EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Tecnológico Nacional de México

Centro Nacional de Investigación y Desarrollo Tecnológico

Tesis de Maestría

Navegación con asociación para robot explorador con visión y unidad de medición inercial

presentada por

Ing. Ariel Eliezer Vazquez Dominguez

como requisito para la obtención del grado de

Maestría en Ciencias Computacionales

Director de tesis

Dr. José Ruiz Asencio

Cuernavaca, Morelos, México. Diciembre de 2019.

cenidet
Centro Nacional de Investigación y Desarrollo Tecnológico

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Centro Nacional de Investigación y Desarrollo Tecnológico

"2019, Año del Caudillo del Sur, Emiliano Zapata"

Cuernavaca, Mor., 12/noviembre/2019

OFICIO No. DCC/157/2019
Asunto: Aceptación de documento de tesis
CENIDET-AC-004-M14-OFICIO

DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO
PRESENTE

Por este conducto, los integrantes de Comité Tutorial del **Ing. Ariel Eliezer Vazquez Dominguez**, con número de control M17CE104, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis de grado titulado **"Navegación con asociación para robot explorador con visión y unidad de medición inercial"** y hemos encontrado que se han atendido todas las observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

Dr. José Ruiz Ascencio
Doctor en Ciencias
5009035
Director de tesis

Dr. Raúl Pinto Elías
Doctor en Ciencias en la Especialidad de Ingeniería Eléctrica
3890453
Revisor 1

Dr. Andrés Blanco Ortega
Doctor en Ciencias en Ingeniería Eléctrica
6559298
Revisor 2

C.c.p. Depto. Servicios Escolares.
Expediente / Estudiante

JGGS/lmz

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Centro Nacional de Investigación y Desarrollo Tecnológico
Subdirección Académica

"2019, Año del Caudillo del Sur, Emiliano Zapata"

Cuernavaca, Mor., 12/diciembre/2019
No. de Oficio: SAC/362/2019
Asunto: Autorización de impresión de Tesis

ING. ARIEL ELIEZER VAZQUEZ DOMINGUEZ
CANDIDATO AL GRADO DE MAESTRO EN CIENCIAS DE LA COMPUTACIÓN
P R E S E N T E

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado **"Navegación con asociación para robot explorador con visión y unidad de medición inercial"**, 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.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
"Conocimiento y tecnología al servicio de México"

SEP TecNM
CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO
SUBDIRECCIÓN ACADÉMICA

DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO

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

GVGR/ego

cenidet
Interior Internado Palmira S/N, Col. Palmira, C. P. 62490, Cuernavaca, Morelos.
Tel. (01) 777 3 62 77 70, ext. 4104, e-mail: acad_cenidet@tecnm.mx
www.tecnm.mx | www.cenidet.edu.mx

Dedicatoria

El presente trabajo va dedicado con mucho cariño a mi familia la cual me ha apoyado desde mi niñez hasta el momento. Su amor, cariño, comprensión, motivación y paciencia han sido importantes en el transcurso de mi carrera profesional. Realizando un énfasis especial a mi abuelo Tomas Dominguez quien este año partió.

Mi empeño y dedicación se debe a ellos ya que su motivación me ha hecho persistente y nunca dejar mis sueños y persistir en lo que quiero realizar.

Agradecimientos

En primer lugar, Agradecen al CONACYT, al Tecnológico Nacional de México y al Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET).

Doy las gracias a los profesores y personal del CENIDET que contribuyeron a mi formación como Maestro en Ciencias, especialmente a mi director el Dr. José Ruiz Ascencio, por su tiempo, dedicación, consejos y enseñanzas. Así mismo agradezco a mis revisores el Dr. Raúl Pinto Elías y el Dr. Andrés Blanco Ortega por sus enseñanzas, tiempo, consejos y críticas. Así mismo, agradezco a la oficina de Servicios estudiantiles por su tiempo.

Finalmente agradezco a todo el departamento de ciencias de la computación, en especial al área de inteligencia artificial.

Resumen

Este trabajo se centra en la asociación de lugares o también conocido como cerradura de ciclos, que tiene como finalidad reducir el error acumulativo generado por errores en la medición, procesamiento, etc., del recorrido de un robot.

La asociación ya es un problema muy estudiado por un amplio catálogo de investigadores, algunos dando resultados muy buenos, sin embargo, un problema que se tiene es que conforme va creciendo la cantidad de imágenes procesadas van creciendo también los tiempos de consulta de la base de datos, para aminorar este problema se planteó el uso de una unidad de medición inercial (IMU por sus siglas en inglés) utilizando en específico su magnetómetro para capturar la orientación con la cual fue obtenida la imagen de entrada y solo consultar una parte de la base de datos (lo que se hizo fue dividir toda la base de datos en varias dependiendo la cantidad de orientaciones que se utilizan).

Esta técnica obtuvo buenos resultados, reduciendo los tiempos de consulta hasta en un 30% del tiempo de una sola base de datos para las últimas imágenes y un 62% en promedio.

Abstract

This work is centered on place association, also referred to as loop closure, which has the intention of reducing cumulative error generated by measurement, processing, etc. of a mobile robot's visual register.

Place association has been intensively studied by a broad group of researchers, some with very good results, yet a persistent problem is the growth of data base response time as the number of processed images grows. To mitigate this problem, the use of an inertial measurement unit is proposed, specifically the use of its magnetometer to capture the orientation of each input image, and thus only query part of the data base. The data base was partitioned depending on the number of orientations used.

This technique achieved good results, reducing response times of a single partition data base by 30% for the final images and by 62% on the average over all images.

Índice de contenido

Índice de Figuras

Índice de Tablas

Acrónimos

BOVW	Bag of visual Words
CENIDET	Centro Nacional de Investigación y Desarrollo Tecnológico
IMU	Unidad de medición inercial
SLAM	Simultaneous localization and mapping
VO	Odometría visual
SIFT	Scale-invariant feature transform
SURF	Speeded-Up Robust Features
ORB	*Oriented FAST and Rotated BRIEF*
DOG	Diferencia de gaussianas
SVM	Máquinas de soporte vectorial

Definiciones

Asociación de lugares o cerradura de ciclos:

El cierre de ciclos se refiere al proceso de afirmar correctamente que un robot móvil ha regresado a una ubicación visitada anteriormente [1].

Odometría:

Estudio de la estimación de la posición de vehículos con ruedas durante la navegación. Se basa en la información sobre la rotación de las ruedas para estimar cambios en la posición a lo largo del tiempo [2].

Odometría Visual:

Proceso mediante el cual se determina la posición y la orientación de una cámara o de un sistema de cámaras mediante el análisis de una secuencia de imágenes adquiridas, sin ningún conocimiento previo del entorno [3].

1. Capítulo 1 Introducción

1.1 Motivación

Cuando un robot atraviesa un entorno, el desafió de darse cuenta si ya había sido visitado se llama detección de cierre de ciclos [4] o también llamado asociación de lugares. Con la asociación se pueden resolver varios problemas dentro del mundo de la robótica autónoma, uno de los ejemplos más populares es el del robot secuestrado, el cual consiste en tomar un robot con odometría visual, cargarlo con la cámara tapada y llevarlo a otra habitación y si ya había estado en ese lugar entonces puede estimar una posición. Otro uso es el que menciona el autor Khan [5] en el dominio de la robótica autónoma, se requiere la detección de cierre de bucle en SLAM para construir un mapa topológico o métrico consistente.

Para lograr la asociación de lugar se planteó el uso de una Bolsa de palabras visuales (“Bovw” por sus siglas en inglés), el cual detecta correctamente las cerraduras de ciclos, sin embargo, Bovw tiene un problema, es que conforme se van agregando más imágenes a la base de datos también van creciendo los tiempos en milisegundos de la consulta de esta, como se puede observar en la Tabla 1.1.

Para aminorar este problema se planteó el uso de una unidad de medición inercial (“IMU” por sus siglas en inglés) con la tarea de asignar a cada nueva imagen, la orientación con la que fue adquirida, además de dividir la base de datos en 8 bases de datos (utilizando la orientación) y solo consultar en las que sean necesarias dependiendo de la orientación actual.

Tabla 1.1 Tiempos de consulta

Correspondencias	Promedio	σ	Min	Max
Fast loop-closure detection using visual-word-vectors from image sequences [6]	4.87	7.72	0.66	12.47
Detección de Cierre de Ciclos en el Problema de Localización y Mapeo Simultáneo Visual Monocular presentada [7]	282.85	223.962	0.008	945.145
Real-time loop detection with bags of binary words [4]	0.82	1.29	0.25	5.79
Bags of binary words for fast place recognition in image sequences [8]	1.60	2.64	0.61	18.55
Detección y cierre de ciclos en sistemas SLAM basados en visión estéreo [9]	384,9	317	42	922

1.2 Delimitación del problema especifico

Independientemente del tipo de odometría que se esté utilizando, esta tiende a acumular error, un ejemplo es un robot con encoder colocado en el eje de sus llantas para calcular el desplazamiento, sus llantas de plástico y se mueve en un piso muy resbaloso, a este robot se le programa para moverse un metro en línea recta y se mide físicamente el desplazamiento realizado, muy difícilmente va a medir exactamente un metro ya que con estas condiciones las llantas tienden a patinar generando error, a esto se le suma el error propio del sensor. Ahora a este robot se le pone a generar un mapa como el de la Figura 1.1 A. Sin embargo, debido al error acumulativo genera un mapa como la Figura 1.1 B, este problema se soluciona con la asociación o cerradura de ciclos.

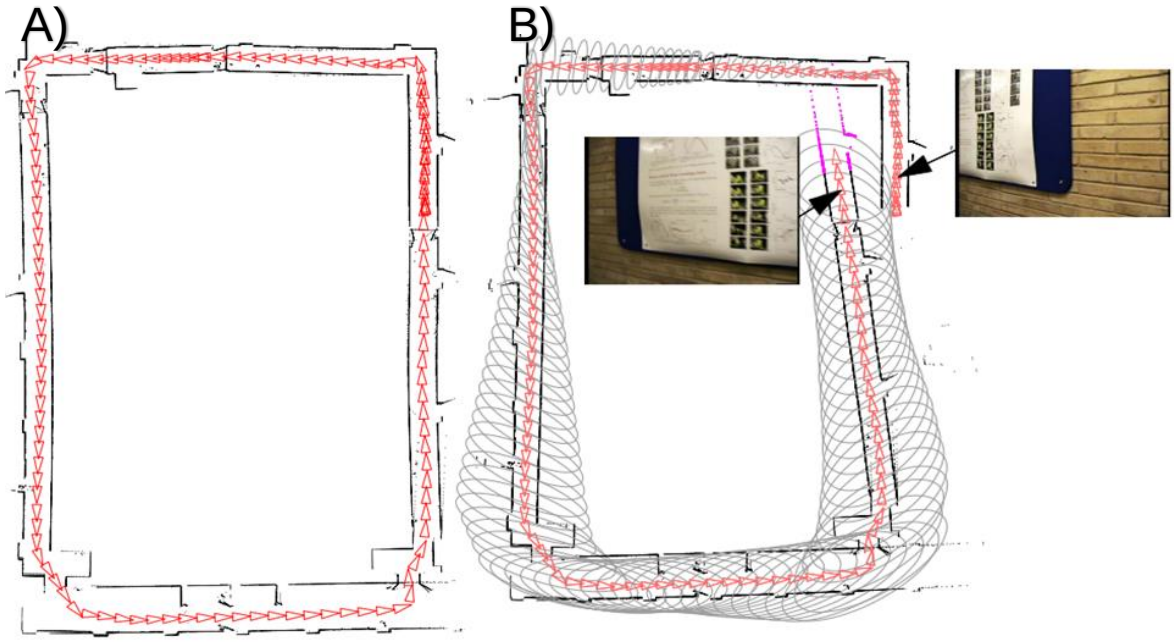

Figura 1.1 A) mapa original B) mapa creado sin asociación [10]

La detección de cierre de ciclos es de gran importancia, ya que permite detectar cuando un robot ya estuvo un lugar para así corregir su posición, disminuyendo el error acumulado. Es un método que está compuesto por el proceso de extracción de datos y de asociación de datos.

Para el proceso de detección y descripción de puntos destacados en una escena existen muchos algoritmos, por ejemplo SIFT(*Scale-invariant feature transform*) [11], SURF(*Speeded-Up Robust Features)* [12] o ORB [13] cada uno proporciona una representación diferente de la misma imagen, dando un vector de características, SIFT da un vector de 128 caracteres para cada punto importante en la imagen. Puede parecer una tarea simple, pero decidir cuál emplear es lo complicado, alguno puede ser mejor que los demás en una característica, pero a costa de otra, por ejemplo, utilizar un algoritmo que sea robusto pero que sea lento, no podrá ser utilizado para aplicaciones en tiempo real o utilizar un algoritmo que sea rápido, pero a cambio de su robustez, permitiría utilizarlo en tiempo real pero no sería muy preciso.

La asociación de datos es la encargada de decidir si el robot ya había estado en este lugar. El problema de esta operación está dado por la base de datos de escenas que aumenta con la longitud del recorrido, de esta forma crecen los tiempos de consulta ya que la tiene que revisar completa cada vez que cambia la escena. La importancia de saber cuándo revisar la base de datos radica en que, si la base de datos ya tiene muchos elementos, revisarla constantemente costaría mucho poder computacional.

Cómo se menciona en la tesis Comparación de Algoritmos de Extracción y Asociación de Rasgos Para Visión Robótica [14], existen varios enfoques para la detección y la asociación, elegir el mejor para cierta condición no es tarea fácil. Ahí propone un sistema para comparación de extractores y asociadores, pero ninguno lo hace en tiempo real.

Por esta razón en esta tesis se aborda el problema de detección de cierres de ciclos en tiempo real. El resultado será un sistema capaz de extraer información y lograr la asociación de datos en un tiempo que se pueda considerar tiempo real.

1.3 Beneficios esperados

Existen dificultades que superar en la visión robótica en general y en las aplicaciones de SLAM, en particular. Entre ellas, la detección de cierre de ciclos, lo cual se refiere a el desafío de observar si un lugar ya ha sido visitado cuando un robot móvil atraviesa un entorno [4].

De acuerdo con lo revisado en la literatura se puede concluir que la detección de cierre de ciclos en tiempo real es un punto clave por resolver, ya que, disminuyendo los tiempos de procesamiento y la carga computacional, permite al robot solucionar problemas como el del robot secuestrado, el cual consiste en recuperar la posición, después de un movimiento a ciegas o externo (cuando se mueve manualmente al robot), esto puede ocurrir por un fallo de la cámara o una oclusión.

En el cenidet se han desarrollado proyectos de tesis que logran la asociación de lugares fuera de línea, sin embargo, no se ha podido implementar en tiempo real.

Con el desarrollo de esta tesis se cuenta con un sistema de detección de cierres de ciclos, que permite:

1. Detectar puntos de interés en imágenes.
2. Detectar lugares ya visitados.
3. Cuenta con criterios para saber dónde y cuándo extraer y asociar.
4. En tiempo real

1.4 Organización de la tesis

Con el fin de orientar al lector acerca del contenido de la presente tesis, se enumeran y describen brevemente las partes constitutivas: el Capítulo 2 contiene el estudio del estado del arte; el Capítulo 3 contiene la metodología de solución, detalla la implementación de la modificación a BOVW; el Capítulo 4 contiene distintos experimentos con la modificación implementada y los resultados. Finalmente, las conclusiones obtenidas y el trabajo futuro son mencionados en el Capítulo 5.

2. Capítulo 2 Contexto Teórico / Práctico

Se describirán algunos conceptos necesarios para la comprensión de la terminología empleada en esta tesis. Además, se mencionarán algunos trabajos realizados en el CENIDET que sirven como antecedentes para este trabajo y publicaciones de trabajos referentes al tema.

2.1 Marco conceptual

Para el desarrollo de este trabajo se requiere de ciertos conocimientos, algunos conceptos importantes se describen a continuación.

2.1.1 Cierre de ciclos o asociación

Cuando un robot móvil atraviesa un entorno, el desafío de observar si un lugar ya ha sido visitado se denomina detección de cierre de ciclos [4], otro nombre que recibe es Asociación. La detección de cierre de ciclos es un problema importante para cualquier sistema de SLAM (del inglés "*simultaneous localization and mapping*"), y dado que las cámaras se han convertido en un sensor común en aplicaciones de robótica móvil, cada vez más personas recurren a la visión artificial para lograrla. Según el artículo de comparación de métodos de cierre de ciclos [15], existen 3 diferentes tipos:

Map-to-map: Se buscan correspondencias entre las características en dos sub-mapas teniendo en cuenta tanto su apariencia como sus posiciones relativas.

Image-to-image: Se buscan correspondencias entre la última imagen de la cámara y las imágenes vistas anteriormente. Este es el método que se utiliza en esta tesis.

Image-to-map: Se buscan correspondencias entre el último fotograma de la cámara y las características del mapa.

2.1.2 Mapas

En la robótica móvil se habla, principalmente, de dos tipos de mapas para la navegación autónoma de robots. Estos tipos de mapas son los mapas geométricos y los mapas topológicos [16].

Mapa topológico:

Una manera de representar el espacio en el que un robot estará ejecutando sus tareas es a través del uso de grafos, de hecho, los mapas topológicos son representaciones espaciales en forma de grafos en los cuales los nodos generalmente representan estados en los que se encuentra el robot y las aristas representan trayectorias que llevan al robot de un estado a otro [16]. Además, como no registran el espacio vacío, los mapas topológicos tienden a ser más compactos que los mapas métricos. Los seres humanos utilizamos básicamente mapas topológicos [17].

En la Figura 2.1 se muestra un ejemplo de mapa topológico, en el mapa se puede ver la ruta que siguió un robot, comenzando en el nodo A y terminando en el nodo J.

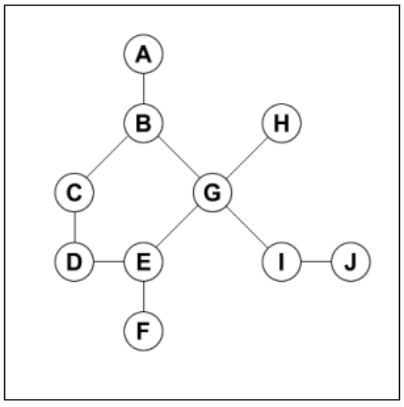

Figura 2.1 Ejemplo de mapa topológico [17].

Mapa métrico

Estos tipos de mapas son representaciones exactas del mundo. Esto es precisamente lo que la mayoría de las personas piensan de mapa, un directorio de calles o un plano de una planta. Estos mapas se pueden dibujar a escala y tienen una orientación específica. En la Figura 2.2 se muestra el mismo mapa de la Figura 2.1 pero en representación métrica [17], comenzando en el punto A, siguiendo la ruta de B,C,D,E,F,G,H,I y terminando en J. En comparación con el mapa en la Figura 2.1 este mapa es mucho más fácil de entender e interpretar por un ser humano.

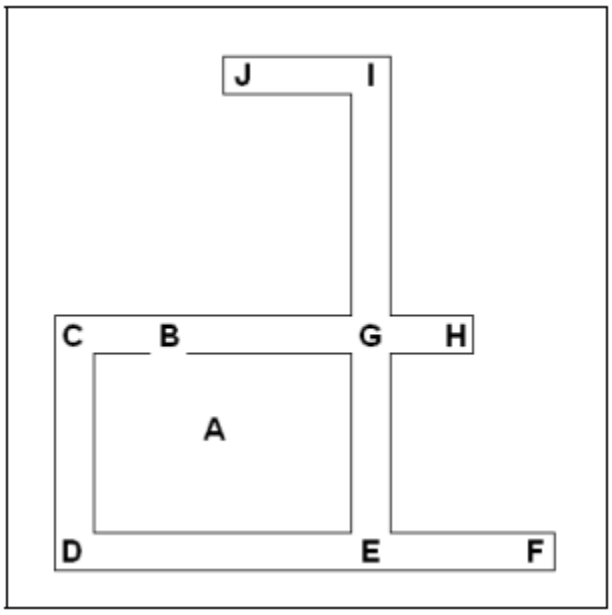

Figura 2.2 Mapa métrico [17]

2.1.3 Odometría

Una determinación precisa de la ubicación es un requisito fundamental cuando se trata de problemas de control de robots móviles. La localización en robótica móvil significa deducir la posición de un robot comparando la información sensorial con

un mapa del entorno [18]. La atención al problema de la odometría ha ido en crecimiento, ya que es muy importante tener una buena odometría, debido a esto se han propuesto varias técnicas para solucionarlo, como por ejemplo GPS, basada en medidas internas (encoder), odometría visual, etc.

Estas técnicas varían dependiendo del tipo de propósito a realizar, los sensores disponibles, el tipo y el conocimiento del entorno en el que va a navegar el robot [19]. En la Figura 2.3 se puede ver un cuadro resumen de los distintos sistemas de estimación de la posición.

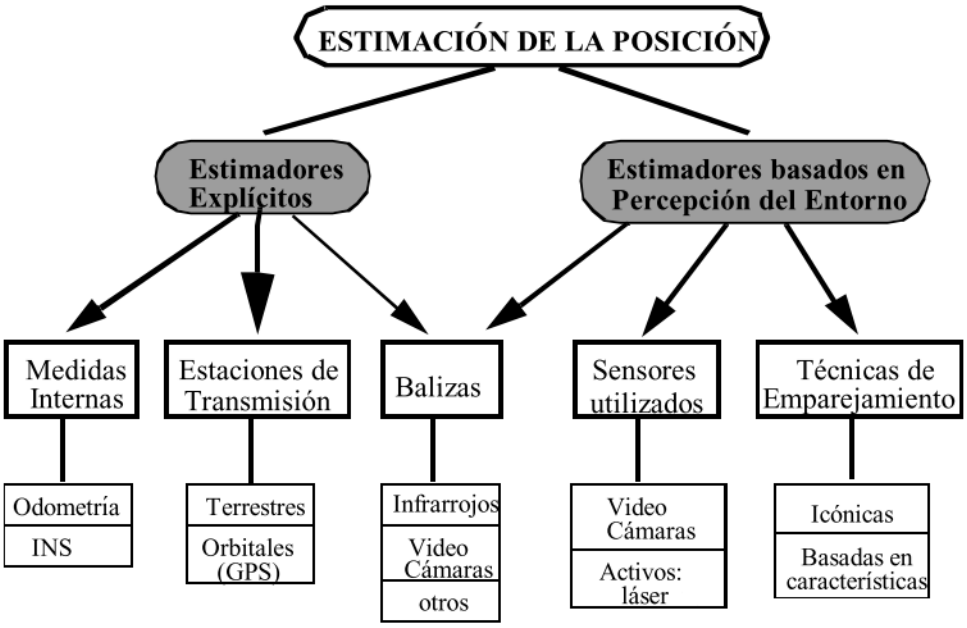

Figura 2.3 Resumen de los distintos sistemas de estimación de la posición [19].

2.1.4 Unidad de medición inercial (IMU)

La navegación inercial es una técnica de navegación autónoma en la que las mediciones proporcionadas por acelerómetros y giroscopios se utilizan para rastrear la posición y la orientación de un objeto en relación con un punto de partida, orientación y velocidad conocidos. Las unidades de medición inercial (IMU)

típicamente contienen tres giroscopios ortogonales de velocidad angular y tres acelerómetros ortogonales que miden la aceleración lineal. Al procesar las señales de estos dispositivos, es posible rastrear la posición y la orientación de un dispositivo. La navegación inercial se utiliza en una amplia gama de aplicaciones, incluida la navegación de aeronaves, misiles tácticos y estratégicos, naves espaciales, submarinos y barcos [20].

Falta mencionar las diferencias con la odometría visual y la mecánica, la IMU no necesita información del ambiente, es un sensor propioceptivo, es decir puede censar, aunque no tenga idea de cuál es su entorno, puede medir su desplazamiento con respecto a una posición inicial.

En la Figura 2.4 se puede observar un esquema de los sentidos de funcionamiento del giroscopio y el acelerómetro de la IMU (hacia donde sus volares son positivos).

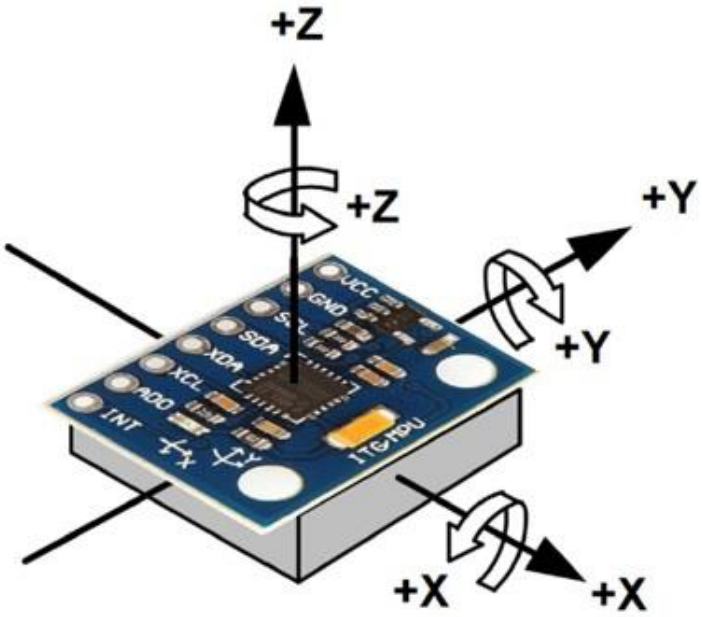

Figura 2.4 IMU [21]

2.1.5 Odometría Visual

Una aplicación importante de la visión por computadora es la navegación autónoma de vehículos y robots. El uso efectivo de sensores de video para la detección de obstáculos y la navegación ha sido un objetivo en la robótica de vehículos terrestres durante muchos años [22]. La odometría visual (VO) es un método para estimar la posición de una cámara a partir de una secuencia de imágenes. En VO, los cuadros de imagen consecutivos en una secuencia se combinan para la correspondencia y las poses relativas entre los cuadros se acumulan. Esto estima el camino recorrido con hasta seis grados de libertad (DOF) [23].

2.1.6 Características Locales

En los últimos años, las características locales (local features, en inglés) han mostrado ser muy efectivas en la búsqueda de características o rasgos distintivos entre diferentes vistas de un escenario. La idea tradicional de estos métodos es primero detectar estructuras o puntos significativos en la imagen y obtener una descripción discriminante de estas estructuras a partir de sus alrededores, la cual sería utilizada luego para la comparación usando una medida de similitud entre estos descriptores [24].

Una característica local es un patrón de imagen que difiere de su vecindad inmediata [25]. Generalmente se asocia con un cambio de una propiedad de la imagen o varias propiedades simultáneamente, aunque no necesariamente se localiza en este cambio. Las propiedades de la imagen comúnmente consideradas son intensidad, color y textura. Las características locales pueden ser puntos, pero también bordes o pequeños parches de imagen. Por lo general, algunas mediciones se toman de una región centrada en una característica local y se convierten en descriptores. Los descriptores se pueden usar para diversas aplicaciones[25]. La

Figura 2.5 muestra algunos ejemplos de características locales, las cuales como se ha mencionado, pueden ser cambios de intensidad, color y textura.

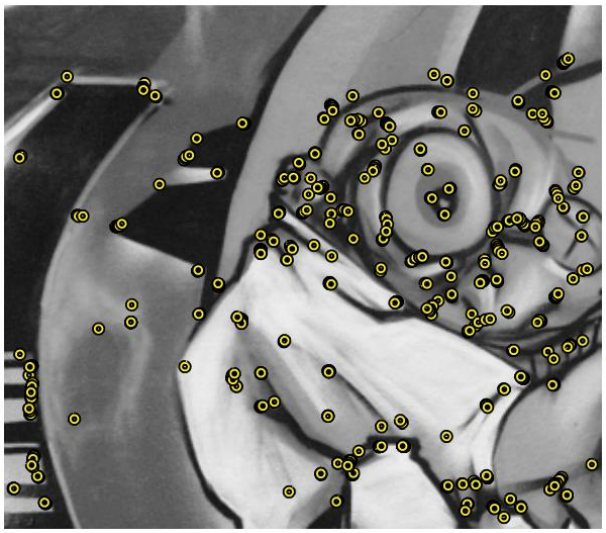

Figura 2.5 Puntos destacados [25]

2.1.7 ORB

Es un algoritmo propuesto en el artículo ORB: An efficient alternative to SIFT or SURF [13] en el año del 2011 y es una propuesta para reemplazar a los algoritmos detectores y descriptores tales como SIFT o SURF, con un rendimiento similar y con menor costo computacional, según los autores. ORB u Orientado FAST y rotado BRIEF (en inglés "*Oriented FAST and Rotated BRIEF*") está basado en el conocido detector puntos clave FAST, pero con un componente de orientación y del descriptor BRIEF más un componente para la invarianza rotacional. En la Figura 2.6 se observa cómo se detectan puntos característicos del lado izquierdo y del lado derecho aún se mantienen a pesar de que existe una rotación.

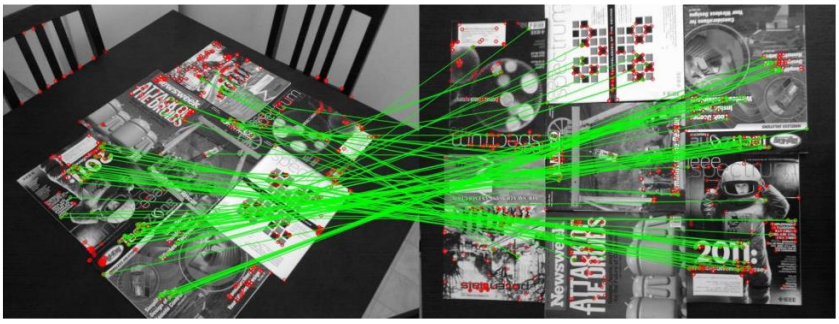

Figura 2.6 Resultado típico usando ORB [13]

2.1.8 Bag of words

La bolsa de palabras es una técnica que utiliza un vocabulario visual para convertir una imagen en un vector numérico disperso, lo que permite administrar grandes conjuntos de imágenes [8], en la Figura 2.7 se observa una mujer del lado izquierdo y del derecho las “palabras visuales” que la caracterizan. El estado de extracción de características del modelo se calcula utilizando descriptores de imagen como SIFT o SURF. SIFT y SURF, ambos detectan características locales en una imagen dada [26]. El modelo BOVW tiende a formar grupos de características similares y asignarles una palabra visual. Según Sirishaa et al. [27] el proceso de BOVW se divide en los siguientes pasos:

i) Detectan automáticamente puntos característicos, líneas, regiones de interés de las imágenes.
ii) Los descriptores locales se calculan sobre los puntos característicos detectados.
iii) Cuantifican los descriptores locales en palabras visuales para formar el vocabulario visual.
iv) Encuentran las instancias / ocurrencias en las imágenes de entrada de cada palabra visual en el vocabulario visual para la creación de una bolsa de palabras visuales, presenta un vector o un histograma de frecuencias de palabras visuales.

Figura 2.7 Ejemplo de BOVW [28]

2.1.9 K-means

El algoritmo K-means, creado por MacQueen en 1967 es el algoritmo de clustering más conocido y utilizado ya que es de muy simple aplicación y eficaz [29]. Sigue un procedimiento simple de clasificación de un conjunto de objetos en un determinado número *K* de clústeres, *K* determinado a *priori*. Sin embargo, cuando se tienen muchos datos se vuelve computacionalmente costoso.

El nombre de K-means viene porque representa cada uno de los cluster por la media (o media ponderada) de sus puntos, es decir, por su centroide.

La representación mediante centroides tiene la ventaja de que tiene un significado gráfico y estadístico inmediato.

Cada clúster por tanto es caracterizado por su centro o centroide que se encuentra en el centro o el medio de los elementos que componen el clúster. K-means es traducido como K-medias.

2.2 Antecedentes

Dentro del CENIDET, más específicamente dentro del grupo de inteligencia artificial se han desarrollado algunas tesis que son las bases para la realización de este

trabajo y por lo tanto conforman los antecedentes directos para esta tesis. A continuación, se mencionan algunas de los más relevantes.

2.2.1 Detección de Cierre de Ciclos en el Problema de Localización y Mapeo simultáneo Visual Monocular [7]

En esta tesis, el autor realizó un método para la detección de cierre de ciclos en secuencias de imágenes monoculares. Dicha detección de cierre de ciclos consistió en identificar de manera rápida y precisa los lugares que ya había visitado el robot. En la Tabla 2.1 se muestran los tiempos que toma la realización de todo el sistema en la base de datos de Bovisa04 con 11,391 imágenes.

Tabla 2.1 Tiempos del sistema del autor [7]

Etapa		Tiempo (ms)			
		Promedio	σ	Min	Max
Blobs SURF	Detección	35.586	5.386	25.265	56.721
	Descripción	39.718	15.860	0.466	87.026
Esquinas OFAST	Detección	3.776	0.903	1.899	9.1152
	Descripción	123.118	37.091	0.139	210.960
Identificadores Visuales		35.646	10.969	0.1422	65.427
Agenda visoespacial	Agrupamiento	3.949	1.740	0.019	8.268
	Correspondencia	61.729	45.382	0.0235	383.887
	Olvido	4.170	1.691	0.006	14.864
	Verificación	0.002	0.001	0.0004	0.051
	Codificación	52.042	22.651	0.0093	109.446
Ejecutivo central	Consulta	282.85	223.962	0.008	945.145
	Almacenamiento permanente	14.055	8.388	0.0041	59.081
	Recuperación	1.940	5.737	0.5434	35.871

Sistema Completo	355.974	197.77	25.983	1670.8

En la Tabla 2.1 se menciona *blob*, el cual es una región brillante sobre un fondo obscuro o viceversa. Y una esquina se caracteriza por una región en la imagen con cambios de intensidad en diferentes direcciones. Las partes de identificadores visuales, agenda visoespacial y ejecutivo central son parte de BOVW.

Como se puede observar en la Tabla 2.1, el tiempo promedio de cada imagen fue de 355.974 segundos. Las imágenes fueron adquiridas con cámaras montadas en robots móviles y automóviles, son de interiores y de exteriores. Este trabajo obtuvo un resultado de detección de cierre de ciclos de 100% de precisión. Su importancia radicó en la corrección de los errores de la localización que se acumulan con el transcurso del tiempo, cuando se detecta un cierre de ciclo, se corrige la posición.

2.2.2 Comparación de Algoritmos de Extracción y Asociación de Rasgos Para Visión Robótica [14]

En esta tesis, el autor realizó un sistema para la detección de cierre de ciclos que en caso de detectar una nueva escena la almacena en la base de datos. El sistema propuesto consta de 3 módulos: 1) Adquisición, 2) Extracción y 3) Asociación. En la parte de adquisición es posible adquirir tanto imágenes monoculares como estéreo-binoculares. Está diseñado basado en características locales por lo que se compone de un algoritmo de detección y uno de descripción (SIFT), en la parte de detección utiliza varios algoritmos para comparar los entre ellos (ASIFT, MSER, Hesse afín, Harris afín), además este módulo se basa en un filtro de seguimiento y rechazo de frames. El módulo de asociación fue desarrollado considerando técnicas de recuperación de información, implementando por esta razón dos algoritmos de asociación (Bolsa de palabras y el Árbol de vocabulario). El autor recomienda utilizar a Hessian affine como el algoritmo de detección y Árbol de vocabulario (Vocabulary

tree) como el algoritmo de asociación, para el ambiente en donde se realizaron las pruebas.

2.2.3 Discusión

Después de haber realizado un estudio sobre los trabajos que se han desarrollado dentro del CENIDET, se puede denotar que ha tenido cierto impacto el tema de la asociación de lugares, pero ninguno ha podido lograrla en tiempo real. La asociación en tiempo real disminuye el error acumulado, ayudando en la correcta ubicación del robot.

2.3 Estado del arte

En esta parte se presentan algunos trabajos relacionados que plantean soluciones al problema de cerradura de ciclos, mediante BOVW.

2.3.1 *Real-Time Loop Detection with Bags of Binary Words* [4]

En este trabajo se propone un método para detectar lugares ya visitados (Cerradura de ciclos) en tiempo real y en imágenes monoculares, utilizando el algoritmo de bolsa de palabras, el detector de FAST y el descriptor de BRIEF. Además de un control geométrico para la comprobación de *Bag of words*. En la Tabla 2.2 se muestran los tiempos que tarda todo el sistema propuesto probándolo con una base de datos con 2,723 imágenes.

Tabla 2.2 Tiempos de [4]

		Promedio	σ	Min	Max
Características	FAST	2.39	1.07	1.37	7.84
	Suavizado	1.27	0.05	1.21	1.43

	BRIEF	1.44	0.37	0.11	1.74
Bag of words	Conversión	3.00	0.78	0.21	3.66
	Consulta	0.17	0.13	0.00	0.62
	Islas	0.06	0.02	0.01	0.11
	Inserción	0.02	0.04	0.00	0.13
Verificación	Correspondencias Y RANSAC	0.82	1.29	0.25	5.79
Todo el sistema		6.15	4.03	1.37	16.34

Los tiempos máximos solo fueron de 16.34 segundos, en el artículo se muestra otra tabla con los tiempos de 19,344 imágenes y se puede observar que la parte de consulta crece de 0.17 milisegundos a 4.95 milisegundos.

2.3.2 *Bags of Binary Words for Fast Place Recognition in Image Sequences* [8]

Este artículo es la continuación de [4], en donde el autor propone además de utilizar el algoritmo de BOVW para la asociación, FAST como detector y BRIEF como descriptor ahora construye un árbol de vocabulario que se discretiza en un espacio binario, se utiliza ese árbol para aumentar la velocidad de la correspondencia geométrica. La Figura 2.8 muestra un ejemplo de cómo se organiza el árbol, es un árbol de 9 palabras visuales.

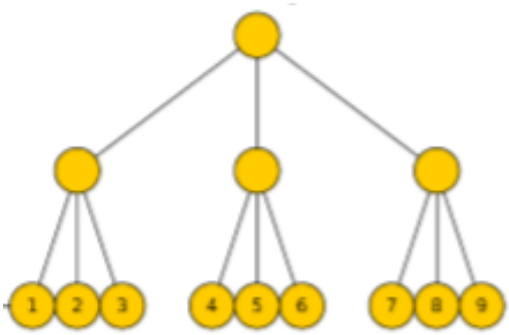

Figura 2.8 Árbol de vocabulario [8]

El árbol de vocabulario se construye fuera de línea y a pesar de que se probó con conjuntos de datos muy diferentes, con la misma configuración, se lograron resultados sin falsos positivos. En la Tabla 2.3 se pueden ver los tiempos que se obtuvieron con una base de datos de 26,292 imágenes.

Tabla 2.3 Tiempos de [8]

		Promedio	σ	Min	Max
Características	FAST	11.67	4.15	1.74	30.16
	Suavizado	0.96	0.37	0.79	2.51
	BRIEF	1.72	0.49	1.51	4.62
Bag of words	Conversión	3.59	0.35	3.27	8.81
	Consulta	3.08	1.91	0.01	9.19
	Islas	0.12	0.04	0.08	0.97
	Inserción	0.11	0.02	0.06	0.25
Verificación	Correspondencias Y RANSAC	1.60	2.64	0.61	18.55
Todo el sistema		21.60	4.82	8.22	51.68

Considerando la cantidad de imágenes, los tiempos de procesamiento de todo el sistema fueron muy buenos, ya que el autor compara sus resultados con los de *FAP-MAP* [30], [31] obteniendo resultados parecidos pero con menor tiempo de ejecución.

2.3.3 *High performance loop closure detection using bag of word pairs* [32]

En este artículo, el autor propuso un método para eliminar la debilidad de BOVW ante el "perceptual *aliasing*", incorporando información de coocurrencia espacial directamente en el diccionario. Esto lo hizo mediante la creación de un diccionario adicional, que contiene pares de palabras. Una diferencia notable de este artículo es que no sólo ocupa el arreglo espacial para la verificación, sino que las utiliza desde la detección y por lo tanto influyen en todas las demás etapas. Utiliza el algoritmo de SURF para la detección y descripción. No crea soló un diccionario de palabras, sino que crea 2, uno de palabras utilizando árboles K-D y otro para los pares de palabras utilizando la estructura de datos multimap. En la Figura 2.9 se muestra un ejemplo de palabras pares, como se puede ver son las palabras visuales que están muy cercanas con otras.

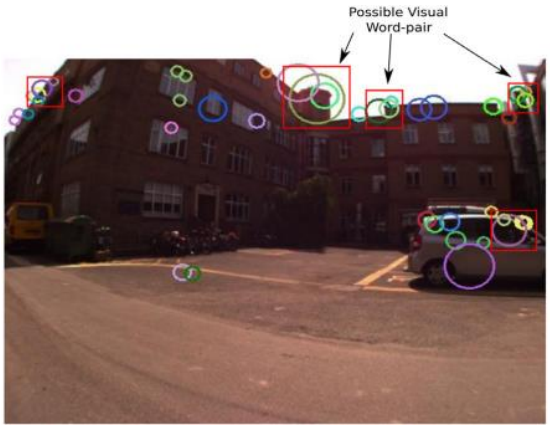

Figura 2.9 Palabras pares [32]

2.3.4 *Original Loop-closure Detection Algorithm for Monocular vSLAM* [33]

En este trabajo se propone un método combinado de imagen a imagen y de imagen a mapa para lograr robustez y precisión, debido a que los métodos comunes de imagen a imagen dependen mucho del vocabulario. El método utiliza la posición de la cámara para la búsqueda de cierre de ciclos en la base de datos, solamente teniendo en cuenta las que están en su ángulo de visión. Como detector y descriptor, el autor escogió a ORB. Con este método se obtuvo el mismo porcentaje de error que los métodos de ORB-SLAM Y LSD-SLAM que va del 1.5% al 2.5% mientras que reduce el tiempo de procesamiento entre un 7% y un 10 %. La Figura 2.10 muestra el resultado de comparar el mapa generado por el método propuesto contra el mapa generado por ORB-SLAM, LSD-SLAM y el mapa real del terreno.

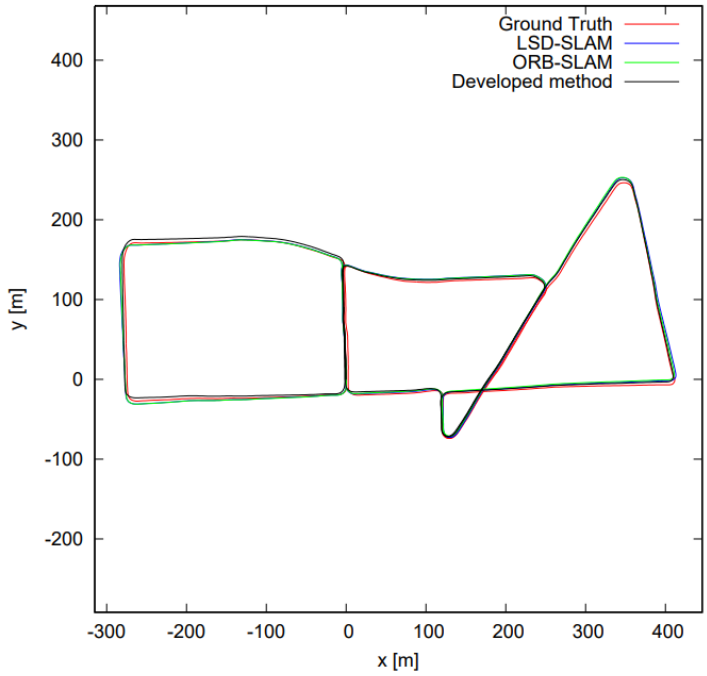

Figura 2.10 Comparación del autor [33]

2.3.5 *Analysis of two visual odometry systems for use in an agricultural field environment* [23]

En este trabajo se expone la comparación de 2 algoritmos de odometría visual, *Granty* y *Libviso*. Evalúa su desempeño con diferentes parámetros de configuración. Los compara utilizando 2 base de datos, KITTI *Vision Benchmark Suite* y una que el autor diseñó utilizando el motor gráfico de Unity. Algo importante de resaltar sobre este artículo es que propone una tasa de fotogramas máxima, a 20 Hz ya que con una mayor tasa se aumenta el error acumulativo por metro.

El algoritmo *Libviso* con la base de datos de KITTI obtuvo un 3.6% de error en la posición y 0.0482° en la orientación.

2.3.6 *Static and dynamic validation of inertial measurement units* [34]

En este artículo se presenta una evaluación de la exactitud y precisión de 2 unidades de medición inercial comerciales y las compara con sistemas de medición diseñados para un uso en concreto, utilizando una máquina para crear un ambiente de pruebas tanto estáticas como dinámicas controladas. Los resultados mostraron un error dentro de ± 1°, teniendo mejores resultados con pruebas de giros menores a 90°, además el porcentaje de error se mantiene relativamente constante a todas las velocidades angulares.

2.3.7 *Monocular Visual Inertial Navigation for Mobile Robots using Uncertainty based Triangulation* [35]

En este artículo se presenta un filtro de Kalman de restricción multi estado robusto (MSCKF) modificado, para la navegación inercial visual de robots móviles. Utiliza

una IMU y una cámara monocular. Lo novedoso de este artículo es porque, en el algoritmo propuesto no se utilizan las incertidumbres de la pose de la cámara, ni las mediciones de las imágenes en el proceso de triangulado del punto 3D. En la parte de pruebas utiliza la base de datos de KITTI y una que genera por computadora. Para la detección de características utiliza el algoritmo de FAST, en el seguimiento utiliza el Lucas-Tomasi (KLT). Según el autor sus resultados fueron mejores que los del algoritmo MSCKF normal, pero no muestra una tabla comparativa.

2.3.8 Discusión

Se ha realizado una búsqueda de información sobre varios temas, pero el principal fue la detección de cierre de ciclos en tiempo real, algunos autores han logrado este objetivo por diferentes métodos, en la Tabla 2.4 se muestra una comparación de estos trabajos.

Tabla 2.4 Comparación trabajos relacionados

Trabajo relacionado	Detector	Descriptor	Asociador
Real-Time Loop Detection with Bags of Binary Words [4]	FAST	BRIEF	Bag of words
Bags of Binary Words for Fast Place Recognition in Image Sequences [8]	FAST	BRIEF	Bag of words
High performance loop closure detection using bag of word pairs [32]	SURF	SURF	Bag of words
Original Loop-closure Detection Algorithm for Monocular vSLAM [33]	ORB	ORB	Bag of words

3. Capítulo 3 Metodología de solución

En esta sección se presenta la metodología que se utilizó para la solución de la problemática.

3.1 Detectores y descriptores de puntos característicos

Existen un gran número de algoritmos para detectar, otros para describir y otros que pueden detectar y describir, pero sólo se utilizan en esta tesis los más populares en el estado del arte para Bovw, los cuales son ORB [13], SURF [12] y SIFT [36].

3.1.1 SIFT

The Scale Invariant Feature Transform (SIFT) fue introducido por Lowe [36] en 1999 y la idea principal es poder encontrar puntos invariantes a la traslación, el escalado y la rotación, dentro del espacio-escala de la imagen. Se realiza en cuatro etapas, la primera es la detección de extremos en el espacio escala, la segunda es la localización exacta de los puntos destacados, la tercera es la asignación de la orientación y la cuarta es la descripción de puntos destacados. En otras palabras SIFT se realiza con una diferencia de gaussianas (DOG) para la detección y su descriptor, sin embargo se pueden utilizar por separado, un ejemplo sería Jaime Robledo que en su tesis titulada “Comparación de Algoritmos de Extracción y Asociación de Rasgos Para Visión Robótica” [14] utiliza *Hessian affine*, *Harris affine*, MSER y ASIFT para la detección y el algoritmo de SIFT para la descripción de sus puntos característicos, un ejemplo del funcionamiento de SIFT utilizando las librerías de OpenCV en Python se puede observar en la Figura 3.1, tiene 1980 puntos destacados que se muestran como círculos.

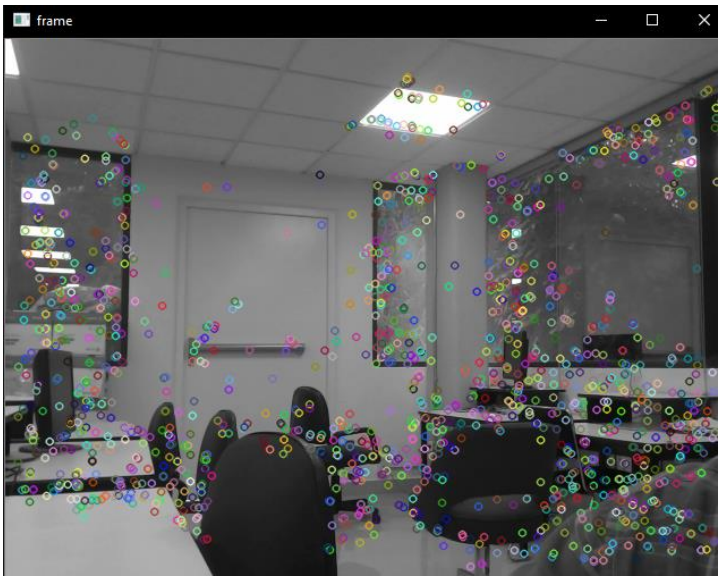

Figura 3.1 SIFT en python

3.1.2 SURF

A medida que los detectores y descriptores de puntos de interés se han popularizado, las implementaciones eficientes son cada vez más importantes [37]. Una de los que cabe destacar es Speeded-Up Robust Features (SURF) que fue propuesta como una alternativa eficiente a SIFT, propuesta por Tuytelaars [12] en el 2006.

SURF propone unas mejoras a SIFT, una de las más importantes son las de que SURF en lugar de depender de derivados gaussianos utiliza transformadas Haar de 2D aplicándolos sobre imágenes integrales, cambia el tamaño de las Haar en lugar de las pirámides gaussianas que utiliza SIFT.

Cornelis en su artículo titulado " Fast scale invariant feature detection and matching on programmable graphics hardware" [38] presenta una modificación a SURF aplicándolo sobre una GPU logrando rebasar los 100 cuadros por segundo.

En la Figura 3.2 se puede observar una implementación de SURF utilizando las librerías de OpenCV en el lenguaje de programación Python, se realizó en las

instalaciones del CENIDET, cada uno de los 1800 círculos significa un punto característico.

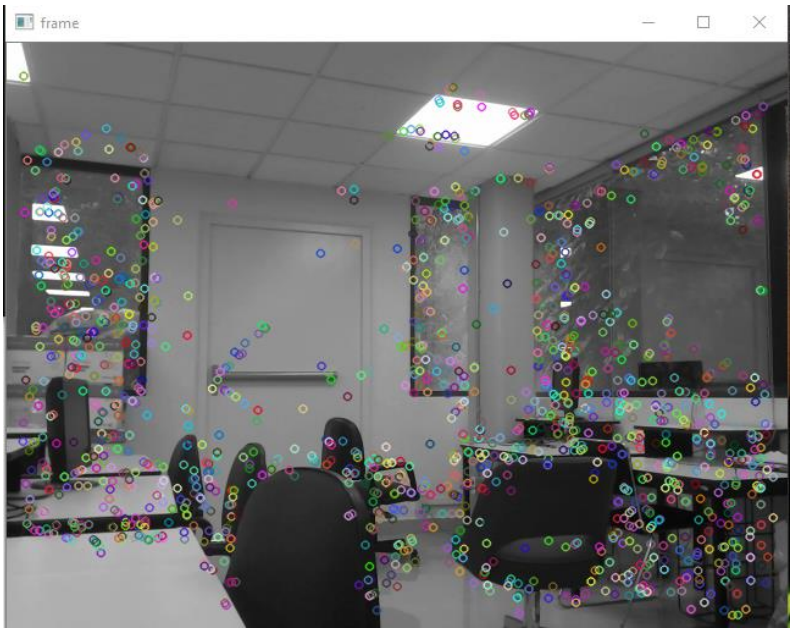

Figura 3.2 SURF en python

3.1.3 ORB

Este algoritmo fue propuesto por Rublee en su artículo titulado "*ORB: An efficient alternative to SIFT or SURF*" [13] en el 2011 y como su título lo dice, es una alternativa a los algoritmos de SIFT y SURF. ORB básicamente es una combinación de los algoritmos de FAST para la detección y BRIEF para la descripción, pero con bastantes mejoras, tales como una parte de escala para FAST, y una parte de orientación para BRIEF. Una cosa para resaltar es que ORB es de licencia libre, mientras que si se quiere utilizar SIFT o SURF para usos comerciales se tiene que pagar por una licencia. La Figura 3.3 se muestra el resultado de la implementación de ORB utilizando OpenCV y python.

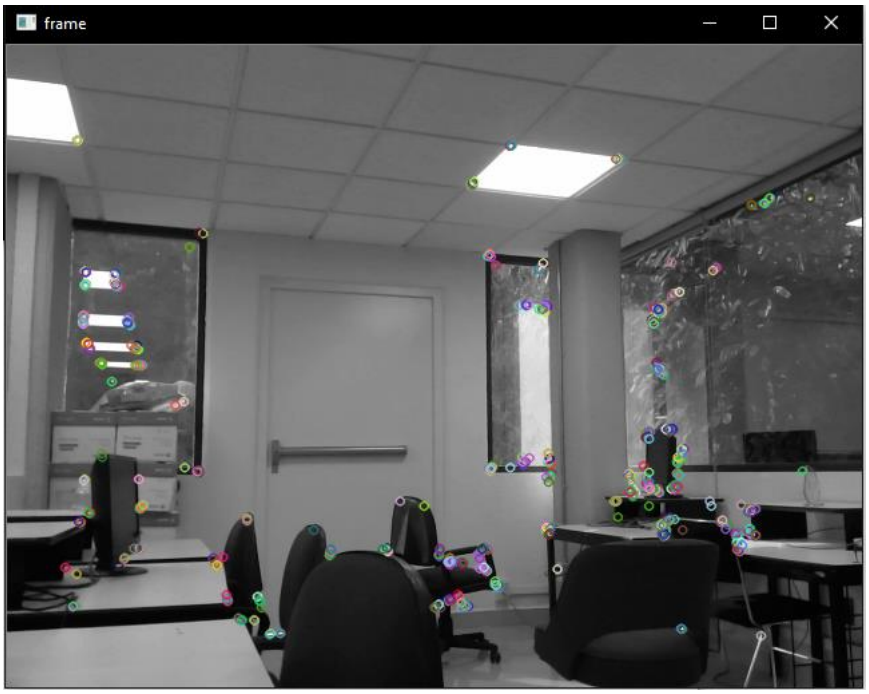

Figura 3.3 ORB en python

3.1.4 Tiempos

Utilizando las configuraciones por defecto que tienen en OpenCV, los tiempos y el número de puntos característicos detectados se muestran en la Tabla 3.1. Y como se puede observar el más rápido es ORB a pesar de ser el que menor número de puntos característicos detecta.

Tabla 3.1 Tiempos y puntos detectados ORB, SIFT y SURF en un procesador I7-4710 HQ a 2.5GHz

Detector y descriptor	Puntos característicos detectados	Tiempos en segundos
ORB	500	0.025
SIF	1980	0.34
SURF	1800	0.25

3.1.5 Conclusiones de Detectores y descriptores de puntos característicos

SIFT es el que mayor número de puntos destacados detecta, siguiéndolo SURF y al último ORB, pero por los tiempos que se tardan los dos primeros no son viables para aplicaciones en tiempo real, además siguiendo los pasos de varios autores como, por ejemplo, Bampis que en su artículo llamado "*Fast loop-closure detection using visual-word-vectors from image sequences*" [6] el detector y descriptor que maneja es ORB, por lo tanto se utiliza ORB en esta tesis, para lograr la asociación en tiempo real.

3.2 Bancos de imágenes

Para poder logar una buena asociación dentro del CENIDET (Centro Nacional De Investigación y Desarrollo Tecnológico) se necesita realizar un árbol de vocabulario con imágenes relacionadas con su ambiente, además de que es uno de los bancos de imágenes más citados en el estado del arte, se escogió el banco de imágenes de Bovisa [39], en específico la secuencia "Bovisa_2008-09-01" la cual consta de 65,045 imágenes tanto en color como en blanco y negro de un recorrido por el campus del "*Politecnico di Milano*" ubicado en Milán, Italia, el cual realizado por el interior de algunos edificios, además en el exterior cuenta con vista de árboles, edificios, autos tanto estáticos como en movimiento adicionalmente el tránsito de personas, algunos ejemplos se pueden ver en la Figura 3.4.

Figura 3.4 Bovisa_2008-09-01

Por otra parte, también era necesario incluir en el árbol de vocabulario imágenes del CENIDET, por lo cual se realizaron tres recorridos de las instalaciones del edificio de ciencias de la computación los cuales se realizaron los días 28 y 29 de marzo a las 6 pm y a las 8:30 am respectivamente. En estos recorridos se pueden ver el salón de los alumnos del departamento de ciencias de la computación con pocos alumnos y prácticamente sin movimientos, así como también los demás salones del edificio Figura 3.5.

Figura 3.5 Laboratorio de computación CENIDET

3.3 Árbol de vocabulario

Para la generación del árbol de vocabulario se utiliza la librería de Gálvez y Tardós [40] la cual fue utilizada en su artículo titulado "*Real-time loop detection with bags of Binary words*" [4], el algoritmo se explica a continuación.

El árbol de vocabulario se puede generar desde la cuantificación de los puntos característicos a partir de un conjunto de descriptores extraídos de un grupo de imágenes (Corpus) de entrenamiento, en otras palabras, primeramente se necesita tener un conjunto de imágenes de entrenamiento relevantes para el lugar en donde se va a desenvolver el robot, por ejemplo, si el robot se va a mover en interiores, se necesitan fotos de pasillos y de cuartos, posteriormente detectar y describir sus puntos característicos de todas las imágenes utilizando algún algoritmo especializado en ese proceso como puede ser, SIFT, SURF, ORB, etc., en este caso se utiliza ORB.

A todo el vector resultante que adentro contiene más vectores se le puede aplicar algoritmos como *Approximate k-means* [41] que aproxima a K-means utilizando ocho árboles k-d aleatorios o Hierarchical k-means [42], [43], este es el que se utiliza en este proyecto.

Para la creación del árbol de vocabulario utilizando Hierarchical k-means se utiliza el vector resultante de la parte de extracción de características de todo el conjunto de entrenamiento y se les aplica K-means recursivamente.

Una forma sencilla de explicar esto es imaginando que los descriptores entregan un vector de dos dimensiones (x, y) y dibujando todos los puntos característicos descritos en un plano cartesiano, así como se puede observar en la Figura 3.6 a) para después aplicarle K-means [8] y dividir en cuatro regiones (k), con sus respectivos centroides (Figura 3.6 b)).

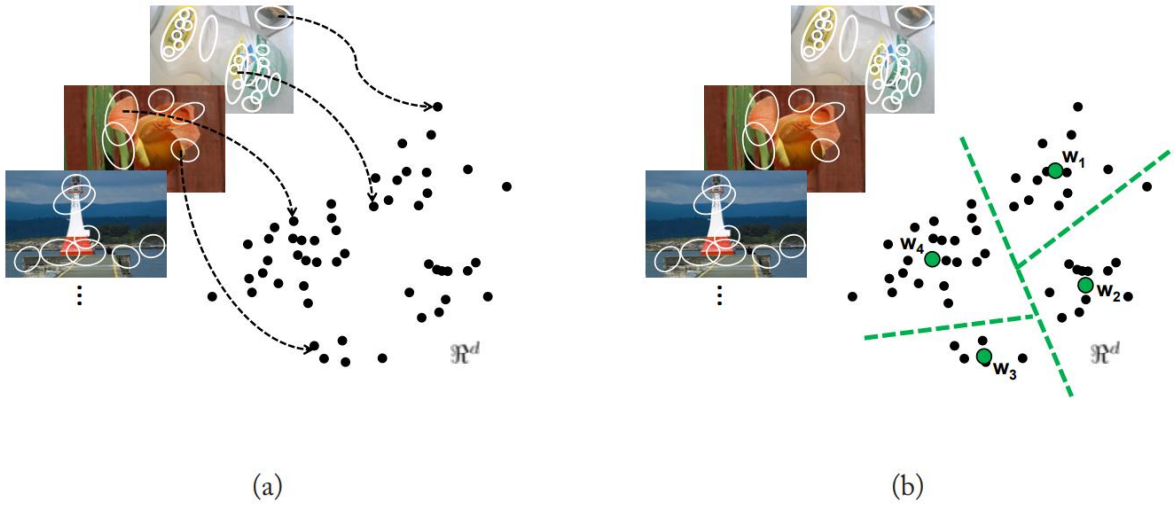

Figura 3.6 a) Dibujado de los puntos característicos descritos en el plano. b) Primera aplicación de K-means [12]

Cada región se divide de nuevo en *k* regiones, y este proceso se repite hasta *L* veces, como se puede ver en la Figura 3.7, en donde de izquierda a derecha se observa cómo se va subdividiendo en regiones más pequeñas.

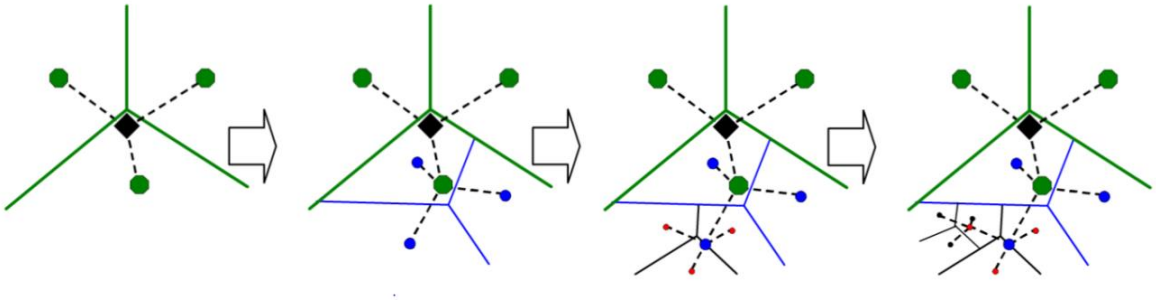

Figura 3.7 Aplicaciones de K-means [11]

Las palabras visuales son los centroides del último nivel. La cantidad de palabras visuales se puede calcular a través de la Ecuación 3.1 [43].

$$\frac{K^{L+1} - K}{K - 1} \approx K^{L}$$ Ecuación 3.1

Donde *K* representa la ramificación o en cuantas partes se va a dividir cada nivel y *L* representa la profundidad o los niveles deseados.

Algunos autores, como Tardós [4] proponen una ramificación (*K*) de 10 y una profundidad (*L*) de 6, generando un total aproximado de $10^6 = 1{,}000{,}000$ palabras visuales.

La parte final de la creación del árbol de vocabulario es la de asignarle un peso a cada palabra visual, dependiendo de lo discriminante que es en texto, un ejemplo sería las palabras “la”, “el” tendrían un menor peso que la palabra “ferrocarril” debido a que son más comunes. Una forma de pesar es “inverse document frequency o idf“ Ecuación 3.2 [42].

$$log\left(\frac{N}{n_i}\right)$$ *Ecuación 3.2*

En donde *N* representa la cantidad de imágenes de entrenamiento y n_i el número de veces que aparece la palabra *i* en dichas imágenes.

En la Figura 3.8 se puede observar un ejemplo de un árbol de vocabulario el cual está en formato YAML, para su construcción se utilizaron 78 mil imágenes las cuales fueron la unión del banco de datos de imágenes capturadas dentro del CENIDET con el banco de imágenes de Bovisa.

- { nodeId:1107229, parentId:1107141, weight:1.1251703547775568e+01,
descriptor:"249 55 251 230 180 86 215 188 115 205 254 108 115 249 221 55 216 254
52 249 249 251 188 170 251 253 175 140 237 251 118 47 " }
- { nodeId:1107230, parentId:1107141, weight:1.1251703547775568e+01,
descriptor:"109 93 243 102 116 236 111 214 115 193 122 74 115 245 249 48 58 254
22 202 129 50 188 106 119 44 32 40 231 243 98 185 " }
- { nodeId:1107231, parentId:1107141, weight:1.1251703547775568e+01,
descriptor:"237 229 235 102 116 4 103 21 112 201 125 94 115 249 216 50 80 238 20
127 233 242 188 10 127 45 164 40 239 242 242 226 " }
- { nodeId:1107232, parentId:1107141, weight:1.1251703547775568e+01,
descriptor:"109 237 235 86 244 12 103 21 112 203 127 78 114 233 216 114 88 238
52 14 233 242 252 106 118 45 165 40 239 250 242 224 " }

Figura 3.8 Ejemplo Árbol de vocabulario

3.4 IMU

La navegación inercial es una técnica de navegación autónoma en la que las mediciones proporcionadas por acelerómetros y giroscopios se utilizan para rastrear la posición y la orientación de un objeto en relación con un punto de partida, orientación y velocidad conocidas[44].

Para este proyecto se utilizó la IMU de Adafruit *"9-DOF IMU Breakout-LG3D20H +LSM303"* la cual consta de un acelerómetro, un giroscopio y un magnetómetro. La IMU físicamente se muestra en la Figura 3.9.

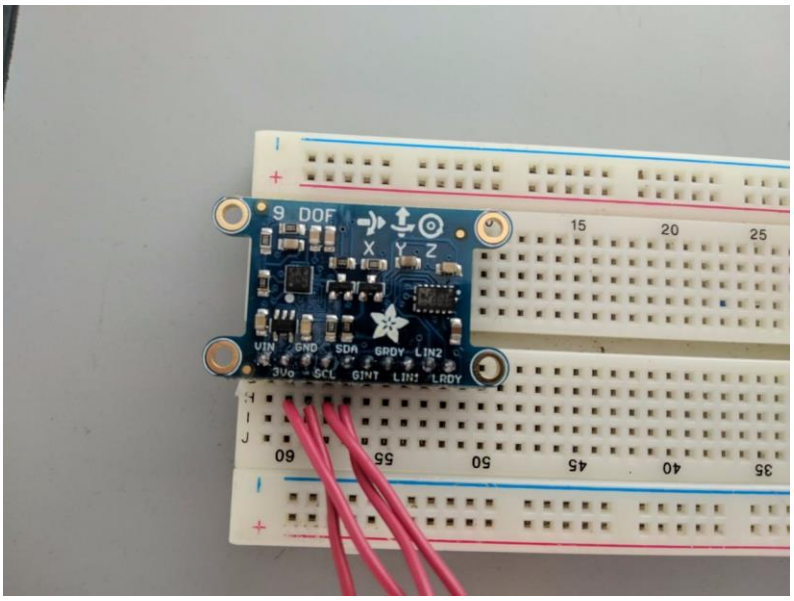

Figura 3.9 IMU físicamente

La función de la IMU en este proyecto es la de dividir la base de datos en ocho partes, esto con la finalidad de disminuir los tiempos de consulta, para esto se utilizó el magnetómetro, para asignarle a cada imagen la orientación con la que fue capturada, la Figura 3.10 muestra una rosa de vientos con números del 1 al 8 representando las zonas en las que se divide la base de datos.

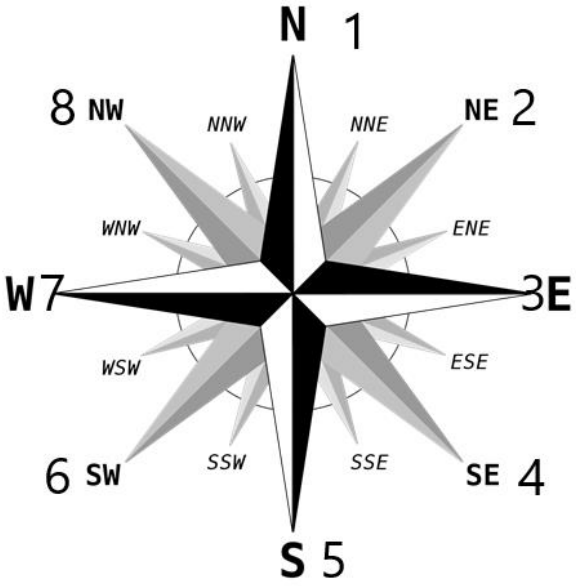

Figura 3.10 Ejemplo de orientaciones [45]

Se realizaron varias pruebas iniciales adquiriendo la orientación de la IMU sin embargo se llegó a la conclusión de que el magnetómetro es sensible a cualquier campo electromagnético. Por lo cual fue necesario realizarle una calibración a la IMU, para esto se utilizó el software *"PJRC MotionCal"* [46], el cual a través de las mediciones tomadas realizando movimientos circulatorios entrega valores de calibración.

La Figura 3.11 muestra una nube de puntos generada por el movimiento circular de la IMU, al aparecer una esfera sin ruido significa que tiene una buena calibración.

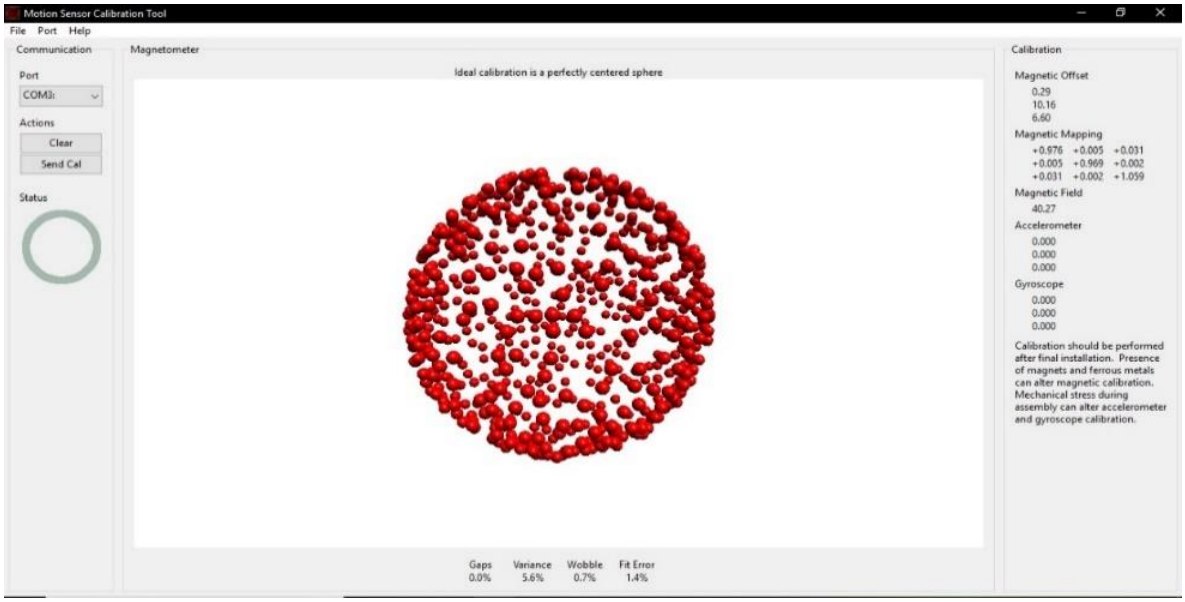

Figura 3.11 Calibración Magnetómetro

Posteriormente se prosiguió con la creación del programa encargado de leer los valores de la IMU, promediarlas y retener el último promedio para que cuando sea requerido mandárselo al programa en C++.

El programa comienza leyendo 10 veces la IMU y le saca el promedio de esos valores, prosigue a asignarle una orientación a ese valor promediado, en este caso siempre se utilizan las ocho orientaciones así que se le asigna un valor entre 1 y 9, considerando el 9 como una falla de la IMU, repite este proceso constantemente esperando que le llegue el carácter 'a' por el puerto serial, cuando se recibe el carácter manda la última orientación asignada, esto debido a que si constantemente

le está mandando el ultimo valor promediado por el puerto serial, el programa en C++ no puede leerlos correctamente, ya que intenta leer valor por valor en su cola y no el que necesita en ese momento, por esta razón el programa en C++ es el encargado de solicitar la orientación.

3.5 Bag of visual words (Bovw)

Bag of visual words es una técnica que representa una imagen como un vector numérico mediante la cuantificación de sus puntos característicos [47], existen dos artículos los cuales se consideran los precursores para Bovw, uno es "*Visual Categorization with Bags of Keypoints*" [48] en donde el autor propone la cuantificación con K-means y la clasificación utilizando diferentes algoritmos como SVM (máquinas de soporte vectorial) y Naïve Bayes. El segundo artículo, es del autor Sivic con el nombre de "*Video Google: A Text Retrieval Approach to Object Matching in Videos"* [42] en el cual maneja el enfoque de bolsa de palabras que funcionaba en Google en ese momento para buscar un texto, y lo modifican para poder ser utilizado en imágenes.

El modelo de Bovw para la asociación se basa en dos procesos, el primero es una fase de "entrenamiento" o también llamado creación del árbol de vocabulario, la cual fue explicada en la sección 3.3 y el segundo es un proceso en tiempo real, el cual se explica a continuación.

3.5.1 Conversión de imágenes a Bovw

Para las aplicaciones en tiempo real, primeramente, se necesita la adquisición de una imagen, detectar sus puntos característicos y mandarlos al árbol de vocabulario, el cual le asigna una palabra visual a cada punto.

La forma en que hace este proceso es a través de calcular distancia entre el vector del descriptor y los centroides del primer nivel, escoge el que tenga la distancia menor, después continua con los centroides hijos del centroide seleccionado y así prosigue hasta llegar a la raíz. La distancia que se calcula depende del tipo algoritmo descriptor que se utiliza, pueden ser distancia Haming, euclídea, etc.

Posteriormente se le agrega otro peso a cada palabra visual de manera local, se llama frecuencia del término (t*erm frecuency* o tf) (Ecuación 3.3 [42]).

$$\frac{n_i I_t}{n I_t} \qquad \text{Ecuación 3.3}$$

Donde n_i significa el número de veces que la palabra i parece en la imagen I_t , nI_t es el total de palabras visuales en la imagen I_t. Es decir, para cada palabra visual se le agrega un peso tf-idf, donde idf es calculado anteriormente.

3.5.2 Base de datos

Al momento de capturar una nueva imagen debe ser posible obtener todas aquellas imágenes anteriores que compartan algún grado de similitud con esta, para poder lograr una detección de lugares correcta. Para esto se construye una base de datos, la cual está conformada por un índice directo y un índice invertido.

El índice invertido alberga una relación entre cada palabra visual y en qué imágenes fueron encontradas, esto se realiza con la finalidad de obtener una mayor facilidad entre las consultas y disminuir tiempos de estas. Una forma de ver eso es en la Figura 3.12, en el lado izquierdo de la Figura 3.12 parecen 3 imágenes, en ellas hay palabras visuales, que se guardan en el índice invertido que está del lado derecho, si quisiera buscar en que imágenes aparece la palabra visual número 7, se consulta

el índice invertido y dice que solo aparece en las imágenes 1 y 2, dejando de lado la imagen 3 ya que en esta no aparece la palabra visual 7. En la Figura 3.13 del lado izquierdo se puede ver el índice indirecto ya con los pesos de cada palabra visual.

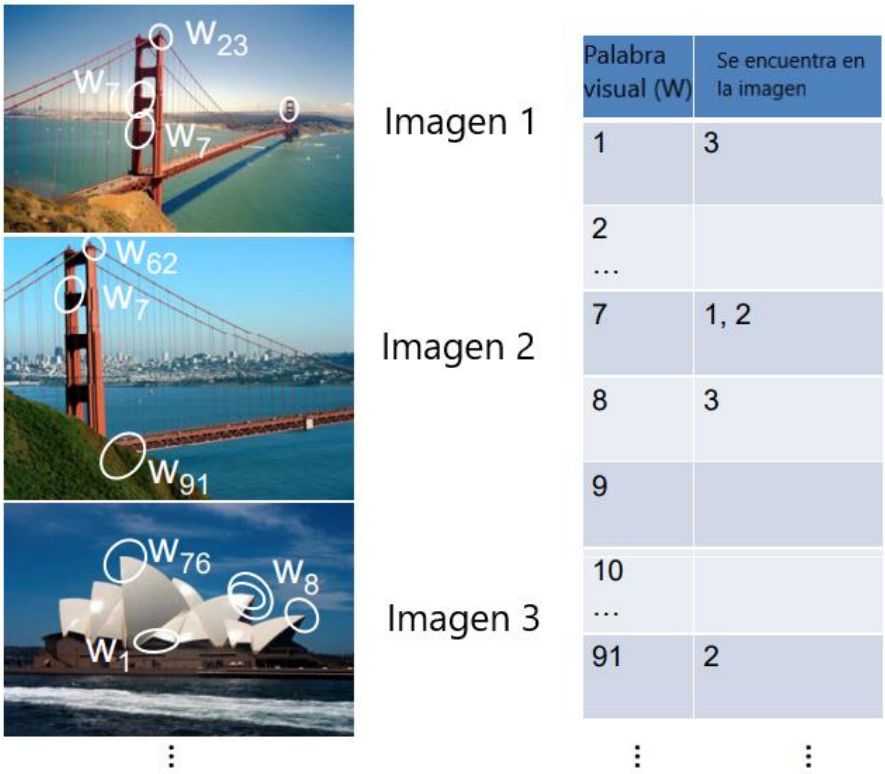

Figura 3.12 índice invertido [24]

Por otro lado, el índice directo guarda la relación de todas las palabras visuales que fueron encontradas en cada imagen, como se ve en la Figura 3.13 del lado derecho se muestra que en la imagen 1 se encontró la palabra visual 1 una vez con un peso de 0.5, la palabra 2 se encontró dos veces con un peso de 0.25.

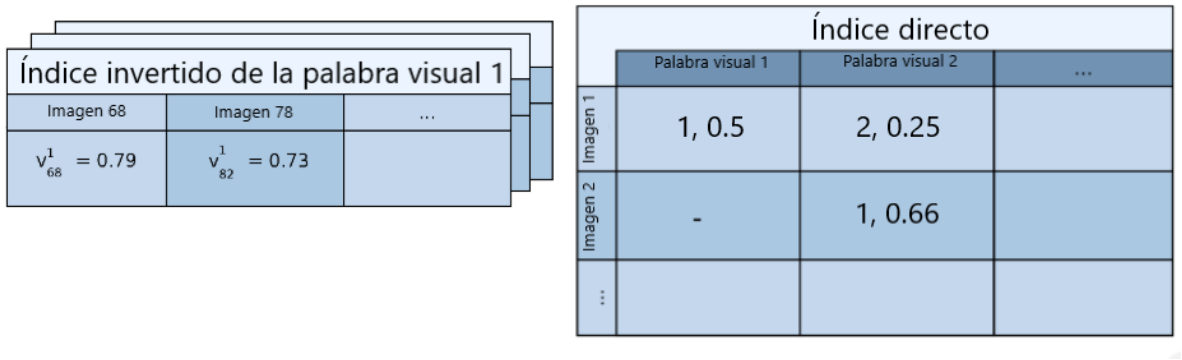

Figura 3.13 índice directo e inverso [1]

3.5.3 Medida de similitud

Se utiliza el vocabulario visual para obtener el vector *bag of visual words* asociado a cada nueva imagen recibida. Este vector es comparado contra el vector asociado a toda otra imagen anterior presente en la base de datos, que comparta alguna palabra en común. Para obtener qué tanta similitud hay entre dos vectores de *bag of visual words* el autor Nistér en su artículo “Scalable Recognition with a Vocabulary Tree” [43] hace mención de la norma L1(Ecuación 3.4).

$$S(V1, V2) = \frac{V1}{|V1|} - \frac{V2}{|V2|}$$

Ecuación 3.4

En donde, V1 y V2 son vectores de Bovw.

Los que tengan una menor diferencia, son considerados posibles candidatos a cerradura de ciclos. Habiendo obtenido ese conjunto de posibles candidatos y junto con sus respectivas valoraciones de similitud se procede a un análisis de consistencia que permitirá determinar si el sitio en el que se encuentra el robot había sido previamente visitado o no.

3.5.4 Detección de cerradura de ciclos

Los pasos anteriormente descritos tienen la finalidad de obtener un vector *de bag of visual words* y calcular la similitud que existe entre los dos vectores. Sin embargo, para detectar una cerradura de ciclos no es suficiente con poder comparar dos vectores, se necesita un análisis más profundo. Un ejemplo sería cuando un robot recorre un pasillo visualmente repetitivo, el cual genera imágenes muy similares, pero esto no significa que debe detectarse como una cerradura de ciclos. Analizar la relación temporal entre candidatos de alta similitud permite discernir entre este tipo de situaciones y verdaderas cerraduras de ciclos.

Cuando una nueva imagen I_t es adquirida, se obtiene su vector de *bag of visual words* V1 asociado y se lo utiliza para consultar la base de datos. De esta consulta se obtiene una lista de todos los posibles candidatos a cerradura de ciclos, <V1, V2>, <V1, V3>,…<V1, VC> junto con sus respectivos valores de similitud S (V1, VJ), con 1≤J≤C. El valor de similitud se normaliza con el valor de similitud entre las imágenes I_t y $I_{t-\Delta t}$, como se ve en la Ecuación 3.5 [4].

$$\eta(V1, VJ) = \frac{S(V1, VJ)}{S(V1, Vt - \Delta t)} \qquad \textit{Ecuación 3.5}$$

Donde $Vt - \Delta t$ representa el vector *bag of visual words* de la imagen anterior a I_t y $S(V1, Vt - \Delta t)$, el grado de similitud entre I_t y $I_{t-\Delta t}$. Aquellos que su valor de $\eta(V1, VJ)$ no supere un umbral α son descartados. Además, podría darse el caso que su valor de $S(V1, Vt - \Delta t)$ sea muy pequeño (por ejemplo, un giro muy rápido del robot) generando erróneamente que el valor de $\eta(V1, VJ)$ sea muy alto. Para solucionar este problema se rechaza la cerradura de ciclos si el valor de $S(V1, Vt - \Delta t)$ (La similitud entre la imagen actual y la anterior) no llega a un mínimo establecido.

Por otro lado, para evitar que imágenes cercanas en el tiempo compitan entre ellas, se agrupan en “islas”, y son tratadas como un único candidato [4]. Se utiliza la

notación T_i para representar un intervalo comprendido por $t_{n_i}, \ldots, t_{m_i}$ y V_{T_i} para islas que agrupen vectores de *bag of visual words* asociados $V_{t_{n_i}}, \ldots, V_{t_{m_i}}$. De esta manera, varios candidatos <V1, $V_{t_{n_i}}$>,…, <V1, $V_{t_{m_i}}$> son considerados como un único candidato <V1, V_{T_i}>. Entonces estas islas se valoran de manera conjunta, utilizando la Ecuación 3.6 [4].

$$\mathrm{H}(Vt, Vt_i) = \sum_{j=n_i}^{m_i} \eta(Vt, Vt_j) \qquad \textit{Ecuación 3.6}$$

La isla de mayor similitud acumulada *H* es seleccionada como el mejor grupo candidato al cual se lo nota como V_T,. Sobre este grupo se aplica una restricción temporal: < Vt, V_T, > debe ser consistente con *k* detecciones positivas anteriores < $Vt - \Delta t$, V_{T_1} >,..., < $Vt - \mathrm{k}\Delta t$, V_{T_k} > donde los intervalos T_i y T_{i+1} son cercanos. Si V_T, cumple esta restricción temporal, se considera como una cerradura de ciclos.

3.5.5 Adquisición e implementación de la orientación

Hasta el momento todo el sistema se presenta tal y como está en la literatura, pero lo novedoso es la implementación de una IMU (Unidad de medición inercial) para la adquisición y aplicación de la orientación. La Figura 3.14 se muestra la IMU utilizada. Es la Adafruit *9DOF* y tiene 3 sensores, un acelerómetro, un giroscopio y un magnetómetro [49].

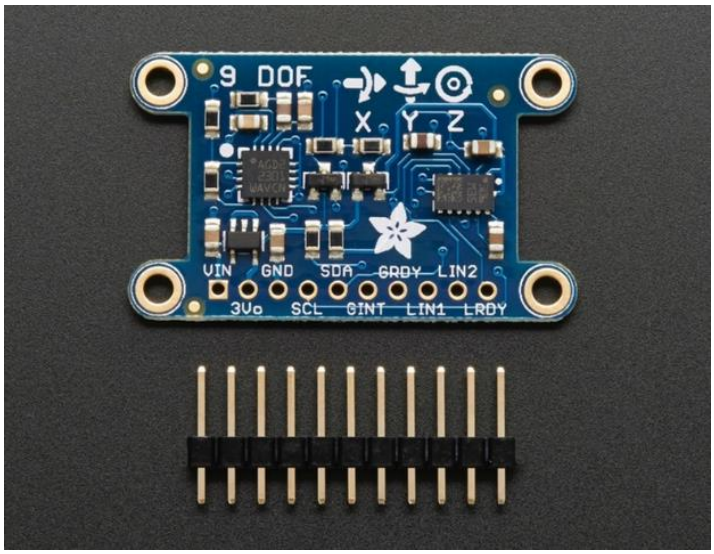

Figura 3.14 IMU [25]

La función de la IMU es utilizar su magnetómetro para agregarle a cada vector de *bag of visual words*, la orientación con la cual fue adquirida la imagen y de esta forma no tener que consultar toda la base de datos, si no solo la que corresponde a la zona donde fue capturada, todo con el objetivo de poder disminuir la forma en la que crecen los tiempos de consulta, así como se mostró en la Tabla 1.1.

Un diagrama de flujo para mejorar la explicación de cómo se complementa Bovw con la IMU se puede observar en la Figura 3.15, se comienza adquiriendo una imagen desde la cámara, a esa imagen se le aplica ORB para detectar y describir sus puntos característicos, se consulta el árbol de vocabulario con esos puntos y se transforman en palabras visuales, después se utiliza la IMU para adquirir la orientación de esa imagen y se consulta el índice invertido pero solo el de esa orientación, para posteriormente decidir si el lugar es reconocido o no, si lo es se corrige la posición, pero si no lo es, se agrega la imagen a la base de datos.

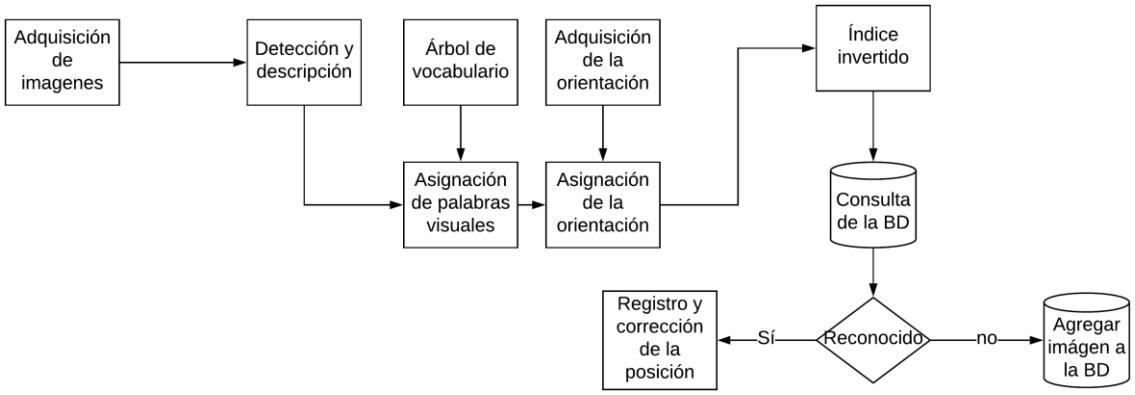

Figura 3.15 Diagrama de flujo

4.0 Capítulo 4 Pruebas y resultados

En esta sección se explican las pruebas más importantes realizadas con la IMU y BOVW.

4.1 Bovw con diferentes valores

Para estas pruebas se utilizaron las siguientes variables:

Distancia local: es la cantidad de imágenes en la base de datos las cuales se van a descartar de la consulta, siempre descarta las primeras, esto lo hace con la finalidad de evitar falsas cerraduras de ciclos.

Cantidad de bases de datos: como su nombre lo indica es la cantidad de bases de datos que se utilizaron para esa prueba. Corresponden al número de direcciones cardinales que se utilizan para la IMU. El algoritmo original es el que solamente utiliza una base de datos.

Salto de imágenes: como todas las imágenes fueron tomadas por una cámara a 30 frames por segundo e intentado replicar con fidelidad el comportamiento en tiempo real, se decide si saltar frames o no dependiendo de que si el tiempo de procesamiento del anterior frame fue mayor a 33 ms.

Banco de imágenes: se refiere a si se utilizó el de banco de imágenes Bovisa o el personal.

Similitud (*S*): Es el nivel de parecido que tienen dos imágenes con respecto a sus vectores de Bovw, puede tener valor desde 0 hasta 1, donde 1 es totalmente parecido y 0 es nada parecido.

Alfa (α): Si la similitud es mayor al umbral alfa, el frame se considera un candidato a asociación o también llamado cerradura de ciclos.

Número de palabras visuales: Es la cantidad de palabras visuales presentes en el árbol de vocabulario, en esta tesis es un valor fijo de *K*=10 y *L*=6 dando un total de 995,161 palabras visuales.

Falso positivo: Es necesario distinguir entre falso positivo en identificación de una imagen (gracias a su gran parecido, en el ambiente) y un falso positivo en la cerradura de ciclos (se pierde el robot). El primero puede llevar al segundo si no se toman medidas.

Las pruebas de Bovw entregan un archivo con información de las bases de datos, en esta se incluyen el número de frame, el estado de la detección, el tiempo en milisegundos, la base de datos a la cual se está agregando esta imagen, el número de imágenes que tienen almacenadas cada base de datos y el en el caso de ser cerradura de ciclo incluye el número de imagen de la imagen de referencia, un ejemplo de este archivo se encuentra en la Tabla 4.1. Este archivo tiene la función de facilitar la comprensión de cómo crecen las bases de datos, de verificar las asociaciones realizadas y la de revisar los tiempos de procesamiento por *frame*.

Tabla 4.1 Respuesta del sistema

Frame:	DB	Detección	En DB	Tiempo	LDB1:	LDB2:	LDB3:	LDB4:	LDB5:	LDB6:	LDB7:	LDB8:	Numero
65037	7	1	0	100 ms	10168	836	14499	3441	12723	1905	15837	5539	0
65038	7	1	0	97 ms	10168	836	14499	3441	12723	1905	15838	5539	0
65039	7	1	0	96 ms	10168	836	14499	3441	12723	1905	15839	5539	0
65040	7	0	7	99 ms	10168	836	14499	3441	12723	1905	15840	5539	15200
65041	7	1	0	99 ms	10168	836	14499	3441	12723	1905	15840	5539	0
65042	7	1	0	100 ms	10168	836	14499	3441	12723	1905	15841	5539	0
65043	7	1	0	98 ms	10168	836	14499	3441	12723	1905	15842	5539	0

65044	7	1	0	112 ms	10168	836	14499	3441	12723	1905	15843	5539	0
65045	7	1	0	77 ms	10168	836	14499	3441	12723	1905	15844	5539	0
65046	7	1	0	73 ms	10168	836	14499	3441	12723	1905	15845	5539	0

4.1.1 Pruebas para calibración de valores

En estas pruebas se realizan pequeñas modificaciones a *'α'* y a la distancia local intentando obtener mejores resultados, para validar estos resultados se realiza una prueba con la subsecuencia del banco de imágenes de bovisa, que comprende desde la imagen 19,000 hasta la 22,147 donde se seleccionó este segmento debido a que en él hay una zona con alta repetividad, además de una cerradura de ciclo.

Lo que se busca con estas pruebas es encontrar valores que ayudan a encontrar las imágenes en las que hay una ocurrencia de cerradura de ciclo sin afectar con muchos falsos positivos, además de intentar disminuir los falsos positivos sin quitar las detecciones correctas.

Si se aumenta *'α'* entonces el sistema no permite que aprueben tantos candidatos así que descarta los falsos positivos, pero si es muy elevada también descarta los verdaderos positivos y si se disminuye pasa lo contario.

En un tipo de prueba en específico, se mide el nivel de similitud (*S*) que tienen dos imágenes tomadas consecutivamente y el resultado está entre 0.4 y 0.7 dependiendo la escala de la imagen, también se prueba con imágenes cercanas alrededor de 30 imágenes de distancia unas de otra y los resultados obtenidos están entre los 0.08 y 0.15 en el valor de similitud, de ultimó se prueba con imágenes de zonas completamente diferentes y los resultados obtenidos están entre 0.001 y 0.012.

Para el caso de la distancia local, si esta fuera por ejemplo de 80 entonces esto significa que manda a traer las imágenes de la base de datos menos las 80 primeras, por lo tanto, si se disminuye manda a traer las imágenes más recientes a competir para una cerradura de ciclo ocasionando un falso positivo.

Después de 25 pruebas se llega a la conclusión que en este caso los mejores valores son:

α = 0.22

Distancia local = 120

4.1.2 Prueba 2

Para esta prueba se utiliza una sola base de datos, el banco de imágenes de Bovisa, α =0.22, distancia local de 120 y sin salto de imágenes.

Se obtienen detecciones en 126 *Frames*, 101 en los tres lugares correctos y 25 detecciones que son falsos positivos. Sin embargo, es muy fácil detectar cuando existe un falso positivo ya que aparecen aislados y pasan muchos *Frames* para que vuelva a salir otra detección.

Los tiempos promedio por cada 100 imágenes se pueden ver en la Figura 4.1.

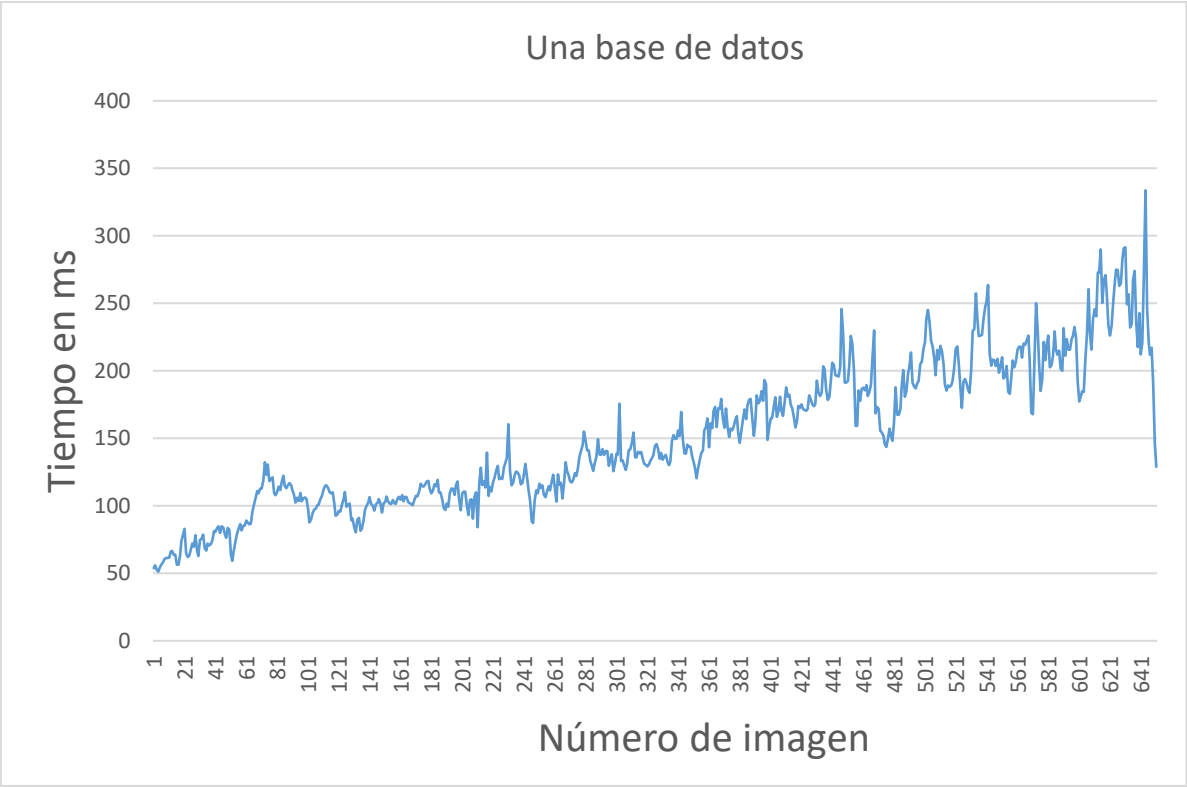

Figura 4.1 Bovisa una base de datos con distancia local de 120, alfa de 0.22 y sin salto

Como se puede observar el tiempo de consulta de la base de datos tiende a subir mientras más imágenes parecidas tenga en su interior, al final se procesaron las 65,045 imágenes y se registraron en la base de datos 64,919 imágenes con un tiempo promedio de 150.20 milisegundos por imagen y un tiempo total de 163 minutos.

4.1.3 Prueba 3

Para esta prueba se utiliza una sola base de datos, el banco de imágenes de Bovisa, α=0.22, distancia local de 120 y con salto de imágenes.

Los tiempos promedio se ven en la Figura 4.2.

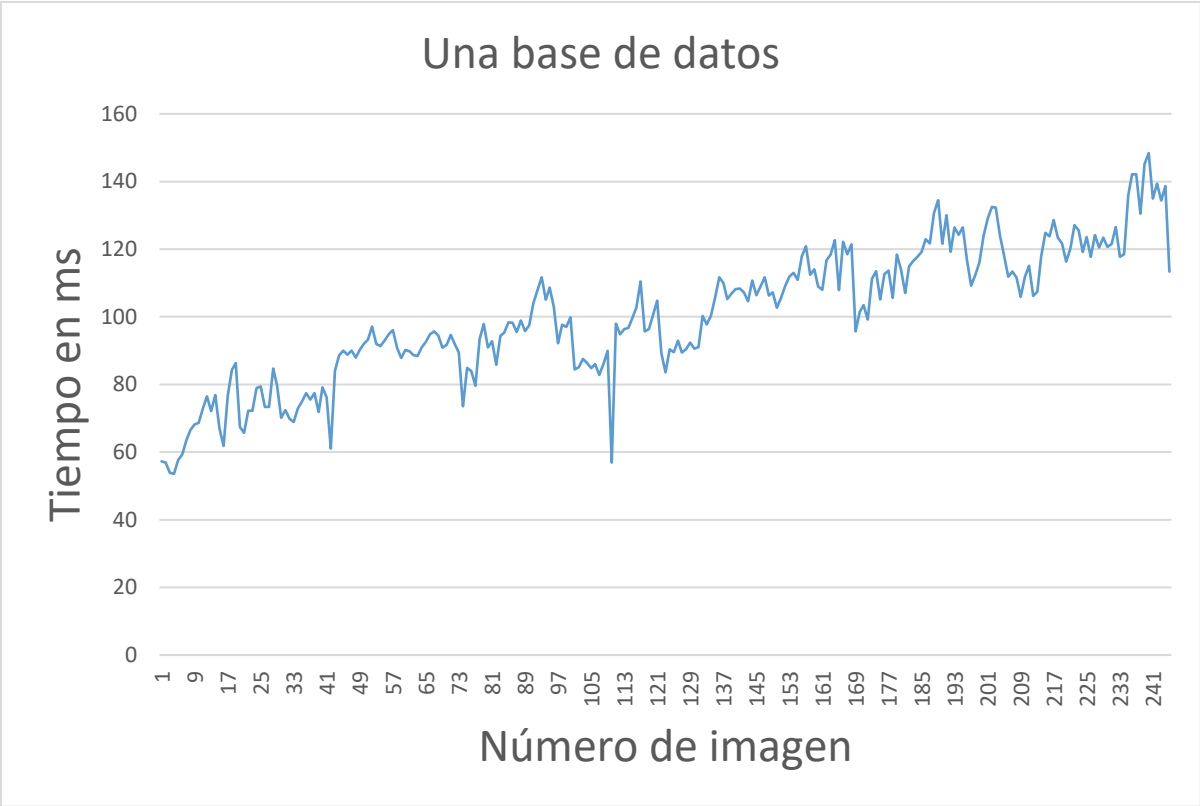

Figura 4.2 Bovisa una base de datos con distancia local de 120, alfa de 0.22 y con salto

En la Figura 4.2 se observa que no se utilizaron todas las imágenes en la base de datos esto debido a que en muchas ocasiones el tiempo de procesamiento de una imagen es superior a los 33 *ms* y por lo tanto se tiene que saltar un frame o más.

Los tiempos de procesamiento van desde los 50 *ms* hasta los 150 *ms* y con promedio de 100 *ms* por imagen y un tiempo total de 41 minutos. Se procesaron 24,493 imágenes de las 65,045 del banco de imágenes y la base de datos tiene 24,428 almacenadas.

Se obtienen detecciones en 65 *Frames*, 53 en los tres lugares correctos y 12 detecciones que son falsos positivos, todos muy alejados de la próxima detección.

4.1.4 Prueba 4

Para esta prueba se utilizan ocho bases de datos, el banco de imágenes de Bovisa, α=0.22, distancia local de 120 y sin salto de imágenes.

Los resultados de esta prueba se muestran en la Figura 4.3.

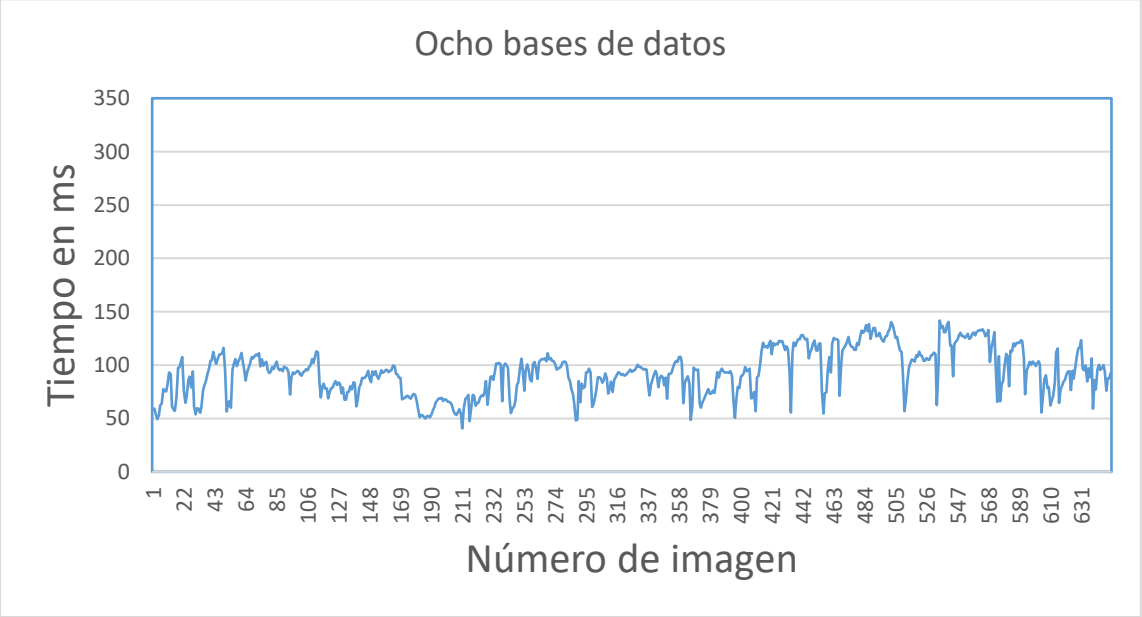

Figura 4.3 Bovisa ocho bases de datos con distancia local de 120, alfa de 0.22 y sin salto

Como se puede observar en la Figura 4.3 el tiempo de procesamiento va creciendo conforme más imágenes parecidas tenga la base de datos, sin embargo, como se divide en ocho bases de datos en lugar de una, el tiempo de procesamiento se mantiene más estable.

Los tiempos de procesamiento van desde los 50 *ms* hasta los 150 *ms* con un tiempo promedio de 93.46 *ms* por imagen y un tiempo total de 102 minutos.

Se procesaron las 65,045 imágenes del banco de imágenes y la distribución de imágenes en cada base de datos es la siguiente:

Longitud de la base de datos 1 (LDB1): 10,168 imágenes

Longitud de la base de datos 2 (LDB2): 836 imágenes

Longitud de la base de datos 3 (LDB3): 14,470 imágenes

Longitud de la base de datos 4 (LDB4): 3,441 imágenes

Longitud de la base de datos 5 (LDB5): 12,723 imágenes

Longitud de la base de datos 6 (LDB6): 1,905 imágenes

Longitud de la base de datos 7 (LDB7): 15,843 imágenes

Longitud de la base de datos 8 (LDB8): 5,539 imágenes

Se obtienen detecciones en 120 *Frames*, 94 en los tres lugares correctos y 26 detecciones que son falsos positivos, todos muy alejados de la próxima detección.

4.1.5 Prueba 5

Para esta prueba se utiliza ocho bases de datos, el banco de imágenes de Bovisa, α=0.22, distancia local de 120 y con salto de imágenes.

Los resultados de esta prueba se muestran en la Figura 4.4.

Los tiempos de procesamiento van desde los 40 ms hasta los 110 ms con un tiempo promedio de 72.11 ms por imagen y un tiempo total de 50 minutos.

Se procesan 49,245 de las 65,045 imágenes del banco de imágenes y la distribución de imágenes en cada base de datos es la siguiente:

Longitud de la base de datos 1 (LDB1): 9,281 imágenes

Longitud de la base de datos 2 (LDB2): 578 imágenes

Longitud de la base de datos 3 (LDB3): 9,661 imágenes

Longitud de la base de datos 4 (LDB4): 1,370 imágenes

Longitud de la base de datos 5 (LDB5): 7,459 imágenes

Longitud de la base de datos 6 (LDB6): 681 imágenes

Longitud de la base de datos 7 (LDB7): 14,853 imágenes

Longitud de la base de datos 8 (LDB8): 5,302 imágenes

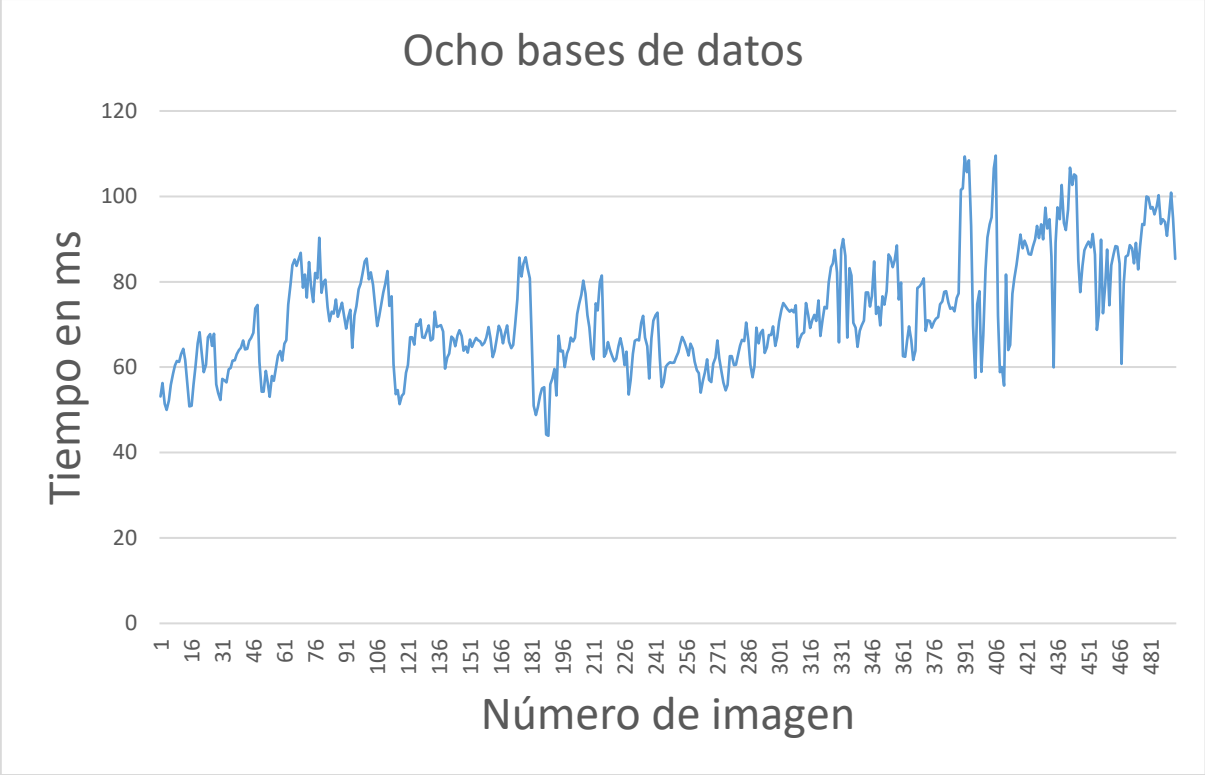

Figura 4.4 Bovisa ocho bases de datos con distancia local de 120, alfa de 0.22 y con salto

Como se puede observar en la Figura 4.4, los tiempos de consulta se comportan de la misma manera que la prueba anterior, pero tiende a subir menos, esto debido a que en la zona con más recurrencia de palabras visuales es la zona en donde se tarda más tiempo en procesar cada *frame*, por lo tanto, en este caso ahí es donde más *frames* se saltan.

Se obtienen detecciones en 60 *Frames*, 50 en los cinco lugares correctos y 10 detecciones que son falsos positivos, todos muy alejados de la próxima detección.

4.1.6 Prueba 6

Para esta prueba se utiliza ocho bases de datos, el banco de imágenes propio, α=0.22, distancia local de 120 y sin salto de imágenes.

Como se puede observar en la Figura 4.5 los tiempos de consulta prácticamente no subieron, esto debido a que son muy pocas imágenes para que puedan subir.

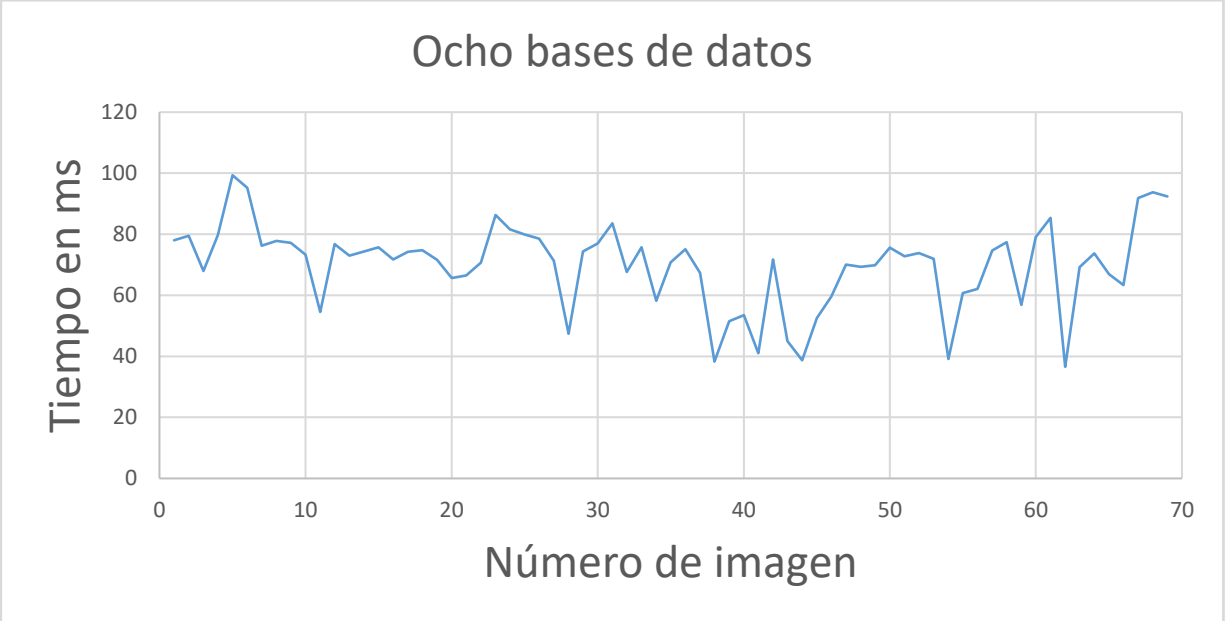

Figura 4.5 Propia ocho bases de datos con distancia local de 120, alfa de 0.22 y sin salto

Los tiempos de procesamiento van desde los 38 ms hasta los 100 ms con un tiempo promedio de 70 ms por imagen y un tiempo total de 8 minutos.

Se procesaron todas imágenes del banco de imágenes y la distribución de imágenes en cada base de datos es la siguiente:

Longitud de la base de datos 1 (LDB1): 1,247 imágenes

Longitud de la base de datos 2 (LDB2): 565 imágenes

Longitud de la base de datos 3 (LDB3): 1,079 imágenes

Longitud de la base de datos 4 (LDB4): 1,721 imágenes

Longitud de la base de datos 5 (LDB5): 646 imágenes

Longitud de la base de datos 6 (LDB6): 346 imágenes

Longitud de la base de datos 7 (LDB7): 602 imágenes

Longitud de la base de datos 8 (LDB8): 477 imágenes

Se obtienen detecciones en 143 *Frames*, 128 en los tres lugares correctos y 15 detecciones que son falsos positivos, igual que en los casos anteriores muy separados entre ellos. En esta secuencia se detectaron muchos frames como cerradura de ciclo debido a que en verdad se están dando muchas vueltas.

4.1.7 Prueba 7

Para esta prueba se utiliza una base de datos, el banco de imágenes propio, α=0.22, distancia local de 120 y sin salto de imágenes.

Los resultados de esta prueba se muestran en la Figura 4.6.

Los tiempos de procesamiento van desde los 41 ms hasta los 122 ms con un tiempo promedio de 84 ms por imagen y un tiempo total de 9.5 minutos.

Se obtienen detecciones en 160 *Frames*, 142 en los cinco lugares correctos y 18 detecciones que son falsos positivos, todos muy alejados de la próxima detección.

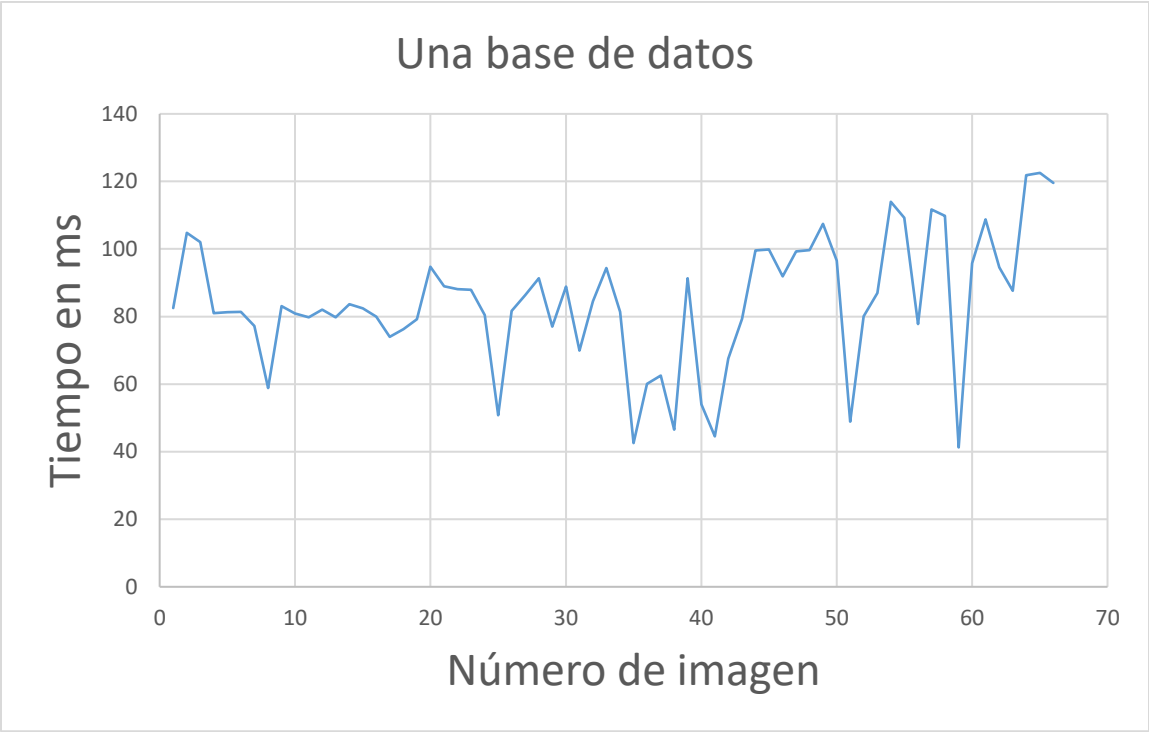

Figura 4.6 Propia una base de datos con distancia local de 120, alfa de 0.22 y sin salto

4.2 Comportamiento de las palabras visuales

Con la finalidad de observar el comportamiento de las palabras visuales se opta por generar un programa que asigne un píxel a cada palabra visual para posteriormente activarlo cuando la palabra aparezca.

Para esta prueba se inicia con la cámara apuntando hacia la pared, captura una imagen, detecta y describe sus puntos característicos, luego los transforma en palabras visuales y activa los pixeles correspondientes, en la Figura 4.7 se puede ver la distribución de la primera imagen.

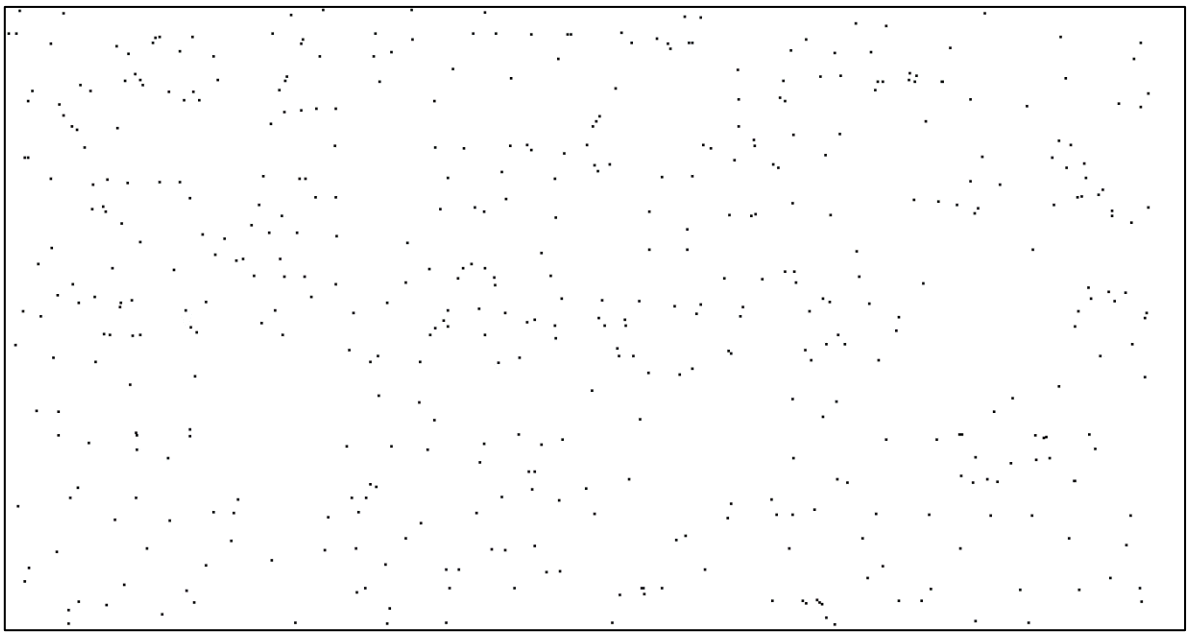

Figura 4.7 Primera iteración

Tras procesar 100 imágenes sin mover la cámara, repitiendo el proceso explicado anteriormente y guardando los puntos que ya se habían activado en las iteraciones anteriores, se obtiene el resultado de la Figura 4.8

Figura 4.8 Después de 100 iteraciones

Como se puede observar aún sin mover la cámara, la baja repetitividad que tienen las palabras visuales hace que se utilicen una gran cantidad de las mismas en la misma toma, a pesar que solo detecta alrededor de 500 puntos característicos por toma.

Finalmente se hacen movimientos bruscos con la cámara durante 2,300 capturas más, para dar un total de 2,400 iteraciones, repitiendo el proceso anterior de captura, detección, descripción, transformación y activación, se obtiene el resultado que está en la Figura 4.9. En este punto la mayoría de las palabras visuales ya son utilizadas, aunque sea una vez, esto muestra que las palabras visuales tienen muy baja repetitividad.

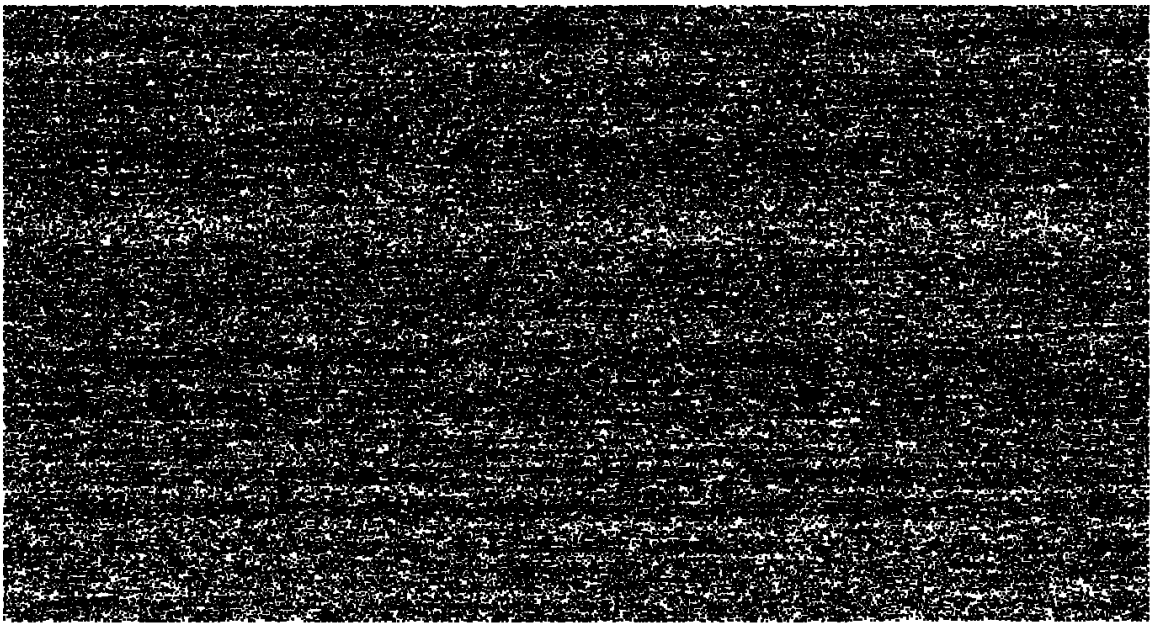

Figura 4.9 2,400 Iteraciones

4.3 Comprobación de cerradura de ciclos

Para verificar los resultados de los experimentos se estudia el archivo que entrega Bovw, buscando el número de imagen de entrada y el número de imagen con la

cual se le relaciona. Un ejemplo extraído de una prueba con el banco de imágenes propio está en la Figura 4.10. En esta Figura se puede observar que las dos imágenes son exactamente del mismo lugar, sólo que una tiene oclusión parcial. Sin embargo, el sistema es capaz de detectarla como cerradura de ciclos correcta.

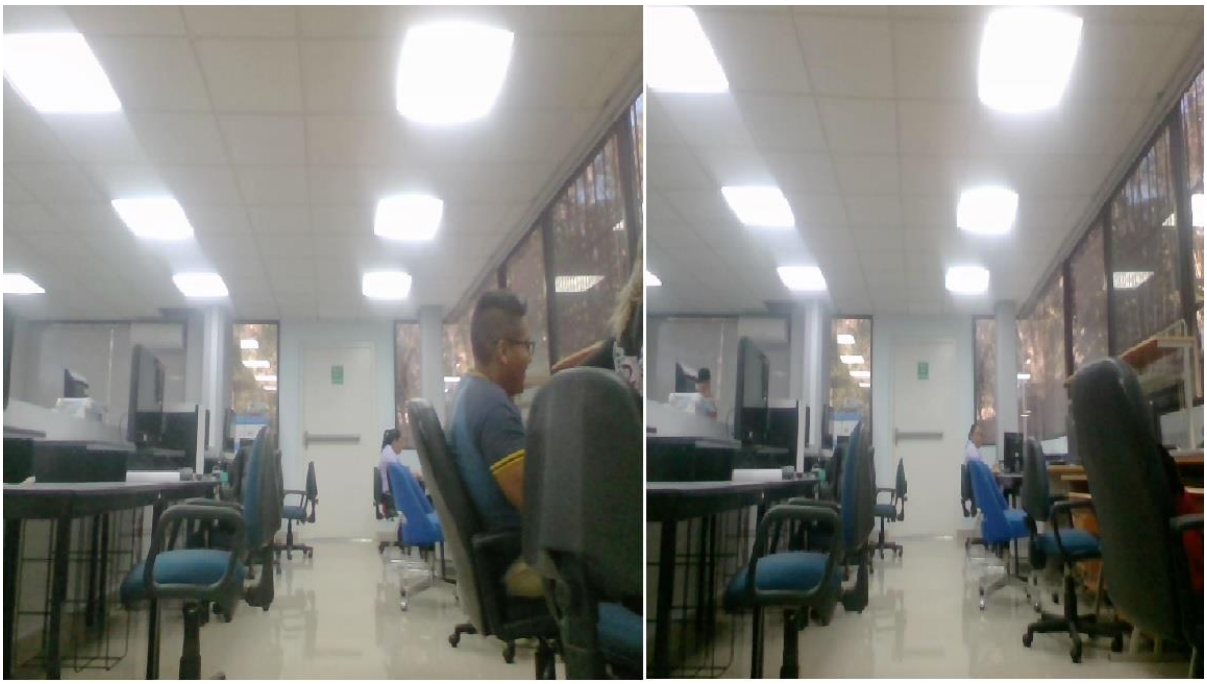

Figura 4.10 Detección correcta

4.4 Análisis de los resultados

A pesar de que las palabras visuales tienen un muy bajo índice de repetitividad sí cumplen su función y permiten detectar las cerraduras de ciclos, claro siempre y cuando Bovw tenga unos parámetros adecuados.

Cuando dividen la base de datos en 8 orientaciones, los tiempos de consulta no tienden a subir como lo hace con una sola base de datos, esto representa una ventaja en grabaciones de más de media hora.

El dividir la base de datos en ocho, baja la cantidad de *frames* donde se detecta cerradura de ciclos, pero eso no significa que no detecte correctamente la zona donde se encuentra una cerradura de ciclo.

5.0 Capítulo 5 Conclusiones

5.1 Conclusiones

En este trabajo se presentó la metodología empleada para la detección de cerraduras de ciclos, así como la experimentación y los resultados.

- Se presenta el método de *Bag of visual Words* utilizando las librerías de Gálvez y Tardós [40], pero realizándole una serie de modificaciones y adaptaciones para que procese más de una base de datos, utilizando ocho direcciones cada una dependiendo de la orientación proporcionada por la IMU.
- En la experimentación se muestra que el utilizar más de una sola base de datos elimina la tendencia en crecimiento de tiempos que tiene Bovw, al consultar su índice inverso en zonas con gran parecido como por ejemplo un pasillo, esto se debe a que no todas las imágenes que tienen un cierto parecido se encuentran en la misma base de datos y al mismo tiempo como tienen una orientación asignada, sólo consulta las imágenes que tienen un cierto parecido con la imagen actual en la orientación vigente.
- *Bag of visual words* tiende a detectar falsos positivos en áreas con gran parecido debido al *aliasing* perceptual, el cual es muy difícil de eliminar, sin embargo, esto no representa un problema ya que es muy fácil detectarlos, una posible solución sería detectar la distancia que tienen dos detecciones de cerradura de ciclos y si su distancia es menor a un umbral tomarla como positiva si no, entonces sería un falso positivo.
- La evaluación de los resultados revela que el utilizar las ocho bases de datos tiende a detectar reconocimientos positivos en menor cantidad de Frames, pero no elimina cerradura de ciclos, por lo tanto, esto no significa una desventaja.

- Cuando se consulta el índice inverso en zonas con gran parecido como por ejemplo un pasillo, es donde se tienen los mayores tiempos de procesamiento.
- Bovw es tolerante a tener objetos en movimiento dentro del área en donde se desenvuelve.
- En la Figura 5.1 se puede observar una comparación directa entre utilizar una base de datos y ocho, en el recorrido de bovisa. En donde se puede distinguir una notable disminución de tiempos después de la segunda mitad del recorrido.

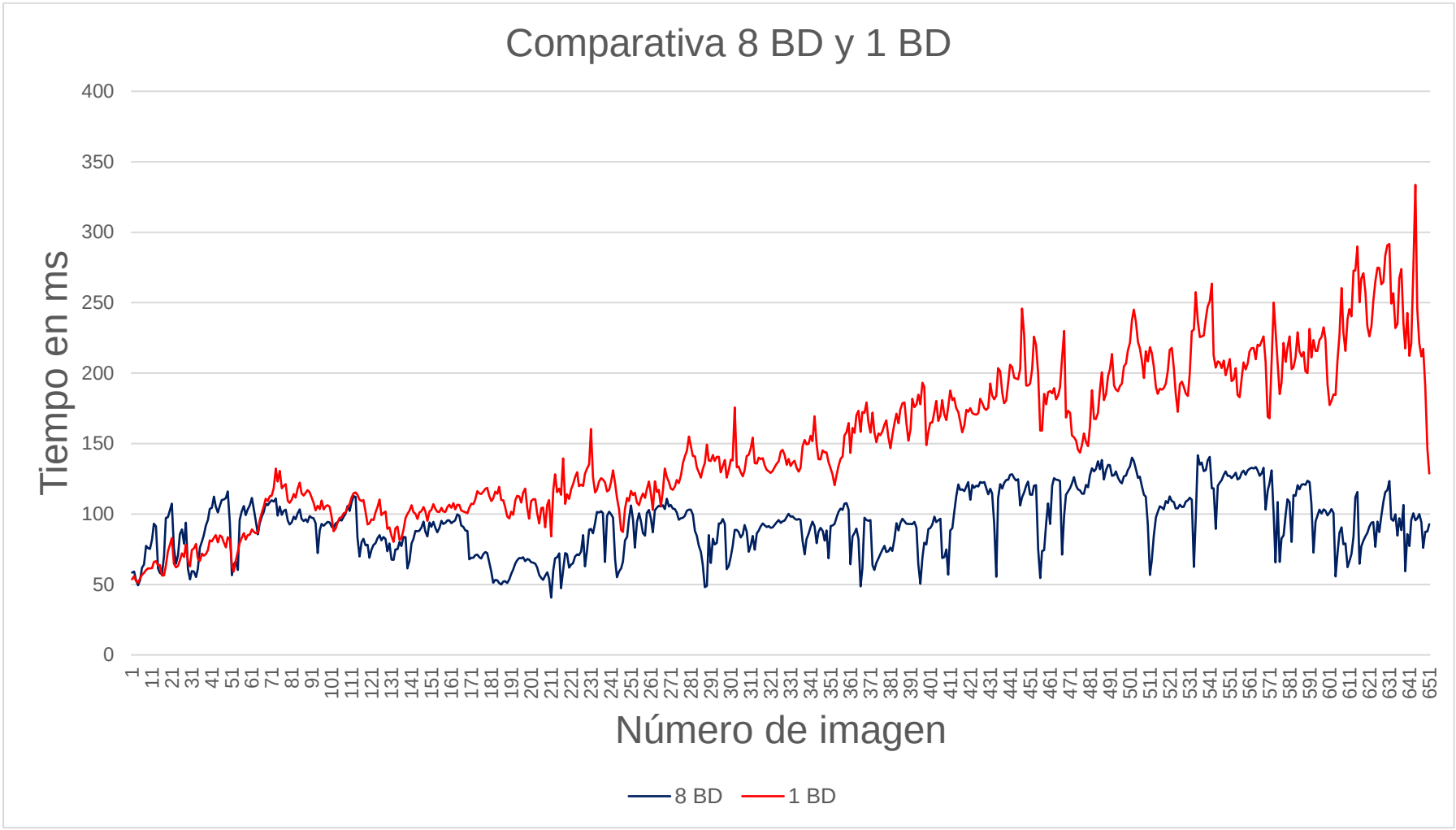

Figura 5.1 Comparación 1BD y 8 BD

5.2 Aportaciones

Las aportaciones que se obtuvieron con la realización de este proyecto fueron:

1.- Respuesta en tiempo real. Se logró la detección de cerraduras de ciclos manteniendo los 10 FPS en un recorrido de más de 65 mil *Frames* con una resolución de 320 x 240 pixeles.

2.- Se propone una novedosa modificación al algoritmo de Bovw al utilizar una IMU para obtener la orientación de un *Frame* y así poder dividir la base de datos en varias bases de datos.

3.- Actualización de librerías para la comunicación entre Arduino y C++ utilizando comunicación de llamada y respuesta.

4.- Se logró reducir el crecimiento de los tiempos de consultas a la base de datos y al reducir estos, el tiempo de procesamiento general también se reduce, como se puede ver en la Tabla 5.1.

Tabla 5.1 Diferencia entre tiempos de consulta

Nombre del banco de imágenes	Una DB	Ocho DB	Diferencia
Bovisa	163 minutos	102 minutos	61 minutos
Propio	9.5 minutos	8 minutos	1.5 minutos

5.3 Trabajos futuros

Los trabajos futuros que se podrían realizar sobre este proyecto son:

La paralelización en hardware especializado, ya que algunos autores logran obtener tiempos de cerradura de ciclos mucho menores pero debido a que utilizan software y hardware especializado y sería interesante ver el comportamiento del sistema en un hardware dedicado.

6.0 Referencias

A continuación se muestran las referencias utilizadas para la elaboración del presente trabajo de investigación.

6.1 Referencias bibliográficas

[1] M. Leonardi and A. Stahl, “Convolutional Autoencoder aided loop closure detection for monocular SLAM ∗,” *IFAC-PapersOnLine*, vol. 51, no. 29, pp. 159–164, 2018.

[2] R. A. de Ingeniería, “odometría.” [Online]. Available: http://diccionario.raing.es/es/lema/odometría. [Accessed: 05-Dec-2019].

[3] R. A. de Ingeniería, “Odometría visual.” [Online]. Available: http://diccionario.raing.es/es/lema/odometría-visual. [Accessed: 05-Dec-2019].

[4] D. Galvez-Lopez and J. D. Tardos, “Real-time loop detection with bags of binary words,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 51–58.

[5] S. Khan and D. Wollherr, “IBuILD: Incremental bag of Binary words for appearance based loop closure detection,” *2015 IEEE Int. Conf. Robot. Autom.*, pp. 5441–5447, 2015.

[6] L. Bampis, A. Amanatiadis, and A. Gasteratos, “Fast loop-closure detection using visual-word-vectors from image sequences,” *Int. J. Rob. Res.*, vol. 37, no. 1, pp. 62–82, 2018.

[7] J. A. F. Pacheco, “Detección de Cierre de Ciclos en el Problema de Localización y Mapeo Simultáneo Visual Monocular,” *Tesis Dr. CENIDET*, 2014.

[8] D. Gálvez-López and J. D. Tardós, “Bags of binary words for fast place recognition in image sequences,” *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, 2012.

[9] G. Castro, P. De Cristóforis, and T. Pire, “Detección y cierre de ciclos en sistemas SLAM basados en visión estéreo,” *4to Concurs. Tesis Licenciatura del Dep. Comput. Fac. Ciencias Exáctas y Nat. Univ. Buenos Aires*, 2016.

[10] P. Newman and K. Ho, “SLAM- Loop closing with visually salient features,” *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2005, pp. 635–642, 2005.

[11] K. Mikolajczyk, K. Mikolajczyk, C. Schmid, and C. Schmid, “A performance evaluation of local descriptors,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, 2005.

[12] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” *Lect. Notes Comput. Sci.*, vol. 3951 LNCS, pp. 404–417, 2006.

[13] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 2564–2571, 2011.

[14] J. B. F. Roblero, “Comparación de Algoritmos de Extracción y Asociación de Rasgos Para Visión Robótica,” *Cenidet*, 2012.

[15] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós, “A comparison of loop closing techniques in monocular SLAM,” *Rob. Auton. Syst.*, vol. 57, no. 12, pp. 1188–1197, 2009.

[16] J. Chacón Murguía, Mario I Sandoval Rodrígez, Rafael Vega Pineda, *Percepción visual aplicada a la robótica*. Alfaomega, 2015.

[17] T. Taylor, *Mapping of Indoor Environments by Robots using Low-cost Vision Sensors*. 2009.

[18] K. D. Harris and M. Recce, “Absolute localization for a mobile robot using place cells,” *Rob. Auton. Syst.*, vol. 22, no. 3–4, pp. 393–406, 1997.

[19] A. González-Jiménez, Javier Ollero, “Estimación de la Posición de un Robot Móvil.” p. https://www.researchgate.net/publication/267222718.

[20] E. P. Mohammad Emal Qzizada, “Mobile robot controlling possibilities of

inertial navigation system," *Procedia Eng.*, vol. 149, pp. 404–4013, 2016.

[21] N. Mechatronics, "Tutorial MPU6050, Acelerómetro y Giroscopio." [Online]. Available: https://naylampmechatronics.com/blog/45_Tutorial-MPU6050-Acelerómetro-y-Giroscopio.html. [Accessed: 25-Nov-2019].

[22] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004, pp. 1063–6919.

[23] B. S. Å. Stefan K.Ericson, "Analysis of two visual odometry systems for use in an agricultural field environment," *Biosyst. Eng.*, vol. 166, pp. 116–125, 2018.

[24] M. M. Duarte Villaseñor and L. Chang Fernandez, "Clasificación de objetos en imágenes usando SIFT," 2010.

[25] T. Tuytelaars and K. Mikolajczyk, "Local Invariant Feature Detectors: A Survey," *Found. Trends® Comput. Graph. Vis.*, vol. 3, no. 3, pp. 177–280, 2007.

[26] Y. V. and R. V. Kushal Vyas, "Using Bag of Visual Words and Spatial Pyramid Matching for Object Classification along with Applications for RIS," *Procedia Comput. Sci.*, vol. 89, pp. 457 – 464, 2016.

[27] Sirishaa, C. S. P. , B Sandhyab, and S. C. Sastry, "Bag-of-Spatial Words(BoSW)Framework for Predicting SAR Image Registration in Real Time Applications," *Procedia Comput. Sci.*, vol. 115, pp. 431–439, 2017.

[28] H. Eraqi, "heraq," *Bag of Visual Words for Image Classification*, 2017. [Online]. Available: http://heraqi.blogspot.com/2017/03/BoW.html. [Accessed: 21-Oct-2019].

[29] C. García and I. Gómez, "Algoritmos de aprendizaje: knn & kmeans," in *Inteligencia en Redes de Telecomuncicación*, 2012, pp. 1–8.

[30] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *Int. J. Rob. Res.*, vol. 27, no. 6, pp. 647–

665, 2008.

[31] M. Cummins and P. Newman, “Appearance-only SLAM at Large Scale with FAB-MAP 2.0,” *Int. J. Rob. Res.*, vol. 30, no. 9, pp. 1100–1123, 2011.

[32] N. Kejriwal, S. Kumar, and T. Shibata, “High performance loop closure detection using bag of word pairs,” *Rob. Auton. Syst.*, vol. 77, pp. 55–65, 2016.

[33] A. Bokovoy and K. Yakovlev, “Original loop-closure detection algorithm for monocular vSLAM,” *Lect. Notes Comput. Sci.*, vol. 10716 LNCS, pp. 210–220, 2018.

[34] L. Taylor, E. Miller, and K. R. Kaufman, “Static and dynamic validation of inertial measurement units,” *Gait Posture*, vol. 57, no. May, pp. 80–84, 2017.

[35] S. Heo, J. Cha, and C. G. Park, “Monocular Visual Inertial Navigation for Mobile Robots using Uncertainty based Triangulation,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 2217–2222, 2017.

[36] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999, vol. 2, pp. 1150–1157 vol.2.

[37] K. Grauman and B. Leibe, “Visual Object Recognition,” *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 5, no. 2, pp. 1–181, Apr. 2011.

[38] N. Cornelis and L. Van Gool, “Fast scale invariant feature detection and matching on programmable graphics hardware,” in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2008, pp. 1–8.

[39] P. di. Milano, “Dataset ‘Bovisa (outdoor + mixed),’” 2008. [Online]. Available: http://www.rawseeds.org/rs/datasets/view/7%0A. [Accessed: 15-May-2019].

[40] D. Galvez-Lopez and J. D. Tardos, “DBoW2,” *github*, 2012. [Online]. Available: https://github.com/dorian3d/DBoW2. [Accessed: 03-Dec-2018].

[41] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Object retrieval with

large vocabularies and fast spatial matching," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[42] Sivic and Zisserman, "Video Google: a text retrieval approach to object matching in videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*, 2003, pp. 1470–1477 vol.2.

[43] D. Nist and H. Stew, "Scalable Recognition with a Vocabulary Tree," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, pp. 2161–2168, 2006.

[44] M. E. Qazizada and E. Pivarčiová, "Mobile robot controlling possibilities of inertial navigation system," in *Procedia Engineering*, 2016, vol. 149, no. June, pp. 404–413.

[45] Wikipedia, "Cardinal direction." [Online]. Available: https://en.wikipedia.org/wiki/Cardinal_direction. [Accessed: 24-Oct-2019].

[46] T. Kevin, "ADAFRUIT," *Magnetometer Calibration*, 2014. [Online]. Available: https://learn.adafruit.com/ahrs-for-adafruits-9-dof-10-dof-breakout/magnetometer-calibration.

[47] C. Cadena, D. Gálvez-López, F. Ramos, J. D. Tardós, and J. Neira, "Robust place recognition with stereo cameras," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 5182–5189.

[48] G. Csurka, C. R. Dance, L. Fan, J. Willamoswski, and B. Cédric, "Visual Categorization with Bags of Keypoints," *Work. Stat. Learn. Comput. Vision, ECCV*, pp. 1--22, 2004.

[49] K. Townsend, "Adafruit 9-DOF IMU Breakout User's," *Adafruit*, 2014. [Online]. Available: https://learn.adafruit.com/adafruit-9-dof-imu-breakout/introduction. [Accessed: 02-Dec-2018].