UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Implementación de un vehículo aéreo no tripulado aplicado al análisis de cultivos agrícolas

Trabajo de graduación en modalidad de megaproyecto presentado por:

Juan Pablo Argueta Cortés
Kevin Emanuel Mazariegos Morales

para optar al grado académico de Licenciados en Ingeniería Electrónica; y

Luis Carlos Velásquez Mansilla

para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala
2015

Implementación de un vehículo aéreo no tripulado aplicado al análisis de cultivos agrícolas

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Implementación de un vehículo aéreo no tripulado aplicado al análisis de cultivos agrícolas

Trabajo de graduación en modalidad de megaproyecto presentado por:

Juan Pablo Argueta Cortés
Kevin Emanuel Mazariegos Morales

para optar al grado académico de Licenciados en Ingeniería Electrónica; y

Luis Carlos Velásquez Mansilla

para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala
2015

Vo. Bo. :

(f) _____

Ing. Pablo Mazariegos

Asesor

(f) _____

Ing. Pablo Mazariegos

Coordinador Megaproyecto

Director del Departamento:

(f) _____

M.Sc. Carlos Esquit

Director Ingeniería Electrónica y Mecatrónica

Fecha de aprobación: 17 de noviembre de 2015

PREFACIO

El presente megaproyecto nace con la ilusión de aplicar tecnología de vanguardia a sectores que históricamente han estado aislados a la implementación de tecnología en sus procesos productivos. Al aplicar vehículos aéreos no tripulados para el análisis de cultivos agrícolas se obtiene información valiosa sobre la salud, calidad y cantidad de los cultivos existentes en el área analizada.

Con la información recolectada se espera contribuir a la seguridad alimentaria del país al facilitar información de calidad para tomar mejores decisiones al hacer frente a los distintos fenómenos meteorológicos y biológicos que azotan a nuestra región.

Agradecemos a nuestras familias, sin su apoyo incondicional este proyecto no habría sido posible. También agradecemos a nuestros asesores Ing. Pablo Mazariegos y M.Sc. Roberto Saravia por habernos guiado a lo largo del desarrollo de este proyecto.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

El presente documento trata sobre el control de un cuadricóptero empleado para el análisis de cultivos agrícolas a través de la captura aérea de imágenes multiespectrales. En el mismo se describe la metodología utilizada para la selección del cuadricóptero y el hardware de control, la integración de sensores para la estimación de posición y la implementación de los algoritmos de control y generación de trayectorias.

Se utilizó un cuadricóptero Iris+ como base para la estructura. Para la implementación de los algoritmos del control de estabilidad, altura y orientación del cuadricóptero se utilizó una tarjeta de control PX4 Pixhawk. Se instaló el firmware y se implementaron las librerías que controlan la postura y orientación del UAV en la tarjeta PX4. Posteriormente se generaron las trayectorias de vuelo en una computadora de compañía. La comunicación entre la computadora de compañía se realizó empleando el protocolo *mavlink* del Robot Operating System (ROS).

Finalmente se comprobaron los resultados por medio de pruebas de vuelo, donde se observó la capacidad del cuadricóptero de permanecer en una posición fija, seguir trayectorias específicas y realizar trayectorias con la cámara multiespectral.

Las imágenes captadas con la cámara multiespectral fueron procesadas durante el vuelo para calcular el NDVI de los cultivos analizados y posteriormente los resultados de los análisis fueron subidos a un repositorio en la nube.

I. INTRODUCCIÓN

La implementación de vehículos aéreos no tripulados (UAVs) para aplicaciones civiles ha incrementado durante los últimos años. Su potencial radica en la capacidad de utilizar distintos sensores a una altura considerable para la medición de variables de interés. Lo que ha permitido pasar de las aplicaciones militares a usos en minería, energía, ingeniería, seguridad, agricultura, entre otros. Considerando la importancia que tiene la agricultura para Guatemala y la región, cómo esta impacta en la economía del país, así como en el nivel de vida del pequeño agricultor, y sin dejar de lado el potencial que representa la tecnología de vehículos aéreos no tripulados, se ideó el presente Megaproyecto.

El objetivo de este este Megaproyecto es la implementación de un vehículo aéreo, un cuadricóptero específicamente, para el análisis de cultivos agrícolas para calcular el nivel de salud de los cultivos presentes en cierta área de interés. Dicho cálculo se realiza a través de la captura aérea de imágenes multiespectrales y la utilización de algoritmos de procesamiento de imagen.

Se utilizó la base de un cuadricóptero prefabricado (3DR Iris+) al cual se le integraron los sensores y hardware de control para controlar el UAV, posteriormente integró la carga útil para la adquisición de imágenes multiespectrales empleadas para calcular la salud del cultivo a través del NDVI. El funcionamiento del UAV y de la fotografía multiespectral comprobado mediante pruebas de vuelo.

II. JUSTIFICACIÓN

De manera similar a las economías de la región, la agricultura es el sector productivo más importante de Guatemala. Casi una cuarta parte del PIB es producto del sector agropecuario y aporta más de la mitad de divisas por exportaciones. Actualmente, Guatemala utiliza un 67% de su territorio para la actividad (Ministerio de Agricultura 2014), brindando oportunidades laborales al 31% de la PEA. (Narciso 2011)

Considerando la importancia del sector para el desarrollo económico de Guatemala: resulta incomprensible la falta de avances tecnológicos en el área, pues aún se emplean técnicas que datan de varios siglos atrás. Lo cual se hace evidente al observar que gran parte del territorio se emplea en agricultura de infra subsistencia dedicada al autoconsumo (Dirección de planeamiento 2013). A esto se suma el hecho que sólo el 36% del territorio dedicado a la agricultura se usa correctamente, 55% se encuentra en sobre-uso y 9% sub-utilizado. (Carrera 2002)

Con los recientes desarrollos en vehículos aéreos no tripulados y tecnología de procesamiento de imagen, se presenta una oportunidad muy importante para Guatemala y Latinoamérica: Innovar, con la aplicación de tecnología vanguardista en la agricultura. Esto a través de la utilización de Vehículos Aéreos no Tripulados Autónomos para la recolección de información sobre cultivos, información que puede ser utilizada para conocer la salud de plantaciones, estimaciones de producción, índices de irrigación, predicción de posibles plagas, clasificación de cultivos y la elaboración de tendencias históricas del proceso productivo.

III. OBJETIVOS

A. OBJETIVO GENERAL

Implementar un vehículo aéreo no tripulado para la obtención del índice de vegetación de diferencia normalizado (NDVI) de plantaciones agrícolas a través de la captura de imágenes multiespectrales.

B. OBJETIVOS ESPECÍFICOS

- Diseñar e implementar el módulo de control para un vehículo aéreo no tripulado aplicado a la agricultura.
- Calcular el índice de vegetación de diferencia normalizado (NDVI) a través del procesamiento de las imágenes multiespectrales capturadas por el UAV.
- Diseñar e implementar una plataforma de comunicación que permita la interacción entre el vehículo aéreo, la base en tierra y el centro de recolección de datos.

C. OBJETIVOS SECUNDARIOS

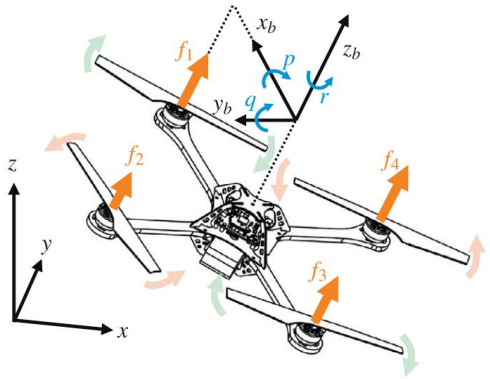
- Determinar el hardware necesario para la implementación de la estructura del vehículo aéreo no tripulado.
- Comparar las opciones de hardware existentes en el mercado y seleccionar la más adecuada para el control del UAV.
- Implementar el software para controlar la estabilidad altura, orientación y posición del vehículo aéreo no tripulado.
- Verificar el funcionamiento del módulo desarrollado mediante pruebas de vuelo.

IV. MARCO TEÓRICO

A. DEFINICIÓN

Un cuadricóptero es un vehículo aéreo no tripulado (UAV por sus siglas en inglés) de alas giratorias. Se caracteriza por su capacidad de permanecer suspendido en una posición determinada y un vuelo preciso al balancear las fuerzas producidas por los cuatro rotores.

Figura 1. Configuración rotores cuadri-rotor



(Boubdallah , Muerrieri y Siegwart 2006)

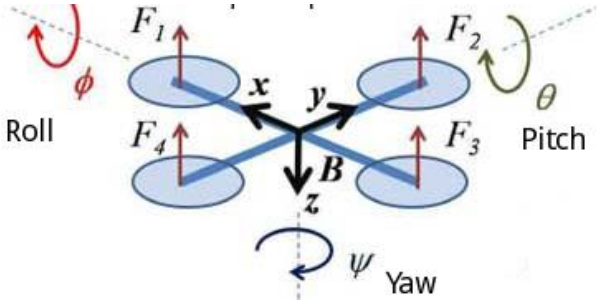
El vehículo se puede describir como un multi-rotor, con una configuración de cuatro hélices dispuestas de forma cruzada. Dos pares de hélices (1,3) y (2,4) tal como se observa en la (Boubdallah , Muerrieri y Siegwart 2006), giran en direcciones opuestas. Al variar la velocidad de los rotores, se puede alterar la fuerza de sustentación y generar movimiento. (Boubdallah , Muerrieri y Siegwart 2006)

Una de las ventajas de un multi-rotor es una elevada capacidad de carga útil respecto a soluciones similares y mayor maniobrabilidad. No obstante esas mejoras vienen a costa de mayor peso de la estructura y alto consumo de potencia. Desde el punto de vista de control, representa un modelo de mayor robustez respecto sus análogos de un único rotor. (Castillo, Lozano y Dzul 2005)

B. ÁNGULOS DE ROTACIÓN

En la industria aeronáutica y aeroespacial la notación más utilizada para describir las rotaciones respecto a los ejes coordenados es la notación *roll, pitch & yaw* o "rpy". El ángulo *roll* representa una rotación respecto al eje "x", *pitch* representa rotación del eje "y", *yaw* por su parte representa una rotación respecto al eje z.

Figura 2.Rotaciones roll pitch yaw

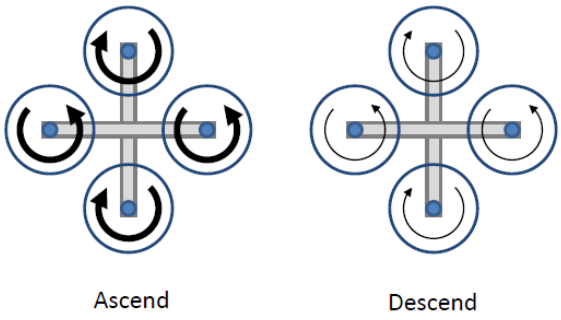


(Sturm 2015)

C. PRINCIPIO DE MOVIMIENTO

Para que el cuadricóptero permanezca en una posición fija es necesario que el empuje total generado por los cuatro rotores sea igual a la fuerza gravitacional ejercida sobre el UAV. De tal forma, al incrementar o disminuir la velocidad de las cuatro hélices simultáneamente se produce un movimiento vertical. Si este es ligeramente superior a la fuerza gravitacional: el UAV se elevará; de manera análoga al ser la fuerza de sustentación menor a la gravitacional, este descenderá. Tal como lo describe la Figura 2 (Boubdallah , Muerrieri y Siegwart 2006)

Figura 3. Movimientos de ascenso y descenso con base en el torque de los rotores



(Sturm 2015)

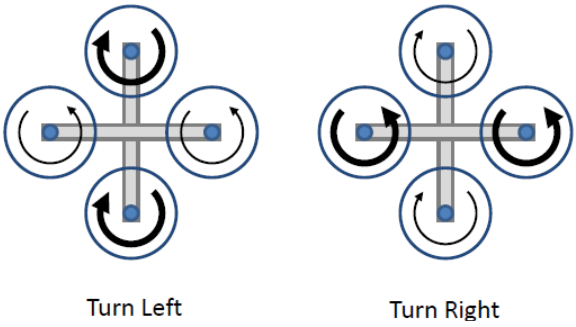
Al variar la velocidad de las hélices 2 y 4 como se muestra en la figura 1, se produce una rotación en el eje x (*roll*) que se traduce en un movimiento hacia adelante o hacia atrás. Una variación en los rotores 1 y 3 produce a su vez una rotación en el eje y (*pitch),* la cual representa un movimiento hacia la izquierda o hacia la derecha. La rotación en el eje z (*yaw*) es un tanto diferente, pues resulta de la diferencia entre torques de ambos pares de rotores (Nice 2004). Esta resulta en un cambio en la orientación del cuadricóptero. Dicho comportamiento se puede observar de mejor manera en la Tabla I, así como en Figura 4, Figura 5 y Figura 6:

Tabla I Principio de movimiento

	Δ Hélice 1	Δ Hélice 2	Δ Hélice 3	Δ Hélice 4
Δ *Roll* +	+	0	-	0
Δ *Pitch* +	0	-	0	+
Δ *Yaw* +	+	-	+	-
Ascender	+	+	+	+

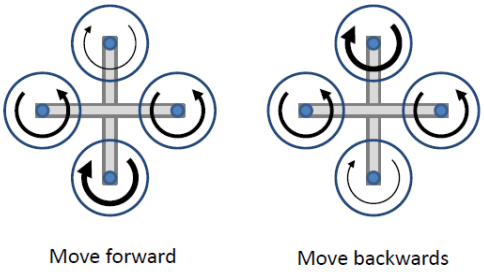
Tabla I Principio de movimiento. Se muestran los cambios a partir de la posición de estabilidad para los distintos movimientos del cuadricóptero

Figura 4. Rotación izquierda y derecha (YAW)



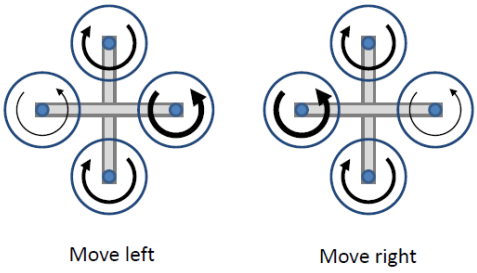
(Sturm 2015)

Figura 5. Movimiento lateral (ROLL)



(Sturm 2015)

Figura 6. Movimiento lateral (PITCH)

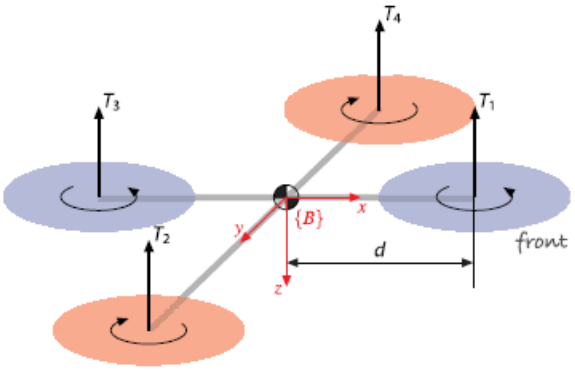


(Sturm 2015)

D. MODELO DINÁMICO

La notación para el modelo del cuadricóptero se muestra en la figura 7. Existe un eje coordenado {B} que se origina en el centro de masa, su eje-z apunta hacia abajo según la convención aeroespacial. El cuadricóptero posee cuatro rotores, identificados de 1 a 4, localizados al extremo de los brazos de la estructura. (Corke 2011)

Figura 7. Modelo teórico del cuadricóptero



(Corke 2011)

La velocidad de los rotores se denota como ω_i, y el empuje es un vector que apunta hacia arriba

$$T_i = b\omega_i^2, i = 1,2,3,4 \quad (1)$$

en la dirección-z negativa, donde $b > 0$ es una constante que depende de la densidad del aire, el radio de las hélices, numero de hélices y el largo de cuerda de las hélices.

La dinámica traslacional del UAV está dada por la segunda ley de Newton:

$$m\dot{v} = \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix} - {}^0R_B \begin{pmatrix} 0 \\ 0 \\ T \end{pmatrix} \quad (2)$$

Donde v es la velocidad del vehículo en el eje coordenado, g es la aceleración gravitacional, m es la masa total del vehículo y $T = \sum T_i$ es el empuje total. El primer término representa la fuerza de gravedad que actúa hacia abajo del eje global y el segundo término es el empuje total transformado al eje global.

A su vez, las diferencias en los empujes de los rotores causan que el vehículo gire. El torque respecto al eje-x es:

$$\tau_x = dT_4 - dT_2$$

Donde d es la distancia del motor al centro de masa. Dicha expresión puede describirse en términos de velocidad al sustituir por la Eq. 1

$$\tau_x = db(\omega_4^2 - \omega_2^2) \quad (3)$$

De manera similar para el eje-y

$$\tau_y = db(\omega_1^2 - \omega_3^2) \quad (4)$$

El arrastre aerodinámico se opone al torque aplicado a cada hélice por el motor.

$$Q_i = k\omega_i^2$$

Donde *k* depende de los mismos factores que la constante *b*. Dicho torque ejerce un torque de reacción en la estructura del UAV que actúa al rotar la estructura respecto al eje del rotor en dirección opuesta a la rotación de los rotores. La reacción total de torques en el eje-*z* es:

$$\tau_z = Q_1 - Q_2 + Q_3 - Q_4 = k(\omega_1^2 + \omega_3^2 - \omega_2^2 - \omega_4^2) \quad (5)$$

Donde los signos opuestos son el resultado de las direcciones opuestas de rotación de los rotores. Un torque en el eje-*yaw* puede crearse al controlar apropiadamente la velocidad de los cuatro rotores. Aceleración rotacional de la estructura se obtiene de la ecuación de movimiento de Euler:

$$\mathbf{J}\dot{\omega} = -\omega \times \boldsymbol{J}\omega + \boldsymbol{\Gamma} \quad (6)$$

Donde $\boldsymbol{J}$ es la matriz de 3 x 3 de inercia, *w* es el ángulo de velocidad angular y $\Gamma = (\tau_x, \tau_y, \tau_z)^T$ es el torque aplicado a la estructura según las Eqs. 3 a 5. El movimiento del cuadricóptero se obtiene al integrar la dinámica directa (Eqs. 2 y 6) donde las fuerzas y momentos son funciones de la velocidad de los rotores:

$$\begin{pmatrix} T \\ \boldsymbol{\Gamma} \end{pmatrix} = \begin{pmatrix} -b & -b & -b & -b \\ 0 & -db & 0 & db \\ db & 0 & -db & 0 \\ k & -k & k & -k \end{pmatrix} \begin{pmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{pmatrix} = \boldsymbol{A} \begin{pmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{pmatrix} \quad (7)$$

Donde la matriz $\boldsymbol{A}$ tiene rango completo si *b, k, d > 0* y puede ser invertida.

$$\begin{pmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{pmatrix} = \boldsymbol{A}^{-1} \begin{pmatrix} T \\ \tau_x \\ \tau_y \\ \tau_z \end{pmatrix} \quad (8)$$

Con el objetivo de dar a los motores las velocidades necesarias para dar un empuje T y un momento Γ determinados.

E. TECNOLOGÍAS DE DETECCIÓN

Escoger los sensores adecuados es de vital importancia al momento de diseñar robots autónomos. Sin un sistema de sensores rápido y confiable, sería difícil obtener una buena estabilización. Para lograr la autonomía, los vehículos aéreos autónomos deben contar con la información concerniente a sus estados, así como el ambiente que los rodea. Debe ser capaz de medir su comportamiento angular para estabilizar su postura. A su vez, los sistemas de detección para localización y percepción del entorno son importantes para lograr completa autonomía. Las características y número de sensores instalados a bordo del UAV deben ser escogidos en base a la tarea del vehículo, adecuándose a la capacidad de carga, consumo de energía y costos de la plataforma. (Garcia, y otros 2013)

La postura del UAV es determinada en su mayoría por sensores inerciales, entre los que destacan:

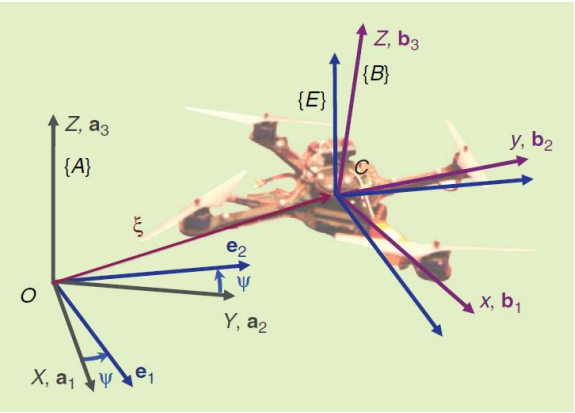
- Giroscopio: mide la velocidad angular del sistema en el marco de referencia inercial. Los giroscopios empaquetan en forma de chip sistemas micro electromecánicos (MEMS por sus siglas en inglés) son comúnmente utilizados en la estabilización de UAVs.
- Acelerómetro: mide la aceleración lineal en el marco inercia. Los acelerómetros modernos usualmente utilizan MEMS, Pueden ser utilizados para medir inclinación, distancia dinámica y velocidad con o sin la influencia de la gravedad.
- Magnetómetro: es un dispositivo utilizado para medir el campo magnético de la Tierra. Generalmente, el magnetómetro provee el ángulo *yaw* de la aeronave.
- Unidad de mediciones inerciales (IMU): es un componente electrónico que mide la velocidad, orientación y fuerzas gravitatorias de la aeronave al utilizar una combinación de acelerómetros, giroscopios y magnetómetros. Las IMUs son utilizadas para maniobrar aeronaves, naves espaciales y satélites. Comúnmente, la IMU es el componente principal del sistema de navegación. La información recolectada de la IMU permite a la computadora recopilar la posición utilizando la navegación por estima (*Dead Reckoning*).
- Sistema de posicionamiento global (GPS): un sistema global de navegación basado en un satélite espacial (GNSS) provee la ubicación absoluta en cualquier punto de la Tierra, cuando existe una línea de vista ininterrumpida a cuatro o más satélites GPS. Los GPS reportan información como la velocidad, altitud y conducta.
- Telémetro láser (LRF por sus siglas en inglés): es un dispositivito que emplea un rayo láser para determinar la distancia hacia un objeto. Los LRFs operan en vuelo al enviar un pulso laser en un rayo angosto hacia un objeto y medir el tiempo que le toma al pulso ser reflejado del objetivo y regresar al emisor.
- Sensor ultrasónico: genera ondas de sonido a alta frecuencia y evalúa el eco que es recibido devuelta por el sensor. Para determinar la distancia se calcula la diferencia de tiempo entre la señal enviada y la señal recibida.

- Sensor infrarrojo: transmite pulsos infrarrojos y mide el tiempo que le toma regresar, estimando de esta manera la distancia. Funciona para distancias más cortas que los sensores ultrasónicos.
- Sensor de presión: genera una señal en función de la presión a la cual se encuentra sometido. En aeronaves, cohetes, satélites y globos meteorológicos, los sensores de presión son utilizados para medir la altitud al utilizar la relación entre los cambios de presión relativos a la altitud.
- Procesamiento de imagen: la medición visual resulta muy atractiva por el hecho de ser pasiva, sin contacto, versátil y de bajo costo. Provee gran cantidad de información sobre el entorno del robot y es probablemente la fuente más rica en información. Sin embargo, presenta diversas complicaciones técnicas: el mapeo del plano de imagen a coordenadas 3D es no-lineal. Además, el procesamiento de imagen es computacionalmente intensivo e introduce latencias entre el tiempo de captura y el tiempo de disponibilidad.

F. ESTIMACIÓN

Los estados clave requeridos para el control de un cuadricóptero son su altura, postura, velocidad angular y velocidad lineal. De dichos estados, la postura y la velocidad angular son los más importantes, ya que son las variables principales utilizadas para calcular la postura del vehículo. El instrumento más básico utilizado es la IMU, a la cual se le acompaña frecuentemente por una medición de altura, ya sea utilizando información acústica, infrarroja, barométrica o laser. Muchas aplicaciones robóticas requieren de sensores más complejos como sistemas VICON, GPS o telemetría laser. (Mahony, Kumar y Corke 2012)

Figura 8. Marcos de referencia del cuadricóptero



(Mahony, Kumar y Corke 2012)

1. Estimación de la postura: Una IMU convencional incluye un giroscopio de tres ejes, acelerómetro de tres ejes y magnetómetro de tres ejes. El giroscopio determina la velocidad angular de {B} respecto a {A} (ver Figura 8) expresada en el marco de referencia {B}.

$$\Omega_{IMU} = \Omega + b_{\Omega} + \eta \in \{B\} \quad (9)$$

Donde η denota el error incremental del ruido y b_Ω representa una constante de error de giro. Generalmente, los giroscopios instalados en cuadricópteros son dispositivos MEMS livianos y robustos al sonido.

Los acelerómetros por su parte, miden la aceleración lineal instantánea de {B} resultado de fuerzas externas.

$$a_{IMU} = R^T(\dot{v} - g\vec{\boldsymbol{z}}) + b_a + \eta_a \in \{B\} \quad (10)$$

Donde b_a es el término de error, η_a denota el ruido incremental y $\dot{v}$ se encuentra en el marco inercial. Los acelerómetros son susceptibles a las vibraciones, y al momento de montarse en el cuadricóptero, requieren un filtro pasa-baja mecánico y/o electrónico. La mayoría de los componentes diseñados para aeronáutica incorporan filtros análogos *antialising* previo a la etapa de muestreo. Una estrategia común para determinar el error b_Ω y b_a es promediar la salida de estos sensores por unos cuantos segundos mientras el UAV se encuentra en tierra y los motores aún no se encuentran activados. El error se supone constante durante el vuelo.

Los magnetómetros proveen mediciones del campo magnético del entorno.

$$m_{IMU} = R^\top {}^A m + B_m + \eta_b \in \{B\} \quad (11)$$

Donde ${}^A m$ es el vector de campo magnético de la Tierra (expresado en el marco inercial), B_m es la expresión de la perturbación magnética local, y η_b denota el ruido de la medición. El ruido η_b es normalmente bajo, sin embargo, la perturbación local B_m puede ser significativa, especialmente si el sensor se ubica cerca a los cables de potencia de los motores.

Los acelerómetros y magnetómetros pueden ser utilizados para proveer información sobre la postura absoluta, mientras que el giroscopio brinda información complementaria sobre las velocidades angulares. La interpretación de la información brindada por el magnetómetro puede ser interpretada directamente.

En el caso del acelerómetro, la estimación requiere un poco más de trabajo. Usando el modelo más simple de (10) donde el error y el ruido son despreciables.

$$a_{IMU} = R^\top(\dot{v} - g\vec{\boldsymbol{z}}) = \left(\frac{T_\Sigma}{m}\right)\vec{\boldsymbol{z}} \approx g\vec{\boldsymbol{z}}$$

Muestra que la aceleración medida siempre apuntara en dirección del vector $\vec{\boldsymbol{z}}$ y no brinda información de la postura. En la práctica, es el componente de aleteo de hélice el que brinda información sobre la postura a la señal del acelerómetro. Tal como lo describen P. Martin y E. Shaun en (Martin y Shaun 2010). A partir de lo cual el modelo de la aceleración puede ser reescrito como:

$$a_{IMU} = -\frac{T_\Sigma}{m}\vec{\boldsymbol{z}} - \frac{T_\Sigma}{m} D R^\top v \quad (12)$$

Por cuestiones prácticas, sólo la respuesta de baja frecuencia o el error aproximado en estado estable de la velocidad se utiliza para construir el modelo de la aceleración. Se establece entonces que:

$$\bar{a}_{IMU} \approx -\frac{T_\Sigma}{m} R^\top \vec{\boldsymbol{z}} \quad (13)$$

Donde $\bar{a}_{IMU}$ denota la componente a baja frecuencia de la señal del acelerómetro. Dicha expresión la componente de aceleración cuando el vehículo está cercano a la posición suspendida. La mayoría de aplicaciones implican que el cuadricóptero permanezca grandes periodos de tiempo en posición suspendida o vuelo lento con aceleraciones cercanas a cero. Por lo que usar la lectura del acelerómetro como referencia de la postura ha resultado útil en la práctica.

La cinemática de la postura del cuadricóptero está dada por:

$$\dot{R} = R\Omega_\times \quad (14)$$

Donde $\widehat{\mathrm{R}}$ representa un estimado de la postura *R* del cuadricóptero. Tal como lo proponen en (Mahony, hamel y Pflimin 2008), el siguiente observador combina la información del acelerómetro, magnetómetro y giroscopio, así como otros estimados de postura directos R_E (como los provistos por un sistema VICON) en caso estén disponibles:

$$\dot{\hat{R}} := \hat{R}\left(\Omega_{IMU} - \hat{b}\right)_x - \alpha$$

$$\dot{\hat{b}} := k_b \alpha$$

$$\alpha := \left(\frac{k_a}{g^2}\left(\left(\hat{R}^\top \vec{\boldsymbol{z}}\right) \times \bar{a}_{IMU}\right) + \frac{k_m}{\left|{}^A m\right|^2}\left(\left(\hat{R}^\top {}^A m\right) \times m_{IMU}\right)\right)_x + k_E \mathbb{P}_{\mathfrak{so}(3)}\left(\hat{R} R_E^\top\right) \quad (15)$$

Donde k_a, k_m, k_e *y* $k_b$son ganancias arbitrarias y positivas del observador y $\mathbb{P}_{\mathfrak{so}(3)}(M) = \frac{M - M^T}{2}$ es la matriz de proyección a matrices anti simétricas. En caso alguno de los datos no esté disponible, la ganancia de dicho observador deberá ser cero.

a. Observabilidad. Considerando un sistema homogéneo descrito por las siguientes ecuaciones:

$$\dot{\boldsymbol{x}} = \boldsymbol{Ax}$$
$$\boldsymbol{y} = \boldsymbol{Cx}$$

Donde **x** es el vector de estado (vector-*n*), **y** el vector de salida (vector-*m*), **A** una matriz de n x n y **C** una matriz de m x n. El sistema es completamente observable si cada estado $x(t_0)$ puede ser determinado por la observación de $y(t)$ sobre un intervalo finito de tiempo, $t_0 \leq t \leq t_1$. El sistema es por tanto completamente observable si cada transición del estado eventualmente afecta cada elemento del vector de salida. El concepto de observabilidad es útil al resolver el problema de reconstruir variables de estado no-medibles a partir de variables medibles en la menor duración de tiempo posible. (Ogata 2010)

El concepto de observabilidad es muy importante porqué, en la práctica, se la dificultad que se presenta con los controles de estado retroalimentados es el hecho de que algunas de las variables de estado no son accesibles a partir de una medición directa, lo que resulta en la necesidad de estimar las variables de estado no-medibles en orden de construir las señales de control.

$$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Bu}$$
$$\boldsymbol{y} = \boldsymbol{Cx} + \boldsymbol{Du}$$

Entonces:

$$\boldsymbol{x}(t) = e^{\boldsymbol{A}t}\boldsymbol{x}(0) + \int_0^t e^{\boldsymbol{A}(t-\tau)}\boldsymbol{Bu}(\tau)d\tau$$
$$\boldsymbol{y}(t) = \boldsymbol{C}e^{\boldsymbol{A}t}\boldsymbol{x}(0) + \boldsymbol{C}\int_0^t e^{\boldsymbol{A}(t-\tau)}\boldsymbol{Bu}(\tau)d\tau + \boldsymbol{Du}$$

Dado que las matrices **A, B, C** y **D** son conocidas y **u**(t) también es conocida, los últimos dos términos de la ecuación son cantidades conocidas. Por lo tanto, pueden ser sustraídas del valor observado de **y**(t).

2. Estimación de la velocidad de traslación. La respuesta de *blade-flapping* provee una forma de construir un observador para la velocidad horizontal del vehículo basado en los sensores de la IMU, tal como se propone en (Martin y Shaun 2010), toda vez el UAV este volando sobre el plano horizontal. Suponiendo que se cuenta con un buen estimado de la postura $\widehat{R}$ y el cuadricóptero vuela a altura constante. Por la proyección de la dinámica horizontal del cuadricóptero, la componente horizontal de la aceleración inercial se puede medir por:

$$ {}^{A}a_h \coloneqq \mathbb{P}_h \, {}^{A}a = \mathbb{P}_h R a \approx \mathbb{P}_h \hat{R} a \quad (16) $$

Donde las señales a y $\widehat{R}$ están disponibles. Dada la suposición de que el vehículo está volando a altura constante, $v_z \sim 0 \; y \; \mathbb{P}_h^T v_h \approx v$. Así como el empuje $T_\sum \approx mg$ debe compensar el peso del vehículo. Tomando la componente horizontal de la aceleración inercial (12), se obtiene:

$$ {}^{A}a_h \approx -g\mathbb{P}_h \hat{R}\vec{\boldsymbol{z}} - g\mathbb{P}_h \hat{R} D R^{\top} \mathbb{P}_h^{\top} v_h \quad (17) $$

Suponiendo que la estimación de postura es acertada ($\widehat{R}$ =R), entonces se puede resolver (16) y (17) en términos de v_h:

$$ v_h \approx -\frac{1}{g}\left[\mathbb{P}_h \hat{R} D R^{\top} \mathbb{P}_h^{\top}\right]^{-1} \left(\quad {}^{A}a_h + g\mathbb{P}_h \hat{R}\vec{\boldsymbol{z}}\right) \quad (18) $$

Generalmente la señal puede llegar a ser muy ruidosa, por lo que su componente de frecuencia baja se puede utilizar para construir un observador que utilice el estimado de la postura. Siendo $\widehat{v_h}$ el estimado de la componente horizontal de la velocidad inercial, se propone el siguiente observador:

$$ \dot{\hat{v}}_h = -g\mathbb{P}_h^{\top}(\hat{R}\vec{\boldsymbol{z}} + \hat{R} D \hat{R}^{\top} \mathbb{P}_h^{\top} \hat{v}_h) - k_w(\hat{v}_h - v_h) \quad (19) $$

Donde v_h esta dado por (18) la ganancia k_w representa un parámetro de ajuste para el filtrado de información.

3. Estimación de posición. El último factor de estado a ser determinado es la posición. Comúnmente se considera de manera separada la posición en el plano y la altura. Para la altura existen dos alturas diferentes: la primera es la altura absoluta del UAV y la segunda es la altura relativa al terreno. No existe una manera efectiva de determinar la altura con la IMU. La mayoría de cuadricópteros incluyen un sensor barométrico que puede determinar la altura absoluta, la cual también puede ser estimada por GPS, VICON, o SLAM. La altura relativa puede ser estimada con sonares, telémetros laser o sensores infrarrojos. (Siciliano y Khatib 2008)

La posición en el plano también se determina de manera absoluta y relativa. La posición absoluta se obtiene de un GPS, o un sistema externo de captura de movimiento como el VICON. Es importante considerar que el GPS tiene un desempeño pobre en interiores y los sistemas de captura de movimiento VICON son costosos y con rango de operación limitado.

La posición relativa se estima al medir la distancia a objetos del ambiente desde los sensores a bordo, generalmente LRFs o sistemas de visión RGBD como el MS Kinect. Un problema con los LRFs es que sólo proveen una área transversal del entorno 3D y las variaciones del UAV dan la impresión de que las paredes del ambiente están en movimiento.

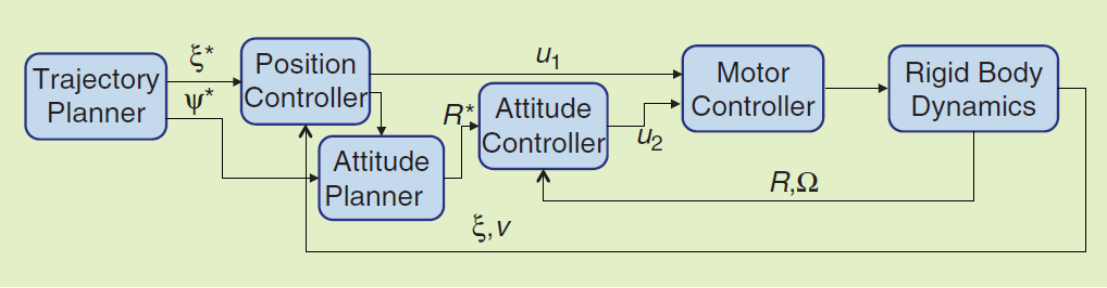
Los sistemas de visión son pequeños, livianos y de bajo consumo, lo que ha resultado en que su utilización vaya en aumento. Los sistemas de visión pueden proveer de información de odometría, postura, mapeo y reconocimiento de objetos. Sin embargo, los sistemas de visión son computacionalmente intensivos, por lo que las tasas de muestreo son bajas. Lo que implica procesadores más potentes para la implementación de los sistemas de visión.

G. ESTRATEGIA DE CONTROL

El problema de control resulta ser retador por varias razones. Primero, el sistema es sub-actuado: existen cuatro actuadores, mientras que el espacio es de seis dimensiones. Segundo, el modelo aerodinámico se basa en aproximaciones. Por último, los actuadores son idealizados. En la práctica, los motores deben contrarrestar fuerzas de arrastre para generar las velocidades requeridas por el controlador. (Mahony, Kumar y Corke 2012)

Una estrategia efectiva es un enfoque jerárquico o anidado. El nivel más bajo es la velocidad angular de los rotores, seguida por la postura del cuadricóptero y finalmente la posición. Tal como se muestra en la Figura 9.

Figura 9. Esquema de Control



(Mahony, Kumar y Corke 2012)

El lazo más bajo emplea un controlador proporcional derivativo (PD) para calcular el torque requerido por el cuadricóptero.

$$\tau_y = K_p\left(\theta_p^* - \theta_p\right) + K_d\left(\dot{\theta}_p^* - \dot{\theta}_p\right) \quad (20)$$

Basado en el error entre el ángulo de *pitch* deseado y el actual. Las ganancias K_p y $K_d$son determinadas con las técnicas clásicas de diseño de control basadas en aproximaciones al modelo dinámico y posteriormente realizando un proceso de sintonizada para mejorar el desempeño.

Considerando un eje coordenado {V} con el mismo origen que {B}(figura 8) pero con sus ejes-*x*-*y* paralelos al suelo. Para mover el vehículo en la dirección *x* se inclina en *pitch* la parte frontal del UAV que genera una fuerza:

$$\boldsymbol{f} = \boldsymbol{R}_y(\theta_p)\begin{pmatrix}0\\0\\T\end{pmatrix} = \begin{pmatrix}Tsin\theta_p\\0\\Tcos\theta_p\end{pmatrix} \quad (21)$$

Que tiene una componente:

$$\boldsymbol{f}_x = Tsin\theta_p \approx T\theta_p \quad (22)$$

La cual acelera el vehículo en la dirección v_x. Se puede controlar la velocidad en dicha dirección con un control proporcional:

$$\boldsymbol{f}_x^* = mK_f\left({}^V v_x^* - {}^V v_x\right) \quad (23)$$

Combinando las ecuaciones anteriores se obtiene el ángulo de *pitch*:

$$\theta_p^* = \frac{m}{T}K_f\left({}^V v_x^* - {}^V v_x\right) \quad (24)$$

Requerido para obtener la velocidad deseada. La velocidad actual ${}^V V_x$ seria estimada por el sistema de navegación inercial y/o GPS.

Si la posición del vehículo en el plano-*xy* del eje global es $\mathcal{P} \in \mathbb{R}^2$ entonces la velocidad deseada está dada por:

$$v^* = K_p(p^* - p) \quad (25)$$

Basado en el error entre la posición actual y la deseada. La velocidad deseada en el plano {V} es

$${}^V\boldsymbol{v} = {}^V\boldsymbol{R}_0(\theta_y)\boldsymbol{v} = {}^0\boldsymbol{R}_V^T(\theta_y)\boldsymbol{v}$$

La cual es una función del ángulo de *yaw*

$$\begin{pmatrix}{}^V v_x\\{}^V v_y\end{pmatrix} = {}^0\boldsymbol{R}_V^T(\theta_y)\begin{pmatrix}v_x\\v_y\end{pmatrix}$$

Para alcanzar la posición deseada se puede computar la velocidad apropiada y a partir de esto el ángulo apropiado que genera la fuerza necesaria para mover el vehículo. Este proceso indirecto es consecuencia de ser un robot sub-actuado. Ya que sólo se tienen cuatro velocidades de rotor pero el espacio de configuración del UAV es de 6 dimensiones. Para mover hacia adelante el cuadricóptero se debe hacer un *pitch* hacia abajo, así el vector de empuje tiene una componente horizontal que lo acelere. Cuando se acerca

a su posición objetivo, el quad debe rotar en la dirección contraria, *pitch* hacia arriba, para que la componente horizontal contraria lo desacelere. Finalmente, el UAV rota a la horizontal con el vector de empuje vertical (*yaw*). Nuevamente, esto es resultado de la condición sub-actuada del robot. (Tedrake 2014)

El eje *yaw* es controlado por un control proporcional-derivativo:

$$\tau_z = K_p\left(\theta_y^* - \theta_y\right) + K_d(\dot{\theta}_y^* - \dot{\theta}_y)$$

La derivada del ángulo *yaw* $\dot{\theta}_y^*$ es ignorada ya que suele ser despreciable. La altura por su parte es controlada por un control proporcional-derivativo:

$$T = K_p(z^* - z) + K_d(\dot{z}^* - \dot{z}) + \omega_0$$

Donde el término aditivo:

$$\omega_0 = \sqrt{\frac{mg}{4b}} \quad (26)$$

H. HARDWARE

Pixhawk Autopilot

El PX4 PIXHAWK es un módulo de control de alto desempeño diseñado para UAVs de ala fija, multi-rotores, helicópteros, carros, botes y otras plataformas robóticas móviles. Está enfocado a la investigación de alto nivel y las necesidades de la industria.

Figura 10. PX4 Autopilot

(PX4 Autopilot 2014)

Características:

- 168 MHz / 252 Mips Cortex-M4F
- 14 salidas PWM/Servo
- Periféricos de comunicación (UART, I2C, CAN)
- Sistema de *backup* para recuperación en vuelo y *override* manual.
- *Switch* seguridad externo
- Piezo-indicador multi-tono
- Tarjeta microSD para registro

Procesador:

- 32bit STM32F427 Cortex M4 core with FPU
- 168 MHz
- 256 KB RAM
- 2 MB Flash
- 32 bit STM32F103 failsafe co-processor

Sensores:

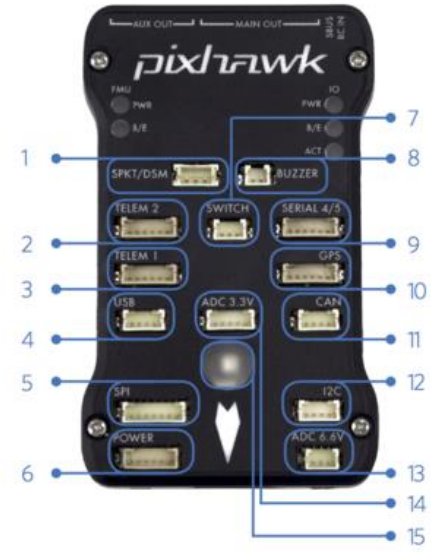
- ST Micro L3GD20H 16 bit giroscopio
- ST Micro LSM303D 14 bit acelerómetro / magnetómetro
- Invensense MPU 6000 3-axis acelerómetro/giroscopio
- MEAS MS5611 barómetro

Interfaces:

- 5x UART (puertos seriales),
- 2x CAN
- Spektrum DSM / DSM2 / DSM-X® Satellite
- Futaba S.BUS®
- PPM sum signal input
- RSSI
- I2C
- SPI
- 3.3 and 6.6V ADC inputs

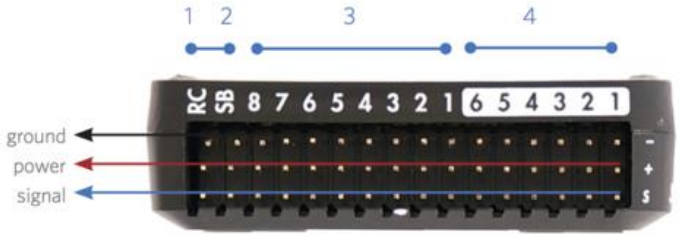
1. Conexiones:

Figura 11. Conexiones PX4



1 Spektrum DSM receiver
2 Telemetry (radio telemetry)
3 Telemetry (on-screen display)
4 USB
5 SPI (serial peripheral interface) bus
6 Power module
7 Safety switch button
8 Buzzer
9 Serial
10 GPS module
11 CAN (controller area network) bus
12 I²C splitter or compass module
13 Analog to digital converter 6.6 V
14 Analog to digital converter 3.3 V
15 LED indicator

1 Input/output reset button
2 SD card
3 Flight management reset button
4 Micro-USB port



1 Radio control receiver input
2 S.Bus output
3 Main outputs
4 Auxiliary outputs

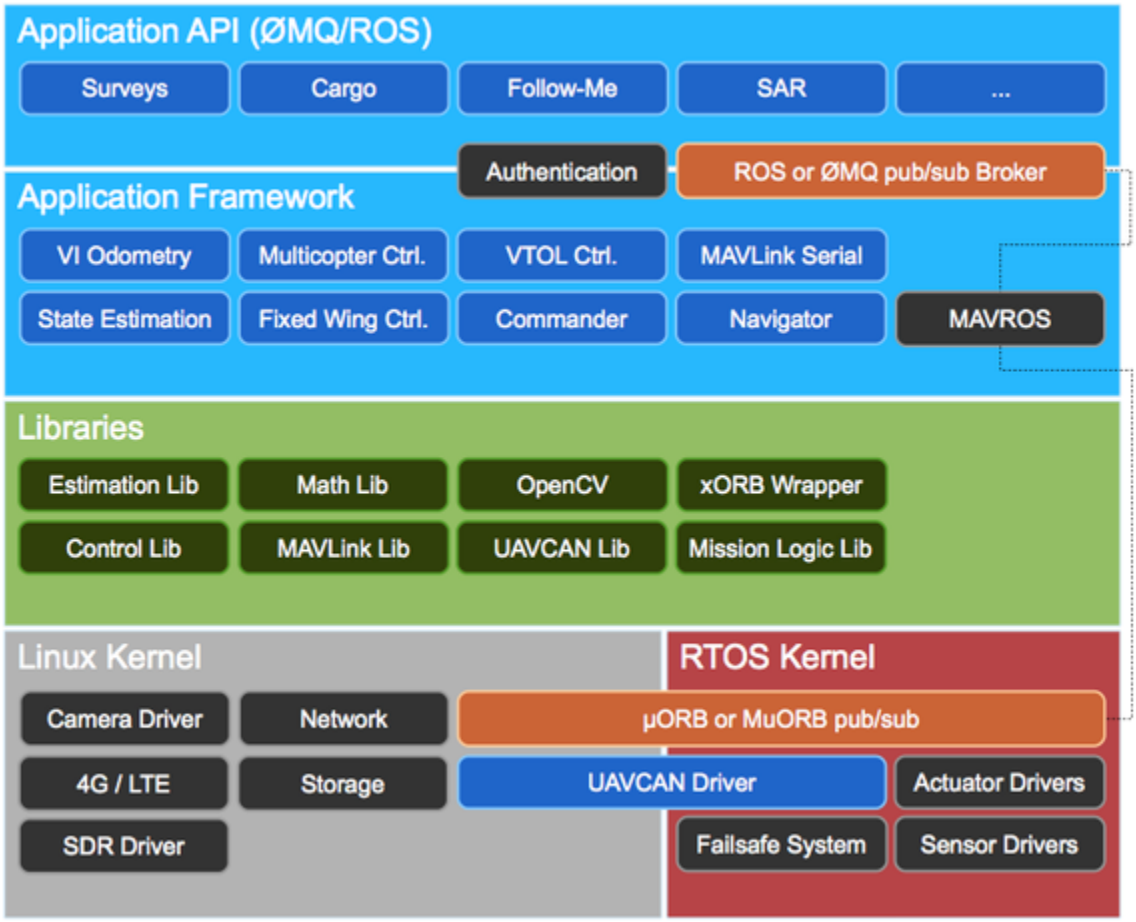
(PX4 Autopilot 2014)

I. ARQUITECTURA DE SOFTWARE:

El PX4 está organizado como un conjunto de nodos interconectados que utiliza canales semánticos como “attitude” o “posición” para comunicar el estado del sistema. El software está apilado en cuatro capas principales: (Pixhawk 2014) (Meier, Honegger y Pollefeys 2015)

- Apps API: Esta interfaz está enfocada para desarrolladores de aplicaciones, i.e. utilizar ROS o la DroneAPI. Dicha API está diseñada para ser liviana e intuitiva y abstraer toda la complejidad posible.
- *Application Framework:* Este es el conjunto de aplicaciones principales (nodos) que operan los controles de vuelo principales.
- Librerías: Esta capa contiene todas las librerías del sistema y la funcionalidad para la operación del vehículo.
- Sistema Operativo: La última capa provee los controladores de hardware, *networking*, y los sistemas de seguridad.

Figura 12. Arquitectura de Software del PX4

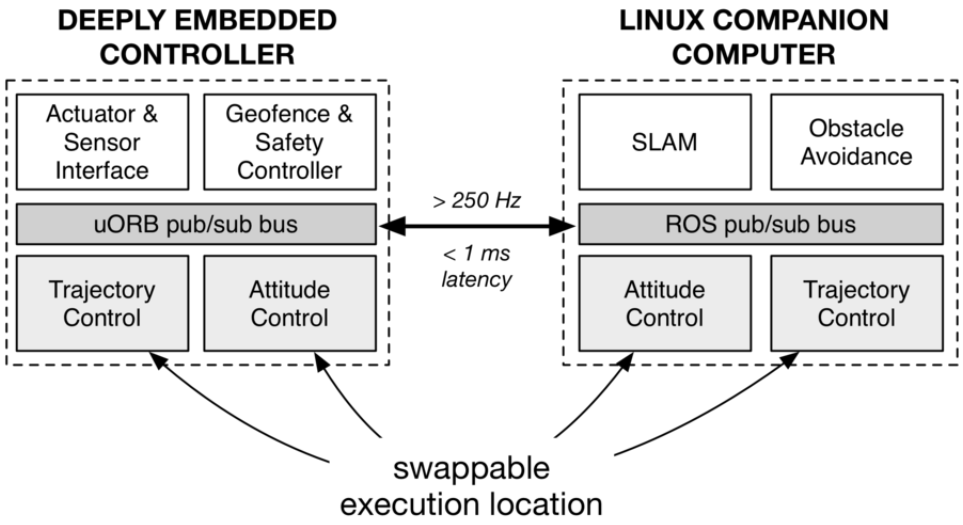


(Pixhawk 2014)

J. SISTEMAS HÍBRIDOS:

Para resolver problemas de mayor nivel como evasión de obstáculos basada en procesamiento de imagen o problemas complejos de control, una computadora de compañía ejecutando Linux embedido puede ser de mucha ayuda:

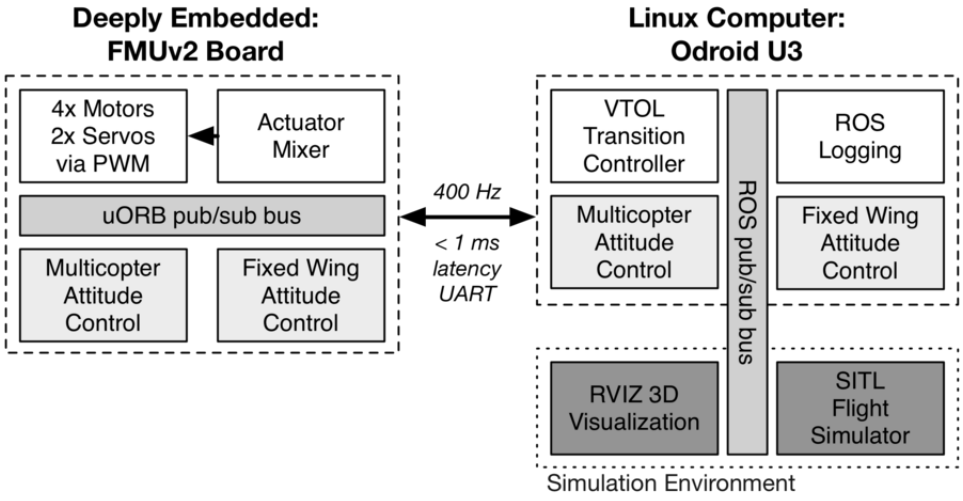
Figura 13. Esquema configuración híbrida



(Pixhawk s.f.)

1. Integración con ROS: El PX4 puede integrarse vía dos diferentes APIs con ROS: ya sea nativamente como un nodo ROS o cuando se ejecuta únicamente en el autopiloto vía *mavros.*

Figura 14. Esquema integración con ROS



(Pixhawk s.f.)

K. APLICACIONES A BORDO

Las aplicaciones centrales son inicializadas cuando la tarjeta es energizada, otras pueden ser activadas al establecer una comunicación con la tarjeta controladora. Las cuales son descritas a detalle en (Pixhawk 2014).

Aplicaciones de sistema:

- mavlink – Envia y recibe mensajes MAVlink por el puerto serial
- sdlog2 – Registra información de vuelo en la SD
- tests – Aplicaciones de prueba
- top – Enumera las aplicaciones activas
- uORB – Administrador de comunicaciones

Drivers:

- mkblctrl – Driver Mikrokopter BLCTRL
- esc_calib – Calibración ESC
- fmu – Configura los pines FMU GPIO
- gpio_led – Driver GPIO LED
- gps – Driver GPS
- pwm – Comando para determinar PWM
- sensors – Sensores
- px4io – Driver PX4IO
- uavcan – Driver UAVCAN

Aplicaciones de control de vuelo:

- Seguridad de vuelo y navegación
    - commander – Maquina de seguridad de estado
    - navigator – Sistema a prueba de fallas

- Estimadores de postura y posición
    - attitude_estimator_ekf – Estimador de postura EKF
    - ekf_att_pos_estimator – Estimador de postura y posición EKF
    - position_estimator_inav – Estimador de navegación inercial

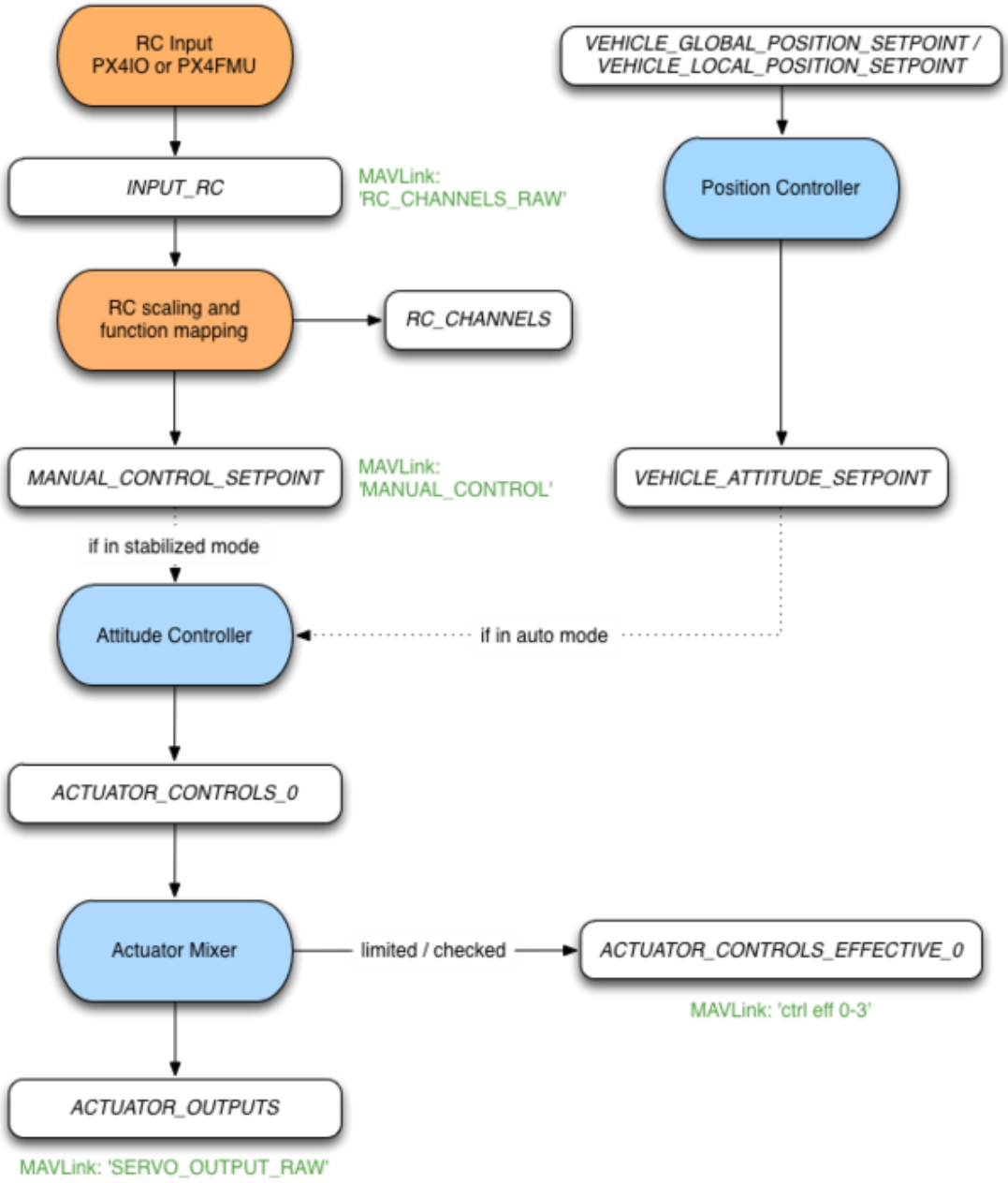
- Controladores de postura y posición
    - mc_att_control – Control de postura del multirotor
    - mc_pos_control – Control de posición del multirotor

L. ARQUITECTURA DEL CONTROLADOR:

La capa media de software utiliza un conjunto de interfaces estandarizadas para leer las entradas de control, generar las señales de control y transmitirlas a los actuadores. Las entradas pueden ser diversas, típicamente valores de radio enviados por el piloto, puntos objetivo, o velocidades deseadas. (Pixhawk s.f.)

A continuación se muestra como las entradas de control son procesadas in la arquitectura del controlador a bordo.

Figura 15. Diagrama de flujo del controlador

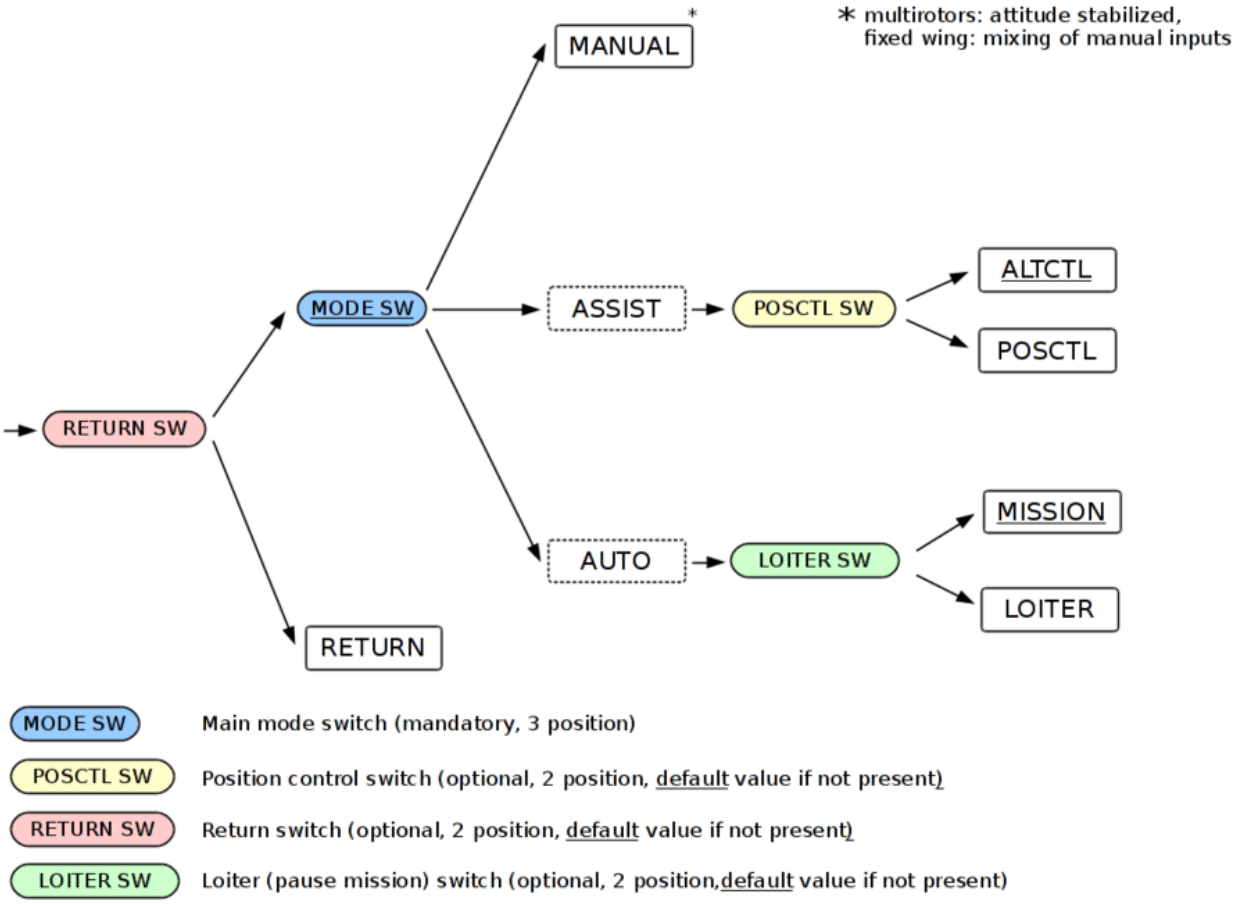


(Pixhawk s.f.)

M.MODOS DE VUELO:

El control de vuelo del sistema es controlado por el radio-controlador, la estación en tierra o parámetros a bordo que alteran los valores por defecto. Si el radio-controlador está configurado para controlar un modo de vuelo particular, este sobrescribirá cualquier requerimiento conflictivo por la estación en tierra. Los posibles modos de vuelo y sus dependencias se detallan a continuación: (Pixhawk s.f.)

Figura 16. Modos de vuelo



(Pixhawk s.f.)

Manual: Los ángulos de *roll* y *pitch* así como *yaw* son estabilizados. El empuje es manual, lo que implica que mantendrá su postura en el plano y se moverá en las direcciones indicadas por el radio controlador.

ALTCTL: *Yaw*, *roll* y *pitch* se trabajan igual que en el modo manual. El acelerador vertical indica si el cuadricóptero asciende o desciende.

POSCTL: *Roll* controla la velocidad izquierda-derecha, *pitch* controla la velocidad frontal-trasera respecto a la tierra. *Yaw* se controla igual que en el modo manual.

N. ROBOT OPERATING SYSTEM

Robot Operating System (ROS) es un *framework* ampliamente utilizado en el campo de la robótica. Creado bajo la premisa de crear una pieza de software que pueda trabajar en otros robots al realizar pequeños cambios de código. Obteniendo de esta manera funcionalidades que pueden ser compartidas y utilizadas en otros robots sin mucho esfuerzo.

ROS fue desarrollado originalmente en el 2007 por el Laboratorio de Inteligencia Artificial de Stanford (SAIL). El desarrollo continuo en el Willow Garage, un instituto de investigación en robótica, con más de 20 instituciones colaborando al proyecto. (Martinez y Fernández 2013)

ROS es un meta-sistema operativo para robots. Provee servicios que se esperarían de un sistema operativo, incluyendo abstracción de hardware, control del dispositivo de bajo nivel, implementación de funcionalidades de uso común, comunicación entre procesos y manejo de paquetes. También provee herramientas y librerías para obtener, construir, escribir y ejecutar código a través de múltiples computadores. (O'Kane 2014)

1. Nivel de sistema de archivos: Similar a un sistema operativo, un programa ROS está dividido en folders, dichos folders contienen archivos que describen su funcionalidad. Tal cómo lo describen en (Martinez y Fernández 2013).

- Paquetes: Los paquetes forman el nivel atómico de ROS. Un paquete tiene la estructura más pequeña, así como el contenido para crear un programa en ROS. Puede tener procesos de ejecución (nodos), archivos de configuración, entre otros.
- Manifiestos: Los manifiestos proveen información sobre un paquete, información sobre licencia, dependencias, banderas de compilación, entre otros. Los manifiestos son manejados con un archivo llamado manifiestos.xml.
- Pilas: Al agrupar varios paquetes con cierta funcionalidad, se obtiene una pila. En ROS existe diversidad de pilas con distintos propósitos, i.e. una pila de navegación.
- Manifiestos de pila: un manifiesto de pila (stack.xml) proveen información sobre una pila, incluyendo su licencia y sus dependencias hacia otras pilas.
- Tipos de mensaje (msg): Un mensaje es la información que envía un proceso a otros. ROS posee varios tipos de mensajes standard. Las descripciones de mensajes están almacenadas en my/package/msg/MyMessageType.msgs.
- Tipos de servicio: Las descripciones de servicios definan los requisitos y respuestas para los servicios en ROS.

2. Nivel de computación gráfico: ROS crea una red donde todos los procesos están conectados. Cualquier nodo en el sistema puede acceder a dicha red, interactuar con otros nodos, ver información que están enviando y transmitir la información a la red.

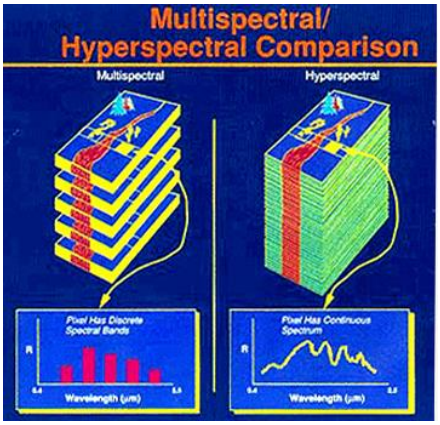
- Nodos: los nodos son procesos donde la ejecución es realizada. Para que un proceso pueda interactuar con otros nodos, es necesario crear un nodo donde dicho proceso pueda interactuar con la red ROS. Usualmente un sistema tendrá varios nodos que controlen diferentes funciones. Resulta conveniente tener varios nodos que provean una única funcionalidad, en lugar de un gran nodo que realice todas las funciones en el sistema.
- Maestro: el maestro provee el registro de nombre y la administración para el resto de nodos. Si no existe un maestro en el sistema, no es posible comunicarse con los nodos, servicios, mensajes y otros. Sin embargo, existe la posibilidad en otra computadora donde exista un nodo maestro.
- Servidor de parámetros: el servidor de parámetros brinda la posibilidad de almacenar información utilizando claves en una locación central. Con dicho parámetro, es posible configurar nodos cuando es siendo ejecutado.
- Mensajes: Los nodos se comunican entre sí a través de mensajes. Un mensaje contiene información que envía información a otros nodos. ROS tiene diversos tipos de mensajes, es posible crear un tipo de mensaje propio.
- Temas: cada mensaje debe tener un nombre al cual ser dirigido por la red ROS. Cuando un nodo envía información, se dice que se encuentra publicando a un tema. Los nodos pueden recibir temas desde otros nodos al subscribirse al tema. Un nodo puede subscribirse a un tema, no es necesario que el nodo este publicando a dicho tema para que exista. Esto permite disminuir el consumo de recursos. Es esencial que el nombre del tema sea único para evitar confusiones y problemas de comunicación.
- Servicios: Cuando se publica un tema, se envía información en una forma *many-to-many*¸ sin embargo cuando se necesita un requerimiento o una respuesta de un nodo, no se puede a través de los temas. El servicio da la posibilidad de interactuar con nodos, A la vez, los servicios deben tener un nombre único. Cuando un nodo tiene un servicio, todos los nodos se pueden comunicar con él.
- Bolsas: Las bolsas son un formato para guardar y ejecutar la información de mensajes de ROS. Las bolsas son un mecanismo importante para guardar información, como sensores, que puede ser difícil de recolectar, pero es necesaria para desarrollar y probar algoritmos.

O. IMÁGENES MULTIESPECTRALES

1. Imagen y espectro. En el área de la óptica, una imagen se define como la reproducción de la figura de un objeto por la combinación de los rayos de luz que proceden de él (Diccionario de la lengua española, 2001). Por otro lado, un espectro se define como la distribución de la intensidad de una radiación en función de una magnitud característica, como la longitud de onda, la energía, la frecuencia o la masa (Diccionario de la lengua española, 2001). A partir de las definiciones anteriores se define a una imagen espectral como la reproducción de la figura de un objeto en función de la intensidad de radiación de una longitud de onda que esté reflejando el objeto en cuestión. Una imagen espectral puede referirse también al set de imágenes del mismo objeto, representadas cada una con diferentes longitudes de onda (Alava Ingenieros, 2015).

A partir de las definiciones anteriores surgieron los conceptos de imagen multiespectral e imagen hiperespectral. Las diferencias entre estos dos tipos de imágenes son múltiples, pero la principal y más importante reside en el número de bandas espectrales que cada una posee. Una imagen multiespectral está formada normalmente por un número pequeño de bandas, entre 3 y 20 bandas, y son bandas no necesariamente contiguas. Una imagen hiperespectral está formada normalmente por un número mayor de bandas respecto a una imagen multiespectral, pero en esta ocasión las bandas son siempre contiguas. Las imágenes multiespectrales son utilizadas en el análisis de valores de intensidad de longitud de onda discretas, mientras que las imágenes hiperespectrales son utilizadas en el análisis del espectro continuo del objeto de análisis (Alava Ingenieros, 2015).

Figura 17. Comparación entre los conceptos de imagen multiespectral e imagen hiperespectral

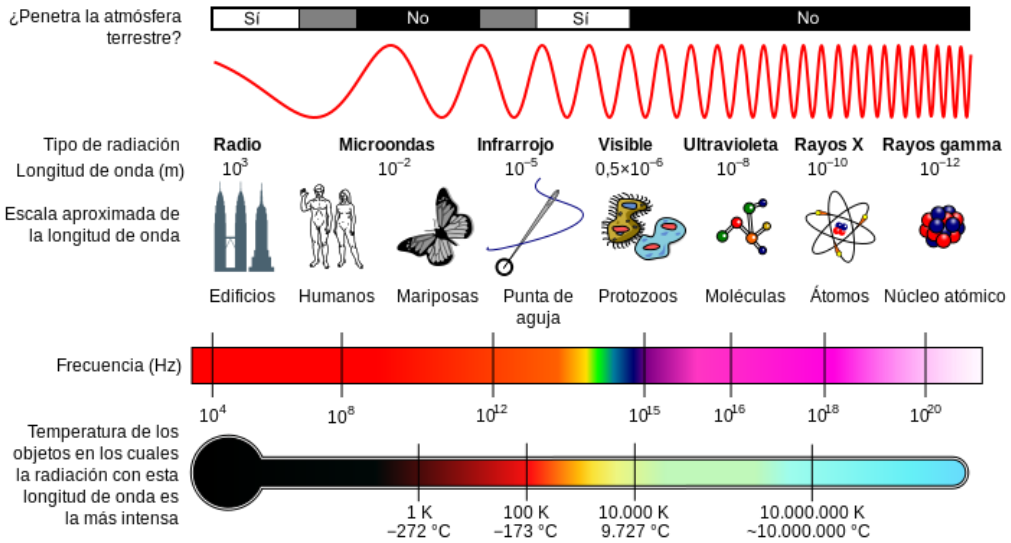


Las cámaras en color proporcionan normalmente tres bandas de información en cada imagen (rojo, verde y azul), pues de esta forma tratan de imitar el proceso de visión del sistema humano. El uso de un número mayor de bandas, o de bandas distintas a las habituales, permiten representar una escena totalmente distinta (INIT, 2011). Es por ello que a través de una imagen multiespectral o hiperespectral se puede apreciar de una

mejor forma la distribución espacial de la respuesta espectral de los elementos que componen a una escena (Lira, 2010). Dependiendo del tipo de sensor utilizado para la captura de las imágenes será la cantidad de bandas disponibles para su análisis. Las primeras aplicaciones que contaron con este tipo de sensores fue la observación de la tierra. De esta forma, sensores de este tipo eran instalados en satélites para el análisis de cambios climáticos, monitorización de cultivos y de ayuda en estudios arqueológicos (INIT, 2011).

2. Espectro electromagnético. Se denomina espectro electromagnético a la distribución de la intensidad de radiación o energía del conjunto de radiaciones u ondas electromagnéticas posibles. El espectro electromagnético se extiende desde las frecuencias bajas, con longitud de onda larga, hasta las frecuencias altas, con longitud de onda corta. Se piensa que la longitud de onda corta está en las cercanías de la longitud de Planck, mientras que el límite de la longitud de onda larga es el tamaño del universo (Pérez, 2014).

Figura 18. Diagrama del espectro electromagnético



El ojo humano es sensible únicamente a la porción del espectro electromagnético denominado espectro visible, el cual se extiende desde la longitud de onda de 0.4 µm, correspondiente al color azul, hasta la de 0.7 µm, correspondiente al color rojo. Sin embargo, gracias a la instrumentación actual se es capaz de detectar objetos pertenecientes al espectro no visible. Los sensores remotos pasivos, *i.e.* aquellos que no irradian energía para fines de detección, empleados en el estudio de recursos naturales operan en cuatro bandas del espectro electromagnético (Fallas, 2004):

- Visible (λ = 0.4um a 0.7um)
- Infrarrojo cercano (λ = 0.7 μm a < 1.3 μm)
- Infrarrojo medio (λ = 1.5 μm a 2.4 μm)
- Infrarrojo Térmico (λ > 5.3 μm)

Los sensores existentes son capaces de registrar la energía reflejada y/o emitida por la superficie terrestre en dichas bandas del espectro electromagnético. La banda correspondiente a una longitud de onda entre 2.5 μm y 4.5 μm no es utilizada pues se ve afectada por la absorción de CO2. La capacidad de un sensor de registrar el calor de un objeto está directamente relacionada con su habilidad de registrar ondas en la porción del infrarrojo térmico. A diferencia del infrarrojo térmico, el infrarrojo cercano y medio no son capaces de registrar el calor de un objeto (Fallas, 2004).

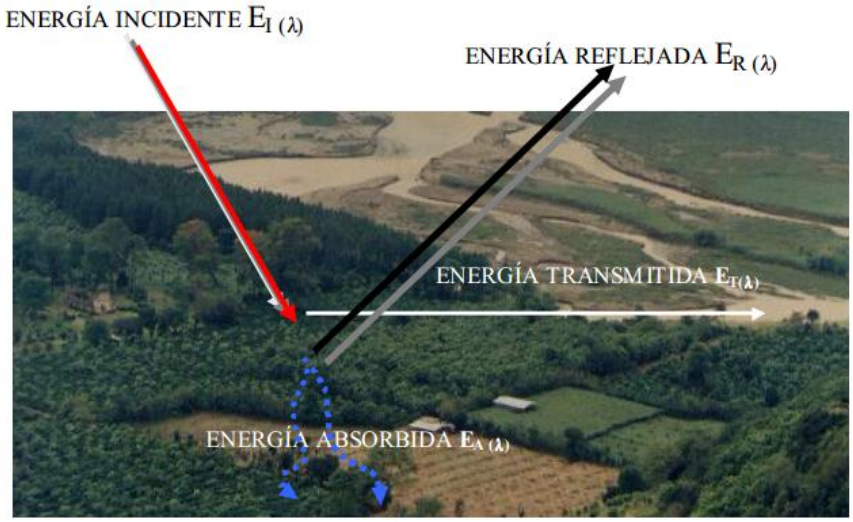
a. Energía electromagnética. La energía electromagnética en una longitud de onda particular λ, en el vacío, posee una frecuencia asociada f y una energía fotónica E. Es por ello que el espectro electromagnético puede expresar en término de cualquiera de estas tres variables, pues se encuentran relacionadas mediante expresiones matemáticas. Las ondas electromagnéticas de alta frecuencia poseen una longitud de onda corta y una alta energía, mientras que las ondas de baja frecuencia poseen una longitud de onda larga y una baja energía (Pérez, 2014).

El principio de conservación de la energía nos dice que la energía no se crea ni se destruye, únicamente se transforma. Debido a ello, la energía que llega a un cuerpo terrestre, *i.e.* irradiancia, se puede expresar de la siguiente forma:

$$E_{I(\lambda)} = E_{R(\lambda)} + E_{A(\lambda)} + E_{T(\lambda)}$$

En donde $E_{I(\lambda)}$ es la energía que llega al cuerpo o superficie, $E_{R(\lambda)}$ la energía reflejada por el cuerpo o superficie, $E_{A(\lambda)}$ la energía absorbida por el cuerpo o superficie y $E_{T(\lambda)}$ la energía transmitida por el cuerpo o superficie.

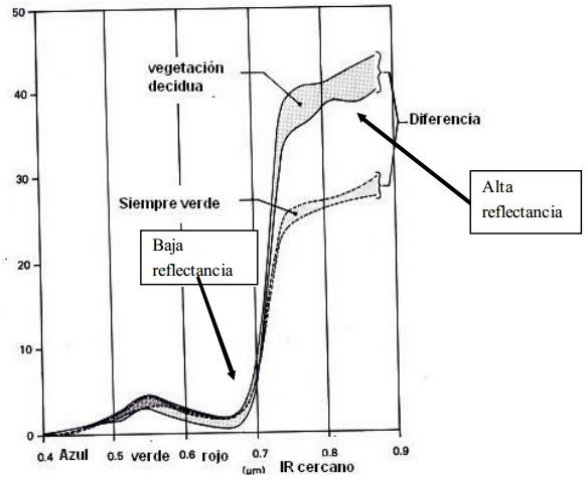
Figura 19. Relación entre energía incidente, reflejada y absorbida por la superficie terrestre



Algunos autores afirman que la luz incidente en los objetos causa vibración en las moléculas, la cual absorbe un tipo de energía, a una longitud de onda específica, y refleja otras, reflejando así un espectro que permite estimar la cantidad de unas y otras moléculas presentes en el objeto o muestra (Botero, 2009). Entonces, la cantidad de energía reflejada, absorbida y transmitida se encuentra en función de la composición de la superficie y su estado. A partir de la respuesta de una superficie se es posible identificar qué tipo de cuerpo o superficie se trata. La interacción entre la energía incidente y la respuesta de la superficie es específica para cada longitud de onda. Por tanto, un mismo cuerpo o superficie experimentara distintas respuestas en cada banda del espectro electromagnético (Fallas, 2004).

El sistema visual del ser humano utiliza la diferencia en reflectividad de los objetos para asignarles un color y a partir de esto formar una imagen. De forma análoga, se han creado sensores capaces de registrar la reflectancia de ciertas bandas del espectro electromagnético, a partir de las cuales se ha sido posible construid curvas de reflectividad. Se le denomina curva de reflectividad a la gráfica de reflectividad en función de la longitud de onda (λ), y es característica de cada material o tipo de cobertura. A partir de las curvas de reflectividad se es posible detectar patrones, establecer comparaciones y caracterizar un material o tipo de cobertura. Las curvas de reflectividad son bastante útiles para el análisis espectral de un cuerpo o superficie (Fallas, 2004). La Figura 4 es un ejemplo de cómo se ven las curvas de reflectividad:

Figura 20. Respuesta espectral de una conífera y de una latifoliada



3. Espectroscopia. La espectroscopia nació como el estudio de la interacción entre la intensidad de la radiación y la materia en función de la longitud de onda (λ) (Pérez, 2014). El principio fundamental de la espectroscopia es que la radiación sufre modificaciones observables y mediables al interactuar con las sustancias y lo cual depende la composición química, atributos físicos y estructurales del cuerpo o superficie (Botero, 2009).

En principio se hacía referencia únicamente al uso del espectro visible dispersado según su longitud de onda, *e.g.* la luz a través de un prisma. Posteriormente el concepto se amplió para el análisis de todo el espectro electromagnético. A partir de lo anterior, la espectroscopia puede entenderse como la respuesta espectral a un campo alternante o frecuencia variante. En el campo de la espectroscopia se conoce como espectro al gráfico de la respuesta espectral, *i.e.* intensidad de radiación, como función de la longitud de onda (Pérez, 2009).

Algunos sensores instalados en plataformas satelitales están diseñados para registrar la cantidad de energía reflejada por la superficie terrestre. Cada cuerpo o elemento en la superficie terrestre puede ser identificado por su respuesta espectral, también conocida en algunos textos como firma espectral. En términos de condiciones iniciales, este concepto es análogo al de la huella digital del ser humano. Sin embargo, a consecuencia del efecto atmosférico, del relieve y de errores propios de instrumentación existe cierto grado de variabilidad en la respuesta espectral de una superficie específica. Es por ello que en la mayoría de los casos la respuesta espectral es difusa y se encuentra en un rango de valores. Para ello se registra la energía reflejada para diferentes longitudes de onda y es con este set de resultados que un analista es capaz de asignar una etiqueta a un objeto o elemento sobre la superficie terrestre (Fallas, 2004).

a. Espectroscopía vegetal. La vegetación sana tiene su máxima reflectividad en el espectro visible a una longitud de onda de 0.55 µm, correspondiente al color verde, así como una alta reflectividad en el infrarrojo cercano o Neer Infra-Red (NIR por sus siglas en inglés), el cual corresponde al espectro no visible. La alta reflectividad de las hojas en el espectro visible correspondiente al verde se debe a su pigmentación. La clorofila, xantofila y los carotenos absorben energía a una longitud de onda de aproximadamente 0.445 µm, correspondiente al color azul, y 0.645 µm, correspondiente al color rojo (Fallas, 2004).

Por otra parte las hojas turgentes, debido a la estructura del mesófilo, poseen una alta reflectividad a una longitud de onda de 0.645 µm, correspondiente al color rojo, así como una alta reflectividad en el infrarrojo cercano (0.7 µm a 1.3 µm). Sin embargo, la reflectividad en la banda correspondiente al rojo es menor a la registrada en la banda correspondiente al verde, pues de igual forma absorbe energía en la banda correspondiente al rojo. La vegetación exhibe una baja reflectividad en el infrarrojo medio (1.4 µm a 3 µm) debido a que la energía es absorbida por el agua contenida en sus tejidos. A la región anterior se le conoce como banda de absorción de agua. En el caso que exista un elemento en el medio que reduzca el contenido de humedad en la planta, *e.g.* reducción en la evapotranspiración y el estrés por falta de agua, incrementará la reflectividad en esta longitud de onda. Entre más alta sea la reflectividad de las bandas rojo e infrarrojo cercano mayor será el vigor de la vegetación y más fácil será diferenciarla de otros tipos de cobertura. Lo anterior hace de estas dos bandas excelentes indicadores de una vegetación sana (Fallas, 2004).

El estrés experimentado en las plantas reduce la producción de clorofila y con ello se disminuye la absorción de las bandas correspondientes al azul y rojo, aumentando a su vez la reflectividad de la banda correspondiente al rojo. Es por ello que visualmente se detecta que una planta se torna amarilla, el cual resulta ser una combinación de la reflectividad del verde y del rojo. La vegetación sana y turgente refleja entre un 40% y un 50% la energía que recibe en la banda del infrarrojo cercano, por lo cual tienen una apariencia brillante en imágenes infrarrojas (Fallas, 2004).

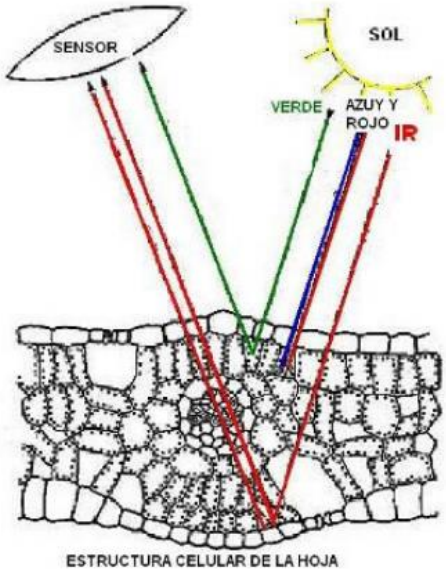
Las propiedades estructurales y el estado fisiológico de la vegetación explican su apariencia en imágenes captura por sensores multiespectrales. Para una vegetación sana se tienen las siguientes observaciones (Fallas, 2004):

- Tonos oscuros en la banda correspondiente al color azul y rojo (baja reflectancia)
- Tonos un tanto más claros en la banda correspondiente al color verde (reflectancia media)
- Tonos brillantes en las bandas del infrarrojo cercano (alta reflectancia)

Existen algunos factores a considerar al identificar cultivos en una imagen multiespectral:

- Tamaño y forma del cultivo
- Tamaño de las hojas, altura de la planta, sombra, espaciamiento, geometría de la plantación
- Ciclo vegetativo del cultivo o planta
- Índice de área foliar (IAF), el cual se refiere a la cobertura de un lado de la hoja de la planta por unidad de área. La reflectividad aumenta con el aumento en el índice de área foliar. La máxima absorción de energía, *i.e.* cuando se tiene reflectancia mínima, en el espectro visible se da cuando el IAF alcanza un valor entre 2 y 3, mientras que para la energía infrarroja cercana se obtiene una máxima absorción cuando la vegetación alcanza un IAF entre 6 y 8.

Figura 21. Relación entre la estructura de la hoja y su reflectividad en diferentes bandas del espectro electromagnético



La curva de reflectancia de una superficie seca tiene menos puntos de inflexión con respecto a la curva espectral de una superficie con vegetación turgente. Esto se debe a que se tiene una estructura menos compleja del suelo. En un tipo de suelo seco se suele registrar un comportamiento monótono en las bandas correspondientes al espectro posible e infrarrojo cercano. Por otro lado, el contenido de humedad del suelo y su textura afectan las curvas espectrales en el área correspondiente a la banda de absorción de agua mencionada anteriormente. Los suelos que poseen un bajo nivel de humedad poseen una alta reflectancia en todas las bandas del espectro visible, mientras que los suelos con un alto nivel de humedad tienden a crear imágenes multiespectrales con colores más oscuros (Fallas, 2004).

La curva de reflectancia del agua se caracteriza por su banda de absorción de agua, correspondiente a la banda del infrarrojo medio. En general, el agua absorbe la energía en todas las bandas del espectro visible, por lo cual tendería a lucir negro en una fotografía normal. Sin embargo, el contenido de sedimentos en suspensión en el agua modifica su reflectancia. A mayor cantidad de sedimentos en suspensión se tendrá una mayor reflectancia en el espectro visible, por lo cual lucirá de un color más claro en una fotografía. El agua absorbe en mayor cantidad la energía radiada por las bandas correspondientes al verde y rojo, por lo cual se suele observar al agua en un tono azulado o verde azulado. El tono verde se debe a que la clorofila de las algas absorbe la energía proveniente de las bandas correspondientes al azul y rojo, reflejando la banda correspondiente al verde (Fallas, 2004).

b. Espectrometría en el campo agrícola. La espectrometría se define como la técnica espectroscópica para tasar la concentración o la cantidad de una substancia mediante el espectro emitido por un cuerpo o superficie. En este escenario, el instrumento que realiza dichas mediciones es conocido como espectrómetro. La espectrometría es utilizada especialmente en el área de detección remota o teledetección para medir la composición química y propiedad físicas de un objeto de estudio (Pérez, 2014).

La primera vez que se utilizó la espectroscopia NIR, *i.e.* empleando para ello la banda correspondiente al infrarrojo cercano, en el área de la agricultura fue en 1964, con el fin de determinar el contenido de humedad en cereales. A partir de ese momento se ha venido utilizando por ser una metodología de análisis precisa y rápida en la determinación de humedad, proteína y grasa, cantidad de sólidos solubles y cantidad de materia seca en diversos alimentos y productos agrícolas. Estos estudios se han basado principalmente en las técnicas de PLSR (Partial Least Square Regression o Regresión Parcial de Mínimos Cuadrados) o MLR (Multiple Linear Regression o Regresión Linear Múltiple), las cuales se basan en la correlación que existe entre la información que provee la respuesta espectral y la concentración de la molécula de interés. En la industria frutícola la técnica de espectroscopia VIS-NIR (Visible and Near Infrared) ha sido utilizada para determinar algunos parámetros de calidad. Gracias a la calibración exitosa y adecuada de los modelos y sensores utilizados, esta técnica se logró utilizar de forma continua durante meses para realizar análisis de grandes cantidades de fruta a relativos bajos costos (Botero, 2009).

1) Espectrometría y suelos. La espectroscopía de reflectancia también ha resultado ser un método alternativo para realizar análisis de suelos para estimar un gran número de propiedades en el suelo, los cuales eran realizados comúnmente de forma química y física en laboratorios tradicionales. Algunos autores afirman que la espectroscopía NIR es una metodología eficaz en la determinación la concentración de nitrógeno y carbono en materiales de origen vegetal, siendo una técnica muy eficiente desde el punto de vista económico. Es así que algunos investigadores han montado sensores

de este tipo sobre tractores como insumo para proceso de agricultura de precisión y para el mapeo de distintas características del suelo (Botero, 2009).

2) Espectrometría y sanidad vegetal. Así como muchas otras variables, el estrés sanitario en una planta es también detectable mediante el uso de espectroscopía NIR. Algunos autores afirman que el estrés patológico en una planta se puede obtener a través de índices de vegetación, ya sea con anchos de banda amplios o estrechos en el espectro electromagnético. Dichos autores emplearon el análisis de componentes independientes (Independent Component Analisys o ICA por sus siglas en inglés) para separar señales estadísticamente independientes, extrayendo información a través del análisis de componentes principales. Además, utilizando la técnica de análisis basado en vector característico (Feature Vector Based Analisy o FVBA por sus siglas en inglés), el cual produce pares de componentes de vectores característicos que representan firmas espectrales y sus respectivos coeficientes, se encontraron coeficientes que resultaron ser proporcionales a la severidad de enfermedades fungosas medidas en campo y al porcentaje de área foliar necrótica, obteniéndose correlaciones bastante altas (Botero, 2009).

Investigaciones posteriores demostraron que se es posible identificar enfermedades en cultivos por medio de la reflectancia y poder diferenciar zonas en cuanto a la severidad de la enfermedad. Esto vuelve a la percepción remota por medio de la espectrometría de reflectancia en una poderosa herramienta para el análisis geoespacial y geo temporal de las características de los cultivos, incluyendo la sintomatología causada por enfermedades (Botero, 2009).

3) Espectrometría y humedad. Uno de los objetivos en la utilización de sensores y percepción remota ha sido la detección del estrés hídrico, tanto por exceso como por defecto, con el fin de implementar sistemas de riego más sofisticados incluyendo a su vez sistemas de drenajes. En algunas investigaciones se lograron detectar niveles leves de estrés hídrico en plantaciones de olivos a través del uso de espectrometría de reflectancia, específicamente a través de la radiación térmica como indicador de algunos parámetros de calidad de frutales y su productividad a libre exposición. Investigaciones posteriores estudiaron el efecto del estrés por agua sobre la reflectancia termal, evaluando la relación entre los síntomas visuales y térmicos del estrés en función del tiempo y el contenido de humedad en ella. Estos estudios concluyeron que mediante imágenes térmicas espectrales se es posible detectar el estrés por sequía e inundación (Botero, 2009).

4) Espectrometría y nutrición vegetal. La capacidad de estimar la concentración de nutrientes en un tejido foliar es una de las aplicaciones más prometedoras en el campo de la espectrometría de reflectancia. Esto se debe a la capacidad de aumentar el número de muestras no solo en investigación, sino también a nivel productivo, con lo cual se podría llegar a optimizar los procesos de producción agrícola y hacer de estos una vía más rentable. Análisis mediante la medición de la bioquímica foliar resultan bastante costosos, y además por ser análisis de laboratorio resultan ser tardados y limitados espacialmente. Por otro lado, la espectroscopía es capaz de brindar un monitoreo del estado ambiental eficiente tanto en espacio como en tiempo. Los cambios en el espectro de reflectancia en las hojas sirven como indicadores de deficiencias nutricionales, siendo rápidos, no destructivos y relativamente económicos (Botero, 2009).

La estimación del nitrógeno foliar mediante espectrometría se ha visto limitada por la presencia de agua en los follajes frescos, ya que enmascara las absorciones en la banda correspondiente al infrarrojo medio. Además, debido a la orientación de las hojas, el efecto de fondo del suelo y la interferencia atmosférica han impedido lograr una estimación precisa. Puesto que el nitrógeno es el nutriente más utilizado en los procesos de fertilización una gran cantidad de trabajo se han centrado en el efecto que este tiene en la espectrometría de reflectancia y como determinar su concentración a nivel foliar en diferentes cultivos. En la actualidad la fertilización en cereales está caracterizada por una baja eficiencia en uso del nitrógeno, básicamente debido a tres causas. La primera de ellas es la baja sincronía entre la fertilización y la demanda del cultivo. La segunda es la aplicación uniforme en los cultivos cuando se sabe bien que existe variabilidad en la necesidad de los cultivos de acuerdo a la variabilidad espacial de las propiedades físico químicas de los suelos. Por último se falla en la valoración temporal de la demanda de nitrógeno de los cultivos (Botero, 2009).

Es por ello que se busca la implementación de sensores y sistemas de fertilización de respuesta inmediata capaces de valorar la necesidad de nitrógeno de una determinada planta para aplicar dosis exactas en zonas donde se tiene un manejo uniforme. Aproximadamente el 10% del nitrógeno total de las plantas se encuentra almacenado en moléculas de clorofila, por lo cual existe cierta correlación entre la reflectancia espectral y el contenido de nitrógeno de las hojas. De forma que se es posible que los productores utilicen los medidores de clorofila como un indicador del estrés causado por el nitrógeno en las plantas (Botero, 2009).

a. Ventajas de la espectrometría. La espectroscopia VIS-NIR tiene varias ventajas sobre los análisis tradicionales hechos en un laboratorio. Entre estos se tiene que físicamente es una técnica no destructiva, no invasiva y además requiere poca preparación de las muestras, obteniendo precisiones altas en la determinación. Además, se trata de una técnica rápida y de exactitud similar a las vías húmedas, pero evita la utilización de reactivos tóxicos o corrosivos en los procesos de análisis. Se trata también de una tecnología simple y de bajo costo, razón por la cual se utiliza en varios campos de la ciencia

como lo son la farmacéutica, agricultura, alimentos, textiles, cosméticos, etc. Presenta un enfoque versátil pues se puede aplicar a sólidos, fluidos y gases (Botero, 2009).

Los equipos para la medición de reflectancia disponibles en el mercado se han adaptado en diferentes campos de la ciencia y la industria para la implementación de técnicas de espectroscopia. Esto ha permitido la determinación de varios parámetros, de forma simultánea, continua, en tiempo real y fácilmente automatizables. Además, los avances en la espectroscopia permiten a las industrias e investigadores incrementar la complejidad de sus procesos y experimentos, obteniendo información cuantitativa más precisa de los productos de interés en tiempo real y de forma rápida y sencilla (Botero, 2009).

b. Incertidumbre espectral. La espectroscopia VIS-NIR presenta grandes ventajas en varios campos de la ciencia e industria además de presentar altos niveles de precisión y exactitud en la determinación de muchas propiedades en diferentes objetos y materiales. Sin embargo, muchos autores se han encontrado que en ocasiones la calibración de datos puede presentar ambigüedad o incertidumbre en la identificación o clasificación de alguna propiedad. Esto se debe a que las características no son exclusivas de ciertas longitudes de onda, sino que pueden estar relacionadas en muchas longitudes de onda comunes (Botero, 2009).

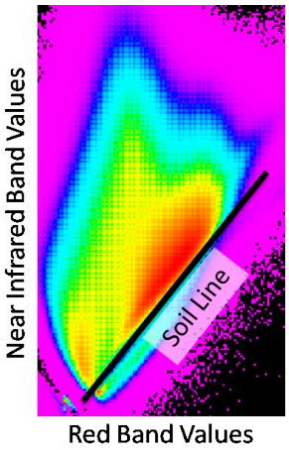
Algunas investigaciones han afirmado que es difícil separar espectralmente entre plantas sometidas a estrés hídrico y estrés causado por infestación del áfido verde, considerando la coexistencia de ambas en la misma área. Además, se ha afirmado que otros tipos de estrés en plantas producidos por enfermedades, deficiencias nutricionales, o sequía pueden tener efectos similares en el espectro de reflectancia de las plantas, por lo cual es necesario profundizar en la diferenciación entre los distintos tipos de estrés (Botero, 2009).

En el caso de la nutrición vegetal, algunos autores presentan que el principal problema es que casi todos los nutrientes afectan la cantidad de clorofila en la hoja, lo que hace se afecte de manera similar el espectro de reflectancia en la banda del espectro visible. De modo que cuando se presenta estrés por alguno o varios nutrientes no se es posible utilizar únicamente la respuesta espectral para obtener una buena discriminación de las deficiencias de nitrógeno, fósforo y potasio. Para lograr identificar de forma correcta la causa de la respuesta espectral de una planta algunos autores afirman que es necesario incluir información espacial de la planta en los modelos de predicción. Dicha información incrementa la efectividad de la discriminación de manera significativa (Botero, 2009).

4. Índices de vegetación. Un índice de vegetación es un número generado por la combinación algebraica de la respuesta espectral de un cuerpo o superficie, la cual ha sido obtenida a partir de un sensor remoto multiespectral. El número generado por la respuesta espectral resulta tener cierta información en relación al cuerpo o superficie de interés. Los índices de vegetación existentes surgieron de forma empírica gracias a las investigaciones realizadas en el campo de la espectroscopia de reflectancia VIS-NIR (Salazar, 2007).

El uso de los índices de vegetación supone algunos escenarios un tanto ideales. Primero que nada se asume que toda el área de suelo desnudo en una imagen multiespectral forma una línea en la gráfica de reflectancia NIR-Rojo, conocida como la línea del suelo (soil line). Esta línea es considerada la línea que índica cero vegetación. Se debe resaltar que la línea del suelo es una línea hipotética en el espacio del espectro, la cual describe la variación del espectro en el suelo desnudo de la imagen (Salazar, 2007).

Figura 22. Ejemplo de una gráfica NIR-Rojo



Este tipo de imagen se realiza graficando la reflectancia en la banda correspondiente al color rojo con respecto a la reflectancia en la banda NIR para cada pixel en la imagen. Los colores en la gráfica representan cuantos pixeles poseen una combinación de Rojo sobre NIR. Colores cálidos representan una mayor cantidad de pixeles, mientras que colores fríos representan una menor cantidad de pixeles (Salazar, 2007).

a. RVI. El índice de vegetación de razón (Ratio Vegetation Index o RVI por sus siglas en inglés) fue uno de los primeros índices propuestos, alrededor del año 1969. Es uno de los índices de vegetación más utilizados alrededor del mundo, aunque rara vez se refieren a él como un índice de vegetación. Una práctica común en el área de teledetección es la utilización de razones de banda para eliminar algunos errores de coeficientes de reflexión. El índice RVI se caracteriza por tener un rango entre 0 e infinito,

lo cual lo hace poco práctico en algunas ocasiones. Muchas personas utilizan la razón entre las bandas de NIR y la correspondiente al color rojo y es de esta forma que se describe al índice RVI (Salazar, 2007):

$$RVI = \frac{NIR}{Red}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente.

b. NDVI. El índice de vegetación de diferencia normalizado (Normalized Difference Vegetation Index o NDVI por sus siglas en inglés) fue descrito alrededor del año 1973, aunque ya se venía hablando de índices de vegetación normalizados a partir del año 1969. Normalmente, cuando las personas se refieren al concepto de índice de vegetación éste es el índice al que están haciendo referencia por su gran popularidad. Este índice presenta una gran ventaja sobre el índice RVI, y es que varía entre los valores de -1 y 1, mientras que el índice RVI varía en un rango del 0 al infinito. Los índices RVI y NDVI son funcionalmente equivalentes y relación mediante la siguiente ecuación (Salazar, 2007):

$$NDVI = \frac{RVI - 1}{RVI + 1}$$

El índice NDVI se caracteriza por describir la actividad fotosintética de un determinado cultivo y es además uno de los índices de vegetación más utilizados mundialmente. Vegetación que se encuentra activa fotosintéticamente tiende a absorber, en particular, una mayor cantidad de energía emitida por la banda correspondiente al color rojo, mientras que refleja una gran cantidad de energía emitida por la banda NIR. Si se trata de vegetación muerta o bajo algún tipo de estrés ésta refleja una gran cantidad de energía emitida por la banda correspondiente al color rojo y refleja, en menor cantidad, energía emitida por la banda NIR. Además, regiones que no poseen algún tipo de vegetación tienden a reflejar en mayor cantidad la energía emitida por todo el espectro visible. Al igual que el índice RVI, el índice NDVI se puede describir a partir de la razón entre las bandas de NIR y la correspondiente al color rojo mediante la siguiente ecuación (Landscape Toolbox, 2013):

$$NDVI = \frac{NIR - Red}{NIR + Red}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente. La interpretación biofísica del índice NDVI es que éste describe la fracción de radiación fotosintética absorbida. Son muchos los factores que afectan la actividad fotosintética de una

planta. Entre ellos se encuentra la cobertura de vegetación, biomasa, estrés en la planta y la humedad, tanto en la planta como en el suelo. Es por ello que el índice NDVI esta correlacionado con muchas variables de un ecosistema que son de interés para investigadores y productores. Además, puesto que se trata de la razón de dos bandas distintas, el índice NDVI ayuda a compensar las diferencias en iluminación en una imagen debido al relieve o al tiempo del día en el que se capturó. De esta forma, índices de vegetación como el NDVI hacen posible comparar imágenes a lo largo del tiempo y buscar cambios ecológicos significativos (Landscape Toolbox, 2013).

Como se mencionó anteriormente, el índice NDVI adopta valores entre -1 y 1. Sin embargo, valores menos a cero suelen no tener un significado ecológico, por lo que los análisis se hacen en referencia al rango entre 0 y 1. Valores altos implican que hay una mayor diferencia entre la radiación de la banda NIR y la banda correspondiente al color rojo registrada por el sensor multiespectral. Esta condición se asocia a una vegetación con alta actividad fotosintética. Valores bajos implican que hay poca diferencia entre la radiación de la banda NIR y la banda correspondiente al color rojo. Esta condición se asocia a una vegetación con baja actividad fotosintética, o a escenarios en donde hay muy poca reflexión de la banda NIR, *e.g.* en el caso de los cuerpos de agua (Landscape Toolbox, 2013).

Debido a que se trata de un índice simple de calcular y a su correlación con muchas variables en un ecosistema, el índice NDVI se ha utilizado en numerosos ecosistemas pastizales. Entre los usos más frecuentes se encuentran el monitoreo y análisis de (Landscape Toolbox, 2013):

- Cambios fenológicos en las plantas a lo largo del tiempo
- Producción de biomasa
- Impactos debidos al uso de distintas técnicas de pastoreo
- Cambios en las condiciones de los pastizales
- Clasificación de la vegetación o tipo de superficie
- Humedad en el suelo
- Flujo de dióxido de carbono (CO2)

El índice NDVI esta correlacionado a un gran número de atributos que son de interés para investigadores y productores en el área de pastizales. Sin embargo, el índice NDVI no se trata de una medición directa de ninguno de dichos atributos. Mientras que el índice NDVI posee una interpretación biofísica (descrita anteriormente), existen muchos factores que influyen en la fuerte relación entre el índice NDVI y los atributos de un ecosistema. Entre dichos factores se encuentran las condiciones atmosféricas, escala de la imagen, humedad en la vegetación, humedad en el suelo, cobertura de vegetación, diferencias en el tipo de suelo de una región, etc. Es por ello que es importante que al realizar análisis en base al índice NDVI se tomen en

cuenta dichos factores, y de ser posible, controlar dichos factores que puedan afectar al índice NDVI antes de realizar cualquier tipo de interpretación (Landscape Toolbox, 2013).

La luz reflejada por la superficie del suelo influye en gran manera los valores calculados por el índice NDVI. Esto se volvió una gran preocupación, en especial en aplicaciones en pastizales pues existen muchas áreas áridas o semi-áridas. Algunas investigaciones llegaron a afirmar que superficie del suelo tiene un mayor impacto en los valores del índice NDVI en áreas en donde el 45% al 70% está cubierto por vegetación. Esta limitación fue la razón para el desarrollo de nuevos índices de vegetación capaces de ajustarse en base a las condiciones del suelo del área de interés (Landscape Toolbox, 2013).

En adición a la influencia de la superficie del suelo cuando se tiene áreas con poca vegetación, el índice NDVI sufre una pérdida de sensibilidad cuando la cantidad de vegetación es muy alta. A medida que la cantidad de vegetación crece, los cambios en el índice NDVI se van haciendo cada vez más pequeño. Entonces, a valores muy altos del índice NDVI, un pequeño cambio en el valor del índice puede deberse a un cambio bastante grande en la vegetación. Estos problemas en la sensibilidad del índice NDVI dificulta el análisis adecuado en áreas con una gran cantidad de actividad fotosintética (Landscape Toolbox, 2013).

c. IPVI. El índice de vegetación de porcentaje de infrarrojo (Infrared Percentage Vegetation Index o IPVI por sus siglas en inglés) fue desarrollado alrededor del año 1990. Algunos investigadores afirmaron que no era necesario realizar la resta en el numerador en el caso del índice NDVI, pues esto era irrelevante. En cambio, propusieron un nuevo índice de vegetación para mejorar el tiempo de cálculo, siendo este el índice IPVI. El índice IPVI se caracteriza por variar en un rango de valores entre 0 y 1, lo cual elimina la necesidad de almacenar el signo para describir al índice de vegetación. Los índices IPVI y NDVI son funcionalmente equivalente y están relacionados mediante la siguiente ecuación (Salazar, 2007):

$$IPVI = \frac{NDVI + 1}{2}$$

$$IPVI = \frac{NIR}{NIR + Red}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente.

d. EVI. El índice de vegetación mejorado (Enhanced Vegetation Index o EVI por sus siglas en inglés) fue desarrollado como un índice de vegetación alternativo para suplir las limitaciones del índice NDVI. Específicamente, el índice EVI fue desarrollado para (Landscape Toolbox, 2012):

- Ser más sensible a cambios en áreas con una alta biomasa
- Reducir la influencia de condiciones atmosféricas en los índices de vegetación
- Corregir las señales registradas por el dosel arbóreo

El índice EVI suele ser más sensible a las diferencias en el dosel arbóreo, fenología de las plantas y en la detección del estrés en diferencia al índice NDVI, que generalmente respondía únicamente a la cantidad de clorofila presente. El EVI es calculado mediante la siguiente ecuación (Landscape Toolbox, 2012):

$$EVI = 2.5 * \frac{NIR - RED}{NIR + C_1 * RED - C_2 * BLUE + L}$$

Donde NIR, RED y $BLUE$ hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo y al color azul, pero que han sido corregidas parcialmente en base a las condiciones atmosféricas. C_1, , C_2 y L son coeficientes encargados de terminar de corregir la ecuación en base a las condiciones atmosféricas. El índice EVI se caracteriza por variar en un rango entre 0 y 1, lo cual lo hace un índice sencillo de utilizar para realizar algún tipo de análisis (Landscape Toolbox, 2012).

e. PVI. El índice de vegetación perpendicular (Perpendicular Vegetation Index o PVI por sus siglas en inglés) fue desarrollado alrededor del año 1977. Este índice se caracteriza por ser un tanto sensible a las variaciones atmosféricas, por lo cual no se podría realizar un análisis a lo largo del tiempo, a menos que se posean datos para aplicar algún tipo de corrección atmosférica. El rango de valores en los que puede variar éste índice es entre -1 y 1. El PVI es calculado mediante la siguiente ecuación (Salazar, 2007):

$$PVI = \sin(a)\, NIR - \cos(a)\, Red$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente; y a es la diferencia entre el valor de NIR de cada pixel con respecto al valor de NIR de una región sin vegetación.

f. Índices que minimizan el efecto del suelo. Como bien se sabe no todos los suelos son iguales. Tipos de suelo distintos poseen una distinta respuesta espectral. Idealmente, lo índices descritos anteriormente asumen que el tipo de suelo en una imagen capturada por un sensor espectral es uniforme, sin embargo, este no siempre es el caso. Además, se sabe que cambios en la humedad del suelo afectan la respuesta espectral de la vegetación, dando como resultados índices de vegetación erróneos. La influencia del suelo en los índices de vegetación suele ser mayor cuando la cobertura de vegetación es muy baja (Salazar, 2007).

Debido al gran impacto que presentaba el suelo en los índices de vegetación se comenzó a desarrollar nuevos índices enfocados a minimizar el ruido introducido por los efectos del suelo. Al igual que los índices presentados anteriormente, estos índices se basan en la razón entre dos o más bandas del espectro electromagnético. Sin embargo, para reducir la influencia del suelo, estos índices son poco sensibles a los cambios en la cobertura de vegetación a bajos niveles de cobertura, en comparación al índice NDVI. Por otro lado, estos índices son más sensibles a las variaciones atmosféricas, en comparación al índice NDVI (Salazar, 2007).

1) SAVI. El índice de vegetación de suelo ajustado (Soil Adjusted Vegetation Index o SAVI por sus siglas en inglés) fue desarrollado alrededor del año 1988. Este índice viene a ser un híbrido entre los índices basados en la razón de dos o más bandas del espectro electromagnético y los índices perpendiculares, como lo es el PVI. El índice SAVI se describe mediante la siguiente ecuación (Landscape Toolbox, 2013):

$$SAVI = \frac{NIR - Red}{NIR + Red + L} * (1 + L)$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente; y L es un factor de corrección que varía entre 0, para vegetaciones muy densas, y 1, para vegetaciones poco densas. Para una cobertura de vegetación intermedia se utiliza un valor de 0.5. El desarrollo del índice SAVI se realizó en base a medidas realizadas en plantaciones de algodón y en pastizales, teniendo distintos tonos de suelo en área. El ajuste el factor L se realizó de forma empírica, empleando una metodología de prueba y error que permitiera obtener una misma respuesta espectral para los suelos oscuros y claros. El factor de $(1 + L)$ se utiliza para que el índice varíe en un rango entre -1 y 1. Para el caso específico en que $L = 0$, el índice SAVI se vuelve el índice NDVI (Landscape Toolbox, 2013).

2) TSAVI. El índice de vegetación de suelo transformado (Transformed Soil Adjusted Vegetation Index o TSAVI por sus siglas en inglés) fue desarrollado

alrededor del año 1989. Este índice asume que la línea del suelo (soil line) tiene una pendiente arbitraría y un intercepto, los cuales utiliza para ajustar el índice de vegetación. Sin embargo, el índice TSAVI también incluye un parámetro "X", para minimizar la influencia del suelo. Sin embargo, no existe una forma de calcular dicho parámetro, en cambio se suele usar un valor de 0.08 de acuerdo a la literatura. El índice TSAVI se describe mediante la siguiente ecuación (Salazar, 2007):

$$TSAVI = \frac{s(NIR - s * Red - a)}{(a * NIR + Red - a * s + X * (1 + s^2)}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente; s es la pendiente la línea del suelo (soil line); a es el intercepto de la línea del suelo (soil line); y X es un parámetro para minimizar la influencia del suelo, el cual posee un valor de 0.08 generalmente (Salazar, 2007).

3) MSAVI y MSAVI2. El índice de vegetación de suelo ajustado modificado (Modified Soil Adjusted Vegetation Index o MSAVI) fue desarrollado alrededor del año 1994. El índice SAVI posee una limitación, y es que el factor de corrección L para realizar el cálculo depende del nivel de cobertura de vegetación, el cual se pretende calcular utilizando el índice de vegetación, creándose un círculo vicioso. El índice MSAVI se caracteriza por proveer un factor de corrección L que es variable y se basa en el producto del índice NDVI y el índice WDVI, índice de vegetación de diferencias ponderado (Weighted Difference Vegetation Index o WDVI por sus siglas en inglés). Al igual que el índice SAVI, éste índice varía en un rango de valores entre -1 y 1. El índice MSAVI se describe mediante la siguiente ecuación (Landscape Toolbox, 2012):

$$MSAVI = \frac{(NIR - Red)(1 + L)}{NIR + Red + L}$$

$$L = 1 - \frac{2 * s * (NIR - Red) * (NIR - s * Red)}{(NIR + Red)}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente; y s es la pendiente la línea del suelo (soil line). Posteriormente surgió un segundo índice de vegetación de suelo ajustado modificado (Second Modified Soil Adjusted Vegetation Index o MSAVI2), el cuál consistió en una recursión del índice MSAVI. Básicamente, se realiza un proceso iterativo sustituyendo 1-MSAVI(n-1) como el factor de corrección L en el cálculo de MSAVI(n). Luego

resolvían la ecuación de forma recursiva hasta tener que MSAVI(n) = MSAVI (n-1). La solución a esta ecuación hizo surgir la siguiente fórmula, la cual elimina la necesidad de depender del valor de la pendiente de la línea del suelo (soil line) (Landscape Toolbox, 2012):

$$MSAVI2 = \frac{2 * NIR + 1 - \sqrt[2]{(2 * NIR + 1)^2 - 8 * (NIR - Red)}}{2}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente.

g. Índices que minimizan el efecto atmosférico. La atmósfera se encuentra cambiando todo el tiempo y los instrumentos empleados en la teledetección deben vencer este obstáculo. La atmósfera atenúa la luz que incide y además la dispersa a lo largo del espacio debido a aerosoles suspendidos. La atmósfera puede variar en gran manera a lo largo de una escena, especialmente en áreas con gran relieve. Todo esto altera la luz que es registrada por los sensores multiespectrales y causa variaciones en los valores calculados para índice de vegetación. Esto se vuelve un problema en particular al momento de analizar la evolución de los índices de vegetación a lo largo del paso del tiempo (Salazar, 2007).

Debido al gran impacto que presentaba los cambios atmosféricos en los índices de vegetación se comenzó a desarrollar nuevos índices enfocados a minimizar el ruido introducido por los cambios atmosféricos. Dichos índices tratan de remediar el problema sin la necesidad de utilizar datos corregidos atmosféricamente. Estos índices logran reducir la sensibilidad a los cambios atmosféricos a cambio de una pérdida en su rango dinámico. Además, son ligeramente menos sensible a los cambios en la cobertura de la vegetación, en comparación al índice NDVI. A bajos niveles de cobertura vegetal, estos índices se vuelven muy sensibles al efecto del suelo (Salazar, 2007).

1) GEMI. Indice de monitoreo global ambiental (Global Environmental Monitoring Index o GEMI por sus siglas en inglés) fue desarrollado alrededor del año 1991. Este índice trata de eliminar la necesidad de poseer información detallada para realizar la corrección atmosférica. Para ello el índice calcula el factor de corrección en base a la respuesta espectral de la fotografía y de algunos coeficientes que fueron hallados de forma empírica en las investigaciones realizadas. El índice GEMI se caracteriza por variar en un rango de valores entre 0 y 1. Además, se trata de un índice no linear. El índice GEMI se describe mediante la siguiente ecuación (Salazar, 2007):

$$GEMI = eta * (1 - 0.25 * eta) - \frac{Red - 0.125}{1 - Red}$$

$$eta = \frac{2 * (NIR^2 - Red^2) + 1.5 * NIR + 0.5 * Red}{NIR + Red + 0.5}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente.

2) ARVI. El índice de vegetación resistente atmosféricamente (Atmospherically Resistant Vegetation Index o ARVI por sus siglas en inglés) fue desarrollado alrededor del año 1992. Al igual que el índice GEMI, el índice ARVI elimina la necesidad de poseer información detallada para realizar la corrección atmosférica. En vez de ello toma como base al índice NDVI y realiza una sustitución en el término correspondiente a la intensidad de reflexión por la banda correspondiente al color rojo (Salazar, 2007):

$$rb = Red - gamma(Blue - Red)$$

Donde Blue y Red hacen referencia a la intensidad de reflexión en las bandas correspondientes al color azul y rojo, respectivamente; gamma es un factor de corrección que generalmente asume un valor de 1. Algunos investigadores han sugerido realizar dicha sustitución en otros índices de vegetación como el SAVI, MSAVI2, TSAVI. Lo anterior daría como resultado la existencia de índices poco susceptibles a los efectos del suelo y a efectos atmosféricos. El índice ARVI se calcula mediante la siguiente ecuación (Salazar, 2007):

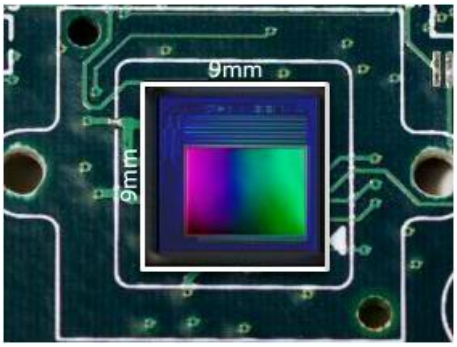
$$ARVI = \frac{NIR - rb}{NIR + rb}$$

Donde NIR hace referencia a la intensidad de reflexión en la banda de NIR.

P. FOTOGRAFÍA MULTIESPECTRAL

1. Funcionamiento de una cámara digital. En una cámara digital el lente forma una imagen de la escena a través un chip electrónico conocido como sensor de imagen. Un sensor de imagen es capaz de medir la intensidad de la luz reflejada y posteriormente la convierte a una imagen digital. Dicho sensor funciona de forma similar a la retina presente en el ojo humano (Bigshot, 2013).

Figura 23. Ejemplo de un sensor utilizado en una cámara digital para la medir la intensidad de la luz reflejada



En la retina del ojo se encuentra una malla densa formada por celdas sensibles a la luz, las cuales convierten la luz incidente en señales eléctricas. Estas señales son transportadas al cerebro, en donde son interpretadas como imágenes. De forma similar, los sensores de imagen empleados en una cámara digital poseen una malla de detectores llamados picture elements, o pixeles como se conocen comúnmente. Las dimensiones de un pixel se encuentran normalmente en el orden de lo micrómetros. El número de pixeles en una malla se conoce como la resolución del sensor de imagen (Bigshot, 2013).

Figura 24. Ejemplo de una malla de pixeles en un sensor a través de un microscopio

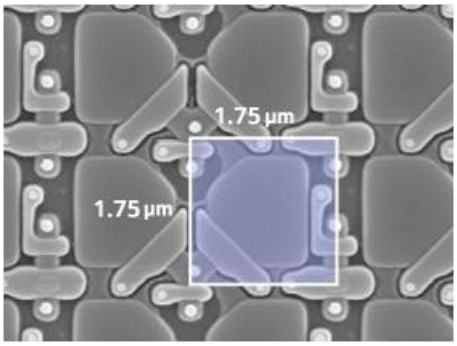
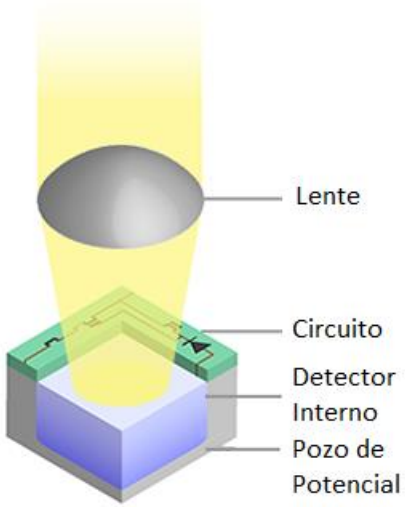


Figura 25. Estructura interna de un pixel



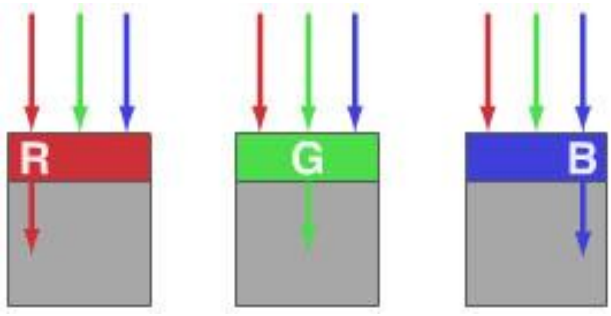
El detector interno en un pixel, mostrado en la Figura 9, está hecho generalmente de silicón y es sensible a luz. Cuando el pixel es expuesto a la luz, una cantidad de energía es transferida a los electrones presentes dentro de los átomos de silicón. Si la energía es suficiente, los electrones comienzan a desprenderse de los átomos. Esto es conocido como el efecto fotoeléctrico. Los átomos libres son recolectados en el pozo de potencial justo debajo del detector interno. El número de electrones libres, *i.e.* el número de electrones dispuestos en el pozo de potencial, depende directamente de la intensidad de la luz percibida por el pixel. Por ello, el voltaje en el pozo de potencial es una medida del brillo de la imagen en el pixel (Bigshot, 2013).

La mayoría de las cámaras digitales actuales utilizan un sensor de imagen CMOS (Complementary Metal Oxide Semiconductor). En un sensor CMOS cada pixel posee su propio circuito encargado de medir el voltaje en el pozo de potencial. Para evitar que la luz incida en la circuitería, se coloca un micro lente en cada pixel encargado de enfocar la luz incidente lejos de la circuitería y en dirección al detector. Los voltajes generados en toda la malla de pixeles son muestreados uno por uno y convertidos a un número a través de un convertidor analógico-digital (Analog to Digital Converter o ADC por sus siglas en inglés). El resultado final del muestreo es un arreglo bidimensional de números conocido como imagen digital (Bigshot, 2013).

El funcionamiento del pixel descrito anteriormente es capaz de detectar únicamente la intensidad de luz incidente en el pixel, sin embargo, éste no es capaz de diferenciar colores. A este tipo de pixeles se les conoce como ciegos al color (color blind). La forma en que un pixel es capaz de medir la intensidad de un color específico es a través de la implementación de filtros. Un filtro es un plástico traslucido que deja pasar únicamente una banda del espectro electromagnético. Al colocar un filtro en frente de un pixel, este detecta únicamente la intensidad de luz de una banda en específica, *i.e.* de un color en específico. Un pixel puede utilizar un único filtro a la vez, por lo cual no es posible detectar varios colores en un solo pixel de forma

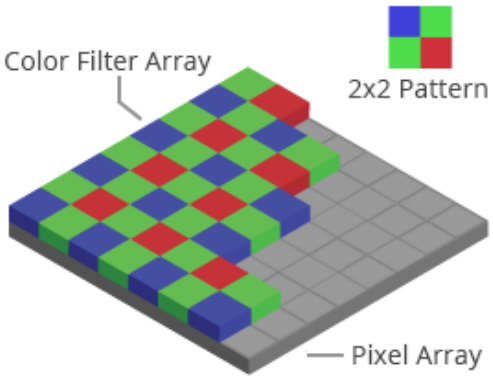
simultánea. Sin embargo, casi todos los colores pueden ser formados mezclando los tres colores primarios de luz, *i.e.* el color rojo, verde y azul. Los sensores de imagen usan tres tipos de filtros para crear imágenes a color, cada uno para detectar uno de los colores primarios en específico (Bigshot, 2013).

Figura 26. Pixeles utilizando filtros de distintos colores



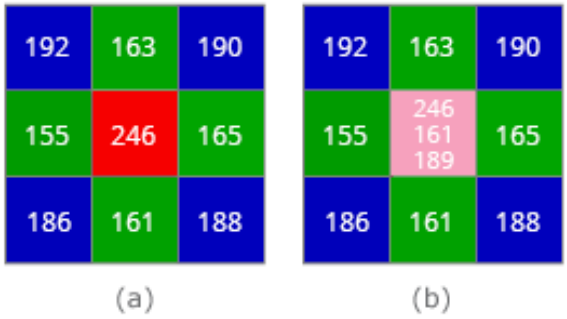
Puesto que un pixel no puede medir de forma simultánea los tres colores primarios, la mayoría de sensores de imagen emplean mosaicos que contienen pixeles con los tres tipos de filtros disponibles. Un diseño de uso popular es el mosaico Bayer mostrado en la Figura 11 (Bigshot, 2013):

Figura 27. Arreglo de pixeles empleando el mosaico Bayer



Puesto que cada pixel posee únicamente la información del color dada por su respectivo filtro, el pixel emplea a sus vecinos para determinar el valor de los otros dos colores. Los dos valores desconocidos son estimados utilizando la medición hecha por los vecinos. A este proceso se le conoce como interpolación de colores. El proceso de interpolación de colores se hace mediante el promedio de los valores obtenidos por los vecinos, para cada uno de los valores desconocidos por un pixel. A partir de esta técnica se es posible obtener la intensidad de incidencia de los tres colores para cada pixel en la malla del sensor de imagen (Bigshot, 2013).

Figura 28. Ejemplo de interpolación de colores



En la Figura 12, el pixel rojo en el centro encuentra los valores correspondientes para el color verde y azul realizando el promedio de los valores registrados por sus vecinos para esos dos colores.

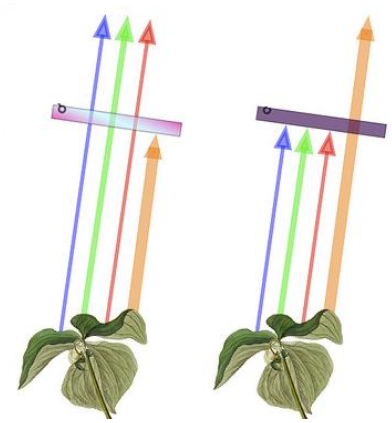
2. Fotografía infrarroja. El estudio ambiental de la tierra desde el espacio empezó en el año 1972, año en que el satélite Landasat fue lanzado. El satélite Landsat contaba con un scanner multiespectral que capturaba imágenes tanto en el espectro visible como en el espectro del infrarrojo cercano (PublicLab, 2015). Los investigadores en teledetección se percataron que al combinar la información del espectro visible e infrarrojo cercano se podía recopilar información crítica de la salud de la vegetación presente en la imagen a través de la implementación de algoritmos conocidos como índices de vegetación. Dichos índices permitían medir la densidad de follaje alrededor del globo. Posteriormente, combinando la información recopilada diariamente por los índices de vegetación en períodos de 8, 16 o 30 días, los científicos eran capaces de crear mapas detallados de la densidad del follaje en la tierra e identificar en que regiones las plantas se encontraban bajo estrés (Weir, 2000).

El campo de la fotografía infrarroja, *i.e.* como se conoce comúnmente a la captura de imágenes tanto en el espectro visible como en el espectro del infrarrojo cercano, se ha vuelto muy popular en el campo de la agricultura y la asesoría ecológica, especialmente en viñedos, plantaciones extensas y en proyectos de investigación a gran escala. El departamento de agricultura de estados unidos brinda algunos recursos de fotografía infrarroja, de acceso público, a través de las instituciones de NAIP y Vegscape. Sin embargo, las imágenes no son recolectadas de forma periódica, ni se encuentran a una escala utilizable para el estudio de localidades específicas. Además, la captura de este tipo de imágenes por cuenta propia resulta muy costoso, por lo cual no todos los agricultores pueden hacer uso de esta tecnología (PublicLab, 2013).

Los sensores de imagen utilizados en una cámara digital común son sensibles al reflejo del espectro infrarrojo. Sin embargo las cámaras digitales poseen un filtro que bloquea dicho espectro para que el sensor de imagen reciba correctamente el espectro visible y las fotografías se vean normales al público. Al remover

dicho filtro se es capaz de captar la respuesta espectral de la banda NIR. Sin embargo, al remover dicho filtro se obtiene una cámara capaz de detectar únicamente el espectro de la banda NIR, perdiendo la información del espectro visible. Este efecto se puede observar en la Figura 13 (PublicLab, 2015):

Figura 29. Efectos del filtro NIR en un sensor de imagen



A la izquierda en la Figura 13 se encuentra un sensor con filtro NIR, dejando pasar el espectro visible pero no el de la banda NIR. A la derecha se encuentra un sensor sin filtro NIR, dejando pasar el espectro de la banda NIR pero no el espectro visible (PublicLab, 2015).

3. Public Lab. El Laboratorio Público para Tecnología Libre y Ciencia (Public Laboratory for Open Technology and Science o Public Lab como se conoce comúnmente) es una comunidad sin fines de lucro que desarrolla e implementa herramientas de uso libre para la exploración e investigación ambiental. Mediante el desarrollo de proyectos Do-It-Yourself (DIY por sus siglas en inglés), Public Lab crea redes de colaboradores que activamente proponen nuevas ideas para el desarrollo del medio ambiente. El programa de Public Lab se enfoca en el concepto de ciencia cívica, en el cual se investiga el desarrollo de hardware y software libre para crear métodos que generen conocimiento y compartan información acerca de la salud ambiental de la comunidad (PublicLab, 2015).

a. Cámara NRG. Public Lab desarrolló un proyecto DIY para la captura de imágenes multiespectrales a pequeña escala, basándose en el funcionamiento de los satélites de la NASA en el área de la fotografía infrarroja. Public Lab desarrolló una cámara digital modificada, capaz de capturar el espectro correspondiente a la banda del infrarrojo cercano (Near-Infrared o NIR por sus siglas en inglés), además del espectro correspondiente a las bandas del color rojo y verde. A este tipo de imágenes se les conoce

como NGR, siendo estas las siglas en inglés de los espectros que conforman a la imagen (Near-Infrared, Green y Red) (PublicLab, 2015).

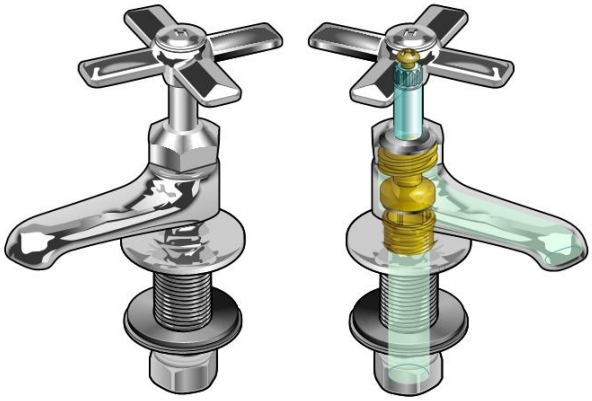
Para el desarrollo de esta cámara NRG fue necesario primero remover el filtro NIR de una cámara digital común, para luego colocar un nuevo filtro conocido como filtro NRG. El filtro NRG filtra el espectro correspondiente a la banda del color azul, dejando pasar en su lugar el espectro de la banda NIR. Al contar con la imagen digital en memoria, la respuesta espectral de la banda NIR se puede obtener accediendo al canal correspondiente al color azul. La respuesta espectral correspondiente al color verde y rojo no se ve afectada por el uso del filtro y continúan utilizando los mismos canales de la imagen digital en memoria (PublicLab, 2015).

Q. TRATAMIENTO DIGITAL DE IMÁGENES

1. Imagen digital. Una imagen digital es una colección ordenada de valores que corresponde a la representación bidimensional de una imagen en forma discreta. Una imagen digital es una señal en dos dimensiones cuyo dominio y rango son cantidades discretas y finitas. Las imágenes digitales se obtienen a través de dispositivos de conversión analógico-digital (Analog-Digital Converter o ADC por sus siglas en inglés) las cuales son almacenadas posteriormente en memoria mediante unidades de información binaria (Binary Units o bits por sus siglas en inglés). Dependiendo de la técnica de codificación utilizada para realizar la conversión una imagen digital puede ser una imagen de mapa de bits o una imagen vectorial (ITE, 2012).

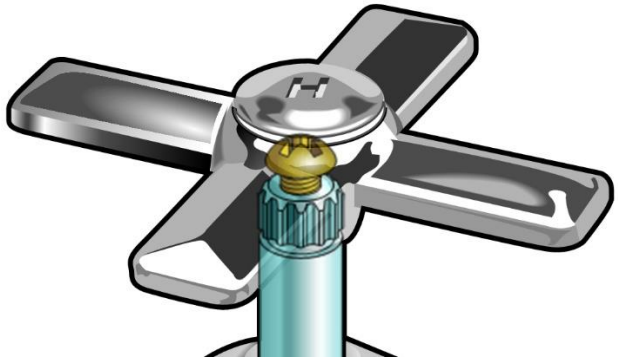
2. Imagen vectorial. Una imagen vectorial está compuesta por entidades geométricas simples e independientes, como lo pueden ser segmentos, arcos, polígonos, círculos, etc. Cada una de estas entidades está definida matemáticamente por un conjunto de parámetros, como lo son el color de contorno, coordenada inicial y final, color de relleno, etc. Este conjunto de parámetros se almacena utilizando vectores, razón por la cual se conocen como imágenes vectoriales (UNSL, 2012).

Figura 30. Ejemplo de una imagen vectorial formada por distintos objetos geométricos



Puesto que una imagen vectorial está compuesta por entidades geométricas simples, las imágenes vectoriales pueden cambiar de escala sin que la imagen se distorsione ni pierda calidad. Al ampliar la imagen vectorial en la Figura 14 se obtiene:

Figura 31. Ampliación de la imagen vectorial mostrada en la Figura 14



Una de las principales ventajas del uso de imágenes vectoriales es su capacidad de almacenamiento. Una imagen vectorial puede ser reducida y compactada para ocupar un mínimo espacio en memoria, ya que sólo se requiere la información necesaria para generar cada uno de los vectores que describen a las entidades geométricas que forman a la imagen. Cuando una imagen vectorial es ampliada o reducida el software mediante el cual se está visualizando la imagen recalcula automáticamente los parámetros de las entidades geométricos para no perder la calidad de la imagen. El principal inconveniente en una imagen vectorial es su falta de realismo fotográfico. Además, para poder visualizar una imagen vectorial en una pantalla o en la mayoría de sistemas de impresión ésta debe ser convertida a una imagen de mapa de bits (ITE, 2012).

3. Imagen de mapa de bits. Una imagen de mapa de bits (también conocidas como bitmap) se crea formando un arreglo rectangular formado por celdas. Cada una de las celdas recibe el nombre de píxel (proveniente del término picture element, elemento de imagen). Los pixeles son las unidades de color más pequeñas que componen a una imagen. Un píxel no tiene una medida concreta, en cambio representa la unidad mínima en que se divide la retícula de una imagen. Específicamente, un píxel almacena la intensidad de reflexión, *i.e.* la cantidad de luz presente, en una imagen para un tono en específico (ITE, 2012).

Una imagen bitmap está formada por una matriz de píxeles con dimensiones (a x b x c), donde a y b representan el alto y ancho de la imagen y c define la cantidad de canales que posee una imagen de mapa de bits, *i.e.* el número de tonos que forman a una imagen. Si la imagen posee un único canal se conoce como imagen monocromática. Este tipo de imagen se dice que es ciega al color, pues no permite diferenciar los colores que componen a la escena, sino únicamente la intensidad o brillo de la escena (ITE, 2012).

La profundidad de color es el número de bits utilizados para describir el color de cada píxel en la imagen. Sea n la profundidad de color, se dice que una imagen tiene 2^n colores o niveles de representación. Cada píxel puede asumir un único nivel de representación o color. Si un píxel tiene una profundidad de 8 bits, dicho píxel puede asumir uno de 256 valores posibles. La profundidad de color es una unidad de medida binaria puesto que cada píxel está formado por bits. A medida que la profundidad de color se hace más grande se tiene una mayor cantidad de niveles de representación, obteniéndose una imagen con mayor cantidad de detalles y de mejor calidad, *i.e.* se obtiene una representación más exacta de la imagen (ITE, 2012).

Figura 32. Imagen con profundidad de color igual a 1 bit, *i.e.* 2 niveles de representación. A la izquierda se encuentran los distintos niveles de representación y a la derecha se tiene la imagen resultante

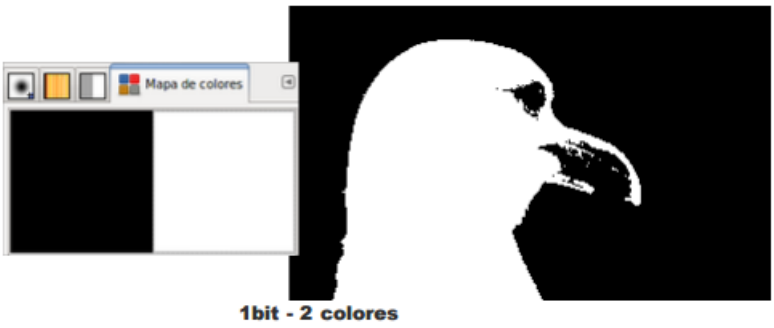


Figura 33. Imagen con profundidad de color igual a 2 bits, *i.e.* 4 niveles de representación. A la izquierda se encuentran los distintos niveles de representación y a la derecha se tiene la imagen resultante

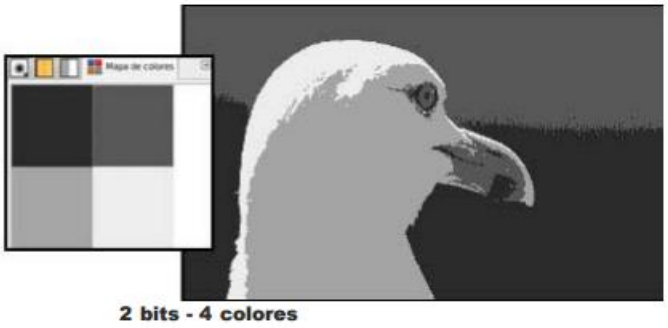


Figura 34. Imagen con profundidad de color igual a 4 bits, *i.e.* 16 niveles de representación. A la izquierda se encuentran los distintos niveles de representación y a la derecha se tiene la imagen resultante

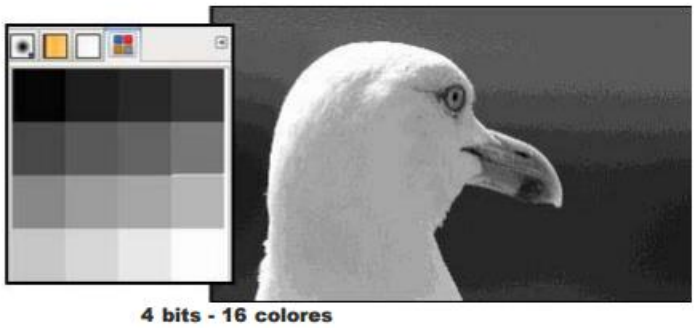
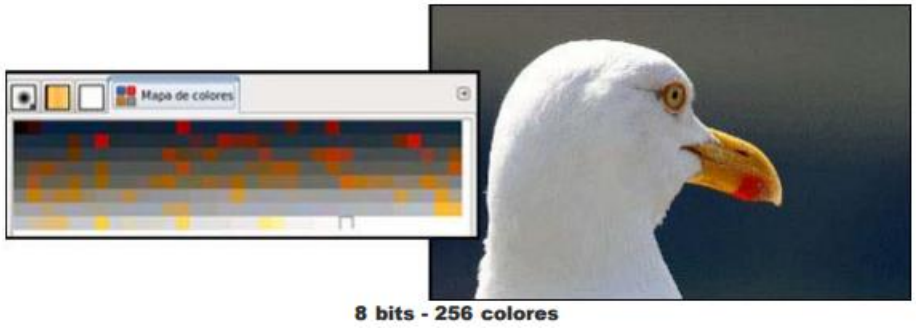


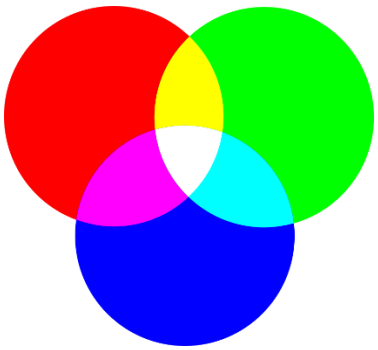
Figura 35. Imagen con profundidad de color igual a 8 bits, *i.e.* 256 niveles de representación. A la izquierda se encuentran los distintos niveles de representación y a la derecha se tiene la imagen resultante



a. Modos de color. Se conoce como modos de color al sistema de coordenadas que sirve para describir los colores de forma numérica. Los principales espacios de color son el RGB (rojo, verde y azul), HLS (tono, luminosidad, saturación) y el CMYK (cian, magenta, amarillo y negro) (UNSL, 2012).

El modo RGB es el que se utiliza en una imagen en la que el color se obtiene por mezcla aditiva de colores. En imágenes utilizando este espacio la gama completa de colores se obtiene a partir de la mezcla de los tres colores primarios, *i.e.* rojo, verde y azul. Un color en el modo RGB se describe mediante la tupla de tres valores numéricos correspondientes a un tercio de la profundidad de color, *i.e.* si se tiene una profundidad de color de 24 bits, cada canal se describe con un número de 8 bits (UNSL, 2012).

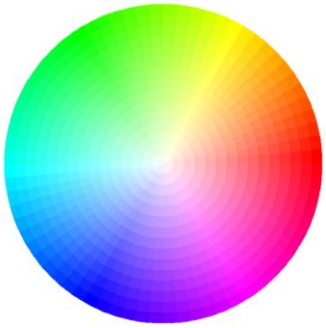
Figura 36. Representación gráfica del modo RGB



El modo HLS clasifica los colores de acuerdo a tres características principales del color, siendo estos el tono, saturación y luminosidad. El tono hace referencia a la longitud de onda dominante en la luz emitida o reflejada por un objeto. Para asignar un valor al tono se utiliza una rueda de color normalizada formada por los tres colores primarios, *i.e.* rojo, verde y azul, y lo tres colores secundarios, *i.e.* cian, magenta y amarillo, de forma alternada. De esta forma cada color está ubicado al lado opuesto de su complementario. El tono se especifica en grado, según su posición en la rueda de color (UNSL, 2012).

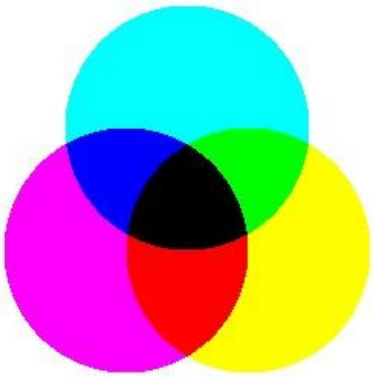
La saturación es la propiedad que describe la viveza del color. Se dice que un color muy saturado es aquel que posee una tonalidad intensa y pura, mientras que un color poco saturado es aquel que posee una tonalidad apagada. La saturación se expresa en porcentaje siendo 100% un color puro y 0% un color apagado cuya tonalidad apenas se distingue. En la rueda de color HLS la saturación se representa a lo largo del radio de la circunferencia. Un color saturado se encuentra cerca del borde mientras que los cercanos al centro son poco saturados. La luminosidad describe la intensidad de luz reflejada. La luminosidad se expresa en porcentaje siento 100% luminosidad total y 0% oscuridad total (UNSL, 2012).

Figura 37. Rueda de color HLS



El modo de color CMYK es aquel utilizado para describir el color que se obtendría si se tiñera un papel con tintas de colores. La razón de éste modo es que la tinta absorbe una parte de las longitudes de onda de la luz incidente, de modo que dicha parte del espectro de la luz blanca no absorbida se refleja. En otras palabras, la tinta sustrae al espectro de la luz blanca una parte de la radiación. Por esta razón se conoce a este modo de obtener colores un método sustractivo. Este es el modo de color preferido en aplicaciones de impresión de una imagen (UNSL, 2012).

Figura 38. Representación gráfica del modo CMYK



b. Resolución de una imagen. La resolución es la densidad de píxeles que tiene una imagen digital, *i.e.* cantidad de puntos en una imagen digital. En otras palabras, la resolución indica la cantidad de píxeles que hay por unidad de superficie. Se suele utilizar como unidad la pulgada cuadrada, *i.e.* 2.54 cm. La resolución de una imagen digital se suele expresar en píxeles por pulgada (pixels per inch o ppi por sus siglas en inglés), o puntos por pulgada (dot per inch o dpi por sus siglas en inglés). A medida que la resolución de una imagen digital sea más alta, más píxeles hay en una imagen y más grande será su mapa de bits. Una alta resolución permite representar un mayor detalle y transiciones de color más suaves en la imagen digital. Cuanto mayor es la resolución de una imagen más calidad tendrá su presentación, sin embargo más espacio ocupará el archivo en memoria (UNSL, 2012).

c. Tamaño y compresión de archivos. El tamaño del archivo es un valor numérico expresado en bits o bytes que describe la cantidad de memoria necesaria para almacenar la información de la imagen digital. El tamaño de un archivo se calcula mediante la siguiente fórmula (UNSL, 2012):

$$Tamaño = R^2 LAP$$

Donde R es la resolución de la imagen, L el largo de la imagen, A el ancho de la imagen y P la profundidad de color. El archivo, con nombre y extensión, no sólo contiene la información de cada píxel sino además posee una cabecera en la que se guarda información destinada al software encargado de abrir la imagen. Todos los archivos gráficos suelen tener tamaños muy grandes, lo cual representa un gran consumo de espacio en memoria. Para solucionar este problema se han desarrollado sistemas capaces de comprimir archivos gráficos. Cada sistema de compresión utiliza un algoritmo propio para reducir la cantidad de bits necesarios para presentar la imagen y marca el archivo resultante con la extensión que caracteriza al algoritmo. Los principales formatos de mapas de bits son: BMP, TIFF, PICT, GIF, PNG, PSD (UNSL, 2012).

1) Formato JPG. Es un formato de compresión con pérdidas que desecha en primer lugar la información no visible, por lo cual las pérdidas son no significativas. El algoritmo jpg está basado en el hecho que el ojo humano percibe peor los cambios de color que las variaciones de luminosidad. De esta forma el algoritmo divide la información de la imagen en color y luminosidad y las comprime por separado. El algoritmo permite modos de escala de grises con una profundidad de 8 bits y en color de hasta 24 bits (UNSL, 2012).

2) Formato GIF. Es un formato que produce imágenes resultantes de tamaño muy reducido. Esta reducción se logra indexando los colores, *i.e.* asocia cada color en la imagen a uno de los 256 colores en su tabla. Por tanto, su profundidad de color máxima es de 8 bits. El algoritmo gif es capaz de hacer transparente uno de los colores indexados en la tabla, permitiendo suprimir fondos. Además permite enlazar varias imágenes almacenadas con el algoritmo gif en una sola secuencia, produciendo lo que se conoce como animaciones gif (UNSL, 2012).

3) Formato PNG. Es un formato que reúne las ventajas de los algoritmos jpg y gif. Es un formato de compresión sin pérdidas con una profundidad de color de 24 bits. El algoritmo png soporta 256 niveles de transparencia, lo que permite unir la imagen perfectamente con el fondo. Sin embargo, el algoritmo no soporta animaciones y, debido a su capa de transferencia, su tamaño siempre es mayor al de un archivo jpg (UNSL, 2012).

4. Procesamiento de imágenes. El procesamiento de imágenes es el conjunto de transformaciones al espacio donde se encuentran originalmente los datos de una imagen, con el objetivo de poner en evidencia o realzar un cierto conjunto de patrones espaciales o espectrales (Lira, 2010). El procesamiento digital de una imagen consta de una serie de etapas o pasos según sea el entorno en donde se realiza la aplicación. En el área de investigación científica el procesamiento de imágenes consta de cinco etapas: captura, pre-procesamiento, segmentación, extracción de características e identificación de objetos y análisis (UNICEN, 2011).

a. Captura. La adquisición de una imagen está a cargo de algún transductor o conjunto de transductores que mediante la manipulación de la luz o de alguna otra forma de radiación reflejada por los cuerpos, se logra formar una representación de un objeto o escena dando lugar a la imagen. Durante esta etapa de adquisición los transductores agregan ruido a la imagen. Además, dependiendo del tipo de transductor utilizado se tendrá una resolución limitada, lo cual tiene implicaciones en la representación de la imagen (Escalante, 2006).

b. Pre-procesamiento. El pre-procesamiento digital de la imagen consiste en la eliminación de la mayor cantidad de ruido que pudo ser agregado durante la captura de la imagen. Además, tiene el objetivo de mejorar las características de dicha imagen, como lo es la definición de contorno, color, brillo, histograma, etc. Para ello esta etapa se vale de aplicación de herramientas y algoritmos matemáticos. En esta etapa también se incluyen las técnicas de codificación para el almacenamiento de la imagen, o bien, para su transmisión a través de un medio (Escalante, 2006).

Esta etapa también es necesaria para hacer correcciones del tipo geométricas. Estas tienen como objetivo corregir las distorsiones ocasionadas por variaciones en la geometría del sensor y de la Tierra, así como la asignación de coordenadas a cada uno de los píxeles. Una imagen corregida geométricamente debe representar correctamente la superficie de la Tierra. Los errores geométricos se deben a varios factores como el relieve del terreno, el movimiento en el sistema que porta el sensor, cambios en la altitud de la plataforma, curvatura de la Tierra, etc. (Fallas, 2004).

c. Segmentación. Una imagen digital contiene una variedad de patrones relacionados directamente con objetos de la escena y habrá otros patrones que no tengan significado alguno. Para analizar con mayor facilidad aquellos patrones de interés es necesario separarlos del resto de la imagen. Se define la segmentación como la partición de una imagen en regiones que pueden o no tener un significado relativo a la escena respectiva. Una imagen puede ser segmentada en función de su contenido de frecuencias espaciales (Lira, 2010). Para ello se suele emplear el uso de filtros espaciales. El objetivo de los filtros espaciales es resaltar o suprimir un patrón espacial particular en la imagen basada en su frecuencia espacial, *i.e.* la frecuencia de las variaciones en tonalidad en la imagen. Las áreas homogéneas tienen una baja

frecuencia espacial y por tanto los valores de sus píxeles son bastante similares. En cambio, zonas con cambios abruptos en tonalidad poseen una alta frecuencia (Fallas, 2004).

Filtrar una imagen de forma espacial consiste en realizar la convolución discreta en 2D de la imagen que se desea filtrar con una matriz cuadrada de orden n llamada comúnmente máscara o kernel. La máscara hace las veces de una ventana móvil cuyo centro recorre todos los píxeles que conforman a una imagen. El procedimiento de filtrado consiste en reemplazar el valor central de la matriz de filtrado por un valor derivado de los valores de los píxeles vecinos y de una operación matemática (moda, mediana, laplaciano, gaussiano, etc.). Cuando la ventana es desplazada el procedimiento de cálculo se repite. Los filtros pasa bajos tienden a homogenizar la apariencia de la imagen. En cambio, los filtros pasa altas tienden a realzar las diferencias entre los límites de diferentes zonas en la imagen. Se dice que los filtros pasa bajas suavizan la imagen, en cambio los filtros pasa altas realzan los bordes. Existen también filtros direccionales o de borde diseñados para resaltar elementos lineales tales como límites entre campos de cultivo o zonas de contacto en formaciones geológicas (Fallas, 2004).

d. Extracción de características. Como su nombre lo indica, esta etapa consiste en extraer la información de interés a partir de la imagen que previamente ha sido pre-procesada y segmentada. Dependiendo de la aplicación, esta etapa puede o no hacer uso de transformaciones. Las transformaciones son operaciones similares a las de realce de imagen, con la diferencia de que se suelen aplicar a un grupo de bandas espectrales y no a una banda individual. Las transformaciones más comunes son el crear nuevas imágenes basados en razones de bandas, *e.g.* en el cálculo de índices de vegetación, y el análisis de componentes principales, *i.e.* la combinación lineal de bandas. El objetivo de las transformaciones es crear nuevas bandas que muestren o resalten mejor los elementos presentes en la imagen (Fallas, 2004).

e. Identificación de objetos y análisis. Esta última etapa consiste en la clasificación de un conjunto de píxeles en una imagen. Por ejemplo algunos algoritmos de clasificación digital utilizan la reflectancia de cada píxel en diferentes longitudes de onda y un criterio estadístico para asignarlo a una clase espectral. Otras aplicaciones utilizan algún modelo matemático para decidir a qué categoría pertenece cada objeto en la imagen. La identificación de objetos es de gran utilidad en el campo de visión computacional (Fallas, 2004).

R. SISTEMAS DE COMUNICACIÓN

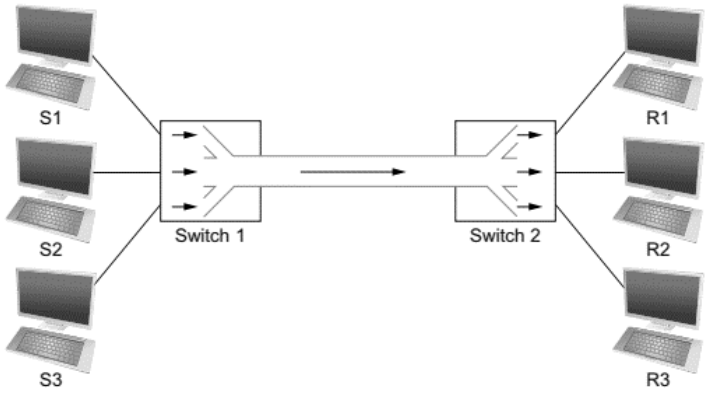
La comunicación es el proceso por el cual se transporta información de un punto a otro punto distante. Un sistema de comunicación electrónico es un sistema por el cual la información que se desea transportar es convertida en señales eléctricas y transmitidas al punto de destino donde se recupera el mensaje enviado. Un sistema de comunicación posee tres componentes básicos, un transmisor, un receptor y un canal o medio de comunicación. Si el medio de transmisión es un par de alambres, se le llama una línea de comunicación y si el medio es el espacio abierto y se utilizan ondas electromagnéticas, se le llama radio comunicación (Singh, 2008: xvii).

En las radio comunicaciones todas las señales son transmitidas en un medio común, el espacio abierto. Al tener una gran cantidad de señales en el espacio es necesario que el receptor pueda discernir cual mensaje es el que debe recibir. Esta interferencia puede ser resuelta utilizando el método de modulación. Este método consiste en traducir el mensaje a señales de radio con diferentes espectros (Singh, 2008: xviii).

Los sistemas de comunicación son limitados principalmente por el medio por el que es transmitida la información. Este medio presenta tres problemas fundamentales, el primero de ellos es la distorsión por atenuación de las señales transmitidas. La atenuación de una señal transmitida presenta un problema para el receptor ya que esta puede ser distorsionada o no llegar con suficiente energía para ser captada e interpretada correctamente. Si otros métodos no son empleados, la atenuación de la señal trasmitida limita al sistema a un alcance de operación. El segundo problema es la distorsión por retraso. La distorsión por retraso se produce cuando la velocidad y frecuencia de una señal varían resultando en una recepción errónea. Por último, el ruido es un elemento no controlable ni eliminable que degrada y distorsiona la señal, provocando que el sistema pueda recibir un mensaje erróneo (Prasad, 2004: 3).

Para realizar su objetivo, un sistema de comunicación debe cumplir con una variedad de funciones específicas lo cual conlleva a la implementación de varios mecanismos en los cuales resaltan la multiplexación, la conmutación de información y la detección de errores. La multiplexación es el mecanismo por el cual un sistema transmite señales de diferentes fuentes en un mismo canal. Al proceso de separar dichas señales en el receptor se le llama demultiplexación. La conmutación de información permite el transporte de la información de un punto a otro punto del sistema a través de otros puntos o nodos intermedios. La detección y corrección de errores es el mecanismo por el cual el sistema corrobora la integridad de la información y las acciones que debe tomar en el caso suceda un error. Este mecanismo es fundamental y es de gran importancia en medios con pérdida de información. (Prasad, 2004: 3).

Figura 39. Multiplexación y de-multiplexación de comunicación entre dos conmutadores de red.



Un sistema de comunicación puede ser clasificado por el tipo de comunicación que este provee. Una comunicación de punto a punto es aquella en la que un punto entabla una comunicación con otro punto. Una comunicación de punto a multipunto es aquella en donde el emisor envía información a múltiples receptores al mismo tiempo. A este proceso se le conoce como multidifusión. Una comunicación de difusión es aquella en donde el transmisor envía información a múltiples receptores de forma simultánea. Esta se diferencia de la multidifusión en que los receptores son pasivos y no hay un canal de comunicación en dirección contraria (Prasad, 2004: 4-5).

Existe también la clasificación con base en la dirección de los canales y la capacidad de estos de viajar en direcciones contrarias en forma simultánea. Una comunicación simplex es aquella en donde el emisor y el receptor no pueden intercambiar papeles. Una comunicación semidúplex es aquella en donde el transmisor y el receptor pueden intercambiar papeles pero solo pueden tomar un papel a la vez. Esto requiere que los puntos se turnen para ser transmisores o receptores. Por ultimo en una comunicación dúplex completa cada punto puede tomar el papel de transmisor y receptor simultáneamente. (Prasad, 2004: 6-7).

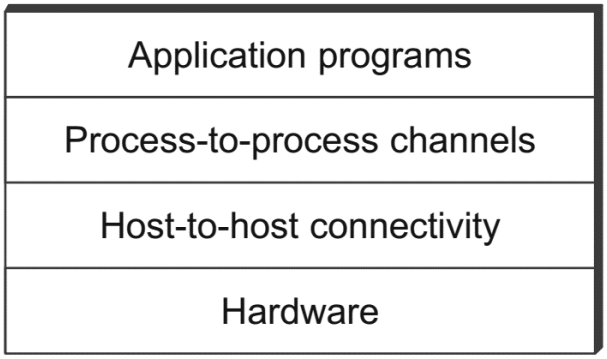
S. REDES DE COMPUTADORAS

A un sistema de comunicación que interconecta hardware programable de propósito general que no ha sido optimizado para una aplicación específica se le conoce como una red de computadoras. Estas comparten diferentes tipos de datos y tienen múltiples aplicaciones. La red de computadoras más grande que existe en la actualidad es el Internet (Peterson, 2012: 2). Las redes de computadoras están basadas en arquitecturas de red que permiten el diseño e implementación de grandes redes de forma eficaz y eficiente. El objetivo de una

red de computadoras es proveer una conectividad general, justa, robusta, barata y eficiente entre un gran número de computadoras (Peterson, 2012: 24).

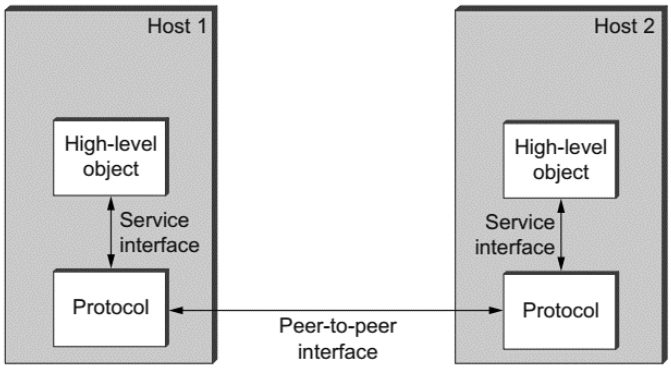
1. Arquitectura de red. Una arquitectura de red es una guía que permite el diseño y la implementación de la misma de manera sencilla. Para manejar la gran complejidad de una red se utiliza la abstracción. La abstracción es un método que oculta los detalles del funcionamiento completo de un sistema y define una interfaz sencilla de interactuar con el mismo. La abstracción normalmente está constituida por una secuencia de capas, cada una con mayor abstracción que la anterior. Esto permite la división del sistema en elementos manejables y la modularidad. De esta forma los problemas presentados en cada nivel de abstracción son resueltos de forma individual y modular requiriendo varias soluciones sencillas en vez de una solución compleja. En una arquitectura de red a cada elemento de la división de capas con diferentes abstracciones se le llaman protocolos. (Peterson, 2012: 24).

Figura 40. Modelo de cuatro capas.



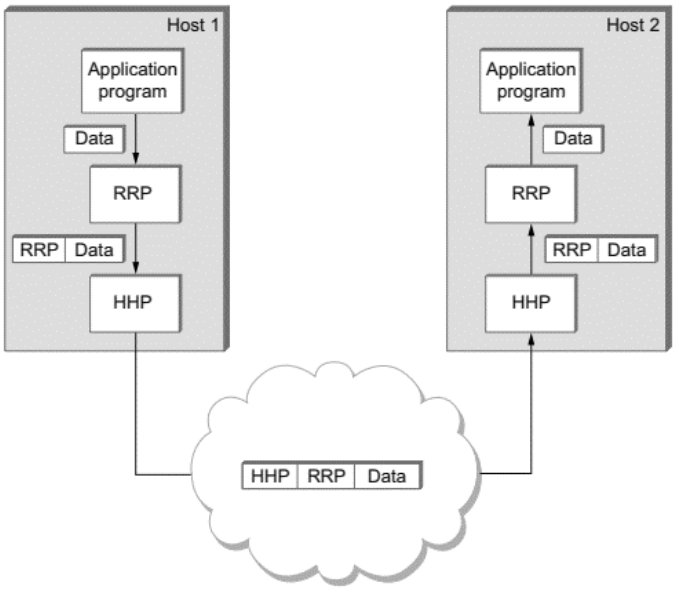
2. Protocolo de red. Un protocolo es un objeto que provee al sistema con un servicio. Un protocolo tiene dos características principales, una interfaz de servicio y una interfaz de par. La primera interface tiene como objetivo definir las funciones que pueden ser implementadas por otros objetos locales en el protocolo. La segunda define la forma y el significado que tiene el mensaje intercambiado con su contraparte en otro dispositivo de la red. Estos protocolos proveen a la red con los mecanismos necesarios para cumplir con los requisitos de un sistema de comunicación. Un protocolo utiliza encapsulación como forma de implementar estos mecanismos. (Peterson, 2012: 26).

Figura 41. Interfaces de un protocolo de comunicación de red.



3. Encapsulación. El método de encapsulación consiste en añadir información en la forma de un encabezado o una cola a un mensaje con objetivo de enviar información necesaria para implementar una función o mecanismo de control. El mensaje es enviado desde la aplicación y en cada una de las capas del modelo que se esté utilizando, un protocolo encapsulara el mensaje proveniente de la capa anterior. De esta manera cuando el receptor reciba el mensaje, el protocolo respectivo de cada capa des-encapsulará el mensaje e interpretara la información respectiva. En el receptor cada protocolo por el que el mensaje pase le quitara el encabezado y/o cola que le agrego el protocolo de su capa en el lado del transmisor pasara el mensaje a una capa superior. De esta forma la aplicación recibirá el mensaje sin encapsulamiento (Peterson, 2012: 29).

Figura 42. Encapsulación de paquetes.



4. Pila de protocolos de internet. Una pila de protocolos es la descripción del modelo de capas de un sistema. La arquitectura de Internet utiliza la pila de protocolos de internet basada en

el modelo OSI. Un modelo más apropiado sin embargo es el modelo TCP/IP. A continuación se describen cinco capas de abstracción.

a. Capa de aplicación. La capa de aplicación es donde las aplicaciones de red y los protocolos de capa de aplicación se encuentran. Por ser la abstracción más alta, esta es la capa que ofrece una interfaz al usuario de la red. Los protocolos de la capa de aplicación son distribuidos para que los sistemas en diferentes puntos de la red puedan comunicarse entre sí (Kurose, 2013: 51).

b. Capa de transporte. La capa de transporte es la encargada de transportar, como su nombre lo indica, un mensaje de un usuario a otro y viceversa. Existen dos protocolos principales para esta capa, el protocolo TCP y el protocolo UDP.

1) UDP. El protocolo de datagramas de usuario (UDP por sus siglas en inglés) es un protocolo sin conexión que permite la multiplexación y demultiplexación de segmentos además de la detección de errores en un segmento. Este no implementa una conexión ni una corrección de datos y por ende es un protocolo no confiable.

2) TCP. El protocolo de control de transmisión (TCP por sus siglas en inglés) es un protocolo orientado a conexión, dúplex completo, de punto a punto y confiable que implementa mecanismos de control de flujo, control de congestión y manejo de errores.

c. Capa de red. La capa de red es responsable de mover los paquetes (datagramas) de un host a otro del sistema a través de una dirección de protocolo IP. Además, existen en esta capa protocolos de enrutamiento sin embargo a esta capa se le conoce como capa IP debido a que es la protocolo IP el que uno a todos los hosts del internet.

d. Capa de enlace. Mientras que la capa IP encamina y mueve información de un host a otro de la red, la capa de enlace mueve información de nodo en nodo del sistema. Algunos protocolos de esta capa son confiables de un nodo a otro nodo del sistema. Esta capa está ligada fuertemente con la capa inferior y muchas veces son unidas por esta razón.

e. Capa física. Esta capa se encarga de mover cada bit de un nodo al otro nodo por el medio. Está siendo la capa inferior contiene la mayor cantidad de hardware.

T. PLATAFORMAS

1. Microcontrolador. Un microcontrolador es un computador contenido en un solo circuito integrado. Este es capaz de almacenar y correr un programa y contiene los elementos de un

computador como lo son una unidad central de procesamiento (CPU), memoria volátil (RAM), memoria no volátil (ROM), líneas de entrada y salida, puertos de comunicación y otros periféricos de utilidad. La diferencia principal es su aplicación, capacidad de procesamiento y costo (Heath, 2003: 11).

Los microcontroladores son fabricados con una amplia gama de aplicación y sus características pueden llegar a ser muy distintas. Existen microcontroladores con procesadores desde 8-bits hasta 32-bits dependiendo del fabricante. De igual manera varia la cantidad de memoria RAM y ROM. Para acomodarse a las necesidades de cada aplicación, los microcontroladores implementan distintos periféricos y funciones como puertos de comunicación, puertos de interrupciones al procesador, módulos analógicos y contadores entre otros (Heath, 2003: 10). Los microcontroladores pueden también contener periféricos estandarizados como puertos USB y conexión a internet.

a. Puerto serial. Un puerto serial es un método de comunicación entre dos dispositivos de un sistema embebido el cual hace uso eficiente de los pines de entrada y salida de un microcontrolador. Estos permiten la expansión de la funcionalidad de un microcontrolador al conectarse con uno o varios dispositivos de entrada y salida como un sensor electrónico o una pantalla LCD. Entre ellos puertos síncronos y asíncronos. Un puerto síncrono requiere una señal de reloj para sincronizar el receptor y el transmisor. Un puerto asíncrono por su parte carece de una señal de reloj y utiliza la misma señal de datos para sincronizarse (Heath, 2003: 140).

1) Bus I2C. El bus I2C (Inter-integrated Circuit) es un bus de comunicación síncrona que utiliza ambos, protocolos de hardware y software, para proveer una interface de comunicación entre varios dispositivos periféricos y puede manejar varios dispositivos maestros. El bus consiste de dos líneas, SDA y SCL, donde los datos se envían de forma serial a través de la línea SDA y la señal de sincronía se envía a través de la línea SCL. Debido a que se comparte la línea de transmisión de datos, esta comunicación es semidúplex (Heath, 2003: 143).

2) Comunicación serial UART. Se le llama UART (Universal Asyncronous Receiver/Transmitter) al hardware que traduce un dato paralelo a su forma serial. Muchos protocolos como el RS232 utilizan la tecnología UART como base para realizar transmisiones. La característica principal es que este no utiliza una línea de reloj. Esto le permite enviar información a grandes distancias e incluso enviar información de forma inalámbrica. Otra de las características principales es que carece de la estructura maestro-esclavo. La comunicación UART es una comunicación de punto a punto (Heath, 2003: 151).

2. Raspberry Pi. Una plataforma Raspberry Pi es un computador de bajo costo y de tamaño pequeño. Esta tiene la capacidad de realizar las funciones de un computador estándar de escritorio con una capacidad de procesamiento menor. Tiene la capacidad de interactuar con otros dispositivos y con

el usuario a través de los convencionales periféricos de un computador y una serie de puertos de entrada y salida para interconexión de circuitos. Esta plataforma utiliza principalmente sistemas operativos basados en núcleos de Linux, entre ellos el sistema operativo Raspbian. Existen diferentes versiones de la plataforma que varían en precio tamaño poder de procesamiento, potencia consumida y periféricos.

3. BeagleBone Black. La plataforma BeagleBone Black es una plataforma muy similar a la Raspberry Pi, con un tamaño pequeño y un bajo costo además de utilizar sistemas operativos basados en Linux. Esta plataforma sin embargo tiene capacidades muy diferentes a las de la Raspberry Pi. Con un procesador potente, dos microcontroladores de 32-bits y más de ochenta pines de entrada y salida le permite la interconexión con otras plataformas y otro hardware de forma sencilla.

U. TECNOLOGÍAS DE COMUNICACIÓN Y RED

1. Red celular y tecnologías GSM y GPRS. El sistema global de comunicaciones móviles GSM es el sucesor a las tecnologías inalámbricas analógicas de comunicación y es la tecnología inalámbrica más utilizada en el mundo. Fue diseñada a mediados de la década de los ochenta como un sistema basado en la conmutación de circuitos en donde se establece una conexión directa y exclusiva entre dos usuarios en cada interfaz en todos los nodos de red del sistema. En la actualidad, el sistema de conmutación de circuitos funciona en forma virtual y muchas veces y muchos de los nodos de red se conectan utilizando conexiones de banda ancha basadas en IP (Sauter, 2011: 1).

Una de las partes más importantes del sistema es la tarjeta SIM. Esta contiene todo la información e suscripción de un proveedor de servicios. Sus dos parámetros más importantes son su identidad internacional del suscriptor móvil (IMSI por sus siglas en inglés) y la clave de autenticación que son utilizados para conectarse al sistema (Sauter, 2011: 53).

Debido a su temprano comienzo, GSM es un sistema orientado a la transmisión de voz. Con la popularización del internet era necesario poder transmitir datos a dispositivos móviles de manera eficiente. El servicio general de paquetes vía radio (GPRS por sus siglas en inglés) fue implementado poder suplir está demando a través de la trasmisión de datos utilizando la conmutación de paquetes (Sauter, 2011: 64).

a. Adafruit FONA. Un módulo Adafruit FONA es un módulo de teléfono celular todo en uno. Este utiliza in modulo celular SIM800 para implementar una plataforma que le permite operar en una red 2G. Entre sus funciones se encuentran la transmisión y recepción de voz, mensajes SMS, datos a través de GPRS, receptor de radio FM y configuración por comandos AT entre otros.

b. Comandos AT. Los comandos AT es el standard de facto para el control de módems desde un computador u otra terminal inteligente. Estos tienen la función de configurar, probar y entablar conversación entre los dispositivos. En un modem GSM los comandos AT pueden preguntar el

estatus de un mensaje, interrogar al modem, controlar la red, recibir información de las configuraciones y variar los parámetros del sistema (Dunlop et al, 1999: 179).

2. Redes inalámbricas de área personal (WPAN) y Bluetooth. Una red inalámbrica de área personal o WPAN por sus siglas en ingles es una red que provee interconexión entre dispositivos personales en la cercanía del mismo. A diferencia de una red de área local (LAN) que provee servicio de datos entre todos los dispositivos conectados a la LAN, una red personal es de naturaleza íntima y personal en donde el usuario desea la conexión con otros dispositivos electrónicos móviles en su posesión o con otros dispositivos móviles de otros usuarios en particular. Las WPAN posea una gran movilidad y pueden conectarse a los dispositivos en su alcance a disposición los usuarios (IEEE, 2002: 19).

Un dispositivo portátil y un dispositivo móvil son muy similares, sin embargo para fines prácticos se ha defino un dispositivo móvil como un dispositivo que utiliza típicamente baterías como fuente potencia y entabla interconexiones fugaces con otros dispositivos. Por su parte un dispositivo portátil es movido con menos frecuencia, entabla interconexiones por periodos de tiempo mayores y usualmente está conectado a una línea de poder (IEEE, 2002: 20).

Las tecnologías de redes inalámbricas de área personal y de área local difieren en tres aspectos esenciales: su cobertura y potencia, el control del medio y el tiempo de vida de la red. Una WLAN esta optimizada para tener una gran cobertura, esto provoca un gran consumo de potencia y por ello tienden a estar conectados a una línea de poder. Una WPAN está orientada a comunicar múltiples dispositivos móviles personales en un área pequeña y por ende tiene típicamente un consumo energético mucho menor. En cuanto al tiempo de vida, una WLAN no deja de existir si todos los dispositivos conectados a esta dejan de existir debido a que la infraestructura ya es existente. Por el contrario en una WPAN si el dispositivo maestro deja de existir, también lo hace la red (IEEE, 2002: 20).

a. Bluetooth. La tecnología Bluetooth es una tecnología de comunicación de red inalámbrica de área personal o WPAN ubicua de bajo costo y baja potencia. El mecanismo por el cual dos dispositivos con Bluetooth habilitado se conectan entre ellos es llamado emparejado. El estándar internacional permite que casi todos los dispositivos con Bluetooth puedan emparejarse en una red ad hoc inalámbrica de corto alcance llamada piconet. Una piconet permite comunicarse simultáneamente con hasta siete dispositivos en la red. Cada dispositivo a su vez puede estar en varias piconets al mismo tiempo

La tecnología Bluetooth opera en la banda no comercial industrial, científica y médica (ISM por sus siglas en ingles) en un espectro ensanchado desde 2.4 1 2.484GHz. Este utiliza una comunicación full dúplex y una tecnología de salto de frecuencia adaptativo (AFH por sus siglas en ingles) para reducir interferencia provocada por tecnología inalámbrica que comparte el espectro. Este mecanismo detecta las frecuencias siendo utilizadas por otros dispositivos en el espectro y las evita para transmisión más eficiente.

Esta tecnología está divida en tres clases diferentes. La clase 1 usada principalmente en aplicaciones industriales tiene un alcance de hasta 100 metros. La clase 2 es la más popular en el mercado y es normalmente encontrada en dispositivos móviles y tiene un alcance de hasta 10 metros. Comúnmente está consume una potencia de 2.5 mW para mantener un bajo consumo. La clase tres tiene un alcance de 1 metro. Aunque existan clases dentro de la tecnología Bluetooth, estas no son mandatarias y existen variaciones de potencia y alcance entre cada dispositivo además de versiones que llegan a consumir una centésima de la potencia normal.

3. Módulos de radio frecuencia XBee. La empresa productora Digi define a un XBee como:

<<…soluciones integradas que brindan un medio inalámbrico para la interconexión y comunicación entre dispositivo. >>

Los XBee son módulos de radio frecuencia con una amplia gama de productos que permiten <<un alto tráfico de datos, una baja latencia y una sincronización de comunicación predecible>> según los presenta Digi. Sus productos varían en consumo energético, su capacidad de programación, alcance, ancho de banda, velocidad de transferencia de datos y protocolos de comunicación entre otros.

V. ELECCIÓN UAV:

A. DISEÑO EXPERIMENTAL:

Para la aplicación es fundamental contar con una estructura funcional de UAV. Para la realización de la estructura se consideraron dos opciones viables: la construcción desde cero del UAV, o implementar piezas y módulos prefabricados. Dada las restricciones de tiempo y considerando el reducido número de integrantes del grupo se optó por utilizar partes prefabricadas.

Entre las estructuras que se consideraron existían algunas que estaban listas para volar y cumplían con todos los requerimientos de hardware, así como otras que requerían de modificaciones para cumplir con los requisitos del megaproyecto. Luego se compararon las ventajas y desventajas de los distintos UAVs para finalmente escoger el adecuado.

B. RESULTADOS

Los modelos de UAV considerados se detallan a continuación:

AscTec Firefly (Ascending Technologies s.f.):

El AscTec Firefly fabricado por Ascending Technologies® es un cuadricóptero con propósitos de investigación. Utilizado en varias universidades alrededor del mundo, entre las que se encuentran MIT, Stanford, Virginia Tech, TUM, entre otros. Se caracteriza por su flexibilidad y rápido tiempo de respuesta.

Figura 43. AsTec Firefly

AscTec Hummingbird (Ascending Technologies s.f.):

El AscTec Hummingbird se caracteriza por su robustez y resistencia los choques, ideal para la investigación. Es el cuadricóptero utilizado en el Politécnico de Zürich (ETH) para sus investigaciones (Meier, Honegger y Pollefeys 2015) (Lupashin, y otros 2014). Posee gran cantidad de carga así como tiempo de vuelo aceptable.

Figura 44. AscTec Hummingbird

3DR Iris + (3D Robotics s.f.):

El Iris+ es un cuadricóptero de consumo masivo dedicado para los entusiastas de la fotografía. Un aspecto clave de este UAV es que el hecho de estar fabricado bajo una licencia OpenSource. Por lo que permite realizar modificaciones a la estructura y acceder a los parámetros de diseño fácilmente.

Figura 45. Iris+

Spreading Wings S900 (Dji s.f.):

El S900 es una estructura para UAV de alto desempeño. Fabricada con policarbonato resulta ligera y a la vez resistente. Dado que posee 6 rotores (Hexacóptero), es capaz de levantar cargas elevadas.

Figura 46. Spreding Wings S900

Luego de realizar la investigación de los UAVs se procedió a realizar una comparación entre los mismos:

Tabla II Comparación de los distintos vehículos aéreos

	Firefly	HummingBird	Iris+	S900
Computadora abordo	Intel® Core™ i7	Intel® Atom™Z530	No	No
Controlador	AscTec Autopilot®	AscTec Autopilot®	PX4 Autopilot	No
Capacidad de carga (g)	600	750	200	4000
Comunicación Wireless	2.4 GHz XBee link	2,4 GHz XBee link	3DR Radio 433MHz	No
Tiempo de Vuelo (min)	12-14	20	20	N/A
Precio (USD)	$ 25,300.00	$ 17,500.00	$ 750.00	$ 1,300.00
OpenSource	No	No	Si	No
Requiere de modificaciones	No	No	Si	Si

Tabla III Comparación por matriz de Pugh

	Importancia	FireFly	HummingBird	Iris+	S900
Precio	7	1	2	4	3
Carga Útil	3	2	2	0	4
Controlador	2	2	2	2	0
Procesador	1	2	2	0	0
OpenSource	1	0	0	3	0
Trabajo Adicional	2	3	3	1	0
Total		29	36	45	25

C. DISCUSIÓN

Finalmente se escogió el Iris+ tal como se muestra en la sección de resultados (Tablas II y III) resulto ser el más barato de todas las opciones lo cual es conveniente considerando lo limitado del presupuesto para el Megaproyecto. Otro factor a considerar era la carga útil, la cual era conveniente que fuera elevada, sin embargo, tuvo que ser sacrificada ante la diferencia de costos. Un aspecto relevante, en el cual el Iris+ superó a sus contrapartes fue el hecho de ser OpenSource pues facilita el acceso a código, librerías y soporte. En cuanto a la computadora de compañía fue necesario sacrificar los potentes procesadores de los UAVs de Ascending Technologies debido a que eran muy costosos para el presupuesto.

VI. INSTALACIÓN DEL *FIRMWARE:*

A. DISEÑO EXPERIMENTAL

Para instalar el *firmware* fue necesario instalar una serie de herramientas de software (*toolkit*). Se decidió utilizar una distribución de Linux (Ubuntu) pues es más flexible en cuanto a paquetes de software y tiene mejor soporte para desarrolladores. A lo que se agrega que la mayoría de paquetes de software están únicamente para Linux, tal es el caso de ROS. De lo contrario habría que crear una máquina virtual, lo cual consume muchos recursos y la velocidad de procesamiento no es comparable a una instalación nativa de Linux.

Se instalaron las siguientes dependencias:

- python: serial, argparse
- libncurses5-dev
- libc6:i386

Asimismo, se agregó al usuario al grupo *dialout* lo cual es extremadamente importante para poder abrir los puertos de comunicación. De lo contrario desplegará errores de comunicación.

Luego de tener las herramientas necesarias se prosiguió con el *build* de las liberarías (las cuales fueron conseguidas en el repositorio del PX4) utilizando el *builder* de C++ que viene instalado por defecto en Ubuntu. Ya con la versión compilada del programa (*firmware*) se instaló el software en la tarjeta controladora.

B. RESULTADOS

Para verificar la correcta instalación del *firmware* se procedió a establecer una comunicación serial entre el PX4 y la computadora. Para esto se usó la aplicación *screen* de Ubuntu. El resultado fue una pantalla en terminal con comunicación a la consola nsh del PX4:

Figura 47. Terminal nsh

```
nsh> ?
NSH command forms:
  [nice [-d <niceness>>>]] <cmd> [> <file>|>> <file>] [&]
OR
  if <cmd>
  then
    [sequence of <cmd>]
  else
    [sequence of <cmd>]
  fi
Where <cmd> is one of:
  [ <expression> ]
  ?
  cat <path> [<path> [<path> ...]]
  cd [<dir-path>|-|~|..]
  cp <source-path> <dest-path>
  dd if=<infile> of=<outfile> [bs=<sectsize>] [count=<sectors>] [skip=<sectors>]
  echo [<string|$name> [<string|$name>...]]
  exec <hex-address>
  exit
  free
  help
  kill -<signal> <pid>
  losetup [-d <dev-path>] | [[-o <offset>] [-r] <dev-path> <file-path>]
  ls [-lRs] <dir-path>
  mb <hex-address>[=<hex-value>][ <hex-byte-count>]
  mkfifo <path>
  mh <hex-address>[=<hex-value>][ <hex-byte-count>]
  mw <hex-address>[=<hex-value>][ <hex-byte-count>]
  ps
  pwd
  set <name> <value>
  sh <script-path>
  sleep <sec>
  test <expression>
  unset <name>
  usleep <usec>
  xd <hex-address> <byte-count>

Builtin Apps:
mavlink
```

C. DISCUSIÓN

A partir de los resultados se pudo garantizar la existencia de una comunicación bidireccional, pues se recibió el *echo* del PX4 y al momento de enviar un comando este respondió. Lo cual garantiza que se contaban con todos los permisos para accesar a los puertos, las direcciones eran las adecuadas y la tasa de transferencia era la adecuada.

VII. INTEGRACIÓN SENSORES, MOTORES Y CONTROLADOR:

A. DISEÑO EXPERIMENTAL

Luego de haber instalado el *firmware* en la tarjeta controladora se procedió a conectar todos elementos del UAV: IMU, barómetro, GPS, radios de telemetría, radio controlador, módulo de potencia y actuadores (motores). Fue necesario instalar los driver de cada sensor e interconectarlos con los niveles de voltaje adecuados. Para verificar el funcionamiento de los actuadores se activaron los rotores (sin hélices). Se enviaron señales del radio controlador al UAV para observar su respuesta a dichos comandos. Por otra parte se activaron las aplicaciones de seguridad, estimación, estabilización y control de postura.

Previo a las pruebas de vuelo iniciales fue necesario realizar una calibración de los sensores: la calibración se realizó de la siguiente manera:

- Giroscopio: rotación en el eje *roll*, *yaw* y *pitch.*
- Acelerómetro: se colocó el cuadricóptero en las posiciones indicadas en la Figura 48
- Magnetómetro: se colocó el UAV en posición horizontal sobre una superficie plana.
- Gps: No fue necesario

Figura 48. Calibración Acelerómetro

B. RESULTADOS

La estructura con los componentes quedo de la siguiente manera:

Figura 49. Distribución componentes UAV

C. DISCUSIÓN

Un aspecto clave al momento de colocar todos los elementos en la estructura del cuadricóptero, fue su posición respecto al centro de gravedad del UAV. Dado el modelo dinámico que se explicó en la teoría, es necesario que el centro de gravedad coincida con el centro geométrico de la estructura ya que fue uno de los supuestos para el desarrollo del modelo teórico, a partir del cual se diseñaron los controladores, por lo que un desbalance en la estructura resultaría en inestabilidad e incluso accidentes.

Como se logra apreciar en la Figura 49, se intentó distribuir de manera uniforme la carga. Dada la complicada geometría de la estructura, el análisis del centro de gravedad fue de tipo cualitativo; realizando las modificaciones pertinentes en base a su comportamiento en vuelo y durante los despegues.

Respecto al funcionamiento en vuelo, se puede apreciar el comportamiento del UAV en el video (Velasquez, Pruebas vuelo manual [Archivo de Video] 2015). Entre los aspectos más relevantes, cabe destacar que se logró que el UAV despegara, lo cual representa el correcto funcionamiento del *firmware* instalado.

El funcionamiento del módulo de estabilidad demostró ser aceptable, pues como se logró apreciar en la prueba de vuelo manual (Velasquez, Pruebas vuelo manual [Archivo de Video] 2015), el cuadricóptero permanece en vuelo y logra aceptar los comandos enviados por el operador.

Parte del comportamiento en vuelo del cuadricóptero radica en la experiencia y habilidad del operador. Dado que aún no hay control de velocidad y posición estos factores dependerán únicamente de los comandos enviados por el operador a través del Radio Control.

VIII. IMPLEMENTACIÓN DE LIBRERÍAS DE CONTROL EN TARJETA Y COMPUTADORA DE COMPAÑÍA:

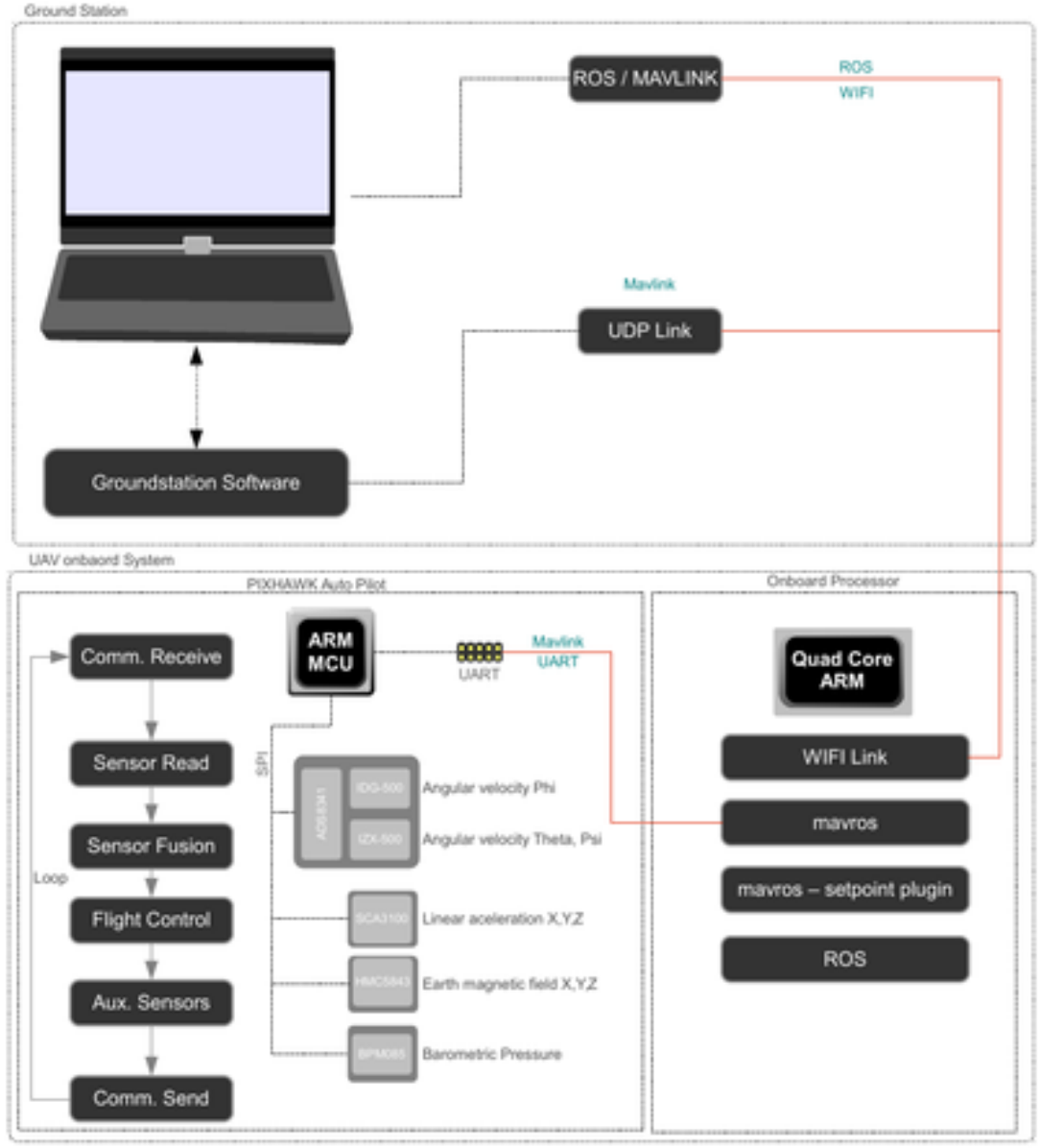
Existen diversas topologías para la conexión de la tarjeta controlador PX4. Dadas las características del megaproyecto se escogió utilizar una computadora de compañía que ejecute Linux de manera nativa, la cual a su vez podrá comunicarse de manera bidireccional con la tarjeta de control (PX4) y con una estación en tierra por medio de comunicación serial. Tanto la tarjeta de control como el PX4 estarán ejecutando ROS (Robotic Operating System) en todo momento, pues será este el encargado de establecer la comunicación y crear los nodos para la comunicación y funcionamiento del UAV.

Se crearon diversos nodos ROS, en la tarjeta de control se ejecutan los nodos encargados de la estabilidad, posición, altura, orientación y actuadores. En la computadora de compañía se ejecutan los nodos de planeación de trayectorias, puntos-objetivo y fotografía. La comunicación se realiza empleando un protocolo *mavlink* nativo de ROS, dicho protocolo es *Mavros.*

En otras palabras, la tarjeta PX4 empleará las librerías que incluye el *firmware* que se instaló en los pasos anteriores para controlar la estabilidad, velocidad, sensores, postura y posición. La computadora de compañía por su parte calculará las trayectorias, tomará las fotografías y realizará el procesamiento de las imágenes. El proceso es completamente transparente para ambas partes, lo que permite que cualquiera de las computadoras sea reemplazada por otra sin realizar ningún cambio; toda vez se estén ejecutando los nodos ROS y sean estos quienes publiquen a los temas respectivos.

Para que la tarjeta PX4 inicializara las aplicaciones necesarias para controlar el UAV en la topología seleccionada se modificó el archivo *boot.txt* del PX4, iniciando las aplicaciones de seguridad, drivers de sensores y actuadores, control de estabilidad, posición, postura y velocidad.

Figura 50. Topología implementada



(PX4 Project s.f.)

IX. ALGORITMOS DE CONTROL EN TARJETA PX4:

A. DISEÑO EXPERIMENTAL

Para el control de la estabilidad, posición y postura se partió de las librerías OpenSource (PX4 Project s.f.) desarrolladas por el Instituto Politécnico de Zurich (ETHZ). Las cuales implementan los sistemas de control descritos en la sección de teoría. Asimismo, se encargan de la integración de la información obtenida de los sensores. Posteriormente se realizaron las modificaciones en los algoritmos de control implementados. Los cuales consistieron esencialmente en los parámetros PID de los controladores de postura y posición así como las ganancias de los observadores de postura y posición.

B. RESULTADOS

La interacción de los programas que intervienen en el control de vuelo (posición, estabilidad y postura) se muestra en el siguiente diagrama: (Pixhawk s.f.)

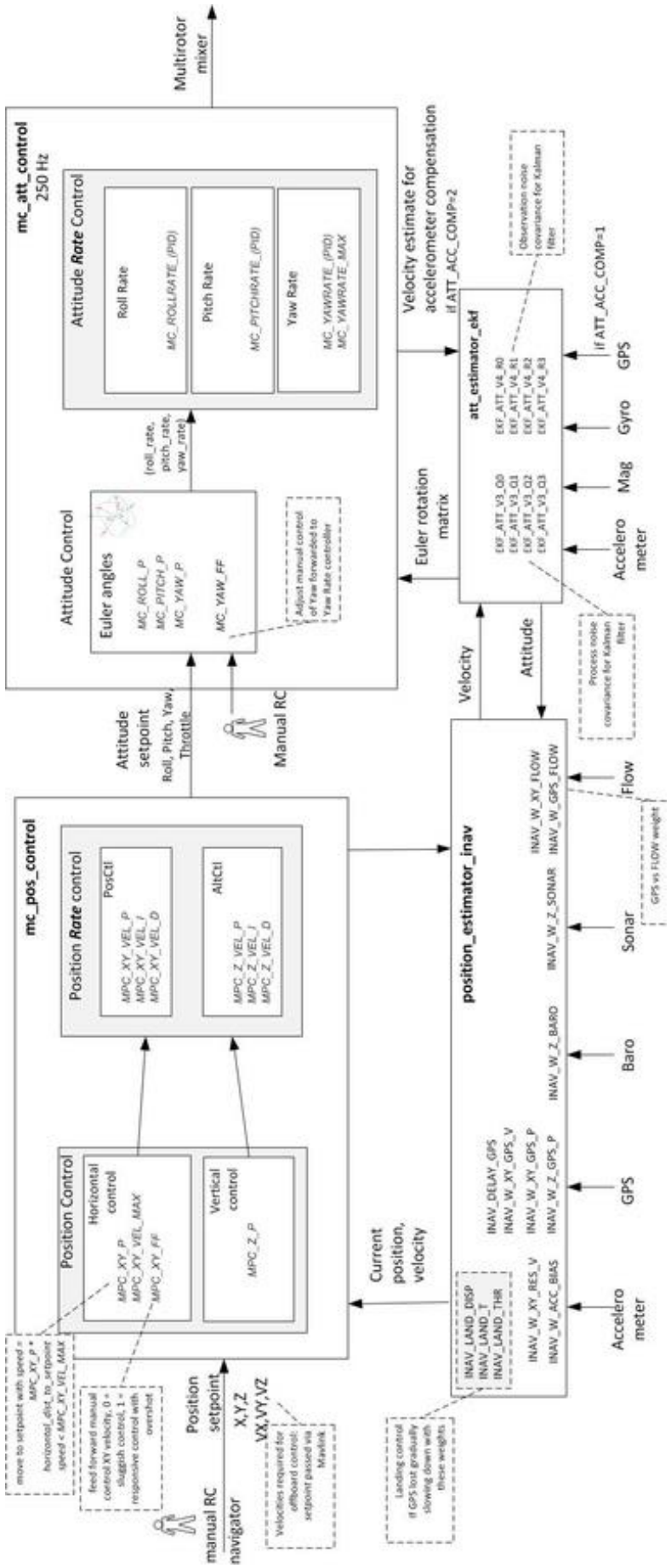


Figura 51. Interacción aplicaciones de control (Pixhawk s.f.)

C. DISCUSIÓN

Como se puede observar en la Figura 51. position_estimator_inav se encarga de obtener la información de los sensores y enviar sus resultados al controlador de posición (mc_pos_control) el cual controla las velocidades en los *ejes-x-y-z* para alcanzar las posiciones deseadas. Este a su vez envía los ángulos necesarios para alcanzar dichas velocidades (tal como se describió en la sección de teoría) al controlador de postura: mc_att_control el cual actúa directamente sobre los cuatro rotores del cuadricóptero. A partir de la postura actual se realiza un estimado de la velocidad para retroalimentar al estimador de postura att_estimator_ekf el cual retroalimenta al controlador de postura y al estimador de posición position_estimator_inav.

Para controlar el cuadricóptero desde la computadora de compañía se interceptarán los datos enviados por el estimador de posición. Con base en la información de posición se realizará el trazado de trayectorias y se enviaran los parámetros ya sea de velocidad y/o posición al mc_pos_control o directamente los ángulos de postura al mc_att_control a través del protocolo Mavlink implementado en la computadora de compañía.

X. COMPUTADORA DE COMPAÑÍA:

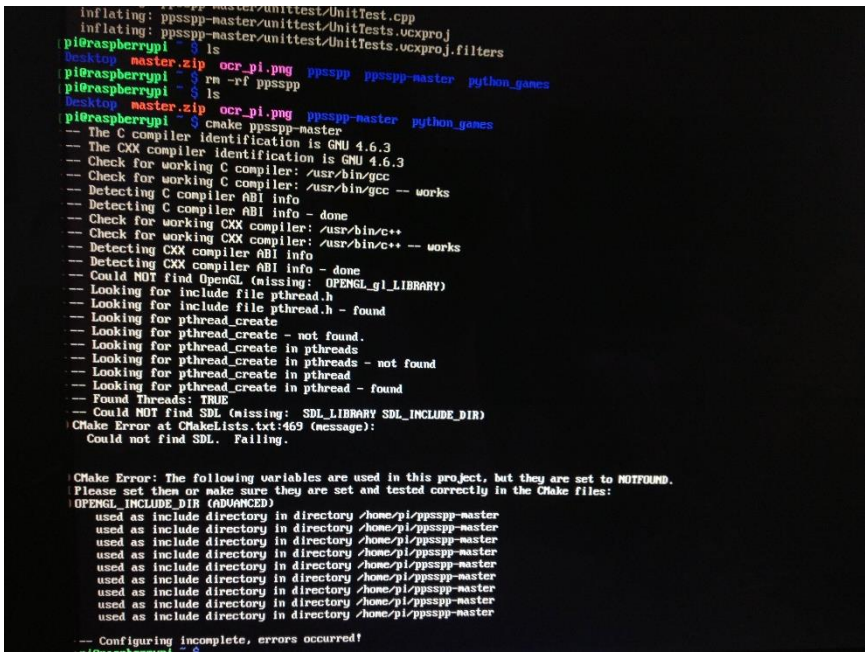
A. ROS EN RASPBERRY PI

1. Diseño experimental. Cómo siguiente paso lógico, se procedió a instalar ROS en una Raspberry Pi (RPi). Dado que no existía una distribución nativa para Rpi, se decidió realizar el build de ROS desde el código fuente. Primero se configuraron las claves del repositorio de ROS, se instalaron las dependencias para la realización del build y se creo el espacio de trabajo (catkin workspace).

Se instaló una versión base de ROS, por ser más ligera y no tan intensiva en el uso de recursos. Se instalaron los paquetes de comunicaciones y ejecución, sin interfaces gráficas de usuario (GUI). Finalmente se procedió a realizar el *build.*

2. Resultados. La instalación de ROS en la Rpi y la construcción del espacio de trabajo tomó alrededor de 12hrs. La memoria se llenó antes de instalar todos los paquetes necesarios, por lo que fue necesario cambiar la memoria SD utilizada (inicialmente 2Gb), a 4Gb.

Figura 52. Instalación ROS en Raspberry Pi



3. Discusión: La instalación de ROS en la RPi tomó más tiempo del esperado, más de 12hrs. Lo cual resulta inquietante considerando el hecho de que esta será la encargada de control de vuelo del UAV, por lo que se esperaría una respuesta rápida y constante. Al establecer comunicación con la tarjeta de control, fue necesario modificar las direcciones de los puertos, pues los parámetros preestablecidos no coincidían con los puertos de la RPi.

Un aspecto crítico es la estabilidad de las comunicaciones. La RPi presentaba un gran tiempo de latencia en las comunicaciones, lo cual provocaba que la comunicación entre la RPi y PX4 se perdiera. Debido a estos problemas se decidió cambiar de computadora de compañía para evitar problemas más graves en vuelo.

B. INSTALACIÓN ROS EN COMPUTADORA PERSONAL:

1. Diseño experimental: Por los problemas que se presentaron con la RPi se decidió utilizar como computadora de compañía una computadora personal, ya que presenta mayor capacidad de procesamiento y permite la instalación de los recursos visuales que vienen incluidos en ROS.

La computadora utilizada fue una Dell XPS L501x con procesador i7 ejecutando Ubuntu 14.02 LTS de manera nativa. Ya que se contaba con más recursos, se decidió instalar la versión completa de escritorio de ROS, la cual cuenta con las herramientas visuales para *debugging* y análisis de comunicaciones. Para la instalación de ROS en la PC se utilizó el paquete *aptitude* que incluye Ubuntu en algunas de sus distribuciones. Aparte del paquete principal de ROS, se instalaron los paquetes *mavros* y *mavros_extras*.

Posteriormente se procedió a establecer comunicación entre la tarjeta de control PX4 y la PC por medio de un radio-modem usb. Para lo cual se configuraron los puertos de la computadora, se agregaron los permisos correspondientes y se modificaron las direcciones USB. Dicha comunicación permitió realizar el *debugging* durante las pruebas manuales y de vuelo autónomo.

Para inicializar los nodos ROS en la PC se creó un archivo *launch* el cual inicializó los nodos necesarios para comunicación con la tarjeta controladora y el control del UAV. Dicho archivo fue elaborado en C++, pues en este lenguaje existía mayor documentación por parte de los desarrolladores.

2. Resultados: Para verificar el funcionamiento de ROS y sus dependencias en la PC se inicializaron los nodos ROS empleando el archivo *launch* creado previamente. Para observar el funcionamiento se utilizó la herramienta de *rxgraph* que viene instalada por defecto en ROS. Se observaron los nodos existentes y los temas que estaban publicando en ese momento.

Figura 53. Inicialización del archivo launch

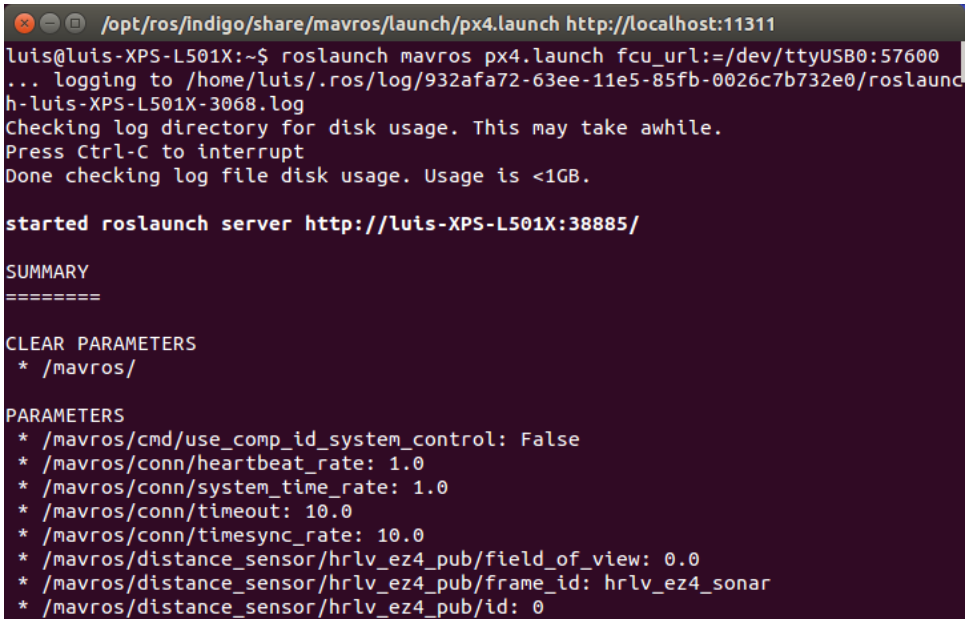


Figura 54. Heartbeat

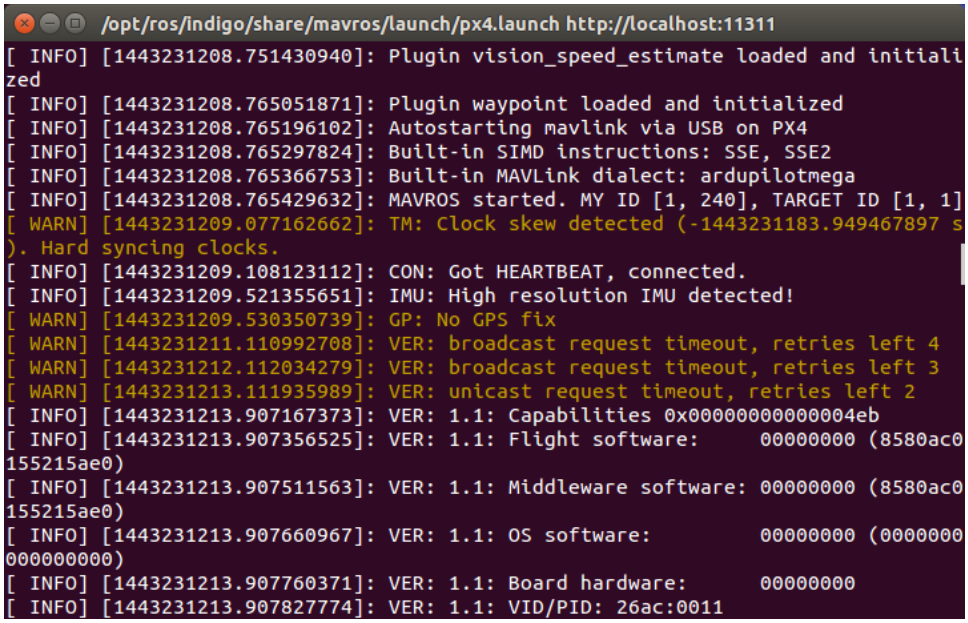


Figura 55. Datos enviados por PX4

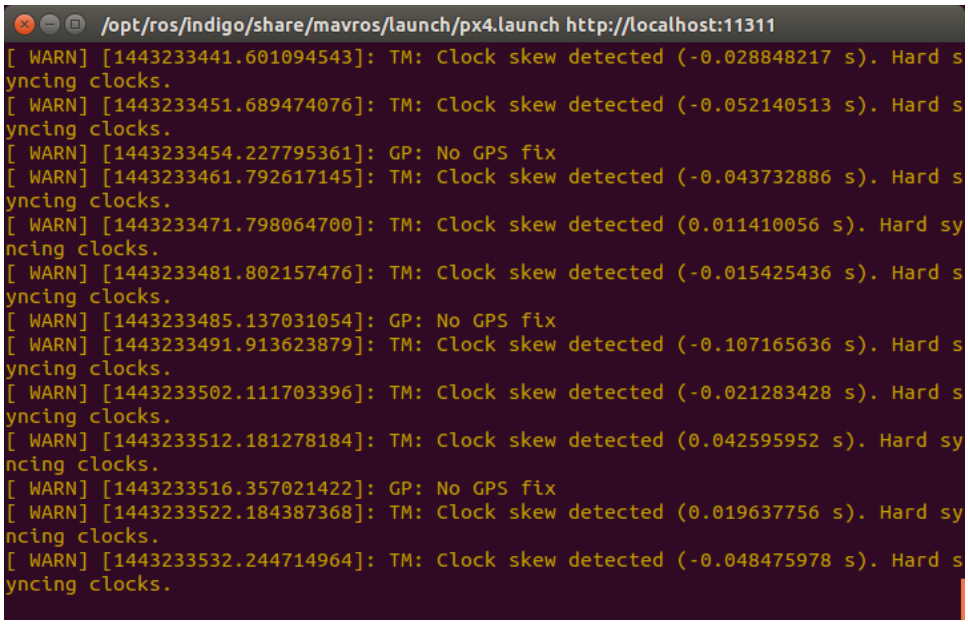


Figura 56. Nodos en ejecución

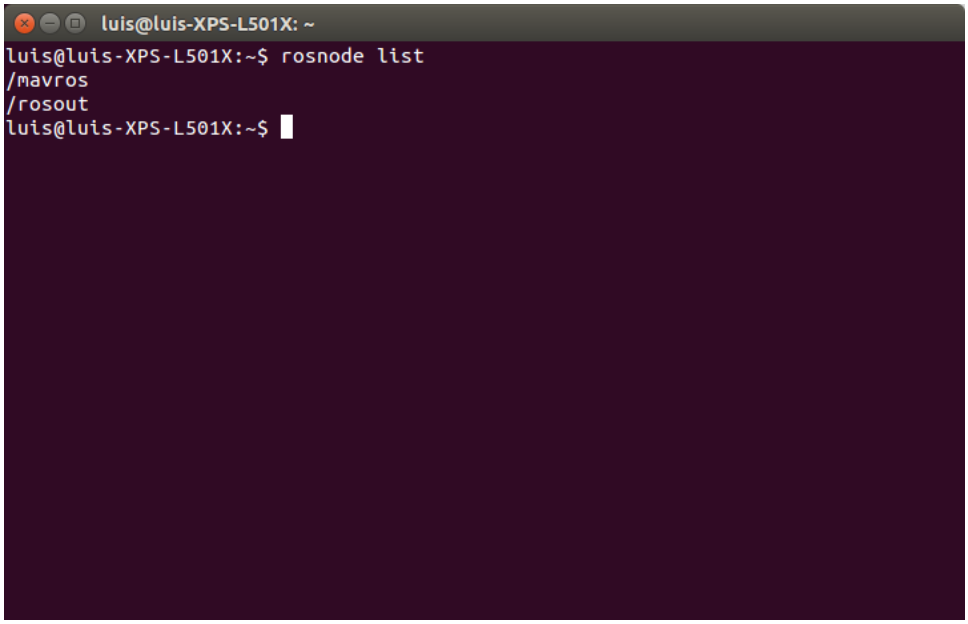


Figura 57. Vista gráfica nodos

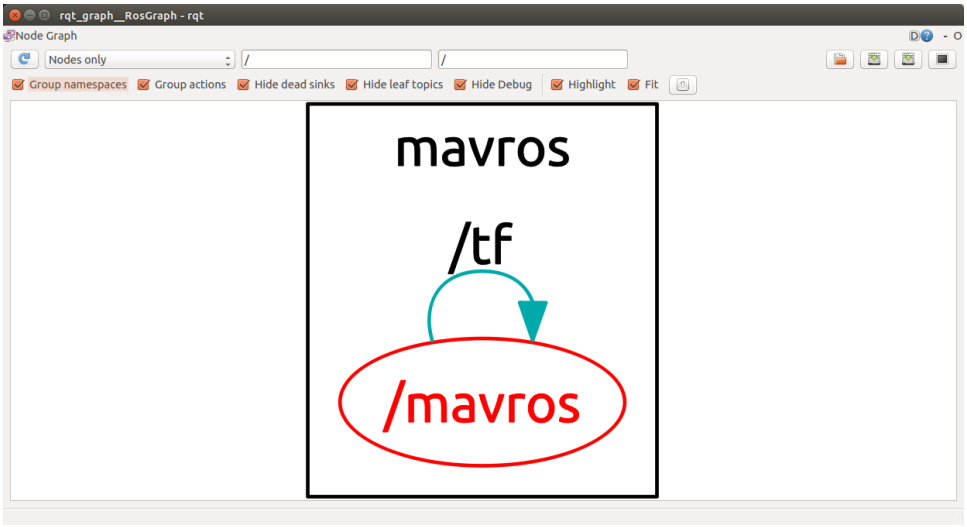


Figura 58. Servicios ROS activos

```
luis@luis-XPS-L501X: ~
luis@luis-XPS-L501X:~$ rosservice list
/mavros/cmd/arming
/mavros/cmd/command
/mavros/cmd/command_int
/mavros/cmd/guided_enable
/mavros/cmd/land
/mavros/cmd/set_home
/mavros/cmd/takeoff
/mavros/cmd/trigger_control
/mavros/ftp/checksum
/mavros/ftp/close
/mavros/ftp/list
/mavros/ftp/mkdir
/mavros/ftp/open
/mavros/ftp/read
/mavros/ftp/remove
/mavros/ftp/rename
/mavros/ftp/reset
/mavros/ftp/rmdir
/mavros/ftp/truncate
/mavros/ftp/write
/mavros/get_loggers
/mavros/mission/clear
/mavros/mission/goto
/mavros/mission/pull
/mavros/mission/push
/mavros/mission/set_current
/mavros/param/get
/mavros/param/pull
/mavros/param/push
/mavros/param/set
/mavros/set_logger_level
/mavros/set_mode
/mavros/set_stream_rate
/rosout/get_loggers
/rosout/set_logger_level
/rqt_gui_py_node_3758/get_loggers
/rqt_gui_py_node_3758/set_logger_level
luis@luis-XPS-L501X:~$
```

3. Discusión: Se logró establecer una comunicación exitosa entre la PC y el PX4. Como se puede observar en la Figura 50 el archivo *launch* inició los nodos ROS necesarios, así como los servicios que serán utilizados en partes posteriores. Si se observa la Figura 54, se logra apreciar el *HEARTBEAT* enviado por el cuadricóptero. Este *heartbeat* verifica que la parte receptora (PX4) esta energizada y recibiendo los datos enviados por la PC. Al observar la Figura 55, se puede ver la retroalimentación enviada por el PX4 a la PC. Dado que las pruebas se realizaron en el laboratorio (el cual se encontraba en un entorno cerrado), el sensor GPS no recibía señal por parte de los satélites. Es por ello que el PX4 envía una advertencia a la computadora indicando que no hay posicionamiento por GPS.

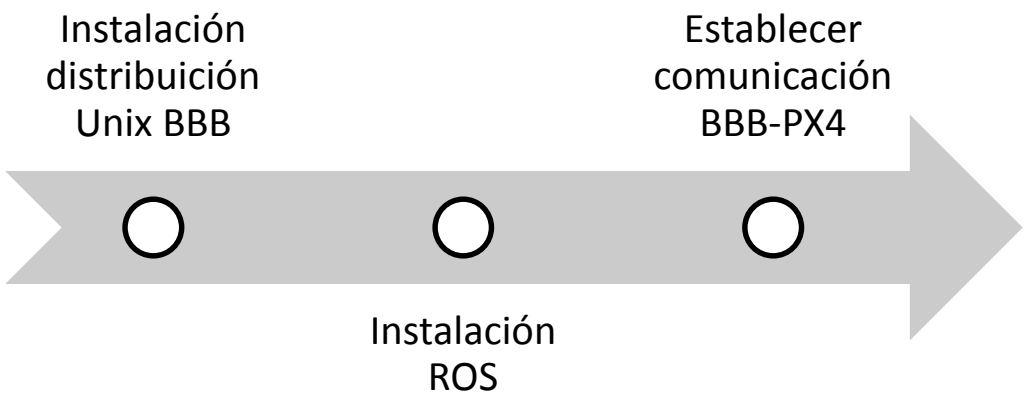
El archivo *launch* inicializó los nodos y los servicios que serán necesarios para enviar información al PX4. Como se logra observar en la Figura 56 y Figura 57 únicamente hay un nodo en ejecución (*mavros*). *Mavros* es el encargado de establecer y administrar las comunicaciones entre ambas computadoras. Por lo cual es necesario que este nodo este en ejecución en todo momento.

Al observar la Figura 58, se identifican los servicios que fueron inicializados por el archivo *launch.* A diferencia de los nodos, hay más servicios en ejecución. Tal como se explicó en la parte teórica, estos servicios son utilizados para realizar requerimientos a otros nodos. En este caso los requerimientos serán realizados por la PC a los nodos inicializados en la tarjeta controladora (PX4).

C. ROS EN BEAGLEBONE BLACK (BBB):

1. Diseño experimental: El proceso para la instalación de ROS en el BBB se puede describir de la siguiente manera.

Figura 59. Diseño experimental instalación ROS en BBB



2. Resultados

Figura 60. Problemas con la Red

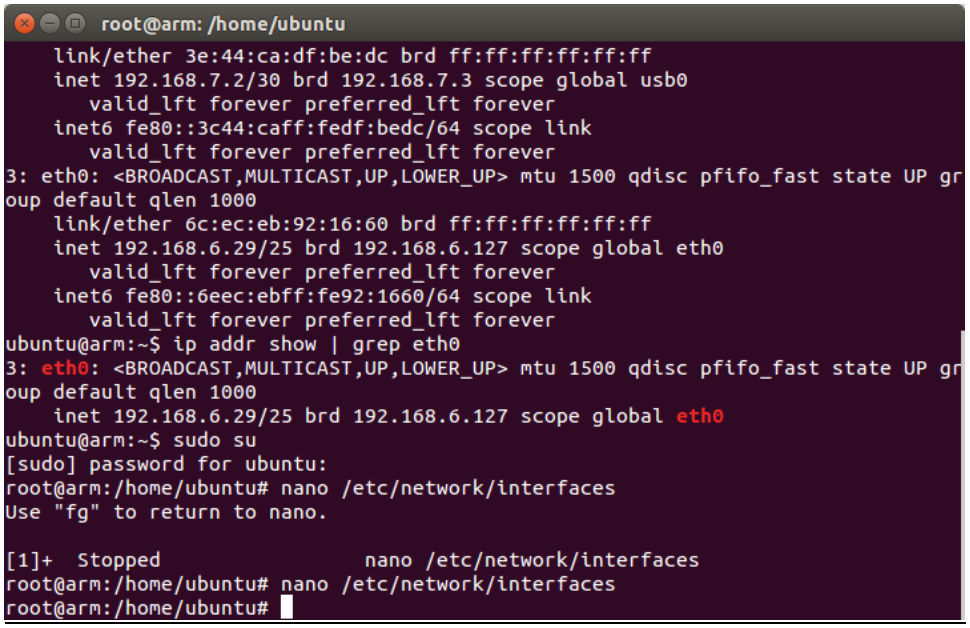


Figura 61. Previo a la extensión de memoria

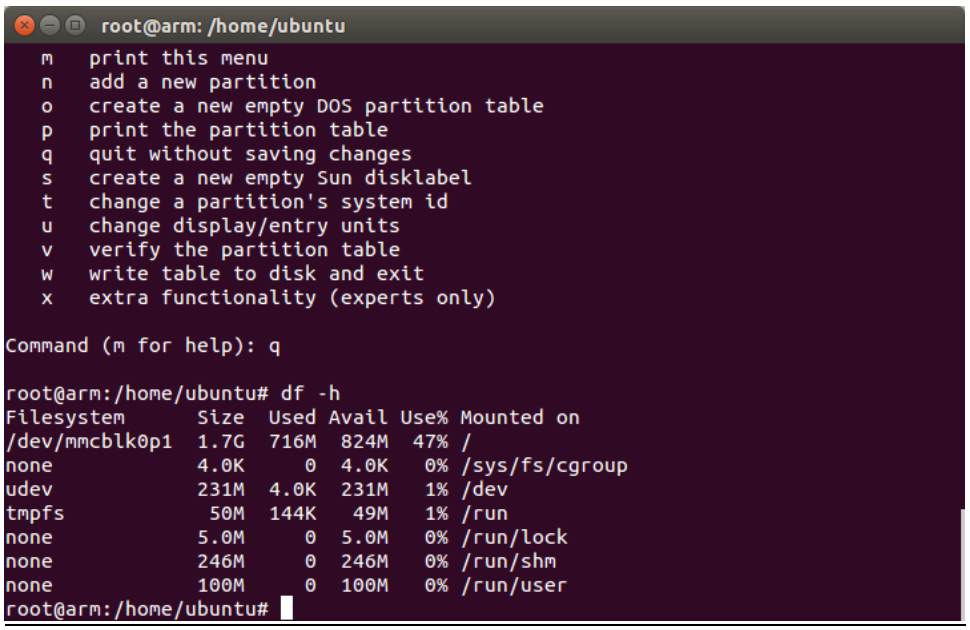
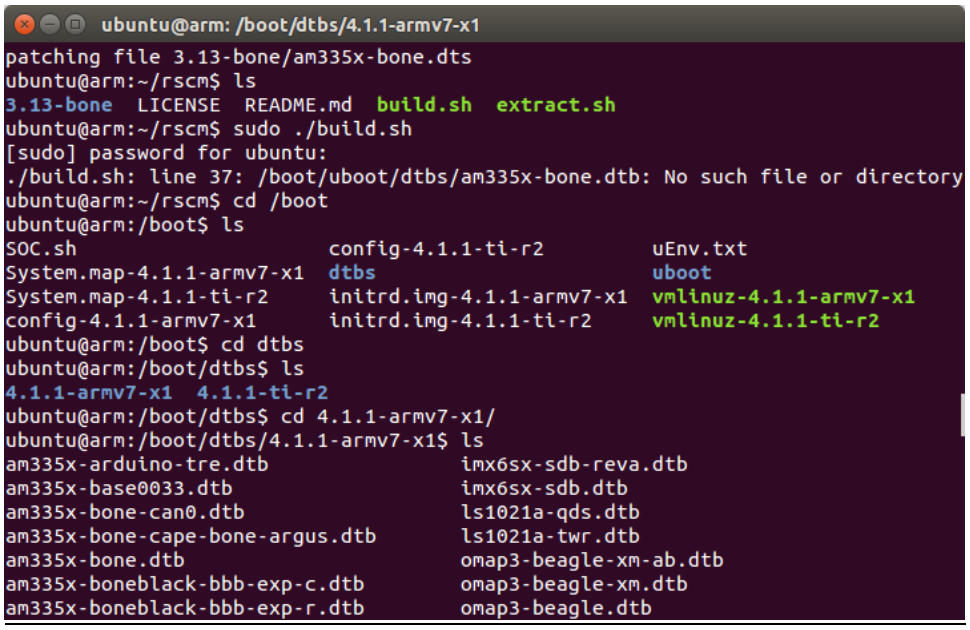


Figura 62. Archivos modificados para activar puertos UART



3. Discusión: Dado el éxito de las pruebas de vuelo autónomas con la PC y la necesidad de lograr autonomía completa. Se migró el software a una computadora de menores proporciones y más liviana. Esto con el objetivo de que dicha computadora pudiera estar a bordo del UAV y controlarla directamente a través de una comunicación serial.

Para este fin se decidió utilizar un *Beagle Bone Black* (BBB) por su capacidad de ejecutar una distribución Unix de manera nativa. El BBB viene con una distribución basada en Unix instalada de fábrica, dicha distribución se denomina *Ärgnstrom.*

Se intentó instalar ROS en *Ärngstrom*, sin embargo no fue posible debido a problemas de dependencias. Se instalaron las dependencias necesarias, mas algunas no existían para dicha distribución. Lo cual implicaría la creación de las dependencias faltantes. Dadas las restricciones de tiempo y considerando el hecho de que los mismos resultados podrían alcanzarse con otra distribución: se decidió migrar al Debían. Esta distribución es la misma que ejecuta la RPi, donde previamente se había logrado instalar ROS exitosamente pero por la velocidad de procesamiento se dejó a un lado. En ROS no existía una imagen de Debian lista para instalarse en el procesador del BBB, por lo cual fue necesario construir ROS desde el archivo fuente.

Nuevamente, se presentaron problemas de dependencias al momento del entorno de trabajo ROS. Conociendo los resultados que tuvieron dichos problemas en la distribución *Ärngstrom,* se decidió migrar a una distribución que aunque más pesada: no había presentado problemas para ejecutar ROS. Dicha distribución fue Ubuntu, misma distribución que corría la PC.

Se instaló Ubuntu exitosamente en la memoria EEPROM del BBB. Al momento de instalar ROS en el BBB se canceló la instalación debido a que la memoria EEPROM estaba llena. Se buscaron alternativas

para solucionar el problema de la memoria y se decidió que lo más factible era ejecutar Ubuntu desde una memoria SD de mayor capacidad que la EEPROM. Para lo cual fue necesario expandir la partición de la memoria SD luego de instalar Ubuntu.

Se logró instalar y ejecutar exitosamente Ubuntu desde la memoria SD. Dado que se estaba ejecutando desde un ambiente para el cual no había sido desarrollado. Se presentaron problemas para conectarse a la red y a internet. Por lo cual fue necesario modificar los *nameservers* del BBB. Con esto se solucionó el problema de la conectividad a internet. Sin embargo la conectividad a la red seguía siendo un problema. Posteriormente se descubrió que el problema era la red de UVG la cual se encontraba restringida y no brindaba los permisos necesarios para conectarse a la red. Por lo tanto fue necesario realizar todas las actualizaciones desde una red externa.

En Ubuntu, ROS se ejecutó de manera exitosa y sin menores contratiempos. Para el procesamiento de imagen era necesario contar con OpenCV. Luego de instalar dicha librería se presentaron problemas de compilación. Al analizar la causa de los problemas se identificó que ROS incluye librerías nativas OpenCV en su distribución, pero estas no se actualizan frecuentemente. Por lo que fue necesario eliminar los paquetes de OpenCV incluidos en ROS e instar OpenCV desde cero.

Para establecer comunicación entre el BBB y el PX4 se decidió utilizar una comunicación serial directa (a través de cables) entre los puertos UART del BBB y uno de los puertos de telemetría del PX4.

El BBB únicamente incluye un puerto UART activado por defecto el *ttyS0*. Se agregó al usuario del BBB al grupo *dialout* para conseguir acceso a los puertos y se logró de manera exitosa enviar comandos por el puerto. Al momento de conectar el pin *tx* del PX4 al *rx* del BBB. El BBB se paralizaba y no respondía a los comandos enviados por la computadora.

Se logró identificar el problema de las comunicaciones. Resulta que el puerto *ttyS0* comparte sus pines con el puerto USB que incluye el BBB. Por lo tanto se hizo evidente la necesidad de activar los puertos UART que no vienen activados por defecto en el BBB.

Se activaron todos los puertos disponibles *ttyS1 al ttyS6* exceptuando el *ttyS3* (pues comparte sus pines con el HDMI), en caso fueran útiles posteriormente. Dados los problemas con la red UVG, fue necesario realizar esta activación desde una red externa.

Finalmente, se logró establecer una comunicación exitosa entre el BBB y la tarjeta de control PX4. Para garantizar la correcta comunicación se enviaron comandos de activación al cuadricóptero desde el BBB. Los cuales ejecutó de manera satisfactoria.

XI. TRAZADO DE RUTAS Y ENVÍO DE PARÁMETROS AL PX4:

A. DISEÑO EXPERIMENTAL:

Para trazar las rutas de vuelo se partió de una matriz de posiciones que se deseaba: fueran alcanzadas por el UAV. En la computadora de compañía se crearía un nuevo nodo que enviara a través de *mavros* los parámetros de posición deseados. Dicho nodo publicará las posiciones deseadas al nodo de control de posición inicializado en el PX4. Par lo cual emplearía los servicios de mensajería inicializados previamente por el archivo *launch.* Asimismo, se utilizaran los datos recabados por los sensores para determinar la posición actual del UAV. Esto se logrará al subscribirse al nodo de estimación de posición que se ejecuta en la tarjeta controladora PX4.

Con la información de posición del UAV recibida por la computadora de compañía se comparará la posición actual con la posición deseada. Cuando se alcance la posición deseada se continuará con el resto de la matriz de posiciones hasta completar la trayectoria.

B. RESULTADOS:

Figura 63. Visualización nodos activos control de trayectoria

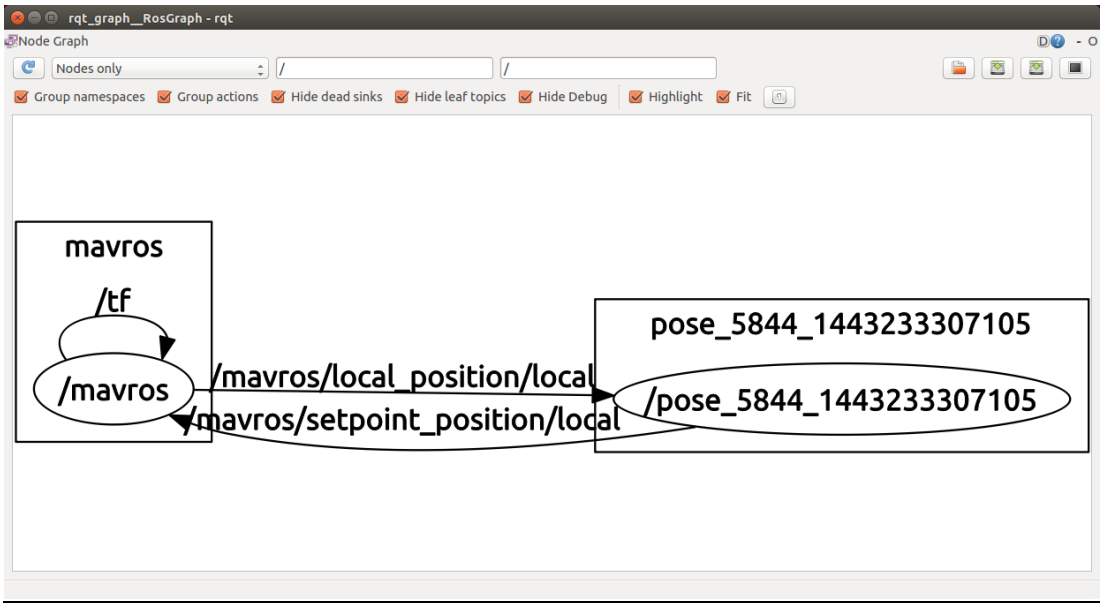


Figura 64. Servicios activos control de trayectorias

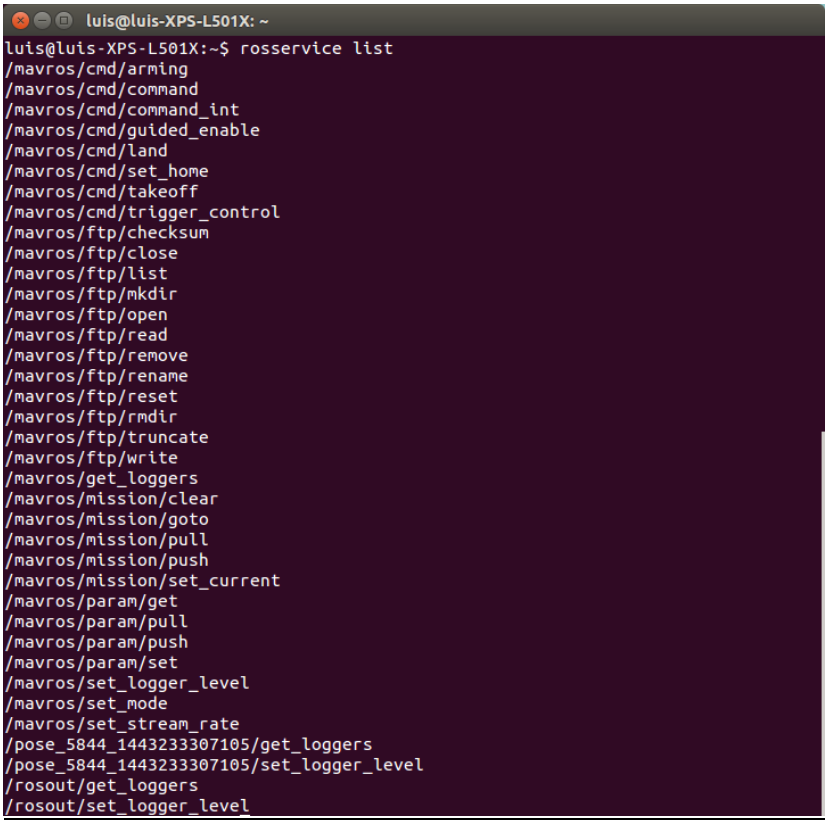
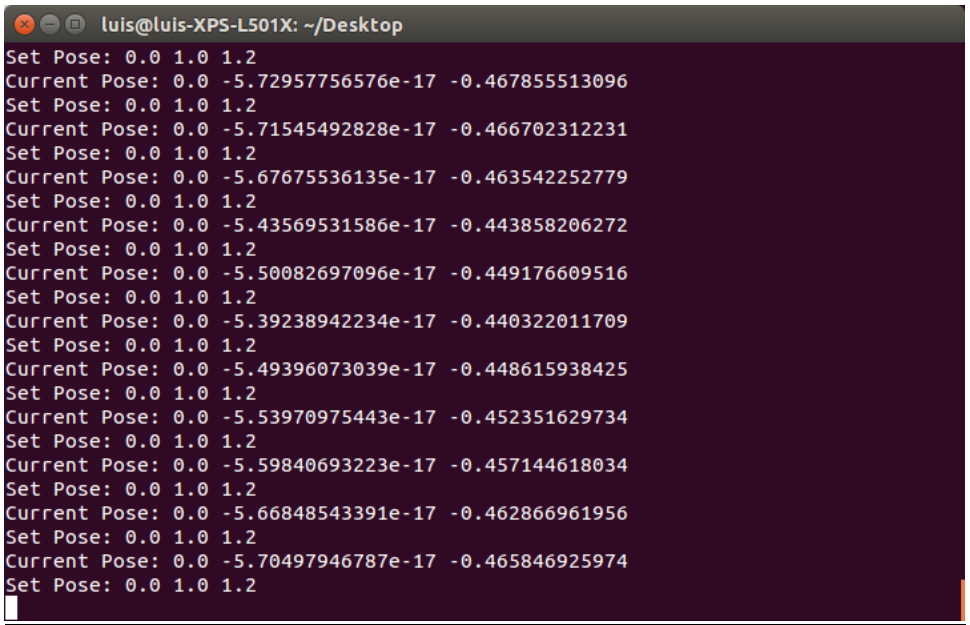


Figura 65. Detección de pose UAV



C. DISCUSIÓN

Como se puede observar en la figura 63, a diferencia de las pruebas de ROS iniciales (figura 57), ya existe la presencia de otro nodo que se subscribe a *mavros* para obtener la información de sensores y publica a *mavros* para enviar las posiciones deseadas.

Asimismo, se puede observar que hay mayor cantidad de servicios inicializados (figura 64) los cuales están solicitando al UAV su posición y enviando los parámetros de posición. La información que recibe la computadora sobre la pose del UAV garantiza la capacidad de suscribirse a los nodos de estimación de postura y posición, lo cual se observa en la figura 65.

Para determinar la capacidad de publicar al tema de control de posición y postura será necesario realizar las pruebas de vuelo autónomo.

XII. PRUEBAS DE VUELO

A. DISEÑO EXPERIMENTAL

Con toda la infraestructura de software y hardware implementada, se dispuso a realizar las pruebas de vuelo con el UAV. Considerando la integridad física del operador y de la estructura del UAV se diseñaron y fabricaron protectores de hélice para evitar que estos causaran algún accidente y a su vez estos protegerían las hélices del UAV.

Por cuestiones de seguridad, fue necesario implementar un sistema de Radio-Control que permitiera recuperar el control del cuadricóptero en todo momento en caso ocurriera algún problema con las pruebas autónomas. El Radio-Controlador utilizado fue un *FlySky RC Er9x*. Este Radio-Controlador trae 6 canales activados de fábrica, para los *switch* de seguridad y de *override* son necesarios 8 canales; por lo que fue necesario modificar el radio controlador para agregar los canales faltantes.

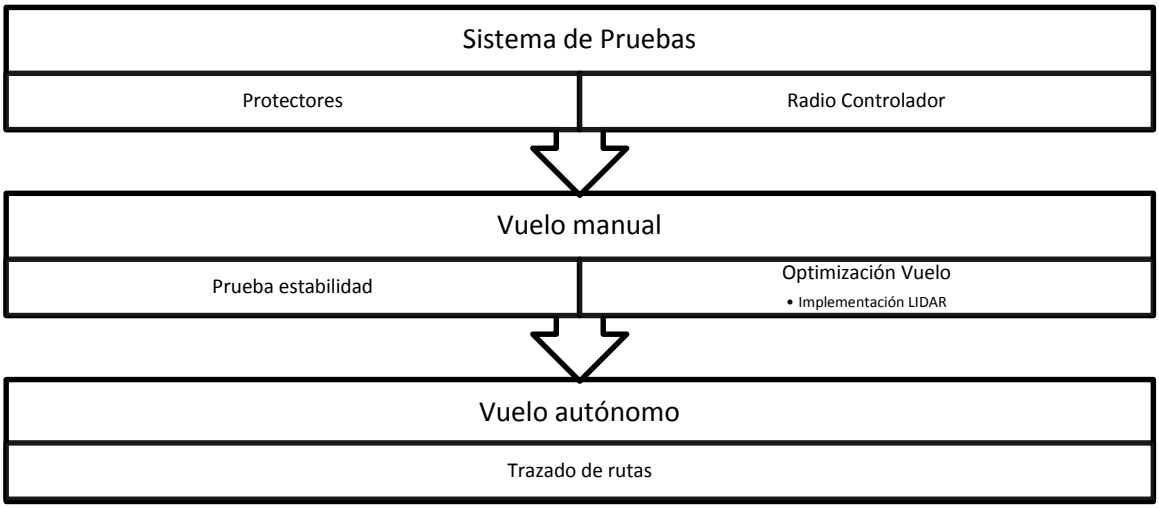
La primera prueba fue de altura y estabilidad, se enviaron comandos por medio del Radio-Controlador para que el UAV variara su altura. Cuando no se envían comandos al UAV, este debe permanecer en su última posición. Para garantizar la robustez del sistema se hicieron pruebas de vuelo en un ambiente hostil con fuertes corrientes de viento. En esa situación el cuadricóptero debería intentar permanecer en una posición fija.

Con base en el comportamiento del UAV, fue necesario realizar ciertas modificaciones para mejorar su desempeño durante el vuelo. Dichos arreglos consistieron en la sintonización de los controladores PID y calibración de los controladores de los motores.

Posteriormente se procedió a realizar la prueba de vuelo autónomo sin carga útil (es decir sin cámara multiespectral), para garantizar que en caso ocurriera algún error en la prueba: la cámara no se dañaría. Para esta prueba se trazaron distintas trayectorias y se verificó que llegara a las posiciones deseadas.

El desarrollo de las pruebas de vuelo se puede observar de mejor manera en el siguiente diagrama:

Figura 66. Diseño experimental pruebas de vuelo



B. RESULTADOS:

Figura 67. Radio-controlador utilizado

Figura 68. Diseño de protector de hélice

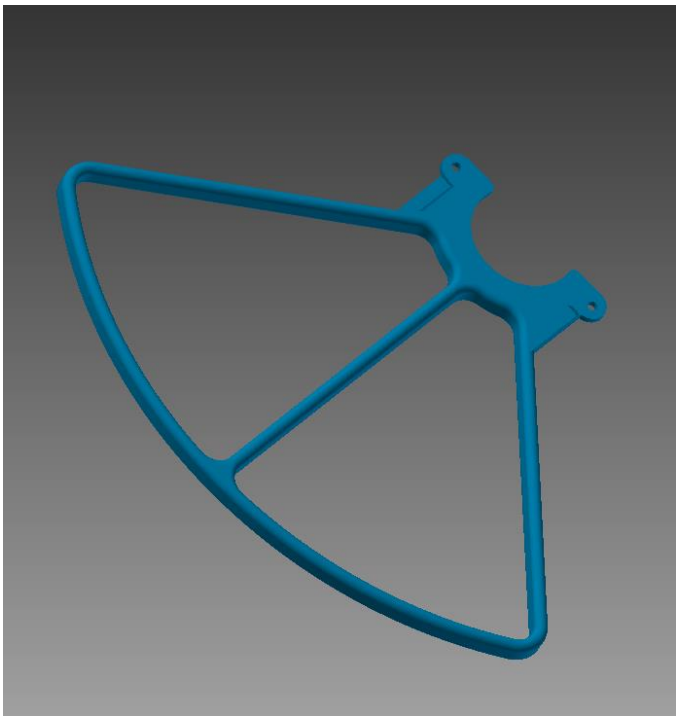


Figura 69. Gráfico actuadores

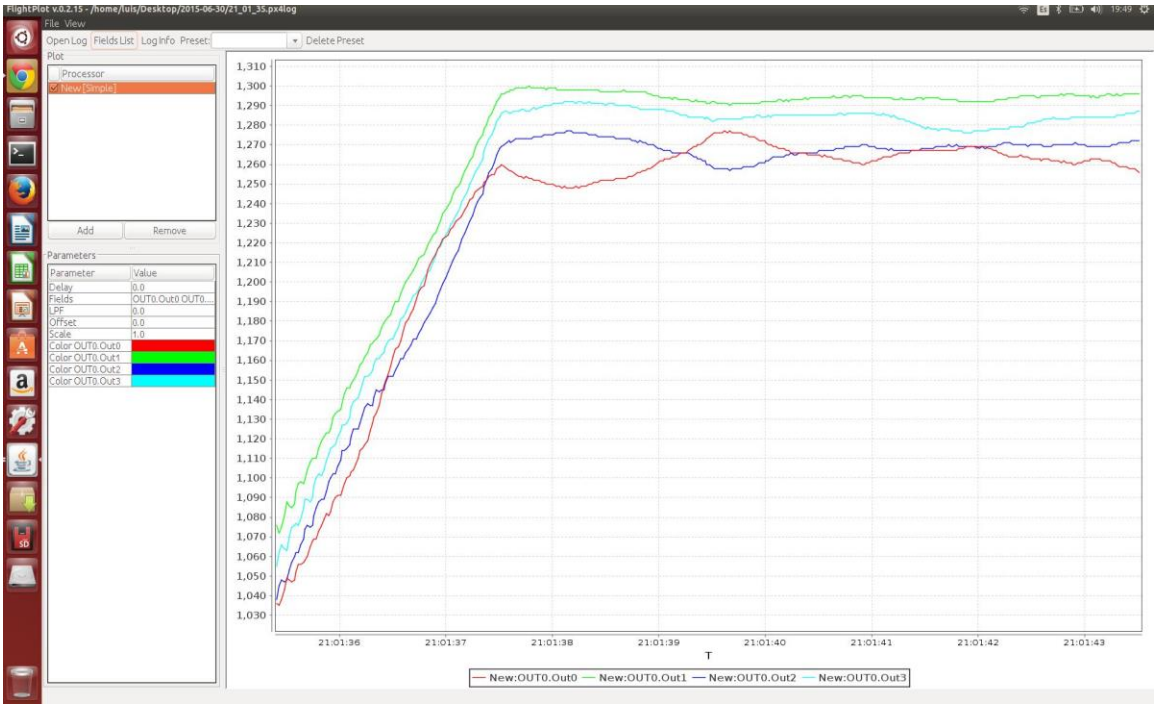


Figura 70. Respuesta del sistema control vertical previo sintonización

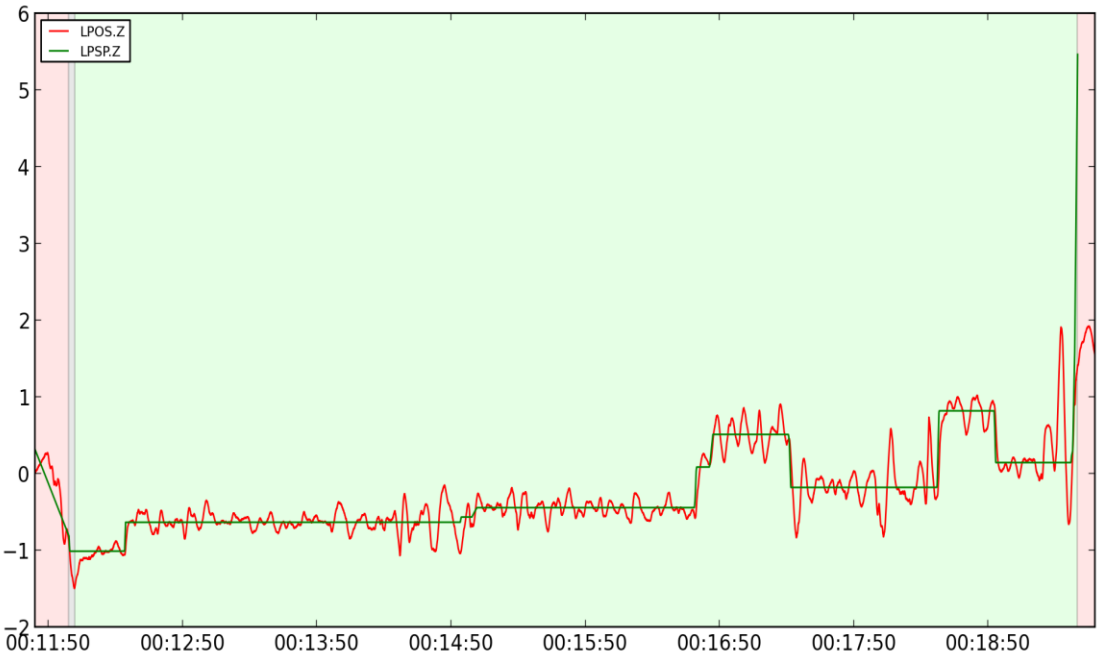


Figura 71. Respuesta posterior a sintonización y calibración ESC

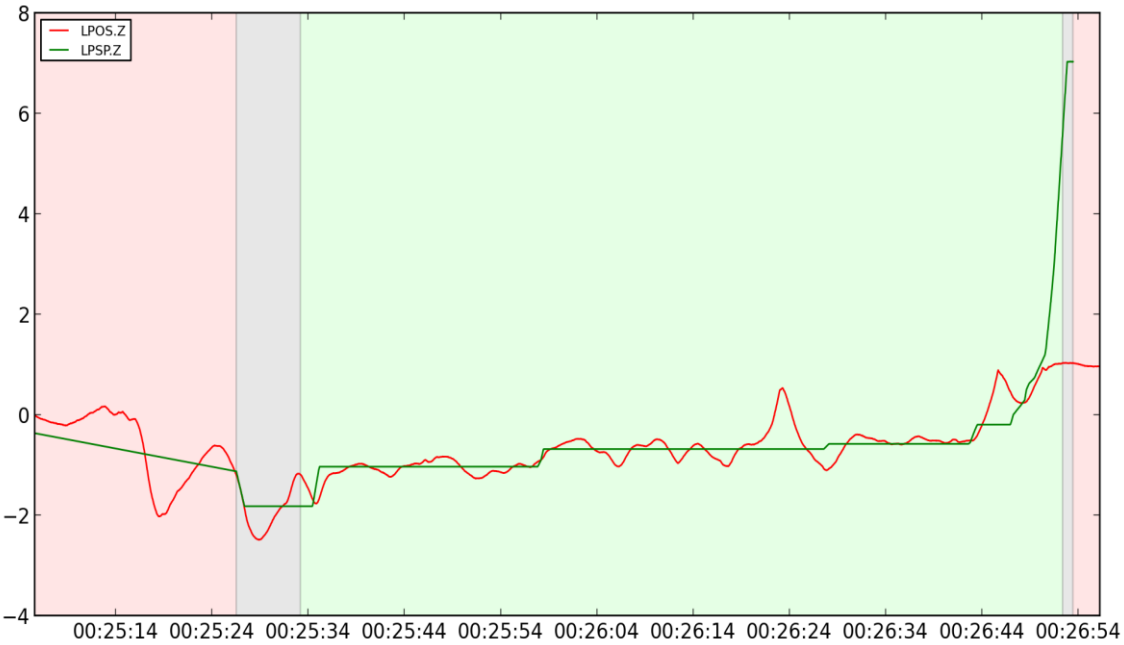
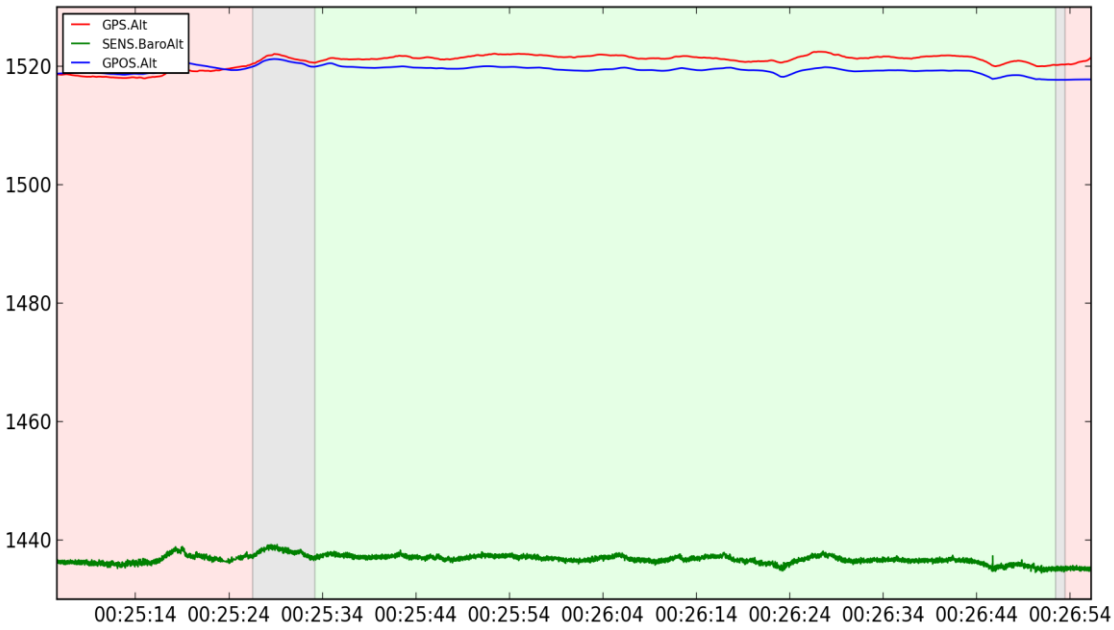


Figura 72. Estimado de altura



C. DISCUSIÓN:

En la Figura 67 se puede observar el controlador utilizado para realizar las pruebas de vuelo. Los *switches* de la parte superior derecha se adaptaron para recuperar el control del UAV durante las pruebas. La palanca derecha controlaba la dirección del cuadricóptero en los *ejes-x-y* mientras que la palanca derecha controlaba la altura y orientación del eje *yaw*.

El protector diseñado (figura 68) se fabricó con una impresora 3D utilizando material PLA. Estos protectores resultaron afectivos para pruebas iniciales a bajas velocidades, pues no presentaban mayor peso extra a la estructura. Sin embargo para pruebas a mayor velocidad y altura no lograron proteger las hélices del UAV. Diseñar un protector más resistente hubiera significado sacrificar capacidad de carga para los sensores extra. Por lo que para las pruebas de autonomía y muestreo con carga útil se decidió no colocar los protectores.

Las pruebas iniciales (Velasquez, Pruebas vuelo manual [Archivo de Video] 2015) demostraron que el cuadricóptero podía permanecer en la posición indicada en el plano-*xy* y su orientación *yaw*, sin embargo presentaba ciertas oscilaciones en el *eje-z* (altura). Al observar la Figura 70Es posible comparar la respuesta esperada contra la verdadera. Como se observa en dicha gráfica, el UAV presenta una respuesta rápida pero con sobreelevación. Dichas sobreelevaciones son en parte la causa de que el cuadricóptero no pueda permanecer en la altura indicada. Para corregir esto se realizó un proceso de sintonización el cual consistió en variar progresivamente los parámetros PID del controlador de posición. (Velasquez, Sintonización UAV [Archivo de Video] s.f.)

Otro aspecto detectado durante las pruebas iniciales y que podría incidir en el desempeño de vuelo del UAV es la velocidad a la que operan los motores. Pues no giraban a velocidades constantes durante el arranque, tal como se observa en la Figura 69. Para lograr que los rotores giraran a la misma velocidad se realizó un proceso de calibración conocido como *ESC Calibration*. Dicho proceso consistió en activar los rotores en todo el rango de operación, fijando los valores máximos y mínimos de cada rotor.

Para verificar la robustez del sistema se realizaron pruebas de posición baja ambientes más agresivos con corrientes de viento (Velasquez, Pruebas con Viento [Archivo de Video] s.f.). En dichas pruebas se logró observar como el UAV intenta contrarrestar exitosamente las corrientes de viento. Asimismo las oscilaciones de altura disminuyeron significativamente gracias a la sintonización efectuada con anterioridad. Si se observa la Figura 71, se ve que la respuesta del UAV si bien es más lenta ya no presenta sobreelevaciones, lo cual contribuye a una mejor respuesta en el *eje-z*.

Si bien la respuesta para mantener la altura mejoró considerablemente se puede observar una discrepancia entre los estimados de altura por presión barométrica y altura por GPS (figura 72). Para mejorar esta diferencia se optó por implementar un sensor de distancia laser (LIDAR) e integrarlo al estimador de posición. Cabe destacar que fue necesario modificar el driver del sensor LIDAR para que se adaptara a la arquitectura del PX4.

En resumen, las pruebas manuales resultaron ser satisfactorias pues se demostró que el UAV logró mantener efectivamente su orientación (ángulo *yaw*), la estabilidad bajo condiciones adversas, su posición en el plano *xy*, así como su posición vertical luego de las modificaciones pertinentes.

Respecto a las pruebas de vuelo autónomo (Velasquez, Prueba Vuelo Autonomo [Archivo de Video] s.f.), el cuadricóptero demostró una respuesta aceptable a las rutas trazadas. Inicialmente la tarjeta de control rechazó el modo de vuelo autónomo, pues no contaba con información de posición de los satélites GPS ya que estaba nublado. Las cámaras de flujo mencionadas en la sección de teoría presentan una alternativa en ambientes donde el GPS no es una alternativa. Dado que la intención del megaproyecto es volar en el campo, el cual generalmente se encuentra en áreas abiertas y considerando que se pretende volar a alturas superiores a los 15m donde las cámaras de flujo no funcionan: se descartó la implementación de una cámara de flujo.

En las pruebas iniciales se trazaron trayectorias geométricas simples, tal como se demuestra en (Velasquez, Prueba Vuelo Autonomo [Archivo de Video] s.f.) donde se realizó una trayectoria cuadrada. Para enviar los parámetros de vuelo se aprovechó el link de radio con la PC creado previamente. Al garantizar que el UAV era capaz de seguir una ruta trazada automáticamente se procedió a integrar el control de rutas al BBB que estaría a bordo del UAV.

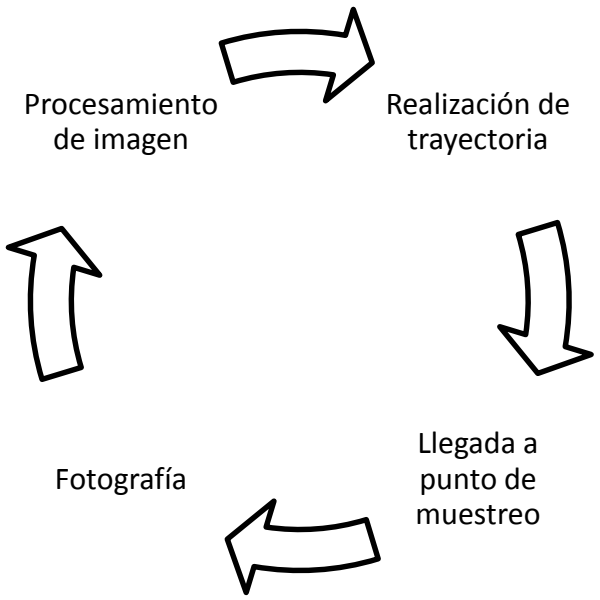
XIII. INTEGRACIÓN MÓDULOS Y RECOLECCIÓN DE DATOS:

A. DISEÑO EXPERIMENTAL

Posterior a las pruebas de vuelo autónomo de procedió a integrar la cámara multiespectral, la computadora de compañía (BBB) y el módulo de alimentación a la estructura. Tanto la toma de fotografías, el procesamiento de imagen como el trazado de rutas se realizó en la computadora de compañía.

Se agregaron las partes antes descritas a la estructura del UAV y se procedió a realizar las pruebas de muestreo. Dichas pruebas consistieron en cargar una ruta a la computadora de compañía la cual recorría los puntos predeterminados, el programa en la computadora de compañía verificaba la posición actual con la posición deseada; cuando estas coincidían la computadora de compañía tomaba una fotografía y continuaba con el recorrido. Dicho proceso se puede observar más claramente en el siguiente esquema:

Figura 73. Ciclo de muestreo UAV



B. RESULTADOS

Dado que los objetivos de este módulo conciernen solamente al control de UAV, únicamente se colocan los resultados del vuelo para la recolección de datos. Los resultados de las imágenes y su procesamiento se presentan en el módulo de procesamiento de imagen. (Argueta 2015)

Figura 74. Actuadores en condiciones de desbalance

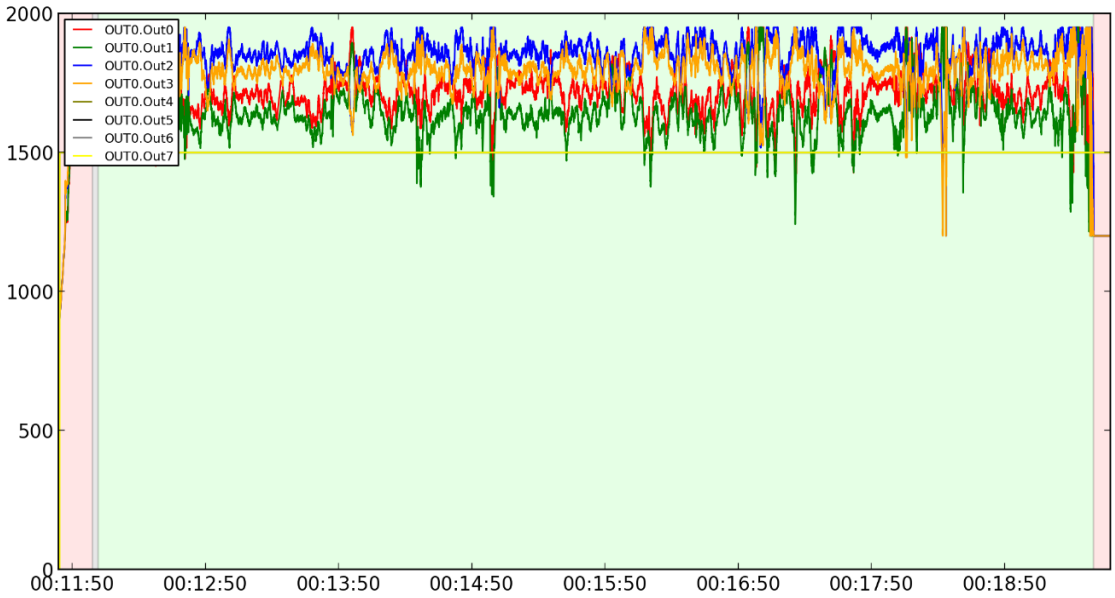


Figura 75. Posición x previo a choque

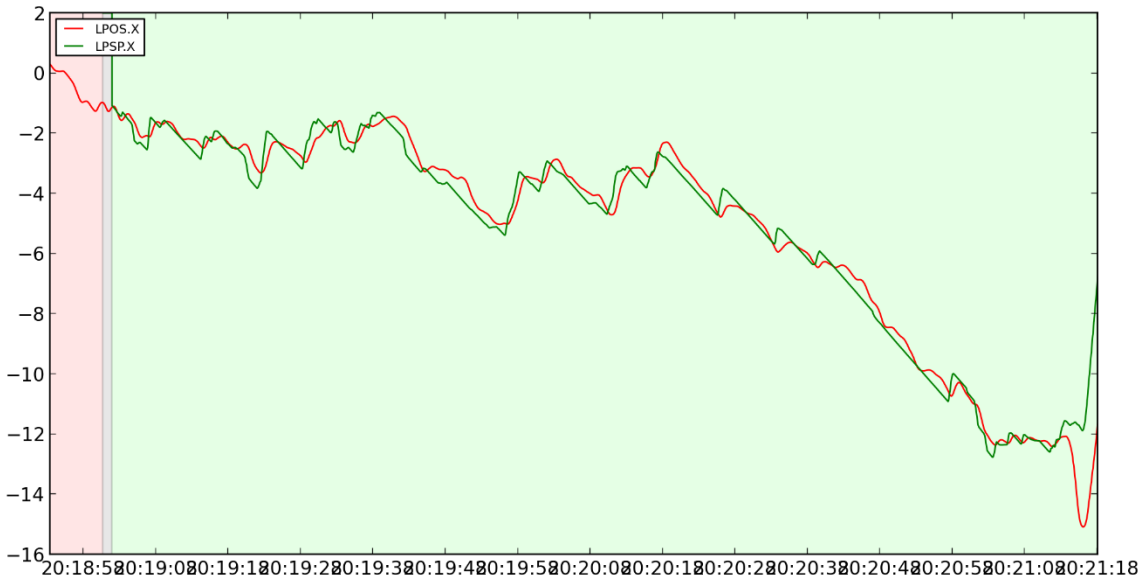


Figura 76. Velocidades previo a choque

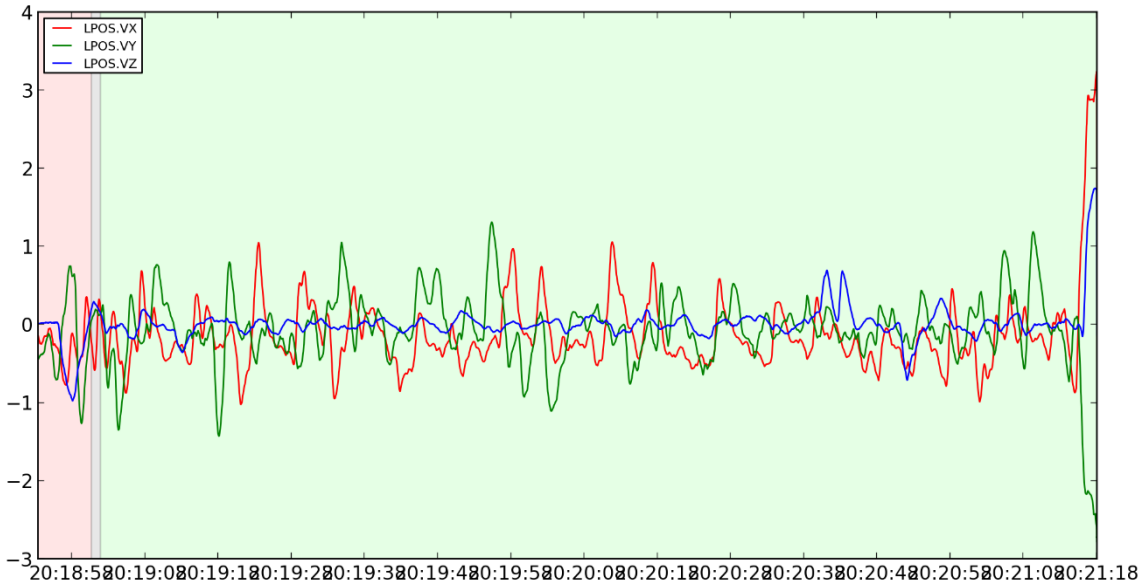


Figura 77. Actuadores previo a choque

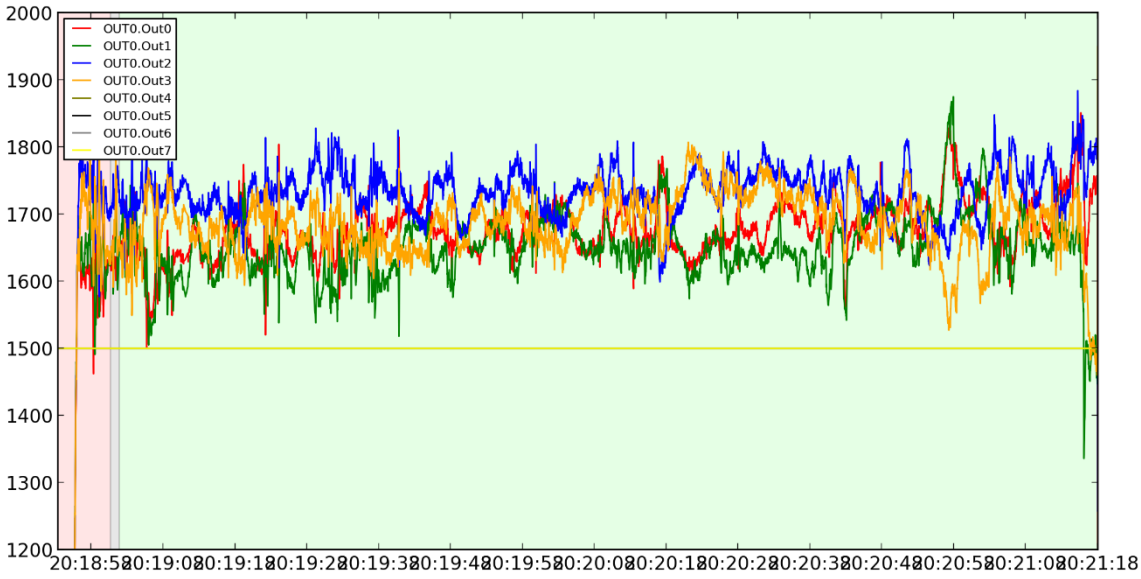
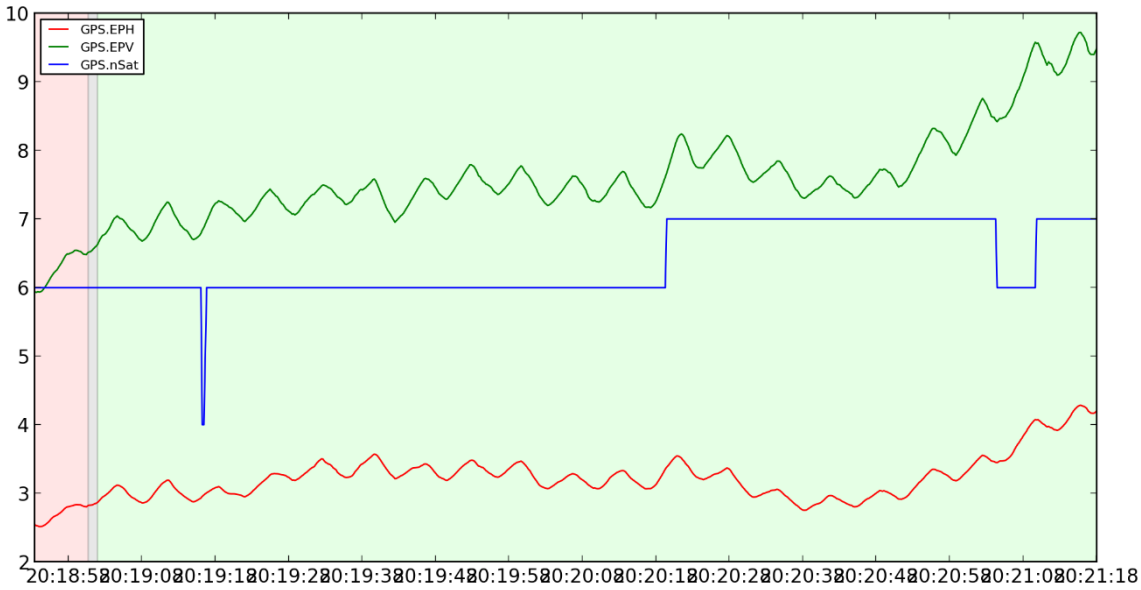


Figura 78. Incertidumbre GPS previo a choque



C. DISCUSIÓN

Durante las pruebas iniciales con la cámara integrada el UAV volcó al momento del despegue (Velasquez, Desbalance UAV [Archivo de Video] s.f.). Al analizar las causas del accidente se identificó en la figura 74, que dos de los actuadores estaban trabajando a más de sus revoluciones nominales y se saturaban. Esto fue producto de un desbalance de la estructura; por lo que se trasladaron los sensores y computadora de compañía hacia la parte frontal de la estructura para contrarrestar dicho desbalance.

Las pruebas de muestreo transcurrieron con normalidad al inicio, el UAV siguió la ruta establecida y realizó la captura de fotografías. Sin embargo al final de la prueba el UAV perdió el control y se trasladó a una posición no considerada donde impacto contra el suelo. Al analizar las causas del accidente se puede observar que los actuadores 1 y 3 (figura 77) disminuyeron su velocidad, produciendo que el UAV se trasladara en la dirección del *eje-x*. Al observar la figura 76 se hace evidente que el cuadricóptero alcanzó grandes velocidades (más del triple de su velocidad nominal). Si se observa la

Figura 75 se logra observar que el UAV perdió noción de su posición *x* brindando señales erróneas. Ante esto el controlador intentó compensar el repentino cambio de posición para llegar a la "supuesta" posición correcta donde finalmente chocó.

Al analizar las causas de la mala lectura de los sensores se logró identificar un aumento en la incertidumbre del GPS (figura 78). Dicho aumento del a incertidumbre fue producto de la ubicación del dron en ese momento. Pues previo al accidente se volaba cerca de un edificio lo cual produjo que la calidad de la señal GPS disminuyera y produjera errores en la estimación. Para las pruebas posteriores se realizaron las pruebas en ambientes más amplios donde el GPS no tuviera mayor interferencia.

XIV. MÓDULO DE PROCESAMIENTO DE IMÁGENES

A. DISEÑO EXPERIMENTAL

1. Selección de cámara. La elección de la cámara a utilizar se basó en la elección del índice de vegetación a implementar. Siendo este proyecto una primera fase en la línea de análisis de plantaciones agrícolas, se optó por utilizar el índice de vegetación de diferencias normalizado (Normalized Difference Vegetation Index o NDVI por sus siglas en inglés). El índice NDVI es uno de los índices más utilizados mundialmente por ser un índice simple de calcular y a su correlación con muchas variables dentro de un ecosistema. El índice NDVI nos devuelve un análisis cualitativo de la actividad fotosintética en un cultivo sin la necesidad de tener que aplicar factores de corrección atmosférica o en base al tipo de suelo.

La implementación del índice NDVI requiere contar con la respuesta espectral de la banda correspondiente al color rojo y con la de la banda correspondiente al infrarrojo cercano (Near Infrarred o NIR por sus siglas en inglés), de acuerdo a su ecuación:

$$NDVI = \frac{NIR - Red}{NIR + Red}$$

Donde NIR y Red hacen referencia a la intensidad de reflexión en las bandas de NIR y la correspondiente al color rojo, respectivamente. Puesto que las bandas necesarias para realizar el cálculo del índice son solo dos, una cámara del tipo multiespectral es capaz de capturar la información necesaria.

Actualmente existen empresas que se dedican a la producción de cámaras multiespectrales para su aplicación en la agricultura, ejemplo de ello es la corporación Tetracam. Tetracam es una empresa que produce y distribuye sistemas de captura de imágenes multiespectrales. Los productos de Tetracam se dividen en dos grupos. El primer grupo son las cámaras digitales para agricultura (Agricultural Digital Camera o ADC por sus siglas en inglés) que consisten en cámaras NGR basadas en el funcionamiento de los sensores presentes en los satélites Landsat. El segundo grupo son los arreglos de múltiples cámaras (Multiple Camera Array o MCA por sus siglas en inglés) que consisten en arreglos de cuatro, seis o hasta doce cámaras digitales que pueden ser configuradas para detectar las bandas deseadas por el usuario. Los productos de Tetracam se caracterizan por ser compactos y livianos. Además permiten la captura de imágenes de forma manual, programada o activada de forma remota. Las cámaras cuentan con un GPS que permite asociar cada fotografía con las coordenadas en donde se capturó la imagen. El costo de los productos de Tetracam asciende a los tres mil dólares y varía de acuerdo al modelo (Tetracam, 2015).

Por otra parte, existen comunidades que desarrollan este tipo de tecnologías a bajos costos, con el fin de promover la investigación y el desarrollo del medio ambiente, como lo es Public Lab. Public Lab desarrolló su propia cámara NGR a través del programa de Kickstarter. El resultado fue la cámara conocida como Infragram Point & Shoot. Esta cámara es en realidad un cámara digital conocida como Mobius Action Cam que ha sido modifica en el proceso de producción para reemplazar el filtro que bloquea la banda del NIR por un filtro Wratten 25A rojo que permite la captura del espectro del NIR a través del canal correspondiente al canal azul. El costo de esta cámara es de ciento veinticinco dólares a la fecha (Public Lab, 2015).

La cámara Infragram Point & Shoot se caracteriza por su portabilidad, midiendo 61 x 35 x 18 mm y pesando 38g. El lente de la cámara posee un ángulo de visión de 87° y es capaz de capturar fotografías de 2304 x 1536 píxeles y video en 1080p a 30 fps (o en 720p a 60 fps). Debido a su portabilidad, no incluye GPS ni pantalla, siendo su única interfaz un puerto mini-USB que también es utilizado para cargar la batería de la cámara (Dashboard Camera Reviews, 2015).

Las cámaras NGR presentan un inconveniente por el tipo de filtro que éstas usan. Los filtros NGR utilizan filtros rojos que permiten el paso del espectro NIR a través del canal correspondiente al espectro del color azul. Sin embargo, una vez el espectro NIR atraviesa el filtro, éste incide en el filtro Bayer que constituye los sensores de la cámara. Al incidir en el filtro Bayer, el espectro NIR no solo pasa a través del filtro azul, sino que se fuga a través de los otros filtros. Se ha comprobado que el espectro NIR se fuga en mayor manera a través del filtro rojo. Por tanto, una imagen capturada por una cámara NGR puede presentar datos erróneos debido a la fuga del espectro NIR a través de los filtros rojos que componen al filtro Bayer. Este efecto sucede también en cámaras NBG pero en menor proporción pues el espectro NIR se fuga en menor cantidad en el filtro azul que compone al filtro Bayer (Public Lab, 2013).

Una característica importante para la elección de la cámara fue también el balance de blanco. Se conoce como balance de blanco (White balance) al proceso en el cuál se ajusta la imagen para corregir variaciones en la calidad de la luz incidente en la imagen de forma que los objetos que son blancos en la escena también aparezcan de color blanco en la imagen. Las cámaras simples tienen la capacidad de manejar este problema de forma automática. Cámaras más avanzadas permiten al usuario modificar el balance de blanco permitiendo ajustes personalizados en base a las necesidades del usuario. El problema con las cámaras modificadas para captar NIR es que la cámara trata de realizar un balance de blanco asumiendo que se está captando únicamente el espectro visible, lo que conlleva a realizar un balance incorrecto generando imágenes con información falsa. Para ello es necesario que la cámara permita realizar un balance de blanco de forma manual (Public Lab, 2012).

Una ventaja que presenta la cámara Infragram Point & Shoot es que, no solo permite personalizar el balance de blanco, sino que además se distribuye con una configuración predeterminada que arregla el

problema de balance de blanco al momento de captar el espectro NIR. La configuración se aplica a través de la memoria micro SD que se puede incorporar a la cámara y puede ser accedida como un dispositivo de almacenamiento en una computadora a través de un adaptador SD. Además, puesto que se trata de un producto hecho por una comunidad científica, tienen a su disposición actualizaciones de los archivos de configuración en base a sus últimos hallazgos realizados (Public Lab, 2015).

De acuerdo a los alcances del proyecto y a las características que esta ofrece, se optó por utilizar una cámara NGR, específicamente la cámara Infragram Point & Shoot para la captura de imágenes multiespectrales.

2. Selección de plataforma. La elección de la plataforma de desarrollo se basó en la decisión de querer realizar el procesamiento de imágenes inmediatamente después de capturar la imagen. En este caso, la plataforma de desarrollo sería la encargada de capturar la imagen, pre-procesarla, segmentarla, transformarla y almacenarla en memoria para que el módulo de comunicaciones fuese capaz de acceder a las imágenes. Por ello, se deseaba que la plataforma de desarrollo a utilizar tuviera el desempeño necesario para ser capaz de realizar todas estas tareas en el menor tiempo posible, posibilitando la adquisición de más imágenes en un menor tiempo.

Para la captura de imágenes se optó por utilizar la cámara NGR Infragram Point & Shoot, la cual podía ser configurada como webcam para mayor facilidad de uso a través de una conexión USB. Esto requería que la plataforma a utilizar contara con una conexión USB y además con una unidad de memoria suficientemente grande para almacenar las capturas e imágenes procesadas.

Para llegar a cumplir con estos requerimientos se optó por utilizar una plataforma de desarrollo basada en el sistema operativo Linux. Este tipo de plataformas poseen la ventaja que, al ser plataformas de desarrollo libre, cuentan con bastante documentación en línea y de acceso público. En los últimos años han sido lanzadas múltiples plataformas de este tipo que varían de acuerdo a sus especificaciones y desmpeño computacional.

En mayo del 2014 Adafruit, una empresa dedicada al desarrollo, aprendizaje y venta de recursos para proyectos electrónicos, llevó a cabo una prueba de mercado (testbench) para comparar las plataformas de desarrollo basadas en Linux más populares. En mayo de 2015 fue la última vez que se actualizó el estudio ofreciendo un análisis basado en desempeño computacional, potencia utilizada, temperatura de funcionamiento y desarrollo. En esta prueba de mercadeo participaron las plataformas Arduino Yun, Beaglebone Black, Intel Galileo y Raspberry Pi. En la sección de anexos se presentan los resultados de la prueba de mercado realizada, la cual fue utilizada como parámetro para la elección de la plataforma a utilizar (Adafruit, 2015).

Los resultados de la prueba de mercado muestran que la plataforma Beaglebone Black (BBB) posee el más alto desempeño en operaciones de memoria y de manejo de enteros, los cuales son fundamentales para la manipulación de los archivos de imagen y el cálculo del índice NDVI. Además, su desempeño de unidad de punto flotante no es significativamente menor al de las otras plataformas. En cuanto a la potencia y temperatura de funcionamiento se observó que el BBB se encuentra en el promedio de las plataformas analizadas, lo cual resulta sorprendente pues con el desempeño computacional que posee se esperaría que fuese la plataforma con mayor consumo de potencia y en llegar a las más altas temperaturas. Además, puesto que posee un pequeño microcontrolador integrado en la plataforma, permite el manejo en tiempo real de los puertos de entrada y salida (Adafruit, 2015).

De acuerdo a los alcances del proyecto y a las características que esta ofrece, se optó por utilizar la plataforma de desarrollo Beaglebone Black para llevar a cabo el trabajo computacional del proyecto.

3. Configuración de la plataforma. BeagleBone Black (BBB) es una plataforma de desarrollo de bajo costo que cuenta con el soporte de una comunidad de desarrolladores conocida como Beagleboard.org. Beagleboard.org es una fundación estadounidense sin fines de lucro dedica a proveer recursos de aprendizaje y promover el diseño y uso de hardware y software de distribución libre para realizar proyectos de computación embebidos. Beagleboard.org es el resultado del esfuerzo de muchas personas, incluyendo entre éstas empleados de Texas Instruments interesados en crear plataformas de desarrollo de alto desempeño. Las plataformas de desarrollo producidas son computadoras de una placa, sin unidades de enfriamiento y de bajo costo basadas en los procesadores de baja potencia de Texas Instruments serie ARM Cortex-A. Las plataformas desarrolladas están diseñadas para correr distribuciones de uso libre Linux (Beagleboard, 2015).

El BBB se caracteriza por tener un procesador AM335x ARM Cortex-A8 con frecuencia de reloj de 1 GHz, una memoria RAM DDR3 de 512 MB, 4Gb de memoria flash interna, un acelerador de gráficos 3D, una unidad de punto flotante NEON y dos microcontroladores de 32 bits integrados. En cuanto a su conectividad posee un puerto micro USB para alimentación y comunicación SHH, un puerto USB genérico, un puerto RJ-45 para Ethernet, un puerto micro HDMI y dos hileras de 46 pines de entrada y salida de propósito general (General Purpose Input/Output o GPIO por sus siglas en inglés). La plataforma es compatible con algunas distribuciones de uso libre, entre ellas Debian, Ubuntu, Android y Cloud9 (Beagleboard, 2015).

La configuración del BBB se basó no solo en los requerimientos de éste módulo, sino además en los requerimientos de otros módulos. Para este módulo se requería que la distribución de Linux a instalar fuese capaz de soportar el software de Python y el compilador de C++, así como sus librerías. Posteriormente, en

base a los requerimientos del módulo de control se optó por instalar la distribución de Linux de Ubuntu 14.04.2 LTS. Puesto que las librerías a utilizar ocupaban mucho espacio en memoria, fue necesario cargar el sistema operativo en la memoria micro SD en vez de utilizar la memoria flash interna de la plataforma. Para ello se expandió la memoria micro SD a una con una capacidad de 8 Gb de almacenamiento. El proceso para cargar el sistema operativo en la memoria micro SD se explica detalladamente en la página oficial de Beagleboard.org (Beagleboard, 2015).

Una vez cargada la imagen de Ubuntu en la memoria micro SD se procedió a comunicarse con el BBB a través del cliente SSH (Secure Shell) por medio de la red de área local. El usuario y contraseña predeterminada para una imagen de Ubuntu recién instalada es de "ubuntu" y "temppwd" respectivamente (sin las comillas). Estando conectado al BBB se requirió expandir la imagen recién instalada en toda la memoria micro SD ingresando para ello los comandos en la consola del BBB (Solarian Programmer, 2014):

- cd /opt/scripts/tools
- git pull
- sudo ./grow_partition.sh
- sudo reboot

Para finalizar la instalación del sistema operativo se descargaron e instalaron las actualizaciones para los paquetes y librerías instalados de forma predeterminada en el BBB ingresando para ello los comandos en la consola del BBB (Solarian Programmer, 2014):

- sudo apt-get update
- sudo apt-get upgrade

4. Desarrollo del algoritmo. Para desarrollar el algoritmo encargado de calcular el índice NDVI se optó por utilizar las librerías de OpenCV (Open Source Computer Vision Library). OpenCV es un conjunto de librerías de software de uso libre utilizadas para desarrollar software en el área de visión computacional (Computer Vision) y entrenamiento automático (Machine Learning). El conjunto de librerías posee más de 2500 algoritmos optimizados empleados para la detección y reconocimiento facial, identificación de objetos, clasificación de movimientos y acciones humanas, registro de objetos en movimiento, extracción de modelos 3D de un objeto, unión de varias imágenes para producir una única imagen de alta resolución de una sola escena, etc. OpenCV fue escrito de forma nativa en el lenguaje de C++ pero posee interfaces para ser utilizado en otros lenguajes de programación como C, Python, Java y Matlab (Iseez, 2015).

La elección de utilizar las librerías de OpenCV se basó en la gran cantidad de documentación que existe para dichas librerías, además de ser capaz de utilizarse en distintos lenguajes de programación sin presentar grandes variantes en el código de la implementación. Siendo este proyecto una primera fase en la línea de análisis de plantaciones agrícolas se creyó conveniente utilizar un conjunto de librerías suficientemente robustas para que en próximas fases se pueda implementar mejoras al código sin la necesidad de hacerlo desde cero. Entre las mejoras a realizarse empleando las librerías de OpenCV se encuentran la detección de objetos, extracción de modelos 3D (para realizar el mapeo topográfico de un cultivo), unión de varias imágenes para producir una única imagen de alta resolución de un solo cultivo, entre otras.

Primeramente, se optó por desarrollar el algoritmo utilizando el lenguaje de programación de Python. Python es un lenguaje de programación sencillo de utilizar, con compatibilidad en la mayoría de plataformas existentes y que por su licencia de distribución libre se puede utilizar incluso para el desarrollo de productos de uso comercial. El ser un lenguaje sencillo de utilizar permite que exista una gran cantidad de documentación en cuanto al uso del lenguaje para el desarrollo de software.

Una vez desarrollado el algoritmo en Python se decidió trasladar el código al lenguaje de programación de C++ con la finalidad de tener un procesamiento de imagen mucho más rápido, siendo este el lenguaje nativo en que fue escrito OpenCV, que permitiera una mayor captura de imágenes. Para desarrollar el algoritmo en el lenguaje de programación de C++ se utilizó el software de Visual Studio. Visual Studio es un entorno de desarrollo integrado para sistemas operativos Windows. Visual Studio soporta múltiples lenguajes de programación, incluyendo entre ellos Python y C++.

El software desarrollado se creó siguiendo las etapas que conlleva el procesamiento de una imagen. En principio el software se encarga de capturar la imagen multiespectral. Utilizando la librería de VideoCapture el programa accede a la cámara Infragram Point & Shoot, configurada como webcam, y esta comienza a tomar video. Posteriormente se lee el primer *frame* de video que se tiene en el buffer y este se envía como parámetro al algoritmo encargado de calcular el índice NDVI.

El índice NDVI es calculado utilizando los canales correspondientes al rojo y al azul, los cuales contienen a su vez la intensidad de reflectancia del espectro rojo y NIR respectivamente. Previó a utilizar la ecuación para el cálculo del índice NDVI, los canales rojo y azul son convertidos a números de punto flotante de 32 bits para evitar perder información al realizar las operaciones matemáticas que involucran el cálculo del índice NDVI. El algoritmo implementado devuelve un matriz del alto y ancho de la imagen original en donde cada celda corresponde al valor NDVI de cada píxel en la imagen.

Como resultado final del programa se deseaba obtener una imagen en color falso que correspondiera a un gradiente de los índices NDVI presentes en la imagen. Para crear esta imagen en color falso se desarrollaron dos alternativas utilizando como base la matriz resultante del cálculo del NDVI para cada píxel. La primera consiste en utilizar las librerías de OpenCV para aplicar lo que se conoce como un ColorMap. Este algoritmo segmenta los valores de 0 a 255 en pequeños intervalos del mismo tamaño y a cada uno le asigna un color. Posteriormente el algoritmo asigna a cada píxel en la imagen uno de los colores en su tabla de acuerdo al valor actual del píxel. Como resultado se tiene una imagen en color falso. El algoritmo posee de forma predeterminada distintas paletas con distintos colores para ser utilizados. Sin embargo, no se puede personalizar una paleta de colores para ser utilizada en la implementación del algoritmo. Para emplear este algoritmo es necesario que el rango de valores para cada píxel se encuentre en el rango de 0 a 255. Puesto que el índice corresponde a un valor decimal que se encuentra en el rango de -1 a 1, este se opera matemáticamente para volverlo un número entero que se encuentre en el rango de 0 a 255 y poder aplicar el algoritmo.

La segunda alternativa consiste en emular el algoritmo creado por OpenCV, con la facilidad de poder personalizar la paleta de colores a utilizar para pintar la imagen. Además, esta alternativa permite segmentar el rango de 0 a 255 en la cantidad de intervalos deseados, en caso desee tener una representación mucho más sensible al cambio de valores en los píxeles o no. Para la implementación de este algoritmo se llevan a cabo una secuencia de operaciones matriciales con el fin de cambiar el valor de cada píxel por un vector que corresponde al color asignado de acuerdo al valor actual del píxel.

5. Implementación del algoritmo en la plataforma. Para implementar los algoritmos desarrollados en la plataforma del BeagleBone Black (BBB) fue necesario primero instalar las librerías necesarias, en este caso las librerías de OpenCV. Para ello primero se instalaron algunos requisitos utilizados por las librerías de OpenCV ingresando para ello los comandos en la consola del BBB (Solarian Programmer, 2014):

- sudo apt-get install build-essential cmake pkg-config
- sudo apt-get install libtiff4-dev libjpeg-dev libjasper-dev libpng12-dev
- sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev

Posteriormente fue necesario descargar la última versión de las librerías de OpenCV del repositorio de GitHub, repositorio oficial de los desarrolladores de OpenCV, ingresando para ello el comando en la consola del BBB (Solarian Programmer, 2014):

- git clone https://github.com/Itseez/opencv.git

Una vez descargados los archivos de las librerías de OpenCV es necesario compilar los archivos para construir las librerías de OpenCV para ser utilizadas ya se en Python o C++. Para ello se utiliza la herramienta de CMake, el cual es una herramienta presente en los sistemas operativos de Linux para compilar de forma sencilla grandes proyectos o librerías, como lo son las librerías de OpenCV. Esto se realiza ingresando para ello los comandos en la consola del BBB (Solarian Programmer, 2014):

- cd opencv
- mkdir build && cd build
- cmake -D CMAKE_BUILD_TYPE=RELEASE -D CMAKE_INSTALL_PREFIX=/usr/local -D WITH_CUDA=OFF -D WITH_CUFFT=OFF -D WITH_CUBLAS=OFF -D WITH_NVCUVID=OFF -D WITH_OPENCL=OFF -D WITH_OPENCLAMDFFT=OFF -D WITH_OPENCLAMDBLAS=OFF -D BUILD_opencv_apps=OFF -D BUILD_DOCS=OFF -D BUILD_PERF_TESTS=OFF -D BUILD_TESTS=OFF -D ENABLE_NEON=on ..
- make
- sudo make install
- sudo ldconfig

Una vez instaladas las librerías de OpenCV se es posible trasladar los algoritmos desarrollados en la computadora a la plataforma de desarrollo. En el caso de Python, puesto que se trata de un intérprete, únicamente es necesario trasladar el archivo a la plataforma y de forma inmediata es posible ejecutarlo ingresando para ello el comando en la consola del BBB (Solarian Programmer, 2014):

- python Nombre_del_Archivo.py

Para la ejecución del código desarrollado en C++ es necesario primero compilar el programa. En el proceso de compilación es necesario además hacer referencia a las librerías utilizadas en el código del programa para que el compilador sepa asociar correctamente los métodos utilizados en el desarrollo del algoritmo. La compilación correcta del algoritmo desarrollado se logra ingresando para ello los comandos en la consola del BBB:

- g++ Nombre_del_Archivo.cpp –o Nombre_del_Ejecutable –lopencv_core –lopencv_imgproc –lopencv_highgui –lopencv_videoio –lopencv_imgcodecs

Luego, para ejecutar el archivo basta con ingresar el comando en la consola del BBB (Solarian Programmer, 2014):

- ./Nombre_del_Ejecutable

6. Pruebas de la implementación en vuelo. Para realizar las pruebas del algoritmo implementado mientras el UAV se encontraba en vuelo fue necesario realizar algunas configuraciones previas, tanto a la plataforma como al algoritmo. En principio fue necesario configurar el algoritmo para que este capturara una determinada cantidad de fotos en un tiempo estipulado. Versiones anteriores del algoritmo permitían capturar únicamente una imagen y procesarla. Las versiones del algoritmo en la computadora permitían capturar imágenes de forma continua, procesarlas de forma inmediata y desplegar una ventana en donde se podía apreciar tanto la imagen capturada como la imagen procesada. Sin embargo, para el caso de la implementación en el BeagleBone Black (BBB) esto no era una opción, pues no se contaría con una salida de video. Además, resultaba proco práctico capturar imágenes de forma continua, pues almacenar todas las imágenes implicaría que la memoria se llenara de una cantidad excesiva de archivos.

La implementación del algoritmo en la plataforma se modificó de forma que éste capturara imágenes en intervalos de tiempo definidos para realizar la prueba. Además, se limitaba el número de imágenes que éste podría capturar, en caso la prueba de vuelo se extendiera en tiempo. Otra nueva característica en el algoritmo fue la capacidad de almacenar las imágenes de acuerdo a un contador, de forma que las imágenes adquiridas durante pruebas futuras no sobre escribieran las imágenes adquiridas en pruebas pasadas. Junto a esta característica se añadió que la imagen almacenada en memoria contara además con un sello del tiempo en ésta fue adquirida, permitiendo asociar cada imagen con la prueba de vuelo en que se capturo o proceso.

Por último, fue necesario configurar la plataforma para que esta fuese capaza de ejecutar el algoritmo desarrollado una vez fuese energizado el sistema. Este requerimiento se debió a que estando en el campo para realizar las pruebas de vuelo no se iba a poder contar con una conexión de red para acceder por SSH al BBB y ejecutar de forma manual el algoritmo. Además, siguiendo los objetivos del proyecto, se deseaba que el sistema fuese lo más autónomo posible, por lo cual era aprovechable que el algoritmo implementado se ejecutara de forma automática. La configuración de la plataforma se realizó ingresando para ello los comandos en la consola del BBB:

- sudo chmod +x /home/debian/Prueba/./Nombre_del_Ejecutable
- sudo nano /etc/rc.local
- sudo /home/debian/Prueba/./Nombre_del_Ejecutable &

Tomando en cuenta la interconexión con los demás módulos, se desarrolló un algoritmo alterno con una funcionalidad distinta. En vez de capturar imágenes de forma automática, este algoritmo espera una señal externa en uno de sus puertos de entrada/salida de propósito general (GPIO) y al ocurrir un cambio el algoritmo captura y procesa la imagen. El objetivo de este algoritmo es el de ofrecer un control manual para

la captura de imágenes, siendo una alternativa ideal al momento de hacer pruebas, o si en caso se desean imágenes de un único lugar.

B. RESULTADOS

1. Escena 1

Figura 79. Imagen en bruto capturada por la cámara multiespectral (escena 1)

Figura 80. Imagen pre-procesada con bordes resaltados (escena 1)

Figura 81. Imagen pre-procesada con histograma ecualizado de forma adaptativa (escena 1)

Figura 82. Imagen de punto flotante que indica los valores NDVI de cada objeto en la escena (escena 1)

Figura 83. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo de OpenCV (escena 1)

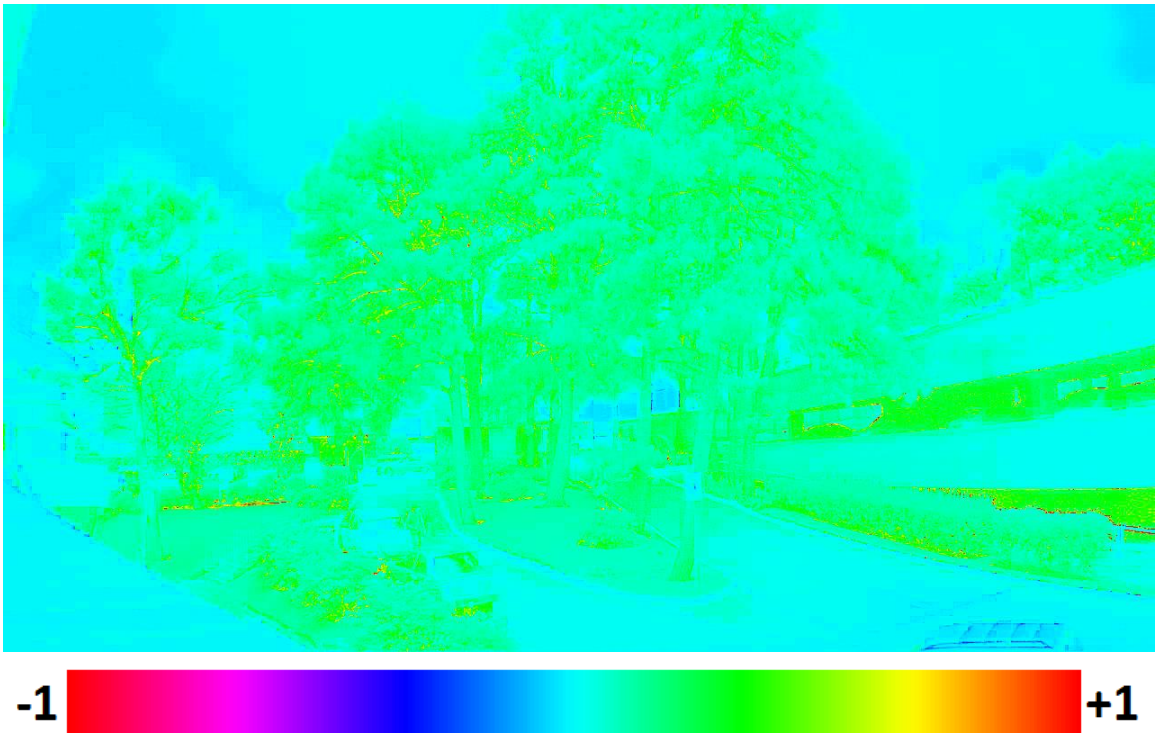
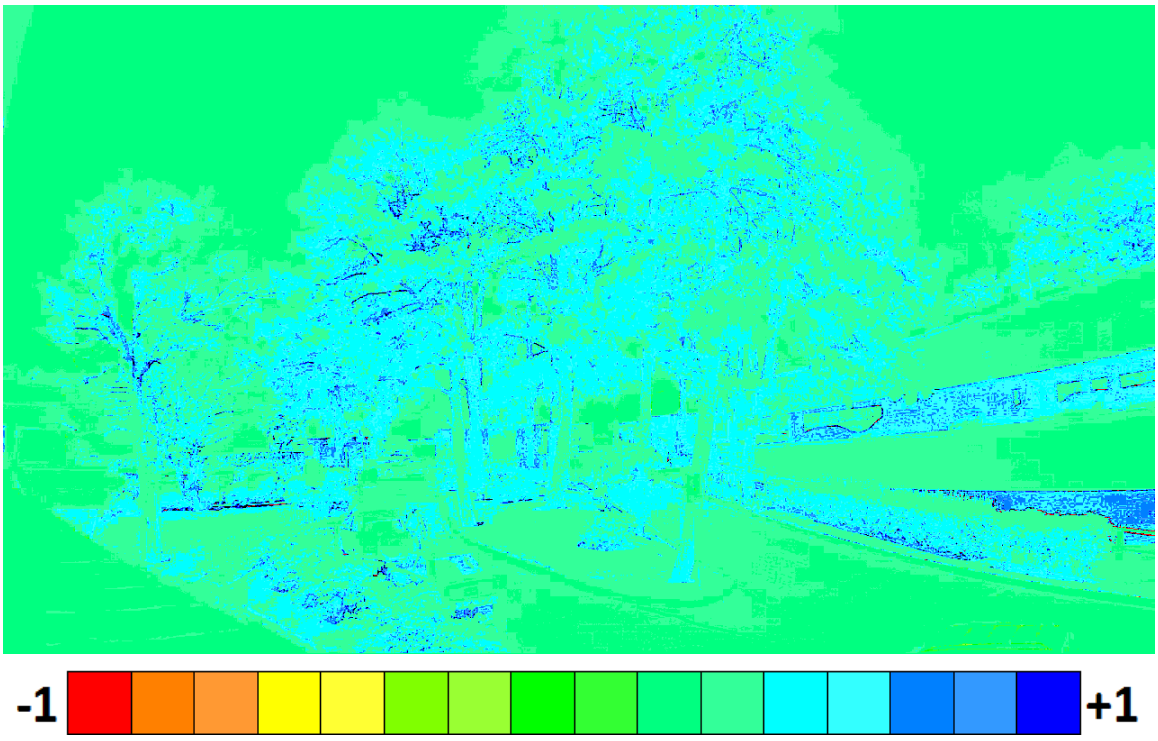


Figura 84. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo propio (escena 1)



2. Escena 2

Figura 85. Imagen en bruto capturada por la cámara multiespectral (escena 2)

Figura 86. Imagen pre-procesada con bordes resaltados (escena 2)

Figura 87. Imagen pre-procesada con histograma ecualizado de forma adaptativa (escena 2)

Figura 88. Imagen de punto flotante que indica los valores NDVI de cada objeto en la escena (escena 2)

Figura 89. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo de OpenCV (escena 2)

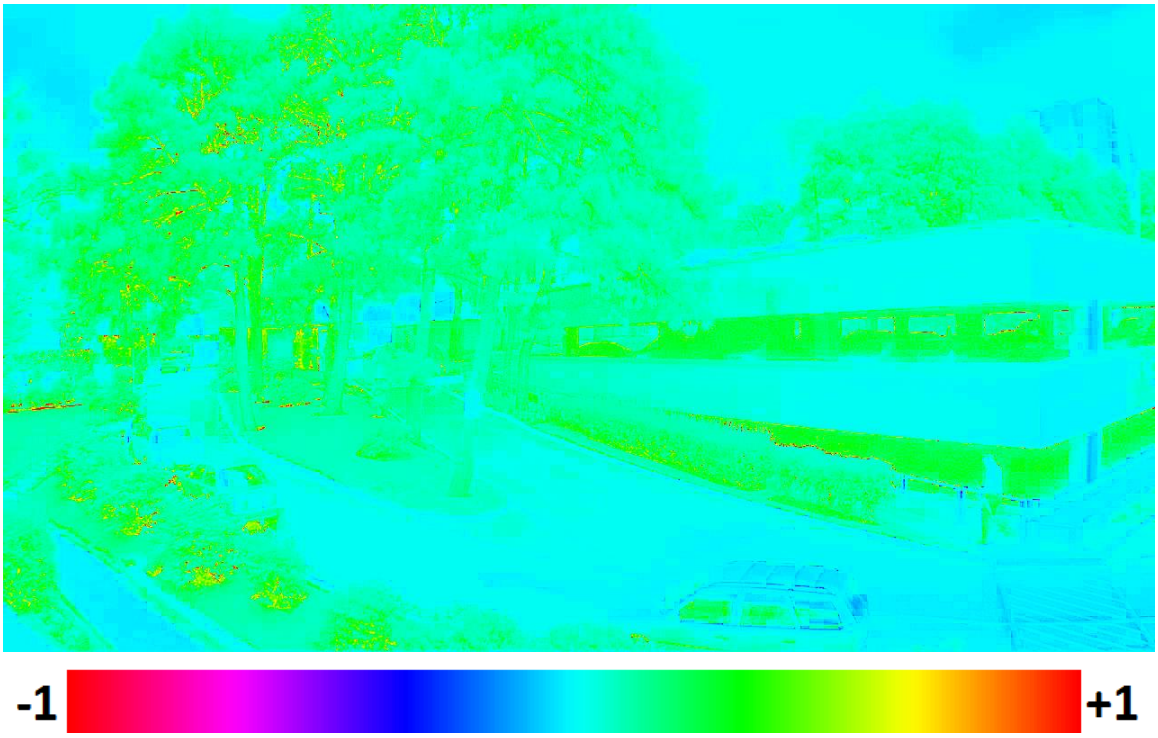
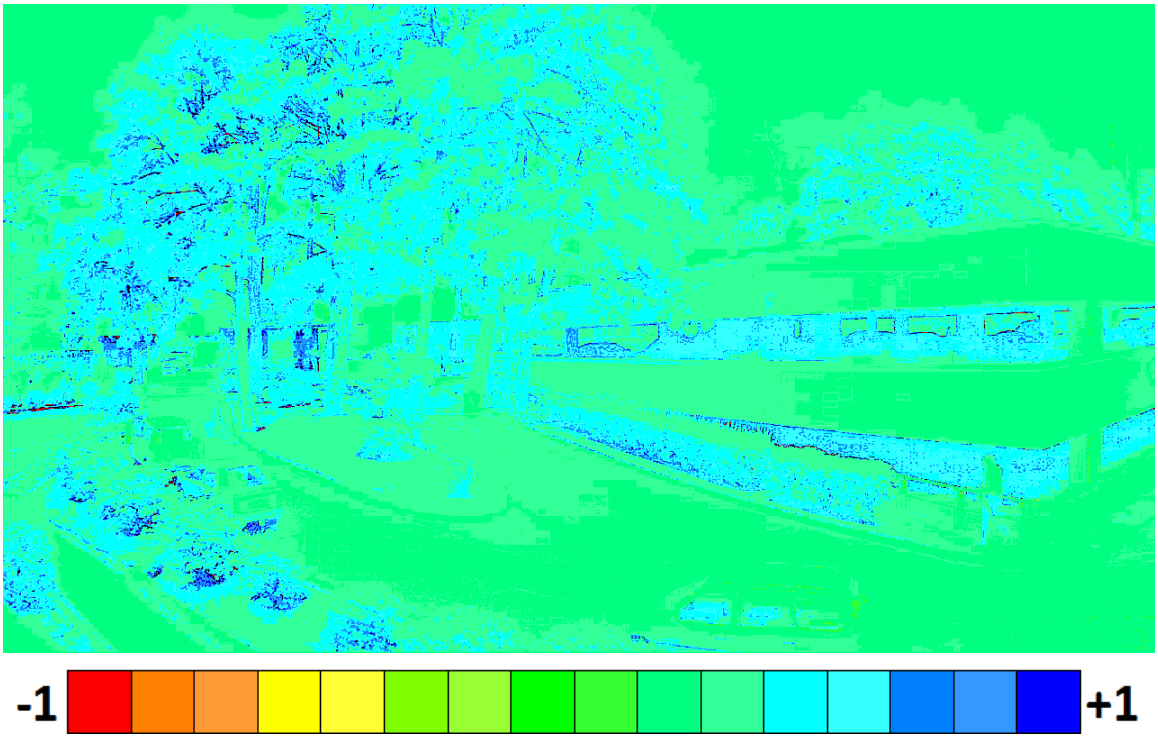


Figura 90. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo propio (escena 2)



3. Escena 3

Figura 91. Imagen en bruto capturada por la cámara multiespectral (escena 3)

Figura 92. Imagen pre-procesada con bordes resaltados (escena 3)

Figura 93. Imagen pre-procesada con histograma ecualizado de forma adaptativa (escena 3)

Figura 94. Imagen de punto flotante que indica los valores NDVI de cada objeto en la escena (escena 3)

Figura 95. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo de OpenCV (escena 3)

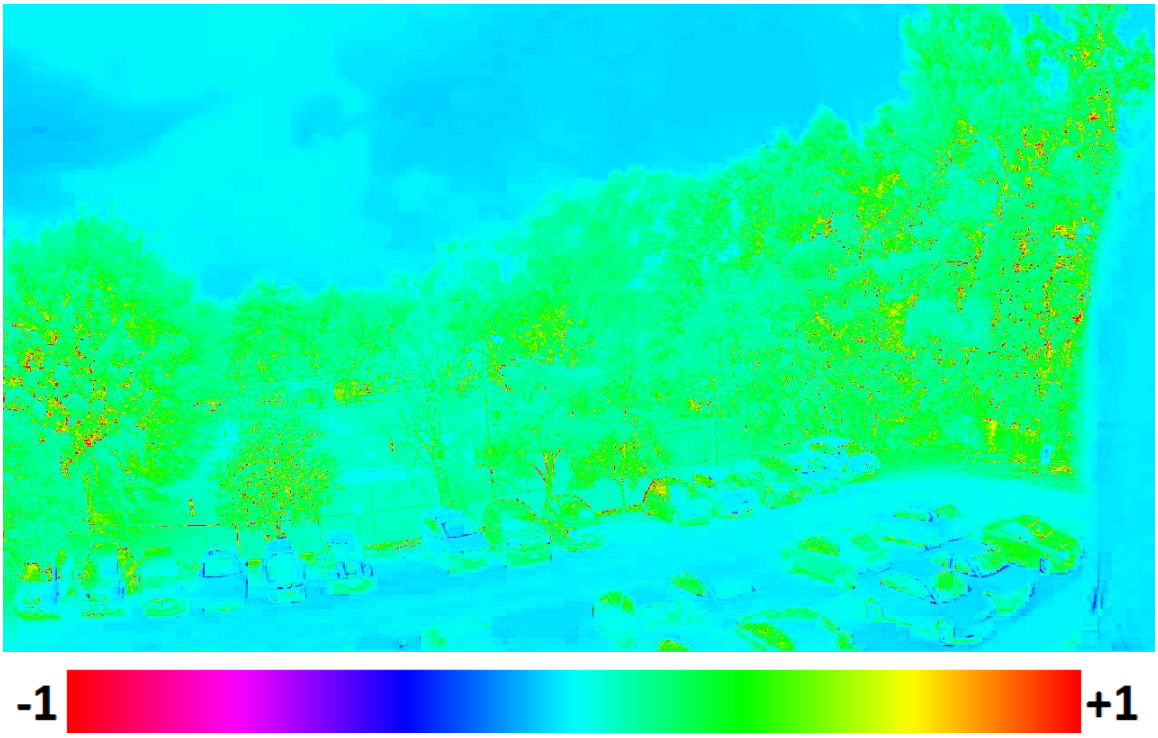
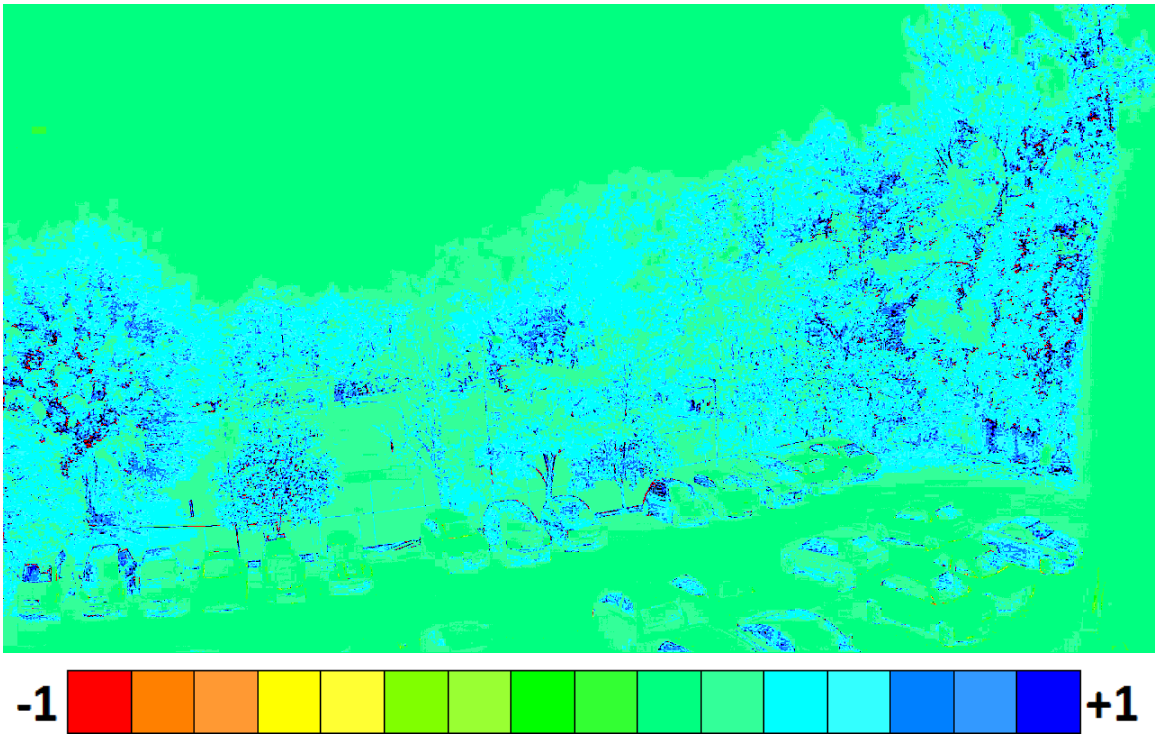


Figura 96. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo propio (escena 3)



4. Escena 4

Figura 97. Imagen en bruto capturada por la cámara multiespectral (escena 4)

Figura 98. Imagen pre-procesada con bordes resaltados (escena 4)

Figura 99. Imagen pre-procesada con histograma ecualizado de forma adaptativa (escena 4)

Figura 100. Imagen de punto flotante que indica los valores NDVI de cada objeto en la escena (escena 4)

Figura 101. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo de OpenCV (escena 4)

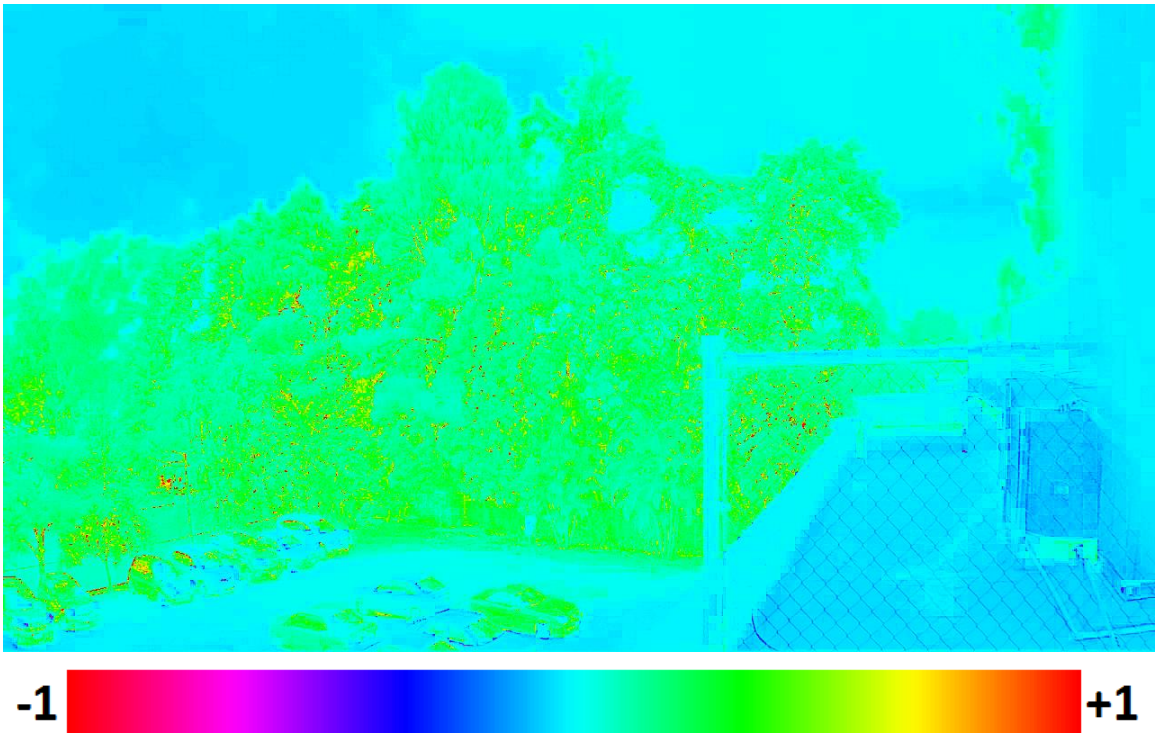
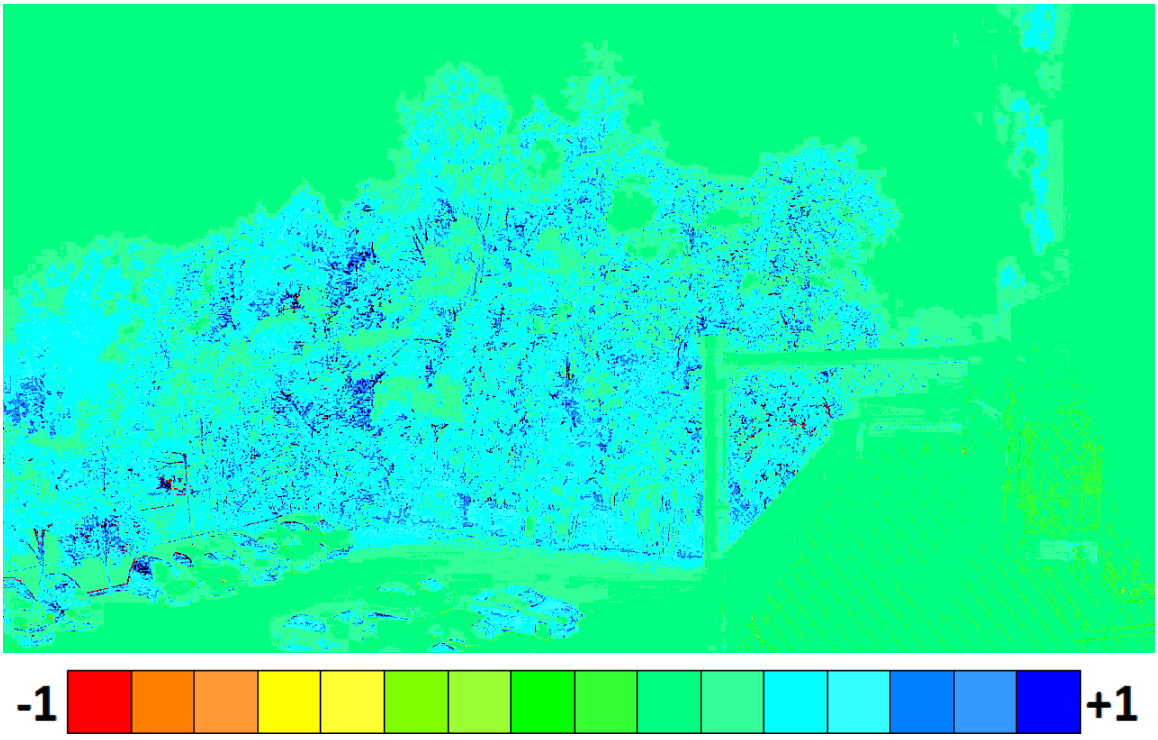


Figura 102. Imagen en color falso que indica los valores NDVI de cada objeto en la escena utilizando el algoritmo propio (escena 4)



C. ANÁLISIS DE RESULTADOS

El objetivo general de este módulo consistió en calcular el índice de vegetación de diferencia normalizado (NDVI) a través del procesamiento de las imágenes multiespectrales capturadas por el vehículo aéreo no tripulado (UAV). Además del objetivo general establecido se planteó una serie de objetivos específicos a alcanzar. El cumplimiento de todos los objetivos específicos suponía a su vez el cumplimiento del objetivo general. Así pues, se estructuró el trabajo correspondiente al módulo enfocado en cumplir cada uno de los objetivos específicos para alcanzar el objetivo general. En la sección de resultados se encuentran cuatro escenas seleccionadas para resaltar los resultados del módulo.

Como primer objetivo específico se tenía capturar imágenes multiespectrales del cultivo a través de una cámara NGR. La cámara multiespectral utilizada se cataloga como una cámara NGR puesto que posee un filtro rojo capaz de captar el espectro infrarrojo a través del canal correspondiente al color azul. En las figura 23, 29, 35 y 41 se puede observar el resultado en bruto de las capturas realizadas con la cámara NGR adquirida. En estas figuras se puede apreciar que toda vegetación, *i.e.* árboles, grama, arbusto, etc., presenta una tonalidad azul. Esto se debe justamente a que la respuesta del espectro NIR se hace presente a través del canal correspondiente al azul. La intensidad de azul observado en estas imágenes corresponde a la intensidad de NIR reflejado por los objetos en la escena. La tonalidad azul también se debe al hecho que los otros dos canales, *i.e.* el rojo y verde, no presentan una respuesta espectral significativa. Esto se debe a que la vegetación absorbe la energía del espectro rojo y verde como parte de su actividad fotosintética para crear alimento. De forma preliminar se puede observar que las imágenes generadas por la cámara NGR se comportan de acuerdo a lo establecido por la teoría, demostrando de forma cualitativa la veracidad de la información presentada por sus imágenes capturadas.

Como segundo objetivo específico se tenía que generar una imagen que mostrara los distintos niveles NDVI de la imagen capturada por el UAV. Para poder aplicar el algoritmo creado a las imágenes capturadas era necesario primero hacerlas pasar por una etapa de pre-procesamiento. En las figuras 23, 29, 35 y 41 se puede apreciar que las imágenes en bruto parecieran estar desenfocadas, sin bordes definidos y demasiado brillantes. Las primeras dos características mencionadas se deben a las limitaciones propias de la cámara NGR adquirida. Por otro lado, la última característica no se deseada se debe principalmente al efecto de los objetos presentes en la escena ajenos a la vegetación, *i.e.* el suelo, las paredes, los carros, entre otros. La aplicación del índice NDVI se ve restringida al uso de áreas en donde un 60% de la escena corresponde a la vegetación. Por debajo de este porcentaje el efecto de la reflexión de otros elementos puede afectar el cálculo del índice NDVI.

Para corregir los primeros dos problemas se optó por aplicar un algoritmo para hacer resaltar los bordes presentes en la imagen. Al tratar con grandes extensiones de cultivos agrícolas este tipo de algoritmo también

resulta útil para detectar los bordes de los terrenos presentes en la escena. El resultado de aplicar este algoritmo se puede apreciar en las figuras 24, 30, 36 y 42. En estas figuras se puede observar que al resaltar los bordes de la imagen ésta se ve mejor delimitada, perdiendo la sensación de desenfoque. El realce de bordes se aplica de forma leve pues un realce demasiado significativo podría alterar el cálculo del índice NDVI, realzando áreas que originalmente no mostraban una alta reflectancia en alguno de los espectros estudiados.

Para corregir la última característica no deseada se optó por aplicar un algoritmo para ecualizar el histograma en la imagen de forma adaptativa. Este algoritmo analiza la imagen en pequeñas regiones, en las cuales calcula el histograma de los canales que conforman a la imagen y los ecualiza para obtener una distribución más uniforme. El análisis en pequeñas regiones permite un resultado que se adopta a las condiciones particulares de cada área, evitando alterar áreas que no necesitan corrección. El resultado de este algoritmo es la mejora del contraste en áreas en las cuales se tiene demasiado brillo como efecto de factores ajenos a las de interés en la escena. El resultado de aplicar este algoritmo se puede apreciar en las figuras 25, 31, 37 y 43. En estas figuras se puede apreciar que la calidad de las imágenes mejora drásticamente. La vegetación ahora se observa en tonos azules mucho más definidos. Además, la distinción entre la vegetación y demás objetos en la escena es ahora más definida. La imagen resultante de esta etapa se encuentra correctamente pre-procesada para poder ser utilizada para el cálculo del índice NDVI.

En las figuras 26, 32, 38 y 44 se puede observar el resultado de aplicar el algoritmo correspondiente al cálculo del índice NDVI para las imágenes previamente pre-procesadas. Estas imágenes son del tipo monocromáticas y el valor de sus píxeles es de punto flotante. En estas imágenes se destaca el hecho que la vegetación se presenta en tonalidades claras, haciendo referencia a índices NDVI altos, en cambio que las regiones oscuras hacen referencia a índices NDVI bajos. Sin embargo, por el tipo de formato en que se encuentran estas imágenes se dificulta observar claramente los gradientes de valores de índice NDVI que presentan los objetos en la imagen, por lo cual es necesario hacer una representación de la imagen en color falso.

Para cada escena presente en la sección de resultados se tienen dos imágenes de color falso. Ambas imágenes representan de forma satisfactoria un gradiente de los valores de índice NDVI presentes en la imagen. Las figuras 27, 33, 39 y 45 son el resultado de aplicar el algoritmo conocido como *ColorMap* de las librerías de OpenCV para generar imágenes en color falso. Las figuras 28, 34, 40 y 46 son el resultado de la implementación del algoritmo creado para generar imágenes en color falso. El algoritmo implementado se personalizó de forma que la imagen resultante utilizara colores similares a los del algoritmo de OpenCV para dar lugar a una comparación más sencilla. Ambos tipos de imágenes en color falso poseen similitudes en la descripción de los niveles NDVI en la imagen, lo cual se puede apreciar a través de los colores que describen a cada tipo de imagen. Ambos tipos de imágenes parecieran responder al mismo rango de valores NDVI de

acuerdo a sus escalas ubicadas en la parte inferior. Sin embargo, se puede apreciar que las imágenes generadas por el algoritmo de OpenCV mantienen el realce en los bordes de los objetos, dando como resultado una imagen más definida, en donde se es fácil distinguir un objeto de otro. En cambio, en la imagen generada por el algoritmo implementado los bordes parecen desaparecer, dando como resultado una imagen poco definida, en donde se es difícil distinguir un objeto de otro. Parece necesario someter a la imagen generada por el algoritmo implementado por una etapa de post-procesamiento con el objetivo de realzar nuevamente los bordes y obtener una imagen mejor definida.

Como tercer objetivo específico se tenía implementar un sistema integrado para realizar el procesamiento de imágenes multiespectrales utilizando la plataforma del BeagleBone Black (BBB). Para ello se desarrolló un algoritmo capaz de realizar cada uno de los objetivos específicos mencionados anteriormente de forma automática al energizar el BeagleBone Black. Este algoritmo se caracteriza por capturar un número limitado de imágenes en un intervalo de tiempo definido. El algoritmo se diseñó de esta manera para que la captura de imágenes no fuese excesiva, evitando sobre-utilizar el espacio disponible en memoria. El algoritmo se caracteriza también por utilizar sellos de tiempo para identificar a las imágenes capturadas. En el ámbito de la Universidad se han realizado algunas pruebas preliminares de esta implementación. Además, se realizó una prueba de vuelo en un cultivo de caña de azúcar en el ingenio San Diego. Dicha prueba de vuelo duró aproximadamente 17 minutos, volando a una altura máxima de 15 metros. A partir de dicha prueba se recolectaron aproximadamente 200 imágenes, siendo estas un resultado más significativo de la implementación del índice NDVI. Sin embargo, haría falta hacer pruebas a lo largo de un período de tiempo, con el fin de dar un mejor uso al análisis cualitativo que brinda el índice NDVI.

Como cuarto objetivo específico se tenía implementar un algoritmo que permitiera la captura de imágenes mediante señales de entrada al BeagleBone Black. Este cuarto y último objetivo específico se encuentra relacionado con la interconexión con otros módulos presentes en el Megaproyecto. Dichas interconexiones no se han podido realizar aún por problemas ajenos a este módulo. Por tanto, aún no se ha definido en su totalidad cómo será la comunicación entre módulos. Sin embargo, considerando la realización de pruebas manuales sencillas, se desarrolló un algoritmo capaz de capturar y procesar la imagen en cuanto este detecta un cambio en uno de sus puertos de propósito general de entrada/salida (GPIO).

XV. MÓDULO DE COMUNICACIONES

A. METODOLOGÍA

La metodología a seguir para la elaboración del módulo de comunicación del Megaproyecto constituye de siete secciones principales. Estas se enuncian a continuación:

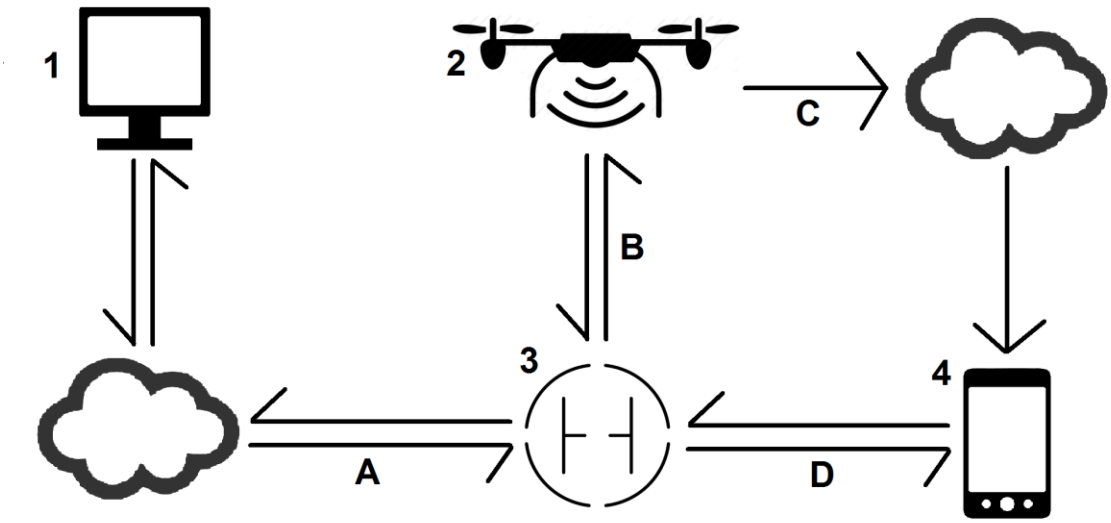
- Encuestas y entrevistas: De manera informal se discutió el proyecto con distintas entidades entre las que se encontraban ingenieros agrónomos con diferentes posiciones dentro de ingenios azucareros y en el centro de investigación de la caña para conocer las necesidades del campo y recibir sugerencias y retroalimentación.
- Diseño general: Según el diseño del proyecto se obtuvieron las funciones principales que se consideraron prioritarias para el módulo de comunicación y se establecieron los objetivos de funcionamiento de las mismas
- Investigación de módulos existentes: Se investigaron las opciones y tipos de dispositivos de comunicación y plataformas de procesamiento existentes en el mercado. A partir del diseño general, se decidió cuáles eran las plataformas necesarias y se determinaron cuáles eran las mejores opciones existentes.
- Selección de módulos comunicación: Con base en la investigación conducida, se procedió a la selección de los dispositivos de comunicación apropiados para el proyecto. Se tomaron en cuenta los aspectos de velocidad, peso, consumo energético, alcance, protocolos de comunicación, precio, disponibilidad en el mercado nacional y su capacidad de ser programados entre otros.
- Selección de módulos de procesamiento: A partir de las pruebas realizadas, se determinaron los requisitos de cada uno de los módulos de procesamiento necesarios. Se procedió a la selección y compra de los mismos en base a los requerimientos del proyecto.
- Prueba de dispositivos de comunicación: Se implementaron los dispositivos seleccionados y se realizaron pruebas para determinar los requerimientos hardware así como las implicaciones del software requerido.
- Diseño e implementación de plataforma de comunicación: Se realizó un diseño en PCB de las plataformas que albergan los dispositivos de comunicación y los módulos de procesamiento de la aeronave, la base en tierra y el sistema de emergencia. A partir de este se programaron todos los módulos de procesamiento para lograr la integración y funcionamiento del sistema y se realizaron las pruebas de funcionamiento respectivas.

B. DISEÑO GENERAL

El módulo de comunicación del proyecto consiste en un sistema de comunicación entre varios nodos que brinde a cada nodo las necesidades que estos requieran. El diseño del mismo contempló varias versiones y modificaciones debido a los cambios del diseño original propuesto en enero del 2015. A partir del diseño del megaproyecto se definieron los objetivos del módulo, así como sus objetivos específicos y posteriormente se definieron objetivos de diseño para cada una de las divisiones del módulo.

El diseño de megaproyecto contempla cuatro nodos principales del sistema: El vehículo aéreo no tripulado o UAV por sus siglas en inglés, la base en tierra para la carga de la aeronave, el usuario del sistema y el ingeniero técnico a cargo del proyecto. De acuerdo al diseño se definieron las siguientes características que cada nodo debía de poseer.

Figura 103. Esquema general del diseño del módulo. 1) Usuario del sistema. 2) Vehículo aéreo no tripulado. 3) Base en tierra. 4) Dispositivo de comunicación del ingeniero o técnico a cargo. A) Conexión a través de XBEE a una frecuencia media de 925MHz a una tasa de baudios de 9600. C) Conexión a través de la red de telefonía utilizando GSM. D) Conexión a través de Bluetooth a una frecuencia de 2.4GHz a una tasa de baudios de 11500.



Para la base en tierra:

La base en tierra debía tener comunicación bidireccional con los otros tres nodos del sistema. Para la comunicación con la aeronave se definió que esta debería de ser por medio de un módulo de radio frecuencia ya que este presenta ventajas en cuanto al tiempo de retraso de los paquetes y la potencia consumida comparado con el uso de la red celular. De igual manera el uso de tecnologías de red de área personal o de área local como lo son Bluetooth y Wi-fi no tienen el rango de comunicación deseado para el proyecto.

En el caso de la comunicación con el usuario se diseñó el sistema para que este creara una red inalámbrica de área personal (WPAN) que le permitiera a través de un dispositivo móvil recibir información detallada de los procesos del sistema para la verificación del correcto funcionamiento del mismo. Se definió que fuera una WPAN por sus ventajas de potencia y facilidad de uso sobre otros medios de comunicación. En el caso fuera necesario, este sistema le permitiría controlar manualmente alguna funciones del mismo he ingresar parámetros de interés.

Con el fin de automatizar el proceso de transferencia de datos al usuario se definió el uso de la red de telefonía móvil a través de un modem inalámbrico. Para ello se requiere un computador que pueda descargar los datos recolectados por la aeronave y enviarlos a través del internet al usuario. Se definió de igual forma que para rebajar el consumo energético, este computador estará encendido de forma intermitente y únicamente cuando sea necesario su uso.

Para la aeronave:

Dadas las sugerencias obtenidas al iniciar el megaproyecto se definió que la aeronave debía de tener comunicación con la base con el fin de compartir información de control y con el ingeniero o técnico a cargo para notificar de fallos en el sistema y para la depuración del mismo. Como se definió en la base en tierra, se utilizó un módulo de radio frecuencia para la comunicación entre ambas.

Cuando se efectuaron las encuestas al principio del proyecto una gran parte de los ingenieros mostro su preocupación por la pérdida de la aeronave en caso esta tuviera un fallo o que fuera derribada con un proyectil. Debido a esto, a la posibilidad de fallos del módulo de radiofrecuencia, a un funcionamiento de la aeronave fuera del alcance de los módulos o de una perdida de potencia fue necesario diseñar un método auxiliar de comunicación. Para ello se definió que esta sería a través de la red de telefonía celular utilizando un módulo GSM. Este método de comunicación debía de ser independiente del funcionamiento de la aeronave y comunicarse solo para obtener información necesaria, si la hubiera, de los motivos de fallo, estado de la batería y ultimas coordenadas recibidas, así como contar con una fuente de alimentación propia.

Para el usuario:

El usuario debe ser capaz de enviar y recibir información desde y hacia la base en tierra. Este permite al usuario enviar rutas al sistema de navegación de la aeronave y descargar las fotos tomadas en cada vuelo. Para poder realizar lo anterior se definió utilizar un repositorio o almacenamiento en línea que permita las funciones anteriores. Una de las ventajas que provee este servicio es la posible implementación de un sistema de actualización de software.

Para el ingeniero a cargo:

Por último en el caso del ingeniero o técnico a cargo se definió que este tuviera acceso al sistema a través de un dispositivo móvil, ya sea a través de una aplicación creada especialmente para este propósito u otra de uso comercial. De igual manera que con el sistema de procesamiento en la base en tierra, el sistema de comunicación debe de ser intermitente y ser activado solo para su uso. En este caso el ingeniero debe de activar el sistema de comunicación para conectarse al sistema completo.

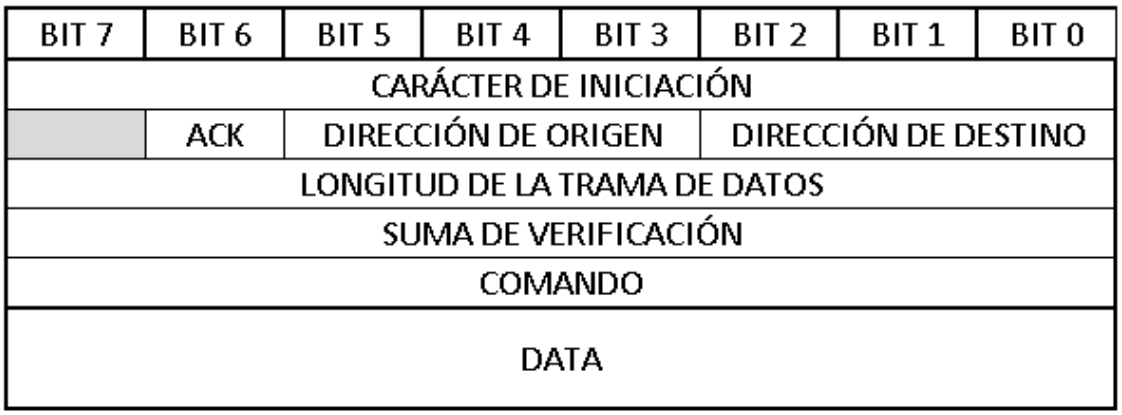
Con base en los criterios anteriores se separó el diseño en tres etapas: el diseño e implementación de las plataformas de comunicación entre la base en tierra y la aeronave, el sistema de emergencia para la aeronave y la base en tierra. A continuación, se presentan los objetivos generales de diseño del módulo.

- Implementar un sistema robusto, programable, expandible y que se pueda configurar con facilidad para los problemas presentados en el módulo.
- Escoger plataformas flexibles que permitan una fácil implantación del sistema.
- Buscar el menor consumo energético en cada uno de los sistemas de forma directa o indirecta.
- Programar el sistema de forma que este pueda ser utilizado y alterado con facilidad.
- Diseñar un sistema de bajo costo con componentes encontrados en el mercado nacional de preferencia.

Como último paso del proceso de diseño se creó un protocolo de comunicación el cual permite el uso del sistema. Se sugiere el uso del protocolo, sin embargo, este puede ser modificado y mejorado con el fin de brindarle un mejor servicio al sistema de navegación. El protocolo fue creado utilizando como base el protocolo de transporte TCP.

El protocolo consiste en un encabezado de cinco bytes de información y una sección de datos para un paquete de hasta 255 bytes. El primer byte del encabezado se utiliza para separar las cadenas de bytes recibidas por el sistema. Se propuso utilizar el carácter ASCII ‘~’ ya que este no es usado en ningún otro momento y es un carácter poco utilizado. El segundo byte consiste de cuatro campos que corresponden a la dirección de destino y de origen del paquete, la bandera de recepción exitosa (ACK abreviado en inglés) y un campo reservado para uso del sistema en caso la suma de verificación coincida con el byte de iniciación. Los últimos tres bytes del encabezado corresponden a los campos de longitud de la trama de datos del paquete, la suma de verificación y el campo de comandos.

Figura 104. Estructura del paquete de comunicación.



C. DISEÑO DE LA PLATAFORMA DE COMUNICACIÓN

La primera parte del módulo de comunicación del megaproyecto es diseñar una plataforma de comunicación que permita la misma entre la base en tierra y la aeronave principalmente. A esto se le agregan funciones de control y de comunicaciones entre otras partes del sistema.

El primer criterio para el diseño de la plataforma de comunicación es que esta debe de ser el centro de coordinación principal entre la base en tierra y la aeronave. Esto implica que ambos elementos deben de poseer una plataforma en ellos que permita la comunicación entre ambas plataformas y entre los componentes de cada uno de los elementos del sistema. La plataforma debe de permitir la programación de diversas funcionalidades entre ellas, la coordinación de despegue y aterrizaje de la aeronave, las funciones básicas de control de los diferentes elementos de la base, la implementación de un protocolo de comunicación y la funcionalidad de conmutar los diversos paquetes de información, así como de traducirlos y trasmitirlos a los diferentes elementos del sistema.

La base para la creación de la plataforma de comunicación es un microcontrolador que le permita realizar todas las funciones necesarias y que le permita comunicarse con todos los dispositivos del sistema. El primer paso para el diseño de la plataforma fue entablar una lista de aquellas funciones que posiblemente pudiera tener que realizar para luego investigar que microcontrolador es el más apropiado para el proyecto. A continuación, se ofrece un listado de las funciones que puede llegar a tener la plataforma en la base en tierra y la plataforma en la aeronave.

En la base en tierra:

- Permitir la comunicación con la plataforma en la aeronave a través de un módulo de radiofrecuencia.

- Permitir la comunicación con el sistema de procesamiento de la base en tierra a través de sus puertos de entrada y salida.
- Activar y desactivar el sistema de procesamiento.
- Activar y desactivación el modo de envió de datos a través de Bluetooth para la depuración del sistema.
- Permitir la comunicación por medio de Bluetooth.
- Permitir la comunicación por medio de USB (esta función no se implementó en el proyecto final).
- Permitir la posibilidad de integrar nuevas funciones de forma modular.
- Controlar el tiempo de activación y desactivación de la aeronave.

En la aeronave:

- Permitir la comunicación con la plataforma en tierra a través de un módulo de radiofrecuencia.
- Permitir la comunicación con el sistema de procesamiento.
- Permitir la comunicación con el sistema de comunicación de emergencia.
- Activar y desactivar el sistema de conmutación del medio de almacenamiento de datos.
- Permitir la comunicación con el sistema de almacenamiento de datos de emergencia
- Permitir la comunicación por medio de USB (esta función no se implementó en el proyecto final).
- Permitir la posibilidad de integrar nuevas funciones de forma modular.

Con base en el listado anterior se definieron diversos parámetros de investigación para el microcontrolador de las plataformas. La principal característica que debían cumplir estos microcontroladores seria la fácil programación de los mismos y la capacidad de implementar el código en otro modelo de microcontrolador de forma sencilla. Esta capacidad de compatibilidad del código se tomó en cuenta previniendo una ampliación de las plataformas de comunicación en un futuro.

Además de sus características de software, el microcontrolador debe de tener módulos periféricos que permitan tener las funcionalidades requeridas por el sistema. Dadas las múltiples comunicaciones que debe entablar el microcontrolador debía de poseer diversos tipos de puertos de comunicación. Debido a que el microcontrolador de la plataforma en base debe de comunicarse de forma serial y asíncrona con el modulo Bluetooth y con el módulo de radiofrecuencia este debe de poseer dos puertos de comunicación UART. De igual manera en la plataforma de la aeronave, el microcontrolador debe de comunicarse de esta forma con la

plataforma de procesamiento y con el módulo de radiofrecuencia. Las razones de esto son expuestas más adelante. Para estos propósitos el microcontrolador debe de cumplir con el requisito de tener al menos dos receptores y dos transmisores de comunicación UART. Estos a su vez debieron de ser configurables a una taza de baudios de hasta 115200 baudios por segundo.

El siguiente requisito que debió poseer el microcontrolador es la capacidad de comunicación con las otras partes del sistema. Se requirió que este tuviera un bus de comunicación I2C que le permitiera comunicarse con el sistema de procesamiento de la base y con el medio de almacenamiento de emergencia en la aeronave. El bus debió de cumplir con el requisito de funcionar con un reloj de hasta 100 kilo-hercios como una medida estándar. Además este debió de poderse configurar como dispositivo esclavo y como dispositivo maestro. El bus I2C también permite la comunicación con otros dispositivos permitiendo así la expansión del módulo y de sus funcionalidades.

Para la activación y desactivación de los sistemas en cada una de las plataformas, el microcontrolador debió de cumplir con el requisito de tener disponibles cierta cantidad de puertos de entrada y de salida así como puertos que le permitan programar interrupciones por cambios en las entradas. Estos puertos debieron tener la capacidad de operar a distintos voltajes ya que diversos módulos trabajan a en un rango de voltajes desde 3.3 hasta 5 voltios.

Por último, se desea que el microcontrolador tenga ciertas características que le permitan tener un mejor desempeño dentro de la plataforma como un alto rendimiento, un bajo consumo energético y distintos periféricos que le permitan realizar sus funciones de forma más eficiente y robusta. En este caso no se tuvo un parámetro específico para escoger el microcontrolador, sin embargo, era deseable que tuviera un oscilador interno con frecuencia de por lo menos de 20 Mega-hercios. Entre los diferentes componentes de hardware adicionales que se quiso buscar se encuentran que el microcontrolador tuviera modos de descanso con bajo consumo energético, hardware especial para operaciones aritméticas, interrupciones con distintos niveles de prioridad, una memoria RAM y una memoria de programa grandes, y un amplio rango de funcionamiento de voltaje y de temperatura.

Un aspecto a tomar en consideración en cuanto a la selección del microcontrolador es su empaquetado. Para un diseño preliminar es deseable un empaquetado que utilice tecnología de aguja pasante (*Through-Hole Technology o THT* por sus siglas en ingles), esto debido a que es posible soldar un porta-integrados e intercambiar el microcontrolador por si esta falla o por si hay que actualizar su programación en el campo. Un diseño posterior podría incluir in microcontrolador montado en superficie, sin embargo la plataforma debe de estar diseñada para poder programar el microcontrolador una vez ensamblado el circuito impreso.

Entre las compañías que diseñan microcontroladores se investigaron los productos de Texas Instruments, NXP, STMicroelectronics, Atmel y Microchip. De todos los proveedores las mejores opciones se encontraron en Microchip dadas las características deseadas del microcontrolador y la opción de tener integrados con

tecnología THT. Los microcontroladores Microchip han sido utilizados anteriormente en otros proyectos y proveen una arquitectura y entornos de desarrollo (IDEs por sus siglas en inglés) familiares.

Dentro de las familias de microcontroladores de Microchip se encuentran principalmente tres, divididas por el número de bits del procesador y estas a su vez se dividen en familias. Las dos familias que se adecuan más a las necesidades del proyecto son la familia de PIC18 con un procesamiento de 8-bits y la familia 24F con un procesamiento de 16-bits. De estas dos se estudiaron a los microcontroladores PIC18F24J50 y PIC24FJ64GB202 como posibles opciones para el proyecto. En ambos casos se tomaron en consideración todos los parámetros necesarios para el proyecto y se concluyó con la decisión de utilizar el microcontrolador PIC18F20J50 como la mejor opción para el proyecto.

Una vez seleccionado el microcontrolador de las plataformas de comunicación se procedió a investigar y a seleccionar cada uno de los módulos y mecanismos adicionales. El primero de los módulos seleccionados fue el módulo de radiofrecuencia. Este módulo es el centro de las comunicaciones entre la base en tierra y la aeronave y por ende es necesario que tenga un alto desempeño. Entre las características buscadas para el modulo se encuentran un amplio alcance, un bajo consumo energético, una alta tasa de transferencia y ser configurable. Los módulos Xbee manufacturados por la empresa DIGI fueron la principal opción para el proyecto. Estos proveen una amplia gama de opciones así como la capacidad de intercambiar diferentes modelos sin realizar cambio de software o hardware.

De igual manera que con el microcontrolador, el primer pasó fue definir los requisitos para el módulo de comunicación. Su principal funcionalidad es la transmisión de datos de la base en tierra a la aeronave de forma serial y asíncrona. La velocidad de transmisión de datos para la primera fase del proyecto no es prioritaria así que se buscó un balance entre la velocidad de transferencia y el alcance. El consumo del módulo también es prioritario. Los módulos de comunicación de diversos tipos por lo general requieren grandes cantidades de potencia durante la transmisión y recepción de datos. Al estar las plataformas sujetas a trabajar con batería y bajo restricciones de peso, este parámetro fue muy importante para la selección del módulo y una de las razones por las cuales se optó por utilizar módulos de radio frecuencia Xbee. Por último la comunicación debe de tener un alcance similar a aquel del alcance de la aeronave. El sistema se diseñó para que la información más importante que es transmitida y recibida por este módulo se transmite cuando la aeronave se encuentra a pocos metros de la base en tierra.

Con estas consideraciones en mente se llegó a dos modelos apropiados para las plataformas de comunicación, el XBEE-PRO 900HP y el XBEE-PRO 868. Debido a que el alcance de la aeronave no fue un valor disponible se escogió el módulo XBEE-PRO 900HP como el más indicado para esta fase del proyecto. Este consume una menor potencia y tiene un costo inferior a costa de tener un alcance inferior. Una consideración importante es la frecuencia de uso del mismo ya que está sujeta a la legislación del país en donde se utilizará. Es por esta razón que se aseguró que el módulo escogido fuera compatible con frecuencias de uso público.

El siguiente módulo a seleccionar fue el modulo Bluetooth. Para este módulo la limitante fue el mercado nacional en el cual solo se distribuyen los modelos HC-05 y HC-06. Ya que el modelo HC-05 cumple con todos los requisitos necesarios y es más económico se procedió a la selección del mismo.

A partir de la selección de módulos se procedió a la configuración y prueba de los mismos. En el caso del microcontrolador se configuró su oscilador interno a una frecuencia de 48 Mega-hertzios utilizando el PLL interno. Se configuraron además los diferentes puertos y periféricos con sus interrupciones definidas. En los Cuadro 1y en el Cuadro 2 se muestran algunas configuraciones implementadas en los microcontroladores.

Cuadro 1. Configuración del microcontrolador en la plataforma de la base en tierra.

Función o Modulo	Pin	Prioridad de interrupción	Configuración
UART TX1	24	Baja	Tasa de transferencia: 115200, sin paridad
UART RX1	23	Baja	
UART TX2	2	Alta	Tasa de transferencia: 9600, Sin paridad
UART RX2	3	Alta	
I2C SDA	26	Alta	Frecuencia de reloj: 100 kHz, dispositivo esclavo.
I2C SCL	25	Alta	
Entrada con Interrupción	21, 27	Baja	Interrupción en flanco positivo
Detección de voltaje alto/bajo	7	Baja	Entrada de comparación externa.
Propósito general	4, 5, 9, 10, 11, 12, 13, 22, 28,	Ninguna	Dependiendo de la aplicación varia su configuración como entrada o como salida

Cuadro 2. Configuración del microcontrolador en la plataforma de la aeronave.

Función o Modulo	Pin	Prioridad de interrupción	Configuración
UART TX1	24	Alta	Tasa de transferencia: 115200, sin paridad
UART RX1	23	Alta	
UART TX2	2	Alta	Tasa de transferencia: 9600, Sin paridad
UART RX2	3	Alta	
I2C SDA	26	Baja	Frecuencia de reloj: 100 kHz, dispositivo maestro.
I2C SCL	25	Baja	
Entrada con Interrupción	27, 28	Baja	Interrupción en flanco positivo
Detección de voltaje alto/bajo	7	Alta	Entrada de comparación externa.
Propósito general	4, 5, 9, 10, 11, 12, 13, 21, 22,	Ninguna	Dependiendo de la aplicación varia su configuración como entrada o como salida

El software de programación utilizado para los microcontroladores fue mikroC Pro con el fin de obtener software de alto de nivel que permitiera el cambio del microcontrolador sin mayores dificultades. Aunque se utilizó un software que incluye librerías para la configuración y uso de los diferentes periféricos del microcontrolador, estas no se utilizaron para tener control completo sobre las variables del sistema. En algunas ocasiones, con el fin de optimizar el código, se programó utilizando lenguaje ensamblador dentro del lenguaje de programación de microC.

En el caso de los módulos Xbee la configuración fue mucho más sencilla. Los módulos Xbee poseen principalmente tres modos de funcionamiento, el modo API, el modo transparente y el modo de configuración por comandos AT. Dada la naturaleza de la aplicación el modo API resulta ser ineficiente ya que no necesitamos utilizar muchas de las funciones que este modo ofrece y resulta ser ineficiente par nuestro propósito. En el modo transparente, el sistema solo transfiere los datos del puerto serial de un lado al otro de manera sencilla y concisa y permite el ingreso al modo de configuración por comandos AT. Los módulos se programaron para utilizar este modo a una tasa de transferencia de 9600 baudios utilizando el software de programación XCTU. Cabe resaltar que se escogió una tasa de transferencia baja para que fuera compatible con todas las plataformas Xbee además de presentar ventajas en cuanto a un mayor alcance (según el

fabricante) y una menor probabilidad de error. En el Cuadro 3 y en el Cuadro 4 se muestran las configuraciones modificadas en cada módulo.

Cuadro 3. Configuración de módulo Xbee para la base en tierra.

Parámetro	Valor
Function set:	ZIGBEE COORDINATOR AT
Operating PAN ID:	2015
Channel Verification:	Enabled
Destination Address High:	0
Destination Address Low:	FFFF

Cuadro 4. Configuración de módulo Xbee para la aeronave

Parámetro	Valor
Function set:	ZIGBEE ROUTER AT
Operating PAN ID:	2015
Channel Verification:	Enabled
Destination Address High:	0
Destination Address Low:	0

El módulo de Bluetooth HC-05 tiene dos versiones que difieren en su capacidad de ser programados por el usuario. En su versión programable, el modulo es muy versátil y provee de una interfaz sencilla con el usuario. Para su configuración la entrada KEY debe de tener voltaje de 5 voltios y luego se debe proceder a encender el módulo el cual puede ser configurado a través de códigos AT. El Cuadro 5 muestra los parámetros configurados para este módulo.

Cuadro 5. Configuración de módulo Bluetooth

Parámetro	Valor
NAME:	BASE_CONTROL
PSWD (contraseña):	2015
ROLE:	0-Slave
IAC:	9e8b33
UART:	115200, 1, 0
CMODE:	1-Connect any address
IPSCAN:	1024, 512, 1024, 512
SNIFF:	0, 0, 0, 0

Una vez configurados los módulos se verificó su funcionamiento utilizando como interface un microcontrolador Arduino por su facilidad de uso. Para hacer la conexión con un dispositivo móvil a través de Bluetooth se utilizó un teléfono celular LG G2 y la aplicación *Bluetooth Terminal* que permite recibir y enviar datos en un sistema operativo Android.

Por último se diseñaron las otras funciones de las plataformas de comunicación no discutidas anteriormente las cuales son:

Encendido y apagado de sistema de procesamiento de la base en tierra a través de un opto-relé.

- Activación y desactivación del módulo Bluetooth a través de un interruptor de lengüeta.
- Detección de conexión USB si el modulo llegara a utilizarse.
- Detección de voltaje de alimentación bajo.
- Puerto de programación para el microcontrolador.
- Botón e indicador programable.
- Botón de reinicio de microcontrolador.
- Regulador de voltaje para alimentación de 5 voltios.
- Interruptor de encendido/apagado.

- Conexiones configurables de entrada y salida a puertos del módulo Xbee y a puertos del sistema de procesamiento de la base en tierra.
- Bus de comunicación I2C con el sistema de procesamiento de la base en tierra.
- Interconexión entre módulos de comunicación y el microcontrolador.
- Bus de comunicación I2C con el sistema de comunicación de emergencia.

D. DISEÑO DEL SISTEMA DE COMNICACIÓN DE EMERGENCIA

El sistema de comunicación de emergencia fue incluido en el diseño bajo la retroalimentación obtenida de las encuestas realizadas a distintas personas en el campo. Su principal preocupación era el robo de la aeronave, aunque luego se tomaron en cuenta otros factores para el diseño.

Se determinó en el diseño general del módulo que se debía utilizar un módulo GSM como medio de comunicación para el sistema de emergencia que cumpliera los requisitos en cuanto a bajo costo, bajo peso, bajo consumo energético, facilidad de uso, soporte técnico y compatibilidad con la red de telefonía móvil en Guatemala. En este caso las opciones en el mercado fueron escasas resaltando de entre ellas un módulo GSM desarrollado por la compañía Adafruit el cual cumplió con todos los requisitos del módulo.

El módulo Adafruit Fona cuenta con un módulo GSM SIM800L el cual le permite tener envío y recepción de llamadas y de mensajes de texto así como transmisión de datos por GPRS, sintonización de radio FM, triangulación de posición y control de dispositivos a través de PWM. El modulo se comunica a través de UART y tiene una interfaz de comandos AT. Este además cuenta con dos versiones que difieren por el tipo de conector de la antena.

Dada la decisión de utilizar el módulo Adafruit Fona para el módulo de emergencia se procedió a comprar la tarjeta SIM, la antena y la batería necesarias para su funcionamiento. Adafruit recomienda utilizar una batería de capacidad de carga mayor o igual a 500 mili-amperios-hora. Se adquirió entonces una batería de polímeros de litio de 3.7 voltios con una capacidad de carga de 500 mili-amperios-hora que le permite al módulo operar más de 24 horas en hibernación. También se adquirió una antena tipo pegatina con conector uFL con ganancia de 3 dBi. Es importante mencionar que la batería recomendada por la empresa Adafruit para el módulo no posee un conector compatible con el módulo y por ende fue necesario cambiarle el conector.

Para continuar con las pruebas se obtuvieron dos tarjetas SIM, una de la compañía Tigo y una de la compañía Claro. Para realizar las pruebas se interconectaron todos los dispositivos con un Arduino Uno que se utilizó para enviar los códigos AT correspondientes. Se probaron todos los códigos AT pertinentes al

proyecto para verificar su funcionamiento y obtener una lista de las respuestas recibidas y del formato de estas para posteriormente programar el sistema en base a ellas.

A partir de las pruebas del módulo Adafruit Fona de prosiguió a determinar el microcontrolador a utilizar en este sistema. Dada la elección de microcontroladores de la marca Microchip para la plataforma en tierra se decidió utilizar un microcontrolador de la misma marca para este sistema con el objetivo de simplificar la programación del mismo. Se determinó que el microcontrolador debía de poseer una alta velocidad de reloj para poder procesar una gran cantidad de datos recibida del módulo GSM además de un puerto de comunicación UART y un puerto de comunicación I2C. Además de esto se determinó utilizar un microcontrolador de la familia PIC16 ya que estos tienen un desempeño más apropiado además de un menor costo. A partir de estos parámetros y de los objetivos de diseño general se determinó que el microcontrolador más apropiado seria el PIC16F1619.

Además del módulo GSM, el sistema cuenta con una memoria EEPROM a la cual tiene acceso el microprocesador del sistema de emergencia y el microprocesador de la plataforma de comunicación. Este diseño se realizó con el objetivo de tener un base de información con parámetros del sistema aéreo sin necesidad de activar el microcontrolador del sistema de emergencia y en el caso de haber un fallo, la información podría ser obtenida por este y ser enviada a través del modulo GSM.

E. DISEÑO DEL SISTEMA DE BASE EN TIERRA

El sistema de base en tierra comprende la interconexión de la plataforma de comunicación con el sistema de comunicación con el usuario y otros elementos del sistema.

El primer paso para su diseño fue encontrar un sistema de cómputo que fuera compatible con un modem inalámbrico y que cumpliera con los objetivos generales de diseño.

Se propusieron las plataformas Raspberry Pi y BeagleBone Black como posibles computadoras y se realizó un análisis de cada una de ellas. Utilizando el análisis de desempeño propuesto por Adafruit con el nombre de *Embedded Linux Board Comparisson* se determinó que el computador más apropiado para esta aplicación era la Raspberry Pi A+ cuyas características le permiten tener el consumo más bajo de potencia promedio sin sacrificar su desempeño general. Entre sus limitaciones comparadas con otros modelos de Raspberry Pi se encuentran la falta de puerto Ethernet y solo tener un puerto de comunicación USB.

A partir de esta decisión se prosiguió a instalar el sistema operativo Raspbian Whezzy descargado el 16 de febrero del 2015. Se decidió utilizar este sistema operativo debido a su amplio soporte técnico y su facilidad de uso.

Se determinaron necesarias las siguientes funciones del sistema:

- Inicio automático sesión (sin usuario ni contraseña).
- Ejecución automática del programa de control al terminar el inicio de sesión.

- Transmisión y recepción de datos a través internet.
- Lectura y escritura de memoria de la aeronave.
- Actualización automática de software.
- Configuración de modem inalámbrico.

Se inició configurando el inicio de sesión automático del sistema. Para configurar esta función se modificó el archivo inittab del directorio /etc de modo que al encender la plataforma no se requiera una contraseña o un usuario para correr un script.

La ejecución de un script en Raspian se logra al incluirlo en el archivo rc.local en el directorio /etc. Para que el script corra debe de ser ejecutable y tener los derechos de autor requeridos. En este proyecto el script ejecuta primero el programa control.py que contiene las funciones que debe cumplir la computadora y luego ejecuta el programa update.py que actualiza el archivo control.py en caso encuentre una versión nueva del mismo.

Se investigaron diferentes métodos para la función de trasmisión y recepción de datos a través de internet. Una de las aplicaciones populares en Raspbian es el uso del servicio de alojamiento de archivo Dropbox el cual provee esta función. Debido a la popularidad del sistema, su facilidad de uso y su diversidad de aplicaciones y plataformas se decidió emplear esta plataforma para sincronizar los datos del usuario con los datos de la base en tierra y de la aeronave. Se utilizó el programa Dropbox-Uploader el cual es compatible con la API de Dropbox. Este programa permite la carga y descarga de archivos de una cuenta de Dropbox además otras funciones de poco interés para el proyecto. En el Cuadro 6 se muestra la configuración utilizada para la creación de la aplicación de Dropbox.

Cuadro 6. Parámetros de configuración de aplicación de Dropbox

Parámetro	Valor
Correo electrónico:	megaproyecto.cropcopter@gmail.com
Contraseña del correo electrónico:	cropcopter
Contraseña de la cuenta en Dropbox:	cropcopter
Dropbox app name:	Megaproyecto_cropcopter
App key:	vgi8max9h95pgin
App secret:	r1pmp1r8hz69t7i
Permission:	Full Dropbox

Para poder transmitir datos desde la Raspberry Pi al usuario, la computadora debe primero poder acceder a ellos a través de un canal de comunicación. Se decidió que la manera más sencilla de realizar esta transacción de datos era a través de un medio de almacenamiento que permitiera su comunicación con la aeronave mientras esta estuviera en vuelo y con la base en tierra al momento de ser sincronizada. Como primera opción se propuso una memoria microSD por su tamaño compacto y su capacidad variable sin embargo el sistema de lectura de datos representa una dificultad y se necesitan tres líneas de datos para su lectura y escritura. Para solucionar este problema se procedió al uso de un adaptador de memoria microSD a USB que permite la fácil conexión con ambos sistemas y que solo necesita dos líneas de datos para su lectura y escritura.

El siguiente paso para la comunicación automática con la memoria es montar la memoria USB en la Raspberry Pi. El sistema operativo automáticamente le asigna un directorio a la memoria USB para que el usuario pueda acceder a la información dentro de la memoria a través de este directorio. Este sistema funciona si se tiene solo un sistema de almacenamiento conectado, sin embargo los módems inalámbricos y otros dispositivos USB pueden llevar a montarse en diferentes directorios cada vez que se enciende la Raspberry Pi. Para resolver este problema se configuró el sistema operativo utilizando el UUID de la memoria a utilizarse (UUID: 380B-E41A). El UUID es una referencia única de la memoria que le permite diferenciarse entre otros dispositivos USB. En base a esta referencia se montó la memoria a un directorio establecido para eliminar la posibilidad de que el sistema intercambie directorios.

Por último se configuró el modem inalámbrico en el sistema operativo. Raspbian no reconoce a los módems inalámbricos como tales, en cambio los reconoce únicamente como dispositivos de almacenamiento masivo. Para solucionar este problema el modem debe de ser compatible con Raspbian. Se probó la compatibilidad de los módems Huawei E353, Huawei E153 y Huawei E173 siendo solo compatible este último. A diferencia de las configuraciones anteriores, existe poca documentación acerca del proceso de configuración de un modem inalámbrico en Raspbian. Para solucionar este problema se utilizaron tres programas distintos, usbswitch-mode, wvdial y sakis3g. A partir de estos tres se pudo entablar una conexión de internet exitoso por medio del modem inalámbrico.

Una vez configurado el sistema operativo se procedió a la programación de las funciones del sistema en un script utilizando el lenguaje de programación Python. En la se observa el algoritmo de programación que se siguió para implementar el control de sincronización de datos mientras que en la se muestra el algoritmo de actualización de software.

Figura 105. Algoritmo de control de sincronización de datos.

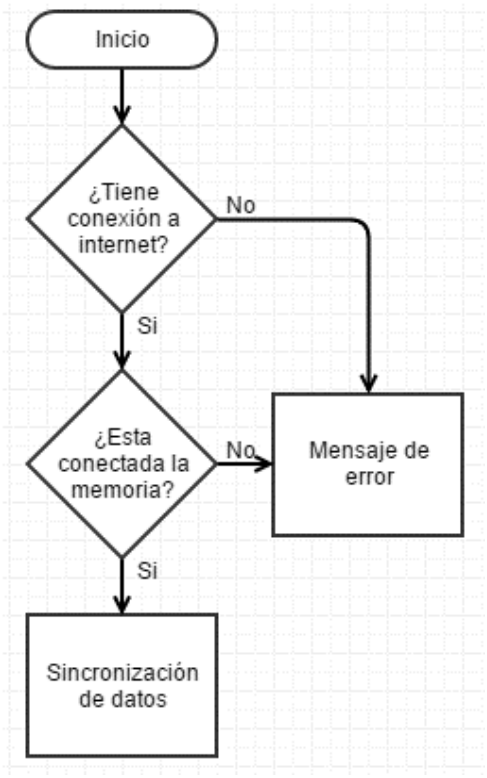
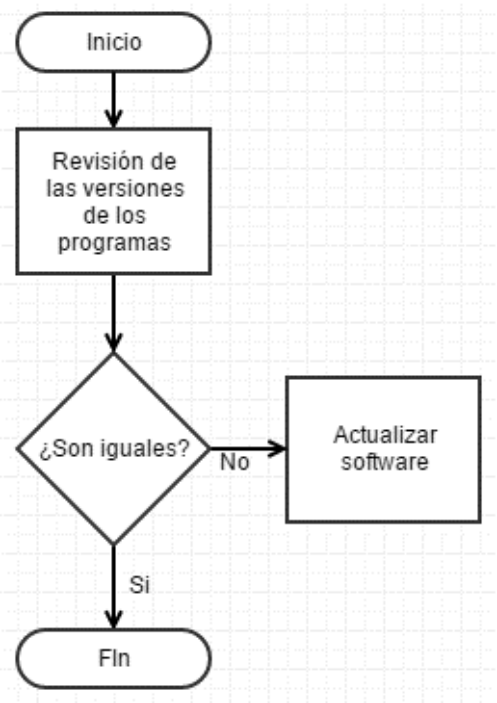


Figura 106. Algoritmo del programa de actualización de software.



Una vez configurada y programada la computadora de la base en tierra se prosiguió al diseño de las demás funciones de la misma. Para que la conexión entre la memoria USB y los sistemas de procesamiento sea posible es necesario realizar un de multiplexor/de-multiplexor para la memoria. Esto se debe a que ambos sistemas de procesamiento tienen el rol de dispositivos maestros y el estándar USB no tiene una funcionalidad multi-maestro. Debido a que los buses de datos del estándar USB son de tipo diferencial y bidireccional, para implementar un dispositivo que seleccione a que dispositivo maestro está conectada la memoria es necesario utilizar opto-relés ya que estos son bidireccionales y no presentan la desventaja de necesitar voltajes mayores como un interruptor analógico.

Además de esta funcionalidad se construyó la base en tierra con buses de comunicación de tal modo que pueda trasmitir y recibir datos cuando los contactos de las patas de la aeronave hagan contacto con los contactos de los buses de la base en tierra. Adicionalmente se construyó una cerradura especial para una llave que permita activar el interruptor de lengüeta y acceder a la WPAN por medio de Bluetooth.

F. RESULTADOS

1. Resultado del diseño de la plataforma de comunicación de la base en tierra

Figura 107. Diseño de la plataforma de comunicación de la base en tierra.

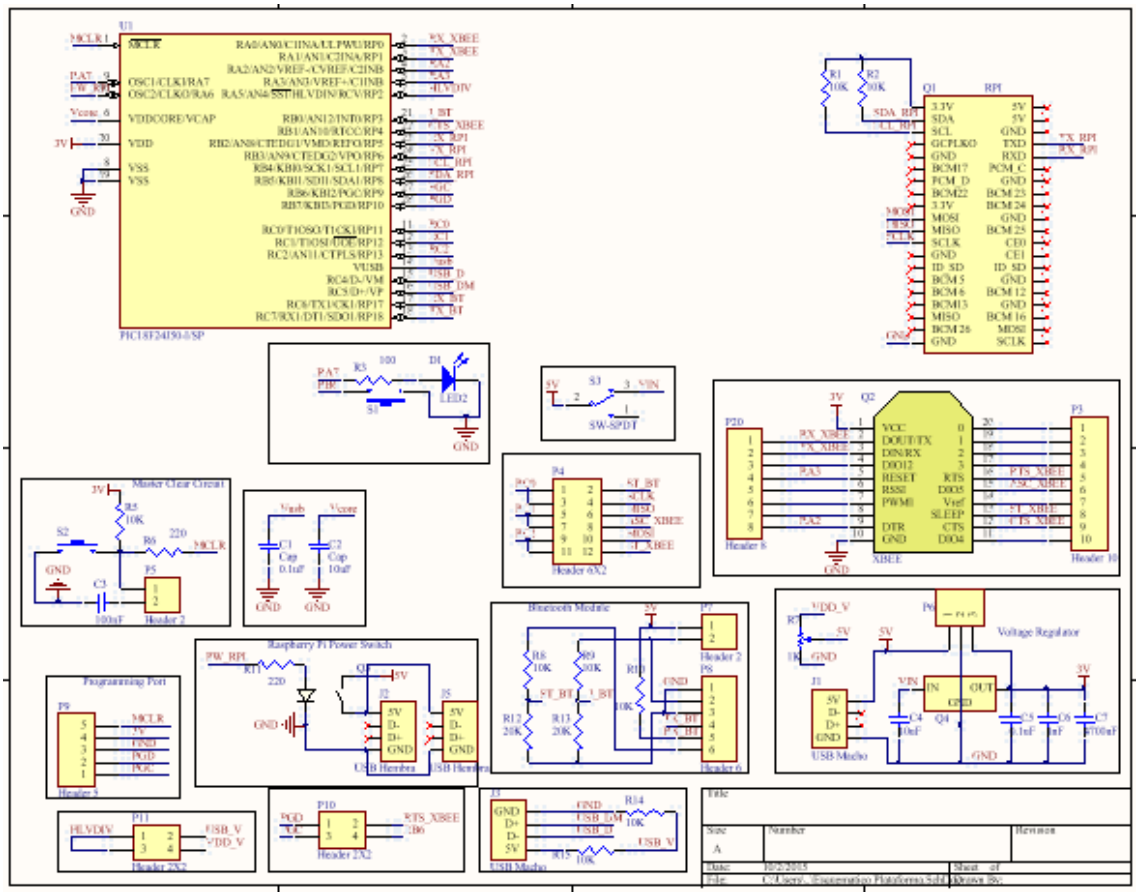


Figura 108. Conexiones del módulo Xbee.

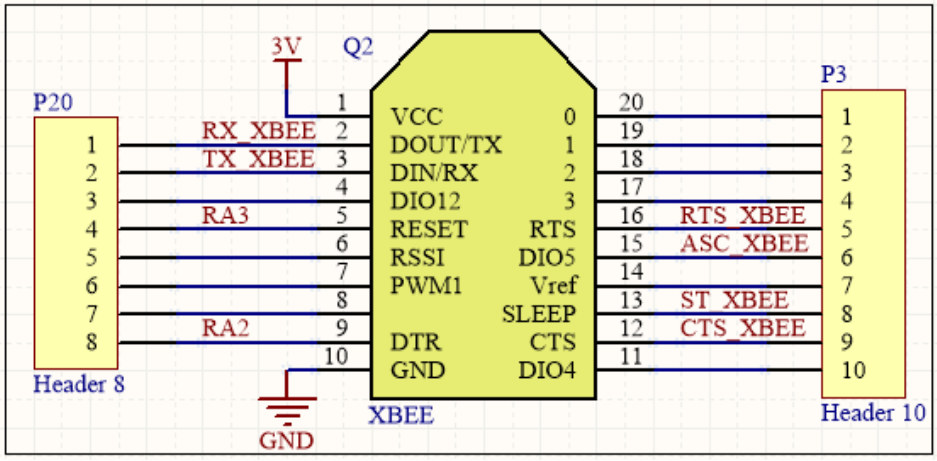


Figura 109. Regulador de voltaje para la plataforma de comunicación. P6 permite obtener voltajes.

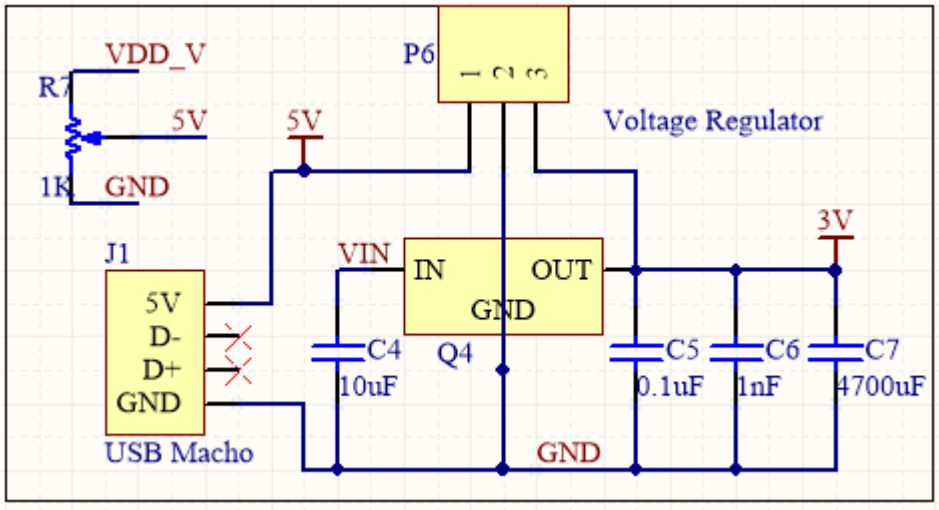


Figura 110. Conector para modulo Bluetooth. P7 es el conector para el interruptor de lengüeta.

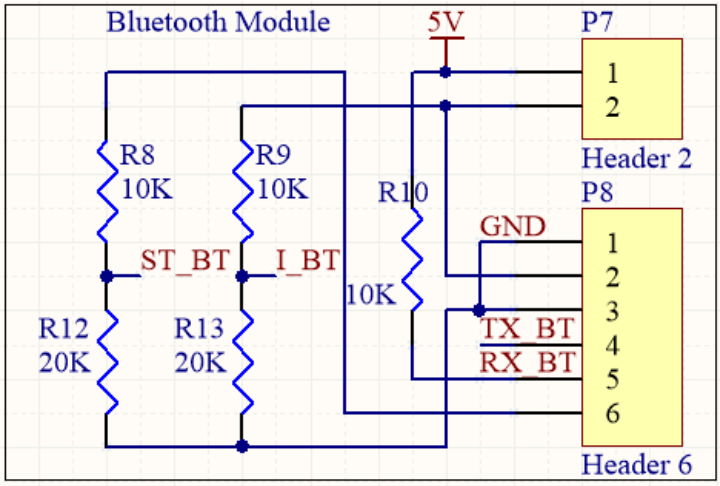


Figura 111. Conector USB del microcontrolador.

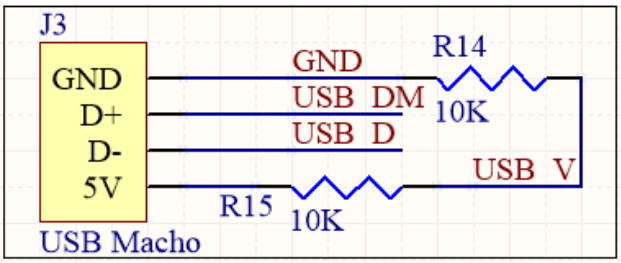


Figura 112. Pines de selección.

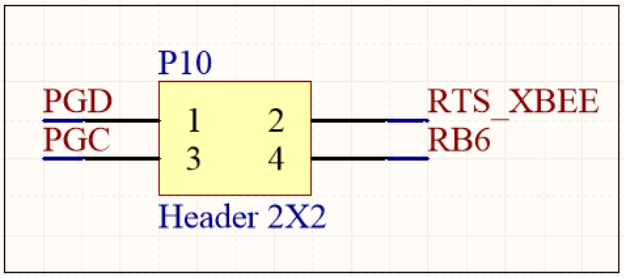


Figura 113. Pines de selección de entrada para la detección de voltajes altos o bajos.

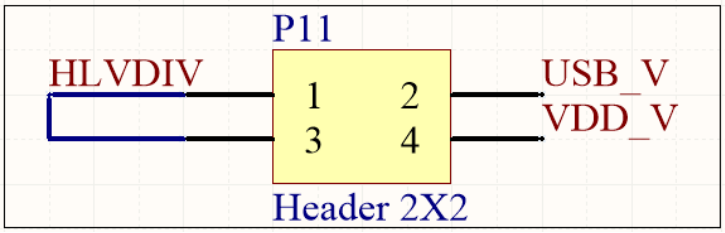


Figura 114. Puerto de conexión para un programador Pickit 3.

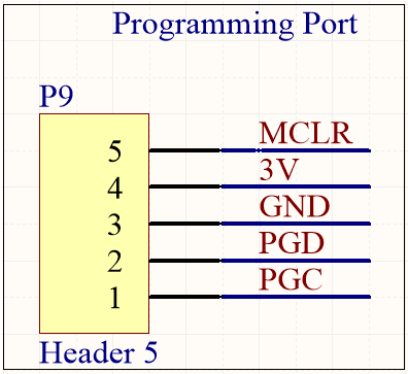


Figura 115. Control de encendido/apagado de la Raspberry Pi.

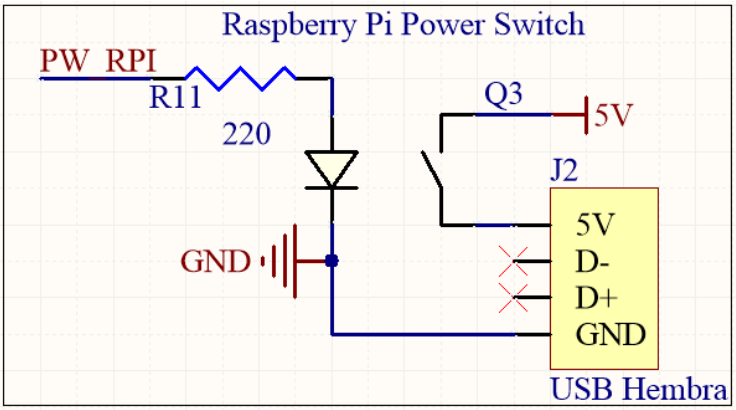


Figura 116. Capacitores de estabilización para el los núcleos del microcontrolador.

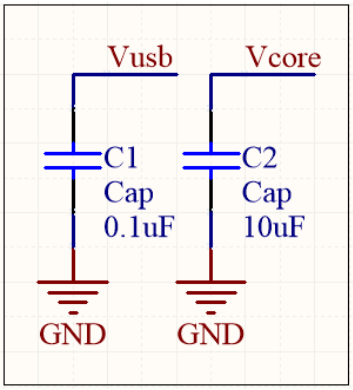


Figura 117. Circuito de botón de reinicio.

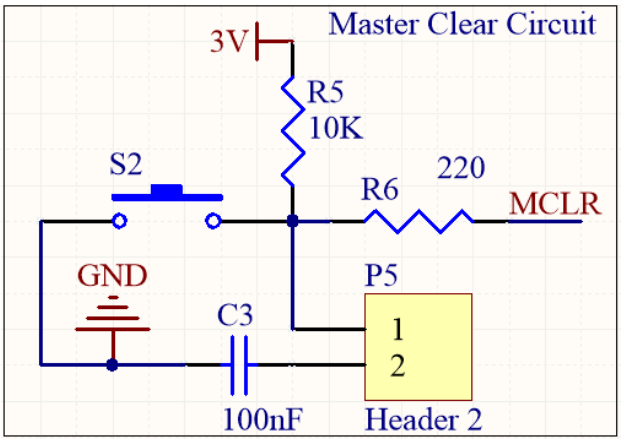


Figura 118. Circuito de botón y luz de prueba.

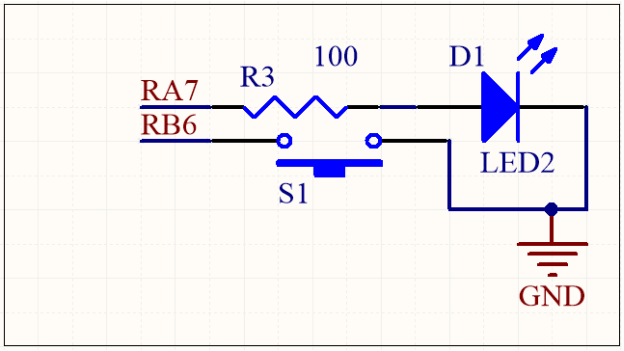


Figura 119. Interruptor de encendido/apagado de la plataforma de comunicación.

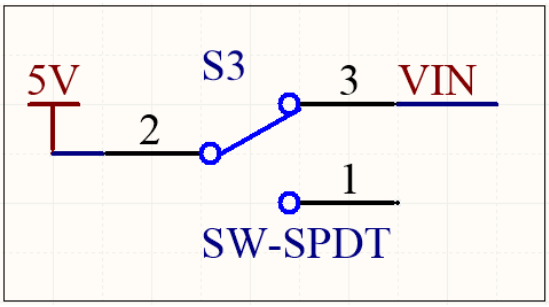


Figura 120. Puerto GPIO de la plataforma Raspberry Pi.

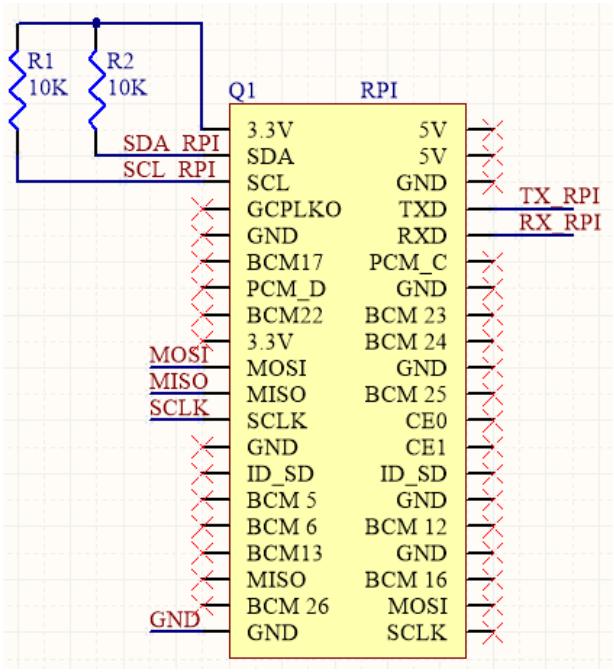


Figura 121. Conexiones para el microcontrolador PIC18F24J50.

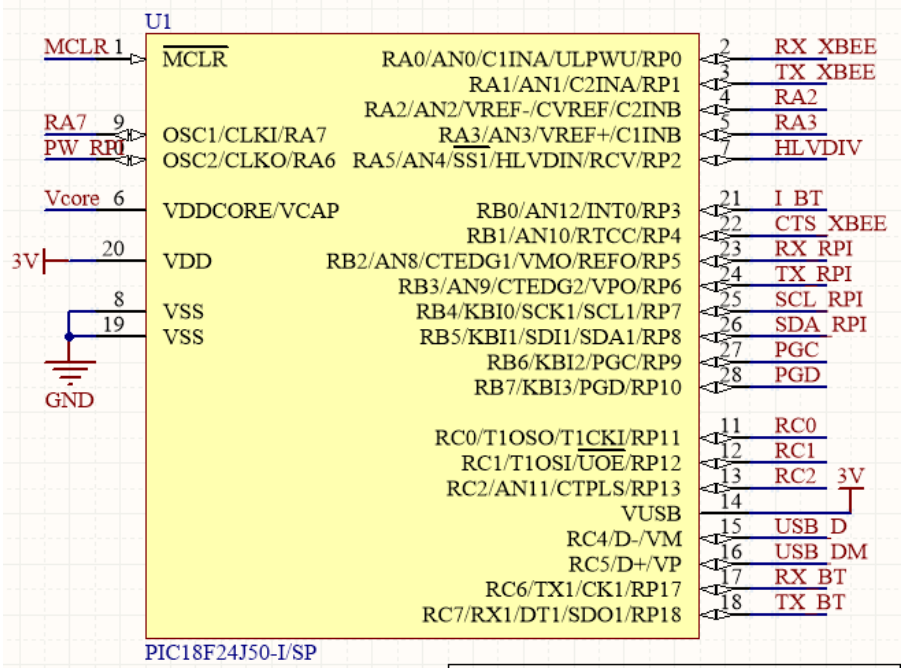


Figura 122. Diseño PCB para la plataforma de la base en tierra. Versión que no incluye conector para el HUB USB.

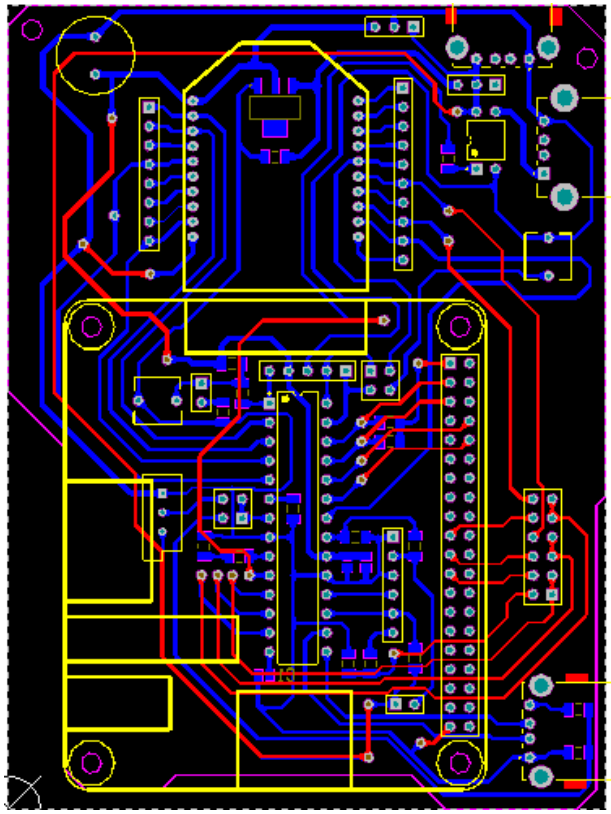


Figura 123. Fabricación de la plataforma de la base en tierra. Vista en planta.

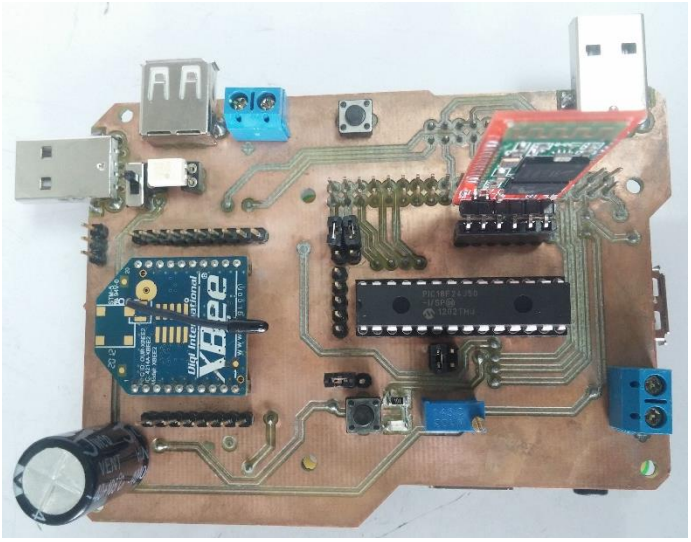


Figura 124. Fabricación de la plataforma de la base en tierra. Vista en base.

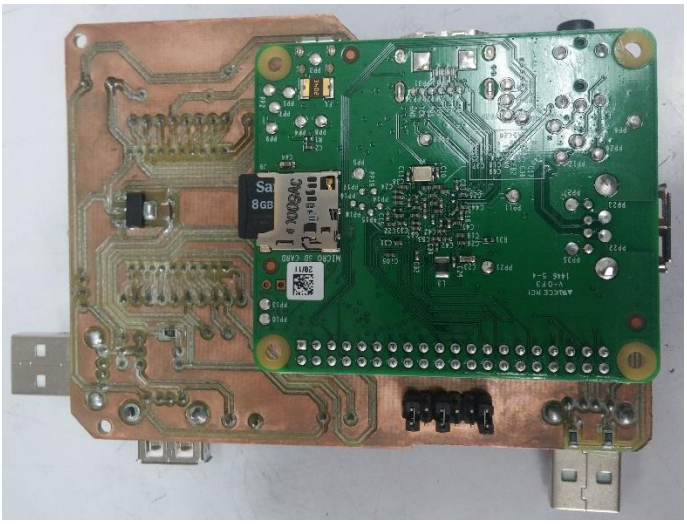
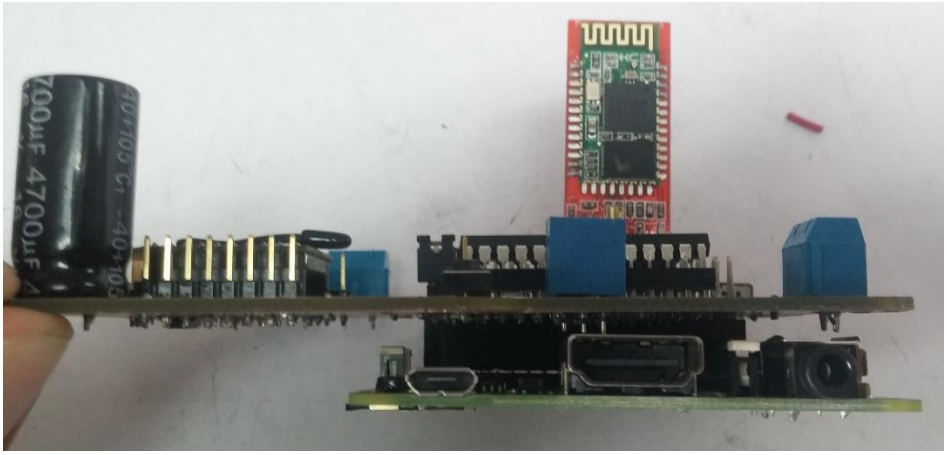


Figura 125. Fabricación de la plataforma de la base en tierra. Vista en frontal.



2. Resultado del diseño de la plataforma de comunicación de la base en tierra

Figura 126. Diseño de la plataforma de comunicación de la aeronave.

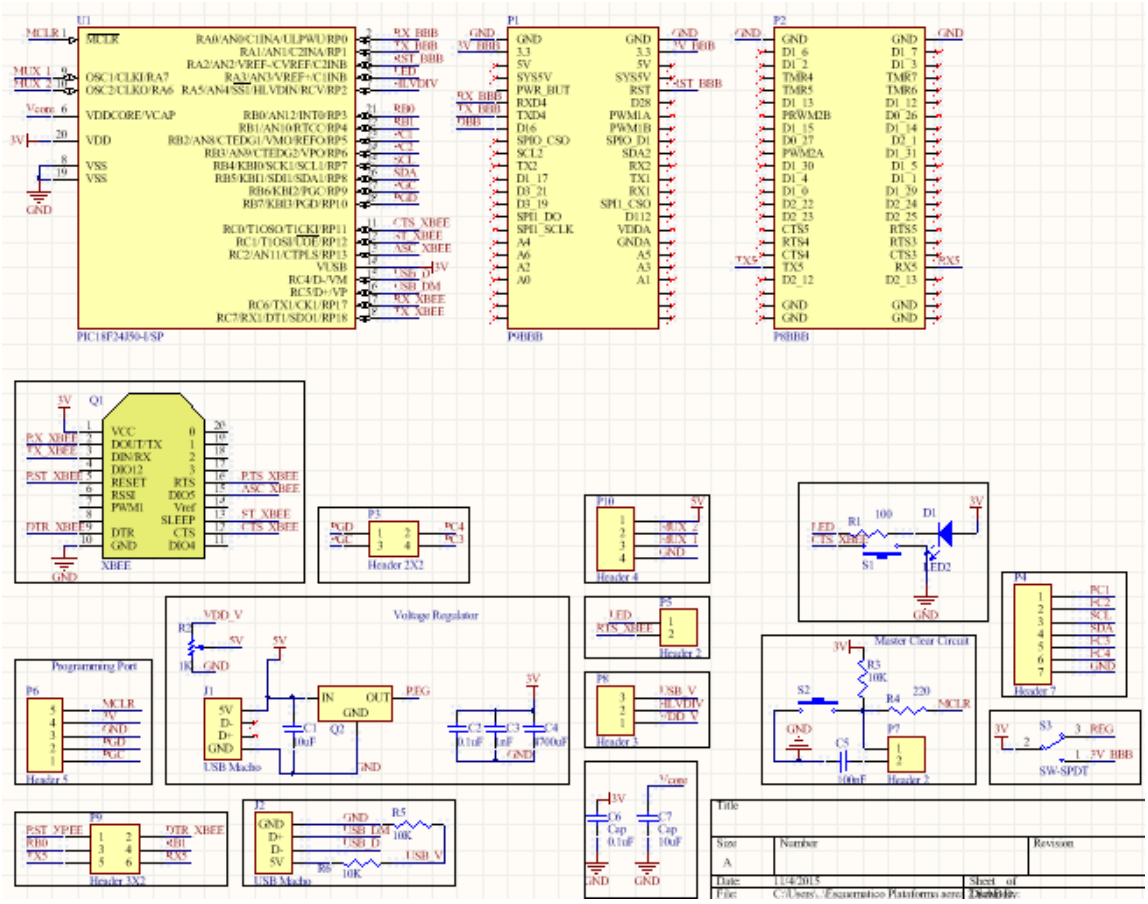


Figura 127. Conexión del módulo XBEE

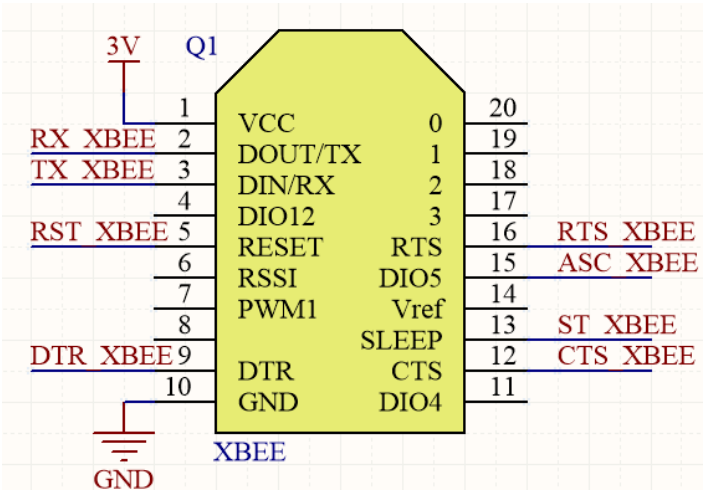


Figura 128. Regulador de voltaje.

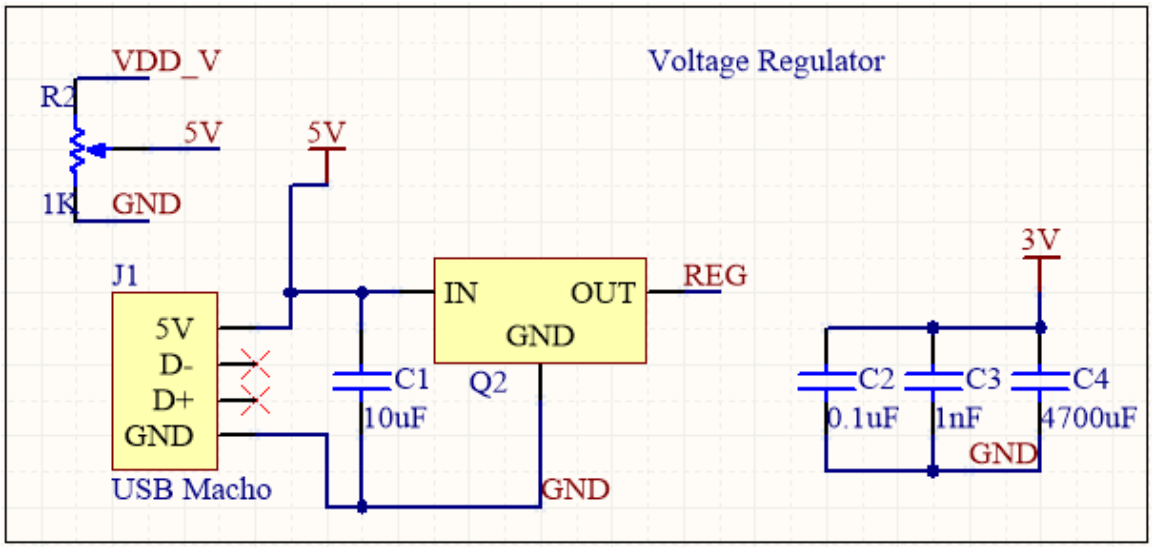


Figura 129. Conexiones a BeagleBone Black.

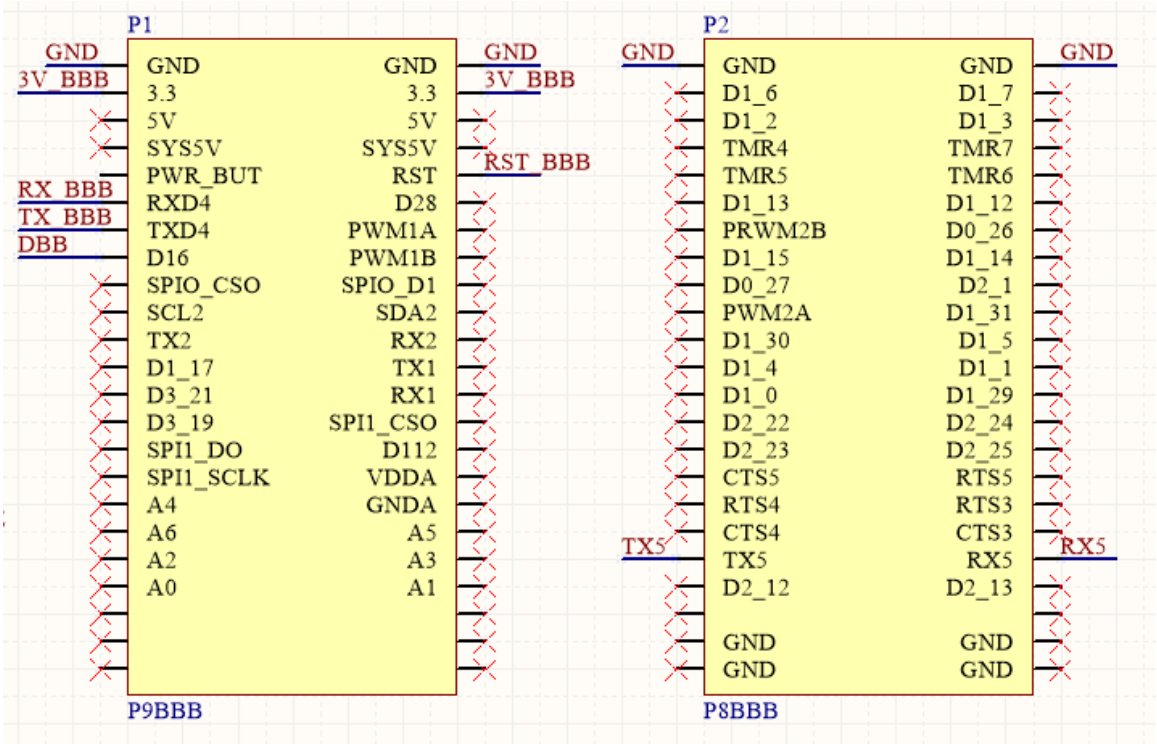


Figura 130. Conexión de PIC18F24J50.

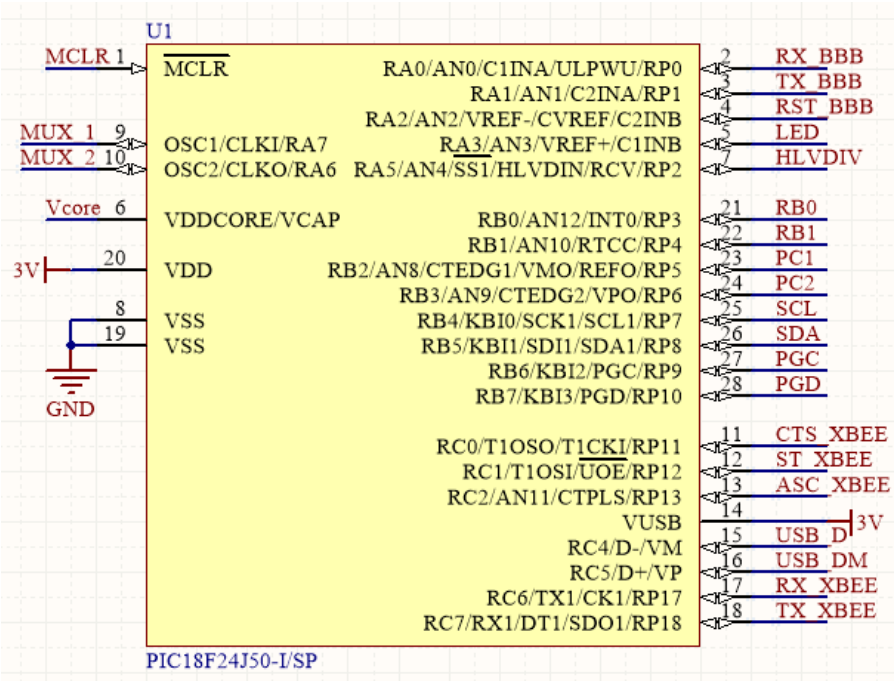
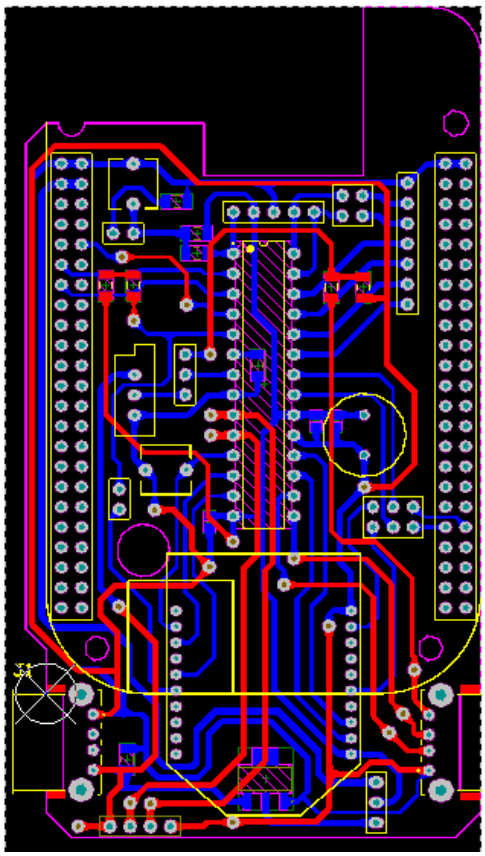


Figura 131. Diseño en PCB de la plataforma de comunicación para la aeronave.



3. Resultados del diseño del sistema de emergencia. El resultado de las pruebas arrojó una preferencia marcada por utilizar la compañía Claro para el módulo Adafruit Fona ya que la compañía Tigo no permitió el uso de la tecnología GPRS ni la triangulación de posición los cuales son características muy deseadas para el módulo. Además de esto se determinó un tiempo mínimo de vida de la batería de 48 horas encendiendo el modulo por una hora diaria y utilizando la tecnología GPRS durante este tiempo.

Figura 132. Diseño de la plataforma de emergencia.

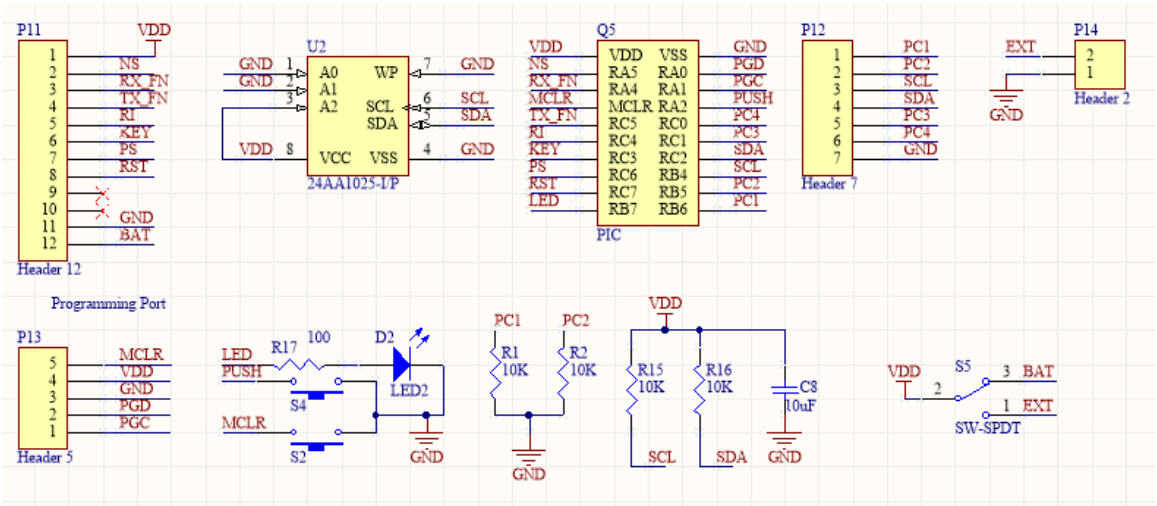


Figura 133. Conexión de la memoria EEPROM a través del bus I2C.

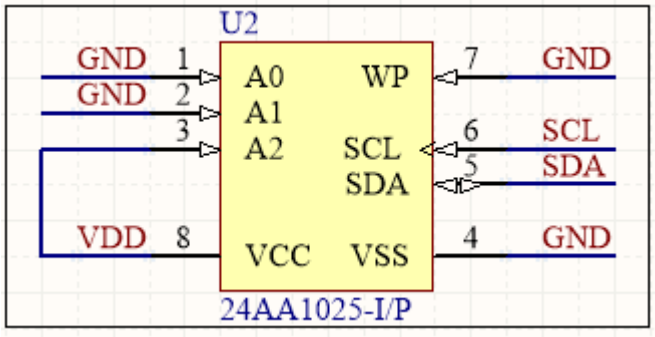


Figura 134. Puerto de conexión para un programador Pickit 3.

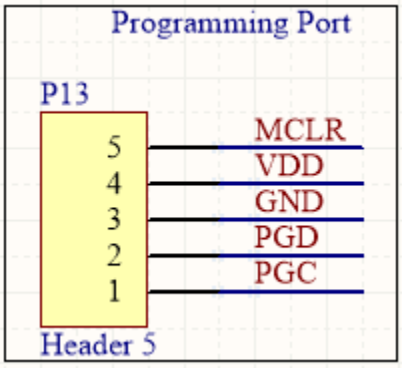


Figura 135. Selector de fuente de voltaje.

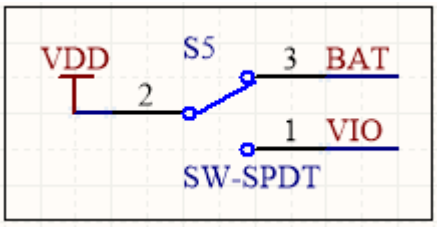


Figura 136. Circuitos de botones y luz de prueba.

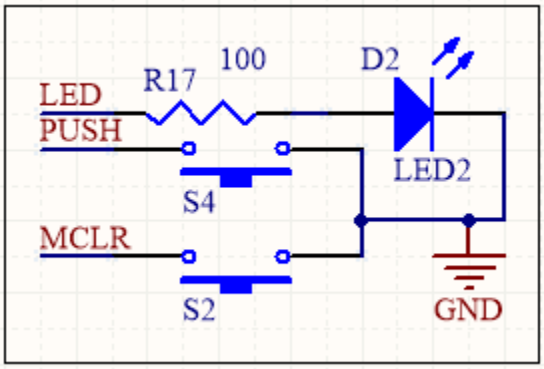


Figura 137. Red de resistencias de enclave a VDD.

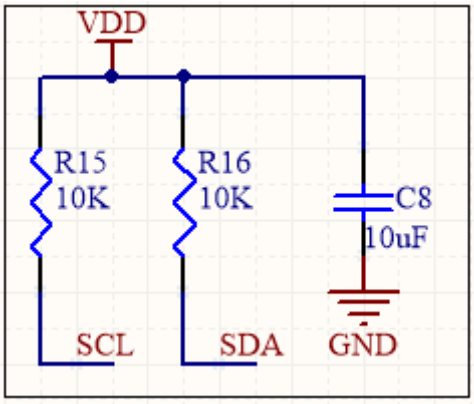


Figura 138. Puerto de conexión a plataforma de comunicación.

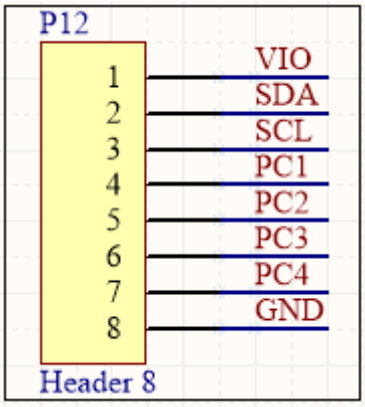


Figura 139. Puerto de conexión para módulo Adafruit FONA.

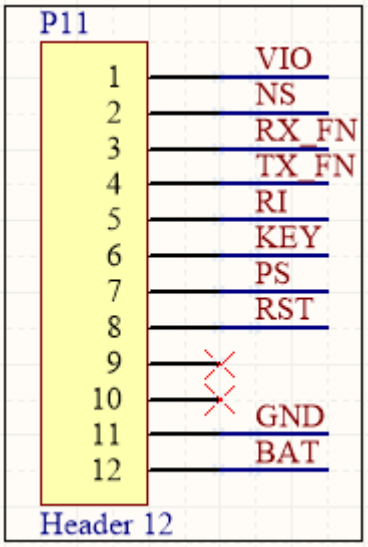


Figura 140. Conexión del microcontrolador PIC16F1619.

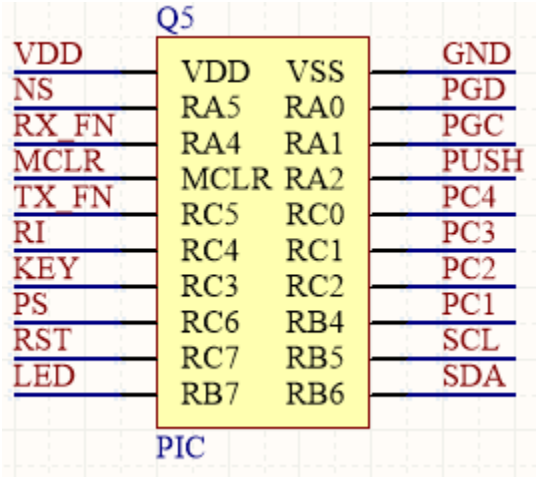


Figura 141. Diseño PCB para el sistema de comunicación de emergencia.

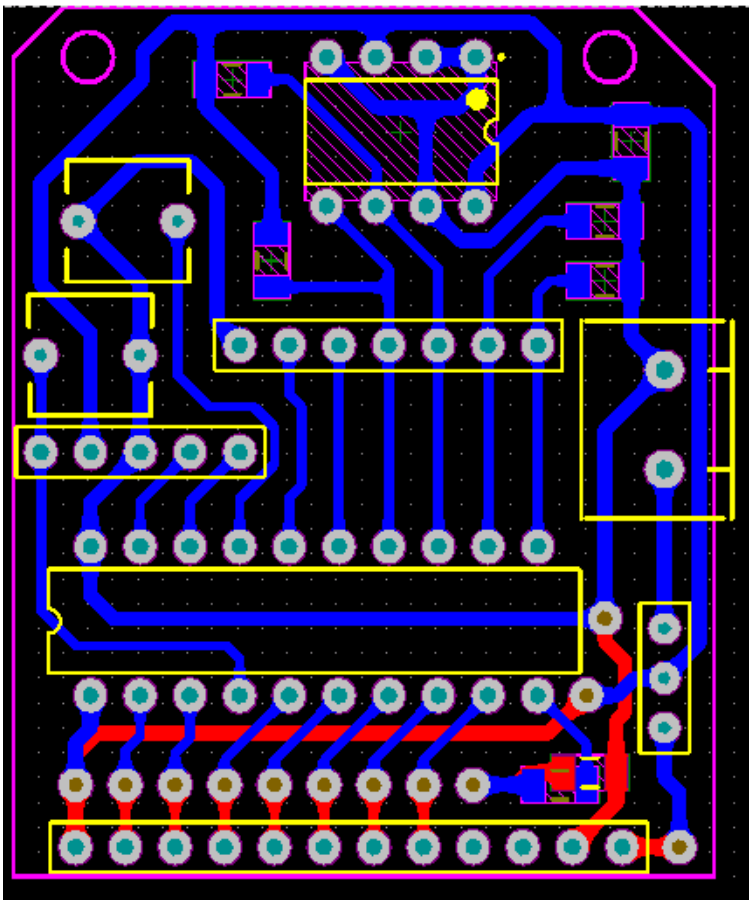


Figura 142. PCB del sistema de comunicación de emergencia sin ensamblar.

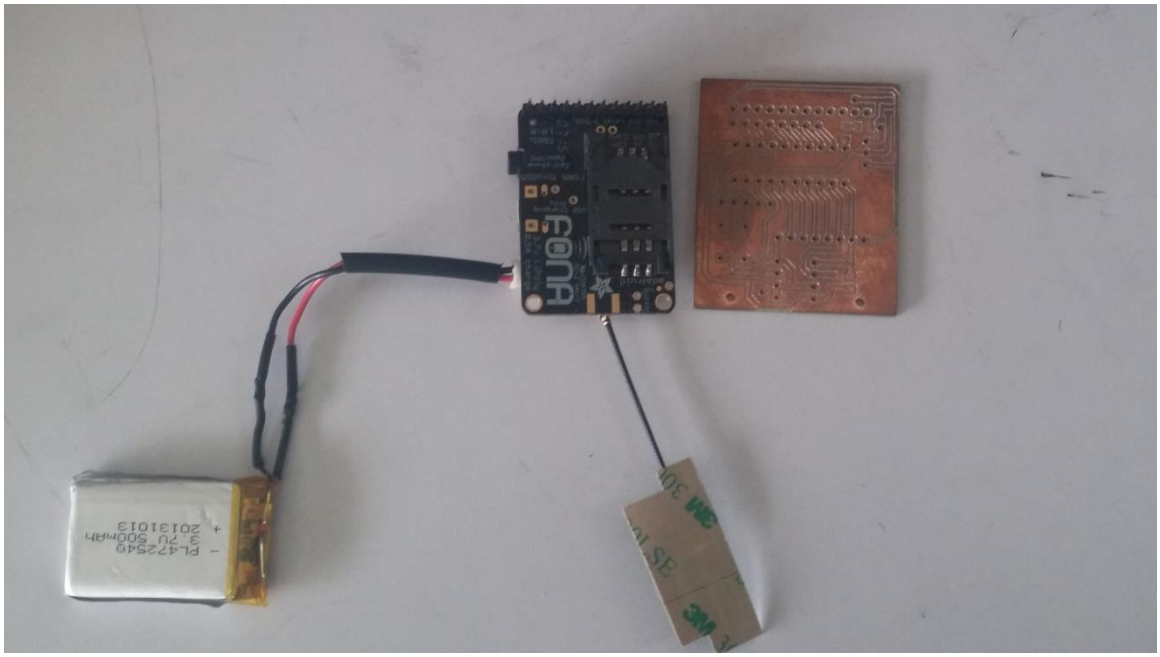
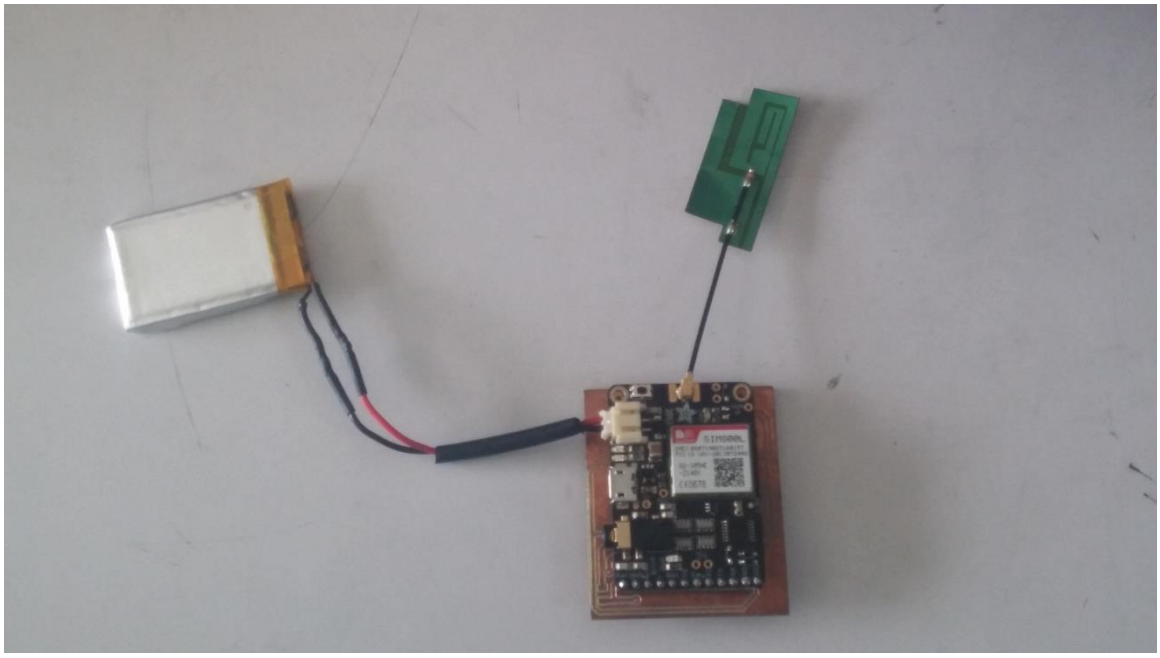


Figura 143. PCB del sistema de comunicación de emergencia ensamblado con los demás componentes.



4. Resultado de la implementación del sistema de comunicación de la base en tierra.

Figura 144. Carpetas de Dropbox sincronizadas con la aeronave.

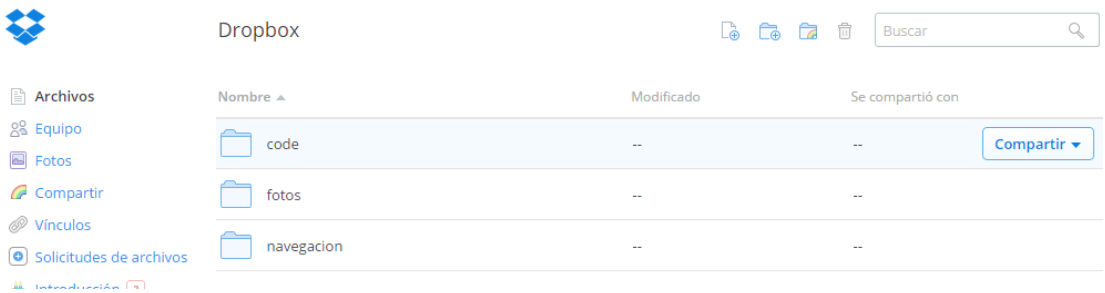


Figura 145. Imágenes recibidas de la aeronave.

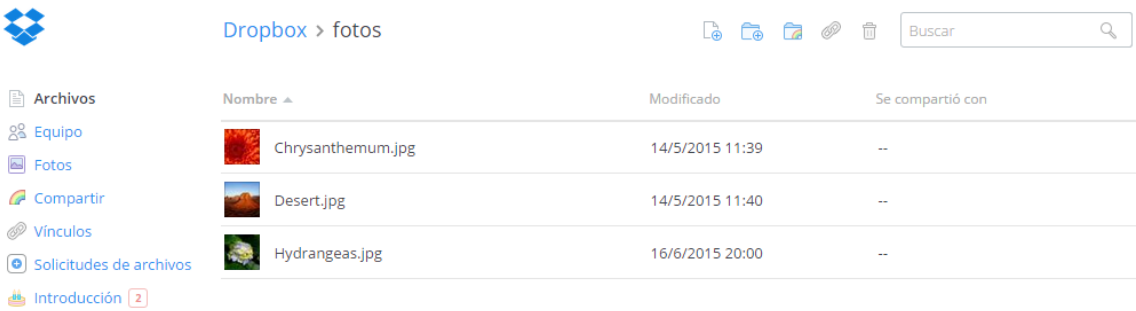


Figura 146. Código sincronizado con la base en tierra.

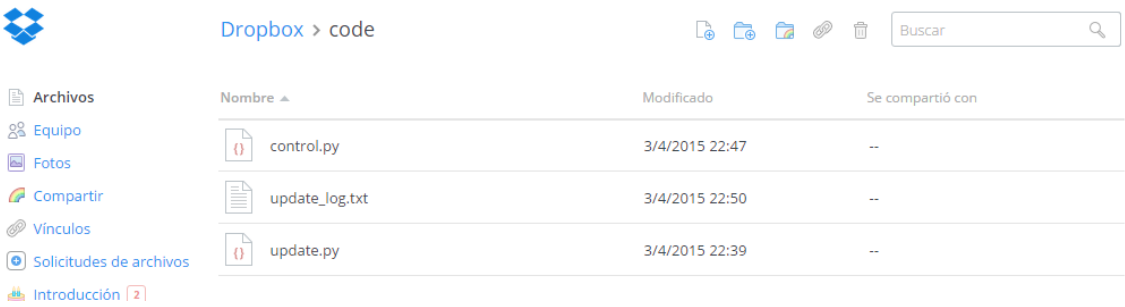


Figura 147. Diseño de la placa de interface entre USB y la base en tierra.

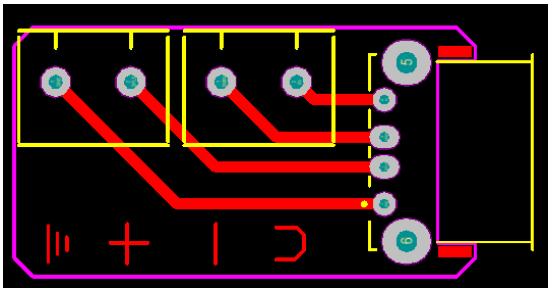
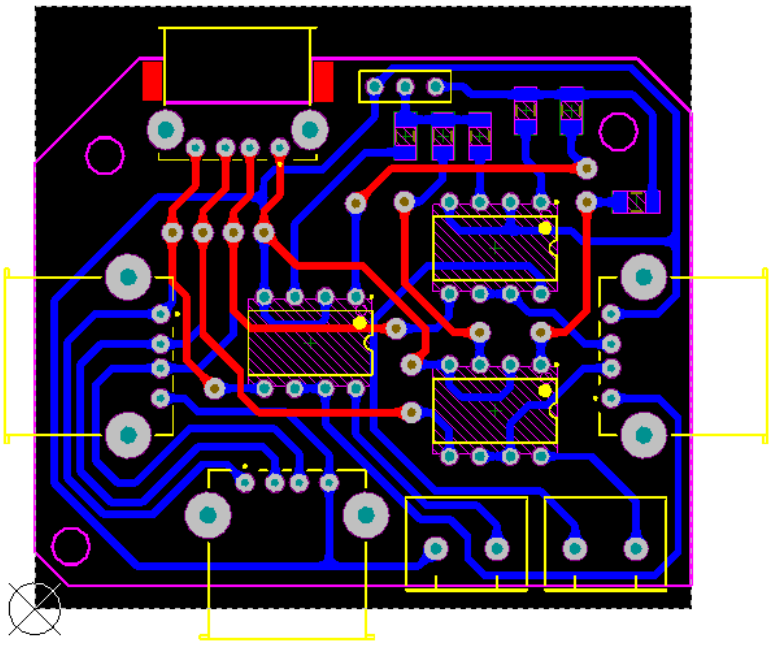


Figura 148. Diseño del multiplexor USB para la aeronave.



G. DISCUSIÓN

El objetivo principal del módulo es diseñar una plataforma de comunicación entre la aeronave, la base en tierra y el usuario del sistema. Podemos observar en la figura 107 que la plataforma está diseñada para la interconexión de todas las partes del sistema. En la figura 144, 145 y 146 se puede observar que existe una sincronización de datos entre la base en tierra y un repositorio, en este caso se escogió Dropbox el cual presenta mucha utilidad ya que puede ser manejado desde un dispositivo móvil o bien se pueden crear aplicaciones con su API. En la figura 146 se puede ver la carpeta de códigos de programa. Esta carpeta permite actualizar el software de la base en tierra de forma automática cada vez que la plataforma se encienda.

En la figura 108 se muestran las conexiones del Xbee. Se puede observar que solo están conectadas directamente las líneas de UART, sin embargo, las conexiones de los pines de control de flujo pueden conectarse a disposición del usuario. Obsérvese que la señal de RSSI no está dentro de las posibles conexiones con el microcontrolador, esto se debe a que experimentalmente la medición de tal señal no resulto ser de utilidad mientras que la potencia de señal recibida puede obtenerse a través de comandos AT.

En la figura 109 se muestra el regulador de voltaje del sistema. Las dos plataformas alimentan al microcontrolador con 3.3 voltios y utilizan los capacitores sugeridos entre sus terminales de potencia. Se puede observar un potenciómetro el cual crea un divisor de voltaje como salida. Este voltaje sirve de referencia en el caso el usuario desee utilizar el detector de voltaje bajo o alto seleccionable en la figura 103. Si no se utiliza este voltaje también puede seleccionarse el voltaje de entrada del puerto USB del microcontrolador mostrado en la figura 111 para detectar la conexión de un dispositivo al puerto.

XVI. CONCLUSIONES

1. Se logró controlar exitosamente la orientación, altura, posición y postura del cuadricóptero. Tal como se demostró durante las pruebas de vuelo.
2. El 3DR Iris+ logró levantar el peso de la tarjeta de control, sensores, carga útil y módulos de alimentación externa.
3. Dada la naturaleza sub-actuada del UAV, este no se puede controlar por completo en una sola etapa. Primero se controlan las velocidades y posteriormente la orientación.
4. Para la implementación de los sistemas de control, la tarjeta de control PX4 demostró ser la opción más adecuada. Debido a su naturaleza OpenSource y las librerías desarrolladas para el controlador.
5. La utilización de una computadora de compañía logró que el controlador no se saturara con cálculos de menor importancia como el cálculo de trayectorias y se concentrará en el control de vuelo.
6. El desarrollo modular de los algoritmos permitió la interacción entre la tarjeta de control (PX4) y la computadora de compañía.
7. El Robotic Operating System (ROS) permitió la comunicación entre los distintos algoritmos que se encontraban tanto en el controlador como en la computadora de compañía.
8. La distribución del peso del UAV es crítica para su correcto funcionamiento.
9. La navegación en ambientes negados de señal GPS puede llegar a ser problemática.
10. La implementación del link de radio entre la PC y la tarjeta de control facilitó el proceso de *debugging*.
11. Para la estimación de posición fue necesaria la integración de sensores de presión, sensores inerciales, GPS y LIDAR.
12. El procedimiento de sintonización, así como la implementación del sensor LIDAR mejoró el desempeño del UAV para controlar su altura.
13. Se logró la autonomía de vuelo al trazar las trayectorias con la computadora de compañía y enviar los parámetros de vuelo a la tarjeta de control.
14. El BeagleBone Black logró trazar las rutas de vuelo y convertirlas a parámetros aceptables para la tarjeta de control.
15. Por su parte la Raspberry Pi de primera generación no presento la capacidad de procesamiento necesaria para trabajar como computadora de compañía.
16. Se logró implementar un sistema de captura de imágenes multiespectrales a través de la utilización de una cámara NGR. Las imágenes capturadas hacen una distinción significativa entre la vegetación y demás objetos presentes en la escena.
17. Se logró mejorar la calidad de las imágenes multiespectrales capturadas por la cámara NGR, eliminando características no deseadas introducidas por las limitaciones de la cámara. La etapa de pre-procesamiento se compone de un algoritmo encargado de realzar los bordes en la imagen y de

un algoritmo encargado de ecualizar el histograma en las distintas regiones de la imagen de forma adaptativa.

18. Se logró generar una imagen en color falso que presentara un gradiente con los distintos niveles NDVI a partir de las imágenes capturadas por la cámara NGR. Para la generación de las imágenes en color falso se dispone de dos algoritmos distintos que presentan ventajas y desventajas de su uso.
19. Se logró implementar un sistema integrado capaz de realizar el procesamiento de imágenes multiespectrales durante las sesiones de vuelo del UAV. El sistema se logró implementar a través de la configuración y personalización de la plataforma del BeagleBone Black.
20. Se logró desarrollar un algoritmo capaz de realizar la captura y procesamiento de imágenes multiespectrales a través de una señal de entrada en los puertos GPIO del BeagleBone Black. El algoritmo resulta ser ideal para realizar pruebas manuales y fotografía de áreas específicas.
21. Se puede utilizar el diseño propuesto para implementar un sistema de comunicación entre todos los nodos del sistema. Con el diseño propuesto se logra obtener un sistema de comunicación de emergencia, la recolección de datos de la aeronave de forma automática, la comunicación entre la base en tierra y la aeronave y sistema de actualización y depuración del mismo.
22. La recolección de datos del vehículo aéreo requiere de un computador con una conexión a internet, una aplicación de alojamiento de archivos en la nube o de transferencia de datos a un servidor y una plataforma entre la aeronave y el computador que permita la transferencia de datos.
23. Los módulos de comunicación Xbee proveen una solución sencilla y robusta para la comunicación entre dos puntos si estos se encuentran dentro de una distancia máxima y en línea de vista.
24. El servicio de alojamientos de archivos Dropbox es una forma integral de implementar la sincronización entre los datos del usuario y los datos del sistema.
25. El uso de una WPAN implementada con Bluetooth es una forma práctica de comunicación con el sistema completo. El uso de un dispositivo móvil como terminal permite la programación de una interfaz intuitiva y brinda facilidad de control y depuración.
26. El uso del sistema operativo Raspbian es inadecuado para la base en tierra si se requiere la compatibilidad con diferentes módems inalámbricos distribuidos en Guatemala.
27. El sistema de actualización de software hace más eficiente el proceso de prueba de software de la base en tierra.
28. El uso de una tarjeta SIM Tigo no es apropiado para el módulo Adafruit Fona.
29. El uso de triangulación de posición en el sistema de comunicación de emergencia es una guia para la localización de la aeronave pero no es suficiente por su poca precisión.

XVII. RECOMENDACIONES

1. Se recomienda la implementación de un sistema de captura de movimiento VICON para el desarrollo de pruebas con robots móviles.
2. Se recomienda la construcción de un espacio dedicado únicamente a las pruebas de robots móviles.
3. Se recomienda liberar los puertos y dar más privilegios de acceso a la red UVG del laboratorio J313 para facilitar el desarrollo la investigación.
4. Se recomienda no utilizar la Raspberry Pi para la generación de trayectorias debido a su velocidad de procesamiento.
5. Para futuras capturas de imágenes multiespectrales se recomienda utilizar un arreglo de cuatro cámaras para la adquisición de las distintas bandas de interés. Esto con el fin primero de contar con más bandas para la implementación de algoritmos más robustos y segundo para evitar que la banda NIR se fugue en las bandas correspondientes al espectro visible.
6. Se recomienda desarrollar un sistema que integre el análisis espectral de un cultivo a través de la captura de imágenes aéreas con el uso de sondas de tierra. El objetivo de este sistema es el de proveer al índice de vegetación con información respecto a las propiedades del suelo, de forma que el índice se pueda ajustar a dichas condiciones.
7. Se recomienda desarrollar un sistema que integre el análisis espectral de un cultivo a través de la captura de imágenes aéreas con el uso de una estación meteorológica. El objetivo de este sistema es el de proveer al índice de vegetación con información respecto a las variantes climáticas, de forma que el índice se pueda ajustar a dichas condiciones.
8. Para obtener un índice de vegetación más personalizado se recomienda implementar un sistema clasificador para implementar un algoritmo adaptativo que halle una correlación más exacta entre un cultivo en específico y el índice de vegetación utilizado.
9. Se recomiendo intentar utilizar Windows 10 IOT core para verificar el funcionamiento de una gama más amplia de módems inalámbricos y una simplificación en el proceso de configuración del sistema operativo.
10. El sistema de comunicación de emergencia puede ser mejorado utilizando un módulo Adafruit Fona 808 para obtener la posición de la aeronave de manera más precisa utilizando el módulo GPS incorporado.
11. Se puede medir la eficiencia energética de las plataformas de procesamiento calculando la potencia requerida para la sincronización de datos y obteniendo el tiempo en el que estas terminan del ciclo de sincronización de tal modo que se pueda calcular la energía consumida.

XVIII. BIBLOGRAFÍA

3D Robotics. s.f. Iris+. Último acceso: 3 de Diciembre de 2014. http://3drobotics.com/kb/iris/.

Ada, Lady. Adafruit Fona. https://learn.adafruit.com/downloads/pdf/adafruit-fona-mini-gsm-gprs-cellular-phone-module.pdf [10 de Abril del 2015]

Adafruit. *Embedded Linux Board Comparison*. https://learn.adafruit.com/embedded-linux-board-comparison/overview [01/10/15]

Adam. Scripts. http://www.raspberry-projects.com/pi/pi-operating-systems/raspbian/scripts [1 de octubre del 2015]

Alava Ingenieros. *¿Qué diferencia una imagen multiespectral de una hiperespectral?* http://www.alava-ing.es/ingenieros/actualidad/que-diferencia-una-imagen-multiespectral-de-una-hiperespectral/ [29/09/15]

APNchangeR. Guatemala. http://wiki.apnchanger.org/Guatemala [1 de octubre del 2015]

Argueta, Juan Pablo. 2015. Aplicación de un vehículo aéreo no tripulado aplicado al análisis de cultivos agrícolas: Módulo de control. Guatemala: Universidad del Valle de Guatemala.

Ascending Technologies. s.f. AscTec Firefly. Último acceso: 28 de enero de 2015. http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-firefly/#pane-0-0.

Ascending Technologies. s.f. AscTec Hummingbird. Último acceso: 3 de diciembre de 2014. http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-hummingbird/.

Beagleboard. *About BeagleBoard.org.* http://beagleboard.org/about [01/10/15]

Beagleboard. *Getting started with beaglebone & beaglebone black.* http://beagleboard.org/getting-started [01/10/15]

Beagleboard. *What is BeagleBone Black?* http://beagleboard.org/BLACK [01/10/15]

Bigshot. *Image Sensor*. http://www.bigshotcamera.com/learn/image-sensor/index [29/09/15]

Bluetooth SIG. Bluetooth Basics. http://www.bluetooth.com/Pages/Basics.aspx [30 de septiembre del 2015]

Botero, Juan Manuel. 2009. *Determinación del nivel foliar de nutrientes mediante espectroscopia de reflectancia*. Tesis de Universidad Nacional de Colombia. Medellín: Facultad de Ciencias, Área de Ciencias Naturales. 115 págs.

Boubdallah , Samir, Pierpaolo Muerrieri, y Roland Siegwart. 2006. «Design and Control of an Indoor Micro Quadrotor.» 6.

Byte Paradigm. I2C vs SPI. http://www.byteparadigm.com/applications/introduction-to-i2c-and-spi-protocols/ [1 de octubre del 2015]

Carrera, J. 2002. Situación actual y perspectivas de la agricultura en Guatemala. Guatemala: Universidad Rafael Landivar.

Carrera, José. Situación actual y perspectivas de la agricultura en Guatemala. http://biblio3.url.edu.gt/IARNA/SERIETECNINCA/4.pdf [29/09/15]

Castillo, Pedro, Rogelio Lozano , y Alejandro Dzul. 2005. Modelling and Control of Mini-Flyin Machines. Londres: Springer-Verlag.

Corke, Peter. 2011. Robotics, Vision and Control. Berlin: Springer-Verlag.

Dashboard Camera Reviews. *Mobius/Mobius*. http://dashboardcamerareviews.com/mobius-action-camera-review/ [01/10/15]

Diccionario de la lengua española. 2001. Real Academia Española de la Lengua. 22ª ed. Madrid: Espasa-Calpe. 2 vols.

Dicola, Toni. Embedded Linux Board Comparison. https://learn.adafruit.com/embedded-linux-board-comparison/overview [30 de septiembre del 2015]

Dirección de planeamiento. 2013. El agro en Cifras 2013. Guatemala: Ministerio de Agrigultura Ganaderia y Alimentación.

Dirección de planeamiento. El agro en cifras 2013. http://web.maga.gob.gt/download/El-agro-en-cifras-small.pdf [29/09/15]

Dji. s.f. Spreding Wings S900. Último acceso: 4 de diciembre de 2014. http://www.dji.com/product/spreading-wings-s900.

Dunlop, John; D. Girma y J. Irvine. 1999. Digital mobile communications and the TETRA system. Chichester: Wiley. 427 págs.

eLinux. RPi adding USB drives. http://elinux.org/RPi_Adding_USB_Drives [1 de octubre del 2015]

eLinux. RPi Verified Peripherals. http://elinux.org/RPi_VerifiedPeripherals. [1 de octubre del 2015]

Escalante, Boris. 2006. *Procesamiento digital de imágenes*. México, D.F.: Departamento de Procesamiento de Señales, UNAM. 12 págs.

Fabrizi, Andrea. Dropbox-Uploader. https://github.com/andreafabrizi/Dropbox-Uploader [1 de octubre del 2015]

Fallas, Jorge. 2004. *Uso de imágenes multiespectrales*. 1ª ed. Costa Rica: Laboratorio de Teledetección y Sistemas de Información Geográfica, Universidad Nacional. 36 págs.

Garcia, Luis Rodolfo, Alejandro Dzul, Rogelio Lozano, y Claude Pégard. 2013. Quad Rotorcraft Control. Londres: Springer.

GitHub. sakis3g-source. https://github.com/trixarian/sakis3g-source [1 de octubre del 2015]

Gobierno de Guatemala. Guatemala utiliza el 67 de su territorio para la agricultura. http://www.guatemala.gob.gt/index.php/2011-08-04-18-06-26/item/8429-guatemala-utiliza-67-por-ciento-de-su-territorio-para-la-agricultura [29/09/2015]

Heath, Steve. 2003. Embedded systems design. 2ª ed. Oxford: Newnes. 430 págs.

IEEE Computer Society. 2002. <<Part 15.1: wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPANs)>>. 802.15.1: IEEE standard for information technology-telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements. Nueva York. IEEE. 1147 págs.

Institute of new imaging technologies (INIT). *Aplicaciones de imágenes multiespectrales*. http://www.init.uji.es/index.php?option=com_content&view=article&id=94:aplicacionesdeimagenes multiespectrales&catid=57:servicios-vision-por-ordenador&Itemid=47&lang=es [29/09/15]

Instituto de Tecnologías Educativas. *Características de la imagen digital*. http://www.ite.educacion.es/formacion/materiales/86/cd/m2/index.html [29/09/15]

Iseez. *About*. http://opencv.org/about.html [01/10/15]

Kurose, James; Ross, Keith. 2013. Computer networking: a top-down approach. 6a ed. Boston: Pearson. 862 págs.

Landscape Toolbox. *Enhanced Vegetation Index*. http://wiki.landscapetoolbox.org/doku.php/remote_sensing_methods:enhanced_vegetation_index [29/09/15]

Landscape Toolbox. *Modified Soil Adjusted Vegetation Index*. http://wiki.landscapetoolbox.org/doku.php/remote_sensing_methods:modified_soil-adjusted_vegetation_index [29/09/15]

Landscape Toolbox. *Normalized Difference Vegetation Index*. http://wiki.landscapetoolbox.org/doku.php/remote_sensing_methods:normalized_difference_vegetation_index [29/09/15]

Landscape Toolbox. *Soil Adjusted Vegetation Index*. http://wiki.landscapetoolbox.org/doku.php/remote_sensing_methods:soil-adjusted_vegetation_index [29/09/15]

Linux die. wvdial.conf(5) - Linux man page. http://linux.die.net/man/5/wvdial.conf [1 de octubre del 2015]

Lira, Jorge. 2010. *Tratamiento Digital de Imágenes Multiespectrales*. 2ª ed. México, D.F.: Instituto de Geofísica, UNAM. 605 págs.

Lupashin, Sergei, Markus Hehn, Mark Mueller, Angela Schoelling, Michael Sherback, y Rafaello D'Andrea. 2014. «A platform for aerial robotics research and demonstration: The Flying Machine Arena.» Mechatronics (24): 41-54.

Mahony, R, T hamel, y J Pflimin. 2008. «Non-linear complementary filters on the special orthogonal group.» IEEE Trans. Automat. Contr. (5): 1203-1218.

Mahony, Robert, Vijay Kumar , y Peter Corke. 2012. «Multirotor Aerial Vehicles.» IEEE ROBOTICS & AUTOMATION MAGAZINE (9): 20 -32.

Martin, P, y E Shaun. 2010. «The true role of accelerometre feedback in quadrotor control.» Proc. IEEE Int. Conf. Robotics and Automation 1623-1629.

Martinez, Aaron, y Enrique Fernández. 2013. Learning ROS for Robotics Programming. Birmingham: Packt Publishing Ltd.

Meier, Lorenz, Dominik Honegger, y Marc Pollefeys. 2015. «PX4: A Node-Based Multhithreaded Open Source Robotics Framework for Deeply Embedded Platforms.» Int. Conf. on Robotics and Automation.

Ministerio de Agricultura. 2014. Guatemala utiliza el 67% de su territorio para la agricultura. 23 de Mayo. Último acceso: 30 de Junio de 2015. http://www.guatemala.gob.gt/index.php/2011-08-04-18-06-26/item/8429-guatemala-utiliza-67-por-ciento-de-su-territorio-para-la-agricultura.

Narciso, Ruben. Algunas características de la población y la población económicamente activa de Guatemala. http://www.ebg.edu.gt/wp-content/files_mf/1411408349RubenNarciso.pdf [29/09/15]

Nice, Eryk Brian. 2004. Design of a Four Rotor Hovering Vehicle. Ithaca: Cornell University.

Ogata, Katsuhiko. 2010. Modern Control Engineering. Nueva Jersey: Prentice hall.

O'Kane. 2014. A Gentle Introduction to ROS. Columbia: University of South Carolina.

Oppehnheim, Alan, y George Verghese. 2010. Signals, Systems and Inference. Massachussets: Massachusetts Institue of Technology.

Pérez, Guillermo. *Espectro Electromagnético*. http://www.espectrometria.com/espectro_electromagntico [29/09/15]

Pérez, Guillermo. *Espectrometría.* http://www.espectrometria.com/ [29/09/15]

Peterson, Larry L.; Davie, Bruce S. 2012. Computer networks: a systems approach. 5a ed. Amsterdam: Morgan Kaufmann. 884 págs.

Pixhawk. 2014. List of on-board applications. Último acceso: 15 de febrero de 2015. https://pixhawk.org/firmware/apps/start.

Pixhawk. 2014. Software Architecture. Último acceso: 15 de February de 2015. https://pixhawk.org/dev/software_architecture.

Pixhawk. s.f. Controller Architecture. Último acceso: 20 de septiembre de 2015. https://pixhawk.org/dev/controller_architecture.

Pixhawk. s.f. Flight Modes. Último acceso: 17 de marzo de 2015. https://pixhawk.org/users/system_modes.

Pixhawk. s.f. Multirotor Flight Control. Último acceso: 8 de abril de 2015. https://pixhawk.org/dev/multirotor/start.

Prasad, K.V. 2004. Principles of digital communication systems and computer networks. 1ª ed. Hingham: Charles River Media. 742 pag.

PublicLab. *About Public Lab*. http://publiclab.org/about [29/09/15]

PublicLab. *Infragram Point & Shoot*. http://publiclab.org/wiki/infragram-point-shoot [01/10/15]

PublicLab. *Near-infrared camera history*. http://publiclab.org/wiki/near-infrared-camera-history [29/09/15]

PublicLab. *Near-infrared camera*. http://publiclab.org/wiki/near-infrared-camera [29/09/15]

PublicLab. *Red vs blue filters for NDVI*. http://publiclab.org/notes/nedhorning/10-30-2013/red-vs-blue-filters-for-ndvi [01/10/15]

PublicLab. *White balance and NIR photography*. http://publiclab.org/notes/nedhorning/11-3-2012/white-balance-and-nir-photography [01/10/15]

PX4 Autopilot. 2014. The PX4 Board. 4 de marzo. Último acceso: 3 de febrero de 2015. https://pixhawk.org/modules/pixhawk.

PX4 Project. s.f. «PX4 Autopilot.» Último acceso: 24 de abril de 2015. https://github.com/PX4.

Rapberry Pi Spy. How to mount a USB flash disk on the Raspberry Pi. http://www.raspberrypi-spy.co.uk/2014/05/how-to-mount-a-usb-flash-disk-on-the-raspberry-pi/ [1 de octubre del 2015]

Raspberry Pi Foundation. What is a raspberry pi?. https://www.raspberrypi.org/help/what-is-a-raspberry-pi/ [27 de septiembre de 2015]

Raspibo. Huawei E353 HSPA+ USB stick. http://www.raspibo.org/wiki/index.php/Huawei_E353_HSPA%2B_Usb_Stick [1 de octubre del 2015]

RasPiTV. How to mount a USB flash drive on Raspberry Pi. http://raspi.tv/2012/mount-a-usb-flash-drive-on-raspberry-pi [1 de octubre del 2015]

RasPiTV. How to use Dropbox with Raspberry Pi. http://raspi.tv/2013/how-to-use-dropbox-with-raspberry-pi [1 de octubre del 2015]

Richee. XBee S2 quick reference guide/cheat sheet and video tutorials to getting started. http://www.tunnelsup.com/xbee-guide/ [1 de octubre del 2015]

Salazar, Luis Alonso. 2007. *Integrating remote sensing, geographic information system and modeling for estimating crop yield*. Tesis de Universidad de New York. New York: Facultad de Ingeniería Eléctrica. 287 págs.

Sauter, Martin. 2011. From GSM to LTE: an introduction to mobile networks and mobile broadband. 1a ed. Chichester: Wiley. 414 págs.

Siciliano, Bruno, y Oussama Khatib. 2008. Springer Handbook of Robotics. Berlin: Springer-Verlag.

Singh, R. P.; Sapre, S. D. 2008. Communication systems: analog & digital. 2ª ed. Nueva Delhi: Tata McGraw-Hill. 608 págs.

Solarian Programmer. *Getting started with OpenCV on the BeagleBone Black with Ubuntu 14.04 LTS*. https://solarianprogrammer.com/2014/04/21/opencv-beaglebone-black-ubuntu/ [01/10/15]

Sturm, Jürgen. 2015. Autonomous Navigation for Flying Robots. Munich: Technische Universität München.

Tedrake, Russ. 2014. «Underactuated Rotobics: Algorithms for Walking, Running, Swimming, Flying and Manipulation (Course Notes for MIT 6.832).» Último acceso: 12 de septiembre de 2015. http://people.csail.mit.edu/russt/underactuated/.

Terence Eden's Blog. 3G internet on Raspberry Pi - success!. https://shkspr.mobi/blog/2012/07/3g-internet-on-raspberry-pi-success/ [1 de octubre del 2015]

Tetracam. *Tetracam products*. http://www.tetracam.com/Products1.htm [01/10/15]

The fan club. How to setup a USB 3G Modem on Raspberry PI using usb_modeswitch and wvdial. https://www.thefanclub.co.za/how-to/how-setup-usb-3g-modem-raspberry-pi-using-usbmodeswitch-and-wvdial. [1 de octubre del 2015]

Trevor's "Raspberry Pi" Wiki. Using multiple USB sticks (flash drives) on the Raspberry Pi. http://www.cpmspectrepi.webspace.virginmedia.com/raspberry_pi/MoinMoinExport/MultipleUsbSticks.html [1 de octubre del 2015]

Universidad Nacional de San Luis (UNSL). *Tratamiento de la imagen digital*. http://tecno.unsl.edu.ar/multimedia/Imagen/notas%20imagenes%20digitales.pdf [29/09/15]

Universidad Nacional del Centro (UNICEN). *Procesamiento digital de imágenes*. http://www.exa.unicen.edu.ar/catedras/pdi/FILES/TE/CP1.pdf [29/09/15]

Velasquez, Luis C. 2015. «Pruebas vuelo manual [Archivo de Video].» 19 de febrero. Último acceso: 10 de septiembre de 2015. https://app.box.com/s/aw922evalwlo6twbllb0uh0695mlncdf.

Velasquez, Luis C. s.f. «Desbalance UAV [Archivo de Video].» Último acceso: 17 de septiembre de 2015. https://app.box.com/s/z4qvgjmwc8iblg27kv7x0eggi0q51w1t.

Velasquez, Luis C. s.f. «Muestreo Cámara Multiespectral [Archivo de Video].» Último acceso: 17 de septiembre de 2015. https://app.box.com/s/2821yy26a2b8fat9cvnml5obh8tj20w9.

Velasquez, Luis C. s.f. «Prueba estabilidad previo a sintonización [Archivo de Video].» Último acceso: 220 de junio de 2015. https://app.box.com/EstabilidadPrevioTunning.

Velasquez, Luis C. s.f. «Prueba Vuelo Autonomo [Archivo de Video].» Último acceso: 4 de Julio de 2015. https://app.box.com/s/zev22decjldelape5dxmsasjf5w1lgnn.

Velasquez, Luis C. s.f. «Pruebas con Viento [Archivo de Video].» Último acceso: 8 de Julio de 2015. https://app.box.com/s/3tlk8g23jislq23dq1qulnwrzf1ukakq.

Velasquez, Luis C. s.f. «Sintonización UAV [Archivo de Video].» Último acceso: 28 de junio de 2015. https://app.box.com/s/c4ztdb99uf7492q5mc8j711vh1p8q3u9.

Weier, John; Herring, David. *Measuring Vegetation (NDVI & EVI).* http://earthobservatory.nasa.gov/Features/MeasuringVegetation/measuring_vegetation_1.php [29/09/15]

XIX. ANEXOS

A. PRUEBA DE MERCADO DE PLATAFORMAS BASADAS EN LINUX REALIZADA POR INDUSTRIAS ADAFRUIT

Figura 149. Tabla comparativa de las especificaciones técnicas para cada plataforma

	Arduino Yun	Beaglebone Black	Intel Galileo	Raspberry Pi
Picture				
SoC	Atheros AR9331	Texas Instruments AM3358	Intel Quark X1000	Broadcom BCM2835
CPU	MIPS32 24K and ATmega32U4	ARM Cortex-A8	Intel X1000	ARM1176
Architecture	MIPS and AVR	ARMv7	i586	ARMv6
Speed	400mhz (AR9331) and 16mhz (ATmega)	1ghz	400mhz	700mhz
Memory	64MB (AR9331) and 2.5KB (ATmega)	512MB	256MB	256MB (model A) or 512MB (model B)
FPU	None (Software)	Hardware	Hardware	Hardware
GPU	None	PowerVR SGX530	None	Broadcom VideoCore IV
Internal Storage	16MB (AR9331) and 32KB (ATmega)	2GB (rev B) or 4GB (rev C)	8MB	None
External Storage	MicroSD (AR9331)	MicroSD	MicroSD	SD card
Networking	10/100Mbit ethernet and 802.11b/g/n WiFi	10/100Mbit ethernet	10/100Mbit ethernet	None (model A) or 10/100Mbit ethernet (model B)
Power Source	5V from USB micro B connector, or header pin.	5V from USB mini B connector, 2.1mm jack, or header pin.	5V from 2.1mm jack, or header pin.	5V from USB micro B connector, or header pin.
Dimensions	2.7in x 2.1in (68.6mm x 53.3mm)	3.4in x 2.1in (86.4mm x 53.3mm)	4.2in x 2.8in (106.7mm x 71.1mm)	3.4in x 2.2in (85.6mm x 56mm)
Weight	1.4oz (41g)	1.4oz (40g)	1.8oz (50g)	1.6oz (45g)
Approximate Price	$75	$55 (rev C), $45 (rev B)	$80	$25 (model A), $35 (model B)

Figura 150. Tabla comparativa de los módulos de entrada y salida para cada plataforma

	Arduino Yun	BeagleBone Black	Intel Galileo	Raspberry Pi
Digital I/O Pins	20	65	14	17
Digital I/O Power	5V	3.3V	3.3V or 5V (switched with jumper)	3.3V
Analog Input	12 with 10-bit ADC, 0-5V (supports external reference input)	7 with 12-bit ADC, 0-1.8V (no external reference input)	6 with 12-bit ADC, 0-5V (no external reference input)	None
PWM Output	7	8	6 (limited speeds prevent fine servo control)	1
UART	2 (1 wired to AR9331)	4	2 (1 exposed through 3.5mm jack)	1
SPI	1	2	1	2
I2C	1	2	1	1
USB Host	1 standard A connector (AR9331)	1 standard A connector	1 micro AB connector	1 (Model A) or 2 (Model B) standard A connector
USB Client	1 micro B connector (ATmega)	1 mini B connector	1 micro B connector	None
Video Output	None	Micro HDMI	None	HDMI, Composite RCA, DSI
Video Input	None	None	None	CSI (camera)
Audio Output	None	Micro HDMI	None	HDMI, 3.5mm jack
Power Output	3.3V up to 50mA, 5V	3.3V up to 250mA, 5V up to 1A	3.3V up to 800mA, 5V up to 800mA	3.3V up to 50mA, 5V up to 300-500mA
Other	- All I/O routed to ATmega processor unless noted otherwise. - Hardware compatibility with most Arduino Leonardo compatible shields.	- Real-time support with programmable real-time units. - Many pins have multiple functions such as I2S audio, CAN bus, etc. See the wiki for more information.	- Mini-PCI Express slot. - Real-time clock with optional battery. - Mixed compatibility with Arduino shields.	

Figura 151. Gráfico de barras comparativo del desempeño computacional para cada plataforma

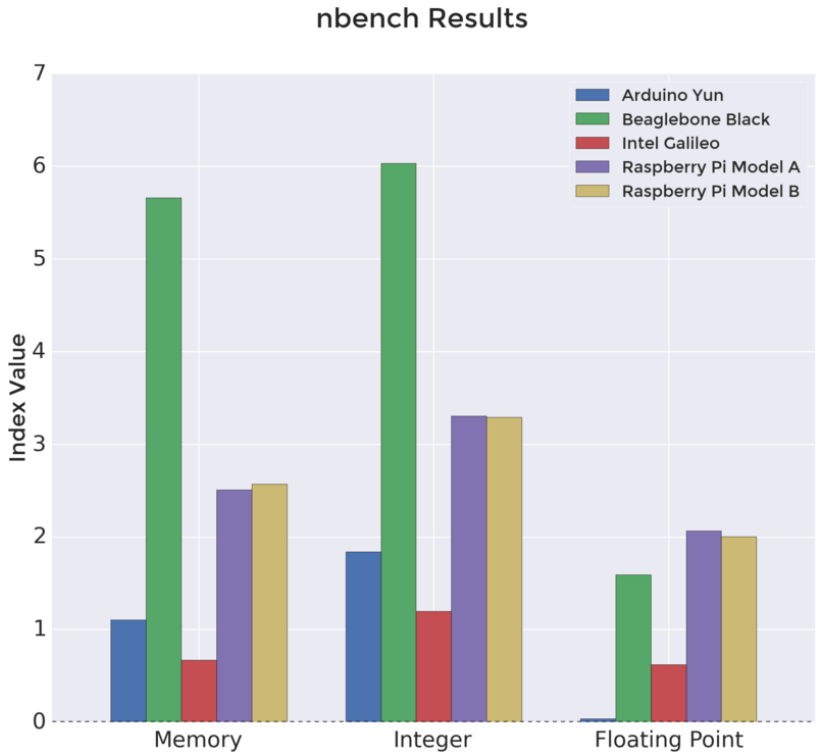
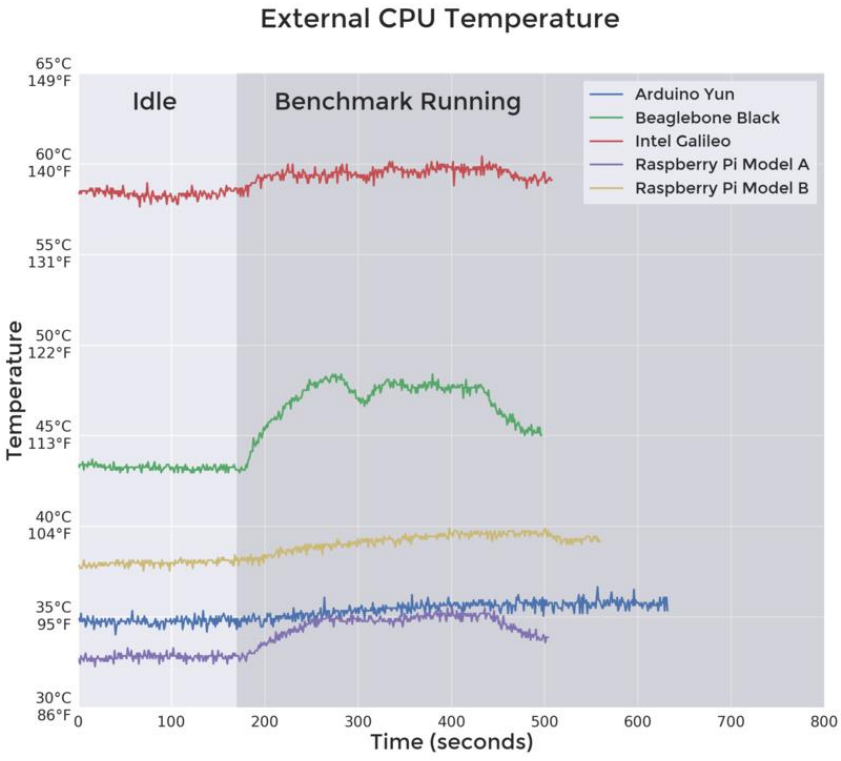


Figura 152. Gráfico comparativo de la potencia requerida para cada plataforma

Figura 153. Gráfico comparativo de la temperatura de operación para cada plataforma



A. CÓDIGO FUENTE DEL PROGRAMA DEL CONTROL DE LA BASE EN TIERRA - CONTROL.PY

```
#!/usr/bin/env python2.7
#version: 1
import RPi.GPIO as GPIO
from subprocess import call
import subprocess
from time import sleep
import time
import sys
import os
import urllib2
import thread

#Inicializacion de variables
estado_usb = "drone_log.txt"

#Verificacion de coneccion a internet
def prueba_conexion():
    try:
        urllib2.urlopen("http://www.dropbox.com")
        enable_upload = 1
        print "Conexion a internet exitosa"
    except urllib2.URLError:
        enable_upload = 0
        print "Sin conexion a internet"
    return enable_upload

#Verificacion de conexion a memoria
def prueba_usb():
    try:
        directorio_usb = os.listdir("/media/usbstick")
        if estado_usb in directorio_usb:
            enable_usb = 1
            print "Conexion USB exitosa"
        else:
            enable_usb = 0
            print "Sin conexion USB"
    except:
        print ("No se encuentra el direccion /medi/usbstick")
    return enable_usb

enable_usb = prueba_usb()
enable_internet = prueba_conexion()
if (enable_internet and enable_usb):
    print "\nEmpezando envio y descarga de dator"
    envio   =  "/home/pi/Dropbox-Uploader/dropbox_uploader.sh   upload
/media/usbstick/fotos /"
    descarga = "/home/pi/Dropbox-Uploader/dropbox_uploader.sh download
/code/update_log.txt /home/pi/"
    permiso1 = "sudo chmod u+rwx /home/pi/code/update_log.txt"
    permiso2 = "sudo chmod g+rwx /home/pi/code/update_log.txt"
    permiso3 = "sudo chmod o+rwx /home/pi/code/update_log.txt"
    try :
        call ([envio], shell=True)
```

```
        call ([descarga], shell=True)
        print "Envio y descarga exitosos"
        print "Agregando permisos"
        call ([permiso1], shell=True)
        call ([permiso2], shell=True)
        call ([permiso3], shell=True)
    except:
        print "Error de envio o descarga"
else:
    sys.exit()
```

B. ARCHIVO DE REGISTRO DE VERSION DE SOFTWARE – UPDATE_LOG.TXT

```
Version control.py:
1
```

C. CÓDIGO FUENTE DEL PROGRAMA DE ACTUALIZACIÓN DE SOFTWARE-UPDATE.PY

```
# -*- coding: utf-8 -*-
"""
Created on Fri Apr 03 20:05:26 2015

@author: Kevin Mazariegos
"""
#!/usr/bin/python
import subprocess
from subprocess import call
import sys

#Iniciaizacion de Variables
programa = "/home/pi/code/control.py"
update_log = "/home/pi/code/update_log.txt"
# Obtener la version del software de control actual
def now_version():
    try:
        print "Obteniendo version del software presente"
        print "Abriendo archivo control.py..."
        ctlog = open (programa,"r")
        print "Leyendo archivo..."
        ctlog_line = ctlog.readlines()
        ctlog_version = ctlog_line[1]
        [int(ver) for ver in ctlog_version.split() if ver.isdigit()]
        print "Version actual: "+ver
        return ver
        ctlog.close()
    except:
        print "Error obteniendo la version actual"
        sys.exit()
# Obtener la version del software ultimo
def update_version():
```

```
      try:
          print "Obteniendo version del software de actualizacion"
          print "Abriendo archivo update_log..."
          uplog = open (update_log,"r")
          print "Leyendo archivo..."
          uplog_lines = uplog.readlines()
          index_version  =  uplog_lines.index("Version  control.py:\r\n")
+1
          up_log_version = uplog_lines[index_version]
          [int(ver1)    for    ver1    in    up_log_version.split()    if
ver1.isdigit()]
          print "Version de actualizacion: "+ver1
          return ver1
          uplog.close()
      except:
          print "Error obteniendo la version de actualizacion"
          sys.exit()

  print "Actualizando software"
  version_actual = now_version()
  version_update = update_version()

  if (version_update > version_actual):
      print "Actualizando el software"
      #borrar = "sudo rm /home/pi/code/control.py"
      descarga = "/home/pi/Dropbox-Uploader/dropbox_uploader.sh download
/code/control.py /home/pi/"
      permiso1 = "sudo chmod u+rwx /home/pi/code/control.py"
      permiso2 = "sudo chmod g+rwx /home/pi/code/control.py"
      permiso3 = "sudo chmod o+rwx /home/pi/code/control.py"
      try:
          #print "Borrando antiguo software"
          #call([borrar], shell=True)
          print "Descargando nuevo software"
          call ([descarga], shell=True)
          print "Agregando permisos"
          call ([permiso1], shell=True)
          call ([permiso2], shell=True)
          call ([permiso3], shell=True)
          print "Actualizacion exitosa"
      except:
          print "Error en la actualizacion"
  else:
      print "Software ya esta actualizado"
```

```
!/usr/bin/env python

# vim:set ts=

#

# Copyright 2015 UAVenture AG.

#

# This program is free software; you can redistribute it and/or modify
```

```
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful, but
# WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY
# or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License
# for more details.
#
# You should have received a copy of the GNU General Public License along
# with this program; if not, write to the Free Software Foundation, Inc.,
# 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
# Updated: Tarek Taha : tarek.taha@kustar.ac.ae
# Adapted by Luis C. Velasquez luiscar_velasquez@hotmail.com
#    - Changed topic names after re-factoring :
https://github.com/mavlink/mavros/issues/233

import rospy
import thread
import threading
import time

from geometry_msgs.msg import PoseStamped, Quaternion
from math import *
from mavros.srv import CommandBool
from mavros.utils import *
from std_msgs.msg import Header
from std_msgs.msg import String
from tf.transformations import quaternion_from_euler

class Setpoint:
```

```
    def __init__(self, pub, rospy):
        self.pub = pub
        self.rospy = rospy

        self.x = 0.0
        self.y = 0.0
        self.z = 0.0

        try:
            thread.start_new_thread( self.navigate, () )
        except:
            print "Error: Unable to start thread"

        # TODO(simon): Clean this up.
        self.done = False
        self.done_evt = threading.Event()
        sub = rospy.Subscriber('/mavros/local_position/local',
PoseStamped, self.reached)

    def navigate(self):
        rate = self.rospy.Rate(10) # 10hz

        msg = PoseStamped()
        msg.header = Header()
        msg.header.frame_id = "base_footprint"
        msg.header.stamp = rospy.Time.now()

        while 1:
            msg.pose.position.x = self.x
            msg.pose.position.y = self.y
            msg.pose.position.z = self.z
```

```
            # For demo purposes we will lock yaw/heading to north.
            yaw_degrees = 0  # North
            yaw = radians(yaw_degrees)
            quaternion = quaternion_from_euler(0, 0, yaw)
            msg.pose.orientation = Quaternion(*quaternion)

            self.pub.publish(msg)

            rate.sleep()

    def set(self, x, y, z, delay=0, wait=True):
        self.done = False
        self.x = x
        self.y = y
        self.z = z

        if wait:
            rate = rospy.Rate(5)
            while not self.done:
                rate.sleep()

        time.sleep(delay)



    def reached(self, topic):
            #print topic.pose.position.z, self.z,
abs(topic.pose.position.z - self.z)
            if abs(topic.pose.position.x - self.x) < 0.2 and
abs(topic.pose.position.y - self.y) < 0.2 and abs(topic.pose.position.z -
self.z) < 0.2:
                self.done = True
```

```
            print "Current
Pose:",topic.pose.position.x,topic.pose.position.y,topic.pose.position.z
            print "Set Pose:",self.x,self.y,self.z
            self.done_evt.set()

def setpoint_demo():
    pub = rospy.Publisher('/mavros/setpoint_position/local', PoseStamped,
queue_size=10)

    rospy.init_node('pose', anonymous=True)
    rate = rospy.Rate(10)

    setpoint = Setpoint(pub, rospy)

    print "move in x axis 1 meter "
    setpoint.set(0.0, 1.0, 1.2, 0)



    while not rospy.is_shutdown():
      print "NOT MANUAL"



if __name__ == '__main__':
    try:
        setpoint_demo()
    except rospy.ROSInterruptException:
        pass
```