UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

CENTRO DE FÍSICA APLICADA Y TECNOLOGÍA AVANZADA

DESARROLLO DE UN SISTEMA DE PERCEPCIÓN REMOTA PARA UN NANOSATÉLITE TIPO CUBESAT

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN TECNOLOGÍA

P R E S E N T A:

DANIEL GARCÍA NÚÑEZ

DIRECTOR DE TESIS:

DR. RAFAEL GUADALUPE CHÁVEZ MORENO

Querétaro, México, 2019

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

CENTRO DE FÍSICA APLICADA Y TECNOLOGÍA AVANZADA
FACULTAD DE ESTUDIOS SUPERIORES, CUAUTITLÁN

LICENCIATURA EN TECNOLOGÍA

Votos Aprobatorios

COMITÉ ACADÉMICO DE LA
LICENCIATURA EN TECNOLOGÍA

Presente

En cumplimiento del Artículo 26 del Reglamento General de Exámenes, nos permitimos comunicar a usted que revisamos la Tesis de título

Desarrollo de un sistema de percepción remota para un nanosatélite tipo CubeSat

que realizó el (la) pasante

Daniel García Núñez

con número de cuenta: 312218830, bajo la opción de titulación por Tesis y Examen profesional en la Licenciatura en Tecnología.

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el EXAMEN PROFESIONAL correspondiente, otorgamos nuestro **VOTO APROBATORIO.**

	NOMBRE	FIRMA
PRESIDENTE	Dr. Saúl Daniel Santillán Gutiérrez	
VOCAL	Dr. Carlos Romo Fuentes	
SECRETARIO	Dr. Rafael Guadalupe Chávez Moreno	
1er. SUPLENTE	Dr. José Alberto Ramírez Aguilar	
2º SUPLENTE	M. en C. Francisco Fernández Escobar	

Atentamente
"POR MI RAZA HABLARÁ EL ESPÍRITU"

UNAM, Campus Juriquilla, Qro. a 23 de julio de 2019.

Dedicatoria

Este trabajo está dedicado a mi familia.
A mi madre, por haberme impulsado en cada etapa de mi vida, cada logro mío viene de un esfuerzo suyo. Por cada día que me despertó para ir a la escuela, por cada vez que se sentó conmigo a estudiar, por cada vez que me preguntó si ya había hecho mi tarea. Porque ella hizo posible que yo me encuentre en el lugar que estoy y me convirtiera en la persona que soy.
A mi padre, por haberme apoyado en cada decisión que he tomado y haber confiado en que alcanzaría cada meta que me propuse. Mi familia siempre esperó que algún día sería un ingeniero como él, y aunque en mi título no aparezca esa palabra, espero que mi trabajo lo represente. Sin embargo, yo seguiré esperando algún día ser, o al menos parecerme un poco, a la gran persona que es él.
A mi hermana mayor, por haber sido mi guía y ejemplo a seguir durante toda mi vida. Porque ella siempre tiene las palabras correctas para darme el valor de tomar las decisiones difíciles.
A mi hermana menor, por motivarme a ser alguien mejor. Porque no hay paso que tome sin pensar en ser el buen ejemplo a seguir que merece.
A mi abuela, por enseñarme el valor del trabajo duro y esfuerzo.
A mi familia de Querétaro, por todo su apoyo incondicional y creer en mí.
Y a toda mi familia, por acompañarme en cada paso que he dado.
Por último, también me lo dedico a mí, porque con este trabajo concluyo el esfuerzo que realicé durante mi carrera.

Agradecimientos

Agradezco especialmente al Dr. Rafael Guadalupe Chávez Moreno por haberme guiado durante un gran periodo de mi carrera, por compartir sus conocimientos conmigo y por el tiempo que le dedicó al proyecto. De igual manera, agradezco a mi comité sinodal por sus observaciones constructivas sobre mi tesis que me ayudaron a darle mayor calidad a mi trabajo. A la Unidad de Alta Tecnología de la Facultad de Ingeniería de la UNAM por prestarme sus instalaciones para realizar mi trabajo, así como al Laboratorio de Sistemas Embebidos de dicha Unidad en el que se desarrolló el circuito impreso para el prototipo. Por último, también agradezco a la Universidad Nacional Autónoma de México por haberme dado la oportunidad de realizar mis estudios de licenciatura en la que obtuve los conocimientos y aptitudes necesarias para llevar a cabo este proyecto.

Resumen

La tecnología espacial ha tomado una gran importancia en el estilo de vida de la sociedad actual. Las telecomunicaciones, los sistemas de navegación y el monitoreo climático son algunas de las actividades que dependen de sistemas espaciales. Para México, es fundamental el desarrollo de sistemas espaciales propios y de bajo costo para comenzar una independencia tecnológica y de igual manera, poder colaborar con otros países que también invierten en el sector espacial. El diseño de subsistemas satelitales es un paso adelante para poder llevar a cabo una misión espacial. En el presente proyecto, se diseñó y desarrolló un prototipo de un subsistema de percepción remota utilizando componentes de fácil acceso, para la carga útil de un nanosatélite tipo CubeSat. Adicionalmente, se implementaron algoritmos de un seguidor de estrellas en un microprocesador Raspberry Pi 3® para integrarlo con el sistema de cámara y así desempeñar la determinación de orientación satelital. Finalmente, se verificó el funcionamiento de ambos subsistemas.

Índice general

Abreviaturas

ADCS "Attitude Determination and Control Subsystem".

AEM Agencia Espacial Mexicana.

CNC Control Numérico Computarizado.

COTS "Components Off-The-Shelf".

FOV "Field Of View".

GEO "Geosynchronous Earth Orbit".

GSC "Guide Star Catalogue".

HEO "High Earth Orbit".

HYG Hipparcos, Yale & Gliese.

I2C "Inter-Integrated Circuit".

IMU "Inertial Measurement Unit".

LEO "Low Earth Orbit".

LIS "Lost In Space".

MEO "Medium Earth Orbit".

NASA "National Aeronautics and Space Administration".

OpenCV "Open Source Computer Vision Library".

SPI "Serial Peripherical Interface".

UART "Universal Asynchronous Receiver-Transmitter".

UAT Unidad de Alta Tecnología.

UNAM Universidad Nacional Autónoma de México.

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. La tecnología espacial

La tecnología espacial surgió desde hace poco más de sesenta años y sus avances han hecho posible el estilo de vida que tenemos actualmente. El primer satélite artificial, SPUTNIK 1, fue lanzado por la Unión de Repúblicas Socialistas Soviéticas (URSS), el 4 de octubre de 1957, dando inicio a la "carrera espacial" junto a Estados Unidos (EE. UU.), que consecuentemente impulsó el desarrollo de dicha tecnología [1].
Actualmente, la industria espacial es un sector globalizado en el cual participan muchos países desarrollando herramientas para generar beneficios sociales e impactar positivamente la vida diaria de las personas mediante diversos servicios. Algunos servicios como las telecomunicaciones, la observación terrestre y la navegación, son directamente dependientes del sector espacial, que también influye en actividades como la agricultura, el urbanismo y el sector defensa. Otras áreas de la tecnología como la automotriz y la médica se han beneficiado por los avances de innovación espacial [2].

1.1.1. La tecnología espacial en México

En esta sección se presentarán acontecimientos descritos en el Plan de Orbita 2.0, documento que fue creado por ProMéxico, junto con la Agencia Espacial Mexicana (AEM) [2].
El desarrollo de proyectos espaciales en México inició el mismo año que el SPUTNIK 1 fue lanzado, con el programa para el diseño y construcción de cohetes de la Escuela de Física de la Universidad Autónoma de San Luis Potosí. De dicho programa, se logró lanzar el primer cohete con fines científicos en el país, llamado *Física 1* que medía 1.7 metros de longitud y pesaba ocho kilogramos.
En los Juegos Olímpicos de 1968 en México, la tecnología espacial comenzó a generar un impacto en la vida diaria de los mexicanos al permitir las primeras transmisiones de la televisión a color, mediante los satélites INTELSAT y la estación terrena de Tulancingo, Hidalgo, la cual contaba con la antena más grande

del mundo en su tiempo. Un par de años después, México adquirió su primer sistema de satélites nombrado Sistema Morelos, así como el Centro de Control Satelital ubicado en Iztapalapa, Ciudad de México. En 1993 y 1994, fueron lanzados los satélites Solidaridad I y II, que remplazaron al sistema Morelos y fue el primer sistema en manejar tres bandas de frecuencia (Ku, C y L).
Poco tiempo después, se empezaron a desarrollar proyectos espaciales con fines científicos por diferentes instituciones académicas. La Universidad Nacional Autónoma de México (UNAM) puso en órbita su primer microsatélite UNAM-SAT B en 1996, cuya misión era realizar estudios estadísticos del impacto de meteoritos en la atmosfera.
Finalmente, con el propósito de impulsar la innovación y el desarrollo del sector espacial, el 30 de junio de 2010, se creó la AEM.

1.2. Misiones espaciales

Llevar a cabo una misión espacial de manera exitosa depende de tres aspectos principales. El primero son todas las herramientas y equipos que se encuentra en el suelo, como la base terrena, en donde se dispone del equipo con el que se comunicará con el satélite al estar en órbita. También los laboratorios y equipos necesarios para validar el funcionamiento del sistema antes de la misión espacial entran en esta clasificación. La segunda parte es el vehículo de lanzamiento que se encargará de llevar al sistema al espacio. El último aspecto, al cual se enfoca esta sección, es el satélite en órbita que realizará la función específica determinada por la misión espacial [3].

1.2.1. Estructura de los satélites

Se puede dividir a los satélites en carga útil y la nave de carga. La carga útil es el sistema encargado de cumplir el objetivo de la misión, mientras que la nave de carga se conforma por diferentes subsistemas que se aseguran de que la carga útil sea capaz de cumplir su función [3].

De acuerdo al libro de Wiley J. Larson y James R. Wertz, titulado “Análisis y diseño de una misión espacial” [4], los satélites consisten de los siguientes subsistemas:

- El **subsistema de propulsión** que impulsa al sistema para ajustar su órbita y orientación, y corrige el momento angular.
- El **subsistema de comunicación** que vincula al satélite con la base terrena o con otros sistemas en órbita.
- El **subsistema de manejo de datos** que distribuye los comandos y almacena la información de tanto de los demás subsistemas, como de la carga útil.
- El **subsistema de control térmico** que mantiene los equipos dentro de los rangos de temperatura adecuados.

- El **subsistema de potencia** que genera, almacena, regula y distribuye la energía eléctrica a los diferentes subsistemas.
- El **subsistema de determinación y orientación del satélite** que estima y controla la orientación angular del sistema. Debido a que en este trabajo se desarrolló un prototipo de este subsistema, se dedica una sección especial para su mejor descripción, la cual se encuentra más adelante.
- La estructura que soporta a los diferentes subsistemas, así como a la carga útil.

1.2.2. Clasificación de satélites

Los satélites se pueden clasificar de tres maneras principales: por el tipo de órbita, por el objetivo de la misión espacial, o por su peso.

Clasificación de satélites por tipo de órbita

De acuerdo al catálogo de órbitas terrestres de satélites publicada por H. Riebeek en *Nasa Earth Observatory* [5], se presenta la siguiente clasificación de órbitas y características de los satélites que se encuentran en ellas.

De manera general, hay tres tipos de órbita terrestre: alta, media y baja. Muchos de los satélites cuyo objetivo es obtener información sobre el clima así como los satélites de comunicación se encuentran en una órbita terrestre alta, mientras que satélites de navegación tienden a encontrarse en una órbita terrestre media. En la órbita terrestre baja se encuentran la mayoría de los satélites con propósitos científicos.

La altura, la inclinación y la excentricidad de la órbita tienen un papel importante para el funcionamiento del sistema. La altura de la órbita afecta de manera directa la velocidad con que se mueve el satélite. La gravedad de la Tierra es el factor principal para el movimiento del satélite, mientras más cerca de la Tierra se encuentre, la fuerza con la que es atraído es mayor y el satélite debe moverse más rápido para mantener su órbita sin gasto energético considerable. La inclinación de la órbita se mide con respecto al ecuador. Una órbita con inclinación de 0°implica que el sistema gira alrededor de la Tierra sobre el ecuador, mientras que en una de 90°el satélite pasaría sobre los polos geográficos de la Tierra. Con respecto a la excentricidad de la órbita, que es el parámetro que determina su geometría, un valor de excentricidad cercano a cero indicaría que la órbita tiene una forma circular, y entre más se acerca al valor de uno, su forma tendería a ser una elipse.

Para un satélite que orbita la Tierra a una altura de 42,164 km, su velocidad iguala a la velocidad de rotación de la Tierra, por lo que permanece en una sola longitud de esta última, todo el tiempo. A esta específica órbita terrestre alta (HEO por las siglas de: “High Earth Orbit”), se le denomina geo-síncrona. Los sistemas que operan en una órbita terrestre geo-síncrona (GEO por su nombre en inglés “Geosynchronous Earth Orbit”) son ideales para misiones espaciales de monitoreo de clima y telecomunicaciones.

En el rango de órbita terrestre media (MEO por las iniciales de "Medium Earth Orbit") hay dos tipos destacables: la órbita semi-síncrona y la órbita Molniya. En la órbita semi-síncrona, los satélites tardan 12 horas en completar una vuelta a la Tierra, por lo que cruzan los mismos dos puntos en el ecuador cada día. Este tipo de órbita es utilizado generalmente para los sistemas de posicionamiento global (GPS). Los satélites que se mueven en una órbita Molniya también tardan un lapso de 12 horas en rodear la Tierra, sin embargo, debido a su excentricidad, esta órbita tiene una forma elíptica, lo que permite al sistema permanecer más tiempo en uno de los hemisferios, por lo que son útiles para servicios de comunicación en latitudes lejanas al Ecuador.

Finalmente, la órbita terrestre baja (LEO por las siglas de: "Low Earth Orbit") es utilizada en su mayor parte con propósitos científicos. En este rango de altura las condiciones ambientales a las que se enfrenta el sistema son menos extremas que en las mencionadas anteriormente, por lo que permite el uso de componentes menos costosos y de fácil acceso ("Components Off-The-Shelf" que se abrevia como COTS). Entre los principales objetivos que cumplen los satélites a esta altura está la observación terrestre con diversos propósitos.

Clasificación de satélites por objetivo de misión espacial

El propósito principal de la misión espacial determina la carga útil que llevará el satélite. De manera general, existen diferentes tipos de satélites dependiendo de su función, por ejemplo, hay satélites para navegación, para comunicaciones, para monitoreo del clima, para la observación terrestre y satélites para estudios astronómicos. También es importante mencionar a la Estación Internacional Espacial, que es un laboratorio que orbita alrededor de la Tierra. Más adelante (sección 1.4) se describen las diferentes misiones espaciales y las cargas útiles específicas para cada caso.

Clasificación de satélites por magnitud de masa

Los satélites cuyo peso es mayor a una tonelada son considerados satélites grandes, mientras que los satélites medianos son aquellos que tienen un peso entre 500 y 1000 kg. Una de las tendencias en el desarrollo de sistemas espaciales es minimizar su tamaño y peso con el objetivo de reducir costos tanto de materiales y equipos utilizados para construirlo, así como por el costo de llevarlo al espacio. La Tabla 1.1 presenta la clasificación de satélites por rangos de masa cuyo peso es menor a 500 kg [6]. En ella se denomina un tipo de satélite con un rango de masa de 1 a 10 kg, que es considerado como Nanosatélite. En el presente proyecto, se desarrollaron sistemas basados en parámetros para este tipo de satélites; de manera más específica, en una estructura estándar de nanosatélites llamada CubeSat.

Tabla 1.1: Clasificación de satélites por rangos de masa.

Clasificación	**Rango de masa**
Femtosatélite	10 - 100 g
Picosatélite	0.1 - 1 kg
Nanosatélite	1 - 10 kg
Microsatélite	10 - 100 kg
Satélite pequeño	100 - 500 kg

1.3. Nanosatélites tipo CubeSat

Los satélites tipo CubeSat, se refieren a un estándar de sistemas espaciales que surgió con el propósito de hacer accesibles proyectos espaciales para la comunidad científica en las universidades. Actualmente, existen muchos programas para el desarrollo de este tipo de satélites de bajo costo en escuelas de diferentes niveles académicos alrededor del mundo, así como también hay diversas agencias gubernamentales y grupos comerciales que se han unido al desarrollo de estos [7].

Un CubeSat debe cumplir con criterios específicos sobre su forma, tamaño y peso. Esto permite la producción en masa de sus componentes y reducir el costo de su desarrollo, en comparación a nanosatélites cuya forma es por criterios particulares de sus fabricantes. Además, su forma y tamaño estandarizados permiten reducir los costos para transportarlos y ponerlos en órbita. La unidad CubeSat estándar, conocida como 1U, es un cubo de 10 cm^3 con una masa aproximada de 1 a 1.33 kg; y en general los CubeSats pueden estar formados por una o más unidades. La figura 1.1 muestra ejemplos de una 1U y 3U [7].

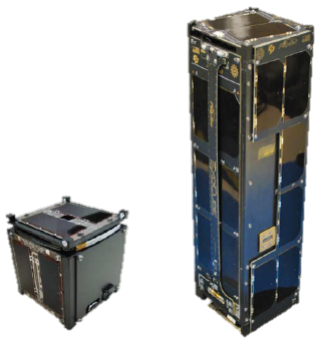


Figura 1.1: CubeSat de 1U (izquierda) y CubeSat de 3U (derecha).

1.4. Carga útil

Como se mencionó anteriormente, la carga útil es parte fundamental de un satélite, ya que es el sistema que va a interactuar con el exterior para cumplir el objetivo principal de la misión espacial. Las tareas del resto de subsistemas consisten en mantener a la carga útil funcionando de manera adecuada [4].

Una carga útil puede ser un instrumento óptico para observar la Tierra o las estrellas, un módulo de comunicación para transmitir señales de televisión, un experimento biológico, tecnología que quisiera ser probada en el espacio, etc. [3]. De acuerdo al tipo de misión espacial es como se define a la carga útil. En la Tabla 1.2 se presentan diferentes sistemas dependiendo del propósito de la misión espacial [4].

En el presente proyecto, se diseñó y desarrolló un sistema de percepción remota para un nanosatélite tipo CubeSat, por lo que a continuación se darán más detalles de este tipo de carga útil.

Tabla 1.2: Diferentes tipos de carga útil dependiendo del objetivo de la misión espacial.

Misión espacial	**Carga útil**	**Ejemplo**
Comunicaciones		
Conexión de banda ancha	Transceptor	Milstar, Intelsat
Radiodifusión	Transmisor	DirecTV, GPS
Percepción remota		
Imágenes	Cámaras	LandSat, Telescopio espacial
Medición de intensidad de energía térmica	Radiómetros	SBIRS
Mapeo de topografía	Altímetro	Observatorio de Rayos X Chandra, TOPEX/Poseidon
Navegación		
Señal de navegación	Reloj y transmisor	GPS, GLONASS
Ciencia in Situ		
Tripulado	Ciencias biológicas	Transbordador espacial, MIR
Robótica	Recolección de muestras	Mars Sojourner, LDEF
Otros		
Micro-gravedad	Planta física y materiales primarios	Transbordador espacial
Energía espacial	Recolector, convertidor y transmisor de energía solar	SPS
Utilización de recursos	Recolector y procesador de suelo lunar	Base lunar
Cementerio espacial	Contenedor de residuos	Pegasus XL

1.4.1. Sistemas de percepción remota

La percepción remota en satélites presenta un gran panorama de aplicaciones. Cualquier imagen o medición obtenida por un satélite sin tener contacto directo con el objeto en estudio es considerado percepción remota. La captura de imágenes de la superficie de la Tierra, las mediciones en la atmósfera u observar el espectro de galaxias lejanas son ejemplos de misiones de percepción remota [4].

Diversos sectores económicos y sociales son beneficiados con información obtenida por imágenes o mediciones efectuadas desde sensores en órbita. Entre ellos, la agricultura y la ganadería pueden utilizar imágenes para optimizar su producción. También se pueden ocupar para la prevención de desastres naturales mediante el monitoreo de volcanes y presas, o la medición de gases contaminantes en zonas urbanas.

La tabla 1.3 muestra la actividad espacial de diversos países en actividades de observación de la Tierra [2]. La mayoría de los países que están involucrados en el sector espacial tienen participación en este tipo de actividades, debido a sus múltiples aplicaciones. México se muestra como un país que está por comenzar, y esta es una gran oportunidad para el desarrollo de tecnología nueva en el país.

Tabla 1.3: Actividad de desarrollo de tecnología espacial para observación de la Tierra para diversos países.

Interés y desarrollo mínimo ○
Con operaciones o está por comenzar, Alianzas internacionales ◍
Completamente desarrollado y con operaciones independientes ●

País	**Actividad**	**País**	**Actividad**
Arabia Saudita	◍	Agencia Espacial Europea	●
Sudáfrica	●	Alemania	◍
Israel	◍	Reino Unido	●
Turquía	●	Rusia	●
Australia	◍	Estados Unidos	●
Malasia	●	México	◍
India	●	Brasil	◍
Indonesia	○	Argentina	◍
Japón	●	Canadá	●
China	●	Perú	○
Corea del Sur	●		

1.5. Subsistema de determinación y control de orientación satelital

Muchas misiones espaciales requieren controlar la orientación del satélite para que sus subsistemas funcionen de manera óptima. El subsistema de potencia necesita apuntar las celdas solares en dirección al sol para adquirir la mayor cantidad de energía posible, el subsistema de comunicación debe orientar las antenas a la base terrena, y también es importante que la carga útil apunte en la dirección correcta. En el caso de los satélites para la observación de la Tierra, es necesario que la carga útil apunte en dirección a la Tierra. Los satélites de comunicación requieren orientar las antenas para interactuar con otros sistemas. Para misiones astronómicas es importante que los sensores apunten hacia alguna estrella específica o al objeto de estudio en el espacio [3]. El encargado de cumplir esta tarea es el subsistema de determinación y control de orientación (ADCS por las siglas de "Attitude Determination and Control Subsystem").

Para sistemas simples, que no requieren tanta precisión, la orientación se controla usando métodos pasivos mediante la interacción con el campo magnético o gravitacional de la Tierra. Para misiones complejas, es indispensable contar con un conjunto de sensores para determinar la orientación, actuadores para modificarla y un sistema de control [4].

En general, para el control de orientación del satélite se utiliza un ciclo retroalimentado como se muestra en la figura 1.2 [3]. El ADCS se conforma por dos partes: La parte de "Determinación" estima la orientación, después se compara con el valor de referencia, y la parte de "Control" la ajusta hacia al valor deseado. También hay que tomar en cuenta las perturbaciones como el arrastre atmosférico, la presión solar y los pares de perturbación magnética, que modifican la orientación del sistema.

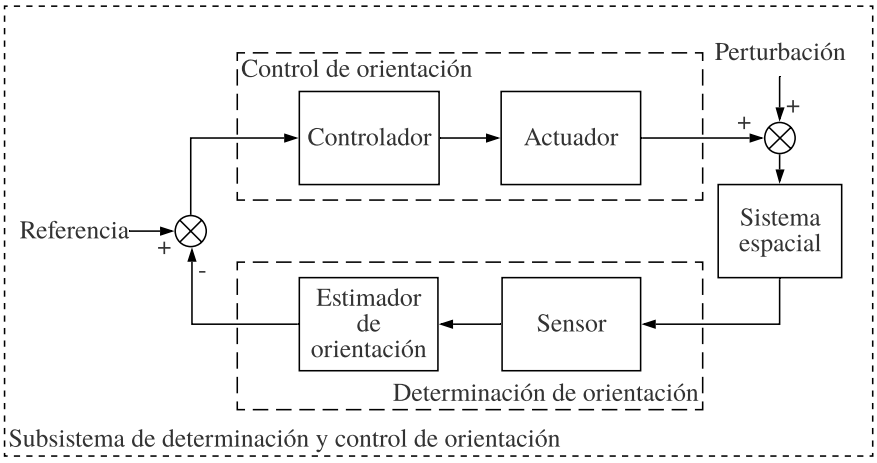


Figura 1.2: Diagrama de bloques del ADCS.

1.5.1. Determinación de orientación

Sensores de determinación de orientación

Para poder controlar la orientación del sistema, primero es necesario conocer su valor actual. Dependiendo de los requerimientos de la misión se eligen los sensores que se utilizarán para estimar la orientación.

Los sensores más comunes para determinar la orientación son los siguientes:

- El **sensor de Sol** calcula la orientación del sistema mediante sensores fotosensibles para determinar la dirección relativa del Sol al satélite y estimando la posición del Sol en el marco inercial.
- El **sensor de Tierra** de manera similar al sensor de sol, mediante sensores ópticos o infrarrojos determina la dirección relativa de la Tierra y estimando su posición en el marco inercial estima la orientación del satélite.
- El **magnetómetro** mide la dirección del campo magnético de la Tierra y utilizando modelos matemáticos del mismo determina la orientación del sistema.
- El **seguidor de estrellas** calcula la orientación del sistema mediante la captura de imágenes de las estrellas y comparando la información obtenida con una base de datos sobre la posición de las estrellas en el marco inercial.
- La **Unidad de medición inercial** (IMU por "Inertial Measurement Unit") mide la velocidad de rotación del sistema. En comparación con los demás sensores, este sólo calcula el cambio de orientación, más no la orientación absoluta del satélite.

La información obtenida por los sensores junto a un modelo del sistema es utilizada para estimar la orientación y la velocidad de rotación del sistema [3]. La Tabla 1.4 presenta la precisión potencial teórica de acuerdo a la referencia utilizada por los diferentes sensores [8].

Tabla 1.4: Precisión teórica de acuerdo al objeto de referencia de diversos sensores.

Objeto de referencia	**Precisión**
Magnetómetro	30 arco minutos
Tierra (Horizonte)	6 arco minutos
Radiofaro	1 arco minuto
Sol	1 arco minuto
Estrellas	1 arco segundo

Seguidor de estrellas

Los seguidores de estrellas determinan la orientación absoluta del sistema, lo que los pone en ventaja sobre otros como los IMU. Por otro lado, el sensor de Sol, de Tierra o el magnetómetro, también son capaces de determinar la orientación absoluta del sistema, sin embargo, son necesarios al menos dos de estos sensores para lograrlo [3]. El seguidor de estrellas toma ventaja de la gran cantidad de estrellas visibles en la esfera celeste para obtener el mayor número de vectores de referencia para determinar la orientación, lo que permite que pueda calcular el valor absoluto de orientación sin necesidad de otro sensor.

1.5.2. Control de orientación

Conociendo la orientación actual del sistema, el siguiente paso es modificarla mediante actuadores a la posición requerida. A continuación, se describen los actuadores más comunes utilizados por los satélites para llevar a cabo esta tarea [3]:

- **Torcas magnéticas:** son solenoides que generan un campo magnético que interactúa con el de la Tierra para crear una torca.
- **Ruedas de reacción:** son un tipo de actuador eléctrico que provee una fuerza resultante de la aceleración de una rueda de inercia. Debido a la conservación del momento angular, cuando la rueda de reacción gira en una dirección, el sistema girará en el sentido contrario.
- **Propulsores:** se puede generar una torca al encender al menos dos propulsores con direcciones opuestas entre sí y ambas perpendículares al eje de rotación, aplicando las fuerzas en puntos simétricos en torno al mismo. De esta manera, no habrá translación, sólo rotación.

1.6. Definición del problema

El sector espacial es una industria en constante crecimiento debido al impacto social y económico que ha generado en las últimas décadas. Sin embargo, adquirir tecnología para llevar a cabo una misión espacial es costoso, así como ponerla en órbita. Por otra parte, el camino para desarrollar tecnología propia está lleno de obstáculos. El primero de ellos es el alto costo de conseguir componentes con herencia de vuelo para la fabricación de los subsistemas, seguido de la complejidad del desarrollo del *software* para los subsistemas, que además debe ser confiable para alcanzar el objetivo de la misión espacial.

Es importante recordar que para la tecnología espacial, a diferencia de algunos otros campos de ingeniería, se deben cumplir criterios más estrictos de calidad, debido a que el cohete despega para llevar al sistema a su órbita y no es posible realizar cambios al satélite, ni repararlo en caso de que falle. También se deben contemplar las condiciones ambientales extremas a las que se enfrentará el satélite al estar en funcionamiento, tales como la radiación cósmica, los frecuentes y grandes cambios de temperatura y los esfuerzos mecánicos que esto genera, aparte de la basura espacial que se encuentra actualmente orbitando al planeta. Por lo anterior, entre otros, se puede afirmar que la planeación de la misión espacial, el diseño de cada uno de los subsistemas y el desarrollo de prototipos son pasos cruciales para cumplir el objetivo principal del proyecto y alcanzar los resultados esperados.

1.7. Justificación del trabajo

Son múltiples las aplicaciones para un sistema de percepción remota que pueden beneficiar a actividades como la agricultura o la ganadería, así como mediciones ambientales que pueden generar impacto en la sociedad actual. El diseño y desarrollo de un prototipo de sistema de cámaras es una primera aproximación para poder desarrollar diversos sistemas con componentes específicos a cada misión. Aparte, lograr desarrollar un sistema que soporte las condiciones ambientales existentes en una órbita baja abre la posibilidad a múltiples entidades académicas a desarrollar tecnología espacial propia.

Por otro lado, la implementación de algoritmos para un seguidor de estrellas que sean muy confiables y de baja complejidad computacional, permite la reducción del costo de componentes para su fabricación. Al reducir el costo de desarrollo del seguidor de estrellas, se amplia la variedad de misiones espaciales en las que puede participar como el subsistema de determinación de orientación.

Finalmente, debe tomarse en cuenta tanto el costo para enviar los equipos al espacio, así como la creciente cantidad de basura espacial que se ha generado por la explotación de dicho sector, por lo que el desarrollo de sistemas espaciales pequeños y con dimensiones estandarizadas es un factor relevante en la planificación de cualquier misión espacial.

1.8. Objetivos

El objetivo general de este trabajo consistió en diseñar y desarrollar un prototipo de un sistema de percepción remota para la carga útil de un nanosatélite estándar tipo CubeSat, utilizando componentes COTS. Además, se implementaron y se validaron algoritmos de un seguidor de estrellas en un microprocesador para integrarlos como un sistema de determinación de orientación.

Lo anterior se lista a continuación tanto como desarrollo cronológico de las actividades al respecto, como para especificar los objetivos particulares que se cubrieron en cada etapa:

1. Diseñar un sistema de percepción remota utilizando componentes COTS y que cumpla con características estándar de un nanosatélite tipo CubeSat.

2. Construir un prototipo del sistema de percepción remota en una tableta de pruebas para validar su funcionamiento.

3. Desarrollar un prototipo del sistema de percepción remota en una tableta de circuito impreso en el laboratorio de sistemas embebidos de la UAT.

4. Generar una base de datos de estrellas e implementar algoritmos necesarios para el funcionamiento de un seguidor de estrellas.

5. Integrar los algoritmos del seguidor de estrellas en un microprocesador Raspberry Pi 3®.

6. Realizar pruebas de funcionamiento del prototipo de sistema de determinación de orientación y analizar los resultados obtenidos.

1.9. Hipótesis

Es posible desarrollar sistemas para nanosatélites tipo CubeSat como la carga útil o un seguidor de estrellas utilizando componentes COTS, que soporten las condiciones ambientales existentes en una órbita LEO y con un desempeño similar a los sistemas que se encuentran actualmente en el mercado.

Capítulo 2

Antecedentes

2.1. Sistemas de percepción remota

Las mediciones realizadas mediante percepción remota han encontrado múltiples aplicaciones que generan un gran impacto en la sociedad actual. Una de las aplicaciones más comunes para los satélites de percepción remota es la evaluación de la calidad del aire. Para ello, diferentes sensores y algoritmos son utilizados para medir la concentración de partículas de aerosol de diferentes compuestos [9] [10]. Uno de los objetivos más comunes de estas misiones espaciales es medir la concentración de partículas PM_2,5 (partículas cuyo diámetro es menor a 2.5 μm) en ciudades con una gran densidad poblacional, debido a que estas partículas pueden causar daño a la salud de las personas [11] [12].

De igual manera, los sistemas de percepción remota también sirven para medir propiedades del suelo, o del agua en el mar, ríos o lagos [13]. Se han realizado misiones para monitorear la humedad del suelo [14] [15], evaluar la erosión causada por el agua [16] [17] y medir el crecimiento de vegetación en volúmenes de agua [18].

La agricultura es uno de los sectores que más se ha beneficiado de la percepción remota. También se han desarrollado proyectos para analizar el área cubierta por ciertos hábitats naturales y los recursos como su flora y fauna que se encuentran en ellos [19].

Del mismo modo, la información obtenida mediante percepción remota puede ser utilizada por diversos modelos matemáticos para alcanzar resultados más específicos, como modelos climáticos [20] o para predecir posibles desastres naturales como erupción de volcanes, terremotos e inundaciones [21] [22]. Incluso sectores como la arqueología se han beneficiado con las mediciones realizadas por estos sistemas [23].

2.2. Seguidores de estrellas

El avance en la tecnología espacial y la necesidad de obtener información más confiable en un menor tiempo, ha aumentado el interés en el desarrollo de sistemas de determinación de orientación con una mayor precisión [8]. Ejemplos claros de esto fueron los experimentos para los cohetes SHEFEX 1 y 2. Para el segundo cohete se requería información más precisa sobre la orientación inercial del sistema (resolución de 0.17 grados), por lo que se optó por utilizar un sistema híbrido de navegación basado en un IMU, un GPS y un seguidor de estrellas [24].

El funcionamiento de un seguidor de estrellas se puede dividir en dos modos de operación. El primer modo de operación ocurre cuando el satélite entra en órbita y no conoce su orientación actual, a este modo se le conoce como "perdido en el espacio" (LIS por "Lost In Space"). En este modo, el algoritmo de identificación de estrellas necesita más tiempo para lograr su objetivo y posteriormente determinar la orientación del sistema. Después de conocer información del estado actual del sistema, inicia el otro modo de operación, conocido como modo de seguimiento, en el cual, la estimación de orientación es más sencilla y rápida [25].

El campo de visión (FOV por "Field Of View") es uno de los parámetros más importantes para la precisión del seguidor de estrellas. La selección de la cámara es fundamental para el diseño del seguidor de estrellas, sin embargo, muchas veces es determinada por otros factores [26]. Por ejemplo, en este proyecto, al utilizar la cámara de la carga útil, entonces esta será definida por los requerimientos de la misión. Otro parámetro importante, es el número de estrellas en la base de datos con la que se compararán las imágenes adquiridas.

2.2.1. Catálogos de estrellas

Un catálogo de estrellas es una base de datos que agrupa la posición y propiedades específicas de cada una de ellas. Entre la información que se puede encontrar está la magnitud visual aparente de las estrellas, que es una medida del brillo de la estrella observado desde la Tierra en el espectro visible (entre más brillante sea el objeto menor será el valor de magnitud visual), o la magnitud absoluta que es la magnitud visual aparente a una distancia de 10 parsecs (32.616 años luz), entre otros. También se puede encontrar su ubicación relativa en el espacio observada desde la Tierra en coordenadas cartesianas (x, y, z), o en coordenadas esféricas (la distancia a la Tierra, la ascensión recta y la declinación). La ascensión recta es el ángulo formado entre la proyección de la estrella en el plano XY (con el eje Z hacía el norte celeste) y el eje X en dirección al equinoccio de verano de J2000 (época juliana del 1^{ro} de enero del 2000 a las 12:00 TT), y la declinación es el ángulo formado entre la dirección de la estrella y el Ecuador terrestre en la misma época.

Uno de los principales retos al desarrollar un seguidor de estrellas es seleccionar a las estrellas del catálogo para formar la base de datos óptima para cada misión espacial, a esta base de datos se le llama catálogo de estrellas guía

(GSC por "Guide Star Catalogue"). El número de estrellas del GSC y su distribución influye de manera directa a la precisión y confiabilidad del sistema. Adicionalmente, el GSC también se relaciona con la memoria de la computadora, la velocidad del algoritmo de identificación y, en caso de ser un sistema independiente, en el diseño de la óptica del seguidor de estrellas [27].

Para el desarrollo de un GSC se deben tomar en cuenta dos puntos importantes: que el número de estrellas para cualquier FOV posible en la esfera celeste sea mínimo, pero suficiente para ejecutar el algoritmo de identificación de patrones y que la distribución de las estrellas sea uniforme. Un GSC óptimo puede lograr que el seguidor de estrellas funcione de manera eficiente, confiable y continua, por lo que podría reducir costos de desarrollo del seguidor de estrellas [27].

Por lo común, se filtra al catálogo de estrellas con base a la magnitud visual aparente. Estrellas igual o más brillantes que un umbral determinado son seleccionadas como estrellas guía para el algoritmo de identificación de patrones [28]. En el 2017, la NASA patentó un *software* para un seguidor de estrellas de bajo costo que utiliza componentes tipo COTS y el catálogo de estrellas *Hipparcos* (que incluye 118,219 estrellas), además de un filtro de magnitud visual de 6.5 [29].

La tabla 2.1 muestra los catálogos de estrellas utilizados en diferentes proyectos y la magnitud visual máxima utilizada para filtrar la base de datos.

Tabla 2.1: Catálogos de estrellas utilizados en diferentes proyectos.

Referencia	Catálogo de estrellas	Magnitud Visual
[30]	SAO J2000	<5
[31]	SAO J2000	<6
[32]	SAO J2000	<6
[24]	Hipparcos	<4.6
[3]	Hipparcos	<5.3
[29]	Hipparcos	<6.5
[25]	HYG	<5

2.2.2. Algoritmos de los seguidores de estrellas

El *software* del seguidor de estrellas está formado por tres diferentes algoritmos: el algoritmo de centroide, el de identificación de estrellas y el de determinación de orientación. El algoritmo de centroide es responsable de detectar el píxel en el centro de cada estrella del campo de visión. El algoritmo de identificación de objetos reconoce a las estrellas del FOV mediante el catálogo de estrellas. Una vez identificadas las estrellas del FOV, es posible determinar la orientación del sistema [25].

La esencia del seguidor de estrellas es la identificación de estas en la base de datos y existen diversos algoritmos que han sido utilizados para realizar esta tarea. De manera general, se pueden categorizar en las técnicas basadas en geometría y las que se basan en reconocimiento de patrones. En los méto-

dos geométricos, se utiliza información sobre las estrellas en el FOV como las distancias entre ellas, los ángulos que forman o el área entre un grupo de estrellas. Para la segunda categoría de algoritmos, como su nombre lo dice, utiliza métodos de reconocimiento de patrones para identificar a las estrellas [31].

Algoritmos de identificación de estrellas basados en geometría

En 1978, Gottlieb usó el ángulo entre dos estrellas del FOV para identificarlas en una base de datos [33]. Posteriormente, Groth sugirió un algoritmo que necesitaba al menos tres estrellas, con las que formaba un triángulo cuya geometría comparaba para reconocer dicho patrón [34]. En el 2004, Mortari presentó un algoritmo en el que formaba una pirámide utilizando cuatro estrellas del FOV, y aunque era más confiable, requería una base de datos más grande por lo que los algoritmos de búsqueda como el binario ya no eran suficientes para lograr el desempeño esperado, por lo que aplicó otro método de búsqueda de índice llamado vector$-k$. El algoritmo de pirámide con el buscador de vector$-k$ ha sido utilizado y modificado para diversos proyectos [25].

M. Samaan y S. Theil utilizaron su versión del algoritmo de pirámide con el buscador de vector$-k$ para la misión SHEFEX. La idea general de su algoritmo fue identificar 3 estrellas del FOV que formen un triángulo, posteriormente tratar de identificar otra estrella que servirá para validar el resultado, ambos pasos requieren buscar en la base de datos con el Vector$-k$. El proceso inicia con la localización de los centroides de las estrellas en el FOV y determinar si hay al menos 4 estrellas. Se forma una pirámide y se busca con el Vector$-k$ las coincidencias en la base de datos de las caras de la figura. El primer triángulo encontrado se utiliza como base, y se toma la estrella restante como pivote. Se calculan los 3 ángulos formados con la estrella pivote y se compara de nuevo con la base de datos utilizando el buscador del Vector$-k$. En caso de que se confirme el resultado, la información pasa al algoritmo de determinación de orientación [24].

M. Shayan et al. (2015) presentaron un método en donde dependiendo del número de estrellas en el FOV, se decidía el algoritmo a usar. Si se encontraban al menos 4 estrellas se utilizaba el algoritmo de pirámide, y en caso de ser 3 estrellas se aplicaría el algoritmo de triángulo [25]. Aparte redujeron el número de estrellas de la base de datos utilizando el algoritmo presentado por Liebe [26]. Con este método se obtuvo el algoritmo con un tiempo de ejecución menor en comparación al algoritmo de triángulo o pirámide por separado, con una efectividad de 94 % (casi el triple que el algoritmo de Liebe).

Algoritmos de identificación de estrellas basados en patrones

Esta categoría inició con el desarrollo del método de cuadrícula por C. Padgett et al. en 1997 [35], que se basó en segmentar la imagen para obtener un patrón de bits. En 2007, H. Lee hizo una variación del algoritmo, al cual llamó de cuadrícula polar [36]. Un par de años después, Na presentó su versión del algoritmo filtrando ruido de las imágenes de entrada, método al que nombró como

cuadrícula elástica gris [37]. Los métodos mencionados anteriormente se enfocan principalmente en la identificación de patrones de las imágenes capturadas, más no en la base de datos [30].

M. D. Pham et al. propuso utilizar una base de datos optimizada en estructura de árbol y aplicando búsqueda en paralelo para mejorar la velocidad del algoritmo. Su método redujo 50 % el tiempo de ejecución en comparación a los métodos antes mencionados[30].

Aunque las técnicas de reconocimiento de patrones mencionadas anteriormente permiten una gran capacidad de precisión al identificar las estrellas en condiciones de operación ideales, muchas de ellas tienen problemas cuando la imagen capturada no es suficientemente clara y tiene variaciones con la información de la base de datos [32]. Esta variación en la imagen puede ser ocasionada por la incertidumbre del sensor óptico en el espacio, la desviación en la posición de las estrellas o estrellas falsas presentes en la imagen capturada.

M. D. Samirbhai et al. buscó resolver el problema utilizando primero una identificación basada en la distancia y ángulo de los cúmulos de estrellas en el FOV, seguido de un proceso de reconocimiento por patrones. Esta técnica mostró ser confiable ante estrellas faltantes o desviación de su posición entre el FOV y la base de datos [31].

Otro método por resaltar es el reconocimiento de patrones basado en la correlación de Separman implementada por D. S. Mehta y S. Chen. Su algoritmo de identificación de estrellas mostró ser confiable ante perturbaciones en la imagen del FOV como estrellas faltantes en comparación a la base de datos, o estrellas falsas detectadas en el FOV. En este articulo consideran que la mejor manera para medir la relación entre el FOV y el catálogo de estrellas es calculando el coeficiente de correlación de Spearman debido a que no asume una distribución normal de datos [32].

T. Delabie et al. presentaron un algoritmo basado en la técnica de procesamiento de imágenes: la transformada de distancia menor. Este método resultó ser confiable a distorsión en la posición de las estrellas en la imagen. La ventaja de este algoritmo sobre otros métodos de reconocimiento de patrones es el bajo costo computacional para el modo de operación de LIS [3]. Esta fue la razón principal por la que se decidió implementar este algoritmo para el presente trabajo.

Capítulo 3

Metodología

3.1. Componentes electrónicos

Para el desarrollo del sistema de percepción remota se utilizó un microcontrolador PIC24EP128MC202 que cuenta con todos los protocolos de comunicación requeridos. Se utilizó un sensor óptico modelo LS_Y201 y una memoria de almacenamiento microSD con capacidad de 4 Gb.

Los algoritmos del seguidor de estrellas fueron implementados en un microprocesador Raspberry Pi 3®, que posteriormente, se comunicó con el sistema de percepción remota para obtener las imágenes de entrada para el algoritmo de identificación de estrellas.

La tabla 3.1 presenta los diferentes componentes electrónicos que se utilizaron para el desarrollo del sistema de percepción remota y para el ADCS.

Tabla 3.1: Componentes electrónicos utilizados para el desarrollo e implemtanción de los subsistemas

Subsistema	Componente	Descripción	Valor	Cantidad
Percepción Remota	PIC24EP128MC202	Microcontrolador (DIP)		1
	LS-Y201	Cámara		1
	MicroSD	Memoria de almacenamiento	4Gb	1
	Módulo MicroSD Catalex			1
	Cristal	Oscilador	20MHz	1
	Resistor		$1k\Omega$	1
			$1{,}8k\Omega$	6
			$3{,}3k\Omega$	6
	Capacitor	Tantalio	$1\mu F$	1
		Cerámico	33pF	2
			$0.1\mu F$	1
			$0{,}33\mu F$	1
			$1\mu F$	2
	L78L0	Regulador de voltaje (DIP)		1
	Botón	Tipo push		1
	Placa fenólica	Una cara, 10 cm x 10 cm		1
	104 pines	Bus de comunicación		1
Determinación de orientación	Raspberry Pi 3® modelo B	Microprocesador		1
	MicroSD	Memoria de almacenamiento	8Gb	1

3.2. Equipos

La tableta de circuito impreso del sistema de percepción remota fue desarrollada en el Laboratorio de Sistemas Embebidos de la Unidad de Alta Tecnología (UAT) de la Facultad de Ingeniería de la UNAM, campus Juriquilla. Se utilizaron diferentes equipos como un Control Numérico Computarizado (CNC), una laminadora y una unidad de exposición a luz ultravioleta de la marca *Bungard Elektronik*.

Para llevar a cabo el corte del borde de la placa fenólica se utilizó el CNC y las medidas estándar para un nanosatélite tipo CubeSat de acuerdo a *"Pumpkin Space Systems"* [38]. Posterior a esto, se adhirió una película fotosensible a la placa fenólica utilizando la laminadora. Después se imprimió el diseño de las trazas en un acetato y se colocó encima de la placa fenólica para su exposición

a la luz ultravioleta.

El siguiente paso fue quitar la película fotosensible que no fue expuesta a la luz ultravioleta y realizar el ataque químico a la placa fenólica para eliminar el cobre no deseado. Finalmente, se colocó una segunda película para proteger la mayoría de las trazas de cobre útiles, dejando descubiertas sólo las terminales de las componentes para su soldadura posterior. Los diseños de la tableta de circuito impreso se muestran en la sección 4.1.2.

3.3. Protocolos de comunicación

Los protocolos de comunicación que se utilizaron fueron los que se presentan a continuación:

- **"Universal Asynchronous Receiver-Transmitter" (UART)** Comunicación entre el microcontrolador del sistema de percepción remota y la cámara para la captura de fotografías y el control de los parámetros de la imagen.
- **"Serial Peripherical Interface" (SPI)** Comunicación entre el microcontrolador del sistema de percepción remota y la memoria microSD para el almacenamiento de imágenes.
- **"Inter-Integrated Circuit" (i^2c)** Comunicación entre el microcontrolador del sistema de percepción remota y la computadora de abordo o para el caso del ADCS, con el microprocesador Raspberry Pi 3®.

La figura 3.1 muestra un diagrama de bloques de la comunicación utilizada en los subsistemas de percepción remota y de determinación de orientación.

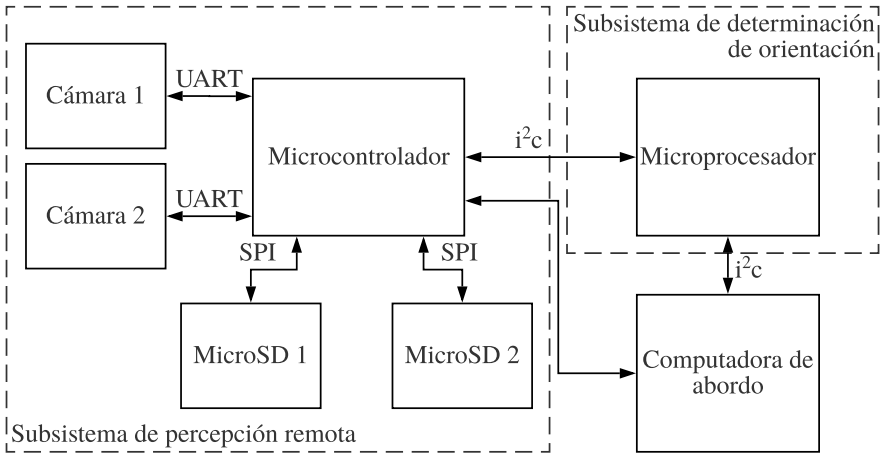


Figura 3.1: Diagrama de bloques de los protocolos de comunicación para los subsistemas de percepción remota y de determinación de orientación.

3.4. Base de datos de patrón de estrellas

En el presente trabajo se utilizó el catálogo de estrellas HYG (versión 3.0), formado por los catálogos Hipparcos, Yale Bright Star (5^{ta} edición) y el catálogo de estrellas cercanas de Gliese (3^{ra} edición), que contiene alrededor de 120,000 estrellas [39].

Para filtrar el catálogo de estrellas y localizar las estrellas guía se realizaron los siguientes pasos:

1. Con base en el valor de magnitud visual aparente, se filtró el catálogo de estrellas con un valor máximo de 6.0. Adicionalmente, se eliminó al Sol del catálogo debido al ruido que ocasionaba al algoritmo.

2. Se graficó la esfera celeste utilizando el valor de magnitud visual aparente para definir el tamaño de cada estrella, así como la declinación y la ascensión recta para seleccionar la posición de cada una.

3. Se seccionó la gráfica de la esfera celeste en partes de 20°x 20°para generar imágenes simulando el campo de visión del seguidor de estrellas.

4. Posteriormente se procesaron las imágenes con la transformada de distancia menor para poder utilizar el algoritmo de identificación de estrellas.

5. Finalmente, se analizaron las imágenes para obtener información indispensable y así simplificar al algoritmo de identificación de estrellas. Se generó un archivo que incluye el número de estrellas de cada imagen, la ubicación de las 3 estrellas con mayor tamaño en la imagen, la ubicación del centroide del triángulo que forman dichas estrellas y la orientación del triángulo. La importancia de estos parámetros será explicada con detalle más adelante, cuando se explique al algoritmo de identificación de estrellas que se implementó.

3.4.1. Transformada de distancia menor

El algoritmo utilizado para identificar las estrellas detectadas en el FOV fue basado en el método de la transformada de distancia menor sugerido en la referencia [3]. Para analizar el funcionamiento del algoritmo, se generó una base de datos de imágenes procesadas con esta técnica.

La transformada de distancia menor es un método matemático usado para el procesamiento de imágenes. La idea principal es que para cada punto de un conjunto Ω, se calcula la distancia mínima al subconjunto Ω^c, expresado en la ecuación 3.1.

$$D(p) = \text{mín}\left(d(p,q) \mid q \epsilon \Omega^c\right) \tag{3.1}$$

En esta ecuación, p es el punto al cual se aplica la transformada de distancia y q los puntos que pertenecen al subconjunto Ω^c. Para el seguidor de estrellas, p representa cada pixel en el FOV, mientras que q representa al pixel que se encuentra en el centroide de cada estrella detectada.

La imagen obtenida tras realizar la transformada de distancia sobre ella contiene en cada pixel el valor de la distancia al centro de la estrella más cercana. Esta distancia es calculada por la distancia Euclidiana al cuadrado, expresada en la ecuación 3.2.

$$d(p,q) = (p_x - q_x)^2 + (p_y - q_y)^2 \tag{3.2}$$

La ventaja de utilizar esta distancia para el funcionamiento del algoritmo es que es una distancia simétrica, por lo que es útil para comparar imágenes que han sido rotadas y trasladadas. Aparte, al utilizar el cuadrado de la distancia Euclidiana, todos los valores resultantes son enteros positivos por lo que se ejecuta en un tiempo menor y sin perder información.

3.5. Algoritmos para el seguidor de estrellas

3.5.1. Algoritmo de centroide

El algoritmo de centroide es el primer paso para identificar a las estrellas en el FOV. El objetivo es encontrar objetos en el FOV, determinar cuáles serán considerados cómo estrellas y encontrar la ubicación del centro geométrico de cada uno. Una vez localizados los centroides, se envía la información al algoritmo de identificación de estrellas.

Para la adquisición y procesamiento de imágenes de este proyecto se utilizó la biblioteca de "Open Source Computer Vision Library" (OpenCV) [40] para el lenguaje de cómputo Python 3. El algoritmo de centroide implementado se basó en la función *SimpleBlobDetector* de la biblioteca. Se eligió esta función debido a que permite seleccionar los parámetros para filtrar los objetos en el FOV de manera sencilla, sin embargo, no es el algoritmo de detección de objetos que tiene menor tiempo de ejecución de la biblioteca.

El algoritmo "*Simple Blob Detector*" utilizado se puede traducir como detector de manchas simple, esto quiere decir que va a identificar cualquier figura regular o irregular en la imagen. El algoritmo inicia utilizando un rango de umbral para segmentar la imagen en varias imágenes binarias. Después el algoritmo agrupa los pixeles blancos en una sola mancha, y prosigue a calcular su centro geométrico. Finalmente, los objetos cuya distancia entre ellos sea menor a un valor definido son unidos en uno solo y el centro de este es calculado de nuevo. La tabla 3.2 especifica los valores de parámetros utilizados para filtrar los objetos en la imagen. En ella se puede observar el umbral mínimo y máximo que se utilizaron como filtro en las imágenes, el área mínima del objeto (número de pixeles), características de la forma y la distancia mínima entre los objetos.

Tabla 3.2: Parámetros utilizados como filtros para el detector de objetos en la imagen.

Parámetro	**Valor**
Umbral	127 - 255
Tamaño	8
Convexidad	0.87
Proporción inercial	0.01
Distancia mínima entre manchas	4

Después de obtener una lista con las coordenadas en la imagen del centro de las estrellas, se ordenan por tamaño, de mayor a menor, como salida del algoritmo. Sin embargo, debe tomarse en cuenta el caso de tener más de una estrella con el mismo tamaño, en el que se toma como criterio diferenciador la menor distancia a la estrella de mayor tamaño, o en caso de que todas las estrellas tengan el mismo tamaño, se seleccionan las estrellas cuya suma de distancias entre ellas sea la menor. Este último paso fue necesario para el algoritmo de reconocimiento de patrones que se desarrolló para este seguidor de estrellas.

3.5.2. Algoritmo de identificación de estrellas

Después de que el algoritmo de centroide obtiene la ubicación del pixel en el centro de las tres estrellas más brillantes dentro del FOV, esta información entra al algoritmo de reconocimiento de patrones que se encarga de identificar a las estrellas en la base de datos. Se debe tomar en cuenta que, por lo general, el satélite se encontrará en un sistema de referencia diferente al del catálogo de estrellas utilizado, por lo que las imágenes capturadas por el seguidor de estrellas no se encontrarán alineadas con las de la base de datos. En la mayoría de los casos, el centro de ambas imágenes no coincidirá, además, existirá una inclinación entre ellas.

Para emparejar las imágenes se utilizó el método de centroide presentado en la referencia [3], en el que el centroide del triángulo formado por las tres estrellas más brillantes del campo de visión es utilizado para alinear el centro de la imagen capturada con las imágenes en la base de datos. Para alinear la inclinación entre ambas imágenes, este método utiliza el ángulo más pequeño entre las tres estrellas detectadas. La figura 3.2 explica el proceso para alinear la imagen.

El proceso del método de emparejamiento inicia con calcular los vectores entre las tres estrellas más brillantes del FOV y determinar los ángulos formados entre dichos vectores. Después se determina la posición del centroide del triángulo formado por las tres estrellas. También se determina cuál es el ángulo más agudo del triángulo y se genera un vector que inicia en el centroide del triángulo y con dirección al centro del ángulo menor. Este mismo proceso es llevado a cabo para todas las imágenes en la base de datos y la información obtenida es precargada para agilizar al algoritmo. Finalmente, se traslada el centroide del triángulo del FOV a la posición del centroide del triángulo de la

imagen de la base de datos con la que se va a comparar y se rota para que los vectores del ángulo menor coincidan.

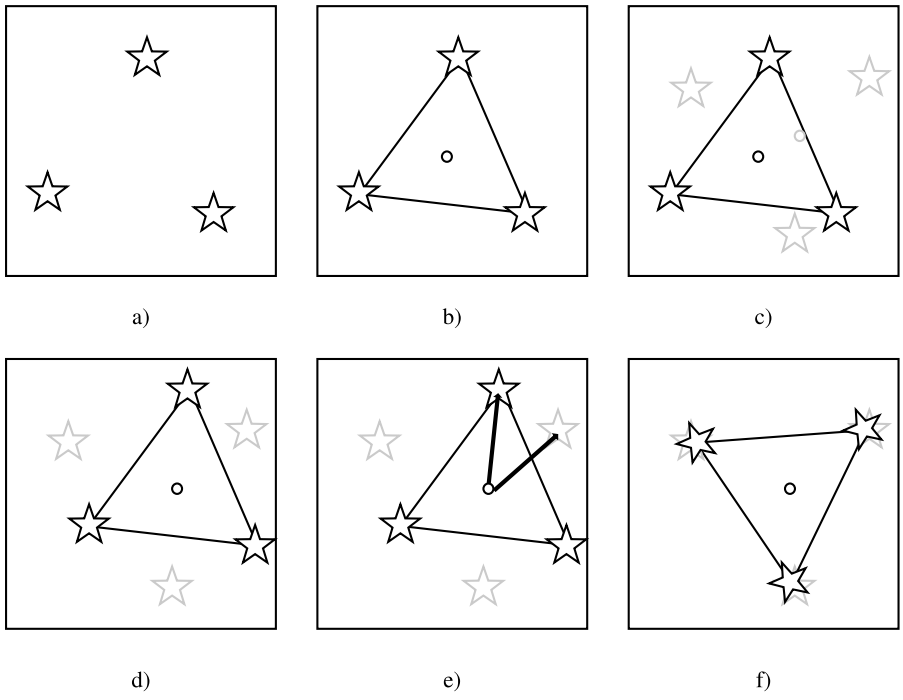


Figura 3.2: Método de centroide para emparejar las imágenes.

Cuando ambas imágenes coinciden, se comparan utilizando la transformada de distancia menor como se muestra en la figura 3.3. Se obtiene un valor para cada una de las estrellas y se almacena. En el caso del ejemplo mostrado, la estrella de arriba tendría un valor de cuatro, la que se encuentra debajo a ella tres y la de su derecha dos. El criterio de aceptación del algoritmo determina la probabilidad de que el resultado sea correcto. Si tal probabilidad es aceptable, esta parte del proceso termina; en su defecto, el sistema prosigue a emparejar la siguiente imagen de la base de datos y así sucesivamente hasta obtener una probabilidad confiable.

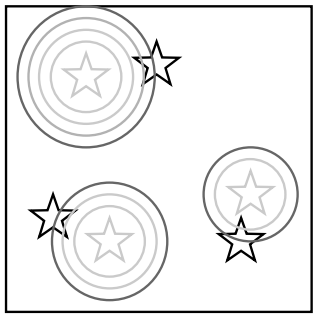


Figura 3.3: Comparación de imagen de la base de datos con el FOV mediante la transformada de distancia menor.

3.5.3. Determinación de orientación

Una vez identificadas las estrellas en el campo de visión, se utiliza el ángulo de rotación y la distancia de traslación de la imagen del FOV al aplicar el método de centroide para emparejar las imágenes en el algoritmo de identificación. También se tiene el valor de ascensión recta y declinación del centro de la imagen de la base de datos que coincidió con el FOV. Al trasladar dicho valor en sentido contrario al que fue trasladada la imagen en primer lugar, se obtiene el valor de ascensión recta y declinación al que apunta el sensor del seguidor de estrellas, mientras que el ángulo de rotación se puede respresentar como el ángulo de alabeo.

3.5.4. Algoritmo de ordenamiento

Para el modo de operación LIS, se utilizó un algoritmo de ordenamiento para ordenar la base de datos de acuerdo al número de estrellas de cada imagen, de esta manera, el algoritmo de identificación de patrones comenzaba comparando con imágenes cuyo número de estrellas fuera similar a las detectadas en el FOV.

Después de tener información sobre la orientación del satélite, se inicia el modo de operación de seguimiento. En este método se reorganiza el orden de búsqueda en la base de datos. Para este proyecto, se buscaron en base a los valores más cercanos a la orientación estimada. Este orden de búsqueda fue determinado por el tamaño del FOV utilizado al generar la base de datos y precargado al seguidor de estrellas. La búsqueda inicia siempre desde el centro de la matriz, por lo que la tarea de este sistema, antes de iniciar el algoritmo de identificación de estrellas, es recorrer los elementos de la base de datos para que el último valor de orientación estimado se encuentre en el centro de la matriz. De esta manera, el algoritmo empieza comparando con las imágenes con mayor probabilidad de encontrar un resultado, reduciendo el tiempo del sistema para determinar la orientación.

Capítulo 4

Desarrollo conceptual

4.1. Diseño de un subsistema de percepción remota para un nanosatélite

El primer paso que se realizó para desarrollar el subsistema fue diseñarlo. Este diseño se dividió en *software* y *hardware*. Este subsistema se compone de un sensor óptico, un dispositivo de almacenamiento y un microcontrolador para comunicarse con el sensor para capturar las imágenes, almacenarlas y comunicarse con la computadora de abordo. El microcontrolador que se seleccionó fue el PIC24EP128MC202 y el código fue implementado en el lenguaje C.

Los componentes utilizados fueron presentados en la tabla 3.1. Con ellos se diseñó el circuito electrónico mediante el *software* **Altium Designer 16.0**. Este diseño fue basado en los estándares definidos por *Pumpkin Space Systems* [38].

4.1.1. Diseño del *software* para el subsistema de percepción remota

Comunicación con la cámara

La cámara utilizada fue la modelo LS_Y201 (figura 4.1) que ocupa una superficie de 32 x 32 mm y se comunica por medio del protocolo serial RS-232 a 115200 baudios por segundo. Los comandos y funciones diseñadas para controlar parámetros de la cámara y capturar imágenes se presentan en el apéndice A.

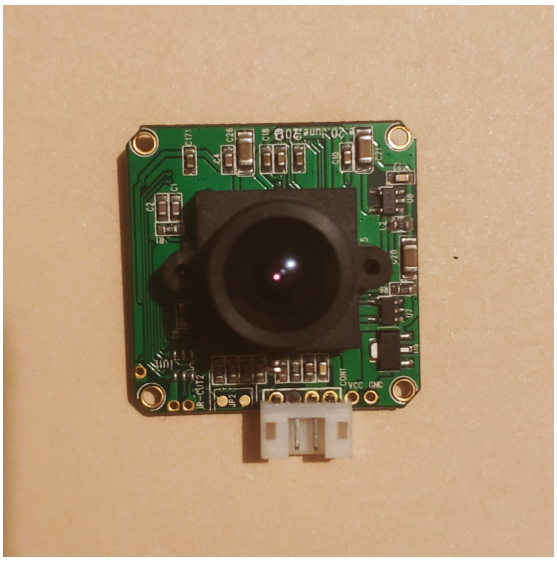


Figura 4.1: Cámara modelo LS_Y201.

Comunicación con memoria de almacenamiento (MicroSD)

Cada imagen de interés capturada por la cámara debe guardarse (para evitar su borrado por la imagen siguiente), por lo que se optó en utilizar una memoria microSD como dispositivo de almacenamiento. Este tipo de memorias poseen dos protocolos de comunicación: BUS SD y SPI. Como el microcontrolador que se utilizó tiene terminales para implementar el protocolo SPI, se decidió utilizar este último. Se tomó como referencia el artículo de C. Henao y E. Cardona para diseñar las funciones de comunicación del protocolo [41], las cuales se presentan en el apéndice B.

Comunicación con la computadora de abordo

Para comunicarse con la computadora de abordo se utilizó el protocolo de comunicación i^2c por medio de dos terminales en el canal de comunicación. La computadora de abordo controla al sistema de percepción remota e indica el momento en el cual debe capturar las imágenes. También puede pedir que envíe las imágenes almacenadas, solicitar información acerca de la capacidad del almacenamiento de memoria restante y liberar espacio de la memoria.

Esta misma línea de comunicación fue utilizada para enviar imágenes al sistema de determinación de orientación.

Diagrama de bloques del *software* del sistema de percepción remota

La figura 4.2 muestra el diagrama de bloques de alto nivel del *software* diseñado para el sistema de percepción remota. Los comandos recibidos por

la computadora de abordo definen cuatro opciones a realizar por el sistema: informar sobre el espacio de almacenamiento de la memoria, eliminar las fotos de la memoria para liberar espacio de almacenamiento, capturar imágenes o enviar las imágenes almacenadas a la computadora de abordo.

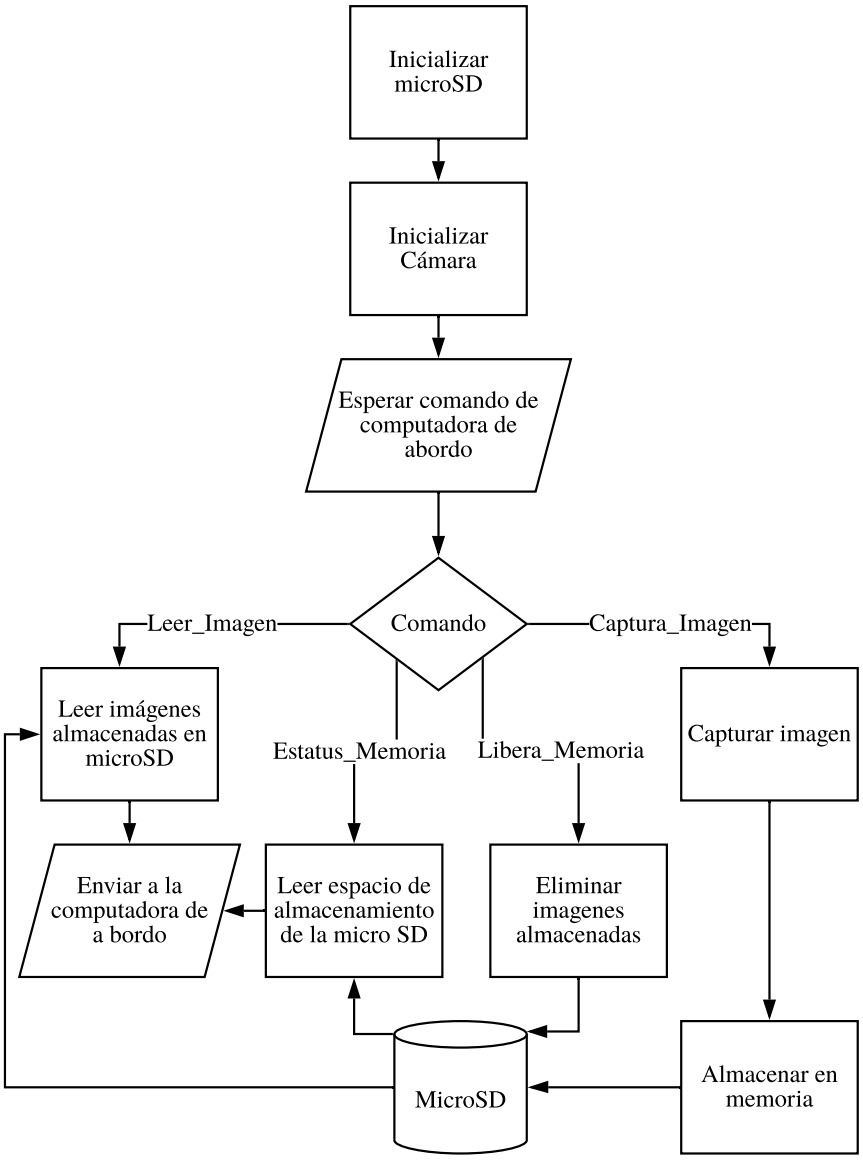


Figura 4.2: Diagrama de bloques del *software* del sistema de percepción remota.

4.1.2. Diseño del circuito electrónico

Diagrama de terminales del microcontrolador

La figura 4.3 muestra las terminales del microcontrolador PIC24EP128MC202. En este diagrama MCLR se refiere al *Master Clear* del microcontrolador, VSS y VDD identifican las terminales para la referencia a tierra y el suministro positivo de voltaje, respectivamente. VCAP es la terminal para el capacitor del filtro lógico. Para el protocolo UART, RX1 y TX2 son las terminales de recepción y transmisión de datos. Para la comunicación por SPI, las terminales *salida del maestro entrada del esclavo* (MOSI), *entrada de maestro salida de esclavo* (MISO) y el reloj (SCK), son las indicadas en los paréntesis respectivos. Finalmente, para el protocolo de comunicación i^2c, los conectores seriales sincrónicos de datos (SDA) y de reloj (SCL), que se indican de la misma manera.

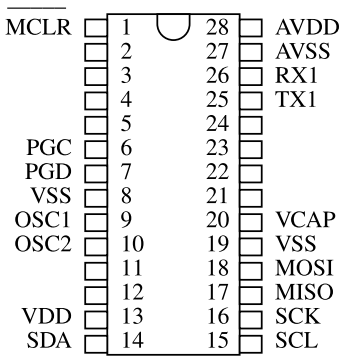


Figura 4.3: Diagrama de terminales del microcontrolador para el sistema de percepción remota.

Diagrama de terminales del canal de comunicación

La figura 4.4 muestra el diagrama de las terminales utilizadas para el canal de comunicación. Este canal de comunicación sirve para mandar y recibir información de la computadora de abordo o de otros subsistemas del satélite. Fue basado en el estándar de *Pumpkin Space Systems*. Las terminales utilizadas son las que se muestran con letras resaltadas en la imagen. De la tira de conectores H1, las terminales 41 y 42 son para la comunicación i^2c con la computadora de abordo y en la tira de conectores H2, las terminales de alimentación (5 V) y tierra son: 25, 26, 29 y 32.

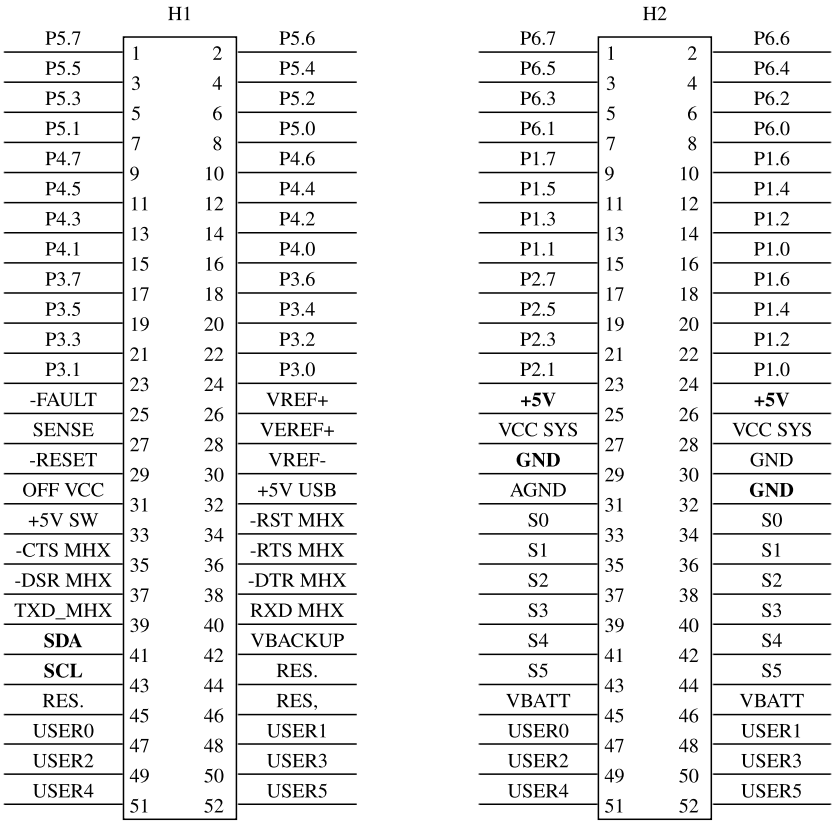


Figura 4.4: Diagrama de terminales del canal de comunicación para el sistema de percepción remota.

Diagramas de la tableta de circuito impreso

La distribución de los componentes en la placa fenólica se muestra en la figura 4.5, lo cual fue elaborado con el *software Altium Designer 16*. También se hizo un diagrama con las medidas milimétricas pertinentes (figura 4.6), otro con los conectores de los componentes electrónicos (figura 4.7) y un diagrama con el diseño de conexiones entre las terminales de los componentes (figura 4.8). Este último también se conoce como diagrama de pistas.

se realizó un diagrama con las dimensiones mostrado en la figura 4.6, otro con los conectores de los componentes electrónicos en la figura 4.7 y un diagrama con el diseño de conexiones en la figura 4.8.

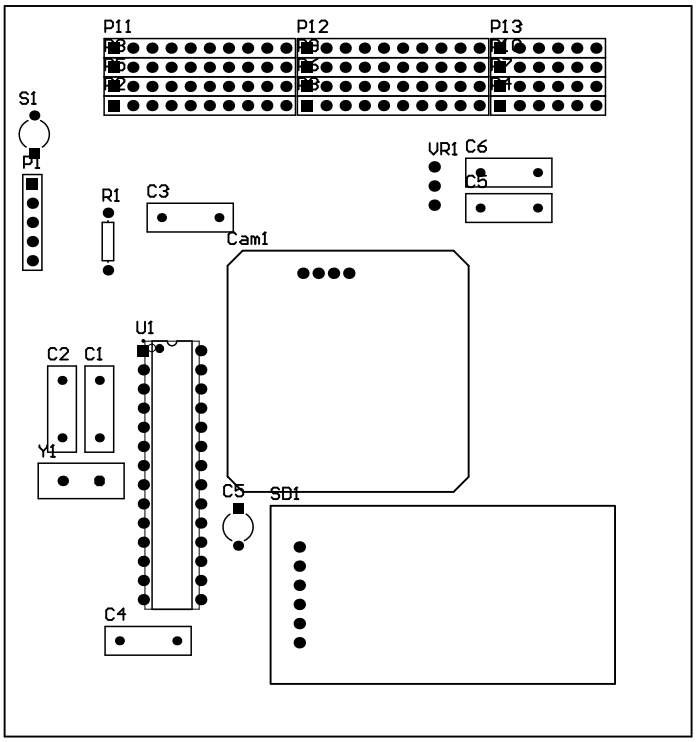


Figura 4.5: Distribución de los componentes en la tableta de circuito impreso del sistema de percepción remota.

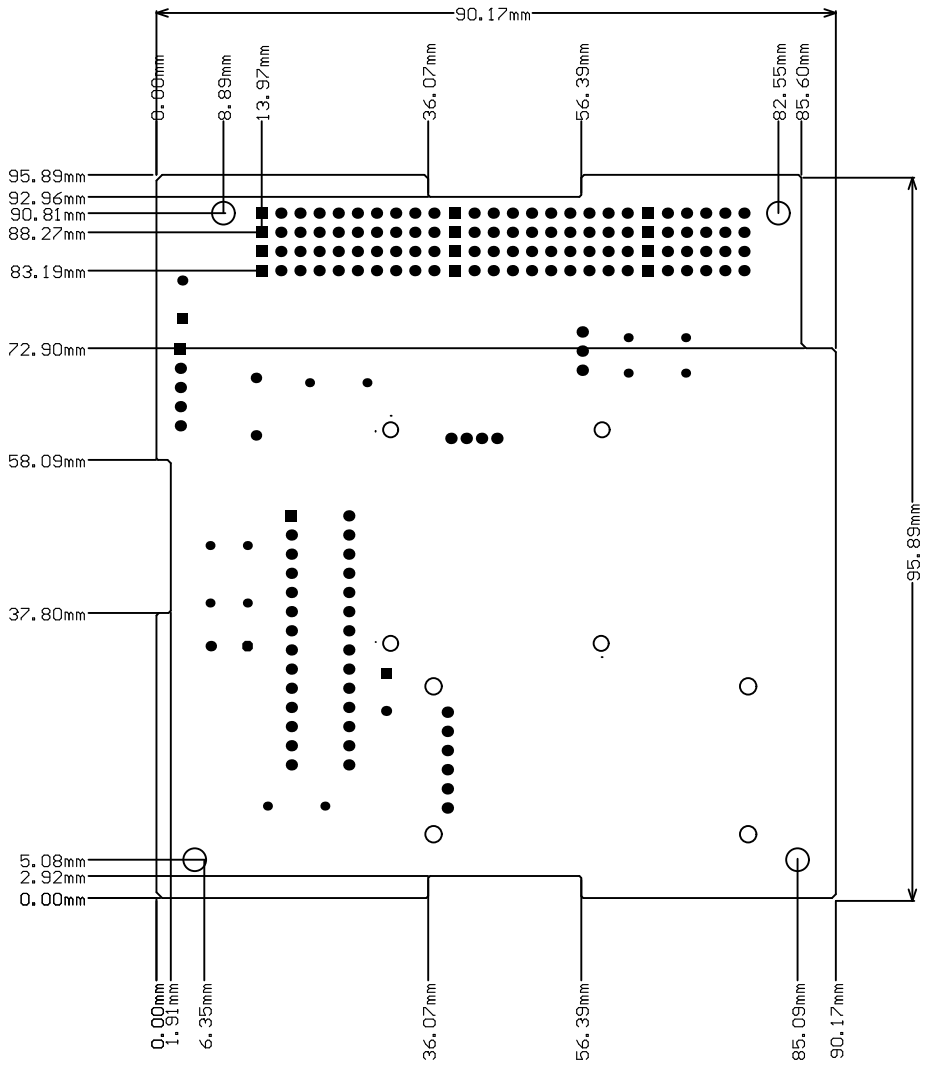


Figura 4.6: Medidas y dimensiones milimétricas en la tableta de circuito impreso.

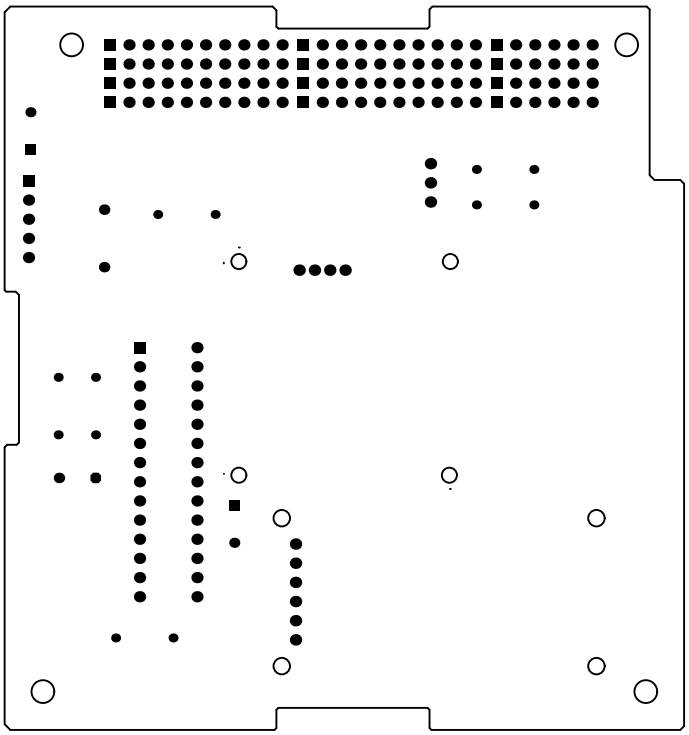


Figura 4.7: Terminales de los componentes de la tableta de circuito impreso.

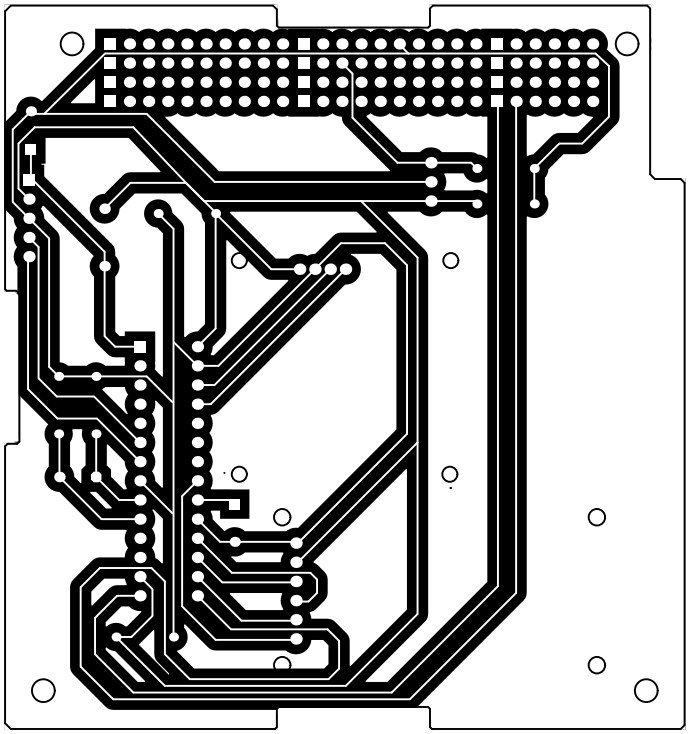


Figura 4.8: Conexiones entre las terminales de los componentes de la tableta de circuito impreso.

4.2. Desarrollo de una base de datos de patrones de estrellas basado en la transformada de distancia menor

4.2.1. Filtrado del catálogo de estrellas con base en la magnitud visual

El primer paso para crear la base de datos fue filtrar el catálogo de estrellas con base a un umbral de magnitud visual máximo. Se realizaron pruebas con diferentes magnitudes visuales buscando obtener una distribución homogénea de estrellas. La figura 4.9 muestra una gráfica del número de estrellas utilizando diferentes valores de magnitud visual aparente como umbral. En la tabla 4.1 se pueden observar algunos de estos valores. El catálogo de estrellas HYG contiene 119615 estrellas, y el valor de umbral seleccionado fue de 6.0, con el que se obtuvo una base de datos con 5043 estrellas.

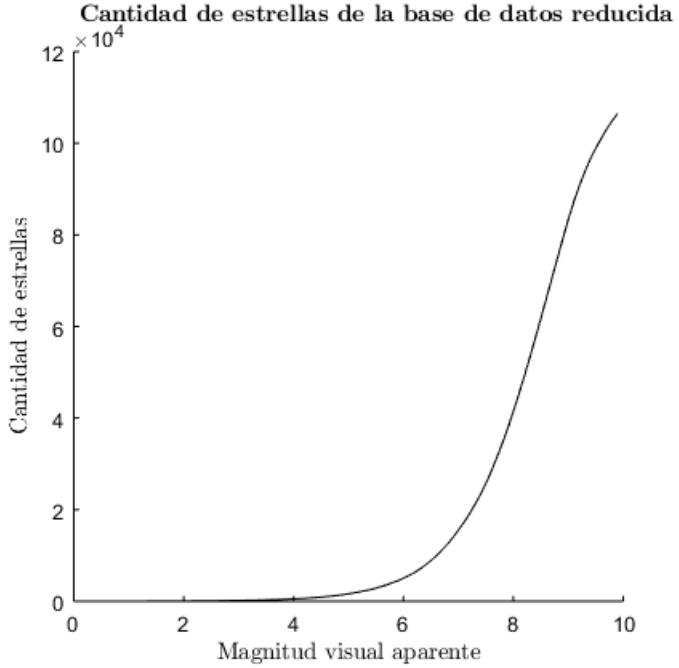


Figura 4.9: Gráfica del número de estrellas de la base de datos según el valor de magnitud visual aparente utilizado como umbral.

Tabla 4.1: Número de estrellas de la base de datos para diferentes valores de magnitud visual aparente utilizados como umbral.

MV	**Número de estrellas**
Sin filtro	119615
7.0	15539
6.5	8872
6.0	5043
5.5	2850
5.0	1627
4.5	921

4.2.2. Gráfica de la esfera celeste

Para obtener la gráfica de la esfera celeste y obtener las imágenes de la base de datos se utilizó MATLAB. Con este *software* se pudo obtener información visual sobre la base de datos. La figura 4.10 muestra la gráfica en coordenadas cartesianas de la base de datos de las estrellas, donde el eje x es la ascensión recta y el eje y es la declinación. También se realizó una gráfica en donde se visualizan las estrellas de la base de datos en una esfera unitaria mostrada en la figura 4.11

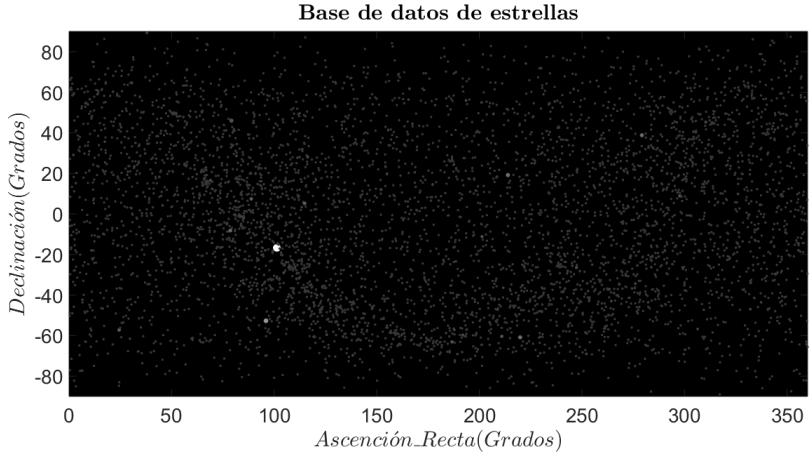


Figura 4.10: Gráfica de la base de datos de estrellas en sistema de coordenadas cartesianas.

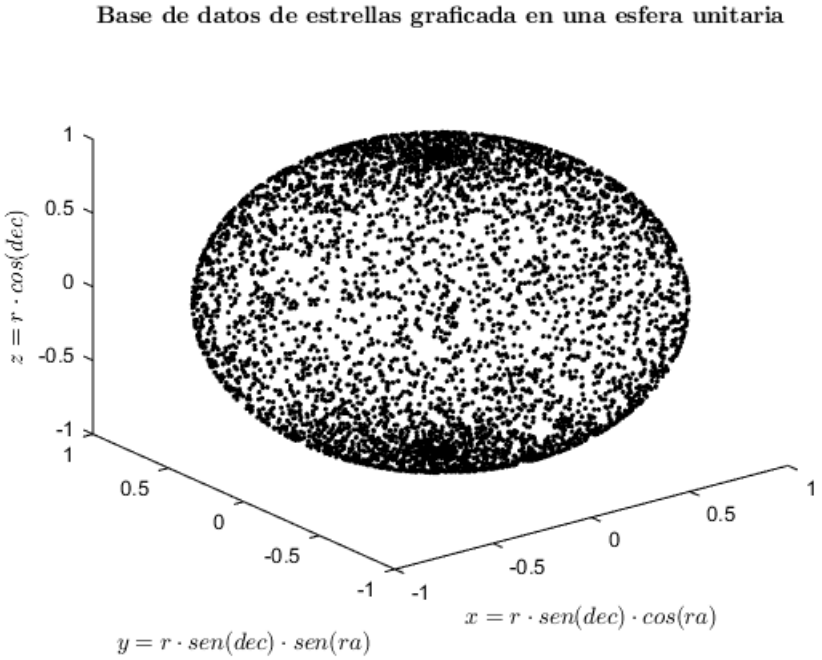


Figura 4.11: Gráfica de la base de datos de estrellas presentadas en una esfera unitaria.

4.2.3. Generación de imágenes basadas en la transformada de distancia menor

Después de obtener la gráfica de la base de datos en coordenadas cartesianas, se seccionó la imagen de acuerdo a las dimensiones de FOV. Se utilizó un FOV de 20 x 20 grados, por lo que se obtuvieron 172 imágenes. Posteriormente, se aplicó la transformada de distancia menor a cada una de las imágenes y se almacenaron, generando una nueva base de datos. El proceso para realizar la transformada de distancia menor está descrito en el algoritmo 1, donde S es una lista que contiene la posición de las tres estrellas más brillantes de la imagen, P es cada uno de los pixeles de la imagen y $D(P)$ es la transformada de distancia menor. La imagen generada es una imagen a escala de grises, por lo que el valor máximo está limitado a 255.

En la figura 4.12 se presentan ejemplos de imágenes de la base de datos. La primera es una sección de la esfera celeste y la segunda es después de realizar la transformada de distancia menor.

Algoritmo 1 Transformada de distancia menor para una imagen de la base de datos de estrellas

procedimiento TRANSFORMADA DE DISTANCIA MENOR PARA UNA IMAGEN DE LA BASE DE DATOS DE ESTRELLAS

para P como cada pixel de la imagen de la base de datos **hacer**

Calcular la distancia Euclideana al cuadrado a las estrellas en la imagen de la base de datos

para S como cada una de las estrellas en la imagen de la base de datos **hacer**

$$d(P,S) = (P_x - S_x)^2 + (P_y - S_y)^2 \tag{4.1}$$

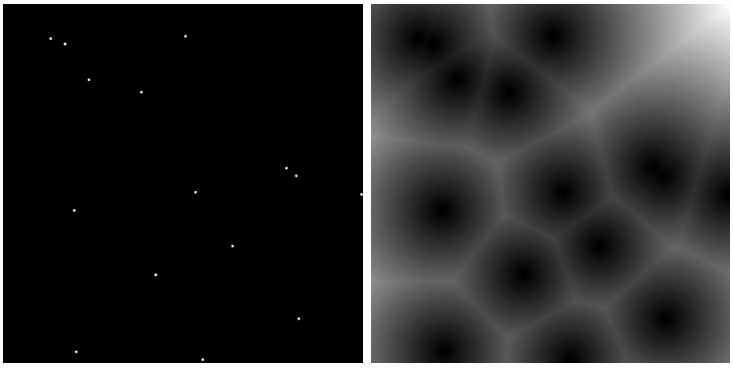
fin de para

Encontrar la distancia menor para cada pixel

$$D(P) = \min d(P,S) \tag{4.2}$$

fin de paradevolver D(P)

fin de procedimiento



(a) Sección de FOV de 20 x 20 grados.

(b) Después de realizar la transformada de distancia menor.

Figura 4.12: Imágenes de la base de datos de estrellas.

4.2.4. Pre-procesamiento a la base de datos

Para que el algoritmo fuera más ágil, se realizó un preprocesamiento a la base de datos. Este preprocesamiento está basado en el algoritmo de emparejamiento de imágenes explicado a detalle en la próxima sección. La información obtenida sobre las imágenes de la base de datos de las estrellas incluyó la posición de las tres estrellas más brillantes en cada imagen, el centroide del triángulo que forman dichas estrellas y la dirección del ángulo más agudo del triángulo. Esta información fue almacenada y es cargada al seguidor de estrellas al momento de iniciar su operación.

4.2.5. Diagrama de bloques del proceso para el desarrollo de la base de datos de estrellas

La figura 4.13 muestra el diagrama de bloques del proceso completo que se llevó a cabo para desarrollar la base de datos de estrellas basada en la transformada de distancia menor.

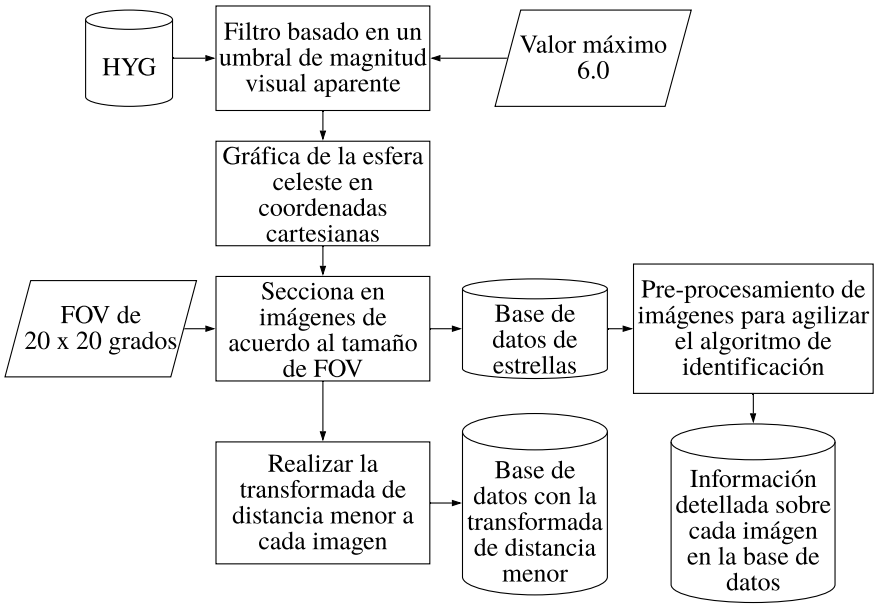


Figura 4.13: Diagrama de bloques del proceso para desarrollar las bases de datos de estrellas.

4.3. Diseño de algoritmos para un seguidor de estrellas

4.3.1. Algoritmo de detección de objetos

El algoritmo 2 presenta el pseudocódigo diseñado para determinar el pixel localizado en el centro de las estrellas detectadas. La información de entrada del algoritmo es una imagen binarizada del FOV, mientras que la salida es una lista ordenada de mayor a menor de las tres estrellas según su área y las coordenadas del centroide de cada una de ellas. La figura 4.14 muestra un ejemplo del resultado obtenido de dicho algoritmo, en el que se remarcan las estrellas detectadas en el FOV.

Algoritmo 2 Centroide

procedimiento ALGORITMO DE CENTROIDE UTILIZANDO EL DETECTOR SIMPLE DE OBJETOS DE OPENCV

Definir los parámetros de los filtros del detector de objetos

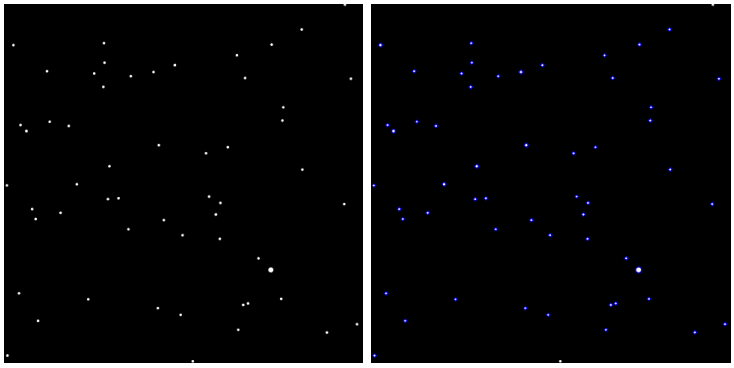
Parámetros de umbral, área mínima, convexidad, proporción de inercia y distancia mínima entre manchas específicados en la tabla 3.2.

Utilizar el detector de objetos

Coordenadas de los objetos = *SimpleBlobDetector*(FOV)

Ordenar la lista

si número de objetos < 3 **entonces**
    Obtener nueva imagen del FOV.
si no si número de objetos $= 3$ **entonces**
    Obtener una lista ordenada de mayor a menor según su área.
si no
    Realizar el algoritmo para seleccionar a las tres estrellas guía.
fin de si
devolver
Lista de las tres estrellas de mayor tamaño.
fin de procedimiento



(a) Ejemplo de una imagen de entrada al algoritmo.

(b) Imagen con las estrellas detectadas resaltadas.

Figura 4.14: Ejemplo del resultado obtenido con el algoritmo de centroide.

Criterio para seleccionar a las estrellas guía

En caso de encontrar tres estrellas en el FOV, se ordenan de mayor a menor. Sin embargo, en el caso de encontrar más de tres estrellas se tuvieron que usar ciertos criterios para ordenarlas y seleccionarlas. En caso de que todas las estrellas tuvieran un tamaño diferente, se selecciona a las estrellas con mayor tamaño. Para el caso de que las estrellas encontradas tuvieran el mismo tamaño se desarrolló el algoritmo 3 para seleccionar a las estrellas guía.

Algoritmo 3 Selección de estrellas guía

procedimiento ALGORITMO PARA SELECCIONAR A LAS TRES ESTRELLAS GUÍA

Ordenar la lista de estrellas según su tamaño de mayor a menor

si el tamaño de la **tercer** estrella en la lista es igual al de la **cuarta** estrella **entonces**

Obtener el último índice de las estrellas con el mismo tamaño que la tercer estrella en la lista

$i = 4$

mientras $i \leq$ al tamaño de la lista **hacer**

si el tamaño de la **tercer** estrella en la lista es igual al de la estrella en la posición **i entonces**

$i = i + 1$

si no

Salir del ciclo

fin de si

fin de mientras

En caso de que las primeras dos estrellas de la lista también tengan el mismo tamaño

si el tamaño de la **primer** estrella en la lista es igual al de la **tercer** estrella **entonces**

Calcular el perímetro formado por las dos estrellas más cercanas a cada una de las estrellas hasta el índice i.

Seleccionar las tres estrellas cuyo perímetro sea el de menor valor.

En caso de que sólo la primer estrella sea mayor que las demás

si no si el tamaño de la **segunda** estrella en la lista es igual al de la **tercer** estrella **entonces**

Calcular la distancia entre la primer estrella de la lista con las demás estrellas hasta el índice i.

Seleccionar las dos estrellas cuya distancia sea menor a la primer estrella.

En el caso que las primeras dos estrellas en la lista sean más grandes que las demás estrellas hasta el índice i

si no

Calcular la distancia (D) entre la primer estrella con las estrellas a partir desde la tercera hasta el ínidice i y múltiplicarlo por el tamaño de la primer estrella.

Calcular la distancia entre la segunda estrella con las estrellas a partir desde la tercera hasta el ínidice i, múltiplicarlo por el tamaño de la segunda estrella y sumarlo a D.

Seleccionar a la estrella cuyo valor en D sea menor.

fin de si

fin de si

devolver Lista ordenada de mayor a menor de las tres estrellas más grandes

fin de procedimiento

4.3.2. Algoritmo de identificación de estrellas basado en la transformada de distancia menor

Emparejamiento de imágenes

El algoritmo 4 describe el método de centroide utilizado para emparejar a la imagen del FOV con las imágenes de la base de datos, en el que se utiliza como entrada la lista de las estrellas más brillantes del FOV obtenida por el algoritmo de centroide, identificadas como A, B y C. En el algoritmo, la variable *centroide* es la posición del centroide del triángulo formado por las tres estrellas. La lista *ángulos* está conformada con los ángulos formados entre las estrellas teniendo como origen el centroide del triángulo, *idx* es el índice del ángulo más agudo de la lista y *dirección* es el ángulo entre el vector que apunta a la mitad del ángulo menor y el eje de ascensión recta. Cuando se compara con la base de datos se obtienen Θ y Δ, que son la diferencia entre la orientación del ángulo menor de las imágenes y la distancia entre los centros de los triángulos, respectivamente. Finalmente, la matriz de rotación R, se muestra en la ecuación 4.3.

$$R(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \tag{4.3}$$

Algoritmo 4 Método de centroide para emparejar las imagenes del FOV con las de la base de datos

procedimiento EMPAREJAMIENTO DE IMÁGENES POR EL MÉTODO DE CENTROIDE

Calcular el centroide del triángulo formado por las estrellas A(0), B(1) y C(2)

$$centroide = \frac{A + B + C}{3} \tag{4.4}$$

Encontrar el ángulo menor del triángulo

$$estrella(3) = estrella(0) \tag{4.5}$$

para i de 0 hasta 2 **hacer**

$$\begin{aligned} V1(i) &= estrella(i) - centroide && (4.6a)\\ V2(i) &= estrella(i+1) - centroide && (4.6b)\\ \theta &= \frac{V1 \cdot V2}{|V1|\,|V2|} && (4.6c)\\ ángulos(i) &= arcoseno(\theta) && (4.6d) \end{aligned}$$

fin de para

$$\begin{aligned} menor &= \text{mín}\,(ángulos) && (4.7a)\\ idx &= indicedelángulomenor && (4.7b)\\ & && (4.7c) \end{aligned}$$

Determinar la dirección del vector que apunta al ángulo menor del triángulo

$$dirección = \frac{V1(idx) + V2(idx+1)}{2} \tag{4.8}$$

Comparar con las estrellas en la imagen de la base de datos

$$\begin{aligned} \Theta &= dirección(BaseDatos) - dirección && (4.9a)\\ \Delta &= centroide(BaseDatos) - centroide && (4.9b) \end{aligned}$$

Rotar y trasladar las estrellas

para i de 0 hasta 2 **hacer**

Cambiar la posición de la estrella i al sistema de referencia con el centro de la imagen como origen

$$estrella(i) = R(\Theta) \cdot estrella(i) \tag{4.10}$$

Cambiar la posición de la estrella i al sistema de referencia original

$$estrella(i) = estrella(i) + \Delta \tag{4.11}$$

fin de para

devolver Lista con la nueva posición de las estrellas

fin de procedimiento

Comparación con la base de datos mediante la transformada de distancia menor

Después de haber emparejado las imágenes, se calcula la transformada de distancia menor de acuerdo al algoritmo 1 para cada una de las estrellas del campo de visión y se suman los valores obtenidos. En el caso de que dicho valor sea menor a una tolerancia definida, eso implica que las imágenes han coincidido entre ellas, por lo que se termina el proceso y se continúa a determinar la orientación. En el caso contrario, el valor es almacenado y se sigue comparando con las imágenes de la base de datos. Si ninguna imagen da un valor menor al criterio definido, se busca el valor menor y, considerando otro criterio de error, se determina si tal valor puede ser considerado como correcto, de ser así, pasa la información al algoritmo de determinación de orientación. Si ninguna imagen da un resultado que sea confiable, se toma una nueva imagen del FOV y el proceso inicia de nuevo.

Ordenamiento de la base de datos para el modo de operación de LIS

Cuando no se tiene información sobre la orientación actual del sistema, es complicado saber por dónde iniciar a comparar con la base de datos para determinarla en el menor tiempo. El método propuesto fue, de acuerdo al número de estrellas que se encuentran en el FOV, ordenar la base de datos para buscar primero en las imágenes que tuvieran el mismo número de estrellas, continuando con las imágenes que tuviera un mayor número de estrellas y finalmente, con las que tuvieran un menor número de estrellas. Se decidió este orden debido a que hay una mayor probabilidad de detectar estrellas falsas en el FOV, a no detectar estrellas que estén contenidas en la base de datos. Esto se explicará más adelante con mayor detalle, al describir el desempeño del sistema.

El proceso que se llevó a cabo fue implementar un algoritmo de ordenamiento de acuerdo al número de estrellas de la base de datos. Después se identificó el índice del arreglo en el que coincide el número de estrellas en el FOV y con él se definió el orden en el que se compararán las imágenes de la base de datos.

Ordenamiento de la base de datos para el modo de operación de seguimiento

Al entrar en el modo de operación de seguimiento y con la orientación actual del satélite calculada, se modificó el orden en el que se comparan las imágenes de la base de datos. La idea principal, es recorrer la base datos para que el valor de orientación estimada quede en el centro de esta forma, se inicia comparando con sus imágenes adyacentes y se continúa con las imágenes adyacentes a estas últimas, formando un espiral hasta llegar al extremo opuesto del último resultado encontrado.

4.3.3. Algoritmo de determinación de orientación

Una vez identificadas las estrellas en el FOV, es posible estimar la orientación absoluta del sistema. Las ecuaciones 4.12 describen los cálculos para determinar cada uno de los ángulos en coordenadas esféricas.

$$orientación(ra) = RA - \Delta(0) \tag{4.12a}$$
$$orientación(dec) = DEC - \Delta(1) \tag{4.12b}$$
$$orientació(alabeo) = -\Theta \tag{4.12c}$$

Δ es la distancia que se desplazó y Θ el ángulo que se giró la imagen capturada para ser emparejada con la imagen de la base de datos con la que el patrón de estrellas coincidió. RA y DEC son los valores de ascensión recta y declinación en el centro de la imagen en la base de datos, y *orientación* es el resultado obtenido del algoritmo.

4.3.4. Diagrama de bloques del funcionamiento del seguidor de estrellas

La figura 4.15 muestra el diagrama de bloques del funcionamiento del seguidor de estrellas. En él se muestra el flujo de los algoritmos diseñados y el proceso para determinar la orientación del sistema.

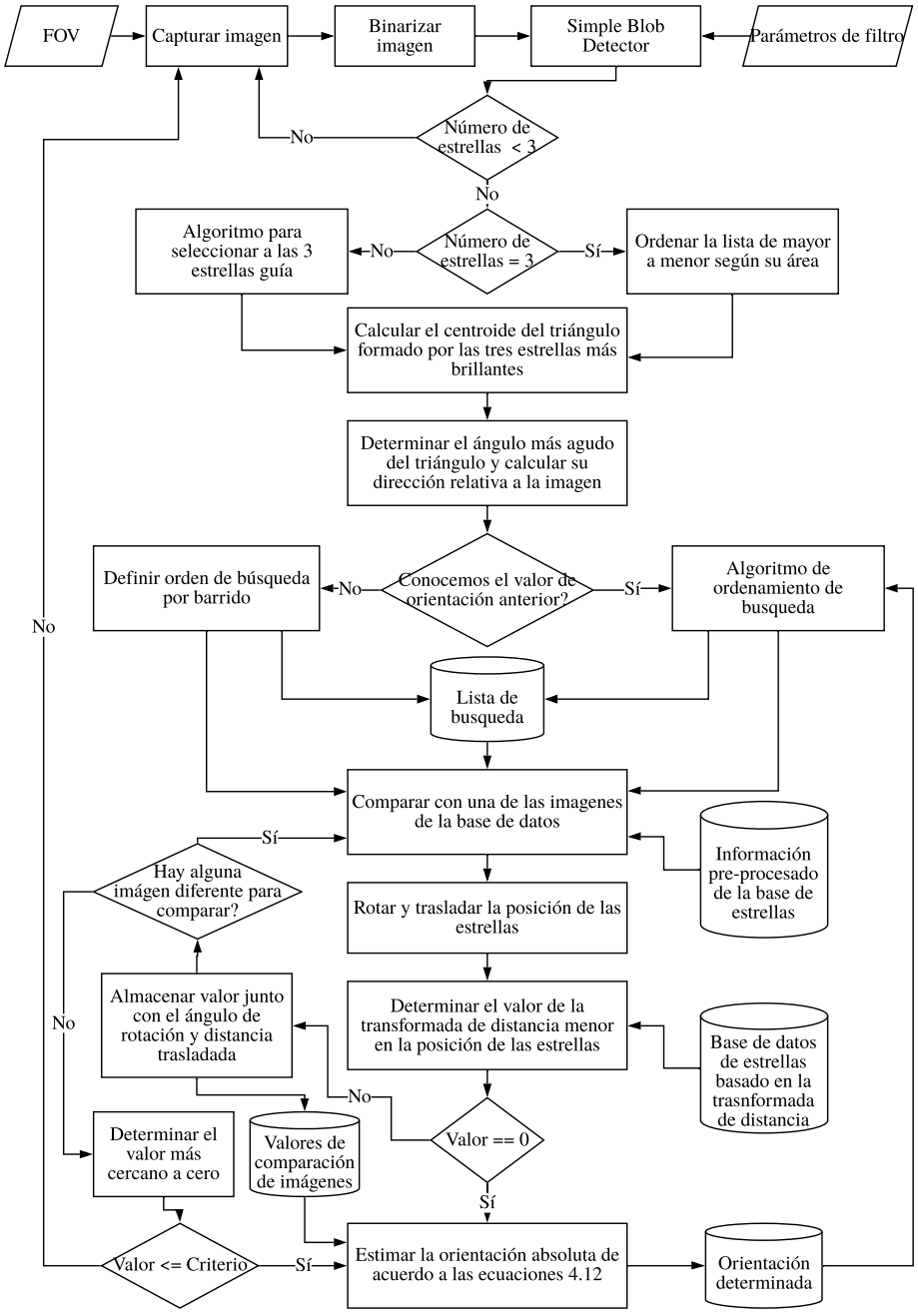


Figura 4.15: Diagrama de bloques del proceso de los algoritmos del seguidor de estrellas.

Capítulo 5

Desarrollo experimental

5.1. Desarrollo de un subsistema de percepción remota para un nanosatélite

5.1.1. Implementación en tableta de pruebas

Primero se implementó el prototipo del subsistema en una tableta de pruebas como se muestra en la figura 5.1. En ella se utilizaron los componentes descritos en la tabla 3.1.

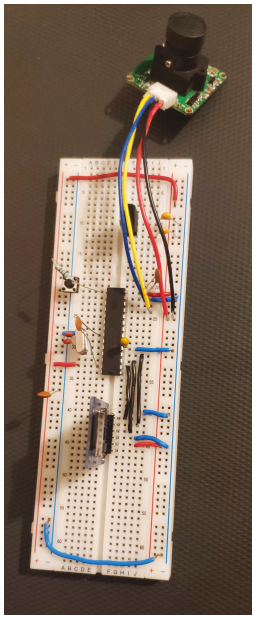
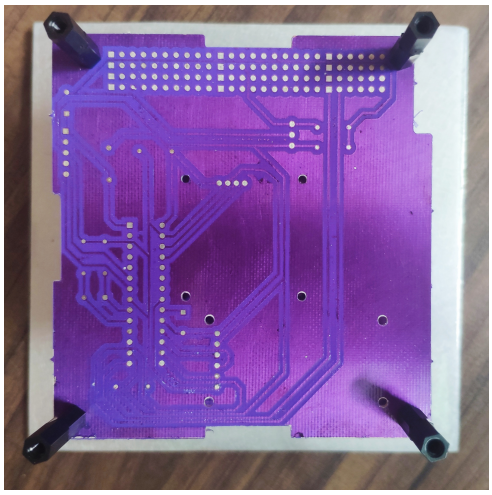


Figura 5.1: Prototipo de sistema de percepción remota en tableta de pruebas.

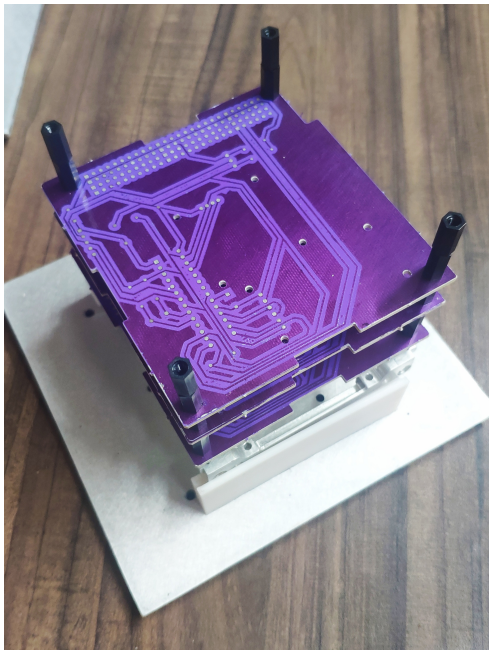
5.1.2. Diseño de la tableta de circuito impreso

EL diseño de la tableta de circuito impreso fue fabricada en el Laboratorio de Sistemas Embebidos de la UAT. Los diseños mostrados en las figuras 4.7 y 4.8 fueron impresos en acetatos. La placa fenólica fue cortada de acuerdo a los estándares del nanosatélite tipo CubeSat "Pumpkin"como se muestra en la figura 4.6. Después se colocó una película fotosensible a la placa fenólica utilizando una laminadora a 100°C y a velocidad moderada. Posterior a eso, se expuso a luz ultravioleta con la impresión de las trazas (figura 4.8) por 50 s, y se limpió la película sobrante con una mezcla de agua y bicarbonato de sodio. A continuación, se realizó un ataque químico utilizando Cloruro Férrico para quitar el cobre que no estaba cubierto por la película. El siguiente paso fue aplicar otra película fotosensible y utilizar el diagrama mostrado en la figura 4.7 y exponiendo la placa a luz ultravioleta para sólo dejar descubiertos los conectores de los componentes. La figura 5.2a muestra la placa estañada antes de ser perforada. La figura 5.2b muestra la manera en que se apilarán los subsistemas dentro del nanosatélite tipo CubeSat.

Como trabajo a futuro, se diseñará y fabricará una tableta de circuito impreso con componentes en otro empaquetado y que incluya redundancia de componentes para lograr una mayor confiabilidad del sistema.



(a) Tableta de circuito impreso (sin componentes soldados).



(b) Tabletas de circuitos impresos apiladas con la estructura de un CubeSat.

Figura 5.2: Tableta de circuito impreso (sin componentes soldados) del sistema de percepción remota.

5.2. Validación del sistema de determinación de orientación

5.2.1. Principales indicadores de rendimiento

Para evaluar los algoritmos implementados en el presente trabajo, se se definieron los parámetros que se consideraron más significativos en el desempeño del sistema al determinar su orientación absoluta.

Precisión

Cuando el seguidor de estrellas determina su orientación con mayor precisión, mejora el desempeño de otros subsistemas en el satélite. Es decir, valores más confiables de la orientación del sistema permiten a la carga útil obtener información más certera, mejorar la comunicación del sistema con la base terrena, o también permitir que el sistema de potencia adquiera una cantidad mayor de energía mediante las celdas solares.

Complejidad computacional

Al reducir la complejidad del algoritmo, el seguidor de estrellas puede alcanzar su objetivo en un menor tiempo, lo que implica un mejor desempeño del sistema. Además, influye en la potencia del procesador requerido, por lo que reducir la complejidad del *software* disminuye el costo y consumo de potencia de la misión [3].

Confiabilidad

Cuando el seguidor de estrellas se encuentra en operación, se enfrenta a un ambiente hostil con muchas perturbaciones para el sistema, por lo que es necesario que los algoritmos sean inmunes a cualquier tipo de ruido. Estas perturbaciones pueden ser ocasionadas por fallas en el sensor óptico, la inercia del sistema o falsas estrellas detectadas en el campo de visión.

5.2.2. Validación de los algoritmos mediante pruebas de simulación

La base de datos contaba con 595 imágenes, que correspondían a un campo de visión de 20°x 20°, cubriendo por completo la esfera celeste. Entre cada una de las imágenes adyacentes existía una diferencia de 10°con respecto a la ascensión recta o la declinación, es decir, si la primera imagen cubre de 0°a 20°en la ascensión recta y de -90°al -70°en declinación, una imagen adyacente cubriría los mismos grados en ascensión recta, pero de -80°al -60°en la declinación. Sin tomar en cuenta la posición de 350°a 10°en ascensión recta y la de 80°a -80°en declinación solar, el rango de ascensión recta se puede segmentar en 35 partes y la de declinación en 17, con lo que se obtuvo un total de 595 secciones. Se

detectaron tres secciones que no contaban con al menos tres estrellas, por lo que algoritmo de reconocimiento de patrones implementados no fue adecuado para estos casos.

Las primeras pruebas fueron para el caso ideal, en donde ninguna perturbación en las imágenes de entrada fue considerada. Se hicieron pruebas para los dos diferentes modos de operación para determinar la frecuencia promedio para cada uno. También se giró la base de datos 90°y se calculó el error promedio para cada uno de los ángulos.

La siguiente ronda de pruebas, incluyó una alteración en la posición detectada de las estrellas. Este ruido se generaba al redondear los valores de ascensión recta y declinación de las estrellas en el campo de visión en diferentes dígitos. El ruido en la posición de las estrellas para este sistema en operación puede ser originado por el error en la precisión del algoritmo de detector de objetos y determinación de centroide. También puede surgir a partir de una falla en el sensor óptico, sobre todo para este tipo de proyectos cuyos componentes son de bajo costo. Además, el movimiento del sistema en órbita puede aumentar el error al determinar la posición de las estrellas. Por ello es importante que el algoritmo sea suficientemente confiable ante este tipo de perturbación.

Otra posible dificultad a la que se enfrentará el seguidor de estrellas al estar en funcionamiento es detectar estrellas falsas en el campo de visión o, al contrario, no detectar todas las estrellas en el campo de visión que sí pasen el umbral de magnitud visual de la base de datos. Esto puede ser generado principalmente por fallas en el sensor óptico. También puede haber otros sistemas artificiales que pasen por el FOV del sistema. De acuerdo a T. Delabie [3], la probabilidad de que un objeto artificial pase por el campo de visión para un seguidor de estrellas con el campo de visión de 20°x 20°y un tiempo de integración de 0.1s en una órbita LEO es alrededor de 1.44 % tomando en cuenta que la Red de Vigilancia Espacial de E.U. tiene 16,094 objetos catalogados de al menos 10cm de longitud en Julio del 2011. La mayoría de estos objetos no serán lo suficientemente grandes para producir una pequeña mancha en el campo de visión que no será detectada como una estrella, sin embargo, una pequeña fracción de ellos, como la Estación Espacial Internacional, podría ser un objeto más brillante que incluso estrellas verdaderas en el campo de visión. También rayos cósmicos, e incluso planetas como Venus, Marte, Júpiter, Mercurio y Saturno podrían ser detectados como estrellas falsas.

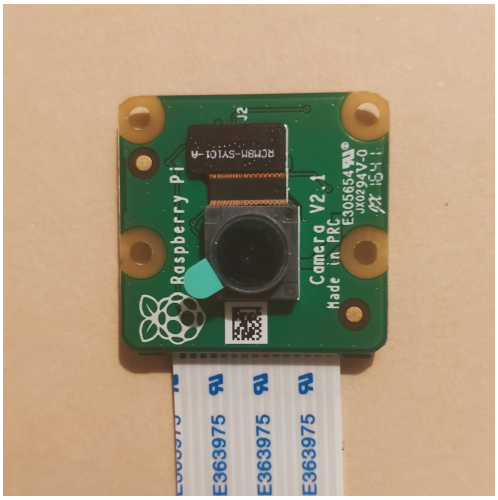
Se realizaron dos diferentes tipos de pruebas para validar el desempeño del algoritmo de identificación de estrellas al detectar de manera incorrecta las estrellas en el FOV. En la primera, se agregó un filtro al algoritmo para eliminar estrellas menores a determinado umbral, y se fue modificando el umbral para observar cómo se comportaba el sistema al no detectar todas las estrellas en el FOV. Para el caso de estrellas falsas, se generaron diferentes bases de datos en MATLAB agregando estrellas falsas en las imágenes utilizando un arreglo de números aleatorios.

5.2.3. Pruebas de funcionamiento de los algoritmos de integración con *hardware*

Se implementaron los algoritmos del seguidor de estrellas en la tableta Raspberry Pi 3® Modelo B que tiene un procesador *Quad Core 1.2GHz Broadcom BCM2837* de 64 bits y 1Gb de memoria RAM y la cámara V2 de la Rasberry Pi®. Se proyectaron las imágenes de la base de datos en un monitor y mediante la comunicación con una computadora se almacenaron los resultados obtenidos del sistema al determinar su orientación. La figura 5.3 muestra la cámara y el microprocesador utilizados.

Los resultados obtenidos en las diferentes pruebas descritas en esta sección serán mostradas y discutidas en el siguiente capítulo.

(a) Raspberry Pi 3 Modelo B.



(b) Cámara V2.

Figura 5.3: Microprocesador y cámara utilizadas para validación de los algoritmos del seguidor de estrellas.

Capítulo 6

Análisis de resultados

6.1. Sistema de percepción remota

El sistema fue capaz de capturar las imágenes y almacenarlas, posterior a eso se procesaron en la computadora para poder visualizarlas.

El prototipo desarrollado en tableta de circuito impreso cumplió con los estándares y protocolos de un subsistema para un nanosatélite tipo CubeSat definidos por *Pumpkin Space Systems* [38]. Sin embargo, no se terminó de ensamblar dicha tarjeta del subsistema debido a que se optó por cambiar el empaquetado de los componentes a uno óptimo para funcionar en el espacio, así como duplicar algunos de los componentes por seguridad en caso de que alguno falle en operación, como la cámara y el módulo de memoria.

El *software* implementado en el sistema se desarrolló de manera modular para facilitar la modificación de los componentes. Sobre todo, si es necesario cambiar la cámara de acuerdo con los requerimientos de la misión espacial en el que sea utilizado.

6.2. Sistema de determinación de orientación

6.2.1. Resultados obtenidos en las pruebas de simulación

Pruebas para el caso ideal

El número de resultados correctos obtenidos en cada una de estas pruebas fue de 99.5 % de las imágenes, debido a que 3 imágenes en la base de datos no contaban con al menos 3 estrellas que son requeridas para llevar a cabo el algoritmo de identificación de estrellas. En la tabla 6.1 la columna “AR” se refiere al ángulo de ascensión recta, “D” al de declinación y “A” al de alabeo. La última columna muestra la frecuencia promedio de estimación de orientación en cada una de las pruebas.

Tabla 6.1: Resultados obtenidos en las pruebas de simulación para el caso ideal.

Modo	**Rotación**	**Error (AR)**	**Error (D)**	**Error (A)**	**Frecuencia**
LIS	0°	$1.74\text{x}10^{-3}$ ”	$1.68\text{x}10^{-3}$ ”	0 ”	11.26 Hz
LIS	90°	6.27 ”	0.04 ”	6.32 ”	9.47 Hz
Seguidor	0°	$1.74\text{x}10^{-3}$ ”	$1.68\text{x}10^{-3}$ ”	0 ”	17.25 Hz
Seguidor	90°	6.27 ”	0.04 ”	6.32 ”	14.68 Hz

La figura 6.1 muestra una gráfica del tiempo de ejecución del sistema para cada una de las entradas utilizadas del caso ideal en ambos modos de operación. En esta gráfica se puede observar que en la mayoría de los casos el tiempo de estimación de orientación en el modo de operación de seguidor es menor al modo de operación de LIS. Sin embargo, puede ocurrir el caso, como la imagen 176, en donde el modo de operación seguidor tarda más de 300 ms en alcanzar su objetivo. Esto se debe a que en esta región de la esfera celeste hay mayor concentración de estrellas, por lo que el algoritmo de búsqueda del modo seguidor tarda en comparar las imágenes adyacentes a la última estimación realizada. Por otro lado, para el modo de operación de LIS, al ordenar la base de datos por el número de estrellas, le permite comparar con menos imágenes para identificar el resultado correcto. El promedio del tiempo de ejecución obtenido para el modo de operación LIS fue de 88.84 ms, mientras que para el modo de operación de seguidor fue de 57.96 ms. El tiempo de ejecución del sistema está determinado por el número de pixeles de la imagen de la cámara, el número de estrellas con el mismo brillo en el campo de visión, el número de imágenes en la base de datos y el pre-procesamiento de la misma.

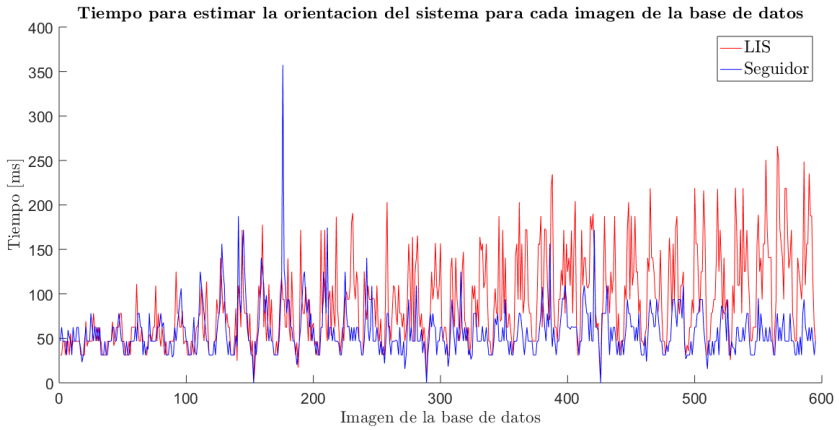


Figura 6.1: Gráfica del tiempo de ejecución para determinar la orientación de cada imagen de entrada utilizada durante las pruebas del caso ideal.

Pruebas para perturbación en la posición de las estrellas

La tabla 6.2 muestra el ruido y el porcentaje de resultados correctos de acuerdo a la magnitud de la alteración en la posición de las estrellas. Se puede observar que el sistema es confiable incluso para una gran alteración de las estrellas en el campo de visión, ya que se obtuvieron todos los resultados correctos incluso con perturbaciones de 0.05° en ascensión recta y declinación. El error promedio fue calculado como:

$$\epsilon_T = \sqrt{\epsilon_{AR}^2 + \epsilon_D^2 + \epsilon_A^2} \tag{6.1}$$

Donde ϵ_T es el error promedio total y ϵ_{AR}, ϵ_D y ϵ_A son el error promedio para cada uno de los ángulos.

Tabla 6.2: Resultados obtenidos de las pruebas con la posición alterada de las estrellas en el FOV.

Perturbación	**Resultados correctos**	ϵ_T
0°	99.50 %	$8.30\text{x}10^{-7}$ °
± $5\text{x}10^{-4}$ °	99.50 %	$2.5\text{x}10^{-4}$ °
± $5\text{x}10^{-3}$ °	99.50 %	$3\text{x}10^{-3}$ °
± $5\text{x}14^{-2}$ °	99.50 %	$2.6\text{x}10^{-2}$ °
± 0.5 °	99.33 %	0.47 °

Pruebas para el error en la detección de las estrellas en el FOV

Los resultados de las pruebas para estrellas no detectadas en el FOV se presentan en la tabla 6.3. Se puede observar que el número de resultados no encontrados depende del número de estrellas no detectadas, sin embargo, esto se debe a que el número de estrellas detectadas en el campo de visión, en algunas de las imágenes fallidas, es menor a tres, por lo que el algoritmo de reconocimiento de patrones no puede llevarse a cabo. Esta fue la serie de pruebas en donde el algoritmo se mostró menos confiable, por lo que se debe tomar en cuenta al seleccionar el umbral para el tamaño mínimo de las estrellas correcto para el sensor óptico y la base de datos definidos por la misión espacial.

Tabla 6.3: Resultados obtenidos de las pruebas con estrellas faltantes en el FOV.

Umbral	**Estrellas no detectadas**	**Resultados correctos**
≥ 4.40	11.24 %	91.76 %
≥ 4.48	24.03 %	72.77 %
≥ 4.50	37.81 %	44.03 %
≥ 4.60	38.38 %	42.52 %

La tabla 6.4 muestra los resultados obtenidos de acuerdo al número de estrellas falsas que fueron agregadas a las imágenes en la base de datos.

Tabla 6.4: Resultados obtenidos de las pruebas con estrellas falsas en el FOV.

Número de estrellas falsas	**Resultados correctos**
400	98.15 %
600	97.48 %
800	96.81 %
1000	96.47 %
1200	95.63 %
1600	93.95 %
2000	91.60 %

6.2.2. Resultados obtenidos en las pruebas de integración con *hardware*

Se capturaron 1,000 imágenes y utilizando el modo de operación LIS se obtuvo un 98.4 % de resultados correctos. Los errores obtenidos en las pruebas pueden ser causados por una inclinación errónea de la cámara al realizar la prueba ocasionando una modificación en las dimensiones de las estrellas. La figura 6.2 muestra un ejemplo de una imagen capturada por el sistema. La frecuencia promedio de estimación de orientación que se obtuvo fue de 10.4 Hz.

Figura 6.2: Imagen capturada por el seguidor de estrellas.

Capítulo 7

Conclusiones

En el presente proyecto se desarrolló un prototipo de un subsistema de percepción remota y un seguidor de estrellas, ambos construidos utilizando componentes COTS. El prototipo del subsistema de percepción remota fue capaz de capturar imágenes y almacenarlas, así como el seguidor de estrellas identificó los patrones de estrellas en su campo de visión de manera exitosa durante las diversas pruebas funcionales que se le aplicaron, sin embargo, aún no están listos para desempeñar estas funciones en órbita. Para poder ser parte de una misión espacial, primero se deben realizar pruebas de certificación para asegurar que cumplen con los criterios de calidad espacial.

El prototipo del sistema de percepción remota cumple con las dimensiones estándar para un nanosatélite tipo CubeSat, pero se debe diseñar un nuevo sistema que tenga respaldo de sus componentes, como se describió en el diagrama de bloques del subsistema (figura 3.1), con al menos dos dispositivos de almacenamiento y dos sensores ópticos. Sin embargo, el *software* desarrollado de manera modular puede ser fácilmente adaptado para diferentes cantidades de tarjetas de memoria o sensores.

Los algoritmos para el seguidor de estrellas probaron ser inmunes a diferentes perturbaciones como estrellas no detectadas y estrellas falsas detectadas en el campo de visión o, a alteración en la posición de las estrellas. Esto fue debido a que en comparación a los algoritmos de identificación de estrellas utilizados comúnmente, que comparan un determinado número de estrellas, el algoritmo que se desarrolló compara todas las estrellas en el campo de visión. Además, la frecuencia de estimación de orientación para ambos modos de operación y la precisión angular de la orientación obtenidas fueron similares a las reportadas en la literatura.

Aunque es largo el proceso para desarrollar un sistema espacial, la creación de prototipos es un gran paso para alcanzar esta meta. Con la implementación de los protocolos de comunicación, los algoritmos diseñados, y el desarrollo de circuitos que cumplen con las dimensiones estándar CubeSat, ya se puede comenzar a probar con diferentes componentes y adaptar los subsistemas de acuerdo a los requerimientos de diversas misiones espaciales.

Capítulo 8

Trabajo a futuro

Como primer meta a futuro, se diseñará un nuevo sistema de percepción remota que incluya respaldos de sus componentes, así como un empaquetado óptimo para un sistema espacial. Se espera agregar otra cámara y otro dispositivo de almacenamiento, con el objetivo de obtener un sistema más confiable.

Para el seguidor de estrellas, se plantea el desarrollo de otros algoritmos que permitan hacer una comparación de desempeño con los implementados en el presente trabajo. Desde probar diferentes algoritmos para detección de objetos y calcular el centroide de ellos, hasta el desarrollo de algoritmos de identificación de estrellas cuyo desempeño y funcionamiento haya sido evaluado en otros proyectos. Aparte, se planea reducir el tamaño de la base de datos e implementar nuevos métodos para seleccionar a las estrellas guía a utilizar, buscando obtener una distribución más homogénea y eliminando la posibilidad de imágenes en la base de datos que contengan menos de tres estrellas. Además, se contempla el desarrollo de un circuito integrado que cumpla las dimensiones estándar de un nanosatélite tipo CubeSat.

Otro objetivo a futuro es realizar pruebas de precertificación tales como, pruebas de vibraciones mecánicas, termo-vacío y compatibilidad electromagnética, utilizando los laboratorios de la Unidad de Alta Tecnología, para verificar el funcionamiento de los componentes en condiciones ambientales similares a las que se enfrentarán en el espacio.

Apéndices

Apéndice A. Comandos y funciones para el módulo LS_Y201

Lista de comandos para configurar y controlar la cámara

La tabla 8.1 muestra la lista de comandos para la comunicación de la cámara. También muestra los comandos de verificación de la cámara para cada uno de los comandos anteriores.

El último byte del comando para configurar el tamaño de la imagen tiene valores definidos, mostrados en la tabla 8.2. Mientras que la tabla 8.3 muestra los valores definidos para configurar la velocidad de comunicación.

Tabla 8.1: Lista de comandos para la comunicación con el módulo de cámara.

Comando	**Bytes**	**Descripción**
INIT	0x36 0x32 0x35 0x0D 0x0A 0x49 0x6E 0x69 0x74 0x20 0x65 0x6E 0x64 0x0D 0x0A	Inicializar el módulo
RESET	0x56 0x00 0x26 0x00	Resetear el módulo
RESET_ACK	0x76 0x00 0x26 0x00	Comando de verificación para resetear
SET_IMAGE_SIZE	0x56 0x00 0x54 0x01 SIZE	Configura el tamaño de la imagen
SET_BAUD_RATE	0x56 0x00 0x24 0x03 0x01 RATE	Configura la frecuencia de comunicación
SNAPSHOT	0x56 0x00 0x36 0x01 0x00	Captura una imagen
SNAPSHOT_ACK	0x76 0x00 0x36 0x00 0x00	Comando de verificación para capturar
READ_FILE_SIZE	0x56 0x00 0x34 0x01 0x00	Leer el tamaño de la imagen capturada
READ_FILE_SIZE_ACK	0x76 0x00 0x34 0x00 0x04 0x00	Comando de verificación de lectura del tamaño de la imagen capturada
READ_IMAGE	0x56 0x00 0x32 0x0C 0x00 0x0A 0x00 0x00	Leer la imagen
READ_IMAGE_ACK	0x76 0x00 0x32 0x00 0x00	Comando de verificación de lectura de imagen
IMAGE_END	0x56 0x00 0x36 0x01 0x03	Terminar de enviar imagen
IMAGE_END_ACK	0x76 0x00 0x36 0x00 0x00	Comando de verificación de fin de imagen

Tabla 8.2: Lista de valores posibles de configuración para el tamaño de imagen.

Byte	**Resolución de imagen**
0x22	160 x 120
0x11	320 x 240
0x00	640 x 480
0x1D	800 x 600
0x1C	1024 x 768
0x1B	1280 x 960
0x21	1600 x 1200

Tabla 8.3: Lista de valores posibles de configuración para la velocidad de transmisión de datos.

Byte	**Velocidad de transmisión de datos**
0xAE	9,600 bps
0x2A	38,400 bps
0x1C	57,600 bps
0x0D	115,200 bps
0x7E	128,000 bps
0x23	230,400 bps
0x56	256,000 bps
0x58	460,600 bps
0x42	512,000 bps
0x9E	576,000 bps
0x13	600,000 bps
0x87	691,200 bps
0x39	921,600 bps
0x72	1,152,000 bps
0x92	1,209,600 bps

Funciones para capturar y leer una imagen utilizando el protocolo RS-232

El algoritmo 5 muestra los pasos para capturar y leer una imagen utilizando el módulo de cámara y el protocolo de comunicación RS-232.

Algoritmo 5 Pseudocódigo de la comunicación con el módulo LS_Y201 mediante el protocolo RS-232.

procedimiento CAPTURAR Y LEER IMAGEN UTILIZANDO EL MÓDULO LS_Y201 **input** Dirección **int** i

Iniciar el proceso

Resetear el módulo.
Esperar de 2 a 3 ms.

Capturar la imagen

Enviar el comando SNAPSHOT.
Esperar el comando de verificación.
si el comando recibido es diferente a SNAPSHOT_ACK **entonces devolver** 0
si no
Recibir el tamaño de la imagen

Pedir que se envíe la imagen

Enviar el comando READ_IMAGE
Enviar la dirección que se desea leer
Enviar el tamaño de la imagen
si el comando recibido es diferente a READ_IMAGE_ACK **entonces devolver** 0
si no
para i desde 0 hasta el tamaño de la imagen **hacer** imagen(i) = Leer un byte
si i $\neq$ 0 & imagen(i-1) = 0xD9 & imagen(i) = 0xFF **entonces**
Salir del ciclo
fin de si
fin de para
Enviar el comando IMAGE_END
si el comando recibido es diferente a IMAGE_END_ACK **entonces devolver** 0
si no
devolver imagen
fin de si
fin de si
fin de si
fin de procedimiento

Apéndice B. Comandos y funciones para la memoria microSD por el protocolo de comunicación SPI.

Para el diseño de las funciones se utilizó como referencia el trabajo presentado en [41].

Terminales de la microSD para la comunicación con SPI

El protocolo de comunicación utiliza cuatro líneas de transmisión: las terminales *salida del maestro entrada del esclavo* (MOSI), *entrada de maestro salida de esclavo* (MISO), el reloj (SCK) y la señal de activación de la tarjeta (CS). La tabla 8.4 muestra las terminales de la microSD para la comunicación mediante SPI.

Tabla 8.4: Terminales de la memoria microSD para la comunicación mediante el protocolo SPI.

Terminal	**Nombre**
1	CD
2	MOSI
3	VSS
4	VDD
5	SCK
6	VSS
7	MISO
8	RSV
9	RSV

Comandos utilizados para el protocolo SPI

Las tarjetas de memoria microSD tienen comandos predefinidos para controlarlos. La tabla 8.5 muestra una lista de comandos utilizados en las funciones para comunicarse con este dispositivo de almacenamiento.

Tabla 8.5: Lista de comandos para la comunicación con la memoria microSD.

Comando	Bytes	Descripción
CMD0	0x40 0x00 0x00 0x00 0x00 0x95	Configurar memoria a modo SPI
CMD1	0x41 0x00 0x00 0x00 0x00 0xFF	Inicializar memoria
CMD16	0x50 0x00 0x00 0x00 0x00 0xFF	Configurar longitud de bus de datos
CMD17	0x51 0x00 0x00 0x00 0x00 0xFF	Leer datos de la memoria
CMD24	0x58 0x00 0x00 0x00 0x00 0xFF	Escribir datos en la memoria
TOKEN	0xFE	
CRC	0xFF	Byte de verificación
R1	0x00	Respuesta de la memoria
R2	0x05	Respuesta de la memoria

Funciones para controlar la memoria microSD por medio del protocolo SPI

Iniciar comunicación con la memoria

Para iniciar la comunicación con SPI a la memoria el primer paso es enviar el comando CMD0 para configurarla a modo SPI y la memoria debe responder el comando R1. Posterior a eso, se debe inicializar la memoria enviando el comando CMD1 y de igual manera se debe de obtener como respuesta el comando R1. Finalmente, se debe indicar la longitud del tamaño del bloque de comunicación, para ello se debe enviar el comando CMD16, e indicar en el tamaño en los cuatro bytes en medio del comando. En ese caso se utilizaron 512 bytes, por lo que el comando completo que se envió fue: 0x50 0x00 0x00 0x02 0x00 0xFF, y se debe recibir una vez más el comando R1.

Escribir y leer información en la memoria

Para escribir la información que se desea almacenar en la memoria, es necesario enviar el comando CMD24. La dirección en la que se desea almacenar se define por los cuatro bytes centrales del comando. La memoria debe responder con un R1 y posterior a eso, se debe enviar el TOKEN seguido del bus de comunicación con el tamaño definido anteriormente. Al terminar de enviar la información se deben enviar dos CRC y la memoria debe responder con el comando R2 para verificar que la información fue recibida de manera correcta. El proceso está descrito en algoritmo 6.

Para leer información de la memoria se debe enviar el comando CMD17. La dirección que se desea leer se define del byte 2 al 4 del comando. Al recibir el comando, la memoria debe responder con el comando R1 y posteriormente enviar el TOKEN junto con un bus de comunicación del tamaño definido anteriormente. El proceso está descrito en el algoritmo 7.

Algoritmo 6 Pseudocódigo para escribir información en la microSD

procedimiento ESCRIBIR EN MEMORIA MICROSD MEDIANTE EL PROTOCOLO SPI **input** Dirección **int** i **int** Comando[6]

Agregar la dirección de almacenamiento al comando

$$Comando(0) = 0x58 \tag{8.1a}$$
$$Comando(5) = CRC \tag{8.1b}$$

para i que va de 1 hasta 4 **hacer**

$$Comando(i) = Dirección(i-1) \tag{8.2}$$

fin de para

Comunicación con la microSD

Enviar el comando a la microSD
Esperar respuesta de la microSD
si respuesta ≠ R1 **entonces devolver** 0
si no
Enviar el comando TOKEN
Enviar los bytes del paquete de información
Enviar dos veces el comando CRC
Esperar respuesta de la microSD
si respuesta = R2 **entonces devolver** 1
si no
devolver 0
fin de si
fin de si
fin de procedimiento

Algoritmo 7 Pseudocódigo para leer información de la memoria microSD

procedimiento LEER INFORMACIÓN DE LA MEMORIA MICROSD MEDIANTE EL PROTOCOLO SPI **input** Dirección **int** i **int** Comando[6]

Agregar la dirección de almacenamiento al comando

$$Comando(0) = 0x51 \tag{8.3a}$$
$$Comando(5) = CRC \tag{8.3b}$$

para i que va de 1 hasta 4 **hacer**

$$Comando(i) = Dirección(i-1) \tag{8.4}$$

fin de para

Comunicación con la microSD

Enviar el comando a la microSD
Enviar respuesta de la microSD
si respuesta $\neq$ R1 **entonces devolver** 0
si no
Esperar a recibir el comando TOKEN
para i que va de 0 hasta el tamaño definido al inicializar la memoria **hacer**

$$datos(i) = Byterecibidoporlamicro SD \tag{8.5}$$

fin de para
devolver datos
fin de si
fin de procedimiento

Bibliografía

[1] Avid Roman-Gonzalez y Natalia Indira Vargas-Cuentas. "Tecnología aeroespacial en el mundo". En: *Electro i+ d* 1.1 (2012), págs. 48-52.

[2] Secretaria de Economía y col. "Plan de órbita 2.0 Mapa de ruta del sector espacial mexicano". En: (2017).

[3] Tjorven Delabie. "Star tracker algorithms and a low-cost attitude determination and control system for space missions". En: (2016).

[4] Wiley J Larson y James Richard Wertz. *Space mission analysis and design*. Inf. téc. Torrance, CA (United States); Microcosm, Inc., 1992.

[5] Holli Riebeek. *Catalog of Earth Satellite Orbits*. `https://earthobservatory.nasa.gov/features/OrbitsCatalog`. 2009. (Visitado 09-03-2019).

[6] Elizabeth Buchen y Dominic DePasquale. "2014 nano/microsatellite market assessment". En: *SpaceWorks Enterprises* 12 (2014).

[7] NASA CubeSat Launch Initiative y col. *CubeSat 101 Basic Concepts and Processes for First-Time CubeSat Developers*. 2017.

[8] Carl Christian Liebe. "Pattern recognition of star constellations for spacecraft applications". En: *IEEE Aerospace and Electronic Systems Magazine* 7.6 (1992), págs. 34-41.

[9] Randall V Martin. "Satellite remote sensing of surface air quality". En: *Atmospheric environment* 42.34 (2008), págs. 7823-7843.

[10] Pawan Gupta y col. "Satellite remote sensing of particulate matter and air quality assessment over global cities". En: *Atmospheric Environment* 40.30 (2006), págs. 5880-5892.

[11] Aaron Van Donkelaar, Randall V Martin y Rokjin J Park. "Estimating ground-level PM2. 5 using aerosol optical depth determined from satellite remote sensing". En: *Journal of Geophysical Research: Atmospheres* 111.D21 (2006).

[12] Yang Liu y col. "Estimating ground-level PM2. 5 in the eastern United States using satellite remote sensing". En: *Environmental science & technology* 39.9 (2005), págs. 3269-3278.

[13] Kali E Sawaya y col. "Extending satellite remote sensing to local scales: land and water resource monitoring using high-resolution imagery". En: *Remote sensing of Environment* 88.1-2 (2003), págs. 144-156.

[14] C Albergel y col. "Monitoring multi-decadal satellite earth observation of soil moisture products through land surface reanalyses". En: *Remote Sensing of Environment* 138 (2013), págs. 77-89.

[15] Lingli Wang y John J Qu. "Satellite remote sensing applications for surface soil moisture monitoring: A review". En: *Frontiers of Earth Science in China* 3.2 (2009), págs. 237-247.

[16] Anton Vrieling. "Satellite remote sensing for water erosion assessment: A review". En: *Catena* 65.1 (2006), págs. 2-18.

[17] Anton Vrieling. *Mapping erosion from space*. 2007.

[18] Tiit Kutser y col. "Monitoring cyanobacterial blooms by satellite remote sensing". En: *Estuarine, Coastal and Shelf Science* 67.1-2 (2006), págs. 303-312.

[19] DS Boyd y FM Danson. "Satellite remote sensing of forest resources: three decades of research development". En: *Progress in Physical Geography* 29.1 (2005), págs. 1-26.

[20] Thomas L Bell. "A space-time stochastic model of rainfall for satellite remote-sensing studies". En: *Journal of Geophysical Research: Atmospheres* 92.D8 (1987), págs. 9631-9643.

[21] David M Tralli y col. "Satellite remote sensing of earthquake, volcano, flood, landslide and coastal inundation hazards". En: *ISPRS Journal of Photogrammetry and Remote Sensing* 59.4 (2005), págs. 185-198.

[22] Laurence C Smith. "Satellite remote sensing of river inundation area, stage, and discharge: A review". En: *Hydrological processes* 11.10 (1997), págs. 1427-1439.

[23] Sarah H Parcak. *Satellite remote sensing for archaeology*. Routledge, 2009.

[24] Malak Samaan y Stephan Theil. "Development of a low cost star tracker for the SHEFEX mission". En: *Aerospace science and technology* 23.1 (2012), págs. 469-478.

[25] M Shayan Arani, A Toloei y Z Eghbaleh. "A geometric star identification algorithm based on triple triangle pattern". En: *2015 7th International Conference on Recent Advances in Space Technologies (RAST)*. IEEE. 2015, págs. 81-85.

[26] Carl Ch Liebe. "Star trackers for attitude determination". En: *IEEE Aerospace and Electronic Systems Magazine* 10.6 (1995), págs. 10-16.

[27] Chen Zhang, Chaoyang Chen y Xubang Shen. "A new guide star selection algorithm for star tracker". En: *Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No. 04EX788)*. Vol. 6. IEEE. 2004, págs. 5445-5449.

[28] Hye-Young Kim y John L Junkins. "Self-organizing guide star selection algorithm for star trackers: thinning method". En: *Proceedings, IEEE Aerospace Conference*. Vol. 5. IEEE. 2002, págs. 5-5.

[29] Jayinder Singh. *Low Cost Star Tracker Software*. `https://technology.nasa.gov/patent/TOP2-265`. 2017. (Visitado 01-04-2019).

[30] MD Pham y col. "A star pattern recognition algorithm for satellite attitude determination". En: *2012 IEEE Symposium on Industrial Electronics and Applications*. IEEE. 2012, págs. 236-241.

[31] Mehta Deval Samirbhai, Shoushun Chen y Kay Soon Low. "A high accuracy star tracker using running sequential angular match technique". En: *2016 IEEE Region 10 Conference (TENCON)*. IEEE. 2016, págs. 3691-3694.

[32] Deval Samirbhai Mehta y Shoushun Chen. "A spearman correlation based star pattern recognition". En: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2017, págs. 4372-4376.

[33] Gottlieb (1978) In: JR Wertz. *Spacecraft attitude determination and control*. Vol. 73. Springer Science & Business Media, 2012, págs. 257-266.

[34] Edward J Groth. "A pattern-matching algorithm for two-dimensional coordinate lists". En: *The astronomical journal* 91 (1986), págs. 1244-1248.

[35] Curtis Padgett y Kenneth Kreutz-Delgado. "A grid algorithm for autonomous star identification". En: *IEEE Transactions on Aerospace and Electronic systems* 33.1 (1997), págs. 202-213.

[36] Hyunjae Lee y Hyochoong Bang. "Star pattern identification technique by modified grid algorithm". En: *IEEE Transactions on Aerospace and Electronic Systems* 43.3 (2007), págs. 1112-1116.

[37] Meng Na, Danian Zheng y Peifa Jia. "Modified grid algorithm for noisy all-sky autonomous star identification". En: *IEEE Transactions on Aerospace and Electronic Systems* 45.2 (2009), págs. 516-522.

[38] Andrew E. Kalman. *Development Opportunities within the CubeSat Kit Architecture*. `http://mstl.atl.calpoly.edu/~bklofas/Presentations/DevelopersWorkshop2008/7-CubeSatKit-Andrew_Kalman.pdf`. 2008. (Visitado 01-05-2019).

[39] The Astronomy Nexus. *Catalog of Earth Satellite Orbits*. `http://www.astronexus.com/hyg`. 2006. (Visitado 30-03-2019).

[40] OpenCV Team. *OpenCV*. `https://opencv.org`. 2019. (Visitado 07-04-2019).

[41] Carlos Alberto Henao y EDISON DUQUE CARDONA. "Manejo de una memoria SD/MMC con un pic16f87x." En: *Scientia et technica* 16.44 (2010).