

Centro Nacional de Investigación y Desarrollo Tecnológico

Subdirección Académica

Departamento de Ciencias Computacionales

TESIS DE MAESTRÍA EN CIENCIAS

**Detección de Objetos en Movimiento con Sensor RGB-D para
un Robot Móvil a Velocidad Constante**

presentada por
Ing. Alida Esmeralda Zárate Jiménez

como requisito para la obtención del grado de
Maestra en Ciencias de la Computación

Director de tesis
Dr. José Ruiz Ascencio

Cuernavaca, Morelos a 27 de mayo del 2016

OFICIO No. DCC/077/2016

Asunto: Aceptación de documento de tesis

DR. GERARDO V. GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO
PRESENTE

Por este conducto, los integrantes de Comité Tutorial de la **Ing. Alida Esmeralda Zárate Jiménez**, con número de control M14CE022, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis profesional titulado "**Detección de objetos en movimiento con sensor RGB-D para un robot móvil a velocidad constante**" y hemos encontrado que se han realizado todas las correcciones y observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

DIRECTOR DE TESIS



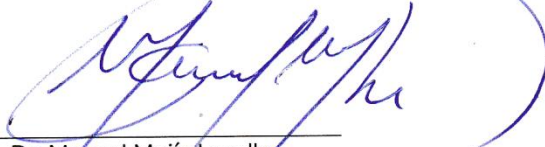
Dr. José Ruiz Ascencio
Doctor en Ciencias
5009035

REVISOR 1



Dr. Raúl Pinto Elías
Doctor en Ciencias en la Especialidad
de Ingeniería Eléctrica
3890453

REVISOR 2



Dr. Manuel Mejía Lavalle
Doctor en Ciencias
Computacionales
8342472

REVISOR 3



Dr. Marco Antonio Oliver Salazar
Doctor en Ciencias en Ingeniería
Electrónica
6526083

C.p. Lic. Guadalupe Garrido Rivera - Jefa del Departamento de Servicios Escolares.
Estudiante
Expediente

AMR/lmz

Cuernavaca, Mor., 3 de junio de 2016
OFICIO No. SAC/186/2016

Asunto: Autorización de impresión de tesis

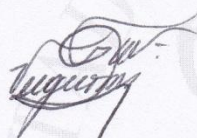
**ING. ALIDA ESMERALDA ZÁRATE JIMÉNEZ
CANDIDATA AL GRADO DE MAESTRA EN CIENCIAS
DE LA COMPUTACIÓN
PRESENTE**

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado **“Detección de objetos en movimiento con sensor RGB-D para un robot móvil a velocidad constante”**, ha informado a esta Subdirección Académica, que están de acuerdo con el trabajo presentado. Por lo anterior, se le autoriza a que proceda con la impresión definitiva de su trabajo de tesis.

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

ATENTAMENTE

“CONOCIMIENTO Y TECNOLOGÍA AL SERVICIO DE MÉXICO”



**DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO**



SEP TecNM
CENTRO NACIONAL
DE INVESTIGACIÓN
Y DESARROLLO
TECNOLÓGICO
SUBDIRECCIÓN
ACADÉMICA

C.p. Lic. Guadalupe Garrido Rivera.- Jefa del Departamento de Servicios Escolares.
Expediente

GVGR/mcr

Dedicatoria:

A mis padres, a mi hermano y a mi compañera Rita.

Gracias por su apoyo incondicional.

AGRADECIMIENTOS

Agradezco a Dios por darme fortaleza, paciencia y estar conmigo en todo momento.

Agradezco al Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET) por las facilidades y el apoyo proporcionado para la culminación de esta tesis.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico que me brindó durante mis estudios de maestría.

Agradezco al Dr. José Ruiz Ascencio asesor de este trabajo, por sus valiosos consejos y asesoría, los cuales contribuyeron a la terminación satisfactoria de este trabajo.

Agradezco a los revisores de mi trabajo: Dr. Raúl Pinto Elías, Dr. Manuel Mejía Lavalle y al Dr. Marco Oliver Salazar, por sus comentarios y observaciones que me ayudaron a culminar este proyecto. Gracias por el trato amable y la confianza que siempre me brindaron.

Finalmente a mis compañeros de la generación 2014 – 2016 de Ciencias de la Computación por los gratos momentos compartidos.

RESUMEN

En esta tesis se presenta un método para detectar puntos móviles que pertenecen a objetos móviles, con una cámara en movimiento. Se usa una cámara RGB-D para recopilar datos 3D. Se construye un modelo de movimiento aparente 3D de la escena, producido por el movimiento propio de la cámara.

Cuando en el entorno exista algo que no es estático el modelo de movimiento aparente 3D revelará una dinámica distinta, esto quiere decir que es un objeto móvil.

El método que se plantea en este trabajo resuelve el problema de la cámara móvil a la hora de detectar objetos móviles.

Se usó un algoritmo de puntos destacados ORB y un algoritmo de correspondencia *BRIEF* con distancia de *Hamming* para obtener los datos necesarios y entrenar el modelo.

La fase de aprendizaje de este método es realizada por medio de un algoritmo de lógica difusa basado en gradiente descendente, el cual produce reglas que reflejan el modelo de movimiento obtenido.

En la fase de reconocimiento la base de reglas del modelo 3D deduce la dinámica de movimiento de los puntos de la escena. Cuando se encuentra un punto con una dinámica de movimiento distinta a la dinámica del modelo, entonces se considera que pertenece a un objeto móvil respecto al resto de la escena.

Este método fue probado con 2 tipos de objetos móviles a diferentes distancias y con distintas direcciones de movimiento.

Los resultados que se obtuvieron de esta investigación indican que de acuerdo al contexto elegido de las pruebas, este método tiene un 88.2 % de eficacia promedio en comparación de los trabajos presentados en el estado del arte, el de mayor porcentaje de eficacia fue de 85.6% [Jin, 2011], pero en este trabajo presentado en el estado del arte el movimiento de la cámara lo realiza después de haber detectado al objeto móvil.

ABSTRACT

This thesis presents a method to detect moving points belonging to moving objects, with a moving camera. An RGB-D camera is used to collect 3D data. A 3D model of apparent motion of the scene, produced by the egomotion of the camera, is constructed.

When in the environment there is something that is not static the model 3D reveal apparent motion a different dynamic, this means that it is a moving object.

The method proposed in this work solves the problem of mobile camera when detect moving objects.

The ORB interest point algorithm and the BRIEF matching algorithm with Hamming distance were used to get the data necessary to train the model.

The learning phase of this method is performed by a fuzzy logic algorithm based on gradient descent, which produces rules that reflect the model of movement obtained.

In the recognition phase, the rule base of the 3D model deduces the dynamics of movement of the points of the scene. When it finds a point whose motion follows a different dynamics, it is considered to belong to a moving object, respect to rest to the scene.

This method was tested with 2 kinds of mobile objects at different distances and with different directions of movement.

The results obtained in this investigation indicate that according to the chosen test context, this method has 88.2% of effectiveness average, in comparison with the papers presented at the state of art, the highest percentage of efficacy was 85.6% [Jin, 2011], but in this work presented in the state of art the camera movement is performed after detecting the moving object.

CONTENIDO

| | |
|--|-----------|
| ÍNDICE DE TABLAS..... | III |
| ÍNDICE DE FIGURAS..... | IV |
| 1 INTRODUCCIÓN..... | 1 |
| 1.1 Antecedentes..... | 1 |
| 1.2 Justificación..... | 2 |
| 1.3 Descripción del problema..... | 3 |
| 1.4 Objetivos..... | 4 |
| 1.4.1 Objetivo General..... | 4 |
| 1.4.2 Objetivos específicos..... | 4 |
| 1.4.3 Objetivos nuevos..... | 5 |
| 1.5 Organización de la tesis..... | 5 |
| 2 TRABAJOS RELACIONADOS..... | 6 |
| 2.1 Antecedentes..... | 6 |
| 2.2 Estado del arte..... | 8 |
| 2.3 Conclusión del estado del arte..... | 14 |
| 2.4 Discusión..... | 15 |
| 3 DISEÑO E IMPLEMENTACIÓN..... | 17 |
| 3.1 Diseño del sistema propuesto..... | 17 |
| 3.2 Implementación del sistema..... | 18 |
| 3.2.1 Aprendizaje de reglas de inferencia difusas basado en el método del gradiente descendente..... | 18 |
| 3.2.2 Fase de entrenamiento: Modelado de la dinámica de movimiento 3D hacia adelante..... | 22 |
| 3.2.3 Segunda fase: Fase de reconocimiento para detectar objetos móviles..... | 32 |
| 3.3 Discusión..... | 37 |

| | | |
|----------|--|-----------|
| 4 | EXPERIMENTACIÓN, RESULTADOS Y ANÁLISIS | 38 |
| 4.1 | Análisis de KINECT | 38 |
| 4.1.1 | Prueba preliminar con algoritmo Nomura | 41 |
| 4.1.2 | Prueba de algoritmos detectores y descriptores de puntos destacados..... | 45 |
| 4.2 | Especificaciones técnicas | 49 |
| 4.3 | Plan de pruebas | 50 |
| 4.4 | Resultados | 52 |
| 4.5 | Análisis de Resultados | 54 |
| 4.6 | Discusión | 55 |
| 5 | CONCLUSIONES | 56 |
| 5.1 | Objetivos alcanzados..... | 56 |
| 5.2 | Aportaciones..... | 57 |
| 5.3 | Productos..... | 57 |
| 5.4 | Conclusiones Finales..... | 58 |
| 5.5 | Trabajos futuros..... | 59 |
| | REFERENCIAS..... | 60 |
| | ANEXO A - PRUEBAS | 64 |
| | ANEXO B – ALGORITMOS DETECTORES, DESCRIPTORES Y DE CORRESPONDENCIA..... | 83 |

ÍNDICE DE TABLAS

| | |
|--|----|
| Tabla 2.1 Técnicas de segmentación para cámara móvil implementadas en este trabajo | 7 |
| Tabla 2.2 Detección de múltiples Humanos [Khan, 2012]..... | 12 |
| Tabla 2.3 Detección y evasión de obstáculos [Khan, 2012]..... | 12 |
| Tabla 2.4 Resumen de los trabajos relacionados | 15 |
| Tabla 3.1 Descripción de los 3 sistemas difusos | 26 |
| Tabla 3.2 Error inicial, Error mínimo y épocas en las que convergió cada sistema | 27 |
| Tabla 3.3 Parámetros a sintonizar cuando las entradas tienen función de pertenencia 2 | 27 |
| Tabla 3.4 Valores de los centros y bases de los triángulos sintonizados, de los 3 sistemas difusos | 29 |
| Tabla 3.5 Valores finales de los pesos de las reglas sintonizadas..... | 30 |
| Tabla 3.6 Descripción de métodos de C++ de la fase de reconocimiento | 37 |
| Tabla 4.1 Características de KINECT Xbox 360 [Han, 2013] | 39 |
| Tabla 4.2 Entrada a sintonizar | 43 |
| Tabla 4.3 Valores de los centros y bases de los triángulos sintonizados, de los 2 sistemas difusos | 44 |
| Tabla 4.4 Lista de pruebas | 50 |
| Tabla 4.5 Promedio de eficacia en porcentaje de las 12 pruebas..... | 53 |
| Tabla 5.1 Objetivos alcanzados | 56 |

ÍNDICE DE FIGURAS

| | |
|---|----|
| 1.1 Plano de movimiento de una nave aérea..... | 3 |
| Figura 1.2 Diagrama del sistema para detección de objetos en movimiento con cámara móvil..... | 4 |
| Figura 2.1 Correspondencia de puntos a) y Flujo óptico b)..... | 7 |
| Figura 2.2 Proceso de segmentación en tiempo real [Cucchiara, 2004]..... | 8 |
| Figura 2.3 Diferencia de SGM y SMG modo dual [Moo, 2013]..... | 9 |
| Figura 2.4 Proceso para la detección de objetos en movimiento [Jin, 2011]..... | 10 |
| Figura 2.5 Aplicación del método [Jin, 2011]..... | 10 |
| Figura 2.6 The Xtion Pro Live [ASUS, 2014]..... | 11 |
| Figura 2.7 Detección de obstáculos [Khan, 2012]..... | 13 |
| Figura 2.8 Cámara con un grado de libertad..... | 14 |
| Figura 2.9 Flujo óptico generado para un objeto estático (a) y un objeto móvil (b)..... | 14 |
| Figura 3.1 Fase de entrenamiento del sistema (Modelado de la dinámica de movimiento 3D hacia adelante)..... | 17 |
| Figura 3.2 Fase de reconocimiento para detectar objetos móviles..... | 18 |
| Figura 3.3 Función de pertenencia usada por este método de aprendizaje..... | 19 |
| Figura 3.4 Efecto de movimiento trasero en Robotino..... | 23 |
| Figura 3.5 Detección de correspondencias falsas positivas con el algoritmo de correspondencias propio..... | 23 |
| Figura 3.6 Carro sobre un riel con KINECT montado..... | 24 |
| Figura 3.7 Resultado del proceso de correspondencias entre la imagen T y $T+1$, donde las líneas verdes representan las correspondencias correctas..... | 24 |
| Figura 3.8 Correspondencias a distintas distancias..... | 25 |
| Figura 3.9 Entrenamiento en Matlab del sistema difuso 1..... | 27 |
| Figura 3.10 Funciones de pertenencia del sistema 1..... | 28 |
| Figura 3.11 Funciones de pertenencia del sistema 2..... | 28 |
| Figura 3.12 Funciones de pertenencia del sistema 3..... | 29 |
| Figura 3.13 Reglas del sistema difuso 1 ($X+$)..... | 31 |
| Figura 3.14 Reglas del sistema difuso 2 ($Y+$)..... | 31 |
| Figura 3.15 Reglas del sistema difuso 3 ($Z+$)..... | 31 |
| Figura 3.16 Gráfica de comparación del error de predicción del modelo, la línea verde muestra el error sin multiplicar por un valor de escala y la línea roja muestra el error multiplicado por un valor de escala..... | 33 |
| Figura 3.17 Fórmula para obtener el valor de diferencia entre el valor obtenido por el proceso de correspondencia y el valor obtenido con el modelo de movimiento aparente en 3D hacia delante de una escena..... | 33 |
| Figura 3.18 Etiqueta al punto móvil con un círculo rojo..... | 34 |
| Figura 3.19 Esquema del sistema de la fase de reconocimiento para detectar objetos móviles..... | 35 |
| Figura 3.20 Diagrama de <i>Warnier Orr</i> de la segunda fase del sistema..... | 36 |
| Figura 4.1 Dispositivo KINECT versión Xbox 360..... | 38 |
| Figura 4.2 Imagen RGB e imagen de profundidad obtenidas con el dispositivo KINECT..... | 38 |
| Figura 4.3 Imagen de profundidad con el borde vacío..... | 39 |
| Figura 4.4 Rango de distancia que detecta KINECT..... | 40 |
| Figura 4.5 Obtención de distancia de tres puntos distribuidos de forma vertical..... | 40 |
| Figura 4.6 Cuadrante perteneciente..... | 42 |
| Figura 4.7 Movimiento de un pixel aplicando el método <i>Global motion models</i> 2D hacia delante..... | 42 |
| Figura 4.8 Modelo de movimiento 2D hacia delante..... | 43 |
| Figura 4.9 Resultado del Método Nomura..... | 44 |
| Figura 4.10 Imágenes para probar los algoritmos detectores de puntos destacados..... | 45 |
| Figura 4.11 Graficas de los resultados obtenidos al aplicar los algoritmos detectores a la imagen Pies..... | |

| | |
|--|----|
| | 46 |
| Figura 4.12 Graficas de los resultados obtenidos al aplicar los algoritmos detectores a la imagen Silla | 46 |
| | 46 |
| Figura 4.13 Graficas de los resultados obtenidos al aplicar los algoritmos detectores a imagen Objeto pequeño..... | 47 |
| Figura 4.14 Ambiente creado para obtención de correspondencias | 47 |
| Figura 4.15 Robotino de FESTO, con KINECT..... | 48 |
| Figura 4.16 Correspondencias con detector SURF y ORB, y descriptor y algoritmo de correspondencia propio..... | 48 |
| Figura 4.17 Comparación de Algoritmo detector SURF y ORB, dentro de un proceso de correspondencias con algoritmo descriptor propio y de correspondencia propio | 49 |
| Figura 4.18 <i>Frames</i> de la prueba 2, donde el objeto móvil se mueve de izquierda a derecha, con una distancia inicial a la cámara de 2.5 metros | 51 |
| Figura 4.19 Gráfica del porcentaje de eficacia del sistema en las 12 pruebas | 53 |

CAPÍTULO 1

1 INTRODUCCIÓN

En el presente capítulo se proporciona una perspectiva general de la detección de objetos móviles con cámara móvil, además se identifica el problema que dio origen al presente trabajo de investigación. Se mencionan los objetivos del trabajo y se describe de manera muy general la metodología a seguir para realizar la experimentación.

1.1 Antecedentes

La detección de objetos en movimiento mediante una cámara móvil es sin duda un tema relevante en el campo de la visión artificial; tiene variedad de aplicaciones tales como video vigilancia, visión para robots, conteo de mercancía o personas, vehículos autónomos y así sucesivamente [Albiol, 2000] [Low, 2010]. Los métodos que se han propuesto y que han demostrado tener éxito para la detección de objetos móviles con cámara estática no se pueden aplicar cuando la cámara presenta movimiento [Stauffer, 1999] [Yachida, 1981], ya que no contemplan el movimiento aparente del fondo. La dificultad de esta tarea está asociada a la presencia de movimiento en las dos partes de la escena, tanto en el primer plano como en el fondo.

La mayoría de métodos que se encuentran en la literatura para detectar objetos móviles con cámara móvil se basan en flujo óptico, los cuales realizan la detección de objetos móviles basados en el flujo óptico producido por el movimiento de la cámara, ya que el flujo óptico es generado tanto en los objetos estáticos como en los móviles. Teniendo un conocimiento a priori de la magnitud y dirección de los vectores de objetos estáticos se puede deducir cuáles vectores pertenecen a objetos móviles.

Estas técnicas detectan objetos móviles pero no proporcionan la distancia a la cual se encuentran ya que están basados en modelos de movimiento de 2 dimensiones [Shibata, 2008] [Mora, 2009].

Basado en esta problemática se refleja que la distancia es un parámetro esencial para detección de objetos móviles a diferentes distancias, ya que modelos mencionados en esta sección no toman en cuenta este parámetro.

El enfoque de este trabajo consiste en tomar en cuenta el parámetro de distancia para poder detectar objetos móviles a distancias distintas. Cabe mencionar que el movimiento aparente de un plano cercano, no es igual al movimiento aparente de un plano lejano, como coloquialmente se dice, "lo que está más cerca se mueve más que lo que se encuentra más lejos". Para poder resolver este problema se propone un método para detección de objetos móviles con cámara móvil.

El método que se propone consiste en realizar un modelo que refleje la dinámica de movimiento aparente del fondo de una escena en 3 dimensiones, con movimiento rectilíneo hacia el frente, con una velocidad constante, por medio de puntos característicos [Rosten, 2006] y un algoritmo de correspondencia [Ruble, 2011].

Para realizar el modelo se utiliza un algoritmo basado en lógica difusa [Nomura, 1992], el cual fue entrenado en su etapa de aprendizaje con datos de recorridos de la cámara RGB-D sin objetos móviles. El algoritmo en su etapa de reconocimiento deduce la dinámica de movimiento aparente hacia delante con una velocidad constante; las correspondencias de puntos destacados que revelen una dinámica de movimiento distinta son consideradas correspondencias pertenecientes a un objeto móvil.

El algoritmo de aprendizaje puede aumentar las variables de entrada, para incluir la velocidad y/o la dirección de su movimiento para posteriormente poder mejorar este proyecto.

1.2 Justificación

En CENIDET se tiene una línea de visión robótica establecida desde hace diez años, donde interesa dar a los robots móviles, a través de la visión artificial, cada vez más autonomía y capacidad de interacción con su medio. Para ello, es necesario que el robot pueda discernir con precisión los objetos que se mueven de los objetos estáticos.

Derivados de esta capacidad se posibilitan mejoras en video vigilancia, ya que entonces podrá ser móvil, así como las capacidades de percepción que requiere un auto que se conduzca por computadora; también en sistemas de visión para personas invidentes, entre otras muchas prestaciones que buscamos con la visión artificial [Khan, 2012].

En CENIDET el único trabajo que ha considerado cámara móvil lo ha hecho de forma heurística [Vidal, 2014], y no mediante un modelo de dinámica de la escena.

Este trabajo propone el uso de un sensor RGB-D [Kinect, 2016] para simplificar esta primera prueba de concepto, pero el caso RGB-D tiene interés por sí mismo ya que este sensor proporciona información de profundidad de una escena, siendo así importante el parámetro de la distancia para el modelado de la dinámica de movimiento hacia delante en 3D.

1.3 Descripción del problema

El problema que se pretende solucionar con este trabajo de tesis se enfoca principalmente en la tarea de detección de objetos en movimiento con una cámara móvil. Cabe mencionar que para la detección del objeto en movimiento (primer plano), se tiene que solucionar primero el problema del fondo dinámico.

El fondo que se encuentra más lejos tiene un movimiento aparente en menor proporción, mientras que el fondo que se encuentran más cerca, tienen un movimiento aparente en mayor proporción.

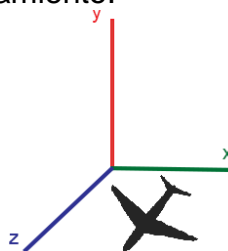
La profundidad a la que se encuentra un punto puede ayudar a decidir si tiene movimiento propio, o sólo refleja el movimiento de la cámara. Esto requiere un análisis de los patrones de movimiento aparente en 3D que se generan con el movimiento de la cámara, para obtener un modelo de movimiento aparente en 3 Dimensiones [Irani, 1999].

En esta tesis se aborda el flujo óptico de tres dimensiones y un grado de libertad, mediante un modelo difuso sintonizable. Esta técnica tiene la ventaja de que puede adaptarse en principio a más grados de libertad, con tal que se cuente con datos suficientes para realizar un buen entrenamiento.

Los problemas que se presentan al querer detectar la parte móvil de una escena con una cámara en movimiento son que el fondo y el primer plano son móviles, por lo cual las dos partes de la escena tienen movimiento aparente.

Para un robot el tener la información de profundidad es algo que puede ayudarle a detectar obstáculos a su paso. Un robot móvil o cámara móvil en tres dimensiones puede moverse hasta con seis grados de libertad: traslación a lo largo de tres ejes perpendiculares X , Y y Z , así como giros en torno a cada uno de estos tres ejes; si se piensa en una nave marina o aérea, cuyo movimiento principal es paralelo a la superficie de la tierra, entonces se asigna el eje Z a dicha dirección de movimiento, y en proximidad de la tierra, la fuerza de gravedad define la dirección del eje Y , quedando X como el eje horizontal perpendicular al movimiento como se muestra en la Figura 1.1.

Para obtener información del eje Z se pueden usar técnicas de visión binocular, pero estas suelen requerir de más procesamiento.



1.1 Plano de movimiento de una nave aérea

1.4 Objetivos

A continuación se presentan los objetivos que se plantearon para este proyecto.

1.4.1 Objetivo General

Estudiar las técnicas de segmentación para detección de objetos en movimiento con cámara en movimiento e implementar un sistema de visión artificial que permita realizar la detección de movimiento por medio de una cámara RGB-D en movimiento.

1.4.2 Objetivos específicos

- Estudiar, implementar y evaluar las técnicas de segmentación para la detección de objetos en movimiento con cámara RGB-D en movimiento.
- Estudiar las técnicas para seguimiento de objetos en movimiento.
- Realizar un sistema de detección de objetos en movimiento de un robot móvil, con una cámara RGB-D, como se muestra en la Figura. 1.2.



Figura 1.2 Diagrama del sistema para detección de objetos en movimiento con cámara móvil

1.4.3 Objetivos nuevos

Los objetivos contemplados para este proyecto y mencionados en los puntos 1.4.1 y 1.4.2, fueron modificados en el transcurso del desarrollo de esta tesis, ya que el enfoque fue modificado con el desarrollo del tema, a continuación se enlistan los objetivos específicos finales.

- Analizar e implementar algoritmos detectores y descriptores de puntos destacados para elegir el mejor y aplicarlo a este proyecto.
- Analizar el funcionamiento del dispositivo KINECT.
- Elaborar un modelo de movimiento 3D con dirección hacia delante.
- Encontrar un método de aprendizaje para aprender el modelo de movimiento.
- Realizar un sistema de detección de objetos móviles con una cámara móvil RGB-D con movimiento rectilínea hacia delante, en tiempo real.

1.5 Organización de la tesis

El presente documento de tesis está organizado de la siguiente manera:

- En el capítulo 2, se proporciona información sobre los trabajos relacionados y el estado del arte.
- En el capítulo 3, se detalla el análisis, diseño e implementación de este proyecto.
- En el capítulo 4, se presenta la experimentación y resultados.
- En el capítulo 5, se proporciona la conclusión de la investigación y el trabajo futuro a desarrollar.

CAPÍTULO 2

2 TRABAJOS RELACIONADOS

En este capítulo se encuentran los trabajos relacionados con el tema de investigación que se han realizado tanto dentro como fuera del Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET).

2.1 Antecedentes

En esta sección se describe un trabajo previo el cual fue realizado en el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET) y está relacionado con este tema de tesis.

Evaluación de técnicas para la detección de las partes estáticas y las partes móviles de una escena en secuencia de video [Vidal, 2014]

Este trabajo aborda la detección de las partes estáticas y las partes móviles de una escena en secuencias de video. Realiza un estudio, implementación y evaluación de las principales técnicas usadas para diferenciar las partes estáticas de las móviles de una escena en secuencias de video, se estudió la resolución de los problemas que se tienen, al querer detectar las partes que conforma un video, son dos básicamente, uno para cámara estática y otro más complicado que es para cámara móvil.

Con una cámara sin movimiento la parte estática de una escena es el fondo, mientras que la parte móvil (objetos en movimiento) de la escena es el primer plano.

Con cámara móvil el fondo y el primer plano presentan movimiento y eso lo hace una tarea más complicada que con cámara estática ya que influyen los factores de la cámara estática que son: sombras, luces, ruido y muchas condiciones cambiantes del entorno no controlado y se le suma el movimiento de la cámara.

En este trabajo se implementaron tres técnicas, para la detección de las partes estáticas y las partes móviles de una escena en una secuencia de video tomado con una cámara móvil, una de ellas fue la técnica de la correspondencia de puntos, otra fue flujo óptico y la última fue una combinación de ambas.

En la Tabla 2.1. Se presentan las técnicas que se implementaron en este trabajo para la detección de las partes estáticas y las partes móviles de una escena en una secuencia de video.

En la Figura 2.1 se muestra dos técnicas utilizadas en este trabajo: a) correspondencia de puntos y b) Flujo óptico.

Tabla 2.1 Técnicas de segmentación para cámara móvil implementadas en este trabajo

| Pasos | Correspondencia de puntos | Flujo óptico | Combinación de Correspondencia de puntos y Flujo óptico |
|-------|---|---|--|
| 1 | Detectar puntos con FAST | Detectar puntos con FAST | Obtener la imagen de la técnica de correspondencia de puntos Obtener la imagen de la técnica flujo óptico Combinarlas. |
| 2 | Aplicar descriptor SIFT | Calcular flujo óptico LK | |
| 3 | Aplicar algoritmo de correspondencia <i>Brute Force</i> | Calcular un grupo de ángulos que más se repitan en la escena | |
| 4 | Dibujar flujo de líneas entre las correspondencias de la imagen 1 con la imagen 2 | Implementar un umbral para dividir el fondo del primer plano | |
| 5 | Calcular un grupo de ángulos que más se repitan en la escena | Hacer una imagen binaria con los puntos que pasaron o no pasaron el umbral. | |
| 6 | Implementar un umbral para dividir el fondo del primer plano | | |
| 7 | Hacer una imagen binaria con los puntos que pasaron o no pasaron el umbral. | | |

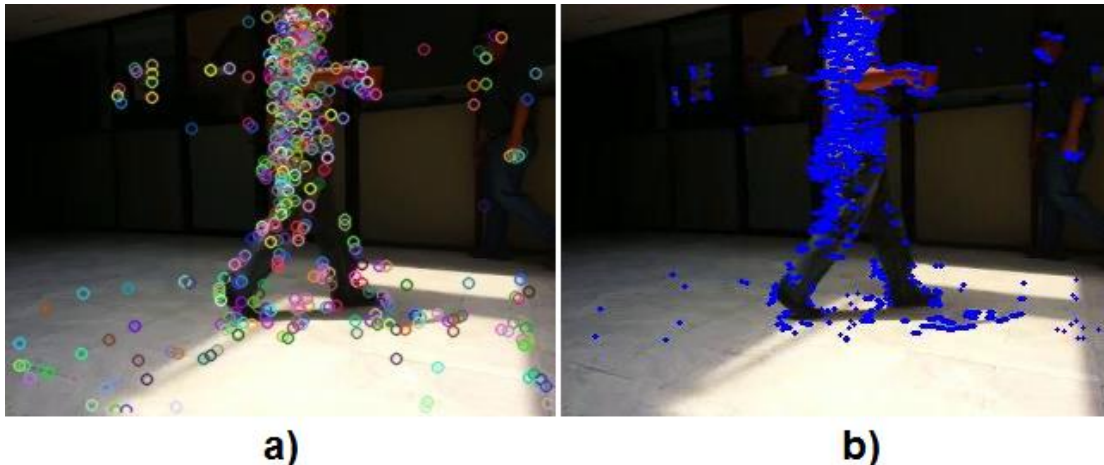


Figura 2.1 Correspondencia de puntos a) y Flujo óptico b)

Cabe señalar que la técnica combinada que se implementó en este trabajo no tuvo éxito ya que las dos técnicas no pudieron fusionarse. El trabajo que se propone, es un

seguimiento del tema de tesis realizado en CENIDET, sólo que en este trabajo se usará una cámara RGB-D, a diferencia del trabajo anterior donde se trabajó con una cámara RGB.

2.2 Estado del arte

A continuación se mencionan los trabajos encontrados que presentan relación con la presente investigación.

Real-time motion segmentation from moving cameras [Cucchiara, 2004]

Este método fue realizado en el 2004 por la universidad Modena y Reggio de Italia, en el cual se realiza la detección de objetos en movimiento con cámara móvil RGB, la cámara se encuentra en movimiento y realiza la detección del objeto, la segmentación se basa en el color y el crecimiento de regiones, mediante la combinación de técnicas como *Matchin Region Particions* para la estimación de movimiento basado en regiones y *Markov Random Fields* para la estimación de movimiento de las regiones fusionadas, con ésta combinación de técnicas se pudo optimizar la segmentación para que ésta fuese rápida, en la Figura 2.2 se presenta cómo se realizó el proceso.

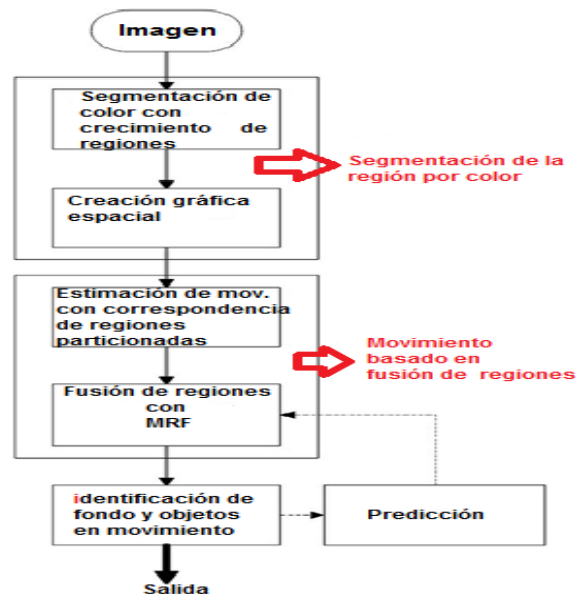


Figura 2.2 Proceso de segmentación en tiempo real [Cucchiara, 2004]

Conclusión: En este trabajo se utiliza la información de color para la segmentación del movimiento, pero no se tiene conocimiento de a qué distancia se encuentra el objeto móvil.

Detection of Moving Objects with Non-Stationary Cameras in 5.8ms: Bringing Motion Detection to your Mobile Device [Moo, 2013]

En la universidad nacional de Seul se realizó un método para la detección de movimiento con una cámara de celular, para modelar el fondo y los objetos en movimiento se usó *Single Gaussian Model* (SGM), a éste método se le realizó una mejora para que fuese de modo *Dual-Mode* para compensar el movimiento de la cámara y mezclar los modelos vecino, también fue en tiempo real, en la Figura 2.3 se puede apreciar el método SGM y SGM Dual. Se usó SGM porque evita que el modelo de fondo sea contaminado por pixeles del primer plano donde la compensación del movimiento es demostrada de una manera simple con el algoritmo de Lucas- Kanade [Tomasi, 1991] [Lucas, 1981].

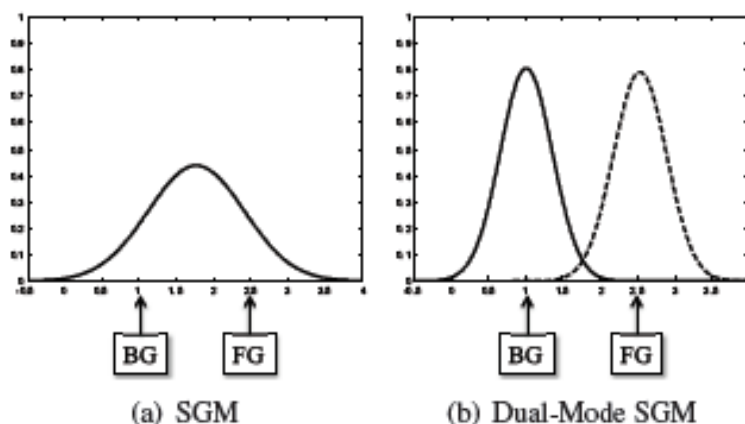


Figura 2.3 Diferencia de SGM y SMG modo dual [Moo, 2013]

Conclusión: En este trabajo se realiza la detección de primer plano, y para eso se usó el modelo SGM modo dual, cabe mencionar que este trabajo tampoco proporciona información de distancia del objeto móvil detectado.

Moving Object Detection and Tracking from Moving Camera [Jin, 2011]

En este trabajo se presenta un método para detección de objetos en movimiento y seguimiento del movimiento con una cámara RGB pero estacionaria, se realiza aplicando el algoritmo KLT (Kanade-Lucas-Tomasi) [Lucas, 1981] para extraer las

características de la imagen y hacer un seguimiento de ellas entre dos *frames*, posteriormente se obtiene una homografía H por medio del algoritmo RANSAC, de la homografía H se comparan todos los píxeles entre dos *frames* y se obtienen los píxeles residuales que no son consistentes en la intensidad de píxeles, porque esos píxeles residuales sólo aparecen en el borde de los objetos con movimiento y es necesario refinar los píxeles. Se aplican procesos morfológicos y da como resultado detección de objetos en movimiento con un 85.6 % de eficacia. En la Figura 2.4 y 2.5 se puede observar el proceso de este método.

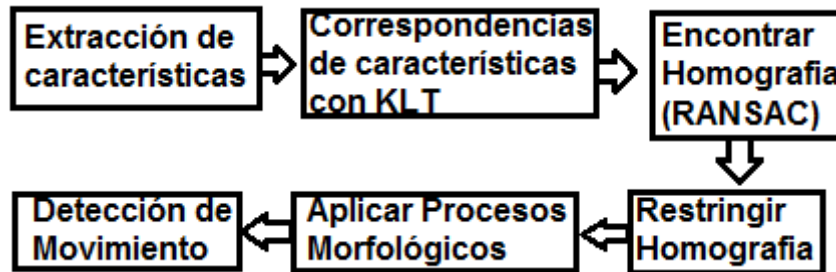


Figura 2.4 Proceso para la detección de objetos en movimiento [Jin, 2011]

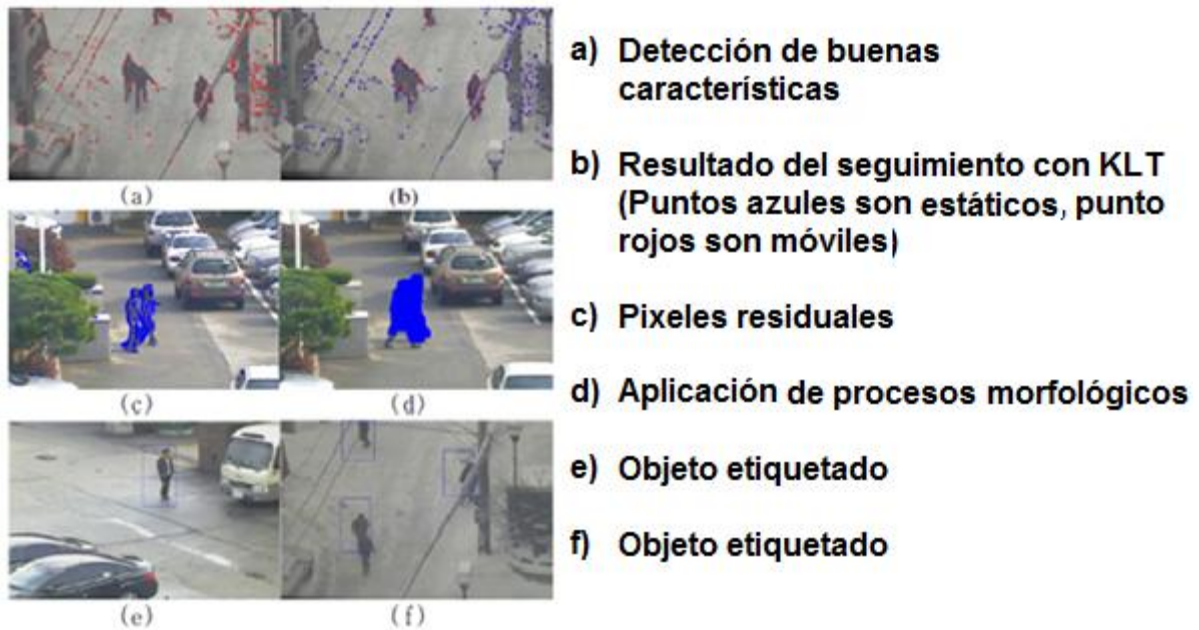


Figura 2.5 Aplicación del método [Jin, 2011]

Conclusión: Este método detecta objetos móviles cuando la cámara está estática, la cámara se mueve siguiendo al objeto después de haberlo detectado, por lo tanto no podría aplicarse este método para la detección de objetos en movimiento con cámara en movimiento.

KinDectect: Kinect Detecting Objects [Khan, 2012]

En este trabajo se presenta un sistema para personas invidentes el cual puede ser usado a la hora de realizar sus actividades en exteriores sin necesidad de un perro guía. Este proyecto se divide en dos partes dentro de una trayectoria, la primera: detección de múltiples humanos, y la segunda: detección y evasión de objetos en movimiento. Esto se realizó con *The Xtion Pro Live*, como se puede observar en la Figura 2.6 es una cámara RGB-D de la marca *ASUS* [ASUS, 2016], se usó el *framework OpenNI* [OpenNI, 2016] ya que cuenta con la función de detección y seguimiento de personas, también se usó la librería de *OpenCV* [OpenCV, 2016]. Cuando el usuario pida al sistema detectar personas, éste tendrá que detenerse para que el sistema realice la detección de personas y sea notificado; cuando el usuario se encuentre caminando el sistema entra en modo detección y evasión de objetos.



Figura 2.6 The Xtion Pro Live [ASUS, 2014]

A continuación en la Tabla 2.2 y Tabla 2.3 se presentan las dos fases de este proyecto.

Tabla 2.2 Detección de múltiples Humanos [Khan, 2012]

| Paso | Proceso |
|------|---|
| 1 | Generar y mostrar la información de profundidad y RGB |
| 2 | Seguimiento de personas y detección de distancia y posición en el espacio 3D |
| 3 | Etiquetar y colorear las personas detectadas según su profundidad para una depuración |
| 4 | Hacer que los datos (profundidad y RGB) dados por OpenNI sean accesible y compatible con OpenCV (RGB a la conversión BGR) |

Tabla 2.3 Detección y evasión de obstáculos [Khan, 2012]

| Paso | Proceso |
|------|---|
| 1 | Dividir el rango original de la imagen que es de 640 (Horizontal) x 480 (Vertical) en bloques de 32(Horizontal) x 40 (Vertical). El total de bloques sería 20 x 12 |
| 2 | De cada bloque de 20 x 12 se calcula un promedio |
| 3 | Los grupos de 20 x 12 bloques se agrupan en regiones quedando 5 x 3 regiones |
| 4 | Se contemplan las direcciones de la imagen dividida en 5 x 3 regiones (Figura 2.7), siendo las direcciones siguientes: superior, inferior, medio, extremo izquierda, izquierda, medio, derecha, extrema derecha |
| 5 | Se clasifican las 16 profundidades medias de cada región, tomando los 10 valores intermedios de cada región. Se calcula la media de los 10 valores y se le asigna a la variable Z |
| 6 | Se asigna Z_m = valor de distancia mínima (umbral) Se usa la métrica $M = Z/Z_m$, para detectar en qué región de las direcciones hay probabilidad de que se encuentre un obstáculo |

En la Figura 2.7 se puede apreciar la fase para detección y evasión de obstáculos.



Figura 2.7 Detección de obstáculos [Khan, 2012]

Conclusión: En este trabajo se usó una cámara RGB-D. Lleva a cabo detección de personas en movimiento cuando la cámara se encuentra en estado estático y detección de la proximidad de objetos cuando la cámara se encuentra en movimiento. Este trabajo en relación con el que se desarrolló en esta tesis tiene en común el uso una cámara RGB-D.

En este trabajo se obtiene distancia y detecta la proximidad de un posible objeto cuando la cámara está en modo estático, a diferencia de esta tesis que trata de detección de objetos móviles pero con cámara móvil.

Moving Object Detection for Active Camera based on Optical Flow Distortion [Shibata, 2008]

En este trabajo se realiza la detección de objetos móviles basados en la distorsión del flujo óptico producido por el movimiento de la cámara, la cual está montada sobre un riel y tiene un movimiento horizontal de izquierda a derecha como se muestra en la Figura 2.8. Los vectores producidos por el flujo óptico son generados tanto en los objetos estáticos como en los móviles. Si se tiene un conocimiento *a priori* de la magnitud y dirección de los vectores de objetos estáticos, se puede deducir cuáles vectores pertenecen a objetos móviles. En la Figura 2.9 se muestra el flujo óptico producido por dos tipos de objetos: (a) Objeto estático, (b) Objeto móvil. Cuando un objeto genera un flujo óptico no acorde con el flujo óptico conocido *a priori* de los objetos estáticos, puede deducirse que es un objeto móvil.



Figura 2.8 Cámara con un grado de libertad



Figura 2.9 Flujo óptico generado para un objeto estático (a) y un objeto móvil (b)

Conclusiones: Este proyecto es uno de los más apegados al enfoque que se decidió dar en este trabajo de tesis; como se puede observar este trabajo no tiene información de distancia por lo cual su modelo de movimiento fue 2D.

2.3 Conclusión del estado del arte

En los trabajos mencionados en la sección 2.2 se muestran investigaciones acerca del tema de la detección de objetos móviles con cámara móvil. Se ha intentado solucionar este problema con distintas técnicas, pero algunos trabajos llegan a ser limitados por la cámara que emplean o por el método utilizado.

Este proyecto a diferencia de los mencionados en el estado del arte, tiene una filosofía basada en modelar la dinámica del entorno estático, debido al movimiento propio de KINECT. Cuando en el entorno existe un objeto que no es estático, ese objeto revelará una dinámica distinta y por lo tanto es rotulado como objeto móvil.

2.4 Discusión

En el estado del arte se detectaron trabajos que llevan a cabo la detección de objetos móviles, pero no todos realizan la detección de la misma forma. En este trabajo de tesis se presenta un método distinto a los del estado del arte, ya que se creó a partir de un modelo de movimiento 3D y un método de aprendizaje. En la Tabla 2.4 se muestra un resumen de los trabajos del estado del arte.

Tabla 2.4 Resumen de los trabajos relacionados

| Trabajo - Año | Técnicas | Logros | Diferencias | Observación |
|---|---|--|--|--|
| Evaluación de técnicas para la detección de las partes estáticas y las partes móviles de una escena en secuencia de video [Vidal, 2014] | -Correspondencia de puntos. -Flujo óptico. -Combinación de ambas. | Separa fondo de primer plano. Flujo óptico fue la mejor técnica. Es con cámara móvil. | Usa cámara RGB. No realiza modelo de movimiento. | Esta técnica no proporciona la distancia a la que se encuentra el primer plano |
| Real-time motion segmentation from moving cameras [Cucchiara, 2004] | -Segmentación por color y crecimiento de regiones. - <i>Partición Región Matching.</i> - <i>Markov Random Fields.</i> | Realiza segmentación de movimiento para cámaras móviles y en tiempo real. | Usa cámara RGB. | Esta técnica no proporciona la distancia a la que se encuentra el movimiento segmentado |
| Detection of Moving Objects with Non-Stationary Cameras in 5.8ms: Bringing Motion Detection to your Mobile Device [Moo, 2013] | - <i>Dual Gaussian Model.</i> -Mezcla de modelos vecinos. | Detección de un objeto en movimiento y seguimiento. Mediante un celular. | La cámara no es móvil, sino que es una cámara RGB de celular. | Esta técnica no proporciona distancia al objeto. Prácticamente no es una cámara móvil. |
| Moving Object Detection and Tracking from Moving Camera [Jin, 2011] | -Extractor de características. - Algoritmo KLT. -Procesos morfológicos. | Detección y seguimiento de personas en movimiento | Usa cámara RGB. Para detección la cámara es estática. | Esta técnica no proporciona la distancia a la que se encuentra la persona. No es cámara móvil. |
| KinDectect: Kinect Detecting Objects [Khan, 2012] | -Métrica de distancia | Divide la imagen en bloques y obtiene probabilidad de colisión por bloque. Con cámara móvil. | Divide en bloques. No realiza modelo de movimiento. No detecta cosas móviles, solo posibles lugares de colisión. | Aunque en este trabajo usan cámara RGB-D no detectan objetos móviles. |

| Trabajo - Año | Técnicas | Logros | Diferencias | Observación |
|---|-----------------|---|--------------------|---|
| <i>Moving Object Detection for Active Camera based on Optical Flow Distortion</i> [Shibata, 2008] | - Flujo óptico | Detecta objetos móviles. Usa cámara móvil. | Usa cámara RGB | No obtiene la distancia del objeto móvil a la cámara. |

En la Tabla 2.4 se muestran los diferentes trabajos que abordan el tema de la detección de objetos móviles, se puede ver que se usan diferentes técnicas, pero la mayor parte de estos trabajos usan cámaras RGB las cuales no proporcionan información de la distancia entre el objeto móvil y la cámara, esta es la principal diferencia en entre estos trabajos y el trabajo que se desarrolló en esta tesis ya que en este trabajo sí se contempla la distancia entre la cámara y el objeto móvil. Además la distancia que proporciona la cámara RGB-D fue importante a la hora de realizar el modelo de movimiento 3D.

CAPÍTULO 3

3 DISEÑO E IMPLEMENTACIÓN

En este capítulo se presenta el diseño del sistema propuesto y su implementación según el proceso en el que se llevó a cabo el sistema, realizando una inmersión en las distintas etapas.

3.1 Diseño del sistema propuesto

El diseño del modelo de este sistema se muestra en las Figuras 3.1 y 3.2, en la Figura 3.1 se muestra la primera fase que se lleva a cabo, esta fase es la encargada de realizar el modelado de la dinámica de movimiento 3D hacia adelante, realizando primeramente detección de correspondencias de puntos destacados entre dos imágenes subsecuentes, para posteriormente con los datos obtenidos entrenar el algoritmo de aprendizaje [Nomura, 1992], el cual genera una base de reglas de inferencia difusa la cual se utiliza en la fase de reconocimiento que se muestra en la Figura 3.2. En la fase de reconocimiento de igual manera que en la fase anterior, se realiza la detección de correspondencias en imágenes subsecuentes para posteriormente obtener los datos de las correspondencias, mismas que son introducidas a la base de reglas de inferencia difusas para predecir el movimiento. Si la predicción fue distinta al comportamiento real de la correspondencia, esto quiere decir que la correspondencia de tal punto pertenece a un objeto móvil. Cabe señalar que la fase de prueba se realiza en tiempo real.



Figura 3.1 Fase de entrenamiento del sistema (Modelado de la dinámica de movimiento 3D hacia adelante)

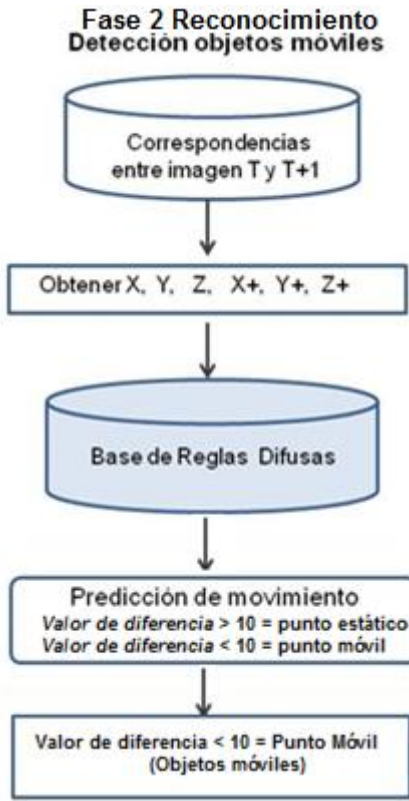


Figura 3.2 Fase de reconocimiento para detectar objetos móviles

3.2 Implementación del sistema

En este apartado se describe todo el proceso que fue realizado para la implementación del sistema final.

3.2.1 Aprendizaje de reglas de inferencia difusas basado en el método del gradiente descendente

En este método se desarrolla una metodología de sintonizado de parámetros de las acciones de las reglas de control difusas y de los conjuntos difusos (base y posición), por medio de un método de gradiente descendente [Nomura, 1992].

Se emplea como información inicial un vector de datos, que consta de dos partes: un vector de entrada (x_1, x_2, \dots, x_i) , y su respectiva salida y , los cuales se pueden describir mediante una regla difusa como se muestra en la Ecuación 1.

$$\text{if } x_i \text{ is } A_{11} \wedge \dots \wedge x_m \text{ is } A_{1m} \text{ then } y \text{ is } w_i \quad \text{Ecuación (1)}$$

Donde, x_1, \dots, x_n son las variables de entrada al sistema (escalados en el intervalo $[-1, 1]$), A_{11}, \dots, A_{1m} son los conjuntos difusos de la parte antecedente, 'y' es la salida del sistema, $i=1, 2, \dots, n$ y w_i es un escalar que representa el peso o aportación de la regla 'i' a la solución del sistema. Las funciones de pertenencia usadas por esta metodología son en forma de triángulo isósceles y se definen mediante dos parámetros: base (anchura sobre el eje de las abscisas) b_{ij} y centros (eje central del triángulo) a_i como se muestra en la Figura 3.3.

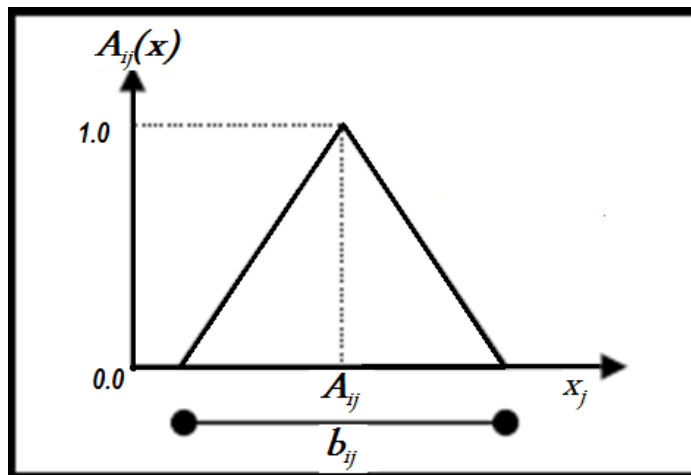


Figura 3.3 Función de pertenencia usada por este método de aprendizaje

El proceso que se realiza para la fusificación [Nomura,1992] en esta metodología es:

- i. Fusificar un valor x_j de entrada Ecuación 2.
- ii. Calcular el grado de disparo μ de cada una de las reglas mediante el producto de los grados de pertenencia $A_{ij}(x_j)$ de los antecedentes que componen a las reglas difusas Ecuación 3.

- iii. Defusificar la salida mediante un promediado de los pesos de cada regla y de su grado de disparo Ecuación 4.

$$A_{ij}(x_{ij}) = 1 - \frac{2|x_j - a_{ij}|}{b_{ij}} \quad \text{Ecuación (2)}$$

$$\mu_i = A_{i1}(x_1) \cdot A_{i2}(x_2) \cdot \dots \cdot A_{im}(x_m) \quad \text{Ecuación (3)}$$

$$y = \frac{\sum_{i=1}^n \mu_i w_i}{\sum_{i=1}^n \mu_i} \quad \text{Ecuación (4)}$$

El proceso de entrenamiento consiste en la optimización de los parámetros del sistema difuso de manera iterativa a partir de los valores calculados por el sistema y los deseados o esperado y^r como resultado de una de entrada u al sistema. Las condiciones iniciales que necesita el sistema difuso son: conjuntos difusos espaciados uniformemente, bases entre conjuntos adyacentes traslapadas entre sí y pesos iniciales de las reglas difusas en 0.5 [Nomura,1992].

El algoritmo de sintonización necesita un vector de dimensiones p definido como $Z=(z_1, z_2, \dots, z_p)$ que contiene los parámetros a sintonizar. Dicho vector en el método de descenso minimiza una función objetivo $\mathcal{E}(Z)$ hasta que esta sea menor que un valor de aceptación \mathcal{E} Ecuación 5.

$$\mathcal{E} = \frac{1}{2} (y - y_r)^2 \quad \text{Ecuación (5)}$$

La optimización de los parámetros para la adaptación del sistema difuso de un estado actual t a un estado siguiente $t+1$ es dado mediante las Ecuaciones 6, 7 y 8, las cuales son obtenidas despejando parcialmente la Ecuación 5 con respecto a a_{ij} , b_{ij} y w_i y ajustan los centros, bases y pesos respectivamente en base al error calculado.

$$a_{ij}(t+1) = a_{ij}(t) - K_a \cdot \frac{\mu_i}{\sum_{i=1}^n \mu_i} (y - y^r) \cdot (w_i(t) - y) \cdot \text{sgn}(x_j - a_{ij}(t)) \cdot \frac{2}{b_{ij}(t) \cdot A_{ij}} \quad \text{Ecuación (6)}$$

$$b_{ij}(t+1) = b_{ij}(t) - K_b \cdot \frac{\mu_i}{\sum_{i=1}^n \mu_i} (y - y^r) \cdot (w_i(t) - y) \cdot \frac{1 - A_{ij}(x_j)}{A_{ij}(x_j)} \cdot \frac{1}{b_{ij}(t)} \quad \text{Ecuación (7)}$$

$$w_i(t+1) = w_i(t) - K_w \cdot \frac{\mu_j}{\sum_{i=1}^n \mu_j} (y - y^r) \quad \text{Ecuación (8)}$$

Donde, K_a , K_b y K_w son constantes que representan la ganancia de aprendizaje o la tasa de modificación independiente para cada tipo de parámetro y $\text{sgn}(\)$ es la función signo.

El algoritmo usado para la sintonización de los parámetros del sistema consta de los siguientes pasos:

1. Contar con una base de entrenamiento.
2. Calcular la aportación de cada uno de los datos de entrenamiento (x_1, x_2, \dots, x_m) en cada w .
3. Calcular para cada w_i el promedio de las aportaciones, tal como una votación que hace más dirigido el proceso de sintonización.
4. Aplicar el incremento promedio a cada w_i .
5. Realizar los pasos 1 y 3 ahora para a_{ij} y b_{ij} .

6. Calcular el resultado de la función de costo \mathcal{E} utilizando los nuevos valores de los parámetros.
7. Si $E > \epsilon$ regresar al paso 1, en caso contrario se llegó a la solución deseada.

Se utilizó este método [Nomura, 1992] porque el sistema desarrollado en esta tesis es un sistema dinámico, y tiene su propia ley de evolución de estado, entonces se necesitaba aproximar los estados futuros y el algoritmo de Nomura es un aproximador universal, por lo cual se puede predecir el estado futuro. Nomura es equivalente a una Red Neuronal Back-propagación en términos de aproximación, pero se decidió usar el algoritmo de Nomura por que tiene la ventaja de generar reglas que son más entendibles que los pesos de la Red Neuronal. También debido a que la lógica difusa maneja grados de pertenencia que indican el grado en que un objeto puede pertenecer a una determinada clase, lo cual ayuda a predecir cuánto se mueve la escena según su respectiva distancia en un recorrido en línea recta hacia adelante.

3.2.2 Fase de entrenamiento: Modelado de la dinámica de movimiento 3D hacia adelante

En esta sección se muestra cómo se realizó el modelado 3D del movimiento hacia adelante a velocidad constante, el cual consiste primeramente en realizar un recorrido hacia adelante, obtener puntos destacados con distancia y correspondencias entre dos *Frames*, cuando se realiza la correspondencia de puntos entre dos imágenes se toman X , Y y Z de la Imagen T y X_+ , Y_+ y Z_+ de la imagen $T+1$, estos 6 datos son considerados como datos de entrada al sistema difuso para realizar el entrenamiento y obtener un modelo de movimiento 3D hacia adelante.

Se realizó primero un modelado con las siguientes especificaciones:

- Robotino en modo sigue líneas
- Robotino a 10 cm/s
- 1 Frame / 2s
- Recorrido 2 metros
- Detector ORB
- Descriptor propio
- Algoritmo de correspondencia propio
- Condición de luz controlada

En la fase de entrenamiento se obtuvieron reglas difusas con las cuales se llevó a cabo la fase de reconocimiento, para aplicar la fase de reconocimiento, se realizaron pruebas preliminares, las cuales consistían en que Robotino avanzara hacia adelante a una velocidad de 10 cm por segundo y que al mismo tiempo un objeto móvil pasara frente al robot.

Las pruebas preliminares mostraron dos problemas; la primera fue que el modelado que se realizó con Robotino y KINECT mostraba error de predicción, ya que cuando Robotino avanza tiene un ligero movimiento trasero y esto causa una pérdida de dirección fija hacia adelante, esto provocó que el modelo no fuese apropiado para avanzar hacia adelante, este error provocaba detección de falsos puntos móviles como muestra la Figura 3.4. El segundo problema que se detectó fue que el algoritmo de correspondencia propio obtenía frecuentemente falsas correspondencias como se muestra en la Figura 3.5.

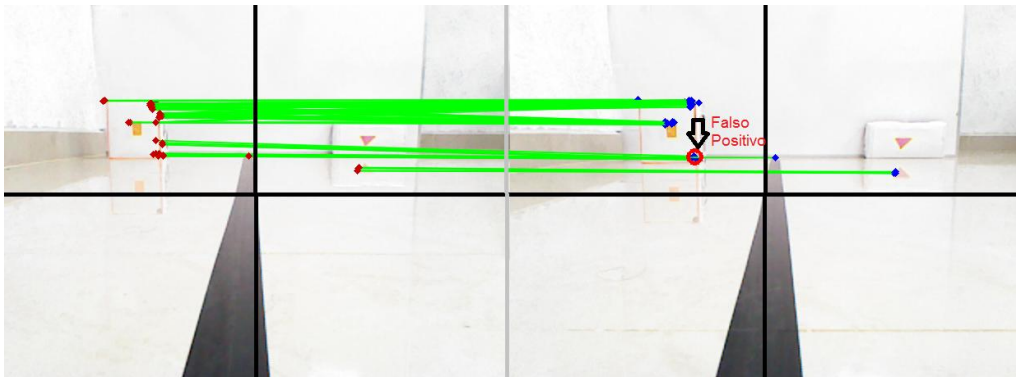


Figura 3.4 Efecto de movimiento trasero en Robotino



Figura 3.5 Detección de correspondencias falsas positivas con el algoritmo de correspondencias propio

Se realizó un segundo modelado para componer el problema de la inestabilidad de movimiento de Robotino; en la siguiente lista se muestran las especificaciones:

- Carrito sobre un riel recto y KINECT montado encima, Figura 3.6

- Se captura 1 *Frame* /s
- KINECT corre a 16 F/s
- Recorrido 2 metros
- Carro a 10 cm/s
- Detector ORB [OpenCV, 2016]
- Descriptor BRIEF (ORB) [OpenCV, 2016]
- Algoritmo de correspondencia *Brute Force* con distancia *Hamming* [OpenCV, 2016]
- Condición de luz controlada (*indoor*)

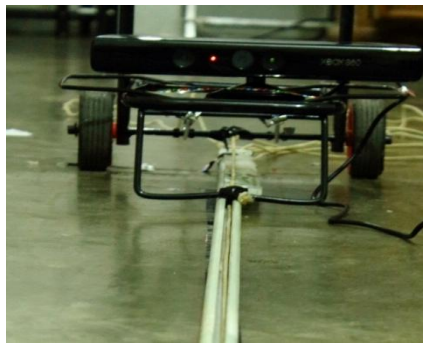


Figura 3.6 Carro sobre un riel con KINECT montado

Con las especificaciones descritas se realizaron en total de 9 recorridos para tener riqueza de correspondencias a diferentes distancias de la cámara; en la Figura 3.7 se puede observar un ejemplo de correspondencia entre dos imágenes de uno de los 9 recorridos.

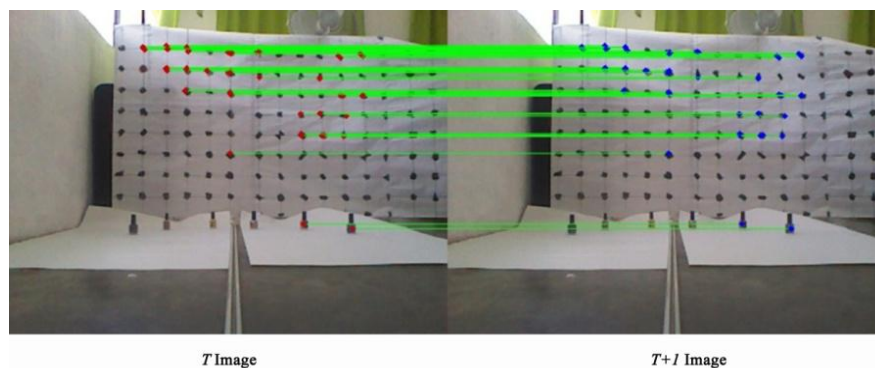


Figura 3.7 Resultado del proceso de correspondencias entre la imagen T y $T+1$, donde las líneas verdes representan las correspondencias correctas

El total de correspondencias obtenidas en los 9 recorridos fueron filtradas: si el incremento de distancia entre un *Frame* y otro fuese más de 10 cm, o la profundidad de algún punto de la correspondencia es menor a 0.8 metros o mayor a 4 metros,

entonces esta correspondencia sería eliminada; también si la profundidad es 0. Después de aplicar los filtros se obtuvo un total de 515 correspondencias.

Este proceso da como resultado una matriz donde se encuentran las coordenadas X , Y , y Z de la imagen T y X , Y , y Z de la imagen $T+1$ del mismo punto, en la Figura 3.8 se puede observar el total de correspondencias obtenidas, divididas por rangos de distancia, se aprecia un distinto comportamiento de movimiento si los puntos están más cerca, el movimiento aparente es grande, si los puntos están más lejos el movimiento aparente es pequeño.

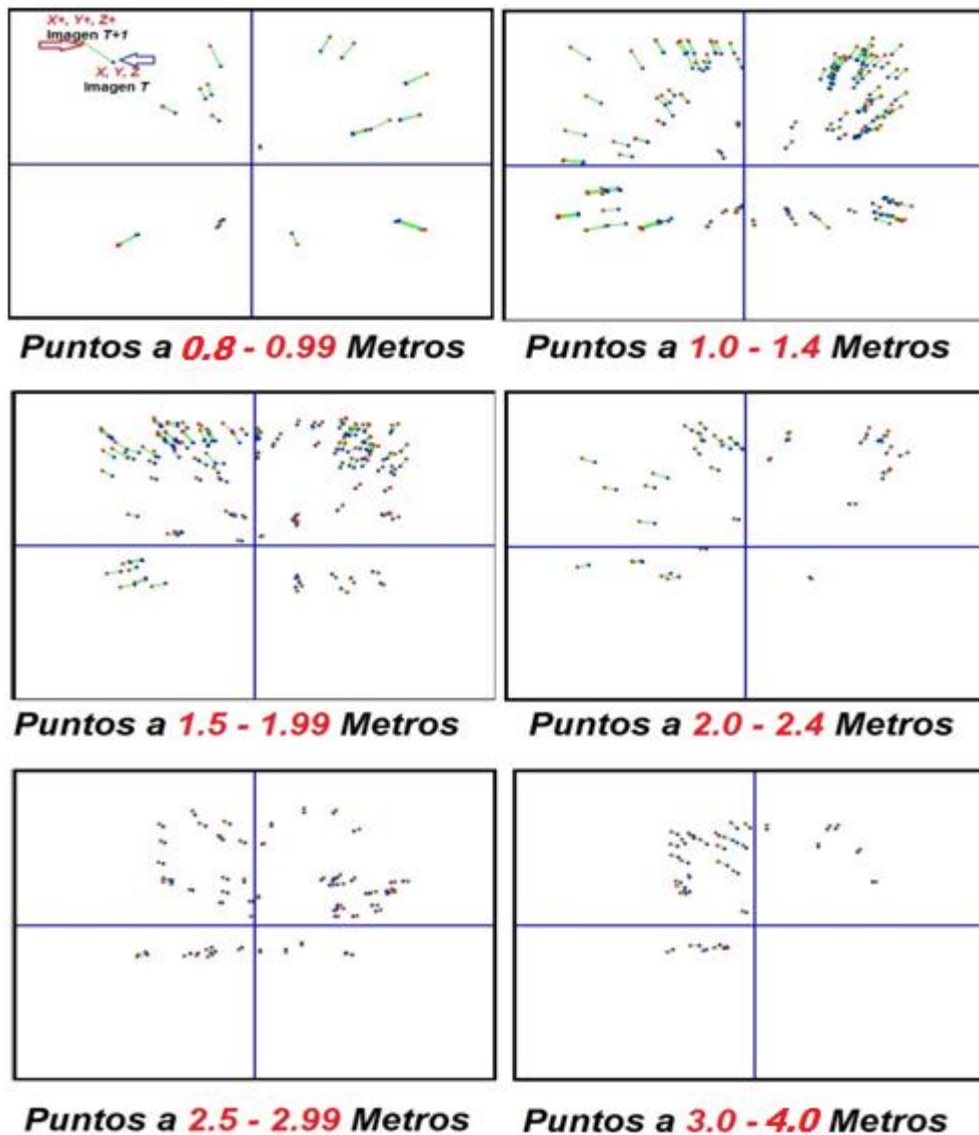


Figura 3.8 Correspondencias a distintas distancias

Como ya se mencionó en la sección 3.2.1 donde se describe el algoritmo de aprendizaje que se utilizó, el proceso de entrenamiento consiste en la optimización

de los parámetros del sistema difuso de manera iterativa a partir de los valores calculados por el sistema y así como los deseados o esperado y^f como resultado de una de entrada u al sistema [Nomura, 1992]. Las condiciones iniciales que necesita el sistema difuso son: conjuntos difusos espaciados uniformemente, bases entre conjuntos adyacentes traslapadas entre sí y pesos iniciales de las reglas difusas en 0.5.

En la Tabla 3.1 se muestra la descripción de los 3 sistemas difusos entrenados para el aprendizaje del modelo de movimiento aparente 3D del fondo de una escena en dirección hacia delante.

El programa de sintonización de parámetros para sistemas difusos fue desarrollado en Matlab [Matlab, 2016] por Rodolfo Castillo Romero [Castillo, 1998] y sobre el cual se realizó el entrenamiento; en la Figura 3.9 se muestra parte del entrenamiento de estos 3 sistemas Difusos, cabe mencionar que se obtuvieron un total de 24 reglas, ya que cada sistema difuso obtuvo 8 reglas, con 2 funciones de pertenencia para cada una de las 3 variables de entrada, se sintonizaran un total de 20 parámetros por cada sistema difuso, estos parámetros son 6 centros a_{ij} , 6 bases b_{ij} y 8 pesos de las reglas w_{ij} el total de parámetros a sintonizar de los 3 sistemas difusos se puede observar en la Tabla 3.3. En la Tabla 3.2 se puede ver el error inicial de cada sistema difuso así también el error mínimo al que se llegó después de la sintonización de parámetros y el número de épocas en las que convergió la sintonización.

Tabla 3.1 Descripción de los 3 sistemas difusos

| Sistema Difuso | Valor de entrada | Sintonizacion para variable | Función de Pertenencia |
|----------------|------------------|-----------------------------|------------------------|
| 1 | X, Y, Z | X+ | X=2, Y=2, Z=2 |
| 2 | X, Y, Z | Y+ | X=2, Y=2, Z=2 |
| 3 | X, Y, Z | Z+ | X=2, Y=2, Z=2 |

Tabla 3.2 Error inicial, Error mínimo y épocas en las que convergió cada sistema

| Sistema Difuso | Error Inicial | Error Mínimo | Épocas |
|----------------|---------------|--------------|--------|
| 1 | 12.362 | 0.0134 | 1500 |
| 2 | 14.130 | 0.0171 | 1500 |
| 3 | 10.492 | 0.0020 | 1500 |

Tabla 3.3 Parámetros a sintonizar cuando las entradas tienen función de pertenencia 2

| Sistema Difuso | Ent. 1 <i>X</i> | Ent. 2 <i>Y</i> | Ent. 3 <i>Z</i> | Centro <i>a_{ij}</i> | Base <i>b_{ij}</i> | Pesos <i>w_{ij}</i> | Total parámetros |
|----------------|--------------------|--------------------|--------------------|---------------------------------|-------------------------------|--------------------------------|---------------------|
| 1 | 2 | 2 | 2 | 6 | 6 | 8 | 20 |
| 2 | 2 | 2 | 2 | 6 | 6 | 8 | 20 |
| 3 | 2 | 2 | 2 | 6 | 6 | 8 | 20 |

```

file edit Debug Desktop window Help
Current Folder: C:\Users\toshiba\Documents\Maestria\4 semestre\Nomura
Shortcuts How to Add What's New
Command History
5000
6000
7000
-1
12/4/2015 12:55 PM --%
Nomura(puntos_x, [2 2 2])
2000
-1
Nomura(puntos_x, [2 2 2])
1500
-1
Nomura(puntos_y, [2 2 2])
1000
2000
12/4/2015 1:10 PM --%
Nomura(puntos_y, [2 2 2])
1000

Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
>> Nomura(puntos_x, [2 2 2])

Asignando valor por omisión en parámetros
Ka = 3.000
Kb = 3.000
Kw = 3.000
Wini = 0.500
INICIALIZANDO PARAMETROS

No datos trn : 160

----- Etapa de aprendizaje -----
i= 1 RMS(n):2.71458 Hasta que epoca hacer PAUSA (-1 Salir) ? 1000
i= 2 RMS(n):2.31305
i= 3 RMS(n):1.97852
i= 4 RMS(n):1.66404
i= 5 RMS(n):1.36946
    
```

Figura 3.9 Entrenamiento en Matlab del sistema difuso 1

En las Figura 3.10, 3.11 y 3.12 se muestran las funciones de pertenencia de los tres sistemas, una antes de sintonizar sus parámetros y otra después de haber sintonizado sus parámetros.

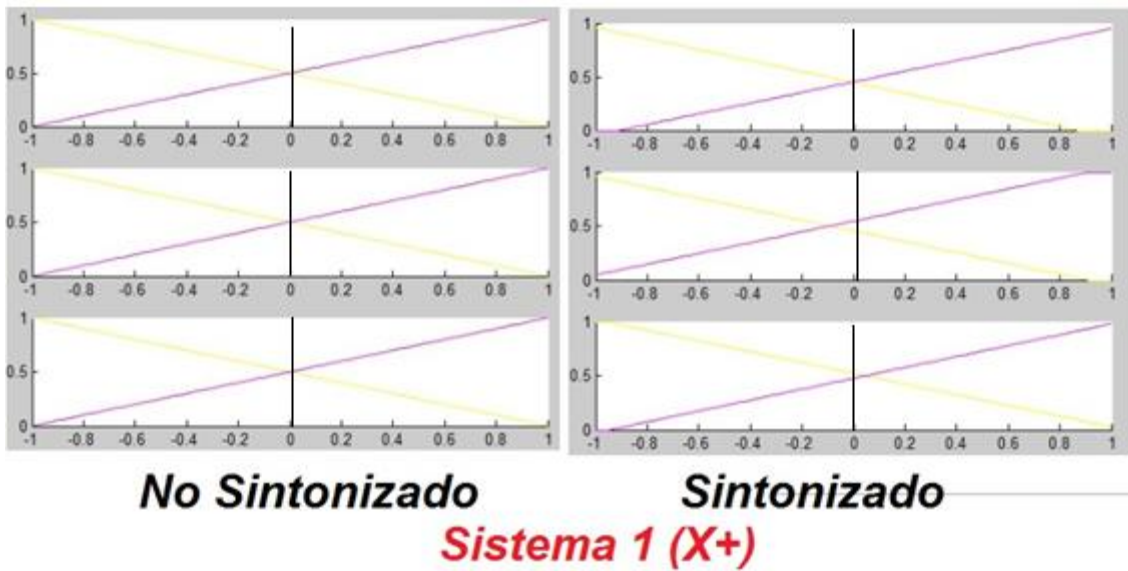


Figura 3.10 Funciones de pertenencia del sistema 1

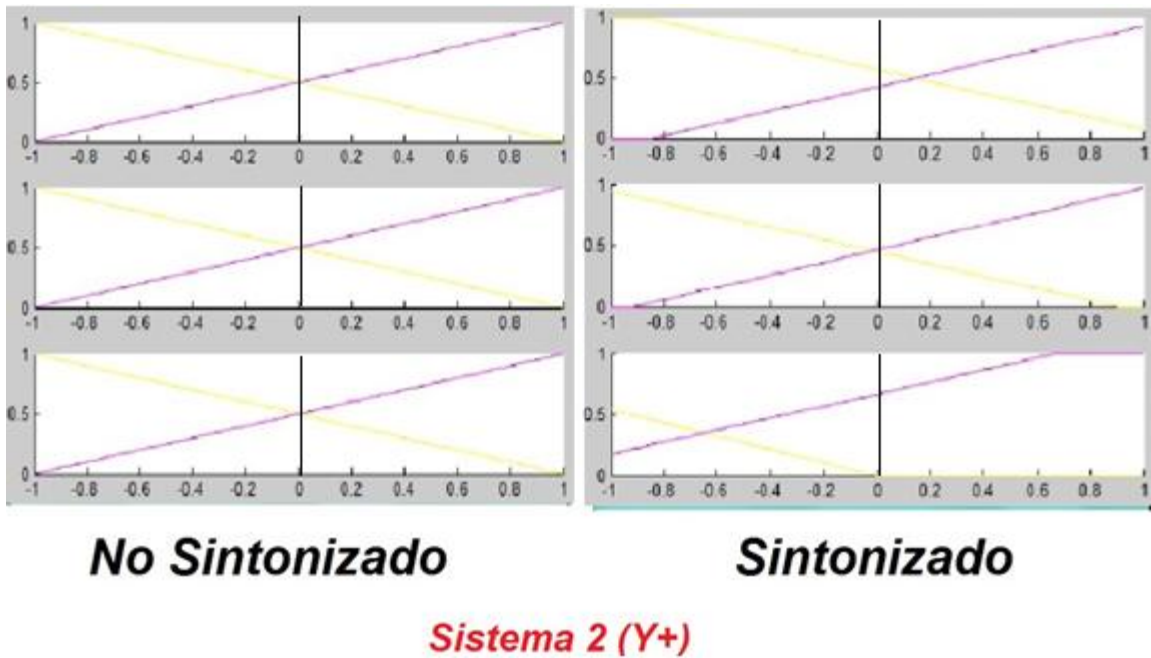


Figura 3.11 Funciones de pertenencia del sistema 2

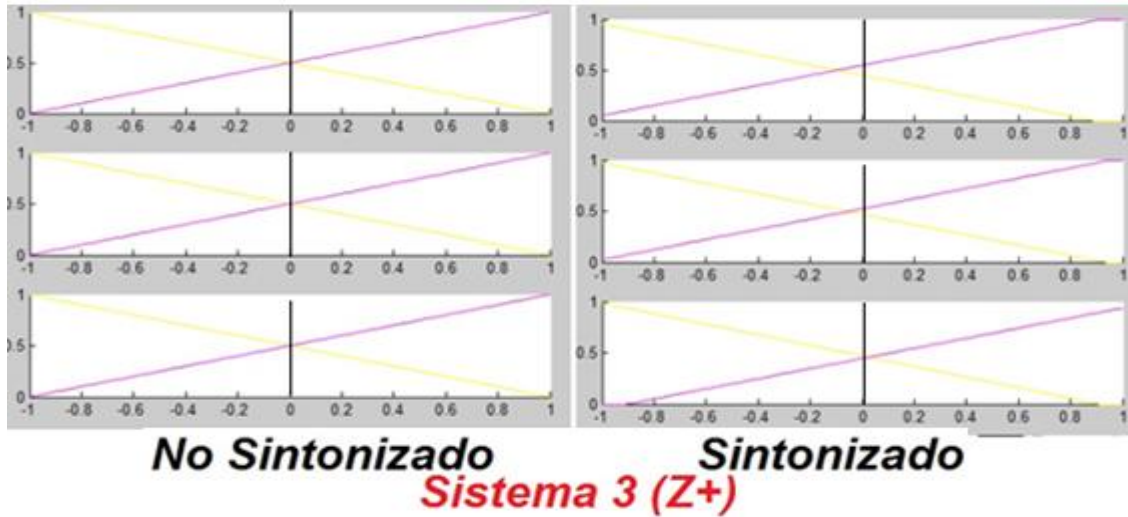


Figura 3.12 Funciones de pertenencia del sistema 3

En las Tabla 3.4 y 3.5 se presentan los valores obtenidos después de la sintonización de los centros, bases y pesos de cada regla, de los 3 sistemas.

Tabla 3.4 Valores de los centros y bases de los triángulos sintonizados, de los 3 sistemas difusos

| Sistema Difuso | Conjunto | Centros a_{ij} | Bases b_{ij} |
|----------------|----------|---------------------|-------------------|
| 1 | C1 | -1.0812 | 3.9806 |
| | C2 | 1.0774 | 3.9396 |
| | C3 | -0.8563 | 4.0387 |
| | C4 | 1.139 | 3.964 |
| | C5 | -0.667 | 4.1056 |
| | C6 | 1.1846 | 3.9702 |
| 2 | C1 | -1.0689 | 3.981 |
| | C2 | 0.9228 | 4.0232 |
| | C3 | -1.0461 | 3.9107 |
| | C4 | 1.1406 | 3.995 |
| | C5 | -0.6994 | 4.1166 |
| | C6 | 1.1335 | 3.9702 |
| 3 | C1 | -1.0668 | 3.9788 |
| | C2 | 0.9274 | 4.0242 |
| | C3 | -0.8582 | 4.0413 |
| | C4 | 1.1391 | 3.9604 |
| | C5 | -1.0741 | 3.9482 |
| | C6 | 1.0785 | 3.9753 |

Tabla 3.5 Valores finales de los pesos de las reglas sintonizadas

| Sistema Difuso | Peso <i>w_{ij}</i> | Valor |
|----------------|-------------------------------|---------|
| 1 | <i>w1</i> | 0.06503 |
| | <i>w2</i> | 0.92836 |
| | <i>w3</i> | 0.03412 |
| | <i>w4</i> | 0.9534 |
| | <i>w5</i> | 0.08167 |
| | <i>w6</i> | 0.9058 |
| | <i>w7</i> | 0.12276 |
| | <i>w8</i> | 0.87165 |
| 2 | <i>w1</i> | 0.03309 |
| | <i>w2</i> | 0.04065 |
| | <i>w3</i> | 0.94377 |
| | <i>w4</i> | 0.94419 |
| | <i>w5</i> | 0.09868 |
| | <i>w6</i> | 0.10615 |
| | <i>w7</i> | 0.92241 |
| | <i>w8</i> | 0.91234 |
| 3 | <i>w1</i> | 0.04763 |
| | <i>w2</i> | 0.05073 |
| | <i>w3</i> | 0.05562 |
| | <i>w4</i> | 0.04921 |
| | <i>w5</i> | 0.95458 |
| | <i>w6</i> | 0.95383 |
| | <i>w7</i> | 0.94646 |
| | <i>w8</i> | 0.96062 |

En las Figuras 3.13, 3.14 y 3.15 se muestran las reglas obtenidas de los 3 sistemas respectivamente, cada regla con su peso correspondiente.

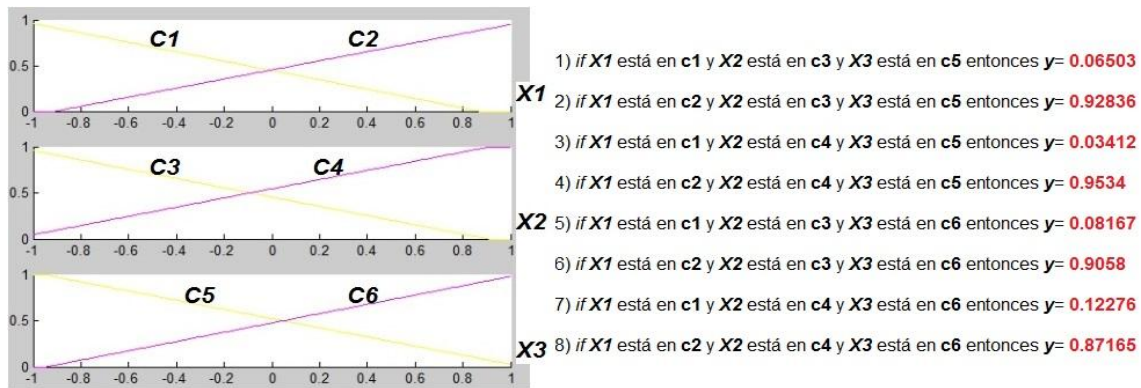


Figura 3.13 Reglas del sistema difuso 1 (X+)

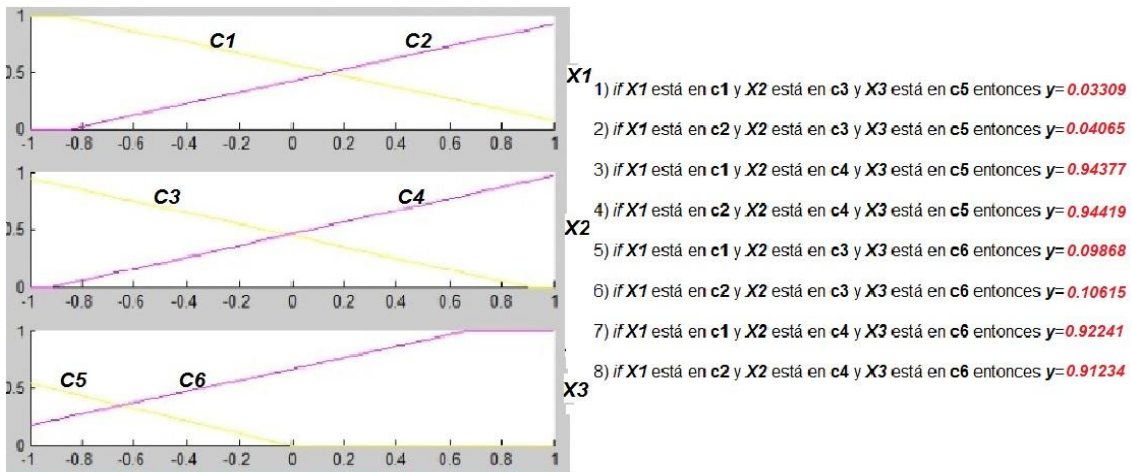


Figura 3.14 Reglas del sistema difuso 2 (Y+)

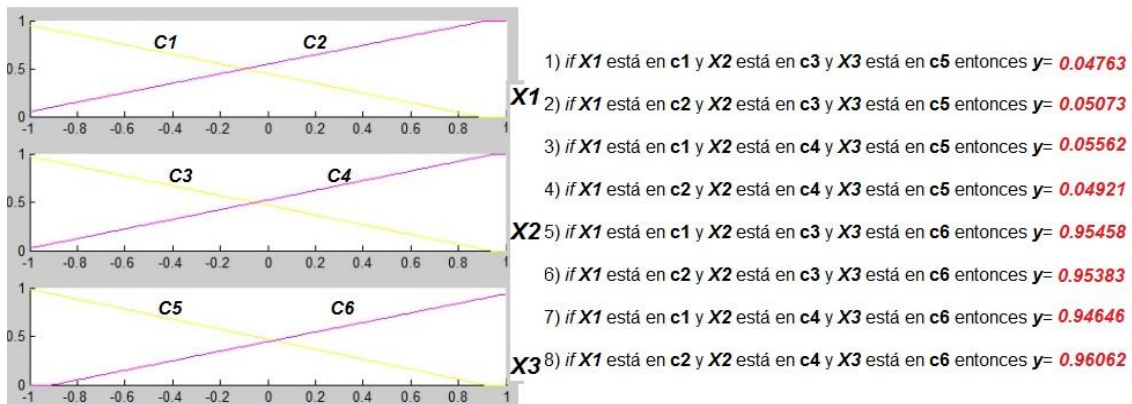


Figura 3.15 Reglas del sistema difuso 3 (Z+)

Obtener las 24 reglas de los 3 sistemas difusos es el final de esta primera fase o fase de entrenamiento ya que en la segunda fase la cual se muestra en la Figura 3.2, es donde se pone a prueba el modelo de movimiento 3D hacia adelante.

3.2.3 Segunda fase: Fase de reconocimiento para detectar objetos móviles

En esta segunda fase se usa el modelo obtenido en la fase anterior para detectar objetos móviles, para esto se programó en el lenguaje C++ un conjunto de procesos los cuales realizan las siguientes tareas:

- Obtener Imagen de profundidad y RGB con KINECT
- Obtener puntos destacados con el algoritmo ORB
- Describir los puntos destacados con ORB (Binario)
- Aplicar el algoritmo de Fuerza bruta con distancia *Hamming* para obtener puntos correspondientes, y obtener X , Y , Z , X_+ , Y_+ y Z_+ , donde X , Y , y Z corresponden a las coordenadas del punto de la Imagen T , mientras que X_+ , Y_+ y Z_+ corresponden a las coordenadas del punto de la imagen $T+1$.
- Fusificar y defusificar con el algoritmo de aprendizaje de Nomura los 3 datos de entrada los cuales son X , Y , y Z para obtener X_+ *estimada*, Y_+ *estimada* y Z_+ *estimada*.
- Se realiza una comparación entre los datos X_+ , Y_+ y Z_+ obtenidos en el proceso de las correspondencias y los datos X_+ *estimada*, Y_+ *estimada* y Z_+ *estimada* obtenidos del proceso de fusificación y defusificación del algoritmo de Nomura; en la Figura 3.16 se muestra un error de predicción el cual fue resuelto multiplicando por un valor de escala como se muestra en la Figura 3.17, con la ecuación 9 se realiza el proceso de comparación para obtener un número (valor de diferencia) el cual sirve para observar el comportamiento de un punto y si este punto se movió acorde al modelo del movimiento 3D hacia adelante, si el punto presenta un valor de diferencia mayor a 10 pixeles, el cual es el umbral propuesto, significa que el punto pertenece a un objeto estático y si el valor de diferencia es mayor al umbral significa que el punto pertenece a un objeto con movimiento propio.
- Se etiqueta con un círculo color rojo aquellos puntos que están por debajo del umbral, esto quiere decir que los puntos móviles son etiquetados con un círculo rojo como se muestra en la Figura 3.18.



Figura 3.16 Gráfica de comparación del error de predicción del modelo, la línea verde muestra el error sin multiplicar por un valor de escala y la línea roja muestra el error multiplicado por un valor de escala

$$\text{Valor de Diferencia} = | X +_c - X +_e + Y +_c - Y +_e | * \text{F.E} \quad \text{Ecuación (9)}$$

| | |
|--|-------------|
| Si el valor de Z se encuentra entre 0 - 1.99 m | F.E. = 0.50 |
| Si el valor de Z se encuentra entre 2 - 2.99 m | F.E. = 0.70 |
| Si el valor de Z se encuentra entre 3 - 4 m | F.E. = 0.80 |

F.E = Factor de escala

Figura 3.17 Fórmula para obtener el valor de diferencia entre el valor obtenido por el proceso de correspondencia y el valor obtenido con el modelo de movimiento aparente en 3D hacia delante de una escena

En la ecuación 9 para encontrar el valor de diferencia al principio sólo se contempló el valor absoluto de la diferencia, pero analizando los datos de un recorrido sin objetos móviles para calibrar el sistema se llegó a la conclusión que el modelo obtenía un margen de error amplio cuando las distancias de los puntos estaban a más de 1.5 metros, por esa razón se contempló la opción de buscar la manera de que el error fuera aproximadamente igual, esto quiere decir que los puntos cercanos tengan el mismo margen de error que los puntos lejanos, en todas las distancias desde 0.8 metros hasta 4.0 metros, para esto se optó por buscar un factor de escala *F.E.* dependiendo de la distancia a la cual se encuentra el punto de KINECT. En la Figura 3.17 se muestran los rangos de distancia y el factor de escala por el cual se multiplica, cabe mencionar que los valores de los 3 distintos factores de escala fueron propuestos después de probar con otros valores y observar que estos 3 valores nivelaban el error.

En la Figura 3.16 se muestra una comparación del comportamiento del natural y el comportamiento del error escalado, se puede ver que el error es aproximadamente equivalente en todas las distancias.

Para encontrar el umbral que discierna si un punto pertenece a un objeto móvil o no, se usó el margen de error que presenta el modelo y que se muestra en la Figura 3.16; se decidió usar como umbral el número 10 porque 10 pixeles de error significa que el punto se movió más de lo que las cosas estáticas se mueven y según la gráfica las cosas que son estáticas no pasan de 6 pixeles de error por lo tanto si un punto se mueve más de 10 pixeles es un punto perteneciente a un objeto que presenta movimiento.

Cuando un punto supera el umbral de 10, el punto es etiquetado con un círculo rojo como se muestra en la Figura 3.18 en la imagen $T+1$.

Corregir el error de predicción mejoro el sistema final ya que de esta manera se minimiza el margen de error de falsos positivos.

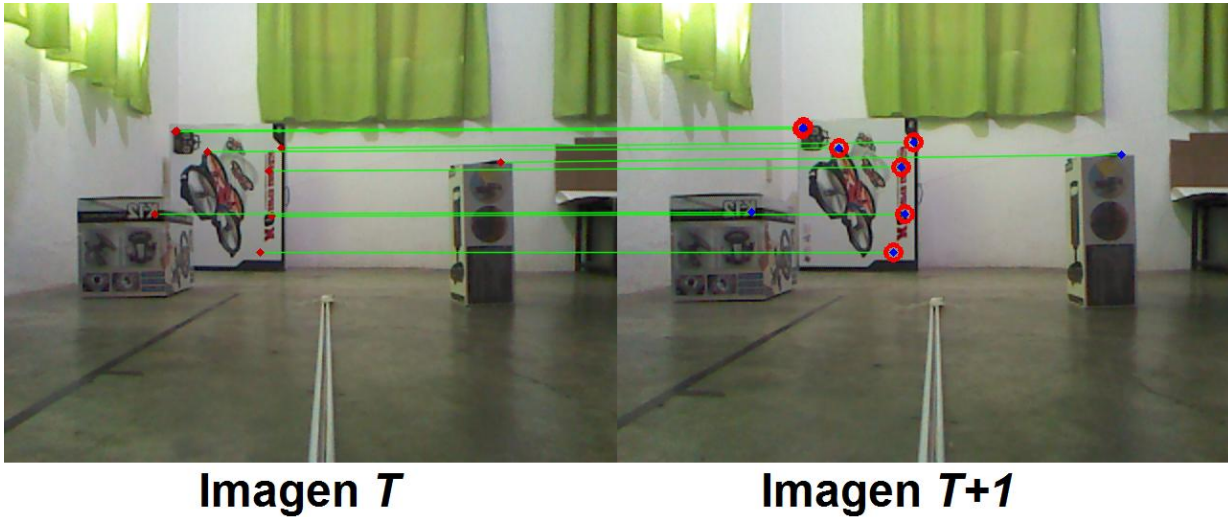


Figura 3.18 Etiqueta al punto móvil con un círculo rojo

En la Figura 3.19 se muestra el diseño de la fase de reconocimiento de este sistema, mostrando cada proceso mencionado en esta sección, desde la detección de puntos destacados hasta el etiquetado de los puntos móviles.

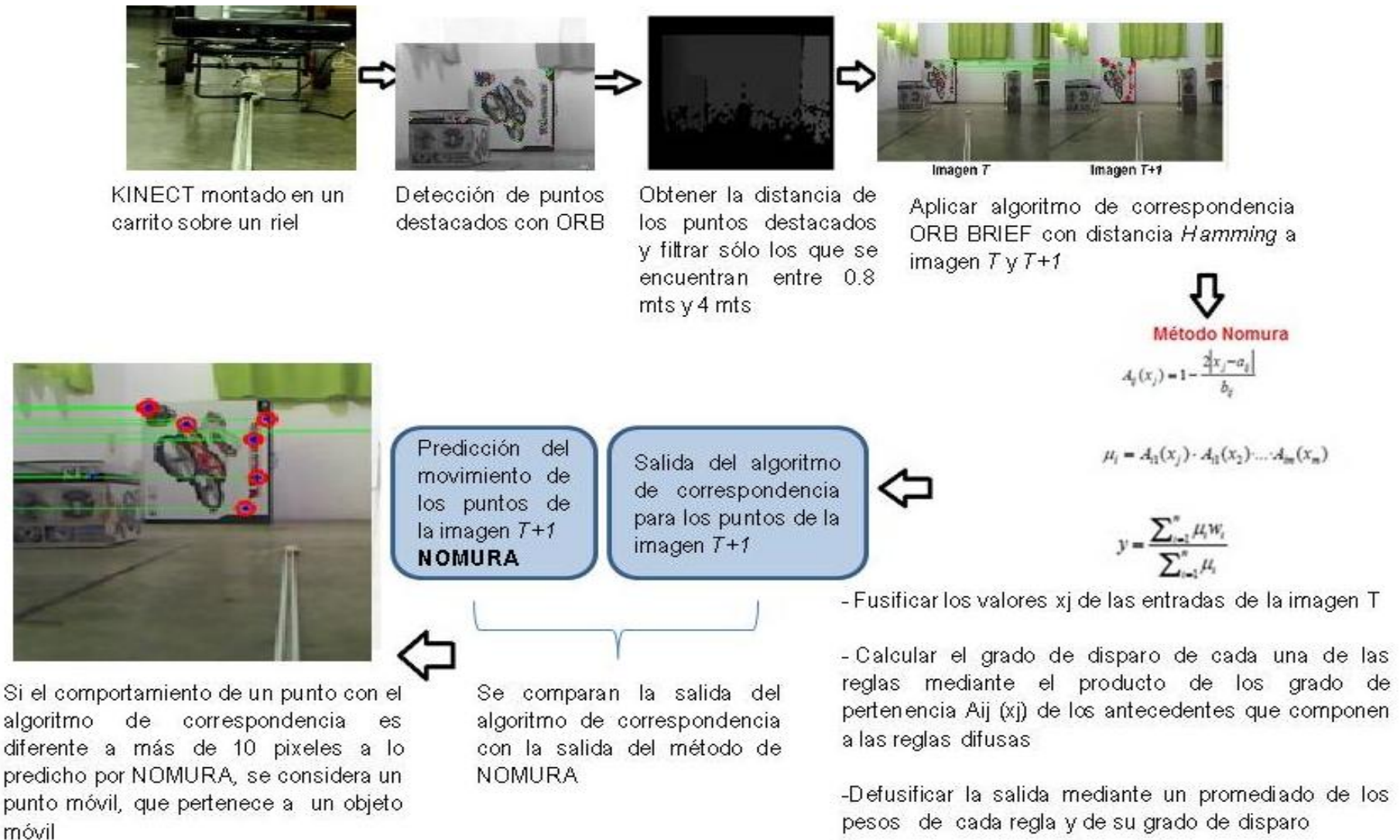


Figura 3.19 Esquema del sistema de la fase de reconocimiento para detectar objetos móviles

En la Figura 3.20 se muestra un diagrama de *Warnier Orr* donde se describe el programa que se realizó para la fase de reconocimiento en este proyecto, el cual contempla el uso de las reglas obtenidas en la primera fase (fase de entrenamiento), en la Tabla 3.6 se muestra los métodos del programa para la fase de reconocimiento.

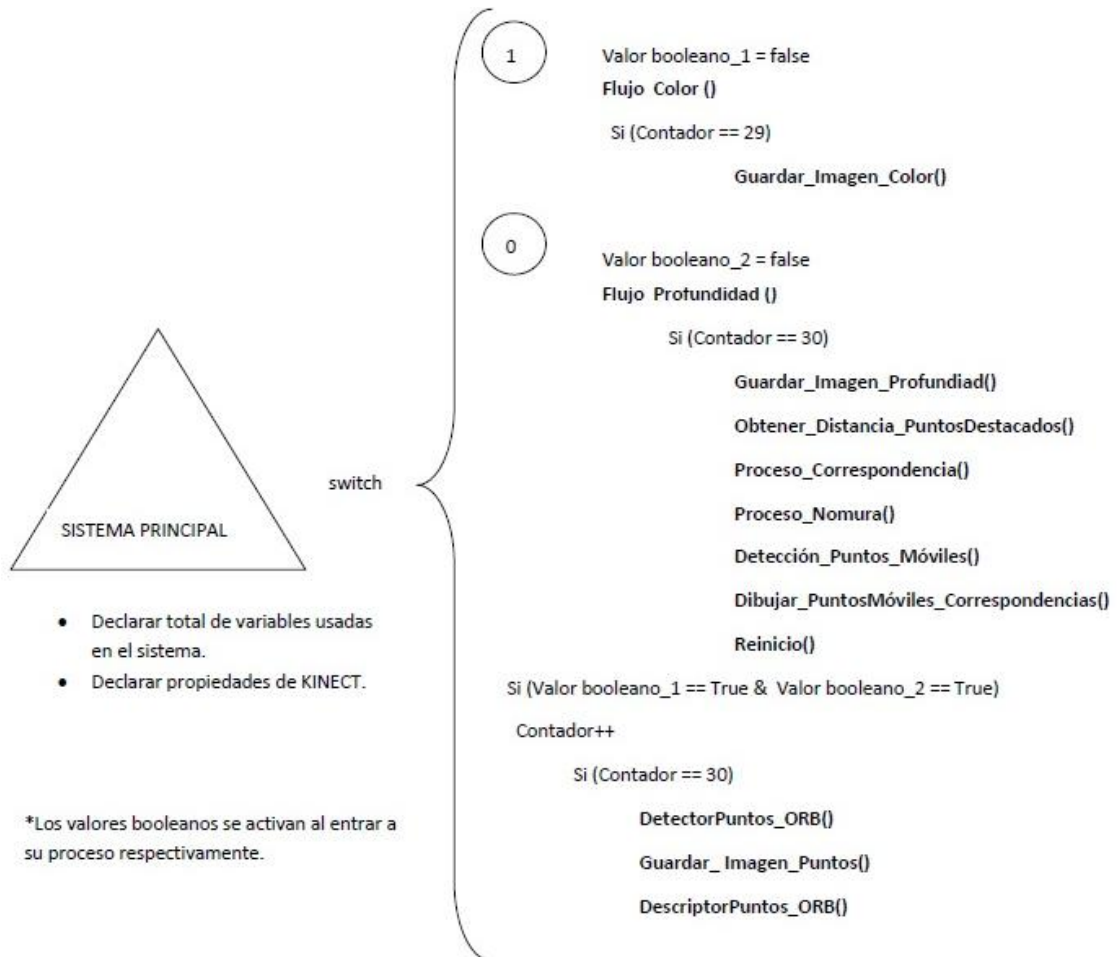


Figura 3.20 Diagrama de *Warnier Orr* de la segunda fase del sistema

Tabla 3.6 Descripción de métodos de C++ de la fase de reconocimiento

| Función | Descripción |
|--|--|
| Guardar_Imagen_Color() | Guarda la Imagen RGB aproximadamente cada 2 segundos la cual se obtiene con el dispositivo KINECT. |
| DetectorPuntos_ORB() | A la imagen que se obtuvo en la función Guardar_Imagen_Color() se aplica el detector de puntos destacados ORB con una función de OpenCV. El detector genera un vector con las coordenadas (x , y) de los puntos encontrados. |
| Guardar_Imagen_Puntos() | Guarda la imagen RGB con los puntos destacados marcados, los cuales se obtuvieron en la función DetectorPuntos_ORB(). |
| DescriptorPuntos_ORB() | A cada coordenada de los puntos destacados encontrados con la función DetectorPuntos_ORB() se le aplica la función de OpenCV la cual describe un punto destacado con el algoritmo ORB. Es un descriptor binario y el vector por cada punto destacado está compuesto por 32 registros, y cada registro por 8 bits. |
| Guardar_Imagen_Profundiad() | Guarda la Imagen de Profundidad aproximadamente cada 2 segundos la cual se obtiene con el dispositivo KINECT. |
| Obtener_Distancia_PuntosDestacados() | Obtiene la distancia de los puntos destacados que se obtuvieron con las función DetectorPuntos_ORB(). |
| Proceso_Correspondencia() | Realiza el proceso de correspondencias de puntos destacados entre la imagen t y t+1. Aplicando distancia de Hamming entre los vectores que se obtuvieron en la función DescriptorPuntos_ORB() de la imagen t con los de la imagen t+1. |
| Proceso_Nomura() | Este proceso se encarga de predecir en donde se encuentra un punto no móvil en un instante posterior. Tiene como vector de entrada las coordenadas X, Y, y Z del instante t, de los puntos que si tuvieron correspondencia en la función Proceso_Correspondencia(). Da como resultado las coordenadas X, Y, y Z del instante t+1. |
| Detección_Puntos_Móviles() | En este proceso se comparan los resultado obtenidos (Coordenadas X, Y, y Z en el tiempo t+1)del algoritmo de correspondencia y el algoritmo Nomura, si alguna coordenada obtenida con el algoritmo de correspondencia tiene una diferencia de 50 a la coordenada obtenida con Nomura, esto significa que ese punto se movió de manera anormal (Según nuestro entrenamiento) y por lo tanto se toma como un candidato a punto móvil perteneciente a un objeto móvil. |
| Dibujar_PuntosMóviles_Correspondencias() | Guarda la imagen RGB en el tiempo t y t+1, dibujando en ella los puntos móviles y sus correspondencias. |
| Reinicio() | Se reinician los contadores utilizados en el proceso, se libera la memoria de todas las matrices de valores usadas y reinicia el proceso hasta que la variable Contador sea igual a 30. |

3.3 Discusión

En este capítulo se describe el diseño del sistema e implementación. Describe problemas que se encontraron en las dos fases; un problema fue realizar el modelado de movimiento 3D hacia adelante, el cual fue resuelto realizando 9 recorridos y así hacer un modelo con mas soporte con un total de 515 patrones para realizar el modelo. Otro problema notorio se encontró en la fase de reconocimiento, en que existía un margen de error de predicción de 8 pixeles que proporcionaba puntos móviles falsos, el cual se corrigió nivelando el error para todas las distancias y así minimizar el error a 5 pixeles.

CAPÍTULO 4

4 EXPERIMENTACIÓN, RESULTADOS Y ANÁLISIS

En este capítulo se presentan pruebas con el sensor que se usó, pruebas preliminares y pruebas de algoritmos detectores de puntos destacados.

También se presenta la experimentación del sistema principal, así como los resultados de la experimentación del sistema principal y el análisis de los resultados, estas pruebas tienen por objetivo detectar puntos móviles pertenecientes a objetos móviles con una cámara móvil.

4.1 Análisis de KINECT

En este trabajo se planteó el uso de una cámara RGB-D, para evitar realizar un arreglo de cámaras como se usa en la visión estereoscópica y para obtener profundidad de las distintas regiones de la imagen. Se decidió usar KINECT [Kinect, 2016], que es una cámara RGB-D diseñada para juegos de la compañía Xbox. Tales juegos requieren interacción con el usuario y es ahí donde ésta cámara, capaz de proporcionar distancia, cumple su objetivo en el juego. Este dispositivo proporciona la profundidad con la técnica de “tiempo de vuelo” (*Time of flight*) [Shao, 2014]. En la Figura 4.1 se muestra KINECT y en la Figura 4.2 se muestra la imagen RGB y la Imagen de profundidad, ambas son obtenidas con KINECT.



Figura 4.1 Dispositivo KINECT versión Xbox 360



Figura 4.2 Imagen RGB e imagen de profundidad obtenidas con el dispositivo KINECT

En la Tabla 4.1 se pueden observar las especificaciones del dispositivo KINECT para la versión Xbox 360.

Tabla 4.1 Características de KINECT Xbox 360 [Han, 2013]

| No | Sensores | Campo de visión | Flujo de datos | Seguimiento |
|----|--|--------------------------|--------------------------|-------------------------|
| 1 | Lentes de color y sensación de profundidad | Horizontal: 57° | DEPTH: 640 x 480 a 30 Hz | Sigue 6 personas |
| 2 | Micrófono multi-arreglo | Vertical : 43° | RGB: 1280 x 1024 a 10 Hz | Sigue 20 articulaciones |
| 3 | Ajuste de sensor con su motor de inclinación | Inclinación: ± 27° | | |
| 4 | | Profundidad: 0.8 – 3.5 m | | |

Para realizar la captura con KINECT de la imagen de color y de profundidad como muestra la Figura 4.2, se usó OpenCV 2.4.9 [OpenCV, 2016] para funciones gráficas, OPENNI 2 y NITE (*PrimeSense*) [OpenNi, 2016] como controlador de comunicación entre KINECT y la computadora, también se realizó un programa en lenguaje C++ sobre la plataforma de programación Eclipse versión Luna [Eclipse, 2016].

Se detectó que la imagen RGB y la imagen de profundidad no son correspondientes de pixel a pixel, ya que la imagen RGB es más grande por 63 mil pixeles. Se recurrió a la función *IMAGE_REGISTRATION_DEPTH_TO_COLOR* de OpenNI, la cual excluye un borde alrededor de la imagen RGB, y la imagen de profundidad refleja el borde de la imagen RGB como un espacio vacío. Esta función proporciona una correspondencia de pixel a pixel. En la Figura 4.3 se muestra la imagen de profundidad con el borde vacío.



Figura 4.3 Imagen de profundidad con el borde vacío.

Por otro lado se verificó la viabilidad de los datos de profundidad que proporciona KINECT, esto se realizó con un programa para obtener la profundidad del pixel central y encontrar el rango en el que KINECT detecta profundidad. Se identificó que

KINECT proporciona el mapa de profundidad a partir de 0.50 metros hasta 4 metros como se muestra en la Figura 4.4.

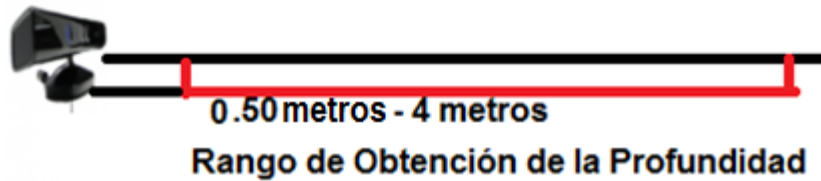


Figura 4.4 Rango de distancia que detecta KINECT

En la Figura 4.5 se muestran 3 puntos rojos que no tienen la misma distancia de la cámara al punto. KINECT da la misma distancia en los tres puntos rojos; 1 metro. Existen funciones en OpenNi que dan la distancia de la cámara al objeto, pero en este proyecto se usó la distancia horizontal.

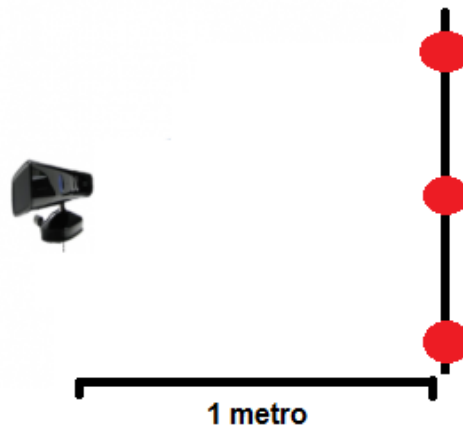


Figura 4.5 Obtención de distancia de tres puntos distribuidos de forma vertical

Otra duda o problema que surge del estudio de este sensor son los llamados hoyos negros (*Black Holes*) [Shao, 2014]. El mapa de profundidad generado por el sensor genera huecos en algunas partes del plano, esos huecos que presentan profundidad 0 se deben a píxeles que son visibles para la cámara RGB pero no para la cámara de profundidad, debido a su posición distinta, o bien a superficies reflectoras de la luz.

Existen técnicas para solucionar la problemática de los hoyos negros, pero para efectos de este proyecto, se eliminan los puntos con distancia 0.

Estas son las características del sensor KINECT que fueron analizadas para su uso en este proyecto.

4.1.1 Prueba preliminar con algoritmo Nomura

Como ya antes se mencionó, esta metodología fue elegida debido a que es capaz de sintonizar tanto conjuntos difusos (bases y centros) como la base de reglas del sistema difuso. Se realizó una prueba preliminar con este método, en el cual se capturó la dinámica de movimiento 2D hacia delante usando un método directo 2D *Global Motion Models* [Irani, 1999], se realizó un programa, en el cual a una imagen en blanco de 640 x 480 pixeles se le colocaron pixeles negros aleatoriamente, y con este modelo de movimiento se puede observar la dinámica de los puntos un instante después.

El modelo de movimiento 2D se realizó de la siguiente manera:

Donde u y v es el incremento en X y Y respectivamente.

X_p = ubicación del pixel en x Real
 Y_p = ubicación del pixel en y Real
 X_i = ubicación del pixel en coordenadas cartesianas
 Y_i = ubicación del pixel en coordenadas cartesianas

Δu = incremento en X
 Δv = incremento en Y

Primero se paso la ubicación de los pixeles reales a pixeles en un plano cartesiano de la siguiente manera:

$$X_i = X_p - (\text{Mitad de las columnas}) = X_p - 320$$

$$Y_i = (\text{Mitad de los reglones}) - Y_p = 240 - Y_p$$

Después de haber obtenido las coordenadas de los pixeles, se obtiene el incremento para X y Y , multiplicando por un valor de escala a X_i y a Y_i , en este caso por 0.1.

$$\Delta u = 0.1 * X_i$$

$$\Delta v = 0.1 * Y_i$$

Ya teniendo los incrementos Δu y Δv se clasifican según el cuadrante al que pertenezcan, para obtener las direcciones. El signo del incremento depende del cuadrante, como se muestra en la Figura 4.6.

- I cuadrante los signos de movimiento son($X > 320$ y $Y < 240$): +, -
- II cuadrante los signos de movimiento son($X < 320$ y $Y < 240$): -, -
- III cuadrante los signos de movimiento son($X < 320$ y $Y > 240$): -, +
- IV cuadrante los signos de movimiento son($X > 320$ y $Y > 240$): +, +

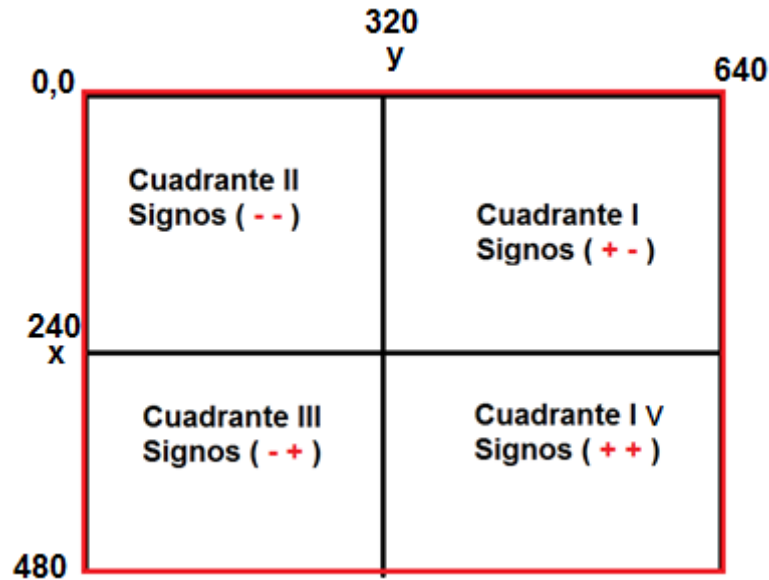


Figura 4.6 Cuadrante perteneciente

Se dibuja una línea entre el pixel en el tiempo t y el mismo pixel en el tiempo $t+1$, en la Figura 4.7 se muestra hacia donde se mueve el pixel después de haber aplicado la dinámica de movimiento 2D hacia delante, donde el punto verde es el pixel en el tiempo t y el punto rojo es hacia donde se movió el pixel en el tiempo $t+1$.

$$X_p(\text{Nueva}) = X_p + \Delta u$$

$$Y_p(\text{Nueva}) = Y_p + \Delta v$$

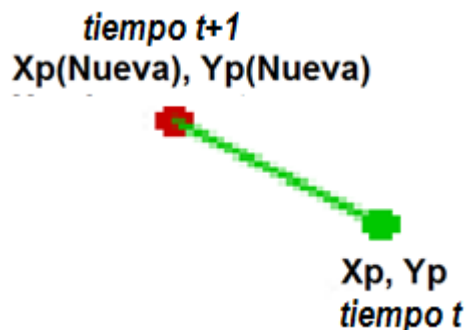


Figura 4.7 Movimiento de un pixel aplicando el método *Global motion models* 2D hacia delante

En la Figura 4.8 se muestra el modelo obtenido con el método *Global motion models* 2D hacia adelante.

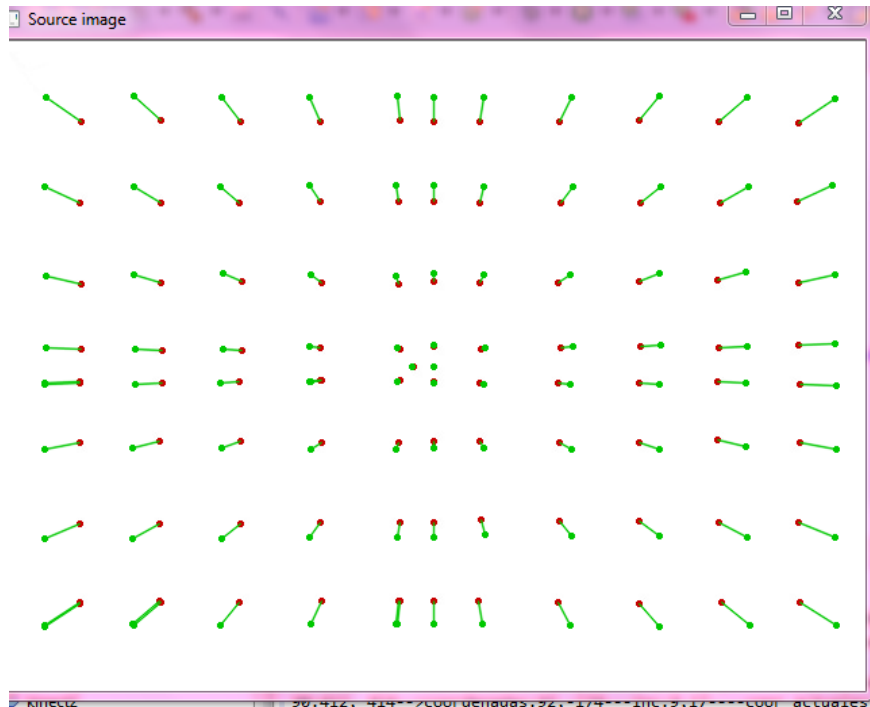


Figura 4.8 Modelo de movimiento 2D hacia adelante

De esta prueba se obtuvo una X y una Y inicial, también una X y una Y final. Con la fase de aprendizaje del algoritmo de Nomura, se entrenaron dos sistemas difusos como se muestra en la Tabla 4.2.

Tabla 4.2 Entrada a sintonizar

| Sistema Difuso | Antecedente | Consecuente | Funciones de pertenencia |
|----------------|-------------|-------------|--------------------------|
| 1 | X, Y | $X+$ | $X = 2, Y = 3$ |
| 2 | X, Y | $Y+$ | $X = 2, Y = 3$ |

Los dos sistemas difusos se entrenaron hasta la época 1,000. El sistema entrenado se probó con 6 patrones diferentes; en la Tabla 4.3 se pueden observar los dos sistemas con sus conjuntos, centros y bases; en la Figura 4.9 se muestra en color azul los puntos de entrada al sistema y en color rojo las salidas dadas por el sistema difuso, cabe mencionar que el punto que se localiza en el centro es justamente el punto central de la imagen, se puede

apreciar un comportamiento apegado al modelo de movimiento 2D hacia delante.

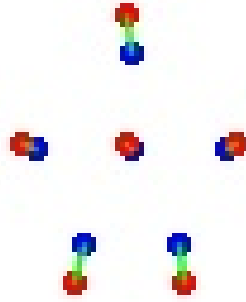


Figura 4.9 Resultado del Método Nomura

Tabla 4.3 Valores de los centros y bases de los triángulos sintonizados, de los 2 sistemas difusos

| Sistema Difuso | Conjunto | Centros | Bases |
|----------------|----------|----------|----------|
| | | a_{ij} | b_{ij} |
| 1 | C1 | -1.07958 | 3.96378 |
| | C2 | 1.07468 | 3.95909 |
| | C3 | -1.12052 | 1.96595 |
| | C4 | -0.03477 | 2.14243 |
| | C5 | 1.09463 | 1.97168 |
| 2 | C1 | -0.97995 | 4.00689 |
| | C2 | 1.01930 | 3.99349 |
| | C3 | -1.06041 | 2.34595 |
| | C4 | 0.00088 | 1.72799 |
| | C5 | 1.08098 | 2.34044 |

La fase de aprendizaje del algoritmo de Nomura se lleva a cabo mediante una sintonización de parámetros y la fase de reconocimiento del algoritmo de Nomura se lleva a cabo mediante Fusificación y Defusificación de reglas difusas.

La sintonización se llevó a cabo con un programa realizado en MATLAB por Rodolfo Castillo [Castillo, 1998], y la Fusificación y Defusificación de reglas difusas se realizó mediante un programa en C++ en el desarrollo de esta tesis.

Los resultados del proceso se muestran en la Figura 4.9, donde el sistema en su fase de reconocimiento se le introdujeron 6 puntos, los cuales muestran hacia donde se movieron los puntos en un tiempo $T+1$.

4.1.2 Prueba de algoritmos detectores y descriptores de puntos destacados

Para efectos de este proyecto se realizó un análisis de los algoritmos detectores y descriptores de puntos destacados, para seleccionar el algoritmo que mejor resultados obtenga, en el Anexo B se encuentra la documentación de los algoritmos detectores, descriptores y de correspondencias.

Se implementaron los 4 algoritmos detectores ORB, SURF, *Harris* y FAST; a continuación se presentan los resultados obtenidos aplicados a dos imágenes consecutivas.

Se realizaron las siguientes pruebas, donde se utilizaron tres escenarios; cada imagen con su imagen consecutiva correspondiente; en la Figura 4.10 se pueden apreciar las imágenes de los escenarios con las que se probaron los algoritmos detectores (Pies, Silla, objeto pequeño) y en las Figuras 4.11, 4.12 y 4.13 los resultados obtenidos al aplicar los algoritmos detectores.



Figura 4.10 Imágenes para probar los algoritmos detectores de puntos destacados

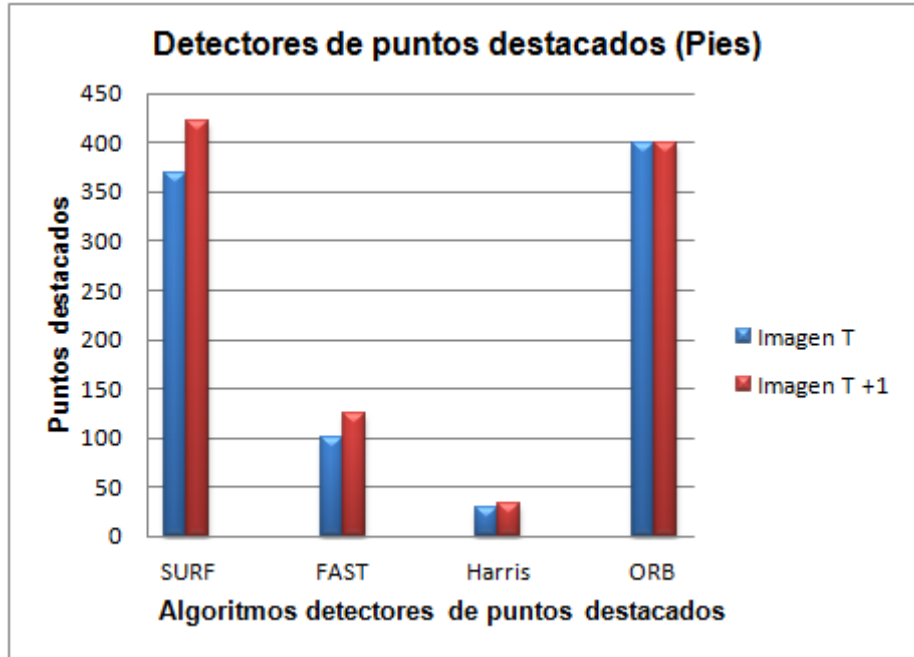


Figura 4.11 Graficas de los resultados obtenidos al aplicar los algoritmos detectores a la imagen Pies

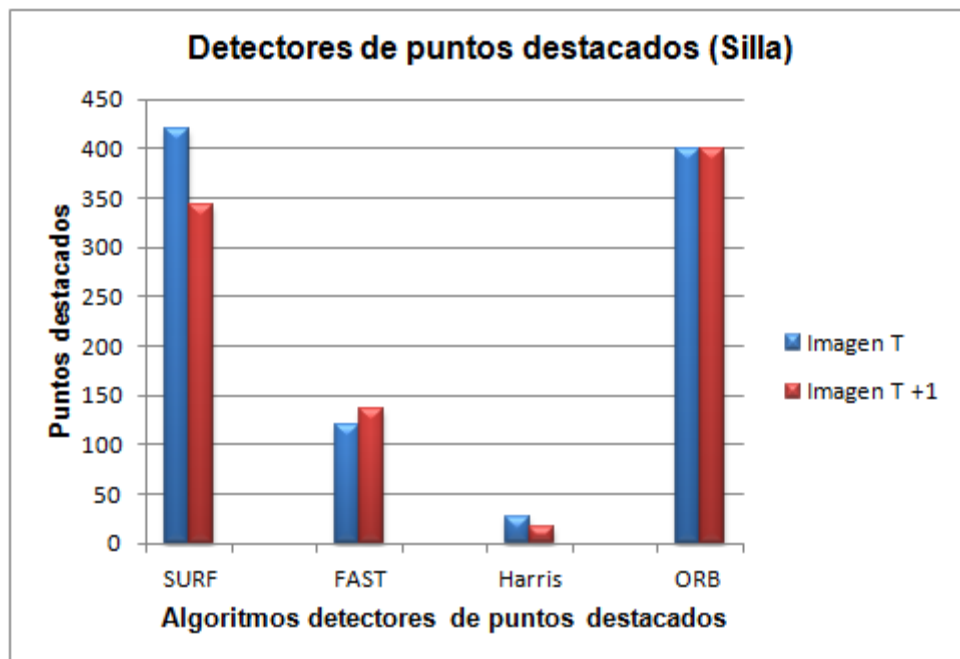


Figura 4.12 Graficas de los resultados obtenidos al aplicar los algoritmos detectores a la imagen Silla

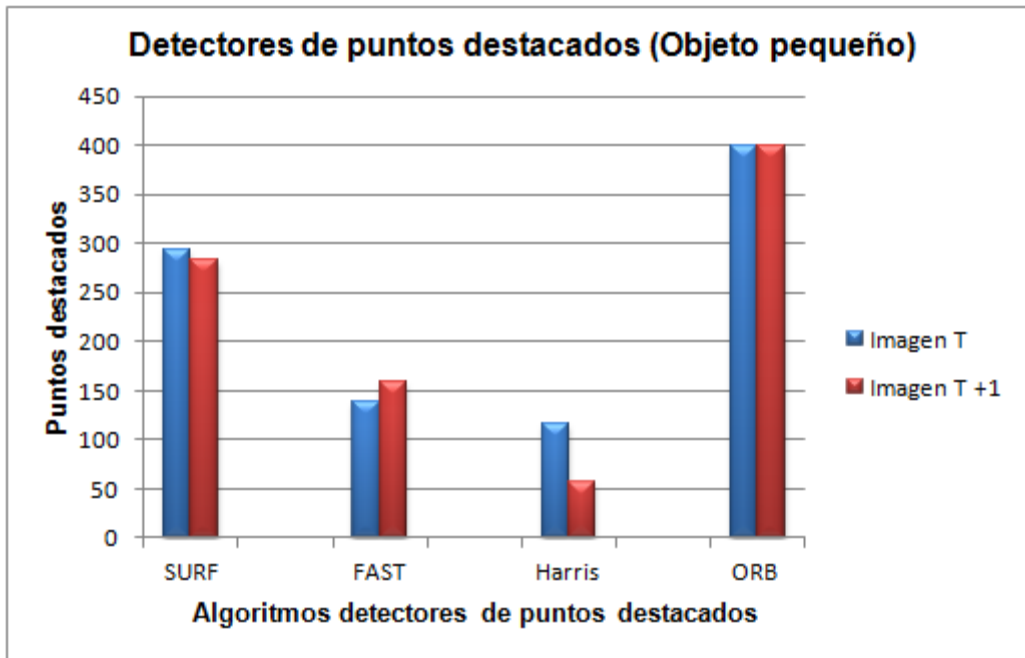


Figura 4.13 Graficas de los resultados obtenidos al aplicar los algoritmos detectores a imagen Objeto pequeño

Como se observa en las Figuras 4.11, 4.12 y 4.13 en las tres pruebas con distinto escenario los algoritmos detectores que obtuvieron mejores resultados fueron SURF y ORB, porque estos dos algoritmos detectaron más puntos destacados, por lo tanto se decidió seguir trabajando con los dos detectores.

Para seleccionar el algoritmo detector de puntos destacados entre SURF y ORB se tuvo que realizar un recorrido hacia delante, en un salón con luz artificial, esto para medir cuál de los dos algoritmos detectores tenía mejor rendimiento a la hora de realizar correspondencias usando como descriptor el algoritmo propio; de igual manera el algoritmo de correspondencia propio, se usó Robotino de FESTO [FESTO, 2015], que avanza a una velocidad de 10 cm/ segundo, se usaron cajas para obtener más correspondencias; el recorrido fue de 1.5 metros, y se obtiene un fotograma cada medio segundos. En la Figura 4.14 se muestra el ambiente creado para la prueba y en la Figura 4.15 se muestra Robotino al cual se le montó KINECT en la parte de enfrente



Figura 4.14 Ambiente creado para obtención de correspondencias



Figura 4.15 Robotino de FESTO, con KINECT

En la Figura 4.16 se observan las correspondencias obtenidas con ORB y SURF en un *Frame*.

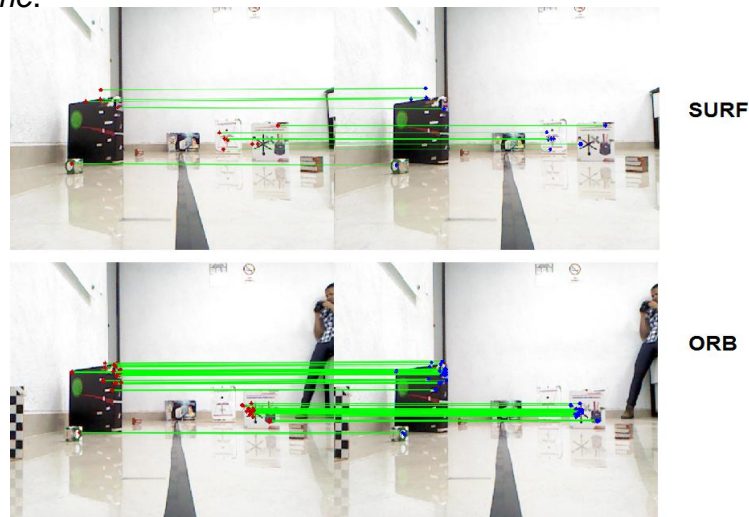


Figura 4.16 Correspondencias con detector SURF y ORB, y descriptor y algoritmo de correspondencia propio

Cabe mencionar que el rango de distancia que proporciona KINECT es de 0.5 a 4 mts. Se desecharon los puntos destacados que tenían distancia menor a 0.6 mts dejando 0.10 mts respecto desde donde KINECT proporciona distancia, así como también desechando puntos con distancia mayor a 4 mts.

En la Figura 4.17 se muestra la gráfica comparando el detector SURF y ORB con el número de correspondencias obtenidas durante todo el recorrido con un total de 30 frames en los cuales se llevó a cabo la obtención de las correspondencias, y como se puede apreciar el algoritmo que mejor resultados obtuvo como detector fue el algoritmo ORB. Dada esta comparación el algoritmo que se usó para seguir desarrollando esta tesis fue ORB por que obtuvo más puntos destacados en todo el recorrido a comparación de SURF.

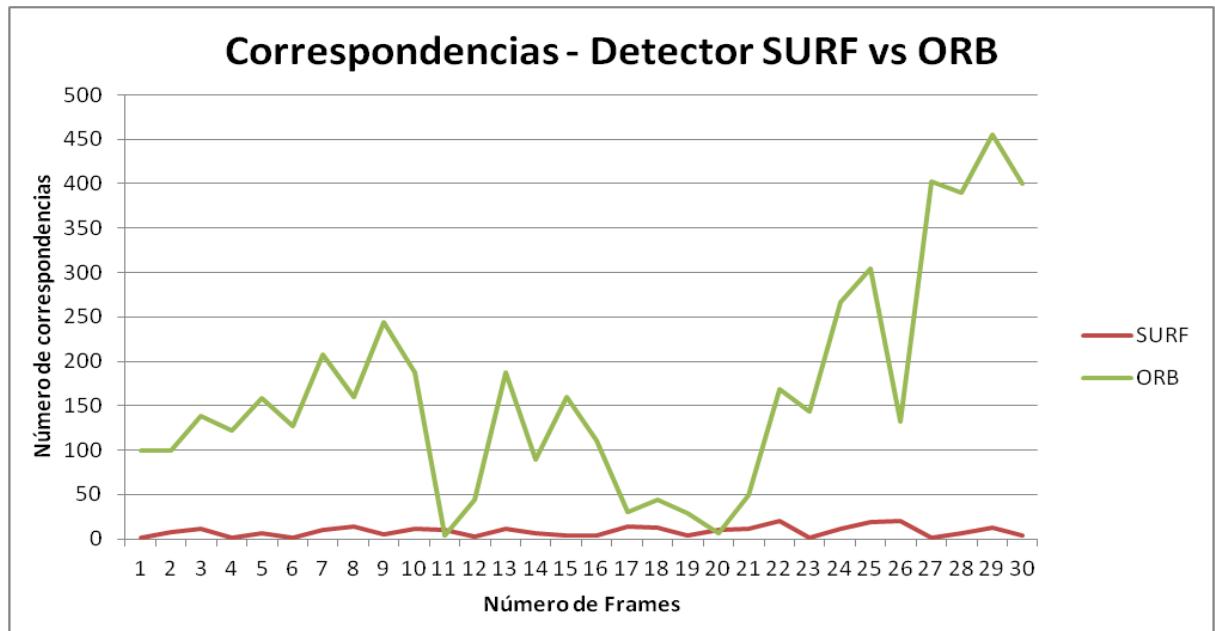


Figura 4.17 Comparación de Algoritmo detector SURF y ORB, dentro de un proceso de correspondencias con algoritmo descriptor propio y de correspondencia propio

4.2 Especificaciones técnicas

Para realizar cada una de las pruebas al sistema principal se requirió de las siguientes especificaciones de *Hardware* y *Software*.

Hardware:

- Procesador: Intel(R) Core(TM) i7-45100 CPU 2.00Ghz 2.60 GHz.
- Memoria RAM: 8 GB.
- Sistema Operativo: 64 bits: Windows 10.
- GPU: Tarjeta Gráfica NVIDIA GeForce 840M.
- KINECT versión de Xbox 360 [KINECT, 2016].

Software de desarrollo:

- Lenguaje C++.
- OpenNi como controlador entre KINECT y la Lap Top [OpenNI, 2016].
- OpenCV [OpenCV, 2016].
- Plataforma de programación Eclipse versión Luna [Eclipse, 2016].

4.3 Plan de pruebas

El objetivo de plan de pruebas es validar el funcionamiento del sistema desarrollado a lo largo de esta tesis, para comprobar el cumplimiento de los objetivos propuestos.

En la Tabla 4.4 se muestran las pruebas realizadas, el tipo de objeto móvil, la distancia a la que se encuentra el objeto móvil de la cámara y el comportamiento del objeto móvil (dirección de movimiento).

Tabla 4.4 Lista de pruebas

| Prueba | Distancia inicial del objeto móvil a la cámara en metros | Tipo de objeto móvil | Comportamiento del objeto móvil |
|--------|--|----------------------|---------------------------------|
| 1 | 3 | Caja | Izquierda-Derecha |
| 2 | 2.5 | Caja | Izquierda-Derecha |
| 3 | 3 | Caja | Arriba-Abajo |
| 4 | 2.5 | Caja | Arriba-Abajo |
| 5 | 3 | Caja | Abajo- Arriba |
| 6 | 2.5 | Caja | Abajo- Arriba |
| 7 | 3 | Pies | Izquierda-Derecha |
| 8 | 2.5 | Pies | Izquierda-Derecha |
| 9 | 3 | Caja | Hacia al Frente |
| 10 | 2 | Caja | Hacia atrás |
| 11 | 3 | Caja | Diagonal-Hacia delante |
| 12 | - | - | Todo estático |

Cada una de estas pruebas se realizó montando KINECT sobre un carrito guiado sobre un riel que avanza a 10 cm / s, KINECT corre a 16 *Frames*/seg, del cual se captura 1 *Frame*/s, cabe mencionar que las pruebas se realizaron en un cuarto con luz controlada. En las Figuras del Anexo A se pueden observar los *Frames* de las 12 pruebas donde los puntos rojos indican un punto móvil, en la Figura 4.18 se muestran los *Frames* de la prueba 2.

El objetivo de las pruebas 1 al 11 es detectar el objeto móvil en diferente dirección o comportamiento, mientras que la prueba 12 su objetivo es no detectar ningún punto móvil, ya que el ambiente de esta prueba es estático.

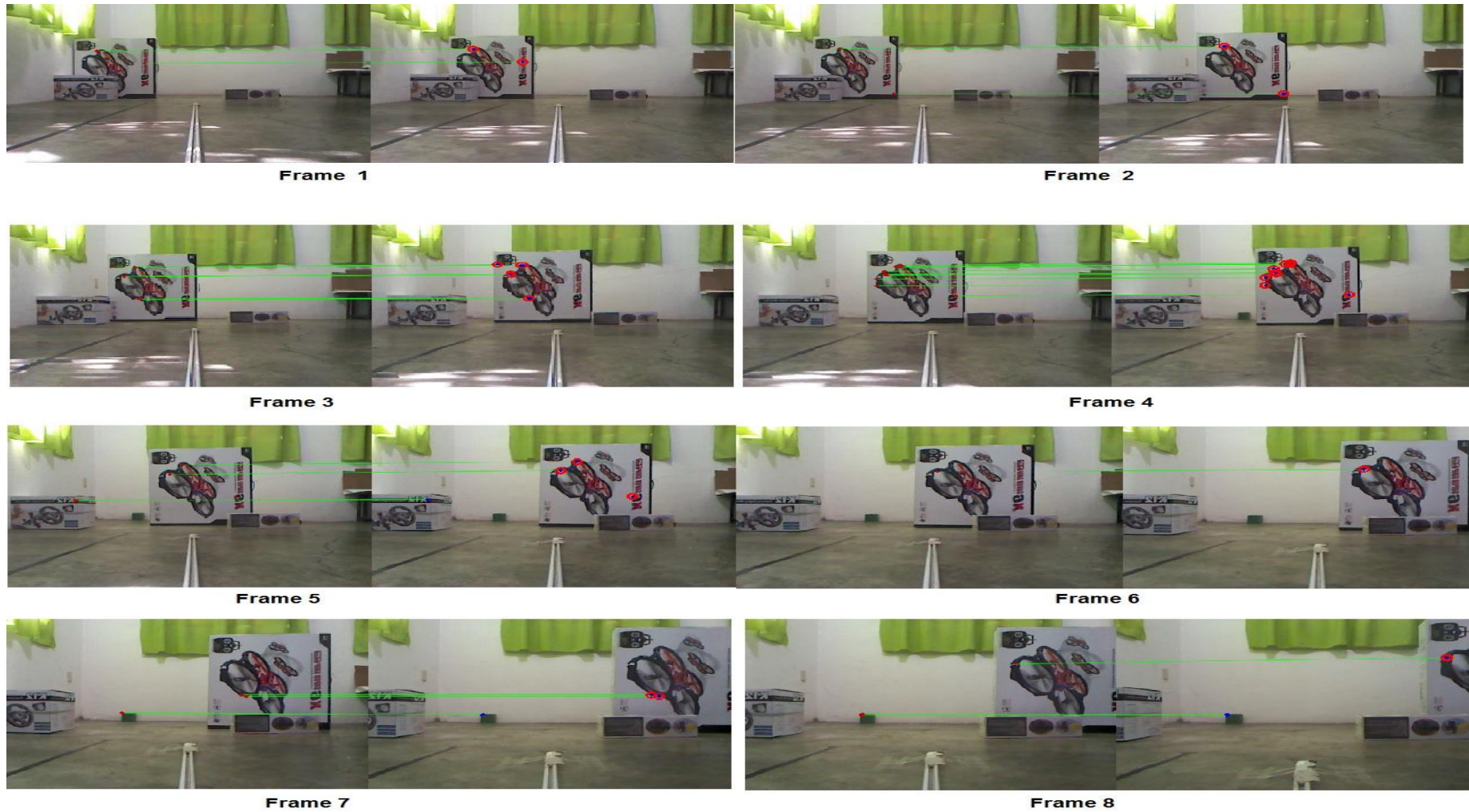


Figura 4.18 *Frames* de la prueba 2, donde el objeto móvil se mueve de izquierda a derecha, con una distancia inicial a la cámara de 2.5 metros

4.4 Resultados

En esta sección se mostrarán los resultados obtenidos en las 12 pruebas presentadas en la Tabla 4.4; en las pruebas 1 al 11 se evalúa el comportamiento del sistema ante un objeto móvil y en la prueba 12 se evalúa el comportamiento del sistema ante un ambiente con objetos estáticos.

Cabe señalar que no se obtuvieron falsos positivos por lo tanto no se incluyó tal columna en las Tablas de resultados de las 12 pruebas. El total de Tablas donde se presenta información de cada una de las pruebas se ubican en el Anexo A. Cada prueba generó 8 *Frames* a excepción de las pruebas 9 y 10, en la prueba 9 esto se debió a que el objeto móvil se acercaba a la cámara a la misma velocidad que la cámara se recorría hacia delante, en el *Frame* 6 dejó de detectar correspondencias debido a que se encontraba a menos de 0.8 metros de distancia, medida a la cual se restringió el proceso de correspondencias dado la limitación del sensor KINECT. La prueba 10 no obtuvo 8 *Frames* dado que el objeto se movía hacia atrás y en el *Frame* 7 el objeto topó con la pared y dejó de ser móvil.

El objetivo del sistema en las pruebas del 1 al 11 es detectar al menos un punto móvil en cada *Frame* que indique que existe un objeto móvil en la escena, con al menos un punto móvil, se asigna el porcentaje de eficacia de 12.5 % en el *Frame*, y en la prueba 12 el objetivo es no detectar puntos móviles, si no se detecta algún punto móvil entonces se asigna el porcentaje de eficacia del sistema de 12.5 % en el *Frame*.

En las Tablas del Anexo A se se muestran los resultados de las 12 pruebas, en las cuales se puede ver el total de correspondencias por *frame*, el total de puntos móviles por *frame* y el porcentaje de eficacia del sistema por *Frame* y en total de toda la prueba.

En la Tabla 4.2 se muestra el total de porcentaje de eficacia de las 12 pruebas, dando un total de 88.2 % de eficacia de todo el sistema.

Tabla 4.5 Promedio de eficacia en porcentaje de las 12 pruebas

| Prueba | Porcentaje de eficacia del sistema % |
|---------------|---|
| 1 | 100 |
| 2 | 100 |
| 3 | 100 |
| 4 | 87.5 |
| 5 | 62.5 |
| 6 | 87.5 |
| 7 | 75 |
| 8 | 50 |
| 9 | 33.2 |
| 10 | 100 |
| 11 | 75 |
| 12 | 100 |
| | 88.2 % |

En la Figura 4.19 se muestra una gráfica que refleja lo descrito en la Tabla 4.5, que es el porcentaje de eficacia por cada una de las 12 pruebas, se puede observar que las pruebas 1, 2, 3, 10 y 12 presentan el 100% de eficacia, esto quiere decir que detectaron puntos móviles donde existe un objeto móvil, las pruebas 4, 5, 6, 7 y 11 con menos porcentaje de eficacia entre 87.5% y 62.5 % logran tener un desempeño no excelente pero bueno, mientras que la prueba 8 y 9 con un porcentaje de efectividad de 50% y 33.2 % respectivamente, no logra satisfacer las expectativas del sistema.



Figura 4.19 Gráfica del porcentaje de eficacia del sistema en las 12 pruebas

4.5 Análisis de Resultados

En este capítulo se han descrito y desarrollado cada una de las 12 pruebas en las cuales se evalúa el sistema con objetos móviles, esto para medir la eficacia del sistema en distintas situaciones donde el objeto móvil se mueve de manera diferente, a continuación se analizan los resultados de las pruebas.

Los resultados que se obtuvieron en las 12 pruebas tuvieron un porcentaje de eficacia de 88.2 %, pero según los resultados de las pruebas individuales se puede observar que en algunas de ellas el sistema se comportó mejor que en otras, esto quiere decir que el sistema no es 100% eficaz para todos los casos, mostró un bajo rendimiento en las pruebas 9 y 8, donde el objeto móvil (caja a 3 m) se mueve hacia al frente, así también en la prueba donde el objeto móvil (pies a 2.5 m) se mueve de izquierda a derecha, la prueba 9 con 33.2 % de eficacia y la prueba 8 con 50% de eficacia respectivamente.

En la prueba 5 donde el objeto móvil (caja a 3 m) se mueve de abajo hacia arriba se obtuvo un 62.5 % de eficacia esto debido a que no encontró puntos móviles en todos los *Frames* sólo en 5 *Frames*,

En la prueba 7 donde el objeto móvil (pies a 3 m) se mueve de izquierda a derecha y en la prueba 11 donde el objeto móvil (caja a 3 m) se mueve diagonalmente hacia delante, se obtuvo un 75 % de eficacia en ambas, esto debido a que en al menos dos *Frames* no se detectaron puntos móviles.

En la prueba 4 donde el objeto móvil (caja a 2.5 m) se mueve de arriba hacia abajo y en la prueba 6 donde el objeto móvil (Caja a 2.5m.) se mueve de abajo hacia arriba, se obtuvo un 87.5 % de eficacia en ambas, esto debido a que en al menos un *Frame* no se detectaron puntos móviles.

En las pruebas 1, 2, 3, 10 y 12 se obtuvieron los mejores resultados con 100% de eficacia, la prueba 1 y 2 fue donde el objeto móvil (caja a 2.5 m y a 3 m respectivamente) se mueve de izquierda a derecha, la prueba 10 fue donde el objeto móvil (caja a 2 m) se mueve hacia atrás.

Cabe mencionar que la razón por la cual no se detectaron puntos móviles en algunos *Frames*, fue porque no se detectaron correspondencias en los objetos móviles y por lo tanto no se detecta como punto móvil. El umbral del detector de puntos destacados de ORB, se tuvo que disminuir a 80, ya que en la prueba presentada en el capítulo 3, cuando se decidió usar ORB, el umbral con el que se probó fue de 400, se disminuyó por el coste computacional que presentaba y por lo cual se tuvo que usar una máquina con un procesador Intel core i7 y tarjeta gráfica NVIDIA GeForce 840M, aparte de que las correspondencias fueron filtradas según las distancias, esto se explica en la sección 3.2.2.

Este sistema es eficaz cuando los objetos móviles se encuentran a una distancia inicial de la cámara de 2 a 3 metros y con dirección izquierda a derecha o derecha a izquierda, de arriba hacia abajo, de abajo hacia arriba, diagonal hacia al frente y hacia atrás, pueden ser objetos pequeños como unas

piernas o objetos grandes como una caja.

Se podría mejorar en el futuro la detección de móviles para los casos en los que el sistema no tuvo éxito realizando un modelo más robusto e implementar el sistema en una maquina con más capacidad de procesamiento.

4.6 Discusión

El sistema final presentado en esta tesis detecta puntos móviles pertenecientes a objetos móviles, mientras la cámara se encuentra en movimiento rectilíneo hacia adelante. Los resultados de la experimentación que se presentan en la Figura 4.19 muestran que 70% de las pruebas realizadas tuvieron éxito detectando puntos móviles, mientras que el resto tuvo dificultad para detectar puntos móviles, esto puede ser debido a la velocidad o dirección a la cual se movía el objeto. Cabe señalar que las deficiencias que tiene KINECT también afectaron a la captura de puntos móviles ya que los hoyos negros presentes en la imagen afectan a la captura de puntos destacados al no detectar distancia en tales puntos. Teniendo una mejor cámara se podrían solucionar tales problemas.

CAPÍTULO 5

5 CONCLUSIONES

En este capítulo se presentan los objetivos alcanzados, las conclusiones finales, aportaciones y los trabajos futuros del presente trabajo.

5.1 Objetivos alcanzados

En la Tabla 5.1 se presentan los objetivos de la tesis y comentarios de los objetivos alcanzados.

Tabla 5.1 Objetivos alcanzados

| Objetivos | Comentarios |
|--|--|
| Estudiar, implementar y evaluar las técnicas de segmentación para la detección de objetos en movimiento con cámara RGB-D en movimiento | En el transcurso de la tesis se estudiaron algunas técnicas de segmentación, pero no se implementó ninguna, se optó por usar puntos característicos para la detección de objetos móviles |
| Estudiar las técnicas para seguimiento de objetos en movimiento | El proceso de la detección de puntos móviles tomó más tiempo de lo propuesto, por lo tanto no se llegó a las técnicas de seguimiento |
| Realizar un sistema de detección de objetos en movimiento de un robot móvil, con una cámara RGB-D | Se realizó 1 sistema que detecta puntos móviles pertenecientes a objetos móviles con una cámara móvil RGB-D a velocidad constante |
| Analizar e implementar algoritmos detectores y descriptores de puntos destacados para elegir el mejor y aplicarlo a este proyecto | Se analizaron e implementaron 4 algoritmos detectores de puntos destacados; SURF, ORB, Harris y FAST. Se analizaron 2 descriptores; BRIEF y propio |
| Analizar el funcionamiento del dispositivo KINECT | Se analizó el dispositivo KINECT, documentando en esta tesis 2 pruebas de su funcionamiento. |
| Elaborar un modelo de movimiento 3D con dirección hacia adelante | Se elaboraron 2 modelos de movimiento 3D hacia adelante, pero se usó el mejor modelo |
| Encontrar un método de aprendizaje para aprender el modelo de movimiento | Se optó por 1 método de aprendizaje [Nomura, 1992] y se implementó |

5.2 Aportaciones

Las principales aportaciones derivadas de este trabajo son:

- Se desarrollo un método para detectar puntos móviles pertenecientes a un objeto móvil.
- Se diseño e implemento un modelo de movimiento 3D hacia delante para la fase de aprendizaje.
- Se detectó la problemática que conlleva el uso de KINECT y en su mayoría fue solucionada con funciones de OpenNI.
- Se desarrollo un algoritmo de correspondencia propio, aunque en el sistema final se usó el algoritmo de correspondencia BRIEF con distancia de *Hamming*.

5.3 Productos

Los productos derivados de este trabajo son:

- Un sistema de visión artificial que detecta puntos móviles que pertenecen a un objeto en movimiento los cuales son tomados con una cámara RGB-D móvil y funciona en tiempo real.
- Se implementó la fase de fusificación y defusificación de un método de aprendizaje de reglas de inferencia difusas basado en el método descendente para predecir la dinámica de puntos 2D.
- Se implementó la fase de fusificación y defusificación de un método de aprendizaje de reglas de inferencia difusas basado en el método descendente para predecir la dinámica de puntos 3D.
- Se implementaron 4 algoritmos para detección de puntos destacados; ORB, SURF, FAST y Harris, finalmente se optó por el uso del detector ORB.
- Se implementó un programa para verificar cómo KINECT proporciona la distancia de puntos.

5.4 Conclusiones Finales

En este trabajo de investigación se estudiaron las técnicas de detección de puntos destacados, así como algoritmos de correspondencias, se estudiaron los aspectos técnicos del hardware KINECT de Xbox 360, para poder comprender cómo obtener el máximo aprovechamiento de este dispositivo. Se observó que KINECT tiene un fallo natural que son los Agujeros negros (*Black holes*), los cuales son huecos donde no se detecta profundidad, ya que este dispositivo fue diseñado para detectar regiones, en este proyecto se detectan puntos característicos y no regiones, esto conlleva a posibles fallos de detección de distancias en algunos puntos destacados y tales puntos son eliminados por que presentan distancia de 0.

Por otro lado aunque se tuvo que disminuir el umbral de ORB se lograron obtener los puntos necesarios para tener buenas correspondencias, esto fue posible porque el descriptor ORB es binario y porque el algoritmo de correspondencia BRIEF se aplicó usando distancia *Hamming*. No se obtuvo ninguna correspondencia falsa positiva en las pruebas.

El algoritmo de aprendizaje Nomura que fue utilizado para el modelado de la dinámica de movimiento de los puntos 3D hacia delante en este proyecto, mostró en las dos implementaciones un buen comportamiento predictivo de la dinámica de puntos tanto en 2D como en 3D, aunque este método es equivalente a una Red Neuronal *Back-propagation* en términos de aproximación, se decidió usar el algoritmo de Nomura porque tiene la ventaja de generar reglas que son más entendibles que los pesos de la Red Neuronal.

El sistema final fue evaluado en tiempo real con 2 distintos tipos de objetos móviles, el primero fue una caja y el segundo fueron unas piernas, el sistema tuvo un porcentaje de efectividad de 88.2% en las pruebas, pero se concluye que el sistema es eficaz para objetos móviles con dirección izquierda a derecha o derecha a izquierda, arriba hacia abajo, abajo hacia arriba, hacia atrás y diagonal hacia el frente.

5.5 Trabajos futuros

Los trabajos futuros que pueden dar continuidad a este proyecto de tesis son:

- Agregar una dimensión más al algoritmo de aprendizaje para hacer un modelado que detecte puntos móviles pertenecientes a un objeto móvil no sólo con la cámara en dirección hacia delante, sino también en curvas, esta dimensión puede ser el parámetro de velocidad angular en torno al eje Y.
- Agregar una dimensión más al algoritmo de aprendizaje para hacer un modelado que detecte puntos móviles pertenecientes a un objeto móvil con una cámara a velocidades distintas.
- Utilizar otra cámara RGB-D que no sea KINECT, u otra versión mejorada de KINECT.
- Utilizar una computadora con mejor capacidad de procesamiento

REFERENCIAS

- [Albiol, 2000] A. Albiol, V. Naranjo ; I. Mora. ***Real-time high density people counter using morphological tools***. Pattern Recognition, 2000. Proceedings. 15th International Conference, 652 - 655 vol.4. IEEE, 2000.
- [Bay, 2006] H. Bay, T. Tuytelaars, L.J. Gool, ***"SURF: Speeded" up Robust Features***, In Proceedings of the 9th European Conference on computer vision, Graz, Austria, 7-13, pp. 404-417, Springer, 2006.
- [Castillo, 1998] Rodolfo Castillo, ***Sintonización de controladores difusos basada en el método de gradiente descendente***, Tesis de maestría en ciencias de la computación en CENIDET, 1998.
- [Cucchiara, 2004] Rita Cucchiara, Andrea Prati, Roberto Vezzani. ***Real-time motion segmentation from moving cameras***. Real-Time Imaging. ELSEVIER. 2014.
- [Derpanis, 2004] Konstantinos G. Derpanis, ***The Harris Corner Detector***, Computation Vision, Springer, 2004.
- [Guerrero, 2012] Wilfrido Rolando Guerrero Alban. ***Detección y descripción de puntos característicos en imágenes multiespectrales utilizando esquemas clásicos***. Proyecto de fin de carrera, Escuela Superior Politécnica de Litoral, 2012.
- [Han, 2013] Jungong Han, Ling Shao, Dong Xu and Jamie Shotton, ***Enhanced Computer Vision with Microsoft Kinect Sensor: A Review***. IEEE transactions on cybernetics, IEEE 2013.

- [Irani, 1999] M. Irani, P. Anandan, ***About direct methods***, Vision Algorithms: Theory and Practice, Lecture Notes in Computer Science Volume 1883, 2000, pp 267-277, Springer, 1999.
- [Jin, 2011] Won Jin Kim, In-So Kweon, ***Moving Object Detection and Tracking from Moving Camera***. 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). IEEE, 2011.
- [Khan, 2012] Atif Khan, Febin Moideen, Juan Lopez, Wai L. Khoo, Zhigang Zhu. ***KinDectect: Kinect Detecting Objects***. 13th International Conference, ICCHP 2012. pp 588-595. Springer 2012.
- [Low, 2010] S. M. Low. ***A wireless webcam-based robotic vision system***. Intelligent and Advanced Systems (ICIAS), 2010 International Conference on 1 - 4, IEEE, 2010
- [Lucas, 1981] Bruce D. Lucas and Takeo Kanade. ***An Iterative Image Registration Technique with an Application to Stereo Vision***. IJCAI, pages 674-679, 1981
- [Moo, 2013] Kwang Moo Yi, Kimin Yun, Soo Wan Kim, Hyung Jin Chang, Hawook Jeong and Jin Young Choi. ***Detection of Moving Objects with Non-Stationary Cameras in 5.8ms: Bringing Motion Detection to your Mobile Device***. IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2013.
- [Mora, 2009] David Mora, Andrés Páez y Julián Quiroga Sepúlveda, ***Detección de Objetos Móviles en una Escena Utilizando Flujo Óptico***, XIV Simposio de tratamiento de señales, imágenes y visión artificial – STSIVA 2009.
- [Nomura, 1992] Hiroshi Nomura, Isao Hayashi, Noboru Wakami. ***A learning method of fuzzy inference rules by descent method***. Fuzzy Systems, IEEE International Conference, Matsushita Electric Ind. Co. Ltd., Osaka, Japan, 1992.

- [Rosten, 2006] Edward Rosten and Tom Drummond, ***Machine Learning for High-Speed Corner Detection***, ECCV 2006, Part I, LNCS 3951, pp. 430–443, Springer, 2006.
- [Rublee, 2011] Ethan Rublee Vincent Rabaud Kurt Konolige Gary Bradski, ***ORB: an efficient alternative to SIFT or SURF***, Computer Vision (ICCV), IEEE, 2011.
- [Shao, 2014] Shao, L., Han, J., Kohli, P., Zhang, Z, ***Computer Vision and Machine Learning with RGB-D Sensors***, Springer, 2014
- [Shibata, 2008] Masaaki Shibata. Yuichiro Yasuda. Masahide Ito, ***Moving Object Detection for Active Camera based on Optical Flow Distortion***, Proceedings of the 17th World Congress The International Federation of Automatic Control Seoul, Korea, July 6-11, ELSEVIER, 2008.
- [Stauffer, 1999] C. Stauffer and W. Grimson. ***Adaptive background mixture models for real-time tracking***, Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference, volume 2, pages 2 vol. (xxiii+637+663), 1999.
- [Tomasi, 1991] C. Tomasi and T. Kanade. ***Detection and tracking of point features***. Technical report, Carnegie Mellon University, Apr.1991.
- [Vidal, 2014] Aimer Christian Vidal González, ***Evaluación de Técnicas para la detección de las partes estáticas y las partes móviles de una escena en secuencias de video***. Tesis de maestría en ciencias de la computación en CENIDET, 2014.
- [Yachida, 1991] Yachida, M. Asada and S. Tsuji, ***Automatic analysis of moving image***, IEEE Transactions on Pattern Analysis and Machine Intelligence,.PAMI-3, No.1, pp.12-20, 1981

Referencias en línea

- [ASUS, 2016] **ASUS**. Disponible en:
http://www.asus.com/mx/Multimedia/Xtion_PRO_LIVE/
Fecha de consulta: 5/10/2016.
- [FESTO, 2016] **FESTO**, Disponible en:
<https://www.festo.com/net/startpage/>
Fecha de consulta: 5/10/2016.
- [KINECT, 2016] **KINECT**. Disponible en:
<http://www.xbox.com/es-MX/kinect>
Fecha de consulta: 1/09/2016.
- [Matlab, 2016] **Matlab**. Disponible en:
<http://www.mathworks.com>
Fecha de consulta: 10/04/2016.
- [OpenCV, 2016] **OpenCV**. Disponible en: <http://opencv.org>. Fecha de consulta: 1/05/2016.
- [OpenNI, 2016] **OpenNI**. Disponible en: <http://www.openni.ru>
Fecha de consulta: 1/05/2016.
- [Eclipse, 2016] **Eclipse** Disponible en: <https://eclipse.org>
Fecha de consulta: 1/05/2016.

ANEXO A - Pruebas

En este apartado se muestran los detalles de cada prueba descrita en la Tabla 4.4.

A continuación se presentan 12 figuras donde se muestran las 12 pruebas.



Frame 1



Frame 2



Frame 3



Frame 4



Frame 5



Frame 6



Frame 7



Frame 8

Prueba 1: El objeto móvil se mueve de izquierda a derecha, con una distancia inicial a la cámara de 3 metros.



Frame 1

Frame 2



Frame 3

Frame 4



Frame 5

Frame 6



Frame 7

Frame 8

Prueba 2: El objeto móvil se mueve de izquierda a derecha, con una distancia inicial a la cámara de 2.5 metros.



Frame 1

Frame 2



Frame 3

Frame 4



Frame 5

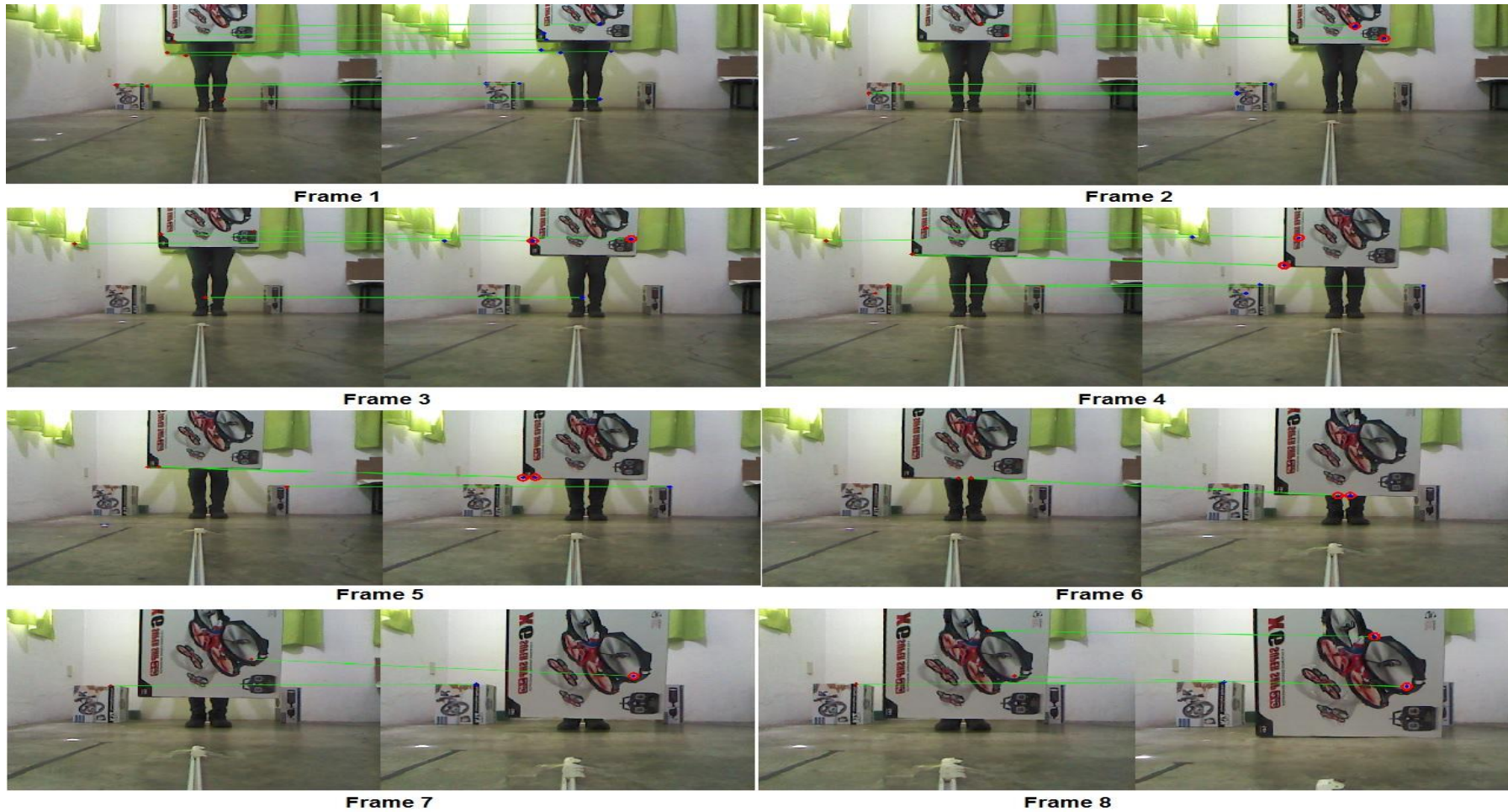
Frame 6



Frame 7

Frame 8

Prueba 3: El objeto móvil se mueve de arriba hacia abajo, con una distancia inicial a la cámara de 3 metros.



Prueba 4: El objeto móvil se mueve de arriba hacia abajo, con una distancia inicial a la cámara de 2.5 metros.



Frame 1

Frame 2



Frame 3

Frame 4



Frame 5

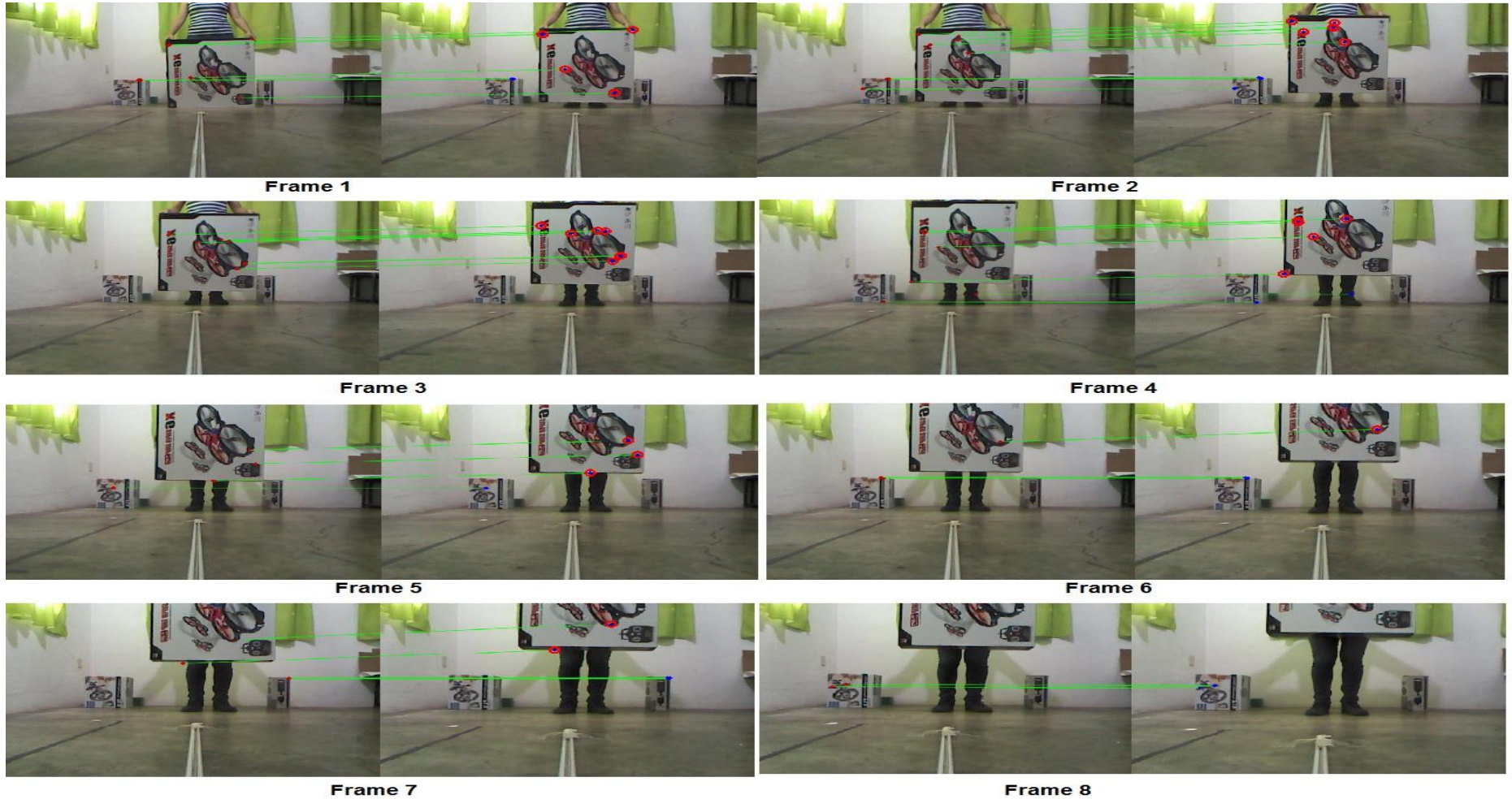
Frame 6



Frame 7

Frame 8

Prueba 5: El objeto móvil se mueve de abajo hacia arriba, con una distancia inicial a la cámara de 3 metros.



Prueba 6: El objeto móvil se mueve de abajo hacia arriba, con una distancia inicial a la cámara de 2.5 metros.



Frame 1

Frame 2



Frame 3

Frame 4



Frame 5

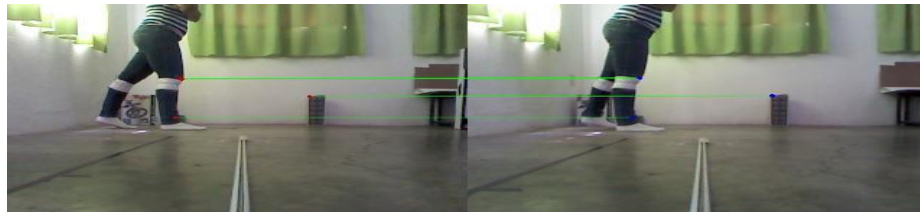
Frame 6



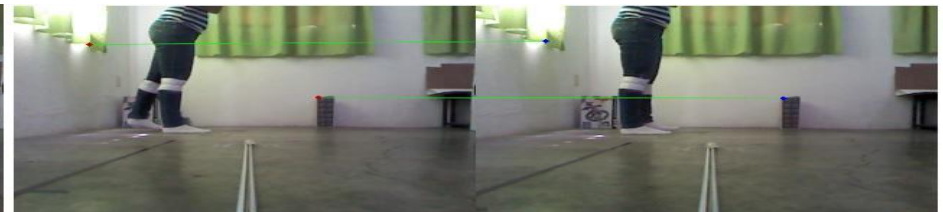
Frame 7

Frame 8

Prueba 7: El objeto móvil se mueve de izquierda a derecha, con una distancia inicial a la cámara de 3 metros.



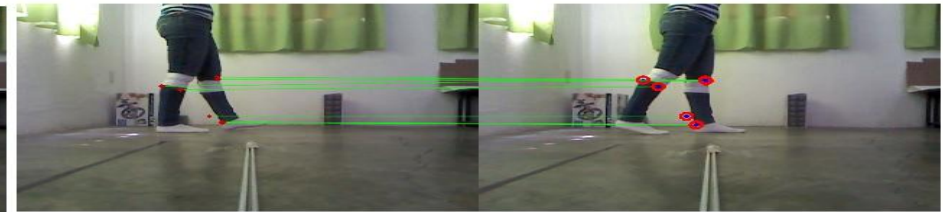
Frame 1



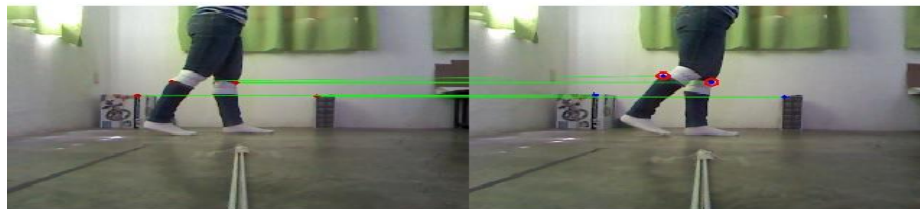
Frame 2



Frame 3



Frame 4



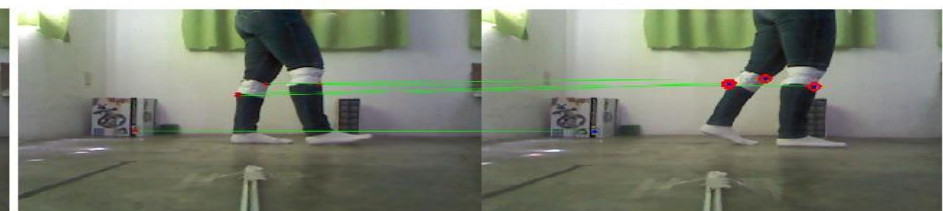
Frame 5



Frame 6

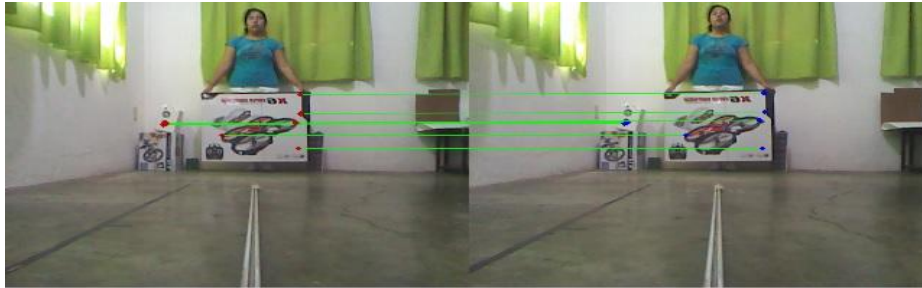


Frame 7



Frame 8

Prueba 8: El objeto móvil se mueve de izquierda a derecha, con una distancia inicial a la cámara de 2.5 metros.



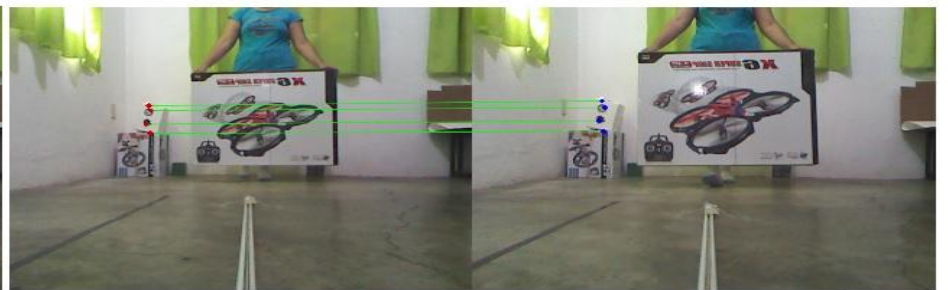
Frame 1



Frame 2



Frame 3



Frame 4



Frame 5



Frame 6

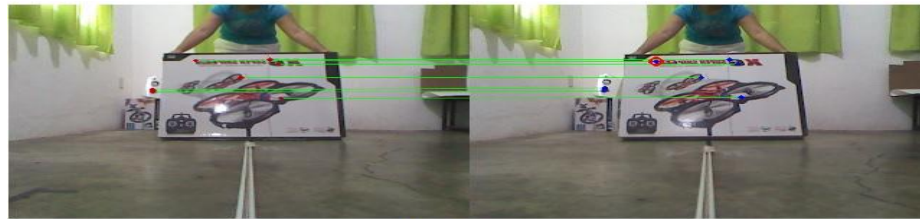
Prueba 9: El objeto móvil se mueve con dirección hacia al frente, con una distancia inicial a la cámara de 3 metros.



Frame 1



Frame 2



Frame 3



Frame 4



Frame 5

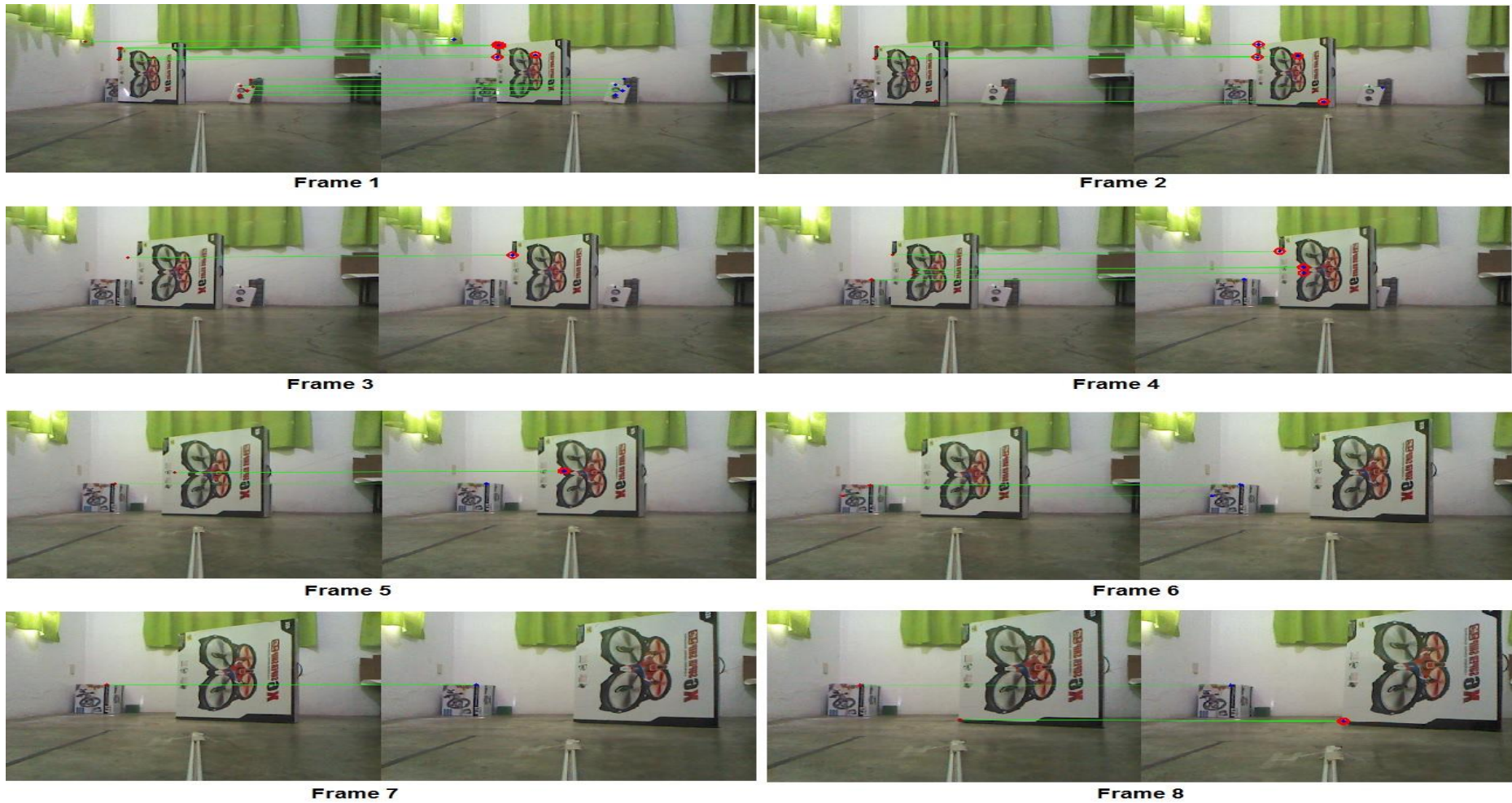


Frame 6



Frame 7

Prueba 10: El objeto móvil se mueve con dirección hacia atrás, con una distancia inicial a la cámara de 2 metros.



Prueba 11: El objeto móvil se mueve con dirección diagonal hacia al frente, con una distancia inicial a la cámara de 3 metros.



Frame 1



Frame 2



Frame 3



Frame 4



Frame 5



Frame 6



Frame 7



Frame 8

Prueba 12: no existen objetos móviles.

A continuación se presentan las Tablas donde se observan los resultados de las 12 prueba descrita en la Tabla 4.3.

Tabla A.1. Resultados Prueba 1

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|--------------|
| 1 | 4 | 2 | 12.5 |
| 2 | 9 | 7 | 12.5 |
| 3 | 8 | 8 | 12.5 |
| 4 | 3 | 3 | 12.5 |
| 5 | 5 | 5 | 12.5 |
| 6 | 7 | 7 | 12.5 |
| 7 | 5 | 5 | 12.5 |
| 8 | 2 | 2 | 12.5 |
| | 43 | 39 | 100 % |

La Tabla A.1 muestra la prueba donde una caja que se mueve de izquierda a derecha a 3 metros la cámara. En cada uno de los *Frames* se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil.

Tabla A.2. Resultados Prueba 2

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|--------------|
| 1 | 2 | 2 | 12.5 |
| 2 | 2 | 2 | 12.5 |
| 3 | 5 | 5 | 12.5 |
| 4 | 10 | 10 | 12.5 |
| 5 | 4 | 3 | 12.5 |
| 6 | 1 | 1 | 12.5 |
| 7 | 4 | 2 | 12.5 |
| 8 | 4 | 1 | 12.5 |
| | 32 | 26 | 100 % |

La Tabla A.2 muestra la prueba donde una caja que se mueve de izquierda a derecha a 2.5 metros la cámara. En cada uno de los *Frames* se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil. Pero se puede apreciar que en el *Frame* 6 y 8 solo se detecto un punto móvil.

Tabla A.3. Resultados Prueba 3

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|--------------|
| 1 | 11 | 6 | 12.5 |
| 2 | 4 | 3 | 12.5 |
| 3 | 9 | 7 | 12.5 |
| 4 | 8 | 5 | 12.5 |
| 5 | 7 | 6 | 12.5 |
| 6 | 5 | 4 | 12.5 |
| 7 | 11 | 11 | 12.5 |
| 8 | 10 | 4 | 12.5 |
| | 65 | 46 | 100 % |

La Tabla A.3 muestra la prueba donde una caja que se mueve de arriba hacia abajo a 3 metros la cámara. En cada uno de los *Frames* se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil.

Tabla A.4. Resultados Prueba 4

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|---------------|
| 1 | 9 | 0 | 0 |
| 2 | 5 | 2 | 12.5 |
| 3 | 4 | 2 | 12.5 |
| 4 | 7 | 3 | 12.5 |
| 5 | 3 | 2 | 12.5 |
| 6 | 2 | 2 | 12.5 |
| 7 | 2 | 1 | 12.5 |
| 8 | 3 | 2 | 12.5 |
| | 35 | 14 | 87.5 % |

La Tabla A.4 muestra la prueba donde una caja que se mueve de arriba hacia abajo a 2.5 metros la cámara. En cada uno de los *Frames* excepto el primero, se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil.

Tabla A.5. Resultados Prueba 5

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|-------------------|---------------|
| 1 | 2 | 0 | 0 |
| 2 | 8 | 6 | 12.5 |
| 3 | 7 | 5 | 12.5 |
| 4 | 9 | 6 | 12.5 |
| 5 | 7 | 3 | 12.5 |
| 6 | 7 | 3 | 12.5 |
| 7 | 2 | 0 | 0 |
| 8 | 2 | 0 | 0 |
| | 44 | 23 | 62.5 % |

La Tabla A.5 muestra la prueba donde una caja que se mueve de abajo hacia arriba a 3 metros la cámara. En cada uno de los *Frames* excepto el primero, séptimo y octavo, se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil.

Tabla A.6. Resultados Prueba 6

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|-------------------|---------------|
| 1 | 8 | 5 | 12.5 |
| 2 | 11 | 6 | 12.5 |
| 3 | 7 | 7 | 12.5 |
| 4 | 7 | 5 | 12.5 |
| 5 | 5 | 3 | 12.5 |
| 6 | 3 | 1 | 12.5 |
| 7 | 5 | 2 | 12.5 |
| 8 | 2 | 0 | 0 |
| | 48 | 29 | 87.5 % |

La Tabla A.6 muestra la prueba donde una caja que se mueve de abajo hacia arriba a 2.5 metros la cámara. En cada uno de los *Frames* excepto el octavo, se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil.

Tabla A.7. Resultados Prueba 7

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|-------------|
| 1 | 2 | 0 | 0 |
| 2 | 7 | 1 | 12.5 |
| 3 | 3 | 2 | 12.5 |
| 4 | 1 | 0 | 0 |
| 5 | 2 | 1 | 12.5 |
| 6 | 7 | 2 | 12.5 |
| 7 | 3 | 1 | 12.5 |
| 8 | 4 | 1 | 12.5 |
| | 29 | 8 | 75 % |

La Tabla A.7 muestra la prueba donde unos pies se mueve de izquierda a derecha a 3 metros la cámara. En cada uno de los *Frames* excepto el primero, y cuarto, se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil.

Tabla A.8. Resultados Prueba 8

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|-------------|
| 1 | 3 | 2 | 12.5 |
| 2 | 4 | 0 | 0 |
| 3 | 2 | 0 | 0 |
| 4 | 4 | 3 | 12.5 |
| 5 | 7 | 7 | 12.5 |
| 6 | 6 | 2 | 12.5 |
| 7 | 3 | 0 | 0 |
| 8 | 1 | 0 | 0 |
| | 30 | 14 | 50 % |

La Tabla A.8 muestra la prueba donde unos pies se mueve de izquierda a derecha a 2.5 metros la cámara. En *Frames* 1, 4, 5 y 6 se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil, en los *Frames* 2, 3, 7 y 8 no se detectaron puntos móviles.

Tabla A.9. Resultados Prueba 9

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|---------------|
| 1 | 7 | 0 | 0 |
| 2 | 3 | 0 | 0 |
| 3 | 3 | 0 | 0 |
| 4 | 4 | 0 | 0 |
| 5 | 3 | 3 | 16.6 |
| 6 | 2 | 2 | 16.6 |
| | 22 | 5 | 33.2 % |

La Tabla A.9 muestra la prueba donde una caja se mueve hacia al frente, con una distancia inicial de la cámara de 3 metros la cámara. En *Frames* 5 y 6 se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil, en los *Frames* 1, 2, 3 y 4 no se detectaron puntos móviles.

Tabla A.10. Resultados Prueba 10

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|--------------|
| 1 | 7 | 4 | 14.2 |
| 2 | 9 | 1 | 14.2 |
| 3 | 7 | 1 | 14.2 |
| 4 | 9 | 5 | 14.2 |
| 5 | 7 | 2 | 14.2 |
| 6 | 3 | 1 | 14.2 |
| 7 | 11 | 4 | 14.2 |
| | 53 | 18 | 100 % |

La Tabla A.10 muestra la prueba donde una caja se mueve hacia atrás, con una distancia inicial de la cámara de 2 metros la cámara. En todos los *Frames* se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil. En la mayoría de los *Frames* se detecto únicamente un punto móvil.

Tabla A.11. Resultados Prueba 11

| Frame | Correspondencia | Puntos Móviles | Móvil % |
|-------|-----------------|----------------|-------------|
| 1 | 12 | 6 | 12.5 |
| 2 | 5 | 4 | 12.5 |
| 3 | 1 | 1 | 12.5 |
| 4 | 5 | 4 | 12.5 |
| 5 | 3 | 2 | 12.5 |
| 6 | 3 | 0 | 0 |
| 7 | 1 | 0 | 0 |
| 8 | 3 | 2 | 12.5 |
| | 33 | 19 | 75 % |

La Tabla A.11 muestra la prueba donde una caja se mueve en diagonal hacia adelante, con una distancia inicial de la cámara de 3 metros la cámara. En *Frames* 1, 2, 3, 4, 5 y 8 se detectaron correspondencias y puntos móviles correspondientes a al objeto móvil, en los *Frames* 6 y 7 no se detectaron puntos móviles.

Tabla A.12. Resultados Prueba 12

| Frame | Correspondencia | Puntos Móviles | No Móvil % |
|-------|-----------------|----------------|--------------|
| 1 | 6 | 0 | 12.5 |
| 2 | 9 | 0 | 12.5 |
| 3 | 16 | 0 | 12.5 |
| 4 | 11 | 0 | 12.5 |
| 5 | 14 | 0 | 12.5 |
| 6 | 8 | 0 | 12.5 |
| 7 | 12 | 0 | 12.5 |
| 8 | 14 | 0 | 12.5 |
| | 90 | 0 | 100 % |

La Tabla A.12 muestra la prueba donde todo es estático y no hay objetos móviles. Ningún *Frame* detecto falsos positivos.

ANEXO B – Algoritmos detectores, descriptores y de correspondencia

Para referirse a los puntos destacados de una imagen se emplean diferentes términos, “características”, “puntos clave”, etc. La detección de rasgos destacados se basa principalmente en la detección de esquinas o manchas.

Las esquinas juegan un papel importante para la visión por computadora en aplicaciones como reconocimiento de objetos, análisis de escenas, correspondencia, etc. Las esquinas son características formadas en las fronteras de dos regiones que presentan gran variación en sus niveles de brillo.

La descripción obtiene de cada punto de interés proporciona información relevante y distintiva de la región que lo rodea, de manera que la misma estructura puede ser reconocida si es encontrada en otra imagen [Guerrero, 2012].

En la actualidad se han propuesto varios algoritmos que permiten llevar a cabo la detección de puntos destacados; en este trabajo se requirió de un algoritmo de este tipo, por lo cual se realizó un análisis de cada uno para encontrar el algoritmo adecuado, a continuación se presentan cuáles fueron los algoritmos analizados:

- SURF (*Speeded Up Robust Features*)
- ORB (*Oriented FAST and Rotated BRIEF*)
- FAST (*Features from Accelerated Segment Test*)
- Detector Harris

Detector y Descriptor SURF (*Speeded Up Robust Features*)

El algoritmo SURF fue desarrollado por Herbert Bay [Bay, 2006] como un detector y descriptor de puntos de interés. El detector de puntos destacados se basa en el uso de la matriz *Hessiana* debido a que se calcula de manera rápida y presenta una buena precisión, dado a estos atributos es llamado el detector “*Fast- Hesse*”.

SURF guarda cierta similitud con la filosofía del descriptor SIFT. Los autores afirman sin embargo que este detector y descriptor presentan principalmente 2 mejoras resumidas en los siguientes conceptos:

- Velocidad de cálculo considerablemente superior sin ocasionar pérdida del rendimiento.
- Mayor robustez ante posibles transformaciones de la imagen.

Estas mejoras se consiguen mediante la reducción de la dimensionalidad y complejidad en el cálculo de los vectores de características de los puntos de interés obtenidos. La normalización o longitud de los vectores de

características de los puntos de interés es considerablemente menor, concretamente se trata de vectores con una dimensionalidad de 64, lo que supone una reducción de la mitad de la longitud del descriptor SIFT. Utiliza el determinante de la matriz *Hessiana* para calcular tanto la posición como la escala de los puntos de interés. El detector se basa en el siguiente principio:

En la Ecuación B.1 se muestra la matriz *Hessiana* para el cálculo de rasgos destacados mediante SURF.

Dado un punto “ $X=(x,y)$ ” en una imagen I , la matriz *Hessiana* $\mathcal{H}(X,\sigma)$ en X a la escala σ .

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix} \quad \text{Ecuación (B.1)}$$

Donde (X) es la convolución de la gaussiana derivativa de segundo orden $\frac{\partial^2}{\partial x^2} g(\sigma)$ con la imagen en el punto X (de manera similar aplica para (X) y $(X\sigma)$). SURF pretende ahorrarse el tiempo de procesamiento mediante el uso de “*filtros-caja (box-filters)*” de tamaño 9x9. Los filtros-caja aproximan las derivadas parciales de segundo orden de las Gaussianas y pueden ser evaluados rápidamente usando imágenes integrales independientemente de su tamaño. Las imágenes integrales se pueden calcular mediante la ecuación B.2, donde $Ii_{\Sigma}(x,y)$ es la intensidad de la imagen en un punto x, y .

$$Ii_{\Sigma}(x, y) = \sum_{i=1}^{i \leq x} \sum_{j=1}^{j \leq y} I(i, j) \quad \text{Ecuación (B.2)}$$

Una vez creada la imagen integral, se puede calcular la suma de las intensidades de una región mediante la ecuación B.3.

$$\sum I = Ii_D + Ii_A + Ii_B + Ii_C \quad \text{Ecuación (B.3)}$$

De esta forma, el tiempo necesario para el cálculo de las operaciones de convolución es independiente del tamaño de la imagen, lo que facilita el ahorro del mismo. Las aproximaciones de las derivadas parciales se denotan como D_{xx} , D_{xy} , y D_{yy} . En cuanto a la determinante de la matriz *Hessiana*, queda definida como muestra la ecuación B.4.

$$\det(H_{aprox.}) = D_{xx}D_{yy} - (0,9D_{xy})^2 \quad \text{Ecuación (B.4)}$$

El valor 0.9 de la ecuación 12 anterior es propuesto por Bay y está relacionado con la

aproximación del filtro gaussiano. El descriptor SURF, al igual que en el caso del descriptor SIFT, está dividido en octavas. Las capas de las octavas se obtienen mediante el filtrado de la imagen con máscaras gradualmente más grandes (9x9, 15x15, 21x21, 27x27).

Finalmente para calcular la localización de todos los puntos de interés en todas las escalas, se procede mediante la eliminación de los puntos que no cumplan la condición de máximo en un vecindario de 3x3. De esta manera, el máximo determinante de la matriz *Hessiana* es interpolado en la escala y posición de la imagen.

Después de haber obtenido la escala, el primer paso es obtener la orientación del punto de interés para posteriormente obtener el descriptor SURF. Para obtener un punto invariante a la orientación se aplica “*Haar-wavelet*” para las direcciones *x* e *y* en una región circular de radio $6s$, donde *s* es la escala del punto de interés.

Una vez obtenida la orientación para todos los vecinos, se estima la orientación dominante sumando todos los resultados dentro de una ventana deslizante que cubre un ángulo de $\pi/3$. El descriptor se realiza construyendo primeramente una región cuadrada centrada en el punto de interés y con un tamaño de $20s$. La región se divide regularmente en 4 sub-regiones y para cada sub-región se procesan pocas características simples. Seguidamente se aplica “*Haar-wavelet*” para *x* e *y* y se suavizan los resultados mediante una gaussiana, obteniendo *dx* y *dy*. Para cada sub-región se suman los resultados *dx* y *dy*, además de obtener su valor absoluto $|dx|$ y $|dy|$. De este modo, cada sub-región proporciona un vector *v* compuesto por: $v = (\sum dx, \sum dy, \sum |dx|, \sum |dy|)$ el cual presenta un tamaño de 64 elementos. El descriptor SURF se obtiene mediante la unión de los vectores de las sub-regiones, en la Figura B.1

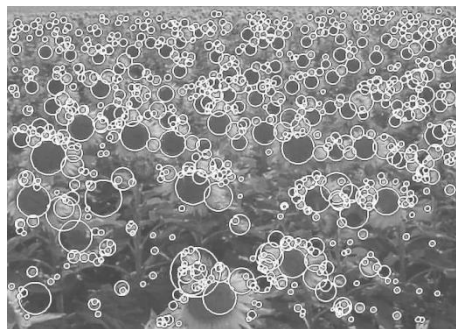


Figura B.1. Puntos de interés detectados en un campo de girasol[Bay, 2006]

ORB (Oriented FAST and Rotated BRIEF)

ORB es básicamente una fusión de detector de puntos destacados FAST y descriptor BRIEF con muchas modificaciones para mejorar el rendimiento. ORB se basa en el detector de esquinas de *Harris* para encontrar los puntos destacados. El descriptor emplea una vecindad de 31 píxeles para llevar a cabo el proceso de descripción del punto clave.

El algoritmo ORB fue propuesto por Ethan Rublee [Rublee, 2011] como una mejor alternativa a los dos algoritmos referentes para la detección y descripción de rasgos destacados (*SIFT* y *SURF*). Rublee enfatiza sobre la eficiencia del detector de puntos destacados *FAST* por lo que implementa una variación del detector *FAST* en su algoritmo y lo llama *FAST-9*. La única diferencia entre *FAST* y *FAST-9* es el radio del círculo de píxeles vecinos que contempla para el análisis de rasgos destacados, siendo 9 en comparación con los 16 propuestos por Rosten en *FAST* [Rosten, 2006].

El descriptor ORB es un descriptor binario, la longitud del vector es de 32 y de 8 bits cada registro (a diferencia de los demás mencionados en el presente trabajo) ya que emplea el descriptor BRIEF el cual aplica una evaluación binaria en un parche de la imagen.

Considere un parche p en una imagen, la evaluación binaria *BRIEF* τ es definida por la ecuación B.5.

$$\tau(\mathbf{p}; \mathbf{x}, \mathbf{y}) := \begin{cases} 1 & : \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & : \mathbf{p}(\mathbf{x}) \geq \mathbf{p}(\mathbf{y}) \end{cases} \quad \text{Ecuación (B.5)}$$

Donde $p(x)$ es la intensidad del parche p en el punto x . La descripción del punto destacado es definida como un vector de n evaluaciones binarias, como se muestra en la ecuación B.6.

$$f_n(\mathbf{p}) := \sum_{1 \leq i \leq n} 2^{i-1} \tau(\mathbf{p}; \mathbf{x}_i, \mathbf{y}_i) \quad \text{Ecuación (B.6)}$$

Cada valor n del vector resultante puede tomar un valor de entre 0 y 255 (debido a las intensidades de los valores RGB).

FAST (Features from Accelerated Segment Test)

FAST es un algoritmo propuesto por Edward Rosten en el trabajo de investigación titulado “*Machine learning for high-speed corner detection k* [Rosten, 2006]” con la finalidad de identificar rasgos destacados en una imagen. De acuerdo con Rosten un rasgo destacado en una imagen es un píxel que tiene una posición bien definida y puede ser detectada con firmeza. Dichos

rasgos tienen un alto contenido de información local y deben ser idealmente repetibles entre diferentes imágenes.

La prueba de FAST para encontrar esquinas de objetos en una imagen digital inicia considerando un círculo de 16 píxeles de diámetro alrededor de un píxel candidato " p " Figura B.2 y Figura B.3. El algoritmo clasifica a p como una esquina, si existe un conjunto de n píxeles continuos en el círculo que sean más brillantes que el píxel p " lp " más un umbral " t " ($lp+t$) o todos son más oscuros que ($lp-t$).

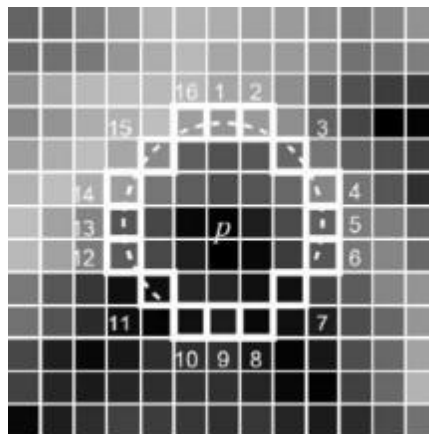


Figura B.2 Pixel "p" [Rosten, 2006]

Rosten propuso un valor de 12 para n debido a que permite una evaluación rápida con la que se pretenden eliminar píxeles que no son esquinas. La evaluación examina solamente los píxeles (1, 5, 9 y 13) y presenta la siguiente lógica: "Si p es una esquina, entonces al menos tres píxeles deben ser más brillantes que $lp+t$ o más oscuros que $lp-t$. Si ninguno de los casos ocurre, entonces p no debería ser una esquina".

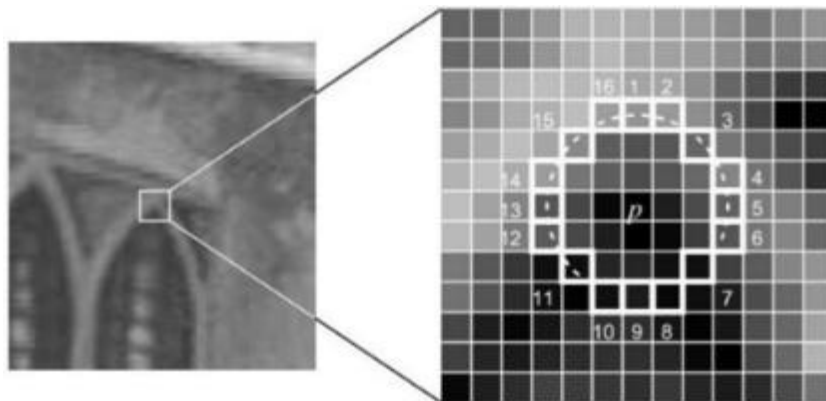


Figura B.3 Detectado como esquina mediante FAST [Rosten, 2006]

Detector *Harris*

El detector de esquina *Harris* es un detector de puntos de interés popular debido a su fuerte invariancia a la rotación, la escala, la variación de la iluminación y el ruido de la imagen [Derpanis, 2004].

El detector de esquina *Harris* se basa en la función de auto-correlación local de una señal; donde la función de auto-correlación local, mide los cambios locales de la señal con parches desplazado por una pequeña cantidad en diferentes direcciones.

Teniendo en cuenta un cambio $(\Delta x, \Delta y)$ y un punto (x, y) , la función de auto-correlación es definida como se muestra en la ecuación B.7. En la Figura B. 4 se muestra una imagen con puntos detectados con el algoritmo *Harris*.

$$c(x, y) = \sum_W [I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y)]^2 \text{ Ecuación (B.7)}$$

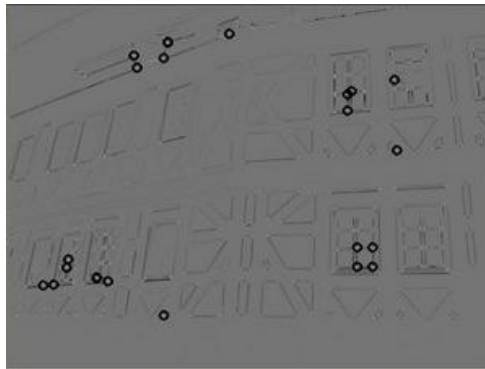


Figura. B.4 Detección de puntos característicos con detector Harris

Descriptor propio

Se desarrolló un descriptor propio para probar con los detectores de puntos destacados *Harris*, ORB, SURF y FAST. El descriptor consiste en un vector de 11 datos, los primeros dos datos del vector son la posición x e y del punto que se describe en la imagen, el tercer dato es la intensidad en escala de grises (valores de 0 a 255) del punto que se describe, los ocho datos restantes son la intensidad en escala de grises de los pixeles vecinos como se muestra en la Figura B.3.

Se realizó este descriptor para tener un manejo más fácil del algoritmo de correspondencia propio, ya que existen descriptores que son difíciles de acoplar otros algoritmos de correspondencia para los que no fueron creados, por tal razón se creó un descriptor que pudiera ser usado con el algoritmo de correspondencia propio.

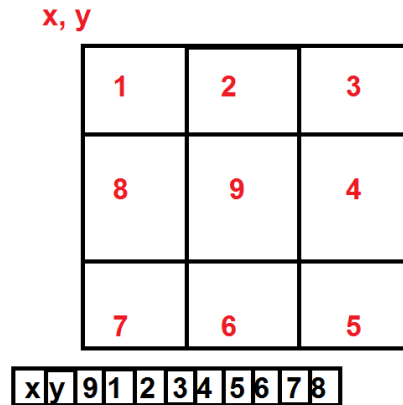


Figura B.3 Vector de características

Algoritmos de correspondencias

Una vez que se han extraído las características de las imágenes se tienen que aplicar otras técnicas para encontrar correspondencias entre ambas imágenes. Basándose en la forma del descriptor, el proceso de correspondencia puede dividirse en dos categorías, correspondencia de vectores de descripción (*BruteForceMatching*, FLANN, etc.) y correspondencia de descriptores binarios BRIEF (*Hamming Distance*), siendo generalmente la correspondencia de descriptores binarios más rápida.

Algoritmo de correspondencia propio

Se desarrolló un algoritmo de correspondencia en C++ para verificar cuál detector de puntos destacados da un mejor resultado y para probar qué tan eficaz es el algoritmo de correspondencia creado.

El algoritmo de correspondencia entre dos imágenes toma la descripción de los puntos de las dos imágenes (Figura B.4), para que el proceso computacional sea más ágil se usa la distancia Manhattan para no comparar todos los vectores de características de la imagen 1 con los de la imagen 2; para comparar que tan lejos está un punto del otro; se usa un umbral de 15 píxeles, si la distancia Manhattan entre dos puntos supera los 15 píxeles, esos dos vectores no son candidatos a ser correspondientes, pero si la distancia Manhattan entre dos puntos es menor o igual a 15 píxeles esto quiere decir que pueden ser puntos correspondientes y son comparados. Se propuso 15 píxeles como umbral porque eso significa que un punto no se mueve más de 7 píxeles en su eje x o en su eje y por lo tanto se encuentra dentro de un radio lógico según el movimiento de la imagen. La distancia Manhattan se muestra en la ecuación B.8.

$$|X_t - X_{t+1}| + |Y_t - Y_{t+1}| \quad \text{Ecuación (B.8)}$$

Posteriormente los puntos que tengan distancia Manhattan menor o igual a 15, se comparan realizando una resta absoluta, como muestra la ecuación B.9.

$$\Delta Z = \sum |Z1_i - Z2_i| \quad \text{Ecuación (B.9)}$$

Vector Z_1

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| x | y | 9 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|

Vector Z_2

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| x | y | 9 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|

Figura B.4 Vector Z_1 de un punto de la imagen 1 y Vector Z_2 de un punto de la imagen 2

Aquellos ΔZ cuyo resultado de la comparación de los dos vectores sea menor a 15, son puntos correspondientes. Si un punto que se encuentra en la parte superior es ilógico que en el siguiente *Frame* se encuentre en la parte inferior, ya que si el movimiento entre un *Frame* y otro es uniforme, el punto debería encontrarse en la parte superior adentro de un radio específico de distancia.