750	0
508	274	40,04996879	1200	600	0
556	238	40,04996879	1400	450	0
261	169	42,04759208	200	150	150
359	240	37,20215048	600	450	150
409	281	41,0487515	800	600	150
462	319	42,01190308	1000	750	150
511	352	41,0487515	1200	900	150
570	393	41,01219331	1400	1050	150
618	430	41,01219331	1600	1200	150
267	460	36,61966685	200	1350	150
314	426	40,01249805	400	1200	150
361	390	41,01219331	600	1050	150
415	351	41	800	900	150
466	317	42,01190308	1000	750	150
515	279	42	1200	600	150
564	238	43,01162634	1400	450	150
255	168	45,09988914	200	150	300
300	204	45,27692569	400	300	300
353	242	45,17742799	600	450	300
407	280	45,27692569	800	600	300
468	319	45,27692569	1000	750	300
522	358	45,09988914	1200	900	300
576	403	44	1400	1050	300
615	437	45,01110974	1600	1200	300
310	449	42	400	1200	300
364	405	43,0464865	600	1050	300
419	366	44,01136217	800	900	300
474	326	45	1000	750	300
530	282	46,01086828	1200	600	300
586	240	46,04345773	1400	450	300
247	163	48,01041554	200	150	450

Continua Tabla 21 →

299	199	48,09365863	400	300	450
364	237	48,01041554	600	450	450
417	275	49	800	600	450
478	334	48,01041554	1000	750	450
535	377	48,09365863	1200	900	450
597	419	48,0416486	1400	1050	450
611	442	47,26520919	1600	1200	450
306	457	42,80186912	400	1200	450
361	417	46,09772229	600	1050	450
420	372	47,09564736	800	900	450
479	332	48,09365863	1000	750	450
537	287	48,01041554	1200	600	450
599	240	50,03998401	1400	450	450
235	154	53,03772242	200	150	600
298	195	54,08326913	400	300	600
364	244	53,03772242	600	450	600
430	291	53,15072906	800	600	600
497	340	53,08483776	1000	750	600
561	393	52,03844733	1200	900	600
612	438	53,08483776	1400	1050	600
369	438	51,0881591	600	1050	600
431	390	51,0881591	800	900	600
612	245	54,03702434	1400	450	600
228	150	57,03507693	200	150	750
297	197	58,07753438	400	300	750
366	249	59,13543777	600	450	750
439	299	58,07753438	800	600	750
510	353	57,03507693	1000	750	750
580	411	58,00862005	1200	900	750
609	444	56,00892786	1400	1050	750
434	406	56,0357029	800	900	750
506	355	57,03507693	1000	750	750
575	300	58,03447251	1200	600	750
609	246	59,07622195	1400	450	750
215	132	63,00793601	200	150	900
283	186	64,00781202	400	300	900
360	246	65,00769185	600	450	900
438	305	64	800	600	900
521	362	63,00793601	1000	750	900
597	425	63,00793601	1200	900	900
606	444	60,00833275	1400	1050	900
443	419	62,03224968	800	900	900
525	364	64	1000	750	900
600	304	64,03124237	1200	600	900
201	115	69	200	150	1050
277	182	71,06335202	400	300	1050
362	248	72,02777242	600	450	1050
457	311	72,11102551	800	600	1050
548	376	71,06335202	1000	750	1050

Continua Tabla 21 →

600	420	72,69112738	1200	900	1050
451	443	64,28063472	800	900	1050
542	379	71,34423593	1000	750	1050
601	337	73,43704787	1200	600	1050
182	129	80,05623024	200	150	1200
289	199	81,74350127	400	300	1200
392	273	81,88406438	600	450	1200
492	340	81,49846624	800	600	1200
596	417	82,3893197	1000	750	1200
598	417	82,21921916	1200	900	1200
575	423	80,23091673	1000	750	1200
597	406	81,74350127	1200	600	1200

Una vez discriminados ciertos valores que se repiten en cada una de las series, se procedió a interpolar estos puntos para hallar una ecuación que se ajuste a los valores recopilados, la Figura 127 muestra el ajuste de curvas en base a la información recopilada.

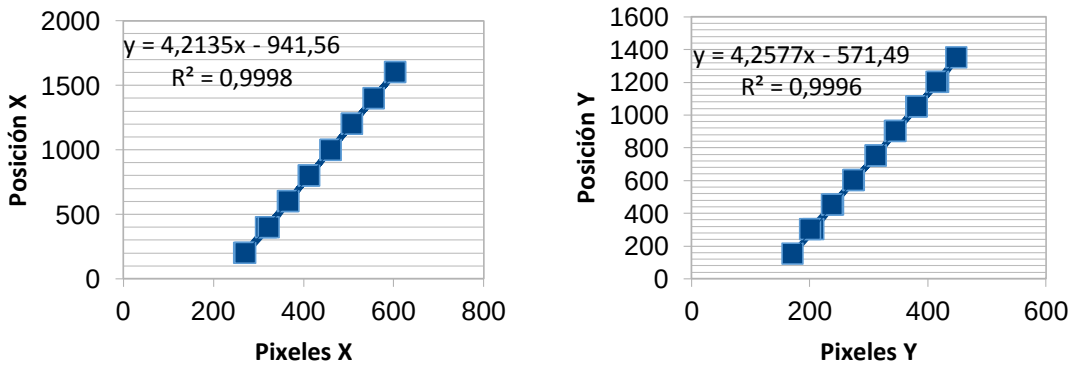


Figura 127 Primera interpolación y ajuste de curvas para cámara 0

En base a las ecuaciones generadas se utilizó sus coeficientes para determinar nuevamente una serie de puntos y que partir de estos se genere un ajuste de nuevas curvas que se satisfaga con sus coeficientes al cálculo de posición en X e Y y altitud en Z en función de los pixeles detectados por las cámaras. En la Tabla 22 se muestra la recopilación de coeficientes de las curvas ajustadas anteriormente.

Tabla 22
Coeficientes obtenidos de la primera interpolación en cámara 0

VALORES X			VALORES Y		
ALTURA	1er coeficiente	2do coeficiente	ALTURA	1er coeficiente	2do coeficiente
0	4,213503836	-941,56452	0	4,25773567	-571,494707
150	3,935839349	-825,695733	150	4,02940526	-521,4978996
300	3,704813811	-735,146038	300	3,75704895	-462,7015883
450	3,544239168	-682,249628	450	3,50872979	-401,302091
600	3,136782509	-547,039491	600	3,11770667	-316,200084
750	3,038786615	-519,442275	750	2,93483469	-282,5260269
900	2,72245905	-389,064212	900	2,68690092	-211,3153242
1050	2,419677921	-285,89768	1050	2,3364677	-129,7567457
1200	2,253482326	-248,150967	1200	2,10437151	-122,4729272

En base a estos valores se genera cuatro graficas de los cuales ajustando los puntos graficados obtenemos los coeficientes finales para el cálculo de posición de la aeronave en función de los pixeles. En las Figuras 128 y 129 y Tabla 23 se muestran las gráficas obtenidas y su posterior utilización de sus coeficientes para el cálculo de la posición del cuatrirotor respectivamente.

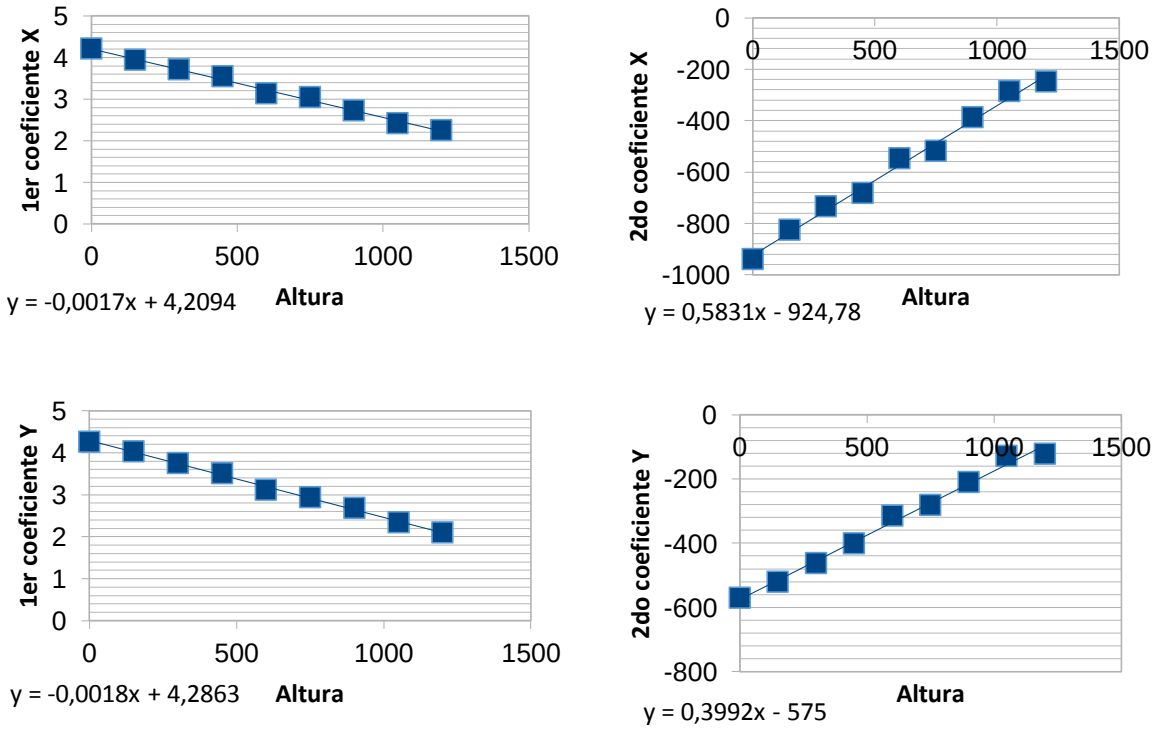


Figura 128 Segunda interpolación y ajuste de curvas para cámara 0.

Las ecuaciones 26 - 29 hacen referencia al cálculo de los coeficientes uno y dos para X e Y respectivamente, estas se basan en el ajuste de curvas mostrado en la Figura 128. Las ecuaciones presentadas de aquí en adelante son específicamente para la cámara 0, mientras que para las cámaras 1, 2 y 3 las ecuaciones se calculan con el mismo proceso.

$$1er\ coef._{X0} = -0.0017 * Altura_0 + 4.2094 \quad (26)$$

$$2do\ coef._{X0} = 0.5831 * Altura_0 - 924.78 \quad (27)$$

$$1er\ coef._{Y0} = -0.0018 * Altura_0 + 4.2863 \quad (28)$$

$$2do\ coef._{Y0} = 0.3992 * Altura_0 - 575 \quad (29)$$

De la misma manera las ecuaciones 30 y 31 muestran las ecuaciones que se ajustan al cálculo final de la posición en X e Y, estas ecuaciones se basan específicamente para la cámara 0 y son dependientes del valor de pixel capturado por cámara.

$$Posición_{X0} = 1er\ coef._{X0} * pixel_0 + 2do\ coef._{X0} \quad (30)$$

$$Posición_{Y0} = 1er\ coef._{Y0} * pixel_0 + 2do\ coef._{Y0} \quad (31)$$

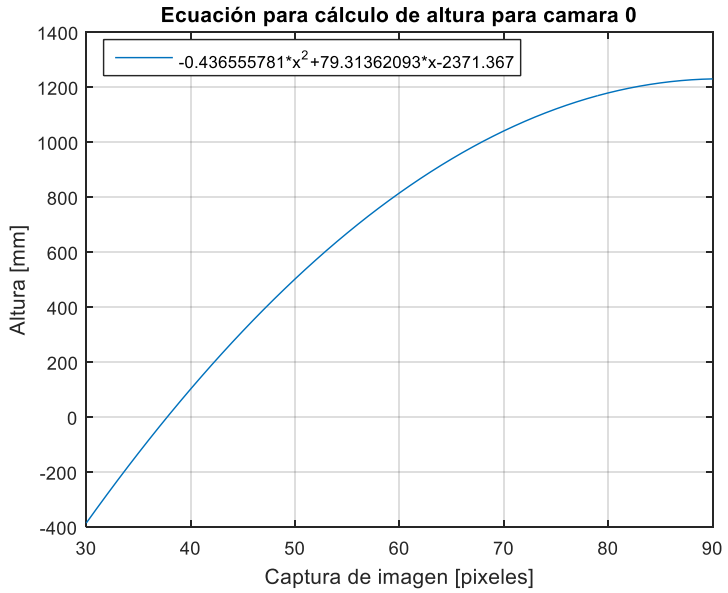


Figura 129 Curva y ecuación general para la cámara 0 para cálculo de altura

Para el cálculo de la altura de la aeronave, se utiliza la ecuación 32 la cual es generada a partir de los datos experimentales obtenidos por cámara.

$$Altura_0 = -0.436555781 * {pixel_0}^2 + 79.31362093 * pixel_0 - 2371.367 \quad (32)$$

Este proceso se lo realiza para cada una de las cámaras de modo que se pueda obtener un sistema de ecuaciones, el cual será implementado como el algoritmo de cálculo.

Mediante la utilización de este algoritmo de cálculo que tiene como variable independiente los pixeles captados por cada cámara se puede determinar la posición en coordenadas X e Y de la aeronave, de la misma manera se obtiene el valor de altitud de la misma. La Figura 130 muestra una parte de la HMI en donde se muestra la posición y altitud obtenidas en base al algoritmo.

Los valores finales de posición en X, Y y Z, se obtienen al promediar los valores de posición obtenidos por cada una de las cámaras, de esta manera se reduce el error debido que mientras mayor sea la cantidad de datos obtenidos el error es el mínimo. Ecuaciones 10 y 11.

$$Final_{position_X} = \frac{\sum_{i=0}^{3} Position_{Xi}}{4} \ [mm] \quad \textbf{(10)}$$

$$Final_{position_Y} = \frac{\sum_{i=0}^{3} Position_{Yi}}{4} \ [mm] \quad \textbf{(11)}$$

Tabla 23
Ejemplo de cálculo final de posición X, Y y Z para altura en cámara 0.

CÁLCULO POSICIÓN X				
PIXEL [pixeles]	ALTURA [mm]	1ER COEF	2DO COEF	POSICIÓN X [mm]
269	0	4,20942489	-924,784686	207,5506101
CÁLCULO POSICION Y				
PIXEL [pixeles]	ALTURA [mm]	1ER COEF	2DO COEF	POSICIÓN Y [mm]
171	0	4,28634194	-574,9981	157,9663712
CÁLCULO POSICION Z				
PIXEL [pixeles]		ALTURA [mm]		
48,01041554		430,2518234		

Figura 130 Posición y altitud calculados y presentados en la HMI

La programación completa del sistema de visión por computador se muestra en el Anexo B.

6.3.3 Implementación en software del HMI

El objetivo de la implementación de una interfaz gráfica se debe a que en esta se puede monitorear y controlar al sistema en general. Específicamente se logra elegir la conexión de la PC con la aeronave ya sea por bluetooth o por radio frecuencia, además esta interfaz permite controlar al cuatrirotor de manera manual con la utilización de botones y de manera automática ingresando un punto inicial y un punto final, generando su trayectoria y permitiendo que la aeronave de forma mecánica la siga.

Esta interfaz debe ser “amigable” con el usuario, permitiendo acceder al mismo a todas sus opciones disponibles, el diseño de la misma está basada en la guía ergonómica de diseño de interfaces de supervisión (GEDIS), la cual orienta el diseño de interfaces para entornos industriales y apuntando principalmente a la ergonomía del usuario.

La guía nos presenta argumentos de diseño para la arquitectura y navegación de la interfaz, de igual manera nos cita conceptos de distribución de elementos en la pantalla y una descripción detallada de parámetros como texto, imágenes, color, tabla, encabezados, pies de pantalla, etc. Esta interfaz se diseñó en base a elementos disponibles en el software Qt Creator en Linux y utilizando lenguaje de programación Python.

- **Arquitectura**

La arquitectura de la HMI nos presenta las diferentes pantallas disponibles con las que contara el usuario para su posterior interacción. La Figura 131 muestra la arquitectura de la interfaz.

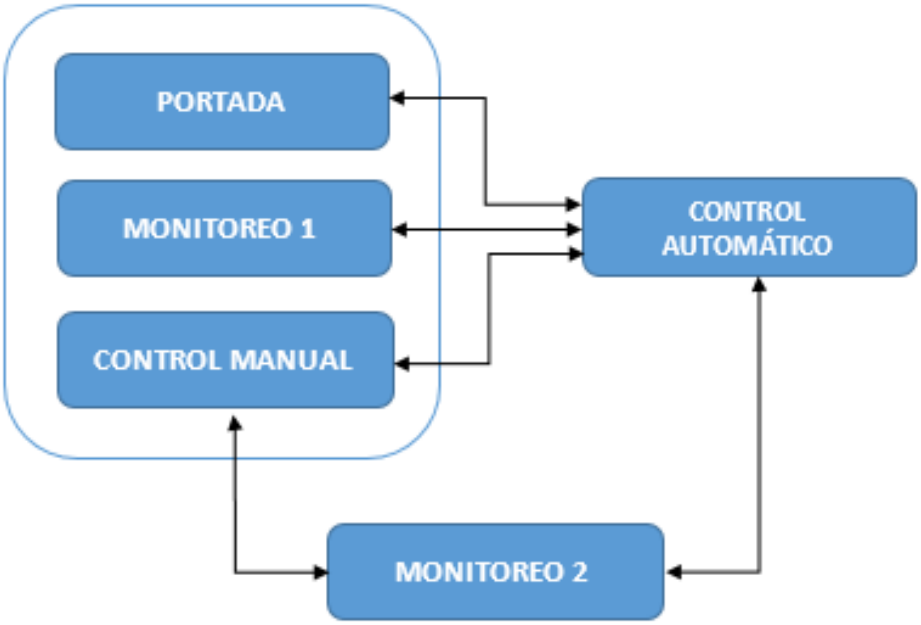


Figura 131 Arquitectura HMI

- **Navegación**

Definida la arquitectura, se deriva a mostrar como el usuario navegara dentro de la interfaz. Las pantallas definidas en la arquitectura se describirán a continuación:

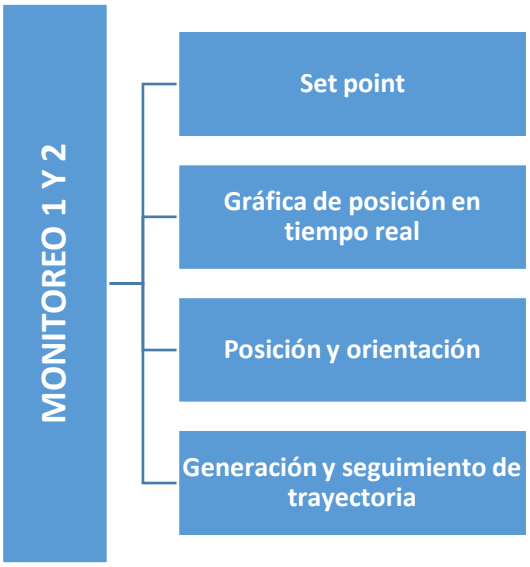


Figura 132 Navegación sección Monitoreo

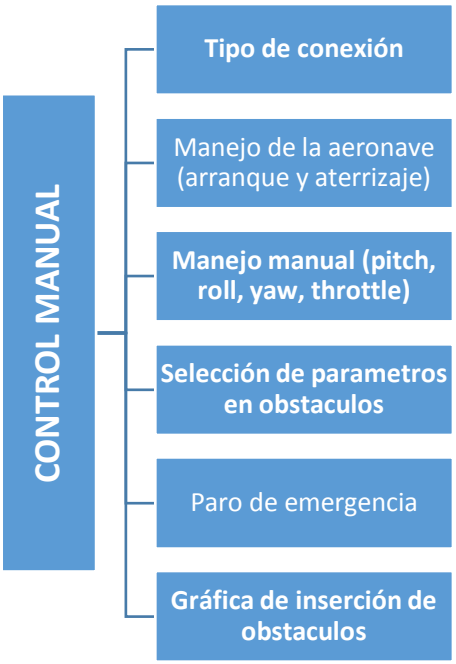


Figura 133 Navegación sección Control Manual

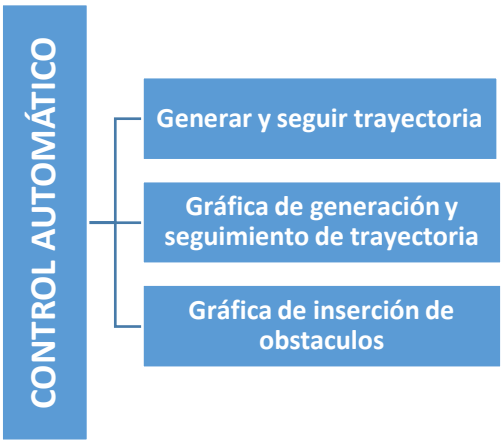


Figura 134 Navegación sección Control Automático

Una vez conocida la arquitectura y navegación en cada sección de la HMI y considerando los parámetros de diseño establecidos en la guía GEDIS. Se procedió a implementar cada pantalla indicada en la arquitectura, en donde la primera pantalla incluye la portada, la sección de monitoreo y control manual. Mientras que para la segunda pantalla se presenta la sección de control automático.

Para la primera pantalla se consideró parámetros en cuanto a la posición de imágenes, colores en texto y elementos. Resumiendo se tiene como fondo de pantalla el color gris y texto en color negro. Ver Figura 135.

Figura 135 HMI sección portada, monitoreo 1 y control manual

La primera pantalla muestra la opción de definir el tipo de conexión a realizar, en este caso sitúa el tipo Bluetooth y tipo Radio Frecuencia, además dispone de visualización de datos como posición y orientación de la aeronave. Así mismo se puede seleccionar el tipo de obstáculo digital a colocar y su tamaño. La sub sección de control manual permite el arranque y aterrizaje del dispositivo así como su manejo en pitch, roll, yaw, throttle y un paro de emergencia. Además de visualizar en forma textual en la pantalla de actividades o Log las acciones realizadas por el sistema.

En la segunda pantalla se permite la inserción de punto inicial, final, obstáculos digitales y la generación de trayectoria mediante un árbol de ramificaciones implementado en base a un algoritmo RRT. Ver Figura 136.

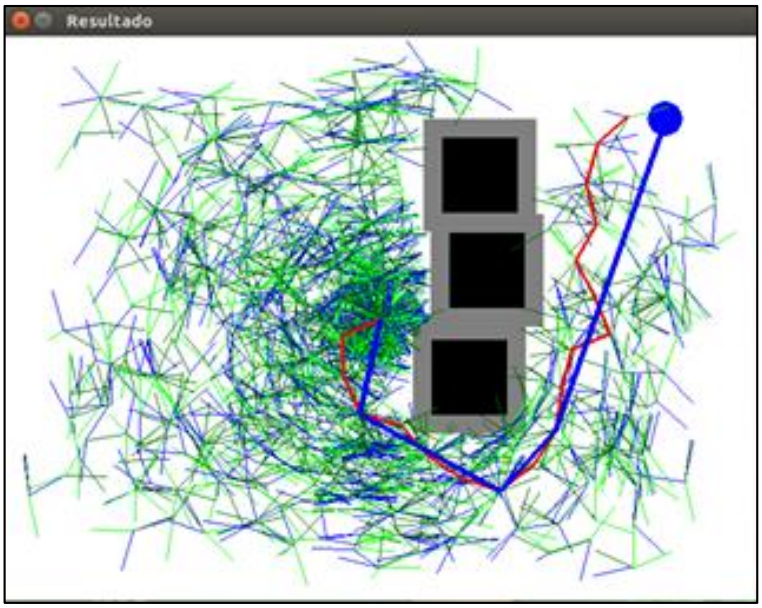


Figura 136 HMI sección control automático

Mientras que en la tercera pantalla (monitoreo 2) se permite realizar la supervisión en tiempo real la aeronave en un espacio en tres dimensiones, así mismo la inserción de obstáculos, la generación y seguimiento de trayectoria. Ver Figura 137.

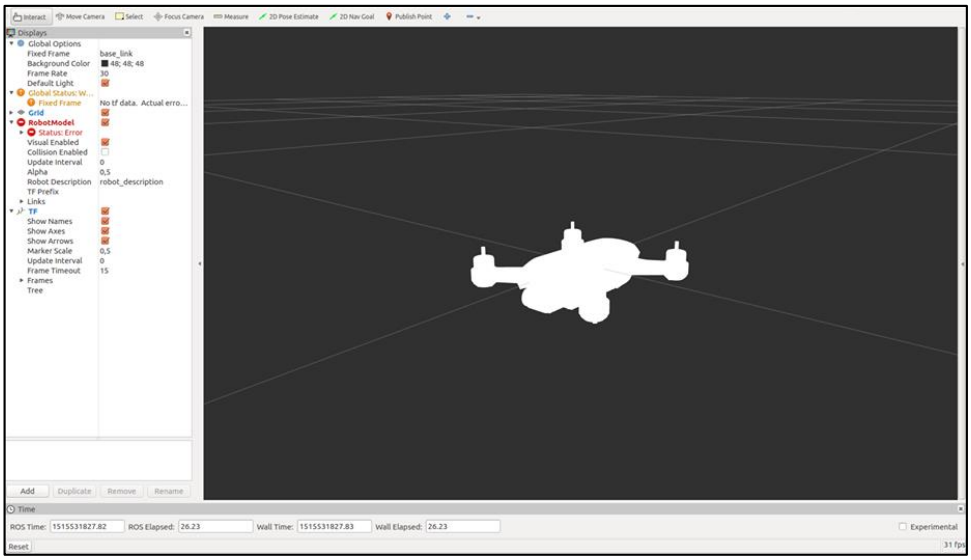


Figura 137 HMI sección monitoreo 2

La programación implementada para la construcción de la HMI se presenta en el Anexo C.

6.3.4 Implementación de algoritmo de planificación de trayectoria RRT

6.3.3.1 Algoritmo RRT

El algoritmo RRT en este proyecto tiene como parámetros de cálculo el punto inicial, punto final y posición y tamaño de los obstáculos. Este genera un árbol de ramificaciones desde el punto de inicio al punto final, que permite obtener una trayectoria evadiendo los obstáculos ingresados. En la Figura 138 se muestra un ejemplo de generación de trayectoria, haciendo referencia a la curva de color rojo.

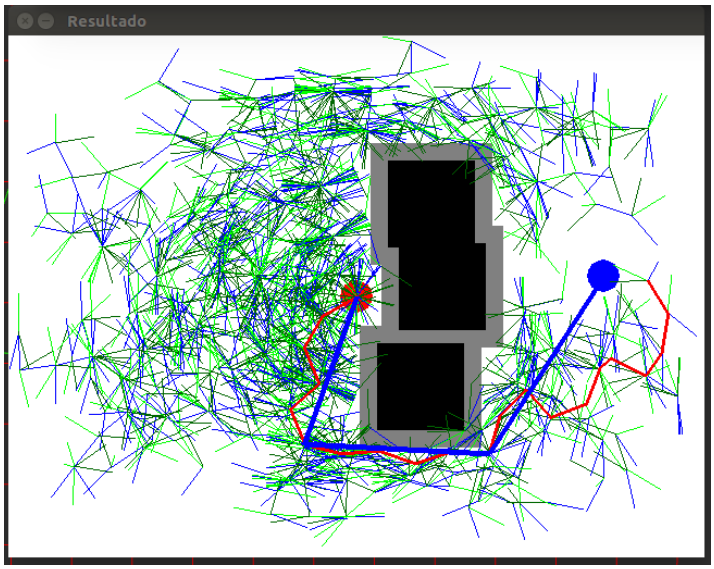


Figura 138 Ejemplo de generación de trayectoria por algoritmo RRT

Este algoritmo devuelve al sistema una matriz de puntos por los cuales se guiara la aeronave para realizar el seguimiento de la trayectoria generada. La Figura 139 presenta los puntos remitidos al sistema de parte del algoritmo RRT.

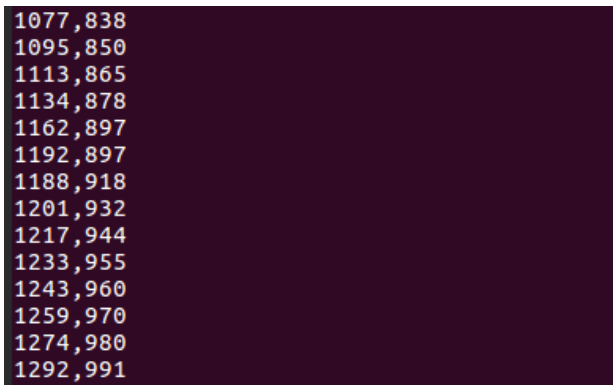


Figura 139 Puntos remitidos al sistema de parte del algoritmo RRT

6.3.3.2 Optimización de trayectoria algoritmo RRT

Para la optimización de trayectoria es necesario disponer de la matriz de puntos obtenidos a través del algoritmo RRT y mediante un algoritmo se simplifica la trayectoria inicial. En la Figura 140 se observa la aplicación del algoritmo de optimización de trayectoria, haciendo referencia a la curva azul.

Figura 140 Optimización de trayectoria

En la gráfica muestra como luego de ejecutarse el algoritmo RRT estándar, se procede a calcular una trayectoria optimizada reduciendo la cantidad de nodos y ajustándose a la trayectoria original.

Los códigos de programación para el algoritmo RRT y su optimización se muestran en el Anexo C.

6.4 Implementación de la plataforma de soporte ROS

Durante el funcionamiento del sistema completo, se ejecutan simultáneamente varios programas, para lo cual se implementó una plataforma de soporte de ROS para la comunicación entre ellos estableciendo nodos, publicadores, suscriptores y tópicos en cada uno de los algoritmos de modo que el coste computacional sea el menor y la ejecución de los programas se realice de forma paralela, permitiendo que el sistema funcione de manera eficiente.

La plataforma de ROS implementada se puede visualizar en la Figura 141, donde se especifican los nodos y tópicos creados para la comunicación, así mismo se puede observar la relación entre publicadores y suscriptores a través de las flechas observadas en la figura mencionada.

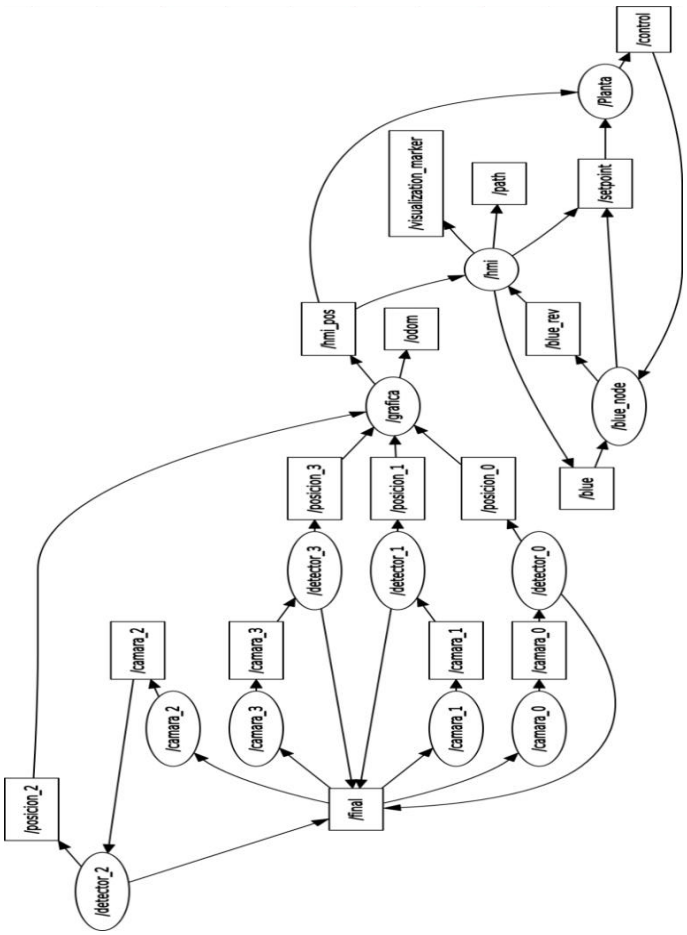


Figura 141 Plataforma de comunicación ROS implementada

El algoritmo de programación para la inicialización de nodos de ROS se presenta en el Anexo C.

6.5 Implementación del sistema de control

La posición del drone en sus coordenadas X e Y, serán las variables de control en este sistema, para esto se implementara un control PID sobre la ejecución de movimiento en pitch y roll, los dispositivos encargados de realizar la retroalimentación al sistema serán las cámaras colocadas en el bastidor.

La programación respectiva para cada movimiento se presenta en el Anexo D conjuntamente con su planta.

6.4.1 Implementación de controladores PID

- **Controlador PID movimientos pitch y roll**

Los valores de las constantes calculadas en el anterior capitulo se implementaron en forma de algoritmo sobre el programa que gobierna las plantas de pitch y roll de la aeronave. Con los parámetros obtenidos se generó un comportamiento de ambas plantas ante un set point de posición (1000,1000) mm como se muestra en la Figura 143.

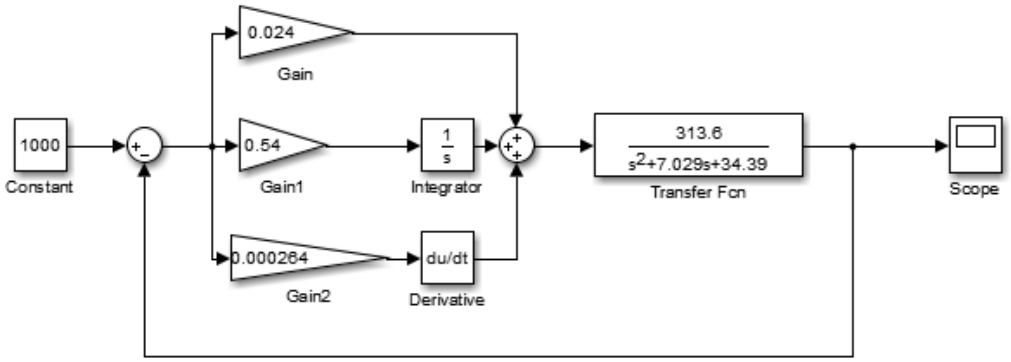


Figura 142 Planta en lazo cerrado implementado un controlador PID con primeras constantes

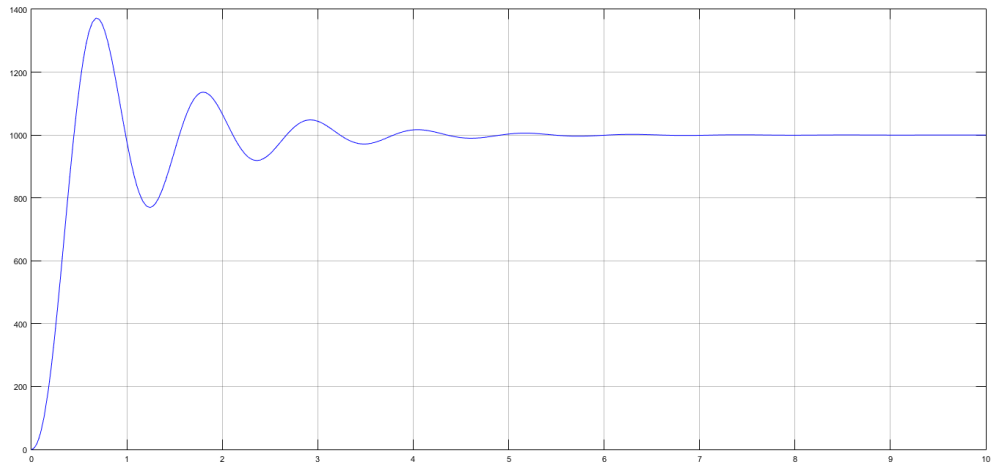


Figura 143 Comportamiento de la planta ante entrada escalón primeras constantes

De acuerdo a la gráfica anterior y observando la actuación real de la aeronave se decidió sintonizar de forma manual los valores de las constantes anteriormente calculadas, debido a que los valores hallados generan oscilaciones para llegar al

punto deseado acumulando un error en el seguimiento de la trayectoria. Por tal razón se ajustó los siguientes valores del controlador:

- Kp = 0.0559333
- Ki = 0.00002559
- Kd = 0.056

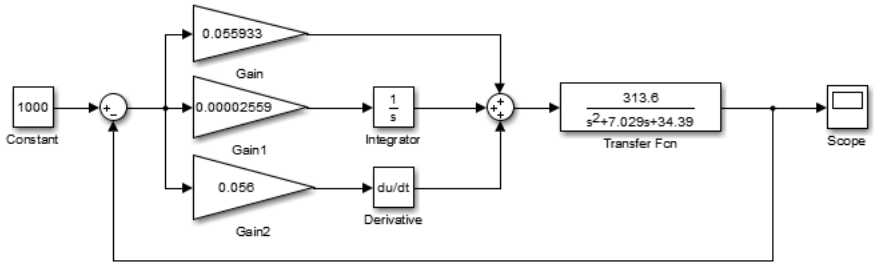


Figura 144 Planta en lazo cerrado implementado un controlador PID con constantes definitivas

Con los valores descritos se logra un comportamiento de estabilización en un punto menor al deseado, ayudando a la aeronave a desacelerarse mientras se traslada de un punto a otro, sin tener el problema de generación de oscilaciones y permitiendo que llegue al siguiente punto sin inconvenientes. El comportamiento de la aeronave se muestra en la Figura 145.

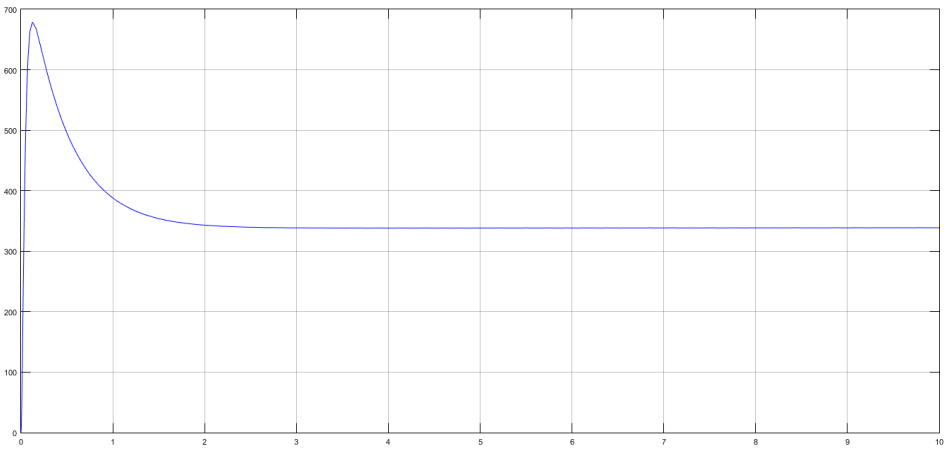


Figura 145 Comportamiento de la planta ante entrada escalón constantes definitivas

CAPÍTULO VII

PRUEBAS Y RESULTADOS

Para el proceso de pruebas y resultados se colocó en funcionamiento el sistema completo, además se utilizó componentes externos como elementos de soporte para la correcta ejecución y toma de datos en este capítulo.

7.1 Pruebas y resultados posicionamiento (X e Y) y altitud (Z)

- **Pruebas:**

Por medio de medición experimental utilizando un flexómetro y el software implementado en el proyecto (HMI) se obtuvo los siguientes datos en cuanto a posición (X e Y) y altitud (Z).

Para el caso de medición experimental se implementó un arreglo de marcas sobre el suelo las cuales están distanciadas 200 mm entre cada una en el eje de las abscisas y 15 mm en el eje de las ordenadas para determinar la posición real, mientras que para la altitud se estableció valores de comparación utilizando elementos que eleven la aeronave.

Prueba 1

Posición real

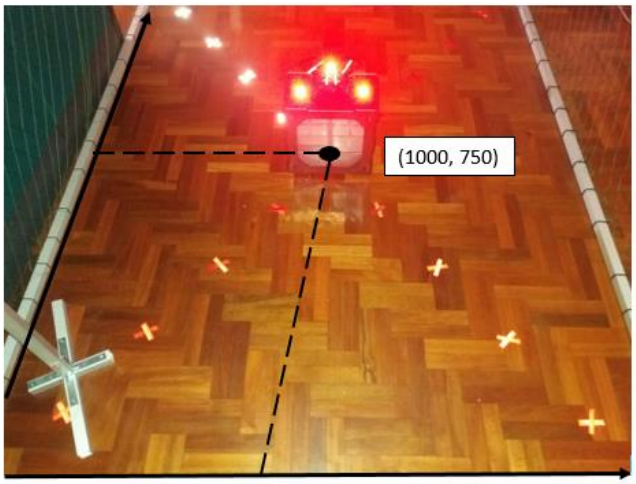


Figura 146 Prueba 1 posicionamiento real, coordenadas X y Y

Altitud real

Figura 147 Prueba 1 altitud real, coordenada Z

En la prueba 1 se posiciono la aeronave en las coordenadas (1000 mm, 750 mm) y en un valor de altitud de 250 mm, para su posterior comparación con los valores generados por el algoritmo de cálculo.

Posición y altitud HMI

Figura 148 Prueba 1 posicionamiento y altitud calculadas, coordenadas X, Y y Z

Prueba 2

Posición real

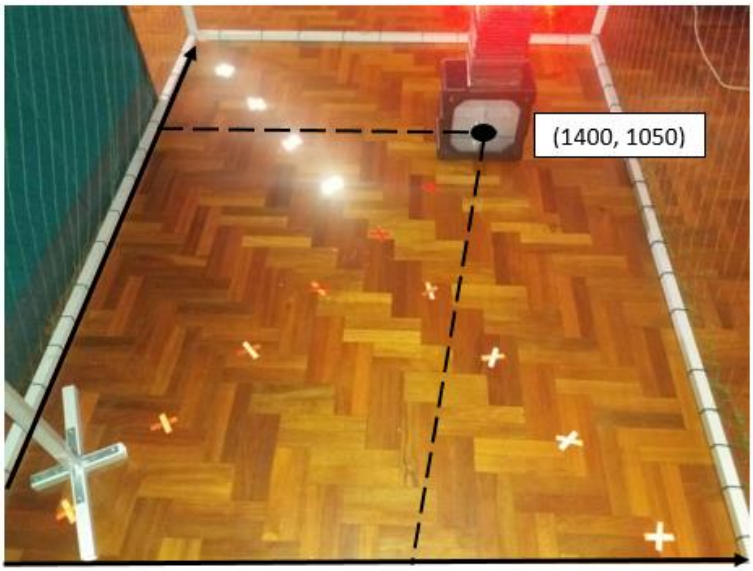


Figura 149 Prueba 2 posicionamiento real, coordenadas X y Y

Altitud real

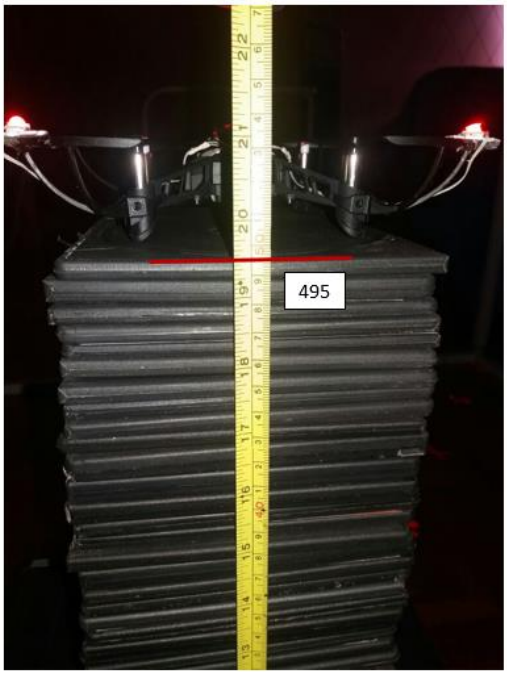


Figura 150 Prueba 2 altitud real, coordenada Z

En la prueba 2 se posiciono la aeronave en las coordenadas (1400 mm, 1050 mm) y en un valor de altitud de 495 mm, para su posterior comparación con los valores generados por el algoritmo de cálculo.

Posición y altitud HMI

Figura 151 Prueba 2 posicionamiento y altitud calculadas, coordenadas X, Y y Z

- **Resultados:**

En la Tabla 24 se detalla las pruebas realizadas de posicionamiento como valor real a través de medidas físicas y valor calculado a través del algoritmo de visión por computador

Tabla 24

Resultados pruebas de posicionamiento coordenadas X, Y y Z

Prueba	Valor real [mm]			Valor calculado [mm]		
	X	Y	Z	X	Y	Z
1	200	150	100	170	166	150
2	200	1350	200	210	1350	220
3	600	450	300	580	450	305
4	600	1050	400	590	1060	400
5	1000	750	500	1016	740	490
6	1200	600	650	1200	594	630
7	1200	900	1050	1202	904	1030
8	1600	300	1100	1610	280	1090
9	1600	1200	1150	1629	1240	1140
10	1800	1350	1200	1810	1350	1150

7.2 Pruebas y resultados costo computacional en cálculo de trayectoria

7.2.1 Pruebas trayectoria RRT estándar

La Tabla 25 enlista el tiempo en relación al coste computacional para la ejecución del algoritmo RRT estándar y la cantidad de nodos o puntos creados.

Tabla 25
Tiempos de ejecución y número de nodos algoritmo RRT estándar

Prueba	Tiempo de ejecución [ms]	Número de nodos
1	1,51665E+12	23
2	1,51665E+12	22
3	1,51665E+12	30

7.2.2 Pruebas trayectoria RRT optimizada

La Tabla 26 enlista el tiempo en relación al coste computacional para la ejecución del algoritmo RRT optimizado y la cantidad de nodos o puntos creados.

Tabla 26
Tiempos de ejecución y número de nodos algoritmo RRT optimizado

Prueba	Tiempo de ejecución [ms]	Número de nodos
1	1,51665E+12	6
2	1,51665E+12	4
3	1,51665E+12	7

7.2.3 Resultados trayectoria RRT estándar y trayectoria RRT optimizada

En la Tabla 27 se presenta la diferencia de tiempo de ejecución entre algoritmos RRT estándar y RRT optimizado, así mismo parámetros de conversión para la posterior comparativa de los algoritmos.

Tabla 27
Tiempo computacional de ejecución de algoritmos

Costo computacional RRT estándar vs RRT optimizada [ms]	Distancia entre nodos [pixeles]	Factor de conversión entre pixeles y mm	Distancia entre nodos [mm]
1	35	3,125	109,375
1	35	3,125	109,375
1	35	3,125	109,375

La Tabla 28 muestra la comparativa entre los tiempo de ejecución real de la trayectoria generada por el algoritmo RRT estándar versus el algoritmo RRT optimizado a una velocidad constante de 100 mm/seg. De igual forma presenta el porcentaje de optimización en relación de los algoritmos mencionados.

Tabla 28

Comparativa de eficiencia entre algoritmos implementados

Distancia RRT estándar [mm]	Distancia RRT optimizado [mm]	Velocidad de prueba [mm/s]	Tiempo RRT estándar [s]	Tiempo RRT optimizado [s]	Porcentaje de optimización
2515,625	656,25	100	25,15625	6,5625	73,91304348
2406,25	437,5	100	24,0625	4,375	81,81818182
3281,25	765,625	100	32,8125	7,65625	76,66666667

7.3 Pruebas y resultados seguimiento de trayectoria

Esta prueba hace referencia a la eficiencia del controlador PID diseñado y a la capacidad de la aeronave para seguir la trayectoria generada. El error es calculado de acuerdo al área proyectada bajo la curva generada por el algoritmo RRT optimizado versus la curva generada por el movimiento del cuatrirotor, de la misma manera se obtendrá el error punto a punto de estas trayectorias.

Para esto se adquirió todos puntos tanto de la curva real y de la trayectoria de la aeronave, de modo que se pueda ajustar una ecuación a cada una de estas permitiendo obtener el área proyectada a través de su integral. Cada una de las pruebas se basó en el seguimiento de una trayectoria rectilínea. Las Figuras 152 a 163 muestran las pruebas realizadas de seguimiento de trayectoria.

- **Pruebas:**

Prueba 1

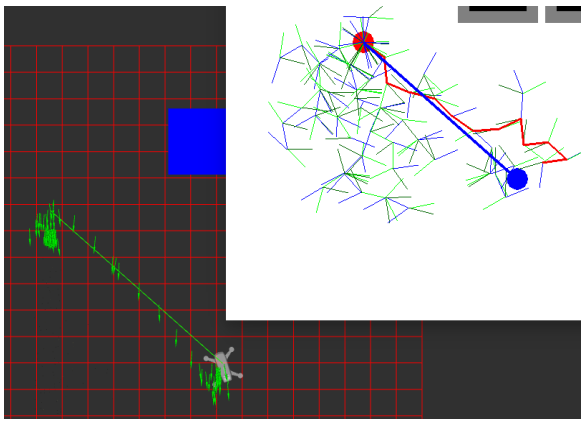


Figura 152 Prueba 1 seguimiento de trayectoria HMI

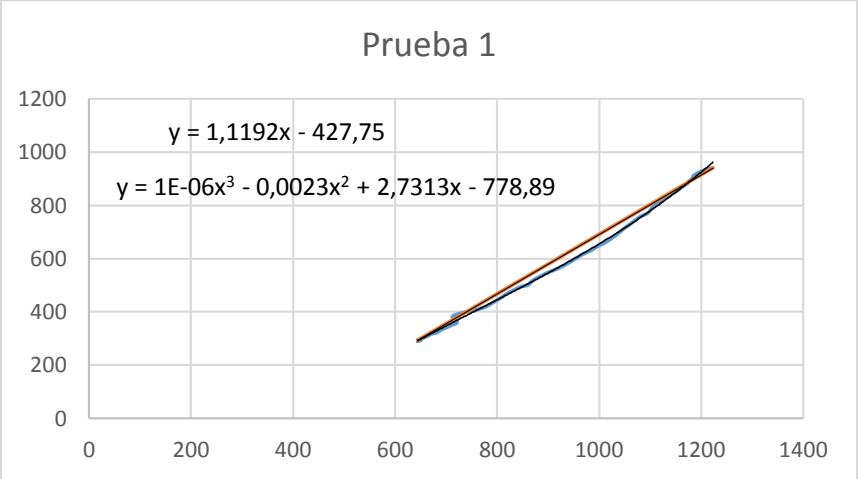


Figura 153 Prueba 1 ajuste de curvas

Prueba 2

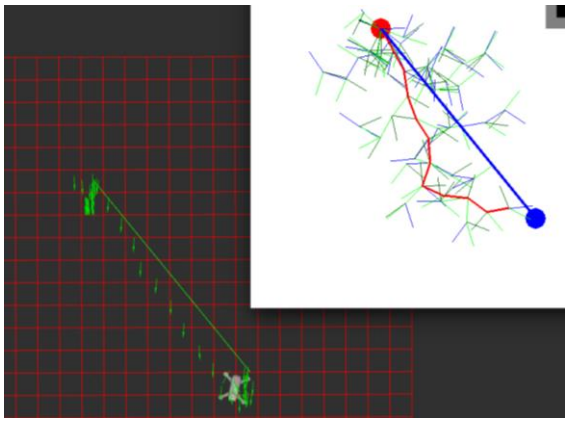


Figura 154 Prueba 2 seguimiento de trayectoria HMI

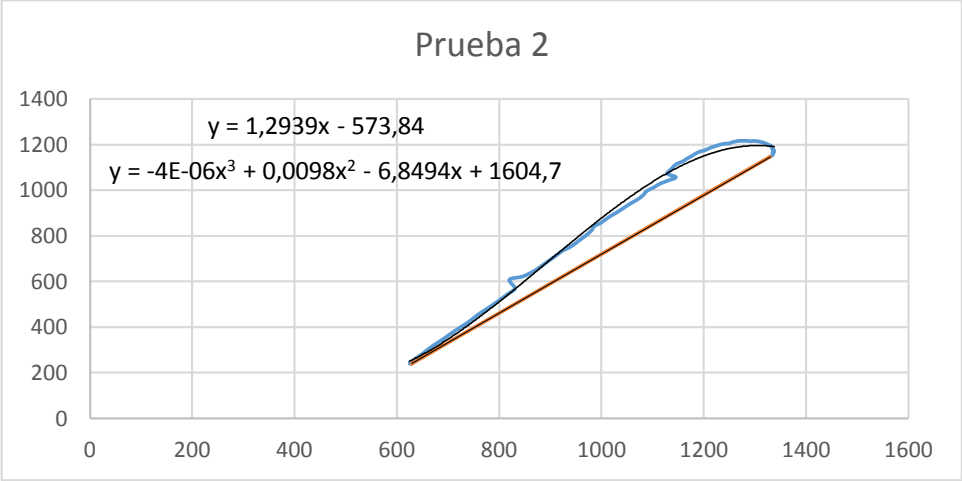


Figura 155 Prueba 2 ajuste de curvas

Prueba 3

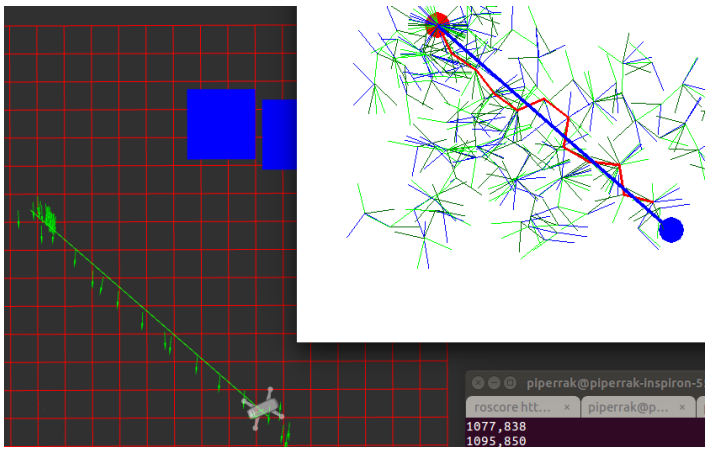


Figura 156 Prueba 3 seguimiento de trayectoria HMI

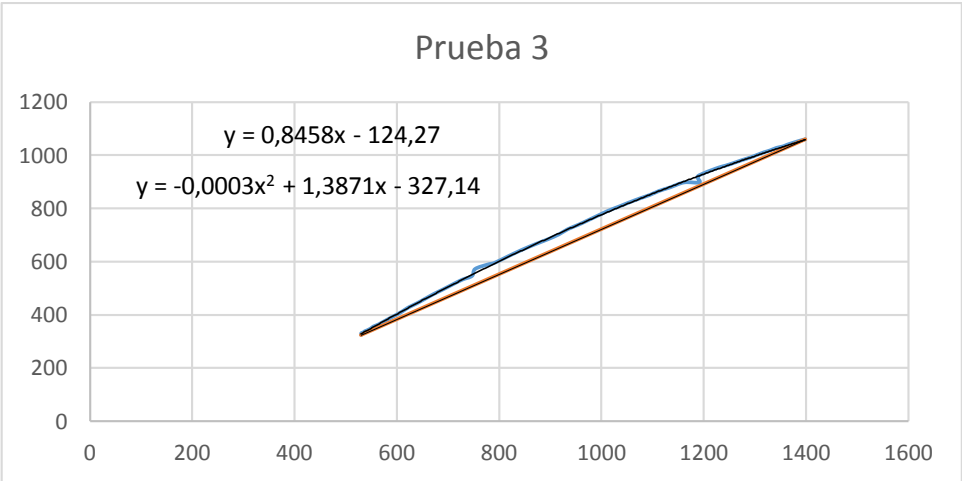


Figura 157 Prueba 3 ajuste de curvas

Prueba 4

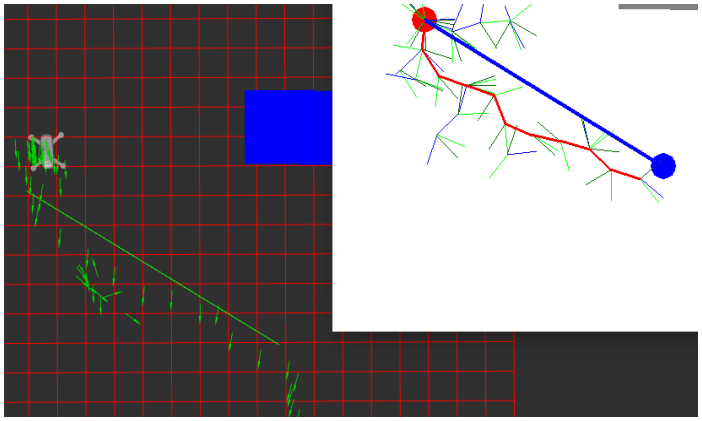


Figura 158 Prueba 4 seguimiento de trayectoria HMI

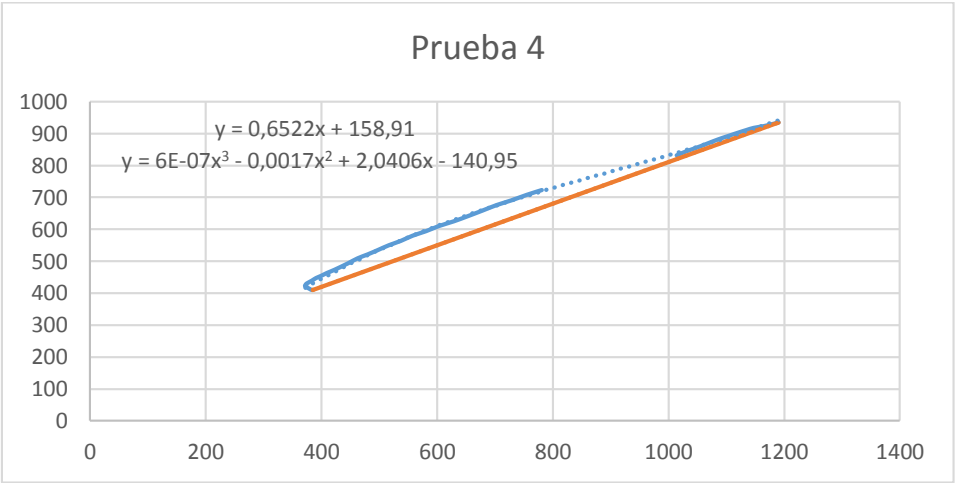


Figura 159 Prueba 4 ajuste de curvas

Prueba 5

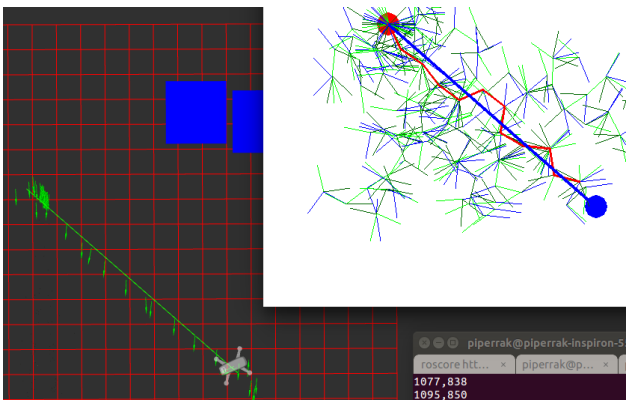


Figura 160 Prueba 5 seguimiento de trayectoria HMI

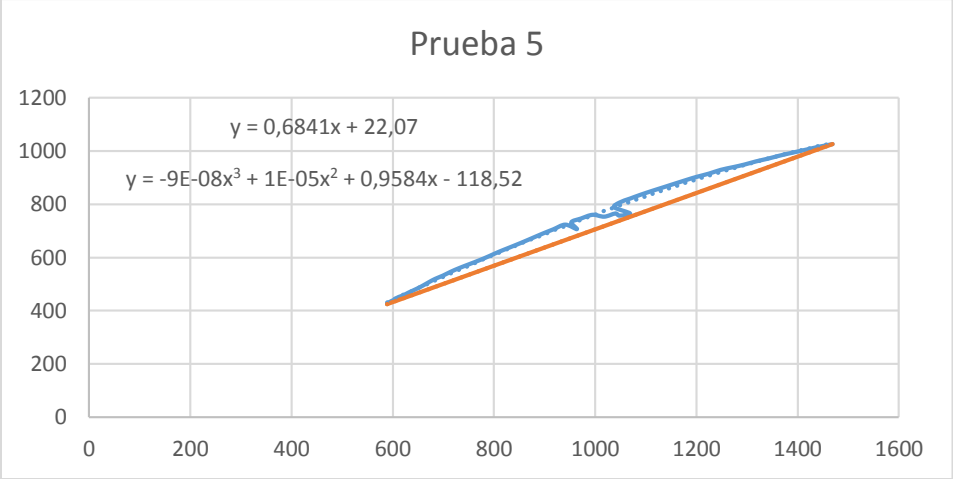


Figura 161 Prueba 5 ajuste de curvas

Prueba 6

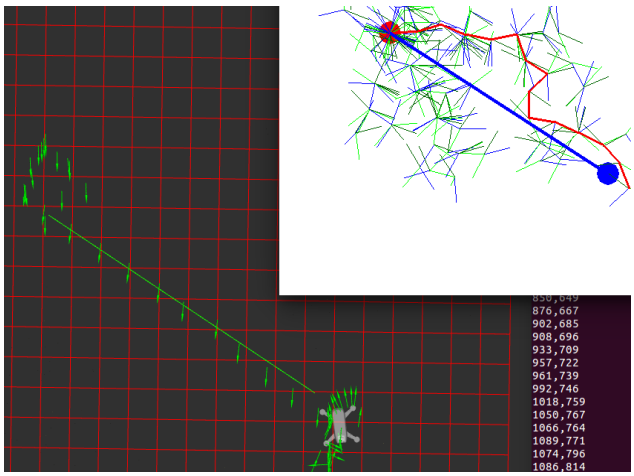


Figura 162 Prueba 6 seguimiento de trayectoria HMI

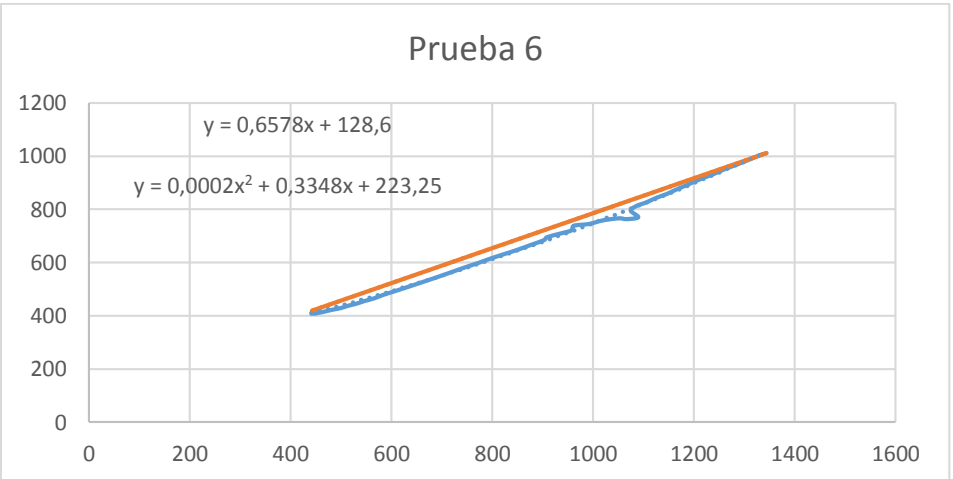


Figura 163 Prueba 6 ajuste de curvas

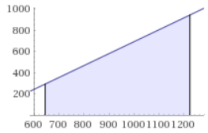
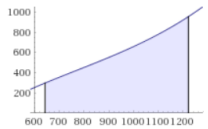
- **Resultados:**

En las Tablas 29 a 34 se detallan los cálculos de área de cada una de las curvas ajustadas anteriormente, así mismo el error en relación al área real versus el área deseada.

Prueba 1

Tabla 29

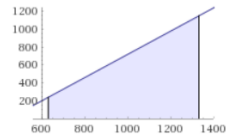
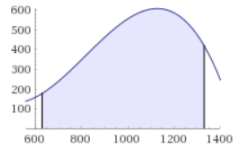
Prueba 1 seguimiento de trayectoria, error calculado

Curva	Imagen	Integral	Área [mm2]	Error Área real vs. Área deseada[%]
Deseada		$\int_{644}^{1223}(1.1192\,x-427.75)\,dx$	357256	3.88
Real		$\int_{644}^{1223}\left(1\times10^{-6}\,x^3-0.0023\,x^2+2.7313\,x-778.89\right)dx$	343904	

Prueba 2

Tabla 30

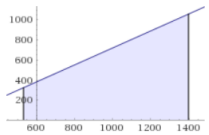
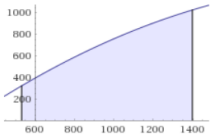
Prueba 2 seguimiento de trayectoria, error calculado

Curva	Imagen	Integral	Área [mm2]	Error Área real vs. Área deseada[%]
Deseada		$\int_{629}^{1330}(1.2939\,x-573.84)\,dx$	321763	33,81
Real		$\int_{629}^{1330}\left(-\left(4\times10^{-6}\right)x^3+0.0098\,x^2-6.8494\,x+1604.7\right)dx$	486168	

Prueba 3

Tabla 31

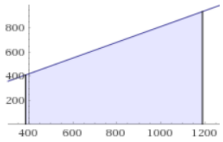
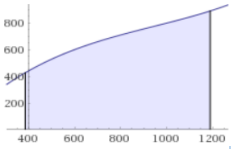
Prueba 3 seguimiento de trayectoria, error calculado

Curva	Imagen	Integral	Área [mm2]	Error Área real vs. Área deseada[%]
Deseada		$\int_{530}^{1399} (0.8458\,x - 124.27)\,dx$	619389	3,07
Real		$\int_{530}^{1399} (-0.0003\,x^2 + 1.3871\,x - 327.14)\,dx$	600917	

Prueba 4

Tabla 32

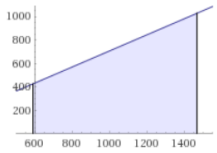
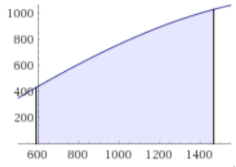
Prueba 4 seguimiento de trayectoria, error calculado

Curva	Imagen	Integral	Área [mm2]	Error Área real vs. Área deseada[%]
Deseada		$\int_{385}^{1190} (0.6522\,x + 158.91)\,dx$	541377	2,46
Real		$\int_{385}^{1190} (6\times10^{-7}\,x^3 - 0.0017\,x^2 + 2.0406\,x - 140.95)\,dx$	555068	

Prueba 5

Tabla 33

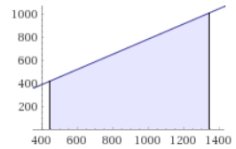
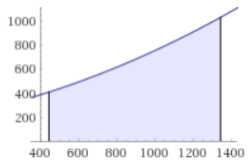
Prueba 5 seguimiento de trayectoria, error calculado

Curva	Imagen	Integral	Área [mm2]	Error Área real vs. Área deseada[%]
Deseada		$\int_{589}^{1469} (0.6841\,x + 22.07)\,dx$	671369	5,08
Real		$\int_{589}^{1469} \left(-(9\times10^{-8})\,x^3 + 1\times10^{-5}\,x^2 + 0.9584\,x - 118.52\right)dx$	638888	

Prueba 6

Tabla 34

Prueba 6 seguimiento de trayectoria, error calculado

Curva	Imagen	Integral	Área [mm2]	Error Área real vs. Área deseada[%]
Deseada		$\int_{443}^{1343} (0.6578\,x + 128.6)\,dx$	625695	2,90
Real		$\int_{443}^{1343} (0.0002\,x^2 + 0.3348\,x + 223.25)\,dx$	644414	

Los errores obtenidos en base a la comparación punto a punto entre la curva real y la curva deseada se detallan en la Tabla 35, donde se presenta el promedio de entre los errores hallados.

Tabla 35

Errores promedio punto a punto para seguimiento de trayectoria

Prueba	Error Promedio Punto a Punto X [%]	Error Promedio Punto a Punto Y [%]
1	1.61	2.80
2	5.60	11.00
3	3.45	4.43
4	5.23	3.99
5	4.66	4.81
6	5.08	3.68

7.4 Interpretación de resultados

- **Posicionamiento (X e Y) y altitud (Z)**

Tabla 36

Cálculo de error en posicionamiento valor real vs valor calculado

Prueba	Valor real [mm]			Valor calculado [mm]			Error [%]		
	X	Y	Z	X	Y	Z	X	Y	Z
1	200	150	100	170	166	150	15,00	10,67	15,00
2	200	1350	200	210	1350	220	5,00	0,00	7,50
3	600	450	300	580	450	305	3,33	0,00	1,67
4	600	1050	400	590	1060	400	1,67	0,95	0,00
5	1000	750	500	1016	740	490	1,60	1,33	2,00
6	1200	600	650	1200	594	630	0,00	1,00	3,08
7	1200	900	1050	1202	904	1030	0,17	0,44	1,90
8	1600	300	1100	1610	280	1090	0,63	6,67	0,91
9	1600	1200	1150	1629	1240	1140	1,81	3,33	0,87
10	1800	1350	1200	1810	1350	1150	0,56	0,00	4,17

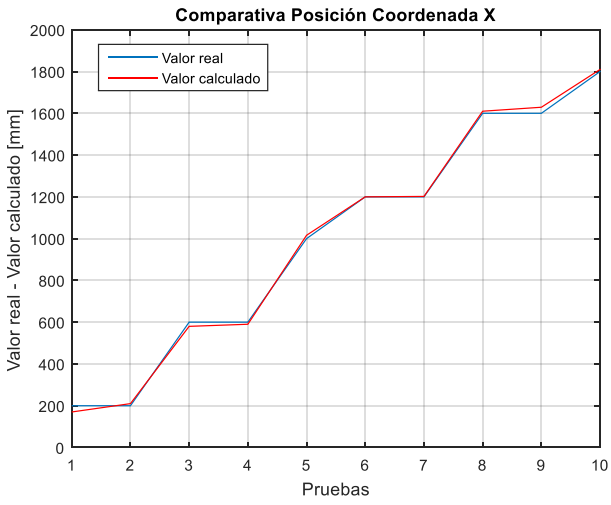


Figura 164 Valores reales vs valores calculados posición en X

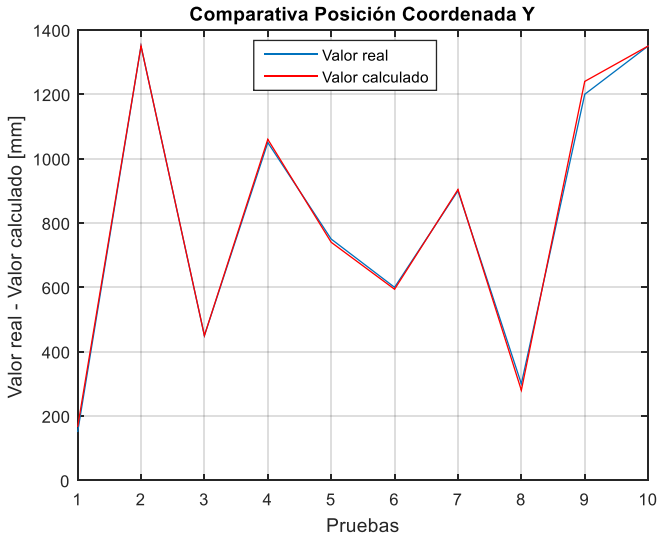


Figura 165 Valores reales vs valores calculados posición en Y

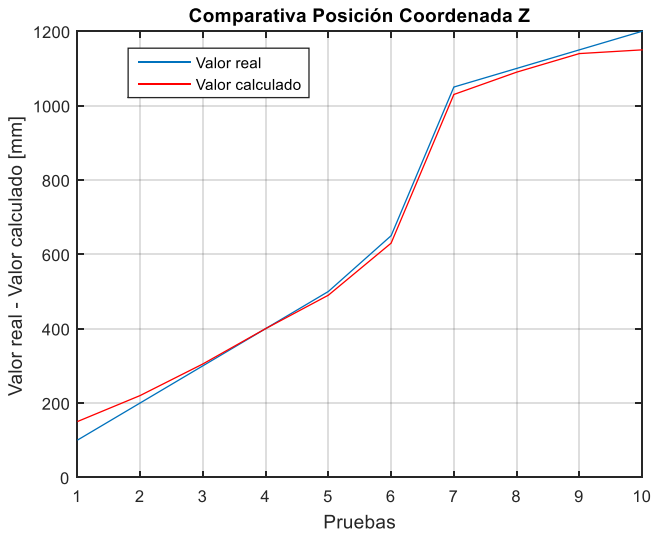


Figura 166 Valores reales vs valores calculados posición en Z

Tabla 37

Resumen de error promedio entre valores reales vs valores calculados

Prueba	Error promedio [%]
Posición coordenada X	2.98
Posición coordenada Y	2.44
Posición coordenada Z	3.71

En la Tabla 37 se presenta el error promedio de posicionamiento en cada una de las coordenadas X, Y y Z. Observando que estos errores no sobrepasa el 5 % de error máximo, aceptando los valores calculados a través del algoritmo de visión por computador.

- **Coste computacional en cálculo de trayectoria**

La Tabla 38 mostrada a continuación enlista los porcentajes de optimización en relación al coste computacional de ejecución del algoritmo RRT estándar versus el algoritmo RRT optimizado, dando como resultado que al optimizar la generación de trayectoria el tiempo de ejecución de su algoritmo es significativamente bajo y su eficiencia se eleva un 75,3 %.

Tabla 38
Porcentaje de coste computacional algoritmo RRT estándar vs algoritmo RRT optimizado

Prueba	Porcentaje de optimización [%]
1	73,91
2	81,81
3	76,66
Promedio	75,3

- **Seguimiento de trayectoria**

En la Tabla 39 se muestra los resultados de error calculados de acuerdo al área generada por la curva real y la curva calculada a través del algoritmo RRT optimizado, en la cual se puede observar que el error promedio no supera el 5 % de error máximo, por lo cual se puede afirmar que la eficiencia de seguimiento de trayectoria dada por la implementación del controlador PID es la adecuada para este proyecto.

Tabla 39
Resumen de error de seguimiento de trayectoria

Prueba	Error [%]
1	3.88
2	33.81
3	3.07
4	2.46
5	5.08
6	2.90
Error promedio [%]	3.48

En la Tabla 40 se presenta el cálculo del error promedio de los errores promedio punto a punto obtenidos de la comparación entre la curva real versus la curva deseada para seguimiento de trayectoria.

Tabla 40
Cálculo de error promedio general punto a punto para seguimiento de trayectoria

Prueba	Error Promedio Punto a Punto X [%]	Error Promedio Punto a Punto Y [%]
1	1.61	2.80
2	5.60	11.00
3	3.45	4.43
4	5.23	3.99
5	4.66	4.81
6	5.08	3.68
Error promedio [%]	4.27	5.11

Se puede notar que los errores promedio obtenidos son 4.27% y 5.11% en X e Y respectivamente, pero al ser valores obtenidos en base a la comparación de datos con unidades en milímetros estos errores son significativamente bajos por la unidad utilizada en el cálculo.

CAPÍTULO VIII

CONCLUSIONES, RECOMENDACIONES Y TRABAJOS FUTUROS

8.1 Conclusiones

- Se diseñó e implemento un sistema de visión por computador para la planeación de trayectoria de un vehículo aéreo no tripulado para entornos cerrados utilizando la plataforma de servicio de ROS. El sistema de visión por computador es capaz de realizar la detección y cálculo de posición y orientación de la aeronave para la posterior utilización de estos datos en la planificación de una trayectoria optima que permita el movimiento y evasión de obstáculos del cuatrirotor a través del espacio de trabajo construido. Adicionalmente el sistema cuenta con una HMI de supervisión y monitoreo del proceso, en la cual se puede controlar la aeronave de forma manual y automática a través de la inserción de puntos de inicio y llegada de la misma, así mismo como la implantación de obstáculos virtuales de diferentes formas y tamaños.

- Se diseñó y construyó un prototipo de vehículo aéreo no tripulado, de modo que pueda utilizarse este en proyectos futuros de mejoramiento y optimización del actual trabajo de titulación.

- Se diseñó e implemento un circuito de comunicación entre el sistema de visión por computador y el prototipo de vehículo aéreo no tripulado, de tal manera que pueda transferirse información entre ambas partes a través de comunicación inalámbrica a 2.4 GHz, permitiendo así una mayor velocidad de funcionamiento, elevando la eficiencia de modulación y obteniendo la capacidad anti-interferencia para posteriores trabajos futuros en base a este proyecto.

- Se construyó e implemento marcadores activos que fueron colocados en la aeronave de modo que pueda facilitarse la detección de la misma por parte del sistema de visión por computador.

- Se diseñó, construyo e implemento filtros físicos en cada una de las cámaras, que permitan la discriminación de luces externas al proyecto, de tal modo que pueda detectarse únicamente la aeronave.

- Se diseñó y construyó un bastidor de dimensiones 2.0x1.50x1.70 metros, facilitando la colocación de cada una de las cámaras de tal manera que estas puedan captar totalmente el espacio de trabajo construido.

- Se estimó la posición de la aeronave a través de la aplicación de filtros digitales de visión por computador y un algoritmo de cálculo de posición diseñado experimentalmente realizado en lenguaje de programación Python.

- Se implementó y optimizo un algoritmo de generación de trayectoria, el cual genera una curva de puntos por la cual se desplaza la aeronave regida por un controlador PID, permitiendo la evasión de obstáculos virtuales introducidos a través de la HMI implementada, realizado en lenguaje de programación Python.

- Se implementó el uso de una HMI para el control y monitoreo de las variables de control que intervienen en el proceso, la cual mejora la interacción entre el usuario y el sistema en general.

- Se aprobó el funcionamiento del sistema en general por medio de pruebas como: validación de estimación de posición, seguimiento de trayectoria y coste computacional.

- Los errores calculados en cuanto a validación de posición y seguimiento de trayectoria son 3.04% y 3.48% respectivamente, cabe mencionar que estos hacen referencia a errores en milímetros por lo que son significativamente bajos. Mientras que el porcentaje de mejoramiento entre el algoritmo de trayectoria original versus el optimado alcanza un valor de 75.3% corrigiendo notablemente la trayectoria que sigue la aeronave.

8.2 Recomendaciones

- Se recomienda para este tipo de proyectos utilizar cuatrirotores que dispongan en su tarjeta controladora de vuelo un software de estabilización, debido que al implementar cualquier metodología de control la sustentación de la aeronave debe ser perfecta sino se tendrá inconvenientes al momento de realizar el seguimiento de trayectoria.

- Para este tipo de aplicación se recomienda utilizar cámaras afines al proyecto, con esto se reducirá la modificación y utilización de elementos y dispositivos externos a las mismas. De igual manera para la conexión de diferentes cámaras es necesario adquirir equipos que permiten una conexión y transmisión de datos de forma multiplexada.

8.3 Trabajos Futuros

- Se propone como trabajo futuro la implementación de un algoritmo de generación de trayectoria en un espacio tridimensional en base al algoritmo actual realizado en espacio bidimensional, así mismo el seguimiento de dicha ruta por una aeronave similar. Este proyecto deberá contener estimación de posición de la aeronave e introducción de obstáculos reales en el espacio de trabajo y la ampliación del espacio de trabajo.

- Se plantea como trabajo futuro la implementación de un algoritmo de generación de trayectoria en tiempo real (modo online), de modo que el sistema pueda recalcular la trayectoria mientras la aeronave se encuentre en movimiento y se encuentre obstáculos imprevistos. Se requiere la ampliación del espacio de trabajo.

8.4 Bibliografía

- Abdou, I., & Pratt, W. (1979). Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors. IEEE.
- Aguilar, W. G., Casaliglla, V. P., & Pólit, J. L. (2017). Obstacle Avoidance Based-Visual Navigation for Micro Aerial Vehicles. Electronics.
- Akinlar, C., & Topal, C. (2012). EDPF: A Real-Time Parameter-Free Edge Segment Detector With a False Detection Control. Eskisehir, Turquia: International Journal of Pattern Recognition and Artificial Intelligence.
- Albornoz, M., & Calahorrano, D. (2016). Seguimiento de objetos basado en vision artificial para cuadrirrotor Parrot Ar.Drone 2.0. Quito.
- Arduino. (2017). Arduino Micro. Obtenido de https://store.arduino.cc/usa/arduino-micro
- Arduino. (2017). Arduino Nano. Obtenido de https://store.arduino.cc/usa/arduino-nano
- Barrientos, A., del Cerro, J., Gutierrez, P., San Martin, R., Martínez, A., & Rossi, C. (2007). Vehículos aéreos no tripulados para uso civil. Tecnología y aplicaciones. Madrid.
- Berni, J., Zarco-Tejada, P., Suarez, L., & Fereres, E. (2009). Thermal and narrow-band multispectral remote sensing for vegetation monitoring from an unmanned aerial vehicle.
- Bradski, G., & Kaebler, A. (2011). Computer Vision with the OpenCV Library. Sebastopol: O'Reilly.
- Budynas, R., & Nisbett, K. (2008). Constantes físicas de materiales. En Diseño en ingeniería mecánica de Shigley. Mexico DF: McGraw-Hill.
- Bueno Garcia, G., Suarez, O., Espinosa, J., Tercero, J., Serrano, I., & Vállez, N. (2015). Learning Image Processing with OpenCV. Packt Publishing Ltd.
- Castillo, P., Garcia, P., Lozano, R., & Albertos, P. (2007). Modelado y estabilización de un helicóptero con cuatro rotores. Revista Iberoamericana de Automática e Informática Industrial RIAI, 41-57.
- Castillo, P., Lozano, R., & Dzul, A. (2005). Modelling and Control of Mini-Flying Machines. Compiegne: Springer.

- Datasheet 1 W High Power LED. (s.f.). Obtenido de http://www.wayjun.com/Datasheet/Led/1W%20High%20Power%20LED.pdf
- Datasheet D-Sun 3A Step Down Switching Voltage. (s.f.). Obtenido de https://www.defiancerc.com/products/d-sun-3a-step-down-switching-voltage-regulator
- Deva Ramanan, X. (s.f.). Face Detection, Pose Estimation, and Landmark Localization in the Wild. Irvine.
- Dufournaud, Y., Schmid, C., & Horaud, R. (2000). Matching Images with Different Resolutions. Montnonnot, France.
- Fridrich, J., Soukal, D., & Lukas, J. (2003). "Detection of Copy-Move Forgery in Digital Images", Proc. Digital Forensic Research Workshop. Cleveland.
- FriendlyElec. (2017). NanoPi NEO Plus 2. Obtenido de http://www.friendlyarm.com/index.php?route=product/product&product_id=196
- Futaba Corporation. (2011). Futaba T6J Manual Book. Yabutsuka, Japón.
- Garcia, A. (2013). ROS: Robot Operating System. Cartagena.
- Gettinger, D., & Holland, A. (2015). Drone Sightings and Close Encounters. New York.
- Gómez-Bravo, D. L. (2006). Planificación de Trayectorias con el Algoritmo RRT. Aplicación a Robots No Holónomos. Valencia, España: Revista Iberoamericana de Automática e Informática Industrial. Obtenido de https://www.researchgate.net/publication/28141977_Planificacion_de_Trayectorias_con_el_Algoritmo_RRT_Aplicacion_a_Robots_No_Holonomos
- Grewal, M., Weill, L., & Andrews, A. (2007). Global Positioning Systems, Inertial Navigation, and Integration. John Wiley & Sons.
- Hull, D. (2007). Fundamentals of Airplane Flight Mechanics. Verlag Berlin Heidelberg: Springer.
- InvenSense Inc. (19 de 08 de 2013). MPU-6000 and MPU-6050 Product Specification Revision 3.4. Obtenido de https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
- Johnson, W. (1994). Helicopter Theory. New Jersey: Dover Publication Inc.
- Kuffner, J. C., & LaValle, S. M. (2000). RRT connect: An efficient approach to single-query path planning. IEEE Int'l Conf. on Robotics and Autom. Vol 10, No 5.

- Kumar, S., Gupta, R., Khana, N., Chaudhury, S., & Joshi, S. (2007). Text extraction and document image segmentation using matched wavelets and mrf model.
- Laumont, J. P., Jacobs, M. T., & Murray, M. (1994). A Motion Planner for Non-holonomic Mobile Robots. IEEE Trans. on Robotics and Autom., Vol 10, No 5.
- LaValle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning. Computer Science. Iowa State University.
- LaValle, S. M., & Kuffner, J. J. (2001). Rapidly-Exploring Random Trees: Progress and Prospects. Algorithmic and Computational Robotics: New Directions. B. R. Donald, K. M. Lynch, and D. Rus, editors.
- Li, Y. (2008). "Dinamics Modeling and Simulation of Flexible Airships" Department of Mechanical Engineering. Montreal: McGill University.
- López, D., Gómez-Bravo, F., Cuesta, F., & Ollero, A. (2006). Planificación de Trayectoria con el Algoritmo RRT. Aplicación a robots no holónomos.
- Lorsakul, A., & Suthakorn, J. (2007). Traffic Sign Recognition for Intelligent Vehicle/Driver Assistance System Using Neural Network on OpenCV. Thailand.
- Luukkonen, T. (2011). Modelling and control of quadcopter. Espoo: Aalto University.
- María, C., & Maldonado, D. (2014). Control por Visión de un cuadricoptero utilizando ROS. Quito.
- Marzal, A., & Gracia, I. (2003). Introducción a la programación con Python. Castellón de la Plana.
- Mejías, L. (2006). Control Visual de un Vehículo Aéreo Autónomo Usando Detección y Seguimiento de Caracteristicas en Espacios Exteriores. Madrid.
- Miniotas, D., Spakov, O., & MacKenzie, S. (2004). Eye Gaze Interaction with Expanding Targets.
- Ogata, K. (2010). Ingeniería de control moderna. En Ingeniería de control moderna (pág. 56). Madrir: Pearson.
- Oliveira , V. A., & Conci, A. (s.f.). Skin Detection using HSV color space. Niterói, Brazil.
- Padmaraja Yedamale, Microchip Technology Inc. (2003). Brushless DC Motor Fundamentals. Chandler.

- Parker, J. (2011). Algorithms for Image Processing and Computer Vision. Indianapolis: Wiley Publishing.
- Petersen, R. (2016). Ubuntu 16.04 LTS Server: Administration and Reference. Surfing Turtle Press.
- Prasantha, H., Shashidhara, H., & Balasubramanya, K. (2009). Image Scaling Comparison Using Universal Image Quality Index. Bangalore, Karnataka, India.
- Preiss, J., Honig, W., Sukhatme, G., & Ayanian, N. (2017). Cracyswarm: A large Nano-Quadcopter Swarm. Singapore.
- Pulli, K., Baksheev, A., Kornyakov, K., & Eruhimov, V. (2012). Real-Time Computer Vision with OpenCV. ACMQUEUE.
- Reston Condit, Microchip Technology Inc. (2004). Brushed DC Motor Fundamentals. Chandler.
- Rodríguez, E. (2015). Sistema de posicionamiento para un drone. San Cristobal De La Laguna.
- Rozantsev, A., Lepetit, V., & Fua, P. (2014). Flying Objects Detection from a Single Moving Camera.
- Rufli, M., Scaramuzza, D., & Siegwart, R. (2008). Automatic Detection of Checkerboards on Blurred and Distorted Images. Nice, Francia: IEEE.
- Russell, M., & Fischaber, S. (2013). OpenCV Based Road Sign Recognition on Zynq. Belfast, Northern Ireland: IEEE.
- Saari, H., Akujarvi, A., Holmlund, C., Ojanen, H., Kaivosoja, J., Nissinen, A., & Niemelainen, O. (2017). Visible, very near IR and short wave IR hyperspectral drone imaging system for agriculture and natural water applications . Jyvaskyla, Finland.
- Seguridad Pent. (11 de 11 de 2016). Usa Python 2.7 y Python 3 en el mismo equipo. Obtenido de https://seguridadpent.blogspot.com/2016/11/usa-python-27-y-python-3-en-el-mismo.html
- Shao, G., & Malkani, M. (2008). Aerial Image Registration Based on Joint Feature-Spatial Spaces, Curve and Template Matching. Zhangjiajie, China: IEEE.
- Soto, D. (2012). Interaccion hombre-robot con vehiculos aéreos no tripulados basada en visión. Tamaulipas.
- Van den Bergh, M., & Van Gool, L. (2011). Combining RGB and ToF Cameras for Real-Time 3D Hand Gesture Interaction. Kona, USA: IEEE.

- Vincent, L. (1993). Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms. Champaign.

8.5 Referencias

1. Aguilar, W. G., & Angulo, C. (2012). Compensación y aprendizaje de efectos generados en la imagen durante el desplazamiento de un robot. En Memorias del X Simposio CEA de Ingeniería de Control (pp. 165-170).
2. Aguilar, W. G., & Angulo, C. (2013). Estabilización robusta de vídeo basada en diferencia de nivel de gris. En Memorias del VIII Congreso de Ciencia y Tecnología. Sangolquí.
3. Aguilar, W. G., & Angulo, C. (2014). Estabilización de vídeo en micro vehículos aéreos y su aplicación en la detección de caras. En IX Congreso de Ciencia y Tecnología ESPE 2014 (pp. 155-160).
4. Aguilar, W. G., & Angulo, C. (2014). Real-time video stabilization without phantom movements for micro aerial vehicles. EURASIP Journal on Image and Video Processing, 2014(1), 46.
5. Aguilar, W. G., & Angulo, C. (2014). Robust Video Stabilization based on Motion Intention for low-cost Micro Aerial Vehicles. En Systems Signals and Devices (SSD), 2014 11th International Multi-Conference on (pp. 1-6). IEEE.
6. Aguilar, W. G., & Angulo, C. (2016). Real-time model-based video stabilization for micro aerial vehicles. Neural processing letters, 43(2), 459-477.
7. Aguilar, W. G., Luna, M. A., Moya, J. F., Abad, V., Parra, H., & Ruiz, H. (2017). Pedestrian Detection for UAVs Using Cascade Classifiers with Meanshift. En IEEE 11th International Conference on Semantic Computing, ICSC 2017 (pp. 509-514). IEEE.
8. Aguilar, W. G., Luna, M., Moya, J., Abad, V., Ruiz, H., Parra, H., & Angulo, H. (2017). Pedestrian detection for UAVs using cascade classifiers and saliency maps. En International Work-Conference on Artificial Neural Networks (pp. 563-574). Springer, Cham.
9. Aguilar, W. G., Luna, M., Moya, J., Abad, V., Ruiz, H., Parra, H., & Lopez, W. (2017). Cascade Classifiers and Saliency Maps Based People Detection. En

International Conference on Augmented Reality, Virtual Reality and Computer (pp. 501-510). Springer, Cham.

10. Aguilar, W. G., Angulo, C., & Costa-Castello, R. (2017). Autonomous Navigation Control for Quadrotors in Trajectories Tracking. En International Conference on Intelligent Robotics and Applications (Vol. 8102, pp. 287-297). Springer, Cham.
11. Aguilar, W. G., Casaliglla, V. P., & Polit, J. L. (2017). Obstacle Avoidance for Low-Cost UAVs. En Semantic Computing (ICSC), 2017 IEEE 11th International Conference on (pp. 503-508). IEEE.
12. Aguilar, W. G., Casaliglla, V., & Pólit, J. (2017). Obstacle Avoidance Based-Visual Navigation for Micro Aerial Vehicles. Electronics, 6(1), 10.
13. Aguilar, W. G., Casaliglla, V., Pólit, J., Abad, V., & Ruiz, H. (2017). Obstacle avoidance for flight safety on unmanned aerial vehicles. En International Work-Conference on Artificial Neural Networks (pp. 575-584). Springer, Cham.
14. Aguilar, W. G., Salcedo, V. S., Sandoval, D. S., & Cobeña, B. (2017). Developing of a Video-Based Model for UAV Autonomous Navigation. En B. C. Barone D., Teles E. (Ed.), Latin American Workshop on Computational Neuroscience. Communications in Computer and Information Science (Vol. 720, pp. 94-105). Springer, Cham.
15. Aguilar, W. G., Abad, V., Ruiz, H., Aguilar, J., & Aguilar-Castillo, F. (2017). RRT-Based Path Planning for Virtual Bronchoscopy Simulator. En International Conference on Augmented Reality, Virtual Reality and Computer (pp. 155-165). Springer, Cham.
16. Aguilar, W. G., & Morales, S. (2016). 3D Environment Mapping Using the Kinect V2 and Path Planning Based on RRT Algorithms. Electronics, 5(4), 70.
17. Aguilar, W. G., Morales, S., Ruiz, H., & Abad, V. (2017). RRT* GL based optimal path planning for real-time navigation of UAVs. En International Work-Conference on Artificial Neural Networks (pp. 585-595). Springer, Cham.
18. Aguilar, W. G., Morales, S., Ruiz, H., & Abad, V. (2017). RRT* GL Based Path Planning for Virtual Aerial Navigation. En International Conference on Augmented Reality, Virtual Reality and Computer (pp. 176-184). Springer, Cham.

19. Aguilar, W. G., Rodríguez, G. A., & Álvarez, L. (2017). On-Board Visual SLAM on a UGV Using a RGB-D Camera. En International Conference on Intelligent Robotics and Applications (Vol. 8102, pp. 298-308). Springer, Cham.
20. Aguilar, W. G., Rodríguez, G., Álvarez, L., Sandoval, S., & Quisaguano, F. (2017). Real-Time 3D Modeling with a RGB-D Camera and On-Board Processing. En International Conference on Augmented Reality, Virtual Reality and Computer (pp. 410-419). Springer, Cham.
21. Aguilar, W. G., Rodríguez, G., Álvarez, L., Sandoval, S., & Quisaguano, F. (2017). Visual SLAM with a RGB-D Camera on a Quadrotor UAV Using on-Board Processing. En International Work-Conference on Artificial Neural Networks (pp. 596-606). Springer, Cham.