Universidad Nacional Autónoma de México

Centro de Física Aplicada y Tecnología Avanzada

ESTIMACIÓN DE PROFUNDIDAD DE UN OBJETO USANDO VISIÓN PERSPECTIVA

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN TECNOLOGÍA

PRESENTA

RICARDO NIETO FUENTES

Asesor:

Dr. Yu Tang Xu

Co-asesor:

M.C. Rafael Guadalupe Chávez Moreno

2015, Querétaro Qro.

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Jurado

Presidente	Dr. Remy Fernand Ávila Foucat
Vocal	Dra. Beatriz Marcela Millán Malo
Secretario	Dr. Ángel Luis Rodríguez Morales
1º Suplente	Dr. Yu Tang Xu
2º Suplente	M. C. Rafael Guadalupe Chávez Moreno

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
"Estimación de profundidad de un objeto usando visión perspectiva"
que realizó el (la) pasante
Ricardo Nieto Fuentes
con número de cuenta: 410059203, 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. Remy Fernand Ávila Foucat	
VOCAL	Dra. Beatriz Marcela Millán Malo	
SECRETARIO	Dr. Angel Luis Rodríguez Morales	
1er. SUPLENTE	Dr. Yu Tang Xu	
2º SUPLENTE	M. C. Rafael Guadalupe Chávez Moreno	

Atentamente
"POR MI RAZA HABLARÁ EL ESPÍRITU"

UNAM, Campus Juriquilla, Qro. a 02 de Octubre de 20 15.

Agradecimientos

A mis padres por el amor, comprensión y apoyo incondicional que me ha permitido dar un paso más en la vida.

A mis asesores por su tiempo, paciencia, dedicación y consejos en cada etapa del presente trabajo.

A mí jurado por el tiempo dedicado en la revisión y evaluación del trabajo.

A mi familia por estar a mi lado durante cada momento.

A todas las personas que me brindaron su apoyo.

Índice

Índice I

Índice de Figuras III

Índice de Tablas IV

Resumen 1

1 Introducción 2

1.1 Visión 2

1.1.1 Aplicaciones 3

1.1.2 Estimación de la profundidad 5

1.1.2.1 Visión estereoscópica 5

1.1.2.2 Visión monocular 7

1.2 Justificación 8

1.3 Hipótesis 9

1.4 Objetivos 9

1.4.1 Objetivo General 9

1.4.2 Objetivos Particulares 9

2 Procesamiento de señales 11

2.1 Imagen Digital 11

2.1.1 Formación de la imagen 11

2.1.2 Adquisición de la imagen 13

2.1.3 Procesamiento de imágenes 17

2.1.3.1 Modelo de color 17

2.1.3.2 Segmentación de la imagen 19

2.1.3.3 Seguimiento de objetos 20

2.1.3.4 Parámetros ópticos de la cámara 23

2.2 Navegación inercial 25

2.2.1 Obtención de la posición 27

2.2.1.1 Métodos numéricos 29

3 Observador de profundidad 32

3.1 Modelo de visión perspectiva 32

3.2 Observador del sistema de perspectiva 35

3.2.1 Estimador de orden reducido 36

3.2.2 Estimador orden completo 38

4 Diseño experimental 39

4.1 Seguimiento del objeto 41

4.2 Obtención de aceleración y velocidad 46

4.3 Determinación de la profundidad 49

4.4 Plataforma experimental 51

5 Resultados 54

5.1 Simulaciones 54

5.2 Software 61

5.3 Estimación de la profundidad 68

6 Conclusiones 75

7 Referencias 77

Índice de Figuras

Figura 1: Visión binocular 6
Figura 2: Visión monocular 8
Figura 3: Modelo de perspectiva central 13
Figura 4: Modelo de lente delgado 14
Figura 5: Campo de visión 15
Figura 6: Filtro de Bayer 17
Figura 7: Espacios de color 19
Figura 8: Sensores inerciales 26
Figura 9: Modelo de visión perspectiva 33
Figura 10 : Diagrama a bloques de la plataforma experimental 39
Figura 11: Algoritmo de seguimiento del objeto 42
Figura 12: Vector desplazamiento 45
Figura 13: Algoritmo de navegación inercial 46
Figura 14: Algoritmo del observador de estado 50
Figura 15: Plataforma experimental 51
Figura 16: Esquema interfaz gráfica 53
Figura 17: Validación del observador 55
Figura 18: Simulación con ruido 56
Figura 19: Simulación de la dinámica 57
Figura 20: Simulación de la dinámica con ruido 58
Figura 21: Coordenadas del dispositivo móvil 66
Figura 23: Resultado de la rutina de seguimiento 68
Figura 24: Procesamiento de los datos adquiridos 69
Figura 25: Estimación en el dispositivo móvil 70
Figura 26: Estimación a 1.6 m 71
Figura 27: Estimación a 0.2 m 72

Índice de Tablas

Tabla 1: Error RMS de Simulaciones 60
Tabla 2: Variación del error con K 60
Tabla 3: Variación del error con K_z 60
Tabla 4: Experimentos variando la distancia 73
Tabla 5: Experimentos variando resolución 74

Resumen

Una de las áreas de investigación de la visión artificial es la estimación de la posición relativa entre una cámara y los objetos que se encuentran en su campo de visión; la cual se requiere en aplicaciones de reconstrucción tridimensional o navegación. Entre las técnicas para realizar esta estimación, podemos encontrar la visión perspectiva la cual utiliza la velocidad y aceleración de la cámara en conjunto con las coordenadas de los objetos en una secuencia de imágenes; para estimar la distancia mediante un observador de estado (modelo matemático que permite conocer la evolución del sistema). En el presente trabajo de tesis se realizó una aplicación que permite utilizar los sensores de los dispositivos móviles actuales para realizar la estimación de la profundidad utilizando un observador de estado. De esta manera se tiene un sistema de bajo costo, preciso y de interfaz simple para el usuario técnico o casual. La aplicación programada obtiene el movimiento de la cámara mediante los sensores inerciales y la imagen es capturada mediante la cámara del dispositivo. Se realiza un seguimiento del objeto durante el movimiento del dispositivo y se emplea el observador para estimar la distancia. Para realizar las pruebas se utilizó un teléfono móvil Xperia™ Z1 D5503, se realizaron mediciones de hasta tres metros en las cuales se obtuvieron errores del 5%.

1 Introducción

1.1 Visión

La luz solar ha sido un factor importante en la evolución de los seres vivos. Esto se ve reflejado en el desarrollo de las cascadas genéticas que determinan la formación de los ojos y la complejidad, diversidad, forma, tamaño y diseño de estos. Existen proteínas sensibles a la luz, previas a la formación de la primera estructura semejante a un ojo, que muestran la importancia evolutiva que tiene la luz solar como fuente de información para los seres vivos [1].

Ver es descubrir qué nos rodea y ubicarlo en el espacio, a partir de la imagen que percibimos mediante la luz solar. La visión es una tarea de procesamiento y representación de la información captada por los ojos, que nos permite conocer la estructura tridimensional del entorno, interpretar la escena e inferir propiedades de los objetos observados. [2, pp. 3-7]

Dada la importancia que tiene la visión para el humano en sus actividades, resulta natural el desarrollo de tecnologías para realizar acciones repetitivas, peligrosas o inaccesibles, en las que se requiere de las capacidades de la vista. El problema radica en cómo lograr que un ordenador sea capaz de realizar la interpretación de la escena y responder en el tiempo requerido. Para resolver este problema, a finales de los 50s surge la visión artificial, la cual es la ciencia encargada de describir el mundo que vemos a través de una imagen [3, p. 3]. Para ello se emplean métodos estadísticos con el fin de extraer e interpretar la información, mediante modelos físicos, geométricos y de teoría del aprendizaje [4, p. 17].

1.1.1 Aplicaciones

La visión artificial moderna tiene sus origines en la década de los 60s con el reconocimiento de patrones. En un inicio la mayoría de las investigaciones se enfocaba en el reconocimiento óptico de caracteres (OCR) [5]. Con el paso del tiempo la investigación se diversificó desarrollándose avances en áreas tales como:

- Medicina: Se desarrollan sistemas para la extracción de información clínica que permita el diagnóstico y tratamiento de enfermedades. [6].

- Agricultura: Se realizan avances para la automatización de la cosecha, enfrentando tres problemas básicos: la navegación del sistema por el cultivo, la identificación de frutos, su selección y recolección. [7]

- Sistemas de vigilancia y monitoreo: En monitoreo de tránsito vehicular se han desarrollado técnicas de conteo de vehículos, reconocimiento de placas y detección de incidentes [8]. En vigilancia se realiza investigación en la detección de movimiento inusual, seguimiento y etiquetado. Para lo que se emplean modelos estadísticos del fondo y los objetos [9].

- Reconocimiento de objetos: El problema fundamental es determinar las características que definen un objeto, de manera que se le pueda identificar en una imagen. La identificación es afectada por factores como la posición, escala, obstrucciones parciales e iluminación de la escena, donde todos estos factores pueden variar en el

tiempo y en el espacio. Un ejemplo del desarrollo en esta área es [10] en donde se propone el algoritmo SIFT (Scale Invariant Feature Transform) para obtener identificadores invariantes en escala y orientación, permitiendo la detección, incluso con objetos parcialmente obstruidos. Otro ejemplo es el desarrollo de aplicaciones para la detección y análisis, como lo es el programa para aplicaciones biológicas CellProfile [11] el cual detecta e identifica células en fotomicrografías.

- Reconstrucción 3D: El objetivo de esta área es obtener la estructura tridimensional de los objetos a partir de una serie de imágenes. Se han desarrollado técnicas para la reconstrucción tridimensional del movimiento humano [12] y la reconstrucción e interacción tridimensional de los humanos con los objetos [13]. Así como la medición de dimensiones tridimensionales en tiempo real usando la técnica de proyección de franjas [14].

- Navegación y control: El objetivo es controlar la posición de un robot o vehículo usando la información visual disponible de una o más cámaras [15, p. 3]. Para esto se utilizan técnicas de reconocimiento y reconstrucción tridimensional con el objetivo de determinar la ubicación espacial del sistema. Un ejemplo es [16] en donde la navegación de un robot se realiza calculando su ubicación mediante una segmentación del entorno, empleando una red deformable de pixeles basada en una región, la cual está formada por un conjunto de vértices y aristas; la posición del robot se obtiene mediante relaciones de área o mediante los vértices.

Otras aplicaciones de la visión que podemos destacar son: inspección de maquinaria, líneas de producción, seguridad automotriz, biometría y detección de rostros [3, p. 5].

Conforme han evolucionado los sistemas basados en visión y aumentado su eficiencia y exactitud, se han convertido en un sensor versátil en la medición de posición, movimiento y estructura del entorno.

1.1.2 Estimación de la profundidad

Una de las áreas que se encuentra en desarrollo es la estimación de la profundidad, problema que se encuentra estrechamente ligado a la reconstrucción 3D, al reconocimiento de objetos y la navegación. La estimación de profundidad consiste en reconstruir el movimiento relativo entre una cámara y el objeto observado por ésta, obteniendo las coordenadas tridimensionales del objeto. Las técnicas para determinar la posición pueden ser clasificadas en activas y pasivas. Las activas consisten en enviar y recibir algún estimulo hacia la escena. Ejemplos de estas técnicas son: ultrasonido, tiempo de vuelo, sistemas de triangulación [17] y proyección de franjas. Las técnicas pasivas son aquellas no invasivas que no interactúan con la escena; dichas técnicas a su vez pueden ser clasificadas en visión estereoscópica y monocular (perspectiva) [18].

1.1.2.1 Visión estereoscópica

La visión estereoscópica requiere de dos cámaras colocadas en diferentes posiciones, de ese modo se obtienen dos imágenes diferidas (espacialmente) de la misma escena. La distancia es calculada con el desfase entre puntos correspondientes en las dos imágenes.

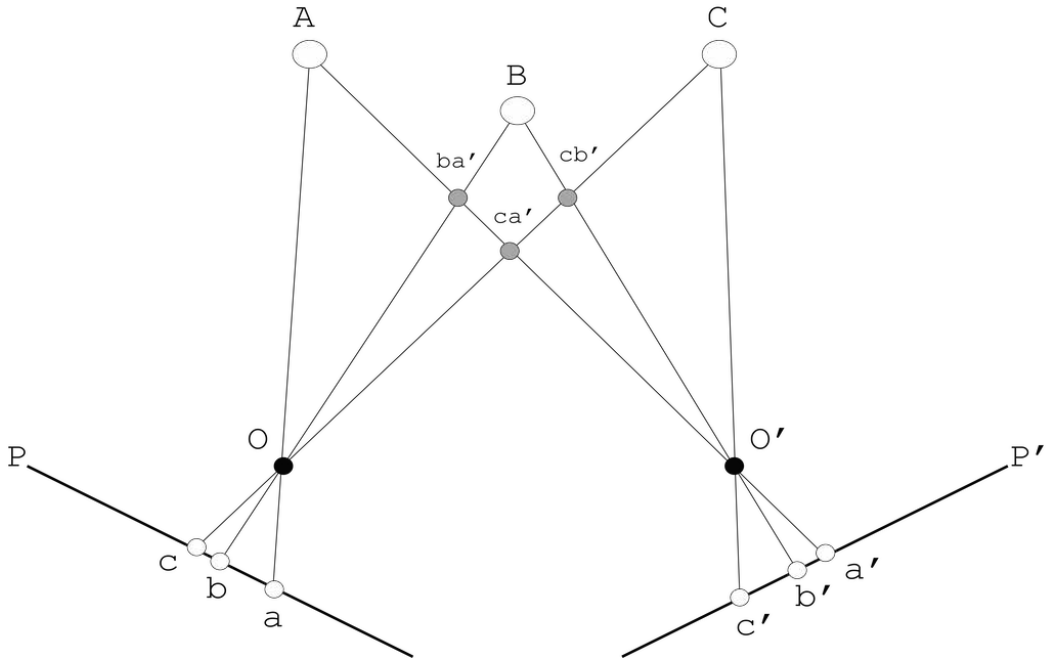

Figura 1: Visión binocular

La visión binocular consiste en dos planos "P" y "P′", en los cuales se proyectan los puntos "A", "B" y "C" mediante los rayos que pasan por las pupilas "O" y "O′".

Esta visión involucra dos procesos: la fusión de las características de dos o más cámaras y la reconstrucción tridimensional de la escena. El emparejamiento de los puntos correspondientes, en principio, puede obtenerse de la intersección de los rayos de luz que pasan por el centro de la pupila [4, pp. 197-198].

En la Figura 1 se ejemplifica el sistema de visión binocular, en él se cuenta con dos planos de imagen "P" y "P′" y sus respectivas pupilas "O" y "O′". Para realizar la reconstrucción tridimensional se requiere de identificar las proyecciones de tres puntos "A", "B" y "C" en cada plano e identificar la correspondencia de las proyecciones "a", "b", "c", entre el plano "P" y el plano "P′". Los puntos "ba′", "ca′" y "cb′" muestran correspondencias erróneas. En una imagen se encuentran millones de pixeles con miles de características, lo que dificulta la búsqueda de los puntos

correspondientes, además el emparejamiento cuenta también con una alta sensibilidad a la textura de las superficies. La resolución de esta técnica decrece al aumentar la distancia al objeto en relación a la distancia entre los planos P y P’.

1.1.2.2 Visión monocular

La visión monocular, también llamada visión perspectiva, emplea una sola cámara y la información tridimensional se obtiene a partir de una secuencia de imágenes adquiridas durante el movimiento relativo de ésta. Para la reconstrucción se requiere de realizar un seguimiento del objeto en la secuencia de imágenes, por lo que en este tipo de visión son necesarios la segmentación de la imagen, el uso de algoritmos de seguimiento y conocer la posición de la cámara [16]. Este tipo de visión es empleada en aplicaciones de navegación y control debido a que reduce los costos de hardware y procesos de calibración.

La estimación de la profundidad usando visión perspectiva puede realizarse mediante una extensión de la visión binocular, empleando la relación entre dos imágenes, “antes y después” del desplazamiento de la cámara, o algoritmos que emplean el movimiento continuo de la cámara y de los puntos, en la secuencia de imágenes, lo cual mejora la precisión de la estimación [18]. Ejemplos de este tipo de estimaciones son la adquisición de la información tridimensional mediante la medición de relaciones de desplazamiento de puntos en la imagen, o de relaciones de área [16] y estimadores basados en el modelo de perspectiva central y la dinámica de la cámara [19].

En la Figura 2 se muestra un esquema de la visión monocular, en donde "Q" es una matriz de transformación homogénea del plano "P". Esto produce una proyección de cada punto para cada instante de tiempo. A diferencia de la visión binocular el problema no es realizar una correspondencia entre las proyecciones de los puntos, sino obtener la transformación "D" en el plano de imagen, lo que permite obtener las coordenadas tridimensionales de los puntos "A" y "B".

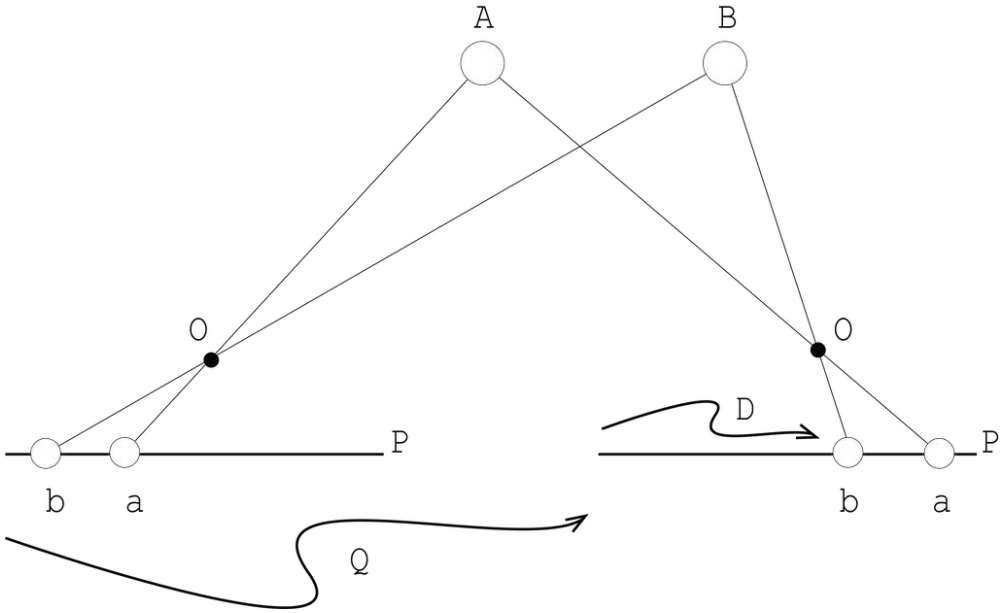

Figura 2: Visión monocular

Se tienen dos puntos "A" y "B" los cuales se proyectan en el plano "P" a través de la pupila "O". El plano "P" se transforma mediante "Q" el cual produce una transformación "D" en las proyecciones "b" y "a".

1.2 Justificación

En aplicaciones como vigilancia, control y navegación se requiere de conocer la dinámica del sistema. Por ello es importante el desarrollo de nuevos sistemas y algoritmos de estimación de profundidad. En el presente trabajo se propone desarrollar un sistema de estimación de profundidad que

facilite la evaluación de algoritmos de estimación que permitan reconstruir la evolución del sistema en el tiempo.

1.3 Hipótesis

Es posible implementar un sistema de visión perspectiva y navegación inercial en un dispositivo móvil. El sistema permitirá la evaluación de algoritmos de estimación de profundidad y se obtendrá la posición relativa de un objeto al dispositivo. Lo anterior sin requerir de sistemas externos para el procesamiento y despliegue de la información.

1.4 Objetivos

1.4.1 Objetivo General

Implementar un sistema móvil de visión perspectiva y navegación inercial mediante una cámara, una unidad de medición inercial y un observador de estado, con el fin de estimar la profundidad a un objeto en el campo de visión del sistema.

1.4.2 Objetivos Particulares

- Desarrollar un programa que adquiera y procese una imagen digital y que permita realizar la detección y seguimiento de un objeto para obtener sus coordenadas de proyección.
- Desarrollar un programa que permita adquirir la aceleración lineal de la cámara mediante sensores

inerciales, realizando el procesamiento para determinar la velocidad y posición de la cámara.

- Elaborar un programa que permita la implementación de un observador de estado del modelo de visión perspectiva, y mediante la dinámica del sistema y el desplazamiento del objeto en la imagen adquirida, estime la información espacial relativa del objeto a la cámara.

2 Procesamiento de señales

2.1 Imagen Digital

Para simular la vista se han desarrollado las cámaras, las cuales pueden considerarse como dispositivos radiométricos que miden la energía lumínica, cuantificando su intensidad y clasificando su tono. La visión por computadora estudia las imágenes digitales, formadas cuando los rayos de luz reflejados en un objeto inciden en el sensor fotovoltaico de la cámara, que se encuentra colocado en el plano de imagen. Este sensor capta y transforma la intensidad luminosa en valores discretos colocados en un arreglo bidimensional [20, p. 46] los cuales pueden ser procesados para extraer información del objeto o escena observada por la cámara.

2.1.1 Formación de la imagen

Para analizar una imagen digital es necesario contar con un modelo que describa la proyección de los objetos tridimensionales en el plano de imagen. Un modelo sencillo y ampliamente utilizado en visión es el modelo de perspectiva central o "pin-hole". En este modelo la imagen es formada por los rayos de luz que atraviesan un punto (pupila) y pasan por el plano de imagen.

En la Figura 3 se esquematiza el modelo de perspectiva central, en donde dos postes "A" y "B" se encuentran a una distancia 2ϵ y 4ϵ respectivamente. La distancia focal ϵ es la distancia a la que se encuentra el plano "P" en donde se forma la imagen. La altura del poste "A" es el doble que la del poste "B"; sin embargo, la altura de la proyección "a"

es la misma que la de "b". Para entender este escalamiento se considera a la pupila "O" como un punto, de esta forma todos los rayos de luz pasan por "O" y son proyectados en el plano "I". Los puntos "B", "b" y "O" son colineales, por lo que podemos expresar el vector que va de la pupila a la proyección, como un escalamiento del vector que va de la pupila al punto, es decir:

$$\overrightarrow{Ob} = \lambda\,\overrightarrow{OB}. \qquad (1)$$

Para cualquier número $\lambda \in R^3$, podemos obtener [4, pp. 3-7]:

$$\begin{cases} y_1 = \lambda\, x_1 \\ y_2 = \lambda\, x_2 \\ \epsilon = \lambda\, x_3 \end{cases} \leftrightarrow \lambda = \frac{y_1}{x_1} = \frac{y_2}{x_2} = \frac{\epsilon}{x_3} \;\therefore\; \begin{cases} y_1 = \epsilon \frac{x_1}{x_3}, \\ y_2 = \epsilon \frac{x_2}{x_3} \end{cases} \qquad (2)$$

Donde "y_1" y "y_2", son las coordenadas de la proyección "b" y "x_1", "x_2", "x_3" son las coordenadas del punto "B" con respecto a "O".

Con la ecuación (2) se obtiene que la coordenada "y_2" de la proyección "b" es $\frac{1}{2}x_2$ y la coordenada "y_2" de la proyección "a" es $\frac{1}{4}\,2\,x_2$, debido a que A = 2B. Lo anterior muestra por que las proyecciones "a" y "b" son de la misma altura.

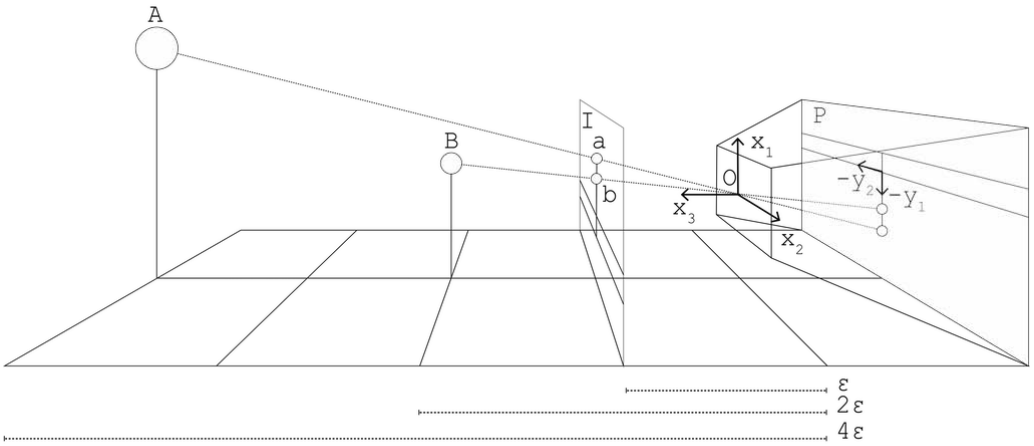

Figura 3: Modelo de perspectiva central

En la figura se muestran dos postes "A" y "B" que se encuentran a una distancia 4ϵ *y* 2ϵ *respectivamente de la pupila "O". Las proyecciones "a" y "b" sobre los planos "I" y "P" son de la misma altura debido a la perspectiva central*

2.1.2 Adquisición de la imagen

La perspectiva central es un modelo ideal; sin embargo, cualquier pupila tiene un tamaño finito, por lo que en realidad cada punto de la proyección está iluminado por un cono de luz. Entre más pequeña sea la apertura de la pupila entrará menos luz, y así se tendrá una mejor nitidez pero pueden existir efectos de difracción. Por el contrario, entre más grande sea la apertura más luz entrará, pero tendremos una imagen borrosa. Por esta razón las cámaras cuentan con lentes, los cuales permiten mantener la nitidez de la proyección mientras captan más luz al abarcar un área más grande.

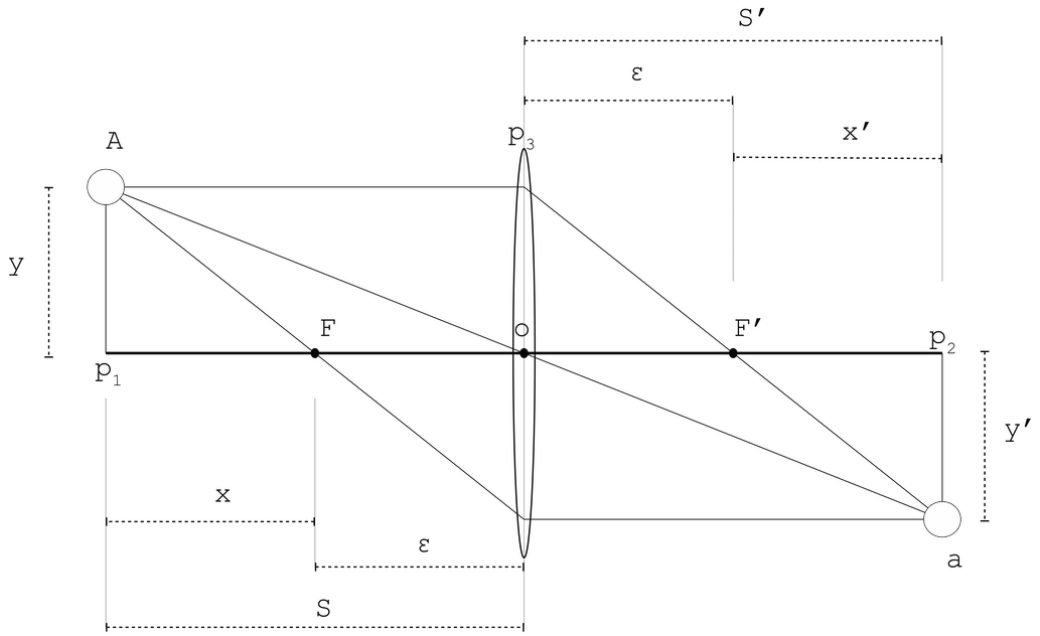

Figura 4: Modelo de lente delgado

En la figura se muestra los dos puntos focales "F" y "F′" en los cuales convergen todos los rayos de luz paralelos al eje óptico. El objeto "A" se encuentra a una distancia "S" y tiene una altura "y", mientras su proyección "a" se encuentra a una distancia "S′" y una altura "y′". "ε" es la distancia focal mientras que "x" y "x′" son las distancias del objeto y la imagen respectivamente al punto focal.

En la Figura 4 se muestra el modelo más simple de una lente. Se observa el objeto "A" que se encuentra a una distancia "S" con altura "y" y su imagen "a" formada a una distancia "S′" con altura "y′". Este modelo ésta regido por la ecuación de la lente delgada

$$\frac{1}{f} = \frac{1}{S} + \frac{1}{S'}. \qquad (3)$$

Ecuación que se obtiene empleando la semejanza de los triángulos "A p_1 o" y "a p_2 o", al igual que la semejanza entre "p_3 o F′" y "a p_2 F′" [21, pp. 160-163]. Las ecuaciones que determinan la posición de "A" y "a" son las mismas que la ecuación (2) si consideramos $\epsilon = S'$

Emplear lentes permite tener un mejor control de la formación de la imagen pero introduce aberraciones de la imagen. Los cinco tipos principales son: esférica, astigmática, curvatura de campo, coma y distorsión. Las primeras cuatro consisten en una degradación de la imagen al desenfocarla, mientras que la distorsión es una modificación de la forma de la imagen completa. Esta deformación se debe a que distintas partes de la lente tienen diferentes distancias focales [4, pp. 10-12].

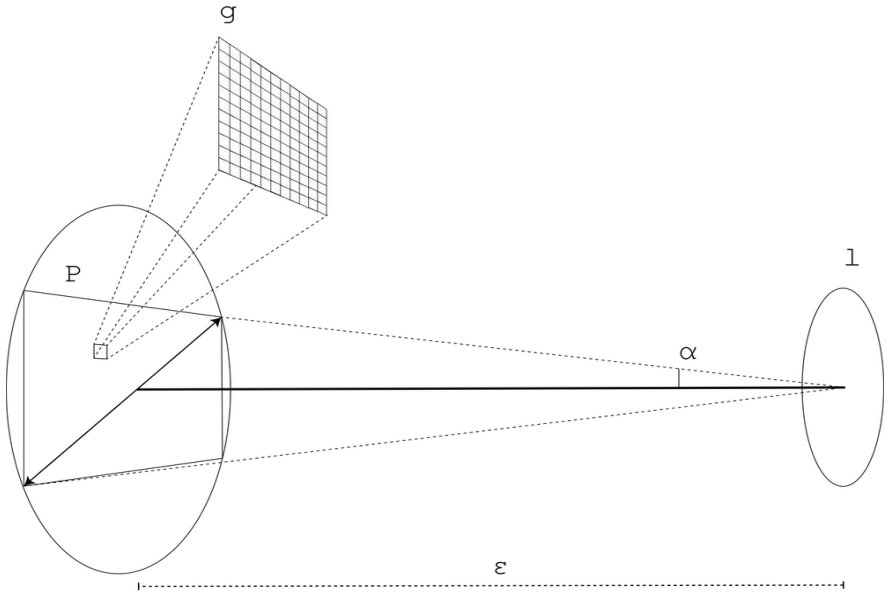

Figura 5: Campo de visión

En la imagen se muestra el campo de visión formado por el ángulo "α" el cual es proyectado sobre la región "P" a la distancia "ϵ".

En la Figura 5 se muestra el campo de visión de la cámara, en donde observamos la porción del espacio que es proyectada en el plano "P" por la lente "l". Para poder utilizar la proyección es necesario almacenarla, en las cámaras digitales el plano "P" es un sensor que tiene una matriz de unidades fotosensibles (pixeles), las cuales se esquematizan en la imagen como la ampliación "g". Los pixeles transforman la intensidad luminosa en un voltaje que después es

digitalizado en un valor numérico mediante un conversor analógico-digital y posteriormente es almacenado en una memoria.

Existen esencialmente dos tipos de sensores, los dispositivos de carga acoplada o CCD (Charge Coupled Device) y los semiconductores complementarios de óxido metálico o CMOS (Complementary Metal Oxide Semiconductor). Ambos funcionan bajo el mismo principio (efecto fotoeléctrico), la diferencia radica en que los CMOS cuentan con una amplificación local de la señal producida en el pixel. Actualmente los sensores CMOS son los más comunes en las cámaras.

Los sensores únicamente captan la intensidad, por lo que, para poder obtener una imagen en color, es necesario realizar un filtrado para distintas longitudes de onda, este filtro es superpuesto sobre los pixeles del sensor para captar una longitud especifica (Figura 6). Uno de los filtros más utilizados es el de Bayer, el cual emplea una matriz de un patrón repetitivo de 2x2 del rojo, verde y azul. Para obtener los valores de cada canal de color se realiza una interpolación de los colores adyacentes, de manera que con una interpolación lineal, si tenemos un pixel con el filtro verde, su valor de rojo se obtendrá con el promedio de los filtros rojos adyacentes [22, p. 482].

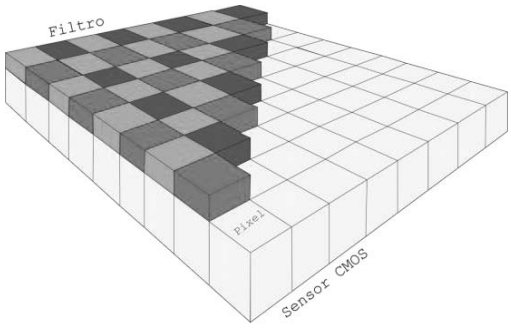

Figura 6: Filtro de Bayer

Para obtener una imagen a color las cámaras emplean un filtro. En la imagen superior se esquematiza el filtro de Bayer superpuesto al sensor CMOS o CCD. Véase explicación en el texto.

2.1.3 Procesamiento de imágenes

Una vez que se cuenta con la imagen digital, el siguiente paso es obtener información relevante a partir de los valores discretos de cada pixel. Para ello se emplean modelos que relacionan las características del mundo real con los valores de intensidad y color de los pixeles. Para el procesamiento de las imágenes a veces se requiere de un pre-procesado, una serie de filtrados para mejorar la calidad de la imagen, rotaciones, escalamientos, etc. Eso facilita la interpretación de la información en la imagen. Después viene la etapa de procesado, en donde se extraen las características que son de interés y finalmente la interpretación y etiquetado de los datos obtenidos a partir de la imagen.

2.1.3.1 Modelo de color

En los inicios de la visión artificial, el procesamiento de las imágenes digitales se realizaba en escala de grises, debido principalmente a la limitación en las capacidades de

cómputo. Al contarse con equipos cada vez más potentes, el análisis de las imágenes a color permitió un avance en los estudios de la visión artificial, por ejemplo, en la segmentación, análisis de texturas, localización de objetos y emparejamiento de puntos de interés.

Para interpretar las imágenes a color se requiere de un modelo o espacio que defina lo que es un color para la computadora. Este espacio se define mediante una base de "n" vectores cuyas combinaciones lineales generan todos los elementos. La escala de grises sería un espacio de una dimensión. Las imágenes digitales utilizan en su mayoría un espacio tridimensional, entre los más comunes encontramos: RGB, HSI y YUV.

En la Figura 7 se representan los diferentes espacios de color. El modelo RGB (por sus siglas en inglés: red, green, blue) es un modelo que descompone un color en tres: rojo, verde y azul. Este es un subespacio euclidiano, en él, el negro se encuentra en el origen y la escala de grises es una diagonal que va del negro al blanco.

HSI y YUV son modelos que se asemejan más a cómo los humanos perciben el color. Sus componentes se dividen en tono, saturación e intensidad. El tono se refiere a la longitud de onda dominante percibida, cuando se hace referencia al color "rojo" describimos el tono. La intensidad es la cantidad de luz recibida. La saturación es la cantidad de blanco en el color. Esta separación de la luminosidad y los valores cromáticos es la que hace que se prefiera este tipo de espacio de colores en aplicaciones de visión [23] [24]. El modelo HSI puede visualizarse como un cono, en el cual se representa el tono mediante un ángulo "H", la saturación como una distancia "S" con respecto al eje de intensidad "I". Mientras que YUV es un cubo similar al RGB, en el cual

la intensidad se encuentra dada por el eje “Y” y los valores cromáticos por “U” y “V”.

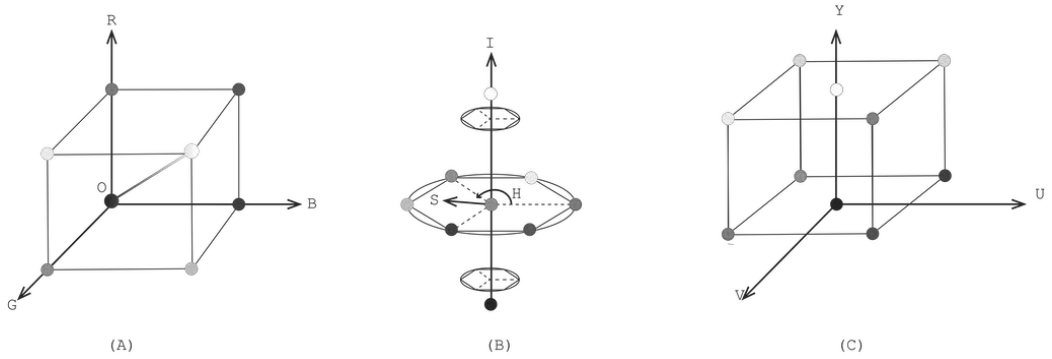

Figura 7: Espacios de color

En la imagen se muestran tres espacios de color (A) es el espacio RGB, (B) es el HSI y (C) representa el espacio YUV. Véase explicación en el texto.

Dependiendo de la aplicación uno u otro espacio de color puede presentar ventajas o desventajas sobre los demás, por ello se realizan mapeos entre los diferentes espacios dependiendo de las necesidades de cada caso. Al procesar una imagen a color se requiere asignar una clasificación a cada pixel con base en alguna característica, lo que permite identificar regiones de la imagen en la cual los pixeles sean homogéneos, lo que da paso a uno de los pilares del procesamiento de imágenes, la segmentación.

2.1.3.2 Segmentación de la imagen

La segmentación es el proceso de dividir una imagen en regiones que deben ser homogéneas con respecto a una característica. Es el primer paso en el procesamiento de bajo nivel de imágenes, un algoritmo de segmentación debe requerir un consumo bajo de recurso computacional y su

inicialización debe ser sencilla e intuitiva [25]. Estos algoritmos se clasifican en tres categorías principales:

- Segmentación por características de espacio: Se asume que los objetos tienen una propiedad constante. Diferentes objetos tendrán diferentes características y aparecen como clústeres o nubes de puntos en la imagen, o bien como picos en sus histogramas.

- Segmentación en dominio de imagen: La segmentación por características de un espacio no asegura que las regiones sean coherentes superficialmente. En un objeto se espera que todos los puntos se encuentren en una zona compacta que define al objeto. Para solventar eso se emplean técnicas que basadas en una división de la imagen, van combinando las divisiones contiguas comparando su homogeneidad.

- Segmentación por características físicas: Se introducen modelos de la interacción de la materia con la luz, en combinación con las segmentaciones anteriores.

Una descripción más detallada sobre el problema de segmentación puede encontrarse en [26]

2.1.3.3 Seguimiento de objetos

El problema de segmentación y el seguimiento se encuentran íntimamente ligados. Si se resuelve el problema de segmentación, la solución al seguimiento se obtiene de manera relativamente sencilla. En algunas aplicaciones de la visión artificial, como en navegación o vigilancia, se requiere de identificar y registrar la trayectoria de un objeto [27].

[Procesamiento de señales]

Por ello los algoritmos de seguimiento han sido ampliamente estudiados a lo largo de los años y pueden clasificarse en tres categorías [28]:

- Kernel: Se obtiene una representación del objeto localizado dentro de una caja o región elipsoidal.

- Silueta: Estos algoritmos requieren de un contorno inicial el cual se modifica para adaptarse nuevamente al objeto, esto se realiza ya sea por un modelo o por una función de energía.

- Puntuales: Emplean algoritmos de minimización de la energía y de segmentación para realizar el seguimiento.

En el estudio del seguimiento de objetos podemos encontrar dos ramas de investigación:

- El objeto es seguido por la cámara de manera que siempre se encuentra en el centro de la imagen.

- La cámara se encuentra fija y el objeto se sigue mientras se encuentre en el campo de visión.

Dentro de la amplia variedad de algoritmos de seguimiento uno de los más utilizados es el algoritmo Camshift, debido a lo robusto y a su ligereza computacional. Este algoritmo utiliza el tono de los pixeles para crear un mapa de probabilidad y conocer el tamaño, posición y orientación del objeto. Detallar el algoritmo Camshift no es objeto de esta tesis, puede encontrarse descrito con amplitud en [29] y [30]. De manera general, consiste en los siguientes pasos:

1. Seleccionar una región de interés la cual contiene el objeto a seguir.

2. Generar una distribución de probabilidad de pertenencia al objeto.

3. Calcular los momentos para obtener el centroide de la imagen.

   1. Obtener el momento cero con la ecuación,

$$Z_{00} = \sum_{x}\sum_{y} I(x,y) \qquad (4)$$

   en donde "I(x,y)" es el pixel del mapa de probabilidad.

   2. El segundo paso es obtener los primeros momentos para "x" e "y".

$$Z_{10} = \sum_{x}\sum_{y} xI(x,y) \qquad (5)$$

$$Z_{01} = \sum_{x}\sum_{y} yI(x,y) \qquad (6)$$

4. Obtener el centroide empleando

$$x_c = \frac{Z_{10}}{Z_{00}} \qquad (7)$$

$$y_c = \frac{Z_{01}}{Z_{00}} \qquad (8)$$

   en donde "x_c" y "y_c" serán las nuevas coordenadas de la región de interés.

5. Colocar la región de interés centrada en las coordenadas obtenidas.

6. Repetir el paso 3 y 4 hasta la convergencia (el cambio en las coordenadas del centroide sea inferior a una tolerancia).

Si la aplicación lo requiere se puede calcular el tamaño y la orientación necesaria para la región de interés usando los segundos momentos.

2.1.3.4 Parámetros ópticos de la cámara

Para poder obtener la información tridimensional de la escena, es necesario convertir los pixeles de la imagen digital en las coordenadas del plano de imagen. Para ello se requiere encontrar los parámetros intrínsecos (características ópticas y geométricas) y extrínsecos (orientación y posición), basados en puntos de los objetos observados cuyas coordenadas son conocidas, contrastándolas con las coordenadas medidas de la imagen [31].

Se espera que un algoritmo de calibración tenga las siguientes características:

- Autonomía: No debe haber interacción humana en el proceso de calibración.

- Precisión: Debe ser capaz de cumplir con los requerimientos de incertidumbre que tenga la aplicación.

- Eficiencia: Muchas de las aplicaciones requieren de una calibración rápida por lo que el algoritmo debe ser capaz de cumplir con los requerimientos computacionales y de tiempo.

- Versátil: Es deseable que el algoritmo funcione para una amplia gama de arreglos ópticos.

En [32] se presenta una clasifican los algoritmos en dos categorías:

- Fotogramétricos: Aquellos que emplean un objeto de dimensiones conocidas.

- Auto calibración: Emplean el movimiento de la cámara en una escena estática de la cuál se tiene conocimiento espacial y geométrico.

En ambas categorías se emplea el siguiente modelo de calibración, el cual se puede encontrar en [33] y [31]. Los parámetros extrínsecos de la cámara están dados por:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = R \begin{bmatrix} x_{1c} \\ x_{2c} \\ x_{3c} \end{bmatrix} + T \qquad (9)$$

En donde el vector $X_c = [\ x_{1c}\ \ x_{2c}\ \ x_{3c}\ ]^T$ representa las coordenadas del objeto en el sistema de coordenadas de la cámara. El vector $X = [\ x_1\ \ x_2\ \ x_3\ ]^T$ son las coordenadas del objeto con respecto al sistema de coordenadas fijo, "T" y "R" son la traslación y la rotación del sistema de coordenadas de la cámara.

Usando el modelo de perspectiva central podemos encontrar las coordenadas ideales en el plano de imagen usando la ecuación (2). Al tomar en cuenta la distorsión radial se obtiene

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} y_{1d} \\ y_{2d} \end{bmatrix} + \begin{bmatrix} D_1 \\ D_2 \end{bmatrix} \qquad (10)$$

En donde "y_1" y "y_2" son las coordenadas ideales en el plano de imagen, "y_{1d}" y "y_{2d}" son las coordenadas distorsionadas y "D_1" y "D_2" son la distorsión radial que se obtiene por:

$$\begin{bmatrix} D_1 \\ D_2 \end{bmatrix} = \begin{bmatrix} y_1 \ (k_1 r^2 + k_2 r^4 + \cdots) \\ y_2 \ (k_1 r^2 + k_2 r^4 + \cdots) \end{bmatrix} \qquad (11)$$
$$r = \sqrt{(y_{1d}^2 + y_{2d}^2)}$$

Finalmente las coordenadas en pixeles están dadas por:

$$\begin{bmatrix} y_{1p} \\ y_{2p} \end{bmatrix} = [S_1 \quad S_2] \begin{bmatrix} y_{1d} \\ y_{2d} \end{bmatrix} + \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \qquad (12)$$

En donde "S_1" y "S_2" son el tamaño de un pixel en ambos ejes y "C_1" y "C_2"son las coordenadas del centro óptico de la imagen en pixeles. La ecuación (12) contiene las características intrínsecas de la cámara, las cuales nos permiten transformar los pixeles en las coordenadas del plano de imagen.

2.2 Navegación inercial

La cinestesia es la percepción de movimiento, los seres vivos cuentan con diversos sentidos que les permite ubicarse y detectar su entorno. En robótica se busca obtener una representación sin ambigüedad y robusta del movimiento, para conocer la posición y orientación de un objeto. Esto se realiza mediante sensores inerciales (emplean la resistencia al cambio del momento), los cuales miden las fuerzas que

actúan sobre el sistema. La posición y orientación se obtiene mediante la integración de estas fuerzas [34].

La navegación inercial es utilizada en espacios en los cuales no se cuenta con infraestructura externa al sistema como sería el GPS (Global Positioning System) o la ubicación por triangulación. Para ello se emplea una unidad de medición inercial o IMU (Inertial Measurement Unit) por sus siglas en inglés. La cual consiste en tres acelerómetros y giróscopos colocados de forma ortogonal. Los sensores más utilizados son los de estado sólido, los cuales son pequeños, livianos, de bajo consumo, y de precio asequible [35].

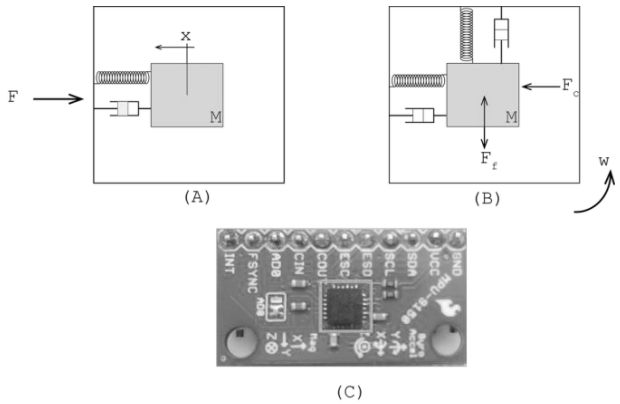

Figura 8: Sensores inerciales

En la imagen se esquematizan el funcionamiento del acelerómetro (A) y el giroscopio (B) de una IMU (C). Véase explicación en el texto.

En la Figura 8 se muestra una IMU (C) en la cual el circuito integrado dentro del cuadrado azul contiene los sensores inerciales. El acelerómetro "(A)" es un sensor que mide la fuerza "F" que actúa sobre él. Este sensor consiste en una masa de prueba "M", soportada por un elemento con elasticidad y amortiguamiento conocidos, el desplazamiento "X" de la masa debido a una fuerza externa aplicada al sistema permite conocer la aceleración resultante. Entre los principales tipos de acelerómetros podemos encontrar

- Capacitivos: La posición de la masa de prueba se obtiene por un cambio de capacitancia.

- Piezoeléctricos: La posición de la masa de prueba se obtiene mediante el cambio en la diferencia de potencial producido por la deformación del material que soporta la masa de prueba.

- Piezoresistivos: En este tipo de sensor la posición se obtiene mediante el cambio de resistencia provocada por la deformación del material que soporta la masa de prueba.

El giroscopio "(B)" es un sensor que mide la razón de cambio en la orientación del sistema. Este sensor consiste en una masa "M" a la que se le aplica una fuerza "F_F" en un eje, cuando el sistema gira la masa de prueba experimenta el efecto coriolis "F_c"; por lo que se produce una desviación con respecto al eje de vibración. Al medir esta desviación es posible obtener la velocidad "w" a la que el sistema se encuentra rotando. Existe otro tipo de giróscopo el cual emplea el efecto Sagnac, que es la difracción de la luz. En este tipo de giróscopos se remplaza el elemento de vibración por dos haces de luz que recorren la misma distancia. Un haz es enviado en el sentido del giro y el otro en contra sentido, el haz en el sentido rotación recorre una distancia mayor que el haz en contra sentido.

2.2.1 Obtención de la posición

El conocer la posición de un objeto mediante navegación inercial se realiza al integrar los valores del acelerómetro empleando:

$$x = \int \dot{x}\, dt = \int \int \ddot{x} - g\, dt\,, \qquad (13)$$

en donde "x" es la posición del objeto, " $\dot{x}$ " es la velocidad lineal del sistema, "$\ddot{x}$" es el valor de aceleración medido por el acelerómetro y "g" es la aceleración debida a la gravedad. Existen dos configuraciones para colocar los sensores inerciales [36, p. 32]:

- Cardan: En esta configuración se mantiene a los sensores independientes a la rotación del cuerpo mediante un sistema activo y los sensores se encuentran alineados con el marco de referencia. Esto permite que la integración de la aceleración se realice directamente sobre los datos proporcionados por el acelerómetro (el desplazamiento se encuentra en las coordenadas del marco de referencia).

- Fijo: Esta configuración es más simple al no contar con partes móviles, los sensores se encuentran sujetos al cuerpo y su sistema de coordenadas se ve afectado por la rotación del objeto. Para obtener la posición se requiere de trasformar las aceleraciones medidas en las coordenadas del cuerpo a las coordenadas del marco de referencia, empleando el giróscopo para obtener la rotación del sistema. [37]. Para obtener la posición del objeto se emplea

$$x_n = \int \int E_b^n\, \ddot{x}_b\, dt \qquad (14)$$

En donde la matriz de rotación "E_b^n", se obtiene empleando los valores del giróscopo, con esta matriz se transforman las coordenadas del cuerpo a las coordenadas del marco de referencia.

[Procesamiento de señales]

Los valores de los sensores inerciales presentan ruido e integrar este ruido conlleva un error acumulativo. Para minimizar el efecto de las discrepancias por ruido y fluctuaciones con la temperatura, los sensores se calibran y la señal se filtra. La combinación de los datos de distintos sensores como lo son GPS, visión, presión, entre otros, permite realizar una mejor estimación de la posición [38] [39] [40].

2.2.1.1 Métodos numéricos

Al derivar e integrar nos podemos encontrar con tres tipos de funciones [41]:

- Una función sencilla (polinomio, trigonométrica,…)
- Una función continua compleja difícil de integrar y diferenciar.
- Una función tabulada en donde los valores de f(x) y x están dadas de forma discreta (la forma más común de tener resultados experimentales).

Los valores discretos y la frecuencia de muestreo de los sensores inerciales hacen que la navegación corresponda al tercer caso. A diferencia del primero en el que es posible emplear las ecuaciones de diferenciación e integración, el segundo y tercero requieren de explotar las capacidades de los métodos numéricos para obtener la derivada y la integral de las funciones. En [42, pp. 585-671] se describen en detalle los distintos métodos.

El esquema más común para realizar una integración es representar una función compleja o los datos mediante una función aproximada.

$$\int_a^b f(x)dx \cong \int_a^b f_n(x)dx \qquad (15)$$

En donde "$f_n(x)$" es un polinomio de orden "n". A estas fórmulas se les conoce como las fórmulas de Newton-Cotes las cuales pueden ser clasificadas en dos tipos:

- Cerradas: Donde los límites de integración son conocidos.

- Abiertas: Donde los límites son desconocidos y se requiere de una extrapolación.

La primera fórmula de Newton-Cotes cerrada es conocida como la regla del trapecio, donde se emplea el primer término del polinomio resultando en:

$$I = h\left(\frac{f(x_1)+f(x_o)}{2}\right) - \frac{1}{12}f''(\xi)h^3 . \qquad (16)$$

En donde el segundo término corresponde al error; se puede apreciar que para funciones lineales la integral es exacta, pero si la función tiene derivadas de orden mayor tendremos un error. Entre más grande sea el intervalo "h", el error será mayor. Para disminuir el error puede dividirse "n" veces el intervalo "h" o emplearse un polinomio de orden mayor.

Así como para la integral se emplean los polinomios, para obtener la derivada se utiliza la serie de Taylor.

$$f(x_{i+1}) = \sum_{n=1}^{\infty} \frac{f^{(n)}(x_i)}{n!}(x - x_i)^n = f(x_i) + f'(x_i)\,(x - x_i) + \cdots \qquad (17)$$

Para obtener la primera derivada se considera la ecuación (17) hasta el segundo término para obtener

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} + O(h)\,. \qquad (18)$$

Obtener la derivada de orden "n" consiste en expresar la expansión de serie de Taylor para "$f(x_{i+n})$" en términos de "$f(x_i)$". Y eliminar las derivadas de orden inferior a "n".

$$f(x_{i+n}) = f(x_i) + f^{(1)}(x_i)h + \cdots \frac{f^{(n)}(x_i)(n\,h)^n}{n!} + \cdots \qquad (19)$$

Así, para la segunda derivada hacia adelante obtendremos la expresión

$$f''^{(x_i)} = \frac{f(x_{i+2}) - 2f(x_{i+1}) + f(x_i)}{h^2} \qquad (20)$$

En caso de requerirse una mejor estimación se pueden considerar más términos de la expansión de Taylor. Por ejemplo, para mejorar la estimación de la primer derivada (18) se sustituye la segunda derivada en el tercer término de la expansión para obtener

$$f'(x_i) = \frac{-f(x_{i+2}) + 4\,f(x_{i+1}) - 3\,f(x_i)}{2h} + O(h^2). \qquad (21)$$

3 Observador de profundidad

Para realizar una estimación de profundidad pasiva mediante visión, se requiere un modelo que permita relacionar las coordenadas de un objeto en el espacio tridimensional, con su proyección bidimensional. A su vez medir la aceleración de la cámara, y los desplazamientos de la proyección. Con estos datos se calcula una estimación de las coordenadas del objeto en el espacio tridimensional [43] [44] [45].

3.1 Modelo de visión perspectiva

En la visión perspectiva se emplea el movimiento entre la cámara y el objeto para determinar la distancia relativa entre ambos. Las propiedades de este sistema son:

- La velocidad angular y la velocidad lineal son derivables.
- Los límites superiores e inferiores de la dinámica son conocidos.
- La distancia en el eje óptico "x_3" es mayor que la distancia focal.
- La distancia focal "ε" es mayor que cero.
- La dinámica del plano de imagen "Y(t)" se encuentra entre dos límites, superior e inferior, los cuales son conocidos.

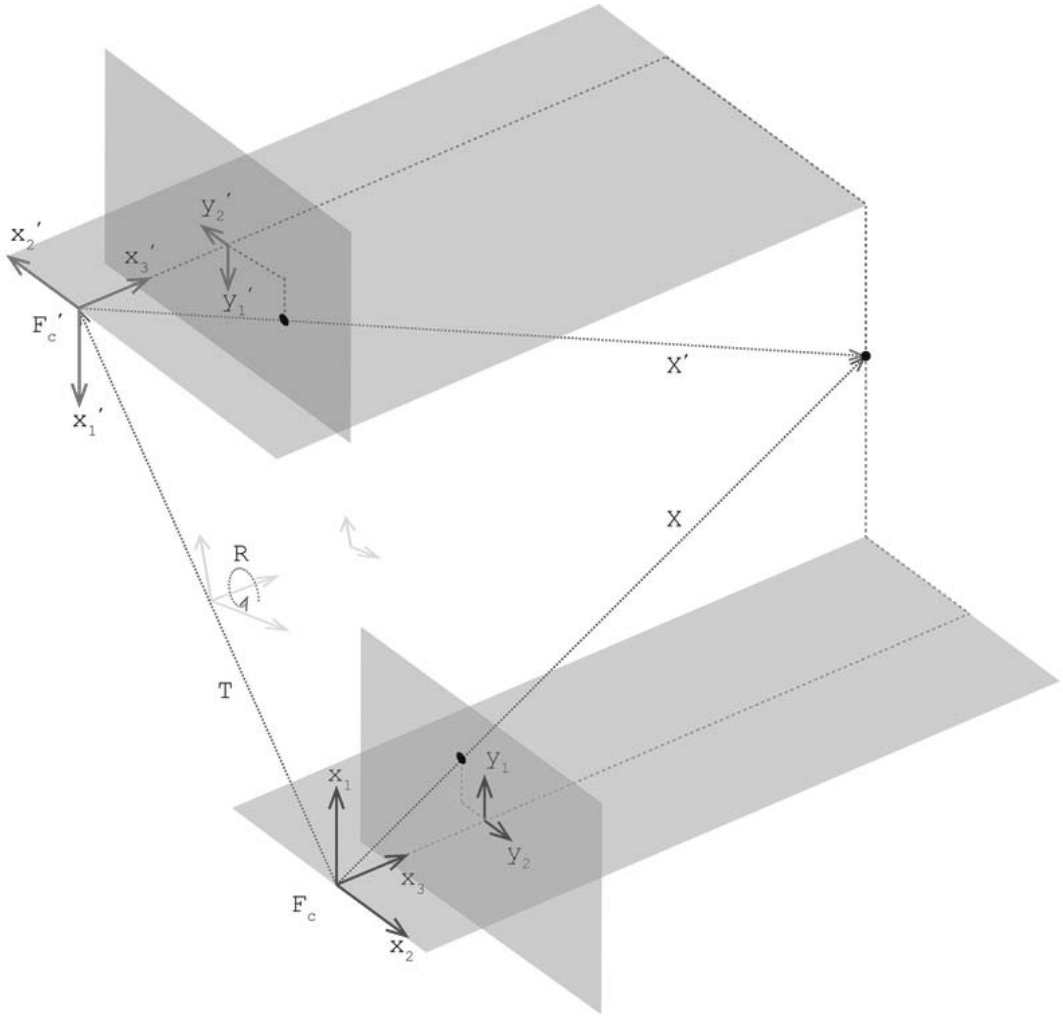

Figura 9: Modelo de visión perspectiva

Se muestra el sistema de coordenadas de la cámara y del plano de imagen en el tiempo inicial y en un segundo tiempo después de sufrir una traslación y rotación.

El modelo de visión perspectiva se muestra en la Figura 9, en la cual se observa el sistema de coordenadas de la cámara y el plano de imagen, en el tiempo inicial y después de una traslación y rotación. Las ecuación que expresa la transformación del sistema de coordenadas es:

$$\begin{bmatrix} x_1{}' \\ x_2{}' \\ x_3{}' \end{bmatrix} = R \begin{bmatrix} x_1 \\ x_2 \\ x_2 \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \qquad (22)$$

[Observador de profundidad]

En donde el vector $X = [\ x_1 \quad x_2 \quad x_3\ ]^T$ es la posición del objeto en el sistema de coordenadas inicial de la cámara y su magnitud es la distancia relativa entre la cámara y el objeto.

El vector $X' = [\ x_1' \quad x_2' \quad x_3'\ ]^T$ es la posición del objeto en el sistema de coordenadas final, el vector $T = [\ t_1 \quad t_2 \quad t_3\ ]^T$ es la traslación de la cámara en el sistema de coordenadas inicial.

Para conocer la dinámica del sistema de visión perspectiva se deriva la ecuación (22) para obtener el movimiento relativo del objeto y la cámara [19] y obtener

$$\dot{X}' = \begin{bmatrix} 0 & w_3 & -w_2 \\ -w_3 & 0 & w_1 \\ w_2 & -w_1 & 0 \end{bmatrix} X + \dot{T}\,. \qquad (23)$$

En donde la matriz anti simétrica que transforma al vector "X´" está formada por vector $W = [\ w_1 \ w_2 \ w_3]^T$ que es la velocidad angular de la cámara. Finalmente, se obtiene el movimiento del objeto en las coordenadas del plano de imagen en función del movimiento relativo entre objeto y cámara. Para ello se considera el vector auxiliar

$$y = [y_1 \quad y_2 \quad z]^T = \left[\epsilon \frac{x_1}{x_3} \quad \epsilon \frac{x_2}{x_3} \quad \frac{1}{x_3}\right] \qquad (24)$$

el cual se deriva, y se combina con el sistema (23) para obtener la dinámica del sistema de visión perspectiva en donde las variables y_1 y y_2 son las coordenadas del objeto en el plano de imagen

[Observador de profundidad]

$$\begin{bmatrix}\dot{y}_1\\ \dot{y}_2\\ \dot{z}\end{bmatrix}=\begin{bmatrix} z\epsilon & 0 & -y_1 z & -\frac{y_1 y_2}{\epsilon} & \epsilon+\frac{y_1^2}{\epsilon} & -y_2\\ 0 & z\epsilon & -y_2 z & \epsilon+\frac{y_2^2}{\epsilon} & \frac{y_1 y_2}{\epsilon} & y_1\\ 0 & 0 & -z^2 & -\frac{y_2 z}{\epsilon} & \frac{y_1 z}{\epsilon} & 0\end{bmatrix}\begin{bmatrix}\dot{T}\\ W\end{bmatrix}. \qquad (25)$$

Se requiere entonces de transformar las coordenadas en pixeles medidas en la imagen digital a las coordenadas del plano de imagen para lo que se emplea

$$\begin{bmatrix}y_{1p}\\ y_{2p}\\ 1\end{bmatrix}=\begin{bmatrix} S_1\,\epsilon & 0 & C_1\\ 0 & S_2\epsilon & C_2\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\frac{x_1}{x_3}\\ \frac{x_1}{x_3}\\ 1\end{bmatrix}+\begin{bmatrix}D_1\,s_1\\ D_2\,s_2\\ 0\end{bmatrix} \qquad (26)$$

En donde el vector $Y_p = [\ Y_{1p} \quad Y_{2p} \quad 1\ ]^T$ describe la posición en pixeles del objeto en la imagen. Los factores "S", "C" y "D" son la relación pixel/m, las coordenadas del centro de la imagen y la deformación radial respectivamente.

3.2 Observador del sistema de perspectiva

Un observador o estimador se define como un modelo matemático encargado de reconstruir el vector de estado de un sistema empleando la señal de salida [46]. Su utilidad radica en que, cuando no es posible conocer la dinámica de un sistema mediante sensores, el modelo matemático permite realizar una estimación de la evolución del sistema, siempre que el sistema sea observable; es decir, que las variables de estado influyen en la salida [47].

El modelo de visión perspectiva (24) requiere de conocer la coordenada "x_3" para calcular las coordenadas del vector

posición con base en las coordenadas del plano de imagen. La dinámica de la cámara establecida por la ecuación (25) permite plantear un observador con el cual reconstruir las coordenadas del objeto.

El sistema de visión perspectiva es observable si la velocidad lineal es diferente de cero [48]

$$(\dot{t}_1\,\epsilon - \dot{t}_3\,y_1)^2 + (\dot{t}_2\,\epsilon - \dot{t}_3\,y_2)^2 \geq \omega > 0, \qquad (27)$$

es decir, si existe algún cambio en la posición del sistema.

En esta tesis se plantea implementar el observador de estado descrito en [49] el cual se basa en la teoría de contracción [50]. Que tiene la característica principal de presentar convergencia exponencial. A continuación se describe el estimador en dos de sus tres versiones.

3.2.1 Estimador de orden reducido

Un estimador de estado reconstruye por completo la evolución de la dinámica del sistema, incluyendo las variables de salida, las cuales son conocidas. Un observador de orden reducido es un modelo que elimina la redundancia de calcular las señales de salida [51].

El observador de orden reducido propuesto por Grave I. Y Tang Y. [49] se obtiene al reescribir la dinámica del sistema de visión perspectiva (25) como:

[Observador de profundidad]

$$\dot{\hat{v}} = f_3\, v_z - \dot{t}_3\, v_z^2 - \alpha\, v_z \; - K\,(\,(\,\dot{t}_1 - \dot{t}_3\, y_1\,)\, f_1 + \,(\,\dot{t}_2 - \dot{t}_3\, y_2\,)\, f_2\,))\text{-}\gamma$$
$$\dot{\hat{\zeta}} = \; \alpha\, v_z + K\,(\,(\,\dot{t}_1 - \dot{t}_3\, y_1\,)\, f_1 + \,(\,\dot{t}_2 - \dot{t}_3\, y_2\,)\, f_2\,)) + \gamma - K_z\,(\hat{\zeta} - \zeta) \qquad (\ 28\ )$$
$$\hat{z} = \hat{v} + \zeta(y, \dot{T})$$

en donde $v_z = (\hat{v} + \zeta)$, las ecuaciones

$$f_1 = \; -y_1 y_2 w_1 + (1 + y_1^2)\, w_2 - y_2 w_3 \;\; , \qquad (\ 29\ )$$
$$f_2 = (1 + y_2^2) w_1 + y_1 y_2 w_2 + y_1 w_3 \;\; , \qquad (\ 30\ )$$
$$f_3 = \; -y_2 w_1 + y_1 w_2 \;\; , \qquad (\ 31\ )$$

Contienen la dinámica rotacional del sistema, a su vez

$$\zeta = K\left(\left(\dot{t}_1 - \frac{\dot{T}_3}{2} y_1\right) y_1 + \left(\dot{t}_2 - \frac{\dot{t}_3}{2} y_2\right)\, y_2\,\right) \qquad (\ 32\ )$$
$$\gamma = \; K\left(\left(\ddot{t}_1 - \frac{\ddot{T}_3}{2} y_1\right) y_1 + \left(\ddot{t}_2 - \frac{\ddot{t}_3}{2} y_2\right)\, y_2\,\right) \qquad (\ 33\ )$$
$$\alpha = K\,(\,(\,\dot{t}_1 - \,\dot{t}_3 Y_1)^2 + (\,\dot{t}_2 - \,\dot{t}_3 y_2)^2\,) \qquad (\ 34\ )$$

Contienen información de traslación del sistema y “K” es la ganancia de convergencia, la cual debe cumplir

$$K > \frac{c(\delta_z)}{\omega} \;\; , \qquad (\ 35\ )$$

en donde w se obtiene a partir de(27) y $c(\delta_z)$ esta dado por:

$$c(\delta_z) = \frac{2\,\bar{\dot{T}}_3}{\epsilon} + 2\,\bar{\dot{t}}_3\,\delta_z - \bar{w}_2\,\bar{y}_1 + \,\bar{w}_1\,\bar{y}_2 \;\; , \qquad (\ 36\ )$$

y contiene los límites superiores de las coordenadas de proyección y la dinámica de rotación.

[Observador de profundidad]

Finalmente, para obtener la estimación de los valores del vector de estado se integra la ecuación (28) y se sustituye la evaluación de "z" en

$$\hat{x}_1 = \mathrm{y}_1\hat{x}_3 \quad \hat{x}_2 = \mathrm{y}_2\hat{x}_3 \quad \hat{x}_3 = \frac{1}{\hat{z}} \ , \qquad (37)$$

3.2.2 Estimador orden completo

El observador de orden completo es:

$$\begin{gathered}
\dot{\hat{v}} = f_3\, v_z - \dot{t}_3\, v_z^2 - \alpha\, v_z \ - K\,(\,(\,\dot{t}_1 - \dot{t}_3\, y_1\,)\, f_1 + \,(\,\dot{t}_2 - \dot{t}_3\, y_2\,)\, f_2\,) - \gamma \\
\dot{\hat{y}}_1 = f_1 + (\,\dot{t}_1 - \,\dot{t}_3\, y_1\,)\, v_z - \alpha\,(\,\hat{y}_1 - y_1) \\
\dot{\hat{y}}_2 = f_2 + (\,\dot{t}_2 - \,\dot{t}_3\, y_2\,)\, v_z - \alpha\,(\,\hat{y}_2 - y_2) \\
\dot{\hat{\zeta}} = \,\alpha\, v_z + K\,(\,(\,\dot{t}_1 - \dot{t}_3\, y_1\,)\, f_1 + \,(\,\dot{t}_2 - \dot{t}_3\, y_2\,)\, f_2\,)) + \gamma - K_z\,(\hat{\zeta} - \zeta) \\
\hat{z} = \hat{v} + \zeta(\hat{y}, \dot{T})
\end{gathered} \qquad (38)$$

El cual extiende la ecuación (28) al incluir la dinámica de la proyección en el plano de imagen. Al estimar la proyección y emplearla para obtener el vector de estado, el estimador de orden completo permite resolver dinámicas del sistema que contengan señales de dinámica de proyección corruptas por ruido.

En el observador de orden completo la ganancia de convergencia está dada por

$$K > \frac{c(\delta_z) + \sqrt{(\,c^2(\delta_z) + \,\omega}}{2\omega} \qquad (39)$$

y de manera análoga a (37) se emplea

$$\hat{x}_1 = \hat{y}_1\hat{x}_3 \quad \hat{x}_2 = \hat{y}_2\hat{x}_3 \quad \hat{x}_3 = \frac{1}{\hat{z}} \ . \qquad (40)$$

4 Diseño experimental

Para calcular la distancia entre un objeto y la cámara mediante visión perspectiva se plantea el siguiente diseño experimental.

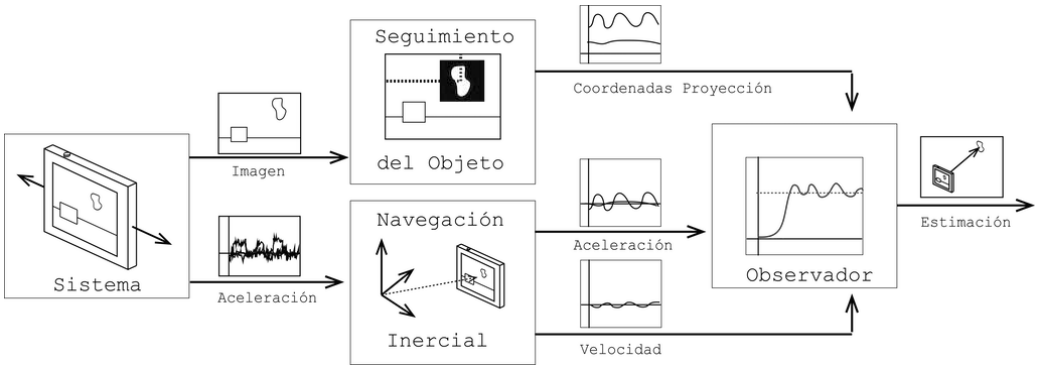

Figura 10 : Diagrama a bloques de la plataforma experimental

La estimación de profundidad tiene como entrada las señales medibles del sistema y pasa por tres etapas: seguimiento, navegación y observador de estado, para dar como resultado una estimación de la distancia relativa entre el objeto y la cámara.

En él se adquieren la imagen de la escena captada por la cámara y la aceleración del sistema medida mediante un sistema de medición inercial. Después se emplean dos etapas paralelas, en la primera se realiza una selección y seguimiento del objeto en la imagen, a lo largo de la trayectoria del sistema. Mientras en la segunda etapa se realiza la integración de la aceleración para obtener la velocidad del sistema. Finalmente, al contar con las coordenadas de la proyección del objeto en el plano de imagen, la aceleración y la velocidad del sistema, se emplea el estimador (28 y 38) para obtener la distancia entre la cámara y el objeto.

Para llevar a cabo el diseño experimental se requiere tener un sistema que cuente con

- Un sensor de visión que permita adquirir la imagen de la escena que contiene el objeto.

- Sensores inerciales que permitan adquirir los valores de aceleración del sistema.

- Capacidad de procesamiento para llevar a cabo el seguimiento del objeto en la imagen y la navegación inercial del sistema, así como para estimar las coordenadas del objeto empleando el observador (28 y 38).

En la actualidad hay sistemas de uso común en la vida diaria que cuentan con esas características. La mayoría de los dispositivos móviles como lo son celulares, reproductores de música y tabletas, cuentan con una gran variedad de sensores, entre los que se encuentran: inerciales, de orientación, de visión y posicionamiento global. Debido a su tamaño y capacidad de procesamiento han sido utilizados como dispositivos de adquisición de datos en diversas aplicaciones [52]. Al contar con las tres características requeridas por nuestro diseño experimental, los dispositivos móviles actuales se presentan como una herramienta poco explorada para la estimación de profundidad.

En este trabajo se propone utilizar los sensores de un dispositivo móvil para realizar la adquisición de los datos y su capacidad de procesamiento para llevar a cabo el procesamiento de las señales y la estimación de la distancia entre el dispositivo y el objeto observado. El dispositivo móvil empleado en este trabajo es el celular Xperia™ Z1 D5503, el cual cuenta con un procesador Qualcomm Snapdragon™ 800 de cuatro núcleos, que trabaja a 2.2 Ghz, tiene 2 GB de memoria RAM, un sensor de cámara Exmor RS™ de 20 MP, acelerómetro, giroscopio, magnetómetro y una pantalla táctil

capacitiva de 4.3 pulgadas en diagonal [53]. El sistema operativo del dispositivo es Android 4.4 ("Kit Kat") el cual, al ser un OS (Operative System) ampliamente extendido facilita la transición de la aplicación desarrollada entre terminales. La programación de la aplicación se realiza empleando el paquete de desarrollo de software o SDK (Software Development Kit) y el entorno de desarrollo integrado o IDE (Integrated Developmetn Enviroment) "Android studio", los cuales se encuentran disponibles en la página oficial de desarrollo de Android. Las pruebas, simulaciones y validaciones se realizan mediante el programa comercial MATLAB.

A continuación se describen las distintas etapas propuestas para la estimación de profundidad.

4.1 Seguimiento del objeto

El seguimiento del objeto se realiza mediante una modificación del algoritmo Camshift. Se programó una rutina en la aplicación, encargada de adquirir y procesar la imagen. Esta rutina corre en paralelo con la rutina de navegación inercial. En la Figura 11 se esquematiza el algoritmo de seguimiento utilizado.

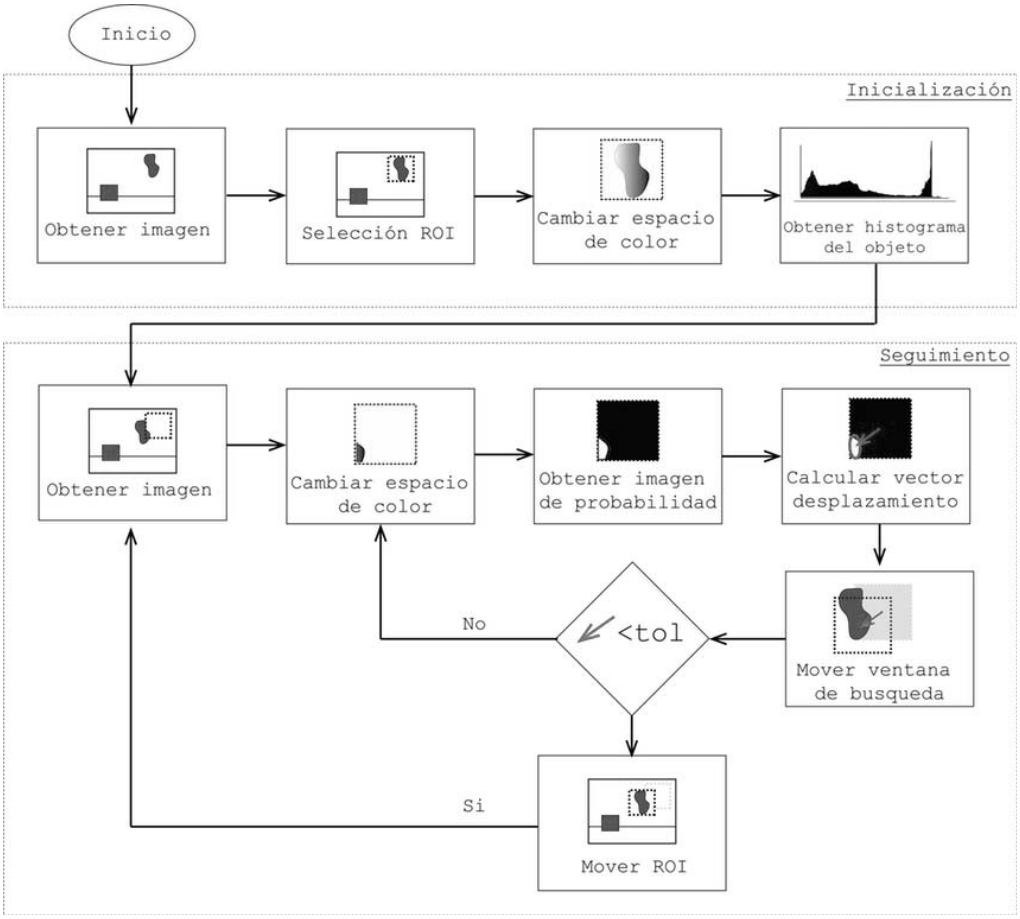

Figura 11: Algoritmo de seguimiento del objeto

El algoritmo consta de dos fases:
- La inicialización, en donde se selecciona la región de interés (ROI) y se obtiene el histograma del objeto.
- El seguimiento, en la cual se realiza el algoritmo Camshift hasta que el desplazamiento (flecha roja) sea menor a una tolerancia (tol).

- Inicialización

1. Obtener imagen: Se inicializa la cámara y empleando la rutina "vista previa" proporcionada por el SO, se controla la adquisición de la imagen. Empleando esta rutina se captura continuamente la escena, agilizando así la captura.

2. Selección ROI: La región de interés o ROI (Region Of Interest) es aquella porción de la proyección de la escena que queremos analizar. Esta región es

seleccionada manualmente por el usuario empleando la pantalla táctil del dispositivo. Limitar la región de interés permite aumentar la velocidad de procesamiento, pues sólo se analiza una porción de la imagen.

3. Cambiar espacio de color: En los terminales con OS Android es usual que el espacio de color de la imagen que regresa el sistema operativo sea YUV, por ello se realiza una conversión al espacio en el que se realiza el procesamiento. En este trabajo se emplea un espacio de color binario, el cual se obtiene al convertir del espacio de color YUV al espacio RGB que a su vez es transformado a escala de grises, empleando el promedio de los tres canales y finalmente, empleando un umbral, se binariza el valor de cada pixel.

4. Obtener el histograma del objeto: Para realizar el seguimiento del objeto se hace una comparación entre los valores de cada pixel de la ventana de búsqueda con el histograma normalizado del objeto. Para obtener el histograma se emplea:

$$H = \{h_j\}_{j=1,2\ldots m} = \frac{1}{n}\sum_{i=1}^{n} C_j(P_i), \qquad (41)$$

en donde el histograma "H" cuenta con "m" intervalos para los valores de intensidad de los "n" pixeles del objeto. "P_i" es el valor de intensidad que tiene el pixel "i", "$C_j(P_j)$" es una función con valor unitario si "P_i" corresponde al intervalo "j" del histograma y nulo si no pertenece.

- Seguimiento

5. Obtener imagen: Se adquiere una nueva imagen de la escena, en la cual la posición del objeto ha cambiado.

6. Cambiar espacio de color: Se inicializa una ventana de búsqueda en la misma posición que la ROI, y se realiza una conversión del espacio de color YUV al espacio de color binario (blanco o negro).

7. Obtener imagen de probabilidad: El siguiente paso es obtener una imagen que represente la comparación entre el valor del pixel y el histograma del objeto. Para ello se emplea

$$I_i = h(P_i), \qquad (42)$$

en donde "I_i" es el valor del pixel "i" de la imagen de probabilidad y "$h(P_i)$" es una función que asigna la probabilidad del pixel "i" de pertenecer al objeto en función del histograma del objeto. De esta manera la imagen de probabilidad tendrá una mayor intensidad en aquella región en la cual se encuentre el objeto.

8. Calcular vector desplazamiento: Al contarse con la imagen de probabilidad se calcula el desplazamiento del objeto entre la imagen anterior y la actual. Para ello se obtienen los momentos y se calcula el nuevo centro de la ROI mediante (7) y (8). Y se calculan las nuevas coordenadas de la ventana empleando:

$$X_n = X_r + R_i - R_c. \qquad (43)$$

En la Figura 12 se muestran los vectores de posición de las regiones, el vector "Xr" es la posición de la ventana de búsqueda, "Rc" es el centro de la ROI en las coordenadas de la ventana, "Ri" es el vector al nuevo centro de la ventana de búsqueda, "Xd" es el desplazamiento del objeto y "Xn" es el vector a la nueva posición de la ventana de búsqueda.

9. Mover ROI: Cuando el desplazamiento de la ventana de búsqueda sea inferior a una tolerancia ("tol") es decir

$$X_d = R_i - R_c \leq tol \qquad (44)$$

el objeto se encontrará en el centro de la ventana. Se asignan las nuevas coordenadas a la región de interés y se repiten los pasos del cinco en adelante. En caso contrario si "X_d" es mayor a la tolerancia, significa que el objeto aún no se encuentra centrado y las nuevas coordenadas se asignan a la ventana de búsqueda repitiéndose los pasos del siete en adelante.

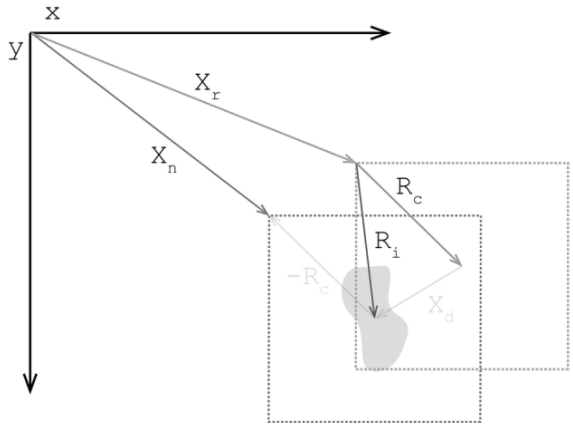

Figura 12: Vector desplazamiento

En figura se muestran los vectores de posición y traslación de la ventana de búsqueda. El objetivo es calcular el vector "X_n" en función de los vectores conocidos "X_r", "R_c" y "R_i".

4.2 Obtención de aceleración y velocidad

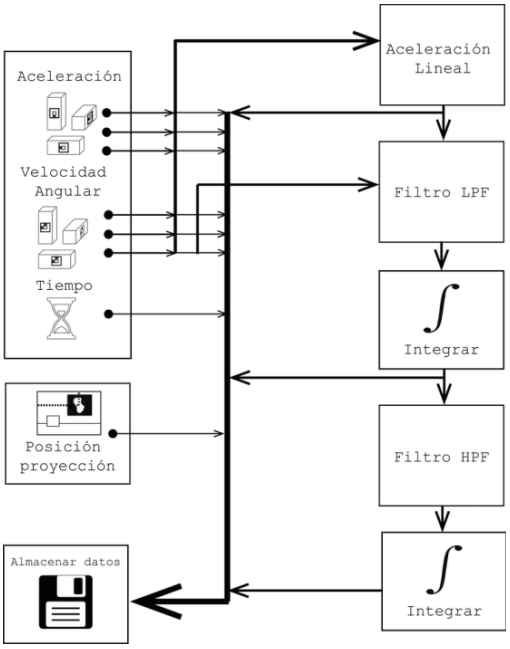

Figura 13: Algoritmo de navegación inercial

Se muestra el esquema del procedimiento realizado para obtener los valores de aceleración lineal y velocidad lineal y angular. Véase explicación en el texto.

En paralelo al seguimiento del objeto se realiza la navegación inercial, rutina encargada de adquirir la aceleración lineal, velocidad lineal y velocidad angular de la cámara empleando una rutina del SO dedicada a monitorear los sensores inerciales del dispositivo móvil. La rutina proporcionada por el SO da la aceleración lineal y aceleración que incluye el campo gravitacional. Para obtener la velocidad y posición se realiza una integración de newton-cotes de primer orden empleando la ecuación (16).

Si existe la presencia de ruido en la señal de aceleración la integración de estos valores resultará en una desviación de la velocidad y posición reales. Para compensar esta desviación se hace un filtrado de frecuencia en la señal mediante un filtro pasa altas o HPF (High Pass Filter) con

el fin de eliminar la deriva por integración y un filtro pasa bajas o LPF (Low Pass Filter) para eliminar el ruido.

En la Figura 8 se muestra el esquema del procedimiento para realizar la navegación inercial

1. Adquisición de los datos: Mediante las rutinas proporcionadas por el sistema operativo del terminal se obtienen los datos de los sensores inerciales (aceleración y la velocidad angular). También se almacena la hora de la adquisición para obtener el intervalo de tiempo entre mediciones, requerido al realizar la integración.

2. Obtener aceleración lineal: La aceleración lineal es aquella que se encuentra libre del efecto del campo gravitacional de la Tierra. Los acelerómetros se ven afectados por el campo gravitacional, por lo que para obtener los valores lineales se emplea la orientación del dispositivo móvil obtenida mediante el giróscopo. Al conocer la orientación del dispositivo es posible transformar el vector de aceleración, del sistema de coordenadas del dispositivo, al sistema de coordenadas de referencia; y así obtener la aceleración lineal al restar la constante de gravedad de los valores del acelerómetro. Se utiliza una rutina proporcionada por el OS Android dedicada a calcular la aceleración lineal.

3. Almacenar los datos: En la memoria del dispositivo se almacenan la velocidad angular, la aceleración lineal, el tiempo y la posición de la proyección (obtenida mediante la rutina de seguimiento).

4. Filtrado de ruido: En el caso de que las señales de aceleración y velocidad angular cuenten con ruido se

emplea un filtro digital LPF de respuesta finita al impulso o FIR (Finite Impulse Response) de la forma

$$y_i = \sum_{k=0}^{N-1} b_k \, x_{i-k}, \qquad (45)$$

En donde "y_i" es la señal filtrada, "N" es el orden del filtro del cual su coeficiente k-esimo es "b_k". Se eligió este tipo de filtro debido a su facilidad de programación y bajo consumo computacional. En el diseño del filtro se emplean las herramientas de la paquetería MATLAB con las cuales se obtienen los coeficientes del filtro los cuales son implementados en el terminal.

5. Obtener la velocidad y orientación: Al tener señales libres de ruido se realiza una integración de la aceleración para cada componente mediante la ecuación

$$\dot{T}_{i+1} = \frac{(\tau_{i+1} - \tau_i)}{2} \, (\ddot{t}_{i+1} + \ddot{t}_i), \qquad (46)$$

y la orientación se obtiene mediante

$$\theta_{i+1} = \frac{(\tau_{i+1} - \tau_i)}{2} \, (w_{i+1} + w_i) \ , \qquad (47)$$

En ambas ecuaciones τ_i es el tiempo en la medición "i", mientras τ_{i+1} es el tiempo de la siguiente medición.

6. Corrección de la deriva: Para evitar cualquier desviación del valor real en la velocidad debido a la integración de ruido en la señal de aceleración que no se lograra eliminar en su totalidad, se emplea un filtro digital HPF tipo FIR,

el cual elimina la constante de deriva generada por la integración del ruido.

7. Obtener la posición: Se realiza una integración de la velocidad para obtener la posición de la cámara a partir de los valores de velocidad calculados en los pasos anteriores empleándo

$$T_{i+1} = \frac{(\tau_{i+1} - \tau_i)}{2} (\dot{t}_{i+1} + \dot{t}_i). \qquad (48)$$

8. Almacenar los datos: Se almacenan los datos de velocidad y posición en la memoria del dispositivo para ser empleados en observador de estado al realizar la estimación de las coordenadas del objeto. Se regresa al primer paso hasta que el usuario detenga la adquisición de los datos y o se realice la estimación.

4.3 Determinación de la profundidad

Al contarse con los datos de aceleración, velocidades, proyección y tiempo, se procede a realizar la estimación de las coordenadas del objeto en el marco de referencia de la cámara empleando el observador, proceso esquematizado en la Figura 14.

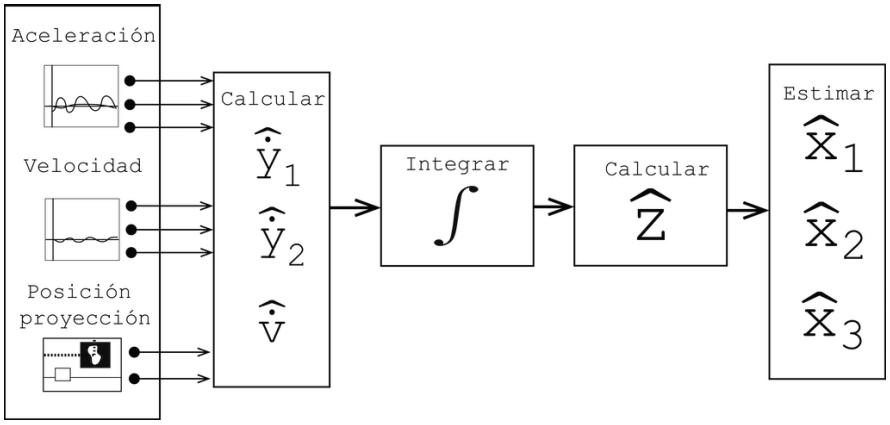

Figura 14: Algoritmo del observador de estado

Se esquematiza el procedimiento de la rutina de estimación. Véase explicación en el texto.

1. Se obtienen los datos almacenados en la memoria del dispositivo móvil, adquiridos en fase de navegación inercial.

2. Se calcula "$\dot{\hat{y}}_1$", " $\dot{\hat{y}}_2$" y "$\dot{\hat{v}}$" mediante (38), empleando los valores de aceleración, velocidad, rotación y proyección.

3. Se integran numéricamente con la ecuación (17) los valores calculados en el paso anterior para obtener "$\hat{y}_1$", " $\hat{y}_2$" y "$\hat{v}$", conociendo el valor de "$\hat{v}$" se puede calcular la estimación de $\frac{1}{x_{3c}}$.

4. Empleando "$\hat{z}$" se obtiene "$\hat{x}_{1c}$",$\hat{x}_{2c}$" y "$\hat{x}_{3c}$" mediante la ecuación (43).

5. Finalmente se almacena la estimación de las coordenadas del objeto y se procede a desplegar la estimación de la distancia.

4.4 Plataforma experimental

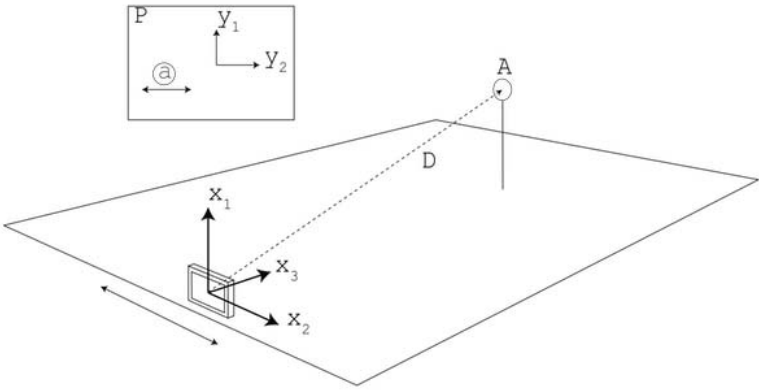

Figura 15: Plataforma experimental

El dispositivo móvil se encuentra a una distancia conocida "D" del objeto "A", se realiza un movimiento oscilatorio sobre el eje "x_2", produciendo un movimiento proporcional de la proyección "a" sobre el eje "y_2". Esta dinámica es adquirida mediante los sensores inerciales y la cámara del dispositivo.

Finalmente al integrar en el dispositivo móvil las tres rutinas anteriormente descritas, se cuenta con un sistema compacto capaz de determinar la distancia entre el móvil y un objeto. Para la validación del sistema se realizan simulaciones en MATLAB, con las condiciones impuestas en el experimento esquematizado en la Figura 15. El dispositivo realiza una traslación oscilatoria unidimensional sobre el eje "x_2" sin rotación sobre los otros ejes. El objeto "A", se encuentra a una distancia conocida "D" del dispositivo móvil. Posteriormente se realiza el experimento físicamente, en donde el movimiento del dispositivo es adquirido por los sensores mediante las rutinas de navegación y seguimiento, obteniéndose aceleración, velocidad y proyección. Finalmente se calcula la distancia "$\hat{D}$" empleando la rutina de estimación.

En el software elaborado en este trabajo, las rutinas se programaron y compilaron en una aplicación para terminales Android. La aplicación contiene rutinas encargadas de:

- El seguimiento del objeto
- La navegación inercial de la cámara
- La estimación de las coordenadas del objeto
- El almacenamiento de los datos adquiridos
- El despliegue de la información.

El manejo de las distintas rutinas se realiza mediante una interfaz gráfica de usuario o GUI (Graphical User Interface) en la pantalla táctil del dispositivo, la cual se muestra en la Figura 16.

Las siete regiones de la interfaz cumplen las siguientes funciones:

1. Estado: En esta región se muestra la acción que realiza la aplicación.

2. Estimación: Botón que inicia la rutina encargada de realizar la estimación de las coordenadas del objeto.

3. Guardar: Botón que llama una rutina que almacena los datos en la memoria interna del dispositivo. Durante la adquisición, procesamiento y despliegue de los datos la información se encuentra en la memoria temporal o RAM (Random Access Memory). Si el usuario lo desea puede almacenar los datos actuales en la memoria interna del dispositivo para poder acceder a estos en otro momento.

4. Graficar: Botón que llama a una rutina que muestra los resultados de la adquisición y estimación, mediante una gráfica.

5. Iniciar/Detener: Botón encargado de iniciar y detener las rutinas de seguimiento y navegación inercial.

6. Previsualización: Se muestra en todo momento lo que capta la cámara del dispositivo.

7. ROI: Región de interés, la cual se puede trasladar y escalar por el usuario.

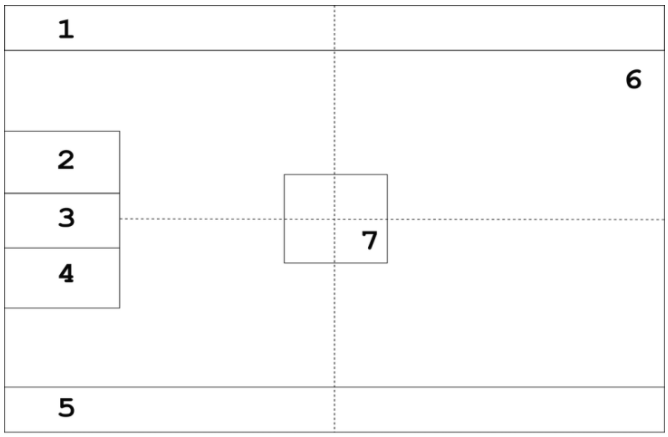

Figura 16: Esquema interfaz gráfica

La pantalla táctil es dividida en siete regiones, 1 es la barra de estado, 2 botón de rutina de estimación de profundidad, 3 botón para guardar en la tarjeta de memoria, 4 aquí se grafica en la pantalla el resultado de la estimación, 5 botón de inicio y pausa, 6 visualización de la escena, 7 región de interés.

5 Resultados

En este capítulo se presentan los resultados obtenidos en simulaciones de los observadores, los programas desarrollados para la adquisición y procesamiento de las señales, y las pruebas realizadas con el dispositivo móvil.

5.1 Simulaciones

Para corroborar que la implementación de los observadores [19] y [49] en su versión de orden reducido (28) y de orden completo (38) sea la correcta se realizaron simulaciones en MATLAB. Para ello se emplean las condiciones iniciales:

- $X(0) = [\ 10 \quad 5 \quad 2\ ]^T$
- $\dot{T} = \left[\ 0.3 \quad 0.4+0.1*\mathrm{Sin}\left(\frac{\pi\ t}{4}\right) \quad -0.3\ \right]^T$
- $W = \left[\ 0 \quad -\frac{\pi}{30} \quad 0\ \right]^T$
- $K = 10$

Para los observadores (28) y (38) se emplea K_z=20 para la estimación de ζ y $\hat{\zeta}(0)=\zeta\left(y(0),\dot{T}(0)\right)$. Las condiciones iniciales para la estimación del observador de orden completo se emplea $\hat{z}(0)=1$ y $\hat{y}(0)=y(0)$ y del observador de orden reducido se utiliza $z(0)=1.5$. Mientras que para el observador propuesto en [19] se emplea un α_o=5. La simulación se realiza con un tamaño de paso de $1*10^{-3}$ y en un intervalo de tiempo de 0 a 2 segundos.

[Resultados]

En la Figura 17 se observan los resultados obtenidos los cuales concuerdan con lo reportado en [19] y [49]. Las graficas “A” y “B” de la Figura 17 muestran las coordenadas de posición del objeto y su proyección. Los observadores estiman las tres coordenadas del objeto, las cuales se muestran en “C1”, “C2” y “C3”. En “D” se muestra la estimación de la profundidad, es decir la magnitud del vector posición del objeto y en “E” se muestra el error de la estimación, el recuadro dentro de la gráfica es un acercamiento del área punteada, en donde se aprecia que el observador de orden reducido “OR” y de orden completo “OC” convergen exponencialmente mientras que el observador propuesto en [19] “DN” presenta una convergencia asintótica.

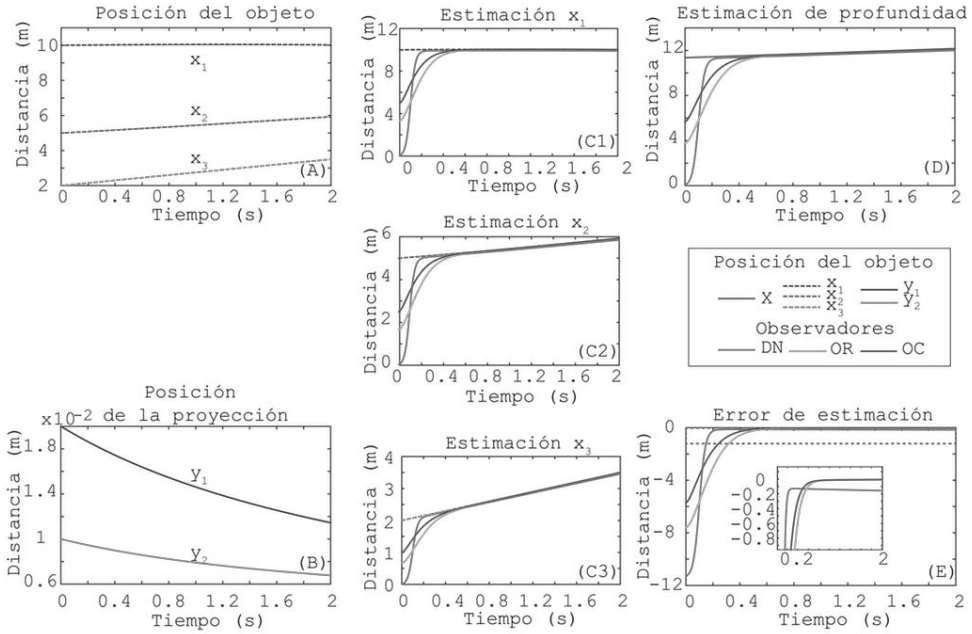

Figura 17: Validación del observador

Las gráficas (A) y (B) muestran la posición del objeto en el tiempo y las gráficas (C1,C2,C3) y (D) muestran la estimación de las coordenadas y la distancia al objeto respectivamente, mientras que (E) muestra el error de la estimación de la distancia.

En la Figura 18 se observa una simulación en la que se considera una constante de convergencia K=1.3, además se

añade ruido gaussiano con una relación señal-ruido de 20 dB a las coordenadas de imagen. Los resultados de la simulación, "A" y "B" muestran las señales de posición del objeto mientras "C" y "D" muestran las estimaciones con los observadores. Debido a su diseño, los observadores "OR" y "OC" presentan un buen desempeño ante la señal corrupta, mientras la implementación del observador "DN" no logra la convergencia con los valores propuestos.

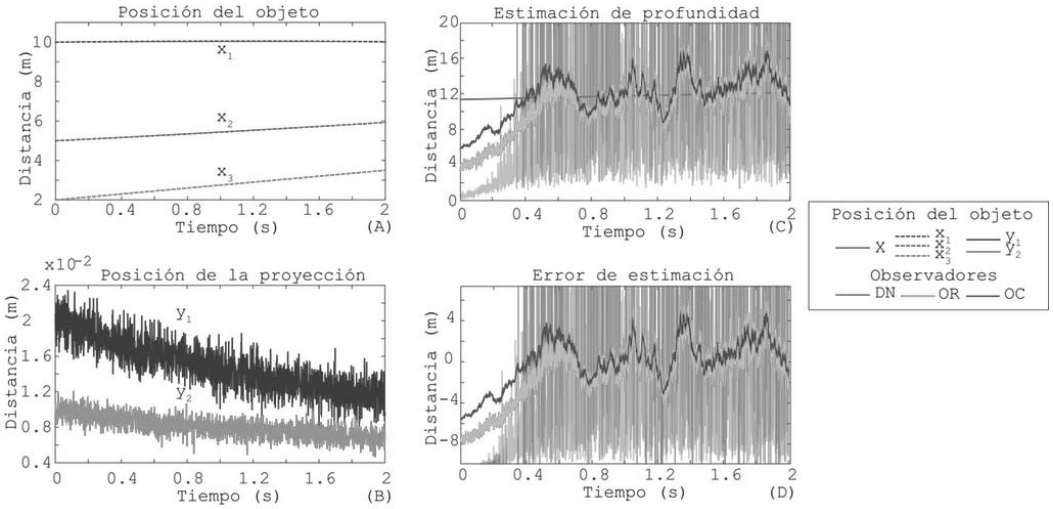

Figura 18: Simulación con ruido

La gráfica (A) muestra las coordenadas del objeto, (B) muestra la posición del objeto en el plano de imagen con una relación señal-ruido de 20 dB. En la gráfica (C) se presenta la estimación de la distancia al objeto y en (D) el error de esta estimación.

Al comprobar que la implementación de los observadores presenta convergencia, se realiza la simulación de la dinámica del diseño experimental, para conocer el desempeño y comportamiento de la estimación bajo la restricción impuesta. Las condiciones iniciales son:

- $X(0) = [\ 1 \quad 0 \quad 2.5\ ]^T$

- $\dot{T} = [\ 0 \quad 0.2*\mathrm{Cos}(\pi\ t) \quad 0\ ]^T$

[Resultados]

- $W = [\ 0 \quad 0 \quad 0\ ]^T$

- $K = 1*10^3$

Se emplea K_z=1 y $\hat{\zeta}(0)=\zeta\big(y(0),\dot{T}(0)\big)$ para los observadores "OR" y "OC". Para el observador "OC" se utiliza $\hat{z}(0)=1$ y $\hat{y}(0)=y(0)$, para el observador "OR" se considera z(0)=1.5 y para el observador "DN" se emplea α_o=5. La simulación se realiza de 0 a 10 segundos con un tamaño de paso de $1*10^{-3}$ s.

En la Figura 19 se observa la dinámica planteada en el diseño experimental. Los observadores presentan convergencia con un error mayor que en las simulaciones anteriores debido al cambio oscilatorio de la posición en las coordenadas de "x_2". La gráfica (A) muestra la posición del objeto y (B) muestra las coordenadas de la proyección, en (C) se muestra el resultado de la estimación de la profundidad y (D) el error.

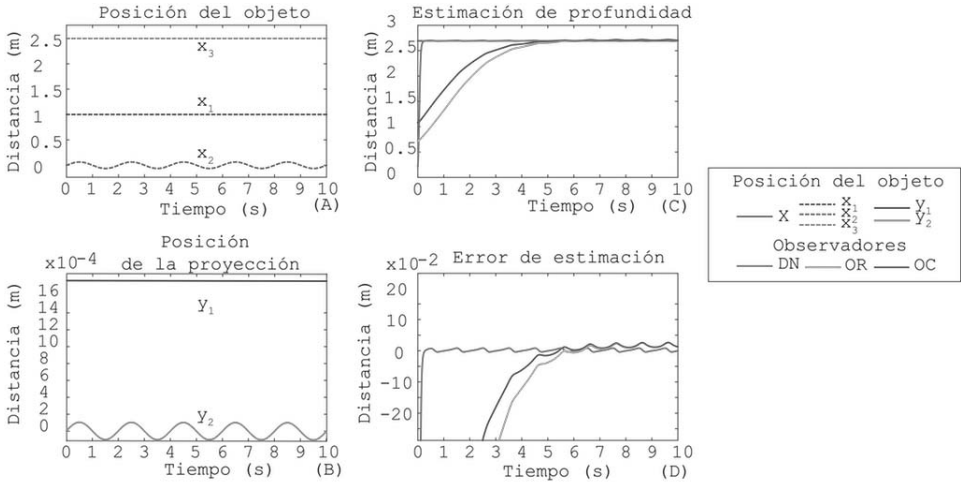

Figura 19: Simulación de la dinámica

La gráfica (A) muestra las coordenadas del objeto, (B) muestra la posición del objeto en el plano de imagen. En la gráfica (C) se presenta la estimación de la distancia al objeto y en (D) el error de esta estimación.

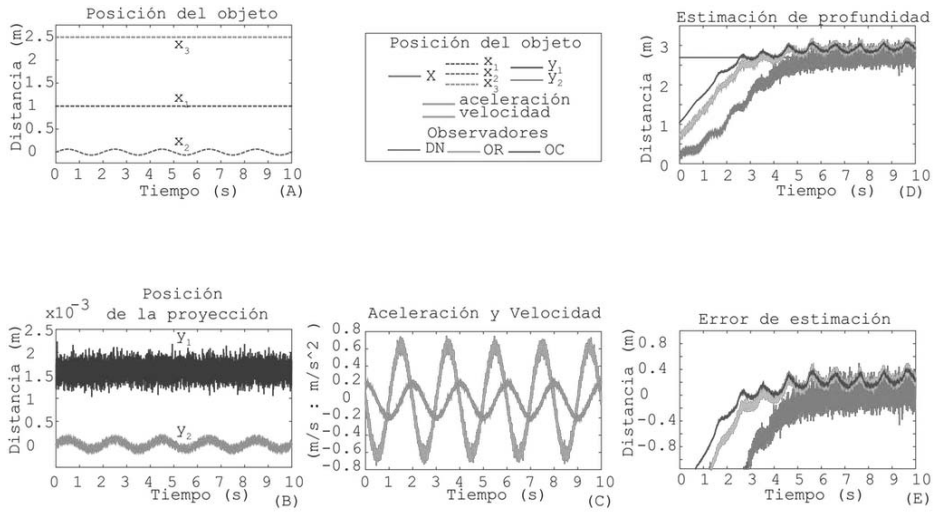

Figura 20: Simulación de la dinámica con ruido

La gráfica (A) muestra las coordenadas del objeto, (B) muestra la posición del objeto en el plano de imagen, (C) muestra la velocidad y aceleración de la cámara en el eje "x_2". En la gráfica (D) se presenta la estimación de la distancia al objeto y en (E) el error de esta estimación.

La Figura 20 muestra una simulación en la que se emplea una constante K=50 para el observador "DN". Se añade ruido gaussiano blanco con una relación ruido-señal de 20 dB a las señales proyección, velocidad y aceleración del sistema. Se observa que la estimación del observador "OC" es menos susceptible al ruido de las señales dinámicas que los otros dos observadores. La gráfica (A) muestra la posición del objeto y (B) su proyección en el plano de imagen. (C) muestra la velocidad y aceleración de la cámara. En (D) se muestra el resultado de la estimación de la profundidad mientras en (E) el error.

Las simulaciones comprueban que la implementación de los observadores es correcta, así como la factibilidad del diseño experimental. Los tres observadores son capaces de resolver la estimación de la distancia empleando un movimiento oscilatorio en una sola dimensión, sin rotación de la cámara.

En la Tabla 1 se resumen las simulaciones realizadas, se muestran los errores cuadráticos medios para los tres observadores en cada simulación empleando

$$ERMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{X}_i - X_i\right)^2}\,, \qquad (49)$$

En donde "N" es la cantidad de datos, "$\hat{X}_i$" es el valor estimado y "X_i" es el valor real. El mejor comportamiento lo presenta el observador de "OC". La diferencia del error entre "OC" y "OR" no es tan notoria como lo es con el observador "DN".

Para analizar el comportamiento de los observadores con la variación de K se realizan simulaciones con las mismas condiciones que la simulación 3. En la Tabla 2 se presenta los resultados obtenidos. Para los tres observadores conforme K aumenta, el error también lo hace, pero el tiempo de convergencia τ_c (s) disminuye.

De manera similar se observa el comportamiento de los observadores para K_z. En la Tabla 3 se muestran los resultados obtenidos en donde, de manera similar a los resultados anteriores, el error aumenta conforme aumenta el valor de KZ y el tiempo disminuye.

Tabla 1: Error RMS de Simulaciones

Simulación	Error RMS		
	"DN"	"OR"	"OC"
Simulaciones de validación			
1	0.1450	0.0470	0.0282
2	202.64	1.87	1.82
Simulaciones de diseño experimental			
3	0.603	0.0160	0.0152
4	0.1324	0.2244	0.2247

Media cuadrática del error en las simulaciones de los tres observadores implementados.

Tabla 2: Variación del error con K

Simulación	K	Variación del error con K					
		"DN"		"OR"		"OC"	
		ERMS	τ_c	ERMS	τ_c	ERMS	τ_c
5	50	$5.8*10^{-4}$	10	$2.8*10^{-3}$	10	$2.9*10^{-3}$	10
6	100	0.0015	4	0.077	6	0.057	6
7	500	0.0033	1	0.0154	5	0.0148	5
8	10^3	0.0043	0.25	0.016	5	0.012	5

Media cuadrática de los errores de los tres observadores implementados en las simulaciones variando K.

Tabla 3: Variación del error con K_z

Simulación	K_z	Variación del error con K_z					
		"DN"		"OR"		"OC"	
		ERMS	τ_c	ERMS	τ_c	ERMS	τ_c
9	10	-		0.033	0.5	0.033	0.5
10	100	-		0.039	0.25	0.039	0.025

Media cuadrática de los errores de los tres observadores implementados en las simulaciones variando K_z.

5.2 Software

A continuación se describe en general el código desarrollado para la estimación de profundidad. El cual puede ser consultado en su totalidad en [54].

Para realizar las pruebas de los observadores en MATLAB se elaboró una rutina que se encarga de simular el sistema de visión perspectiva (línea 11). Obteniendo todos los valores medibles y no medibles del sistema para evaluar a los observadores. Como parámetros de entrada tiene el tamaño de paso "*h*" de la simulación, el tiempo "*t*" y las coordenadas del vector posición en el tiempo anterior. En las líneas 2 y 4 encontramos dos funciones, la primera nos permite obtener la velocidad del sistema a partir de las ecuaciones que son definidas dependiendo de la dinámica que se quiera evaluar. La segunda nos permite obtener la aceleración con base en los valores calculados de la velocidad, empleando una diferenciación numérica. En las líneas 15 y 16 se calculan las coordenadas de la proyección empleando las componentes del vector posición, la constante "*PPM*" es la constante de escalamiento pixel/metro.

```
1   [ax, vx, y, gx, xn] = Simulación del sistema(x1,x2,x3, t, h)
    {
      // Se obtiene la velocidad en el tiempo t
2     [vx1,vx2,vx3] = Dinámica de la cámara(t);
      // Se almacena las velocidades en un arreglo (vector)
3     vx=[vx1;vx2;vx3];
      //Se calcula la aceleración
4     [ax1,ax2,ax3] = Derivación de la velocidad(t,h);
      //Se almacenan las aceleraciones en un arreglo
5     ax = [ax1; ax2; ax3];
      // Se asignan los valores de la velocidad angular
6     gx1 = 0; gx2 = 0; gx3 = 0;
      // Se genera un vector de velocidades angulares
7     gx = [gx1,gx2,gx3];
      // Se obtiene la matriz anti simétrica
8     A = [  0  , -gx3 ,  gx2;
9            gx3 ,   0  , -gx3;
```

```
10          -gx2 ,  gx1 ,   0  ];
      // Se obtiene el cambio de la posición
11    dx=A*x + vx;
      // Se integra para obtener la nueva posición
12    x1 = x1 + dx(1)*h;
13    x2 = x2 + dx(2)*h;
14    x3 = x3 + dx(3)*h;
      // Calculamos las coordenadas de la proyección
15    y1 = e* (x1/x3)/PPM;
16    y2 = e* (x2/x3)/PPM;
      // Se genera el vector posición de proyección
17    y = [ y1, y2]
      // Se almacenan las coordenadas en un arreglo
18    x = [x1;x2;x3];
    }
```

El siguiente paso es realizar las simulaciones de los observadores, para ello se emplea el programa "E*stimación*" encargado de obtener las estimaciones de posición del sistema. Las líneas 4-7 realizan la integración de los valores que regresan las líneas 2 y 3 en las cuales la función "*Observador*" implementa las ecuaciones del observador y los parámetros de entrada son en el tiempo actual y el tiempo previo.

```
1   [Xest,Yest,Dsedaest,v] = Estimación(ax1,ax2,ax3,vx1,vx2,vx3,y1,y2
                            ,y1est,y2est,vest,dsedaest,cnt,h,K,Kz,A)
    {
      // Se obtiene la derivada de V y ζ
2     [dvi, dy1i, dy2i, ddsedai] = Observador ( // t(cnt) )
3     [dv1, dy11, dy21, ddseda1] = Observador ( // t(cnt-1) )
      // Se integra el valor de V, ζ e Y
4     v = vest(cnt) + ( (dvi+dv1)*(h/2));
5     y1e = y1est(cnt) + ( (dy1i+dy11)*(h/2));
6     y2e = y2est(cnt) + ( (dy2i+dy21)*(h/2));
7     dsedae = dsedaest(cnt) + ((ddsedai+ddseda1)*(h/2));
      // Se obtiene la estimación de z
8     z = v + dsedae;
      // Se obtienen las estimaciones
9     Yest = [ y1e, y2e];
10    Dsedaest= dsedae;
11    Xest = [ y1e/z, y2e/z, 1/z];
    }
```

En la rutina "*Observador*" las líneas 2-7 son la implementación de las funciones (29-34) y las líneas 14-17

son propiamente el observador “OC”. Los programas para los otros observadores son similares y se sustituyen las ecuaciones correspondientes a cada observador.

```
1   [dv,dy1,dy2,ddseda] = Observador( ax1,ax2,ax3,vx1,vx2,vx3,y1,y2
                                  ,y1est,y2est,vest,dsedaest,K,Kz,A)
    {
      // Se calculan las funciones de los observadores
2     alpha = getAlpha(vx1,vx2,vx3,y1,y2,K);
3     gamma = getGamma(ax1,ax2,ax3,y1,y2,K);
4     dseda = getDseda(vx1,vx2,vx3,y1,y2,K);
5     f1 = getF1(y1,y2,A);
6     f2 = getF2(y1,y2,A);
7     f3 = getF3(y1,y2,A);
      // Simplificaciones para el observador
8     w1  =   ( (vx1) - (vx3*y1) );
9     w2  =   ( (vx2) - (vx3*y2) );
10    vz = vest + dseda;
      // Errores de las estimaciones
11    y1error = y1est - y1;
12    y2error = y2est - y2;
13    dsedaerror= dsedaest-dseda;
      // Implementación del observador
14    dv = (f3*vz)-(vx3*(vz^2))-(alpha*vz)-(K*w1*f1)-(K*w2*f2)-gamma;
15    dy1 = f1 + (w1*vz) - (alpha*y1error);
16    dy2 = f2 + (w2*vz) - (alpha*y2error);
17    ddseda = (alpha*vz)+(K*w1*f1)+(K*w2*f2)+gamma-(Kz*dsedaerror);
    }
```

A continuación se describe la programación de la aplicación para el sistema operativo Android. Se cuentan con tres rutinas principales.

La primera consiste en la programación del algoritmo de seguimiento. En donde el parámetro de entrada es la imagen digital obtenida por el sistema operativo en formato YUV. Se realiza un ciclo condicional (línea 2) para definir el grado de tolerancia requerido. Se observó que era suficiente con una iteración para que el algoritmo mantuviera enfocado al objeto en todo momento. En la línea 3 la función obtiene la posición y tamaño de la región de interés en la imagen digital. Después se convierte la ROI de la imagen digital al formato RGB empleando la imagen digital, las dimensiones de la región de interés y las dimensiones de la imagen digital.

A partir de esta sub imagen RGB obtenida en la línea 4 se calcula la imagen de probabilidad (línea 5) la cual se emplea para obtener el centroide (línea 6) y finalmente se almacena la nueva coordenada de la región de interés.

```
1   [void ] = Seguimiento(imagen)
    {
      // El algoritmo se repite n veces
2     for(iteración=0;( iteración <1); iteración ++)
      {
3        dimROI=DefinirDimensionesROI();
         // Convertimos la imagen  a RGB
4        imagenRGB=YUVaRGB(imagen,dimROI,dimVP);
         // Se obtiene la imagen de probabilidad.
5        imagenProbabilidad = CalcularImagenProbabilidad(imagenRGB);
         // Se obtiene un vector de las coordenadas del centroide
6        centroide= CalcularCentroide(imagenProbabilidad,dimROI);
         // Definimos las nuevas coordenadas de la ventana de búsqueda
         coordenadasROI=CalcularCoordenadasROI (centroide,
7                                  coordenadasPrevias);
      }
    }
```

Para obtener los valores inerciales se emplea una rutina del sistema operativo Android encargada de monitorear los sensores inerciales. Cuando esta rutina detecta algún cambio en los valores de los sensores llama a la función "*onSensorChanged*". El parámetro de esta rutina es el evento del sensor, el cual contiene los valores y el tipo de sensor que realizó la llamada a esta función. Es aquí en donde se obtiene las señales requeridas por el observador. Las líneas 6-9 y 11-15 se encargan de obtener los valores del sensor y activar una bandera ("*giroscopolisto*" o "*acelerometrolisto*") que indica que se ha realizado la lectura del sensor. Cuando se han tomado ambas lecturas (línea 15) se procede a almacenar los valores en los arreglos (líneas 16 y 17) que serán los parámetros de entrada del observador. En la línea 18 se obtiene el tiempo de adquisición, empleando la constante "*N2S*" se transforman los nanosegundos a segundos. En la línea 19 la función "*Obtenercentroide*" se encarga de

obtener las coordenadas del objeto en la imagen de la rutina de seguimiento.

```
1    [void] = onSensorChanged(evento)
     {
       // Revisamos si es el primer procesamiento.
2      if(esprimerprocesamiento)
       {
         // Se obtiene el tiempo inicial del sistema
3        tiempoinicial=System.nanoTime();
         // No es la primera iteración
4        esprimerprocesamiento=falso
       }
       // Revisamos cual sensor activo el evento
5      if (//giroscopio)
       {
         // Se obtiene los valores del giróscopo
6        gyrox = evento.values[DEP.X]
7        gyroy = evento.values[DEP.Y]
8        gyroz = evento.values[DEP.Z]
9        giroscopolisto=verdadero;
       }
10     if(//acelerómetro)
       {
         // Se obtiene los valores del acelerómetro.
11       accelx = evento.values[DEP.X];
12       accely = evento.values[DEP.Y];
13       accelz = evento.values[DEP.Z];
14       acelerómetrolisto=verdadero;
       }
       // Revisamos si se han realizado ambas adquisiciones.
15     if (giroscopolisto && acelerometrolisto)
       {
         // Asignamos los valores a los arreglos de almacenamiento.
16       xGyro[cnt] = gyrox; yGyro[cnt] = gyroy; zGyro[cnt] = gyroz;
17       xAccel[cnt]=accelx; yAccel[cnt]=accely; zAccel[cnt]=accelz;
         // Se obtiene el tiempo
18       tiempo[cnt] = (System.nanoTime() - tiempoinicial)*N2S;
         // Se obtiene los valores de la imagen.
19       centroide = Obtenercentroide();
         // Cambiamos a la siguiente posición de los arreglos.
20       cnt++;
21       giroscopolisto =falso; acelerometrolisto =falso;
       }
     }
```

Finalmente, se procesan las señales adquiridas y se evalúa el observador con la rutina "*EstimarCoordenadas*".

En las líneas 3-10 se realiza la transformación entre sistemas (dispositivo móvil - observador). La Figura 21-A

muestra el sistema de coordenadas del dispositivo para los sensores inerciales [X Y Z] y en líneas punteadas el sistema de coordenadas que emplea el observador [x_1 x_2 x_3]. El sistema de coordenadas de la imagen se esquematiza en 21-B, en donde [y x] son las coordenadas que maneja el sistema operativo y las coordenadas del plano de imagen en el observador son [y_1 y_2].

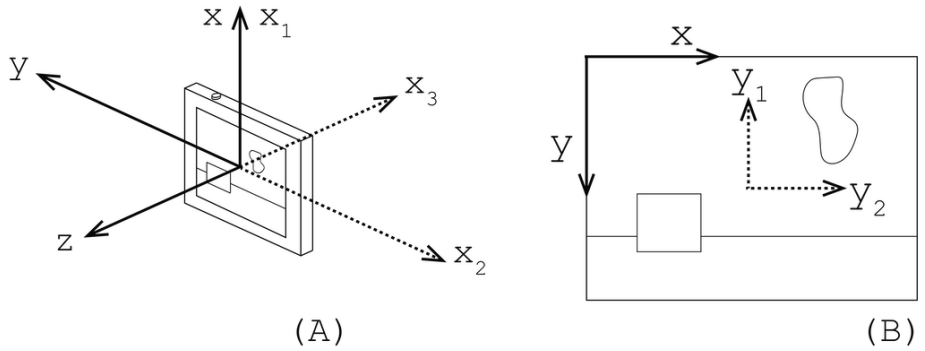

Figura 21: Coordenadas del dispositivo móvil

En la figura arriba se muestra el sistema de coordenadas para el dispositivo móvil y el sistema de coordenadas para el observador. En la otra figura se muestra el sistema de coordenadas de la imagen y el sistema de coordenadas que se emplea en el observador.

Las líneas 12-16 se encargan de obtener la aceleración, velocidad y posición de la cámara. Después se procede a calcular las condiciones iniciales del observador 17-25. Finalmente en las líneas 29-35 se almacenan las estimaciones del observador para ser graficadas y o guardadas.

```
1       void = EstimarCoordenadas( //Mediciones inerciales y de visión
        )
        {
2         for (int i=0; i<numerodemediciones; i++)
          {
3           ax1RAW[i] = -xAccel[i];
4           ax2RAW[i] = yAccel[i];
5           ax3RAW[i] = zAccel[i];
6           gx1RAW[i] = -xGyro[i];
7           gx2RAW[i] = yGyro[i];
```

```
8          gx3RAW[i] = zGyro[i];
9          y1[i]  = -(yY[i]- Y1C) * ( PPM/ϵ );
10         y2[i]  =  (yX[i]- Y2C) * ( PPM/ϵ );
11         t[i]   = time[i];
         }
12       FiltrarDatos( );
13       RotarAceleración( );
14       IntegrarAceleración( );
15       FiltrarVelocidad( );
16       IntegrarVelocidad( );
         // Constantes del observador.
17       zoest = 1;
18       K = 50;
19       Kz = 10;
         // Obtenemos las condiciones iniciales.
20       for (int i=0;i<3;i++)
         {
21         y1est[i] = y1[i]; y2est[i] = y2[i];
22         x1est[i] = y1est[i]/(zoest); x2est[i] = y2est[i]/(zoest);
23         x3est[i] = 1/zoest;
24         dsedaest[i]                                                =
       getDseda(vx1[i],vx2[i],vx3[i],y1[i],y2[i],K);
25         vest[i] = zoest-dsedaest[i]
         }
         // Realizamos la estimación con el observador "OC"
26       for (int i=3;i<numerodedatos;i++)
         {
27         h=t[i]-t[i-1];
28         estimación =Observador( ax1[i-1],ax2[i-1],ax3[i-1]
                                  ,ax1[i-2],ax2[i-2],ax3[i-2]
                                  ,vx1[i-1],vx2[i-1],vx3[i-1]
                                  ,vx1[i-2],vx2[i-2],vx3[i-2]
                                  ,y1[i-1],y2[i-1],y1[i-2],y2[i-2]
                                  ,y1est[i-1],y2est[i-1]
                                  ,y1est[i-2],y2est[i-2]
                                  ,vest[i-1],vest[i-2]
                                  ,dsedaest[i-1],dsedaest[i-2]
                                  ,h,K,Kz
                                  ,gx1[i-1],gx2[i-1],gx3[i-1]
                                  ,gx1[i-2],gx2[i-2],gx3[i-2] );
           //   Guardamos   las   estimaciones   en   los   arreglos
       correspondientes
29         vest[i] =estimación[V];
30         x1est[i] = estimación [X1]
31         x2est[i] = estimación [X2]
32         x3est[i] = estimación [X3]
33         y1est[i] = estimación [Y1]
34         y2est[i] = estimación [Y2];
35         dsedaest[i] = estimación [DSEDA];
         }
       }
```

5.3 Estimación de la profundidad

En la Figura 22 se muestra la interfaz gráfica final del programa, mediante la cual se controlan todas las rutinas de la estimación de profundidad. Y el resultado del algoritmo de seguimiento. El objeto se encuentra dentro de la ROI, en la cual se superpone la imagen de probabilidad calculada (recuadro blanco y negro), lo que facilita al usuario seguir el movimiento del objeto y permite evaluar cualitativamente la calidad de la imagen de probabilidad.

La aplicación obtiene los datos de aceleración, posición de la proyección y velocidad angular, a una frecuencia de muestreo de ~ 120 Hz realizando un sobre-muestreo de los valores de la cámara.

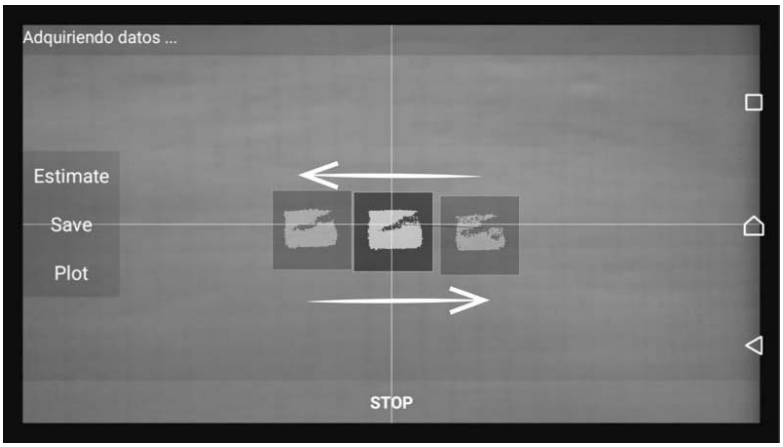

Figura 22: Resultado de la rutina de seguimiento

En la parte central se muestran tres posiciones del objeto en movimiento. En la ventana que rodea al objeto se muestra la imagen de probabilidad en donde el objeto se muestra en blanco y el fondo en negro. Las flechas indican el movimiento horizontal del objeto.

En la Figura 23 se muestran las tres señales entregadas por los sensores del dispositivo móvil (A), (B) y (C). En las tres se aprecia el ruido; como requerimos de integrar la aceleración, se elimina el ruido mediante un filtro LPF (D)

obteniendo la velocidad angular filtrada (E) y la aceleración filtrada (G). Se realiza una rotación (F) de la aceleración empleando los valores de velocidad angular y después se integra (H) para obtener la velocidad. Después se elimina cualquier deriva de la velocidad mediante un filtro HPF (I) y finalmente se obtiene la velocidad lineal filtrada (J).

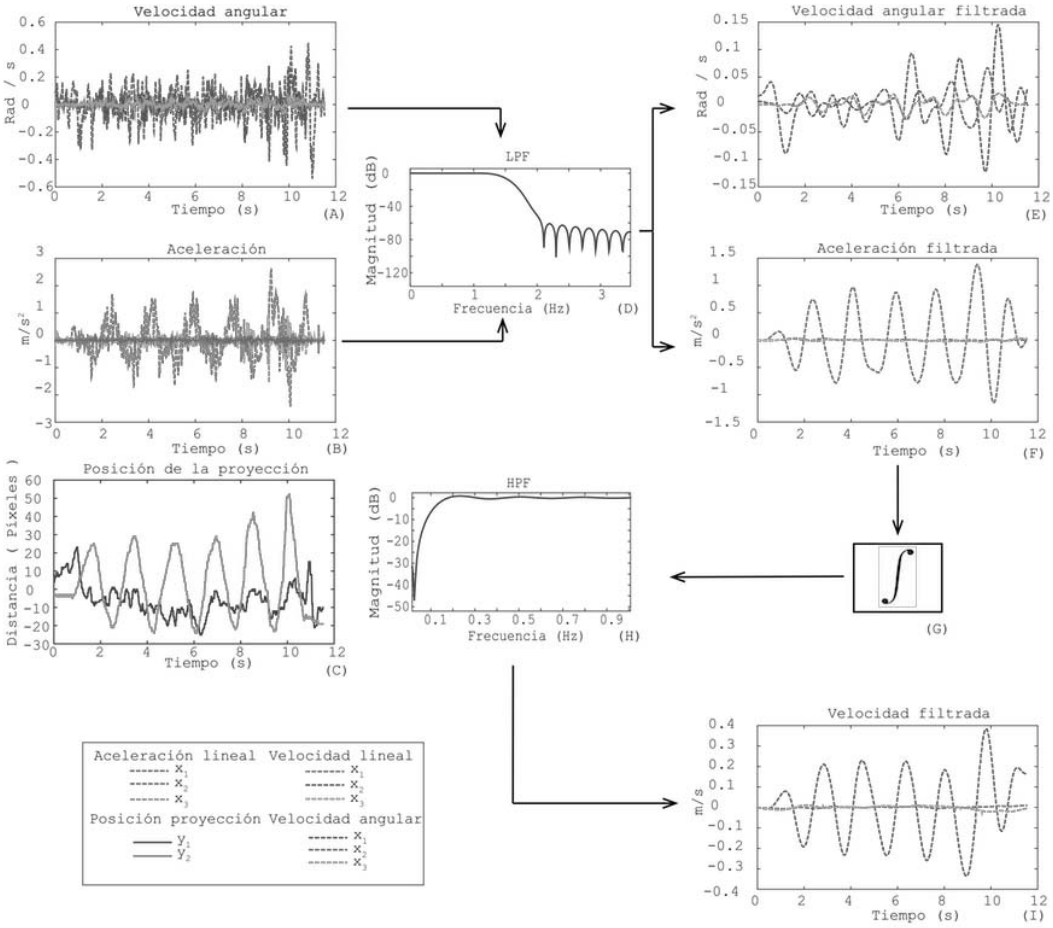

Figura 23: Procesamiento de los datos adquiridos

En la figura (A),(B) y (C) son las señales que adquieren los sensores del dispositivo móvil, (D) y (H) muestran la respuesta de los filtros. Las gráficas (D), (G) y (J) muestran los resultados del procesamiento, los cuales son los que se pasan a la rutina del observador. Los recuadros (F) y (H) indican la rotación de la aceleración hacia el sistema de referencia, empleando la velocidad angular y la integración de la misma.

Los filtros son de orden 700 y se diseñan empleando la herramienta “fdatool” de Matlab, se utiliza una ventana

gaussiana con α = 2.5 y una frecuencia de corte de 1.5 Hz para el LPF y 0.1 para el HPF (D) e (I) muestran la respuesta de magnitud de los filtros respectivamente. Para cargar los filtros al dispositivo móvil se añaden dos archivos de texto en los cuales cada línea contiene un coeficiente del filtro.

Finalmente los valores de velocidad y aceleración lineal así como la posición de la proyección y velocidad angular pasan al observador y son procesados para ser desplegados en la pantalla del dispositivo (Figura 24). Si se desea realizar un análisis posterior se almacenan los datos y se exportan en formato CSV (Coma Separated Value).

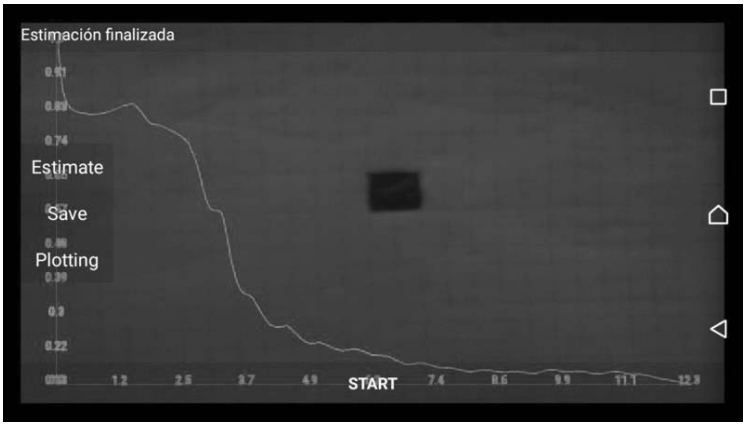

Figura 24: Estimación en el dispositivo móvil

En la imagen se observa cómo es que se muestra el resultado obtenido en la aplicación.

En los experimentos realizados se emplea un tamaño de imagen de 980x720 pixeles. La distancia focal de 0.0049 m se obtiene del fabricante al igual que el tamaño del sensor (6.3 x 4.7 mm) [55]. Para el observador se utiliza $K = 50$ y $K_z = 2$.

La Figura 25 muestra la estimación de la distancia "B" a un objeto que se encuentra a $X = [\ 0 \quad 0 \quad 1.6\ ]^T$ m del dispositivo móvil, al realizar un movimiento oscilatorio "A" sobre el

eje "x_2" de ~0.05 m (experimento 7). Los tres observadores convergen y llegan a estabilizarse alrededor de 1.6 m. El observador "OC" y "OR" presentan un mejor comportamiento con un error cuadrático medio (4 %) menor que el observador DN(9 %).

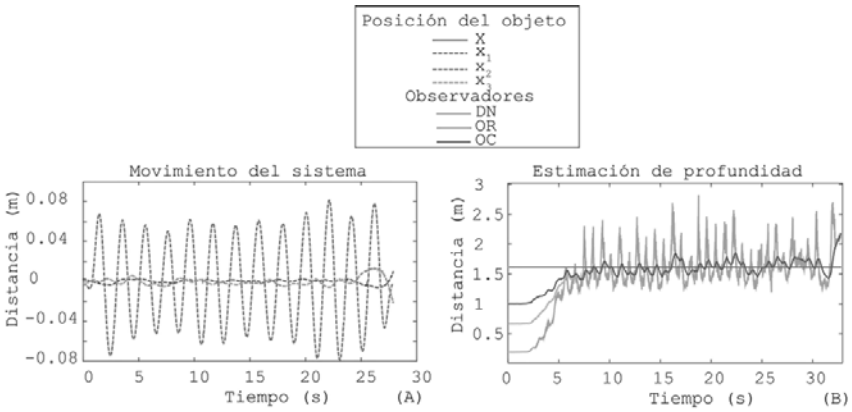

Figura 25: Estimación a 1.6 m

En la figura la gráfica (A) muestra la oscilación que presenta la cámara sobre su eje x_2. En (B) se observa el resultado de la estimación.

En la Figura 26 se muestra el resultado obtenido al colocar el objeto a $X=[-0.05 \quad 0.13 \quad 0.43]^T$ m del dispositivo. Se observa que los observadores convergen a la distancia (0.45 m). Los tres son capaces de obtener las coordenadas del objeto en el sistema de referencia de la cámara (A),(B) y (C). Los observadores OC y OR obtienen un error de 16% mientras que DN obtiene un error del 20%. En (B) se omite la oscilación de la cámara en la gráfica de x_2 para que sea más claro cómo los observadores se adhieren a la trayectoria seguida por la cámara, la cual oscila en la coordenada x_2 = 0.13. En este experimento, al contar con una componente importante sobre el eje x_2, podemos observar cómo el observador OC tiene una estimación más fiel que el observador OR, al realizar el cálculo de $\hat{Y}$. Esta diferencia no es tan clara si solamente se compara la distancia, pues depende de

cuánto contribuyen las coordenadas x1 y x2, que son las que contienen el ruido

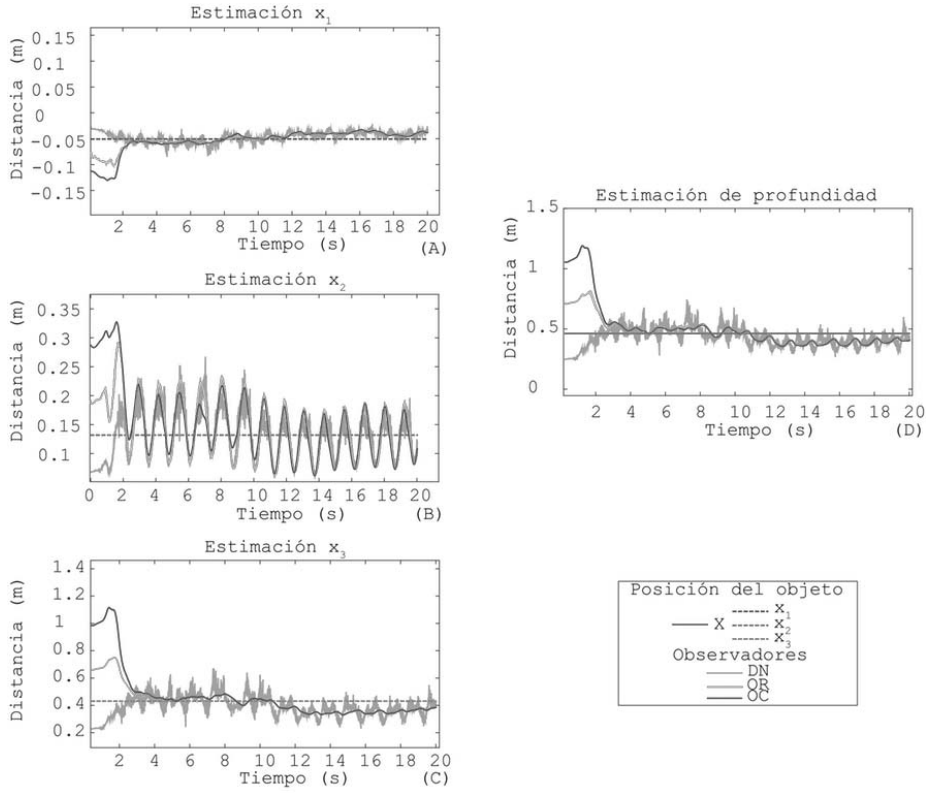

Figura 26: Estimación a 0.2 m

En la figura la gráfica (A),(B) y (C) muestran la estimación de las tres coordenadas del objeto en el sistema de la cámara. En (D) se observa la distancia entre el objeto y la cámara.

En la Tabla 4 se presentan los diferentes experimentos en los cuales se varía la distancia a la que se coloca el objeto. En todos los experimentos reportados los tres observadores presentaron convergencia. El error RMS se consideró a partir de la estabilización de los tres observadores, aproximadamente después de 10 s para todos los experimentos. El porcentaje de error se obtiene al dividir el error RMS entre la distancia medida. El observador OC presenta el menor error lo que muestra la ventaja de realizar la estimación de $\hat{\zeta}$ e $\hat{y}$ en comparación del observador OR que realiza la estimación de $\hat{\zeta}$ y el observador DN el cual estima solamente las coordenadas del objeto.

Tabla 4: Experimentos variando la distancia

Experimento	$\|\hat{X}\|$ (m)	DN		OR		OC	
		ERMS	Error %	ERMS	Error %	ERMS	Error %
1	0.2	.03	15	.02	10	0.2	10
2	0.45	0.09	20	0.07	16	0.07	16
3	0.5	.10	20	.04	8	0.04	8
4	1	.11	11	.03	3	.03	3
5	1.5	.24	16	.08	5	.08	5
6	1.5	.16	10	.06	4	.06	4
7	1.6	.15	9	.06	4	.06	4
8	1.6	.26	17	.09	6	0.09	6
9	3	.5	17	.26	8	.26	8

Error que presentaron los tres observadores en los distintos experimentos variando la distancia al objeto.

Se observa que el error se hace mayor conforme la distancia aumenta o disminuye, lo que se atribuye a la resolución de la cámara (tamaño de la imagen). El error debido a una distancia mayor se debe a que cada pixel encierra un área del plano ubicado a la distancia del objeto, entre más lejos se encuentre el objeto de la cámara el área abarcada por cada pixel se hace mayor y la incertidumbre de la posición del objeto aumenta. Aunque es de esperar que una distancia menor entre el objeto y la cámara presentara una menor incertidumbre en la posición y resultara en una mejor precisión esto no ocurre debido a la capacidad de procesamiento del dispositivo móvil. En este segundo caso el objeto abarca un área de la imagen la cuál crece conforme el

objeto se encuentra más cerca, lo que se traduce en más pixeles a procesar. En la Tabla 5 se ejemplifican las dos situaciones planteadas. De las cuales se realizaron mediciones entre distintas resoluciones, se observó que con una resolución menor mejora la estimación a una distancia de 0.5 m pues el procesamiento de la imagen requiere menos tiempo y el desfase en la proyección de imagen y las mediciones inerciales es menor que con una resolución mayor. En el caso de una distancia de 3 m el área de procesamiento es menor pero la incertidumbre de la posición del objeto aumenta drásticamente el error.

Tabla 5: Experimentos variando resolución

Experimento	$\|\hat{X}\|$ (m)	DN		OR		OC	
		ERMS	Error %	ERMS	Error %	ERMS	Error %
640x480	0.5	.04	7	0.03	5	0.03	5
980x720	0.5	.10	20	.04	8	0.04	8
640x480	1	.08	8	.10	10	.11	11
980x720	1	.11	11	.03	3	.03	3
640x480	3	.48	16	.37	12	.37	12
980x720	3	.5	17	.26	8	.26	8

Error que presentaron los tres observadores en los distintos experimentos variando la resolución de la imagen.

6 Conclusiones

Se implementó con éxito un sistema de visión perspectiva y navegación inercial en un dispositivo móvil con sistema operativo Android, empleando los sensores comerciales del dispositivo.

El sistema permite realizar el seguimiento de un objeto en el plano de imagen, obtener la velocidad y aceleración de la cámara y estimar las coordenadas del objeto empleando los tres programas elaborados para ello, sin requerir de equipo adicional, permitiendo que el sistema pueda ser utilizado como una plataforma de experimentación en el desarrollo de los observadores, o bien ser usado en aplicaciones de navegación.

Se realizó exitosamente la programación de los observadores propuestos para su implementación en este trabajo, coincidiendo en las simulaciones con los resultados presentados en [49] . Los observadores OR y OC presentan convergencia exponencial mientras el observador DN presenta convergencia asintótica. Los observadores OC y OR presentan una gran estabilidad ante ruido en las señales de imagen e inerciales. La diferencia entre ambos se observa en las coordenadas paralelas al plano de imagen, pues el observador OC realiza la estimación de las coordenadas de la proyección, lo que hace que el error en estas coordenadas sea inferior que el observador OR. El diseño del observador DN no considera estimaciones adicionales para la disminución del error por ruido, lo que hace que su implementación presente un error mayor que la de los observadores OC y OR. Por lo anterior, el observador de orden completo fue el que se eligió para su programación en el dispositivo móvil.

La programación del dispositivo móvil se desarrolló teniendo en cuenta una programación lineal orientada a objetos, esto simplificó su desarrollo al no requerir la sincronización entre hilos y simplificó la división de las rutinas encargadas del procesamiento. La interfaz de usuario, al contener las funciones básicas, permite al usuario enfocarse en el movimiento de la cámara y verificar de manera rápida el resultado de la estimación.

Se logró obtener las distancia entre el objeto y la cámara a partir del movimiento realizado a mano alzada. Se observó que el sistema se encuentra limitado por la sensibilidad y frecuencia de los sensores. Siendo los de mayor importancia en este caso particular, la resolución de la imagen y su tiempo de procesamiento. Se obtuvo un error inferior al 10% para los dos observadores propuestos, lo que muestra su estabilidad ante perturbaciones. Si se requiere de calcular mayores distancias se debe modificar la resolución y amplitud del movimiento de la cámara. Es factible emplear el dispositivo móvil para realizar pruebas de observadores y en aplicaciones de navegación. El filtro usado nos permite una rápida y sencilla implementación y modificación de sus parámetros, pero limita el movimiento del sistema a una dinámica oscilatoria sobre uno de los ejes.

7 Referencias

[1] R. D. Fernald, «Evolution of eyes,» *Current opinion in neurobiology,* vol. 10, nº 4, pp. 444-450, 2000.

[2] D. Mar, Vision a computational investigation into the human representation and processing of visual information, London,England: MIT Press, 2010.

[3] R. Szeliski, Computer Vision: algorithms and applications, Springer, 2010.

[4] D. A. Forsyth and J. Ponce, Computer vision: a modern approach, Prentice Hall, 2011.

[5] S. Mori, C. Y. Sue y K. Yamamoto, «Historical review of OCR research and development,» *Proceedings of the IEEE,* vol. 80, nº 7, pp. 1029 - 1058, 1992.

[6] T. McInerney y D. Terzopoulos, «Deformable models in medical image analysis: a survey,» *Medical Image Analysis,* pp. 91-108, 1996.

[7] A. R. Jiménez, R. Ceres y J. L. Pons, «Survey of computer vision methods for locationg fruit on trees,» *Transaction of the ASAE,* vol. 43, nº 6, pp. 1911-1920, 2000.

[8] N. Buch, S. A. Velastin y J. Orwell, «A review of computer vision techniques for the analysis of urban traffic,» *IEE transactions on intelligent transportation systems,* vol. 12, nº 3, pp. 920-938, 2011.

[9] A. Elgammal, R. Duraiswami, D. Harwood y L. S. Davis, «Background and forefround modeling using nonparametric

kernel density estimation for visual surveillance,» *Proceedings of the IEE,* vol. 90, n° 7, pp. 1151-1163, 2002.

[10] D. G. Lowe, «Object Recognition from local scale-invariant features,» *The proceedings of the seventh IEE international conference on computer vision,* vol. 2, pp. 1550-1557, 1999.

[11] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, P. Golland and D. M. Sabatini, "CellProfiler: image analysis software for identifying and quantifying cell phenotypes," *Genome Biology,* vol. 7, no. 10, p. R100, 2006.

[12] G. K. M. Cheung y J.-Y. Bouguet, «A real time system for robust 3D voxel reconstruction of human motions,» *Proceedings IEEE conference on computer vision and pattern recognition,* vol. 2, pp. 714 - 720, 2000.

[13] S. Izadi, D. Kim, O. Hilliges, D. Molynewaux, R. Newcombe, P. Kholi, J. Shotton, S. Hodges, D. Freeman, A. Davison y A. Fitzgibbon, «KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera,» *Proceedings of the 24th annual ACM symposium on User interface software and technology,* pp. 559-568, 2011.

[14] S. Zhang, «Recent progresses on real-time 3D shape measurement using digital fringe projection techniques,» *Optics and lasers in engineering,* vol. 48, n° 2, p. 149–158, 2010.

[15] P. I. Corke, «Visual control of robot manipulators-a review,» *Visual servoing,* vol. 7, pp. 1-31, 1993.

[16] K. M. Shaaban y N. M. Omar, «3D information extraction using Region-based Deformable Net for monocular robot navigation,» *Journal of Visual Communication and Image Representation,* vol. 23, nº 2, pp. 397-408, 2012.

[17] R. A. Jarvis, «A perspective on range finding techniques for computer vision,» *IEEE transactions on pattern analysis and machine intelligence.,* Vols. %1 de %2PAMI-5, nº 2, pp. 122-139, 1983.

[18] Z. Zhang, Y. Han, Y. Zhou y M. Dai, «A novel absolute localization estimation of a target with monocular vision,» *Optik-international journal for light and electron Optics,* vol. 124, nº 12, pp. 1218-1223, 2013.

[19] A. P. Dani, N. R. Fischer, Z. Kan y W. E. Dixon, «Globally exponentially stable observer for vision-based range estimation,» *Mechatronics,* vol. 22, nº 4, pp. 381-389, 2012.

[20] Y. Ma, S. Soatto, J. Kosecka y S. S. Sastry, An invitation to 3-d vision: from images to geometric models, vol. 26, New York: Springer, 2004.

[21] E. Hecht, Optics, 4th ed., San Francisco: Addison Wesley, 2002.

[22] S.-Y. Lee y A. Ortega, «A novel approach of image compression in digital cameras with a bayer color filter array,» *Image Processing, 2001. Proceedings. 2001 International Conference on,* vol. 3, pp. 482-485, 2001.

[23] J. J. Báez Rojas y M. A. Alonso Pérez, «Uso del sistema HSI para asignar falso color a objetos en imágenes digitales,» *Revista mexicana de física e,* vol. 54, nº 2, pp. 186-192, 2008.

[24] J. Bruce, T. Balch y M. Veloso, «Fast and inexpensive color image segmentation for interactive robots,» *Proceedings of the 2000 international conference on intelligent robots and systems,* vol. 3, pp. 2061-2066, 2000.

[25] L. Grady, «Random walks for image segmentation,» *IEEE Transactions on pattern analysis and machine intelligence,* vol. 28, n° 11, pp. 1768-1783, 2006.

[26] L. Lucchese y S. K. Mitra, «Color image segmentation: A state of the art survey,» *Proceedings of the Indian National Science Academy* , vol. 67, n° 2, pp. 207-221, 2001.

[27] W. Song, «Real-time Moving Object Tracking System Using Cam-shift Algorithm,» *TELKOMNIKA Indonesian Journal of Electrical Engineering,* vol. 11, n° 11, pp. 6611-6617, 2013.

[28] D. Comaniciu, R. Visvanathan y P. Meer, «Real-Time Tracking of Non-Rigid Objects using Mean Shift,» *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference,* vol. 2, pp. 142-149, 2000.

[29] J. G. Allen, R. Y. Xu y J. S. Jin, «Object tracking using camshift algorithm and multiple quantized feature spaces.,» *Proceedings of the Pan-Sydney area workshop on Visual information processing,* pp. 3-7, 2004.

[30] H. Chu, S. Ye, Q. Guo y X. Liu, «Object tracking algorithm based on camshift algorithm combinating with difference in frame,» *Automation and Logistics, 2007 IEEE International Conference on,* pp. 51-55, 2007.

[31] J.-Y. Bouguet, « Visual methods for three-dimensional modeling (Doctoral dissertation, California Institute of Technology).,» 1999.

[32] Z. Zhang, «A flexible new technique for camera calibration,» *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 22, n° 11, pp. 1330-1334, 2000.

[33] R. Y. Tsai, «A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses,» *Robotics and Automation, IEEE Journal of,* vol. 3, n° 4, pp. 323-344, 1987.

[34] R. Feliz, E. Zalama y J. G. García-Bermejo, «Pedestrian tracking using inertial sensors,» *Journal of Pysical Agents,* vol. 3, n° 1, pp. 35-42, 2009.

[35] P. Corke, J. Lobo y J. Dias, «An Introduction to Inertial and Visual Sensing,» *The International Journal of Robotics Research,* vol. 26, n° 19, p. 519–535, 2007.

[36] S. Sukkarieh, «Low cost, high integrity, aided inertial navigation systems for autonomous land vehicles,» *Tesis Doctoral. The University of Sydney,* 2000.

[37] J. R. Nistler y M. F. Selekwa, «Gravity compensation in accelerometer measurements for robot navigation on inclined surfaces,» *Procedia Computer Science,* vol. 6, pp. 413-418, 2011.

[38] Ö. Bebek, M. A. Suster, S. Rajgopal, C. M. C¸avus ¸og ˘lu, X. Huang, D. J. Young, M. Mehregany, A. J. van den Bogert y C. H. Mastrangelo, «Personal Navigation via High-Resolution Gait-Corrected Inertial

Measurement Units,» *IEE Transactions on instrumentation and measurement,* vol. 59, n° 11, pp. 3018-3027, 2010.

[39] H. Zhou y H. Hu, «Reducing Drifts in the intertial measurements of wrist and elbow positions,» *IEE Transactions on insrumentation and measurements,* vol. 59, n° 3, pp. 575-385, 2010.

[40] M. Achtelik, M. Achtelik, S. Weiss y R. Siegwart, «Onboard IMU and Monocular vision based control for MAVs in unknown in and outdoor environments,» *IEE International conference on robotics and automation,* pp. 3056-3063, 2011.

[41] J. Steward, Calculus, Belmont: Brooks/Cole, 2011.

[42] S. C. Chapra y R. P. Canale, Numerical Methods for Engineers, New York: McGraw-Hill, 2010.

[43] Z. Said, K. Sundaraj y M. N. A. Wahab, «Depth Estimation for a Mobile Platform Using Monocular Vision,» *Procedia Engineering,* vol. 41, pp. 945-950, 2012.

[44] V. N. Dobrokhodov, I. I. Kaminer, K. D. Jones y R. Ghabcheloo, «Vision-BasedTracking and Motion Estimation for Moving targets using Small UAVs,» *Proceedings of the 2006 American Control Conference,* Vols. %1 de %21428-1433, p. 6, 2006.

[45] Y. L. Murphey, J. Chen, R. P., J. Croosman, J. Zhang y L. Sieh, «DepthFinder, A Real time Depth Detection System for Aided Driving,» *In Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE,* pp. 122-127, 2000.

[46] D. G. Luenberger, «Observing the state of a linear system,» *Military Electronics, IEEE Transactions on,* vol. 8, nº 2, pp. 70-80, 1964.

[47] N. S. Nise, Control Systems Engineering, Jefferson City: John Wiley & Sons, 2011.

[48] H. Inaba, A. Yoshida, R. Abdursul y B. K. Ghosh, «Observability of Perspective Dynamical Systems,» *In Decision and Control, 2000. Proceedings of the 39th IEEE Conference on,* vol. 5, pp. 5157-5162, 2000.

[49] I. Grave y Y. Tang, «A New Observer for Perspective Vision Systems Under Noisy Measurements,» *Automatic Control, IEEE Transactions on,* vol. 60, nº 2, pp. 503-508, 2015.

[50] W. Lohmiller y J.-J. E. Slotine, «On Contraction Analysis for Non-linear Systems,» *Automatica,* vol. 34, nº 6, pp. 683-696, 1998.

[51] D. G. Luenberger, Introduction to dynamic systems, Tehory, models and applications, New York: John Wiley & Sons, 1979.

[52] J. R. Kwapisz, G. M. Weiss y S. A. Moore, «Activity Recognition using Cell Phone Accelerometers,» *ACM SigKDD Explorations Newsletter,* vol. 12, nº 2, pp. 74-82, 2011.

[53] Sony Mobile, «Xperia Z1 Compact,» Sony Mobile Commuinications AB, 2014.

[54] R. Nieto-Fuentes, «DEP,» Octubre 2015. [En línea]. Available: https://github.com/NFR91/DEP. [Último acceso: 2015].

[55] Sony, «IMX220,» *Product Brief.*