SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Tecnológico Nacional de México

Centro Nacional de Investigación y Desarrollo Tecnológico

Tesis de Maestría

Mapeo visual de trayectorias 3D para robots móviles con cámaras RGB-D

presentada por

Lic. Luis García Tovar

comorequisito para la obtención del grado de

Maestro en Ciencias de la Computación

Director de tesis

Dr. José Ruiz Ascencio

Cuernavaca, Morelos, México. Junio de 2018.

cenidet
Centro Nacional de Investigación y Desarrollo Tecnológico

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Centro Nacional de Investigación y Desarrollo Tecnológico

Cuernavaca, Morelos a 20 de junio del 2018
OFICIO No. DCC/187/2018

Asunto: Aceptación de documento de tesis

DR. GERARDO V. GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO
PRESENTE

Por este conducto, los integrantes de Comité Tutorial del **Lic. Luis García Tovar**, con número de control M14CE058, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis profesional titulado "**Mapeo visual de trayectorias 3D para robots móviles con cámaras RGB-D**" y hemos encontrado que se han realizado todas las correcciones y observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

DIRECTOR DE TESIS

Dr. José Ruiz Ascencio
Doctor en Ciencias
5009035

REVISOR 1

M.C. Gerardo Reyes Salgado
Maestro en Ciencias de la Computación
2493370

REVISOR 2

Dr. Dante Mújica Vargas
Doctor en Comunicaciones y Electrónica
09131756

REVISOR 3

Dr. Manuel Mejia Lavalle
Doctor en Ciencias Computacionales
8342472

C.p. M.T.I. María Elena Gómez Torres - Jefa del Departamento de Servicios Escolares.
Estudiante
Expediente

NACS/lmz

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLOGICO NACIONAL DE MÉXICO

Centro Nacional de Investigación
y Desarrollo Tecnológico

Cuernavaca, Mor., 22 de junio de 2018
OFICIO No. SAC/281/2018

Asunto: Autorización de impresión de tesis

LIC. LUIS GARCÍA TOVAR
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 **"Mapeo visual de trayectorias 3D para robots móviles con cámara RGB-D"**, 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.T.I. María Elena Gómez Torres .- Jefa del Departamento de Servicios Escolares.
Expediente

GVGR/mcr

Dedicatoria

Le dedico este trabajo a:

Mi familia y Catalina Vázquez
Gracias por su apoyo incondicional.

Agradecimientos

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

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

Agradezco al Dr. José Ruiz Ascencio asesor de este trabajo, por todo el apoyo brindado, por sus valiosos consejos y tiempo, los cuales contribuyeron a la terminación de este trabajo.

Agradezco a mi comité revisor: Dr. Manuel Mejía Lavalle, Dr. Dante Mujica Vargas, M.C. Gerardo Reyes Salgado, por sus comentarios y observaciones que me ayudaron a culminar este proyecto. Gracias por la confianza y el trato amable que siempre me brindaron.

Finalmente agradezco en general a todos los que de forma directa o indirecta contribuyeron a la realización de este trabajo.

Resumen

Tanto en la odometría visual como en la mecánica existe un error en el cálculo de la trayectoria, este problema ocasiona que se tenga una ubicación errónea con respecto al lugar real donde se encuentra el robot, dicho error se ha intentado resolver durante varios años con algunos métodos para la localización mediante el uso de algoritmos tales como: ORB-SLAM, SVO, PTAM, OKVIS entre otros, sin embargo, el problema solo se ha resuelto parcialmente.

En este trabajo se utilizó odometría visual debido a que la odometría mecánica genera una acumulación mayor del error en las trayectorias, causada por la irregularidad del terreno. La tecnología usada fue el sensor RGB-D de Kinect.

El proceso de creación de trayectorias 3D es en tiempo real. La decisión del método a utilizar se tomó con base a un estudio comparativo de algunos modelos odométricos. El sistema de odometría se apoya sobre las tecnologías de visión artificial OpenCV, PCL y OpenNi, entre otras con los cuales el sistema fue desarrollado. Por propiedades del sensor RGB-D de Kinect, las trayectorias son generadas para entornos de interiores.

Abstract

Both in visual and in mechanical odometry there are errors in the calculation of the trajectory. This problem causes a robot to have an erroneous location with respect to its real position. This error has been tried to solve for several years with some localization methods using algorithms such as: ORB-SLAM, SVO, PTAM, OKVIS among others, however, the problem has only been partially solved. In this work, visual odometry was used because mechanical odometry generates a greater accumulation of error in the trajectories, caused by the irregularity of the terrain. The technology used was the Kinect RGB-D sensor. The process of creating 3D paths is in real time. The decision of the method to be used was based on a comparative study of some odometric models. The odometry system is based on artificial vision technologies such as: OpenCV, PCL and OpenNI among others, with which the system was developed. Due to properties of the Kinect RGB-D sensor, trajectories are generated for interior environments.

Contenido

Índice de Figuras

Índice de Tablas

Glosario

CCD: Dispositivo de carga acoplada (*charge-coupled device*).

Ego-motion: Movimiento propio.

OV: Odometría visual.

OM: Odometría mecánica.

SLAM: Localización y Modelado Simultáneos (*Simultaneous Localization And Mapping*).

SVD: Descomposición de valores singulares (*Singular Values Descomposition*).

Mapa: Conjunto de imágenes consecutivas que conforman una escena.

Mapeo: Conjunto de elementos de datos entre dos modelos distintos.

NaN: No es un número (*Not an Number*)

RGB: *Red, Green, Blue.*

RGB-D: *Red, Green, Blue, Depth.*

GPS: Sistema de posicionamiento global (*Global Positioning System*).

Monocular: Una sola cámara.

Binocular: Dos o más cámaras.

Capítulo 1 Introducción

En este capítulo se da un marco introductorio y se describen los conceptos fundamentales necesarios para la comprensión de este trabajo. Se describe la problemática a resolver en esta tesis, además se da una breve descripción de la organización de la tesis.

El uso de robots cada día es más común en entornos de la vida cotidiana de un ser humano, gracias al desarrollo de procesadores más pequeños y poderosos así como de sensores con mayores prestaciones y más económicos. Los robots pueden realizar tareas que son demasiado peligrosas o agotadoras para que los seres humanos las lleven a cabo; sus implementaciones más comunes son en el ámbito doméstico, militar, minería, exploración submarina, industria espacial, hospitales, vigilancia etc.

Actualmente el uso de cámaras de video en robótica se ha incrementado gracias a que este tipo de sensores brindan mayor información a un menor costo con respecto a otros como el láser, ultrasonido o Sistemas de Posicionamiento Global (GPS). Los sistemas computacionales que emplean cámaras, conocidos como sistemas de Visión Artificial, se destacan por que permiten reconocer objetos, rostros de personas y pueden operar en ambientes no estructurados y dinámicos.

Uno de los objetivos de la visión artificial es conseguir que un robot sea capaz de analizar una secuencia de escenas y mediante ese análisis poder identificar en que parte de la escena se encuentra además generar un mapa de su entono, a esto se le conoce como *Localización y Mapeo Simultáneos*, **SLAM**, *Simultaneous Localization and Mapping* [Durrant 2006].

SLAM se puede determinar de distintas formas, las más comunes son: por *odometría mecánica*, que estudia el movimiento con base a las vueltas de las ruedas de un robot pero esto puede generar error debido a terrenos irregulares. Otro enfoque es la *odometría visual*: en esta se estima el movimiento mediante la información capturada mediante cámaras, evitando el error de la odometría mecánica. La odometría visual se puede generalizar en dos grandes grupos: métodos basado en características y métodos directos [Scaramuzza 2014].

De los métodos basados en características los más utilizados son **Harris**[Harris 1988], **FAST**[Rosten 2006], **SIFT**[Lowe 2004], **SURF**[Bay 2006], **ORB**[Rublee 2011], por otro lado el método directo principalmente utiliza toda la información de la imagen el cual fue propuesto por Iraní [Iraní 2010].

La odometría visual ha logrado grandes avances en el área de navegación robótica ya que permite identificar la posición y orientación de un robot en un ambiente desconocido por él mismo. Los distintos métodos de odometría visual pueden utilizar diferentes cámaras, en nuestro caso nos auxiliamos de la cámara Kinect para Xbox 360.

La localización mediante información visual ha jugado un papel muy importante en el área de reconocimiento del terreno que permite responder la pregunta ¿Dónde estoy? a lo largo de la historia se han creado e implementado diversas formas de resolver el problema de localización mediante el uso de sensores y algoritmos.

Es necesaria la localización de un robot no solo en un entorno 2D sino también se debe considerar la tercera dimensión, ya que en este trabajo se desea tratar el caso general de trayectorias 3D para robots móviles, también se considera utilizar la cámara de Kinect con la cual se obtiene información de profundidad.

La información de sensor Kinect se limita a interiores ya que dadas las condiciones físicas del sensor se requieren espacios pequeños. El funcionamiento del sistema fue implementado para que sea capaz de trabajar en tiempo real.

1.1. Marco conceptual

En esta sección se describen algunos conceptos claves que son empleados a lo largo de este documento, con la finalidad de una mejor comprensión del tema.

1.1.1. Odometría Visual (OV)

Se puede definir la ***odometría visual*** como el 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. A la técnica donde el movimiento del sistema de cámaras es solidario con el movimiento del vehículo se le conoce como "*ego-motion*" (*movimiento propio*) [Burger 2010].

1.1.2. Localización

La localización en sistemas de odometría visual se considera un paso básico que se realiza con una secuencia de imágenes. Se calcula la pose de la cámara entre la imagen actual y la imagen anterior. Así uniendo cada uno de estas posiciones individuales consecutivas se puede construir la trayectoria completa del desplazamiento que realiza un robot.

La localización se puede generalizar en dos grandes grupos: Método directo y Métodos basados en características, pero recientemente surge un nuevo método llamado Método semi-directo el cual es una combinación de los dos métodos anteriormente mencionados [Scaramuzza 2014].

1.1.3. Método directo

La localización en este método se obtiene directamente de la intensidad de los valores de la imagen, explotando toda la información de la imagen al mismo tiempo Figura 1.1.

Otro enfoque es dividir la imagen en decenas de regiones y analizar las regiones por separado, pero siempre considerando la información de la imagen en su totalidad.

Figura 1.1 Imagen utilizada para métodos directos.

1.1.4. Métodos Basados en Características

Este método principalmente consiste en extraer un conjunto de características sobresalientes de un conjunto de imágenes, las cuales pueden ser puntos destacados, líneas o esquinas como se muestra en la Figura 1.2. Al obtener descripciones de pequeñas regiones de la imagen estas permiten, mediante un proceso de *correspondencia* (ver sección 1.1.9), analizar el comportamiento de desplazamiento que se genera en un conjunto de imágenes consecutivas.

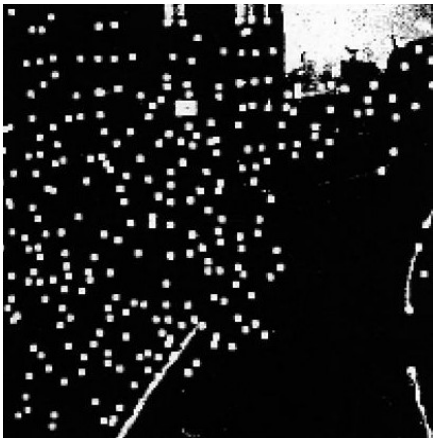


Figura 1.2 Imagen utilizada para métodos basados en características.

1.1.5. Método Semi-Directo

La idea fundamental del Método Semi-Directo es extraer características, pero la imagen debe ser dividida en cientos de pequeñas ventanas.

Cuando se obtienen 2 imágenes consecutivas de la misma escena, se aplica un método de extracción de características con el cual se obtienen puntos destacados, para cada punto se genera una región vecina llamada ventana, a estas características se les da seguimiento, posteriormente se realiza una transformación analizando las características de dos

imágenes consecutivas para obtener la nueva posición de la cámara [Scaramuza 2014]. En la Figura 1.3 se muestra una imagen representativa de la obtención de la pose.

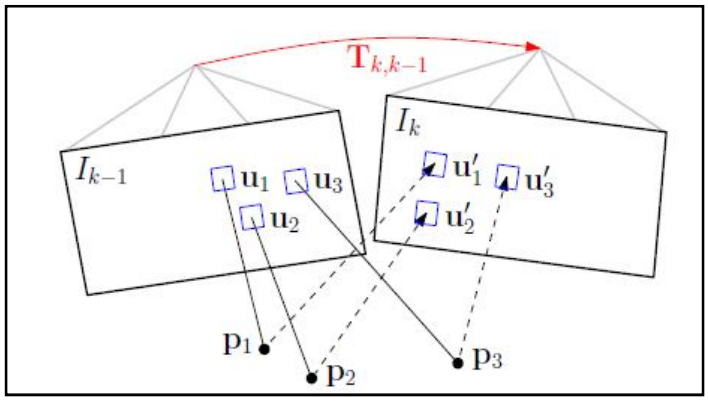


Figura 1.3 Obtención de pose utilizando método semi-directo.

1.1.6. Sensor RGB-D

Este tipo de sensor puede capturar imágenes en formato RGB, agregando un componente más que representa la profundidad.

Actualmente en el mercado existen distintos productos comerciales tales como *Primesense Reference Design*, *Microsoft Kinect*, *Asus Xtion Pro*, *SoftKinetic DS311*, *Creative Technology's Senz3D*. En la Figura 1.4 se muestran algunos de los sensores RGB-D más comunes.

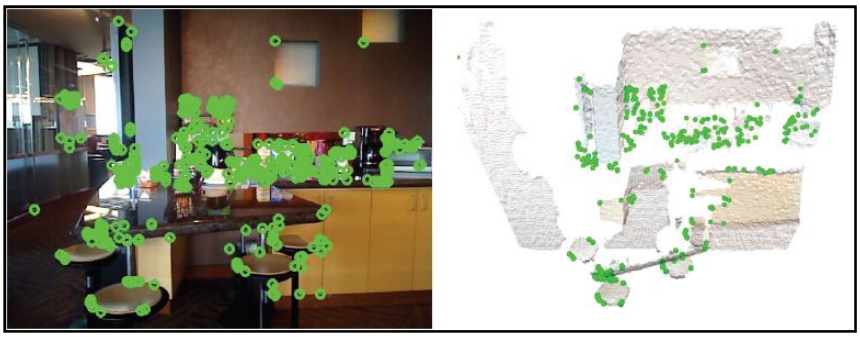
a)Primesense Reference Design **b) Microsoft Kinect** **b) Asus Xtion Pro**

Figura 1.4 Sensores RGB-D.

1.1.7. Puntos destacados

Los puntos destacados de una imagen son aquellos que mediante la descripción de su vecindad obtienen un área distintiva en la imagen debido a cambios bruscos en la intensidad del color, estos pueden ser esquinas, líneas o puntos; como se muestra en la Figura 1.5.

Mediante algoritmos de extracción de características se pueden obtener estos puntos destacados; algunos de los métodos más utilizados en la literatura son Harris, FAST, SIFT, SURF, ORB entre otros más [Peter 2012].



(a) **(b)**

Figura 1.5 Puntos destacados: a) Imagen RGB, b) Imagen de profundidad.

1.1.8. Correspondencia de puntos

La correspondencia de puntos es posible cuando se tiene los conjuntos de puntos de dos imágenes consecutivas de la misma escena, y que estos puntos se encuentren en ambas imágenes Figura 1.6. Lo que se pretende es identificar si un punto de la primera imagen se encuentra en la segunda. Así a cada punto se le puede dar un seguimiento y con ello obtener información del movimiento que la cámara está realizando.

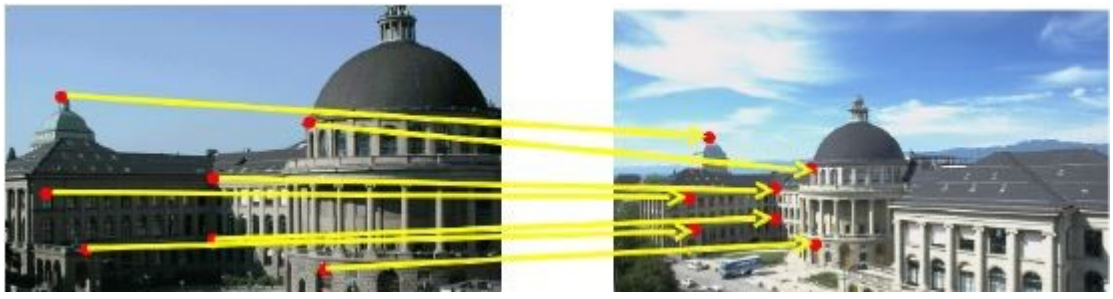


Figura 1.6 Correspondencias de puntos entre imágenes.

1.1.9. SVD

La ***descomposición en valores singulares*** **SVD** (*Singular Value Decomposition*) es una factorización matricial, la cual es aplicada en esta tesis para generar una matriz de rotación 3D entre dos conjuntos de puntos.

La descomposición de valores singulares toma una matriz rectangular de datos (definida como A, donde A es una matriz n x p en la que las n son las filas y p columnas. El teorema SVD establece:

$$A_{n*p} = U_{n*n} S_{n*p} V_{p*p}^{T}$$

Donde las columnas de U son los vectores singulares izquierdos, S son los valores singulares y V^T son los renglones que representan los valores singulares derechos. SVD representa una expansión de los datos originales en un sistema de coordenadas donde la matriz de covarianza es diagonal [Meyer 2000].

1.1.10. PCL

Cuando se adquiere la información de una escena por medio de técnicas 3D, como por ejemplo lidar, triangulación activa, tiempo de vuelo, estereoscopia, entre otras, se obtiene una gran cantidad de información, denominada nube de puntos, que debe ser interpretada y procesada para determinar ciertos detalles de la escena. Aunque esta operación se podría realizar con diferentes programas tales como MatLab®, LabVIew®, entre otros, esto conllevaría una alta carga computacional, además de que no se tendría un estándar o método definido. Es allí donde toma alta relevancia la herramienta conocida como PCL (Point Cloud Library) [Andrés 2012].

PCL es una herramienta de software disponible gratuita para imágenes 2D / 3D, puede ser empleada por dispositivos de video. PCL contiene 15 módulos que permiten generar estructuras, filtrar información, estimar características, reconstruir superficies, registro, ajustar modelos, segmentar entre otras funciones, La Figura 1.7 muestra los módulos disponibles en PCL [PCL 2012].

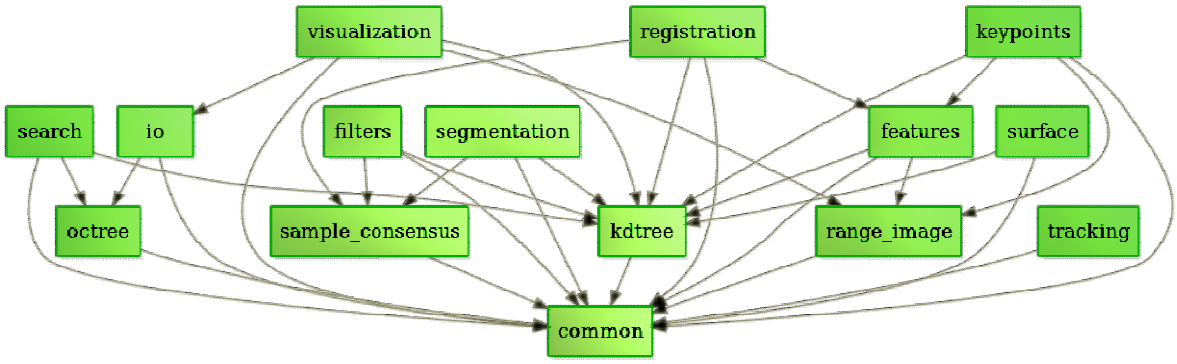


Figura 1.7 Bibliotecas de PCL.

1.2. Organización de la tesis

Este documento está estructurado en cinco capítulos. El primer capítulo presenta conceptos con la finalidad de introducir al lector al tema presentado en esta tesis.
El capítulo dos proporciona información de los trabajos relacionados y estado del arte.
El capítulo tres se detalla el análisis y diseño del sistema.
El capítulo cuatro se aborda la implementación del sistema.
Por último el capítulo cinco se exponen conclusiones y comentarios finales.

Capítulo 2 Trabajos Relacionados

En este capítulo se describe el estado del arte de las técnicas utilizadas en cada una de las etapas siguiendo la arquitectura general de un sistema de odometría visual. Así también se mencionan los trabajos relacionados a esta tesis que se han realizado en Cenidet.

2.1. Antecedentes

En el Centro Nacional de Investigación y Desarrollo Tecnológico (Cenidet) se han desarrollado algunos temas de investigación relacionados a esta tesis de Maestría. A continuación se da una breve descripción de ellos.

En 2011 González presentó un sistema que permite a un robot móvil terrestre navegar y esquivar obstáculos e identificar huecos por donde este puede acceder en un ambiente simulado de exteriores [González 2011].

Se analizaron diversas técnicas de visión binocular, las cuales se utilizan para la construcción de un sistema de visión para robots móviles, abarcando desde la obtención de las imágenes hasta la odometría incluyendo la correspondencia y reconstrucción pero la trayectoria que este generó no fue trazada. También construyó un robot que contenía: dos cámaras, una laptop, dos motores de corriente directa, una tarjeta de potencia con interfaz USB y una plataforma con ruedas, con el cual realizo su experimentación.

Más tarde en 2013 Gómez abordó el tema de: “Fusión de información de pose en robots móviles con visión” donde fusionó la información de odometría mecánica y visual. Se realizó un estudio comparativo entre los métodos de fusión de información de odometría mecánica y odometría visual para la estimación de pose de un robot móvil denominados Intersección de Covarianzas y el Filtro de Partículas, estos métodos fueron aplicados en tramos cortos.

Los resultados de ese trabajo concluyeron que la odometría mecánica funcionó mejor que la odometría visual, y los métodos de fusión que se utilizaron, mejoraron de forma notable las estimaciones de pose al combinar ambas mediciones odométricas, finalmente se concluyo que el método de Filtro de Partículas tuvo un mejor desempeño que la Intersección de Covarianzas [Gómez 2013].

El método de odometría que realizo Peñaloza tenía como propósito determinar la pose del robot con respecto a una vista que había sido tomada previamente para poderse alinear con la vista actual, replicando así la trayectoria visual. Los recorridos de las trayectorias eran circuitos cerrados, una vez que el robot llegaba al final de la trayectoria el cuadro siguiente pertenecía al primer cuadro del trayecto [Peñaloza 2014].

En 2015 Vergara aborda el mismo tema que Peñaloza pero esta vez las trayectorias debían de ser abiertas. Al igual que Peñaloza, Vergara realizó odometría visual pero enfocada a trayectorias abiertas, así que el trayecto constaba de un principio y un final, cuando el robot llegaba al final el proceso terminaba [Vergara 2015].

Se abordaron varios trabajos de visión artificial utilizando cámaras binoculares y monoculares, pero la tecnología de las cámaras RGB-D no había sido explorada en el grupo de inteligencia artificial de Cenidet, donde se abordaran temas de odometría visual.

Finalmente en 2016 Zárate tuvo como objetivo principal identificar el comportamiento de flujo de puntos destacados e identificar objetos en movimiento, aquí la odometría visual que el robot generaba mientras realizaba una trayectoria no era rescatada, sino explorar las técnicas de detección de objetos en movimiento.

Los puntos destacados al paso de una secuencia de imágenes tienen un movimiento o desplazamiento predecible pero este desplazamiento se relaciona como el robot va avanzando, en cambio los puntos de objetos que tiene movimiento dentro de la escena se comportan distintos al conjunto global de puntos destacados de la escena [Zárate 2016].

2.2. Estado del Arte

Durante algunos años se ha trabajado dentro de la navegación robótica con la finalidad de que un robot sea capaz de explorar su entorno e identificar en que parte del mapa se encuentra, surgiendo así el término SLAM (*Simultaneous Localization and Mapping*) Localización y Mapeado Simultáneos, la cantidad de métodos es diversa ya que estos influyen en si están diseñados para robot terrestres marinos o aéreos o el tipo de cámara que estos utilizan.

La base de los métodos de SLAM es la odometría, ya que esta se encarga de brindar información que le permita al robot localizarse dentro de un entorno que se está generando simultáneamente.

La ***odometría visual*** (OV) es el proceso de estimación del movimiento propio de un agente (por ejemplo, vehículo, humano, robot) utilizando sólo la entrada de una o múltiples cámaras conectadas al sistema. Los dominios de aplicación incluyen la robótica, la informática portátil y realidad aumentada [Scaramuzza 2011].

La Figura 2.1 muestra un esquema general de los componentes más importantes que rodean a la odometría.

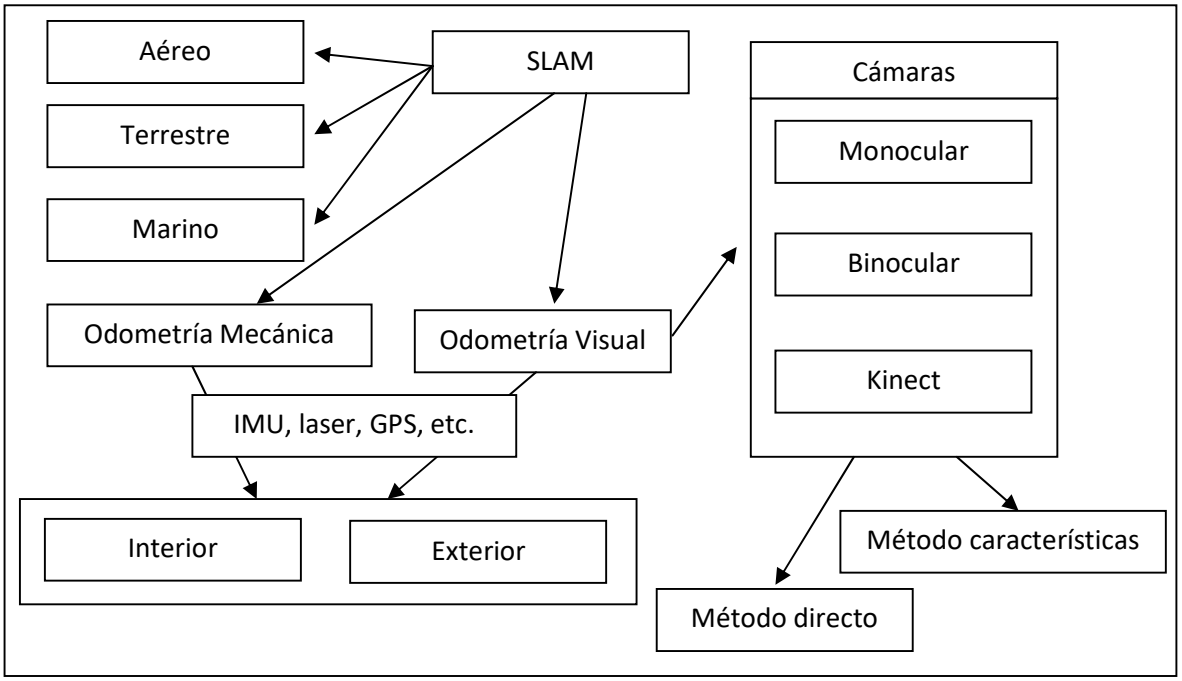


Figura 2.1 Marco general de odometría.

En 2004 Nister introduce el término odometría visual [Nister 2004] la cual se enfoca a resolver el problema de localización de un robot. La odometría visual ha logrado grandes avances en el área de navegación robótica ya que permita identificar la posición y orientación de un robot en un ambiente que se desconoce.

El término fue elegido por su similitud con *odometría mecánica*, que estima el movimiento de un vehículo mediante la integración del número de vueltas de las ruedas considerando el tiempo. Asimismo, la OV opera con la estimación de la pose del vehículo a través de probar los cambios que inducen movimiento en las imágenes de las cámaras a bordo de un robot.

Los componentes principales que contiene un sistema de odometría visual se mencionan en la Figura 2.2 ya sea utilizando métodos basados en características o métodos directos, se necesita un conjunto de imágenes consecutivas para analizar: los métodos basados en características detectan rasgos relevantes de cada una de ellas, utilizando algún método que permita extraer características como puntos, líneas o esquinas, con estas características se realizan correspondencias entre ellas, finalmente se estima el movimiento, utilizando las diferencias de las correspondencias, se puede establecer una trayectoria con esta información mientras que para los métodos directos se utiliza toda la imagen pero a un costo computacional mayor.

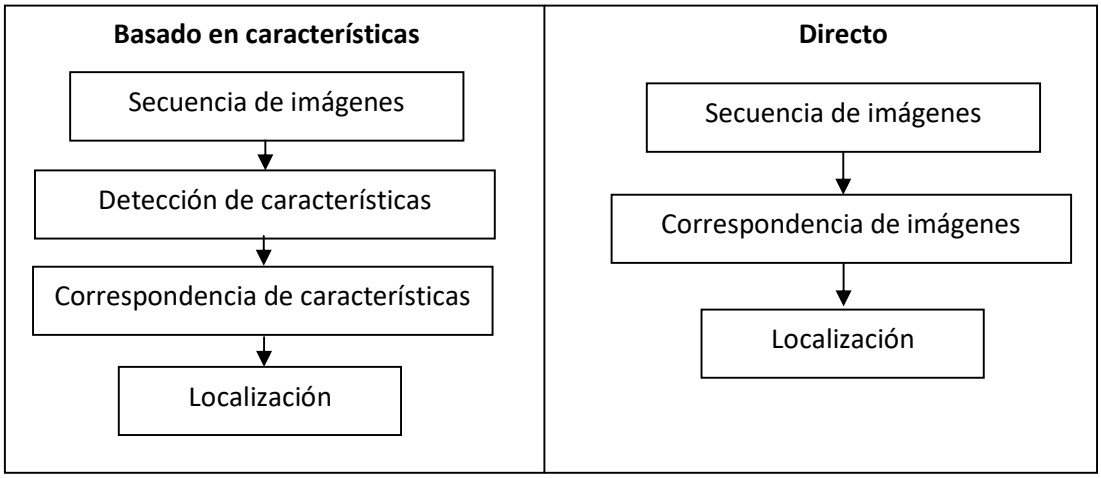


Figura 2.2 Componentes principales de un sistema de Odometría Visual.

Para que la OV trabaje con eficacia, debe haber suficiente iluminación en el ambiente y una escena estática con textura de rasgos suficientemente separables para permitir el movimiento. Por otra parte, los cuadros consecutivos deben ser capturados para garantizar que tengan suficiente solapamiento entre escenas.

La ventaja de la odometría visual con respecto a la odometría mecánica es que no se ve afectada por el deslizamiento de las ruedas en un terreno desigual u otras condiciones adversas. Se ha demostrado que en comparación con odometría mecánica, la OV proporciona estimaciones más precisas de la trayectoria, con error de posición que varía de 0.1 a 2%. Esta capacidad hace de la OV un complemento interesante para la odometría mecánica y a otros sistemas de navegación, tales como el sistema de posicionamiento global (GPS), unidades de medición inercial (IMU), y odometría láser. La OV tiene suma importancia en ambientes que carecen de adquisición de información por medio del GPS, por ejemplo: al interior de minas, bajo el agua o zonas aéreas [Scaramuzza 2014].

La odometría visual se puede dividir principalmente en dos enfoques: monocular o binocular. Esto depende de la cantidad de cámaras que se estén utilizando, relacionado a esto también las técnicas para obtención de pose como métodos directos o basados en características se pueden aplicar independientemente de la cantidad de cámaras que se estén utilizando.

Odometría Visual Monocular

Lo odometría visual monocular se apoya principalmente de una cámara con la cual se obtiene información que permite generar la trayectoria por la cual se va desplazando el robot [Engel 2012], estos métodos puede ser utilizado en interiores [Engel 2013] o exteriores [Pizzoli 2014]. Se pueden utilizar distintos tipos de vehículos ya sean terrestres [Xingshuai 2016], aéreos [Achtelik 2014], marinos [Casagrande 2013].

El uso de una sola cámara permite disminuir el peso del robot y evita el proceso de calibración de cámaras, disminuyendo posibles errores si la calibración se ha hecho mal. La principal desventaja de la odometría visual monocular es que se debe desarrollar un método eficiente para calcular la profundidad de la escena para que este pueda trabajar en tiempo real.

Con la introducción de los MAVs como un vehículo no tripulado, se desarrollan nuevas técnicas de localización utilizando *drones*. Shaojie en 2012 propuso un método donde se determinaban las regiones de mayor exploración en función de la evolución de una ecuación diferencial estocástica que simulan la expansión de un sistema de partículas con la dinámica de Newton [Shaojie 2012].

En el caso del desplazamiento de MAVs en entornos de interiores, se debe ser sumamente cuidadoso para que el drone no se impacte con objetos Figura 2.3 por lo cual se recomienda auxiliarse de algún sensor. De acuerdo a la instalación y posición de la cámara a bordo del drone esta permitirá obtener imágenes para generar la trayectoria.

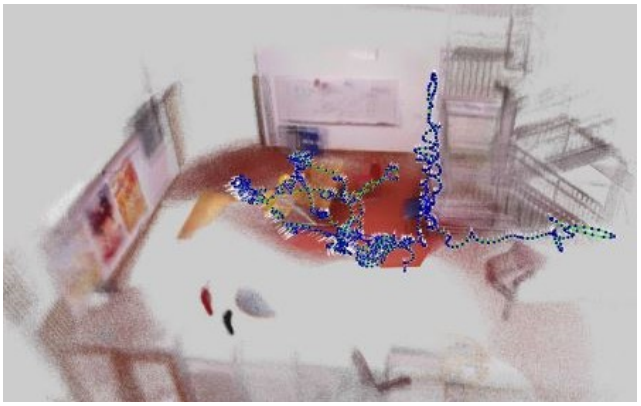


Figura 2.3 Trayectoria 3D de un MAV en interior.

Otro trabajo utilizando cámaras monoculares en el cual permite la exploración de ambientes en tres dimensiones de interiores usando micro vehículos aéreos (MAVs) fue propuesto por Scaramuzza en 2014 abordando el tema de localización utilizando odometría visual monocular donde reporto: un algoritmo de *odometría visual monocular semi-directa* SVO que es preciso, robusto, y más rápido que los métodos actuales [Scaramuzza 2014]. Ya que el enfoque semi-directo elimina la necesidad de extracciones de características costosas y técnicas robustas de correspondencias para la estimación de movimiento.

La principal diferencia con los métodos actuales y SVO, es que SVO no requiere la extracción de características durante la estimación de movimiento que constituye la mayor parte del tiempo en los algoritmos existentes. Además, dada la precisión con la cual el algoritmo trabaja se limita a la extracción de 120 características ya que estas son suficientes para una correcta descripción de la escena mientras que algunos otros algoritmos requieren entre 160 a 220 características. La razón por la que se puede realizar un seguimiento fiable de la cámara con menos características es el uso de filtros de profundidad.

En la Figura 2.4 se muestra el funcionamiento del algoritmo de odometría visual semi-directa el cual está dividido en dos partes: estimación del movimiento y mapeo.

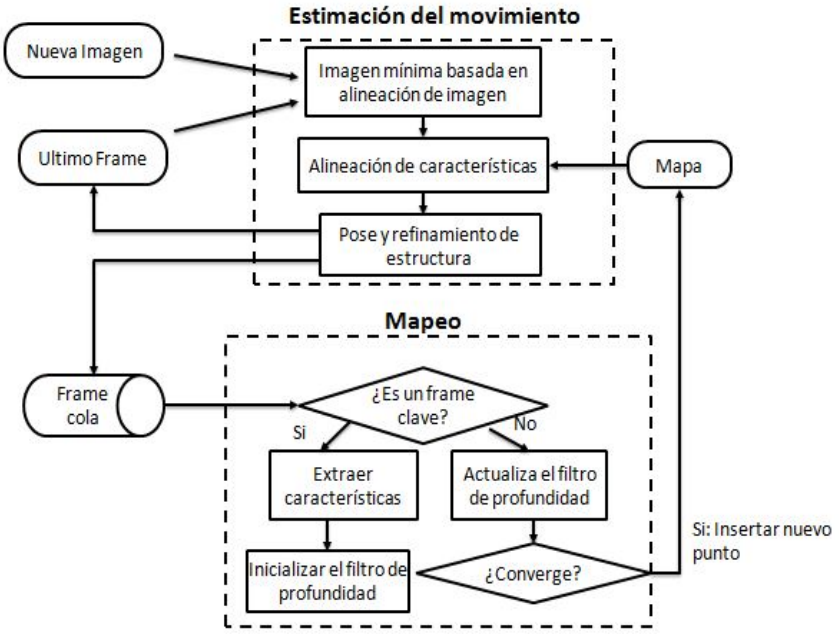


Figura 2.4 Localización y mapeo monocular.

Odometría Visual Estéreo

La odometría visual estéreo hace uso de dos o más cámaras para la obtención de coordenadas que localizan al robot en un entorno, para hacer uso de este tipo de odometría se tienen que calibrar las cámaras del sistema para un correcto funcionamiento [Jürgen 2012].

Al igual que en la odometría monocular esta puede ser obtenida utilizando métodos de extracción de características [Vogiatzis 2011] o métodos directos [Usenko 2016], ya sea en interiores o exteriores.

Odometría Visual usando cámara RGB-D

Al utilizar cámaras RGB-D para generar trayectorias se minimiza el tiempo de adquisición de datos de profundidad ya que estas debido a sus capacidades físicas los proporcionan.

La obtención de trayectorias utilizando cámaras RGB-D minimiza los costos computacionales comparados con utilizar una o más cámaras pero está limitada a uso en interiores ya que en un entorno de exterior la información de profundidad no puede ser calculada. La Figura 2.5 muestra una trayectoria y el mapa que se genero utilizando Kinect.

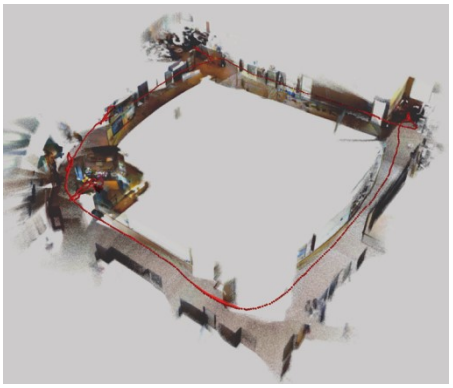


Figura 2.5 Mapa y trayectoria usando Kinect.

En 2012 Jürgen presento un enfoque para el mapeo y localización simultáneos (SLAM), para cámaras RGB-D como el Kinect de Microsoft. Este sistema calcula al mismo tiempo la trayectoria del Kinect y genera un modelo 3D del medio ambiente. La Figura 2.6 muestra un esquema de los componentes para la odometría utilizando Kinect. Se presentan las características clave del enfoque y se evalúa su desempeño a fondo, utilizando bases de datos de publicaciones recientes, que incluyen grandes conjuntos de secuencias de diferentes escenas a distintas velocidades de la cámara y variaciones en las condiciones de iluminación. En particular, se evalúa la precisión, robustez, y el tiempo de procesamiento de tres descriptores de características diferentes (SIFT, SURF, y ORB). Los experimentos demuestran que el sistema puede hacer frente con firmeza a conjuntos de datos difíciles en escenarios de interiores cotidianos sin dejar de ser lo suficientemente rápido.

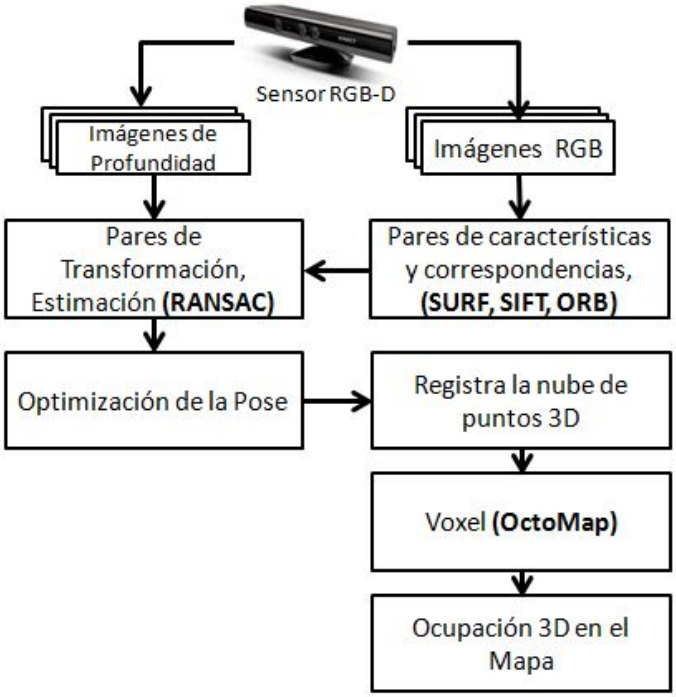


Figura 2.6 Sistema SLAM.

En 2014 Agarwal analizó e implementó el método de odometría visual semi-directa, donde se reemplazo la cámara monocular por un sensor RGB-D.

Los experimentos se realizaron en ambientes donde existía presencia de humo, aun en presencia de humo se puede obtener información de profundidad y con esto generar la trayectoria del robot [Agarwal 2014].

2.3. Discusión

Los artículos citados en el estado del arte de este tema de tesis, explican conceptos fundamentales que aportan al desarrollo de esta tesis, dando las bases para entender todos los conceptos que rodean a este tema, por ejemplo: la localización, mapeo, como se pueden determinar cada uno de los anteriores y las distintas formas que existen para obtenerlos.

A pesar de que el método semi-directo se propone como un método que no realiza costosas extracciones de puntos destacados, analizando el algoritmo que propusieron este si hace uso de métodos de extracción de puntos.

Tabla 1. Comparativa de artículos mencionados en el estado del arte.

Articulo	Sub-problema	Técnica	Observación
Odometría visual, Parte 1: Primeros 30 años y fundamentos (Tutorial) [Scaramuzza, 2011]	Introducción al tema de odometría visual.	Extracción de características, correspondencias, localización.	No se evalúan los métodos mencionados
Una evaluación del sistema SLAM utilizando sensor RGB-D [Jürgen 2012]	Mapeo y Localización.	Extracción de características mediante SIFT, localización y mapeo simultáneos.	Falta comparación con método SVO
SVO: Odometría visual seme- directa [Scaramuzza, 2014]	Odometría Visual Semi-Directa.	Extracción de características utilizando SVO, filtro probabilístico de profundidad.	No utiliza sensor para obtener la profundidad
Exploración 3D autónoma en interiores con Micro-vehículos aéreos [Burger 2010]	Exploración de ambientes.	Ecuaciones diferenciales estocásticas.	El proceso es más tardado que el reportado en SVO
Odometría visual en ambientes ocluidos por humo[Agarwal, 2014]	Odometría visual Semi-Directa utilizando cámara RGB-D.	Extracción de características utilizando SVO, FOVIS	Solo se realiza la comparativa con un método

Capítulo 3 Análisis del problema y propuesta de solución

En este capítulo se realiza un breve repaso de la problemática a abordar en esta tesis, y de la propuesta de solución del problema. Se realiza un análisis de las herramientas de software utilizadas en cada una de las etapas siguiendo la arquitectura general de un sistema de odometría visual así como del hardware.

3.1. Análisis comparativo de métodos de odometría visual

Se han desarrollado varios métodos de Odometría Visual desde que se introdujo el termino por Nister en 2004 [Nister 2004], con la finalidad de obtener la posición y orientación de la cámara o robot en un entorno.

Los métodos de odometría visual pueden ser clasificados de distintas maneras por ejemplo la cantidad de cámaras que utilizan: monocular o estéreo [Scaramuzza 2011] y estos a la vez pueden hacer uso de distintas técnicas para obtener la pose ya sea por medio de métodos de extracción de puntos destacados o métodos directos y auxiliarse con sensores extras como IMU, GPS o instrumentos de medición. La Tabla 2 presenta algunos métodos de odometría visual.

Tabla 2. Comparación de métodos de Odometría Visual.

Método	Descripción	Ventaja	Desventaja
PTAM [Klein 2007]	(Parallel Tracking and Mapping for Small AR Workspaces). Este método principalmente utiliza extracción de puntos, correspondencias y *Bundle Adjustment* entre *Key-Frames*.	En cada iteración del algoritmo puede ser re-calculada la pose, obteniendo mejores resultados.	Al ser un proceso iterativo la pose debe de ser calculada varias veces para poder establecer un resultado aceptable.
PCL [Radu 2011] [PCL 2016]	(Point Cloud Library). Es un conjunto de librerías disponibles como software libre, con las cuales se puede calcular la odometría de una cámara.	Las clases son fáciles de manipular para la creación de proyectos que permiten obtener la pose de la cámara.	La configuración e integración a IDE es complicada y en ocasiones incompatible.
Correspondencias 3D a 3D [Arun 1987] [Scaramuzza 2011]	Método que utiliza extracción de puntos destacados para luego obtener correspondencias de estos puntos y mediante el cálculo de centroide para cada imagen obtener la traslación de la cámara.	Debido a que se utiliza una cámara RGB-D es fácil obtener de cada punto destacado su posición en x, y, z.	Si el trayecto se realiza a alta velocidad se pierde el seguimiento de puntos lo que llevaría a errores de cálculo en la trayectoria.
SVO: Fast Semi-Direct Monocular Visual Odometry [Scaramuzza 2014]	Este método es uno de los más complejos ya que aplica distintos enfoques de odometría como métodos basados en características y métodos directos e integra *Bundle Adjustment*.	El utilizar distintas técnicas hace de este método uno de los más complejos y precisos.	El cálculo del proceso de este método es excesivo, ya que aplica técnicas complicadas como extracción de puntos, calculo de profundidad.

3.2. Tecnologías implementadas

En esta sección se describe el hardware y software utilizado en la implementación del sistema de odometría visual.

3.2.1. Hardware

- Sensor de Kinect para Xbox360.
- Computadora Dell, procesador Intel Core i5-337 1.8GHz, x64, 4 GB RAM.

3.2.2. Software

- Windows Software Development Kit for Windows 7.
- Microsoft .NET Framework 4.0.
- Microsoft Visual Studio 2012.
- SDK Kinect 1.8.
- OpenNI 2.1.0.4 (Este software permite la comunicación entre la cámara de Kinect y la PC, para visualizar la información de la cámara en la PC).
- OpenCV 3.0.0 (Este conjunto de librerías ofrece funciones de visión artificial, como extracción de puntos destacados, correspondencias, etc.).
- PCL 1.6.0 (Contiene un conjunto de librerías con las cuales se puede manipular la información de profundidad).

3.2.3. Sistema Operativo

- Windows 10, x64.

3.3. Localización

Para obtener la pose de la cámara es necesario realizar una serie de pasos previos que permitan obtener una relación entre el mundo real y la información de la imagen de la cámara como la calibración. La localización se puede obtener tanto del desplazamiento de la cámara como la orientación, para ello se empleara el algoritmo de correspondencia entre conjunto de datos en 3D, y para la orientación el método de SVD.

3.3.1. Features_extraction

OpenCV ofrece varios métodos de extracción de puntos destacados a continuación se enlistan.

- "FAST" – **FastFeatureDetector**
- "STAR" – **StarFeatureDetector**
- "SIFT" – **SIFT**
- "SURF" – **SURF**
- "ORB" – **ORB**

- "BRISK" – **BRISK**
- "MSER" – **MSER**
- "GFTT" – **GoodFeaturesToTrackDetector**
- "HARRIS" – **GoodFeaturesToTrackDetector**
- "Dense" – **DenseFeatureDetector**
- "SimpleBlob" – **SimpleBlobDetector**

3.3.2. FlannBasedMatcher

OpenCV ofrece algunas librerías para correspondencias de puntos entre estas se destacan *Matcher* y *Brute-Force*.

3.3.3. SVD

Dentro de OpenCV se encuentra el modulo de SVD al cual se le entrega una matriz que contiene información de todo el conjunto de puntos destacados y se obtiene como resultado una matriz de rotación relacionada a los conjuntos de puntos consecutivos que se está analizando.

3.3.4. PCLVisualizer

PCLVisualizer es la *clase* de visualización completa de PCL. Aunque es más complejo de usar que el *CloudViewer*, también es más potente, ofreciendo características tales como mostrar nubes de puntos, formas de dibujo y múltiples vistas.

3.3.5. StatisticalOutlierRemoval

Este modulo disponible en PCL ofrece la eliminación de valores atípicos en la nube de puntos.

La exploración láser suelen generar conjuntos de datos de nubes de puntos de densidades variables. Además, los errores de medición conducen a valores extremos dispersos que corrompen aún más los resultados. Esto complica la estimación de las características locales de la nube de puntos tales como las normales de superficie o los cambios de curvatura, dando lugar a valores erróneos, que a su vez pueden provocar fallas en el registro de nubes de puntos. Algunas de estas irregularidades pueden ser resueltas realizando un análisis estadístico en la vecindad de cada punto y recortando aquellas que no cumplen ciertos criterios. Esta eliminación de valores atípicos se basa en el cálculo de la distribución de las distancias entre puntos y vecinos en el conjunto de datos de entrada.

Para cada punto, se calcula la distancia media de él a todos sus vecinos. Al suponer que la distribución resultante es Gaussiana con una media y una desviación estándar, todos los puntos cuyas distancias medias están fuera de un intervalo definido por la media de

distancias globales y la desviación estándar pueden considerarse como valores atípicos y recortados del conjunto de datos.

3.3.6. Sensor_origin_ (Eigen :: Vector4f)

Especifica la posición de adquisición del sensor (origen / traslación) esta clase es utilizada para comparar el método de odometría implementado contra la información de traslación de pose que PCL ofrece.

3.3.7. Sensor_orientation_ (Eigen :: Quaternionf)

Específica la posición de adquisición del sensor (orientación) esta clase es utilizada para comparar el método de odometría implementado contra la información de rotación de pose que PCL ofrece.

3.4. Generar mapa del entorno

El mapa de un entrono se puede generar utilizando información que se ha extraído en la obtención de la pose; con ello se construye el entorno. PCL ofrece la librería ***registration*** con la cual se pueden unir varias nubes de puntos a una solo estructura obteniendo así un mapa del entorno.

El proceso de generación de mapa puede ser muy costoso computacionalmente si se utiliza toda la información de las nubes de puntos, para evitar este alto costo computacional se recomienda filtrar las nubes de puntos, obteniendo así una representación significativa de la escena con la menor cantidad posible de puntos.

3.5. Captura de imagen

Existen una gran cantidad de cámaras que pueden ser utilizadas para aplicarlas al área de navegación robótica, ya sean cámaras monoculares o binoculares. Para la realización de esta tesis se utilizo la cámara RGB-D de Kinect.

3.5.1. Grados de libertad de Kinect

El sensor Kinect es capaz de obtener información en tres dimensiones de un entorno, se debe establecer un marco de referencia para visualizar la información que la cámara ha obtenido Figura 3.1. Esta referencia funciona como coordenadas de inicio para el robot, usando esta información para comenzar a desplazar el robot en un entorno de interiores 3D.

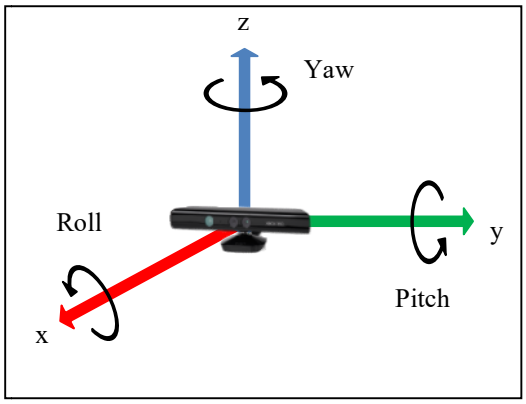


Figura 3.1 Marco de referencia de Kinect.

Al desplazar la cámara en un espacio (mundo) esta cambia su posición y orientación en dicho espacio, pero se debe tomar en cuenta que la cámara en sí tiene su propio marco de referencia Figura 3.2.

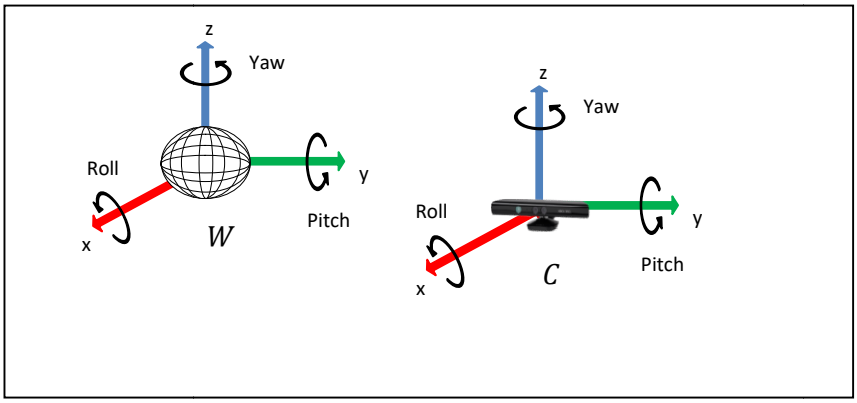


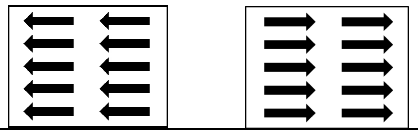
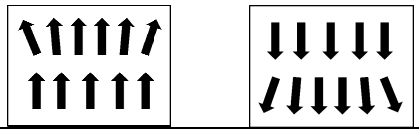
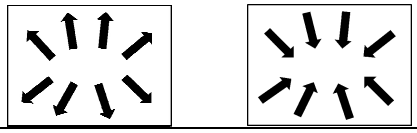
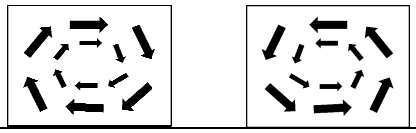
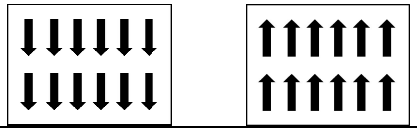
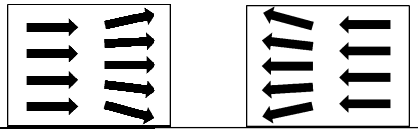
Figura 3.2 Marco de referencia de Kinect con respecto al mundo.

Para construir una trayectoria es necesario identificar la posición de la cámara C respecto al origen del mundo W, en un lapso de tiempo la cámara modificara su posición, al dar seguimiento a estos desplazamientos consecutivos se genera una recorrido de la cámara o trayectoria.

3.5.2. Movimientos disponibles para la cámara

Los movimientos que se pueden realizar con el sensor Kinect se muestran en la Tabla 3 que utiliza el marco de referencia de la Figura 3.1, estos pueden ser combinados para obtener desplazamientos compuestos.

Tabla 3. Comportamiento de puntos destacados.

	Movimiento en y (desplazamiento derecha-izquierda)
	Rotación en *Pitch* (arriaba-abajo)
	Movimiento en x (desplazamiento adelante-atrás)
	Rotación en *Roll*
	Movimiento en z (desplazamiento arriba-abajo)
	Rotación en *Yaw* (derecha-izquierda)

Aunque todos los movimientos están disponibles para la cámara Kinect, se restringen algunos al momento de generar las trayectorias, por ejemplo todas las trayectorias son hacia adelante quedando fuera el desplazamiento hacia atrás. En la Sección 5 Experimentación, se define que movimientos fueron utilizados para cada experimento.

Kinect ofrece gran cantidad de herramientas que pueden ser utilizadas para problemas de visión artificial, una de las principales características de Kinect es que genera imágenes en 3D, estas son muy útiles para problemas de odometría visual, debido a que se reduce el tiempo de cálculo cuando se establecen los valores de profundidad de las imágenes.

Por la naturaleza del sensor es probable tener perdida de información en profundidad, ya que en la imagen de profundidad pueden contener valores que no pueden ser establecidos, esto se debe a distintas problemáticas como: reflejos en superficies, orillas en objetos, saltos en profundidad, generando así valores de tipo NaN.

3.6. Propuesta de solución

En esta tesis se implementó una metodología de odometría visual que permitió determinar la pose de un robot o cámara que reduce el error de localización con respecto a la odometría mecánica, para ello se realizó un análisis de algunos métodos de odometría, se tomó un método y fue implementado, se utilizó la cámara del Kinect para aprovechar la información de profundidad que este ofrece.

3.7. Metodología de solución

Se plantearon cinco fases para la metodología de solución: En la primera fase se introducen los conceptos que fundamentan la investigación en esta tesis, la segunda fase se especifican los problemas que se deben desarrollar, la tercera fase está compuesta por la implementación de los distintos módulos del sistema, la cuarta fase se consideran las pruebas al sistema, finalmente la última fase consiste en la documentación que se generó de este trabajo. La Figura 3.3 se muestra cada una de las fases de la propuesta de solución.

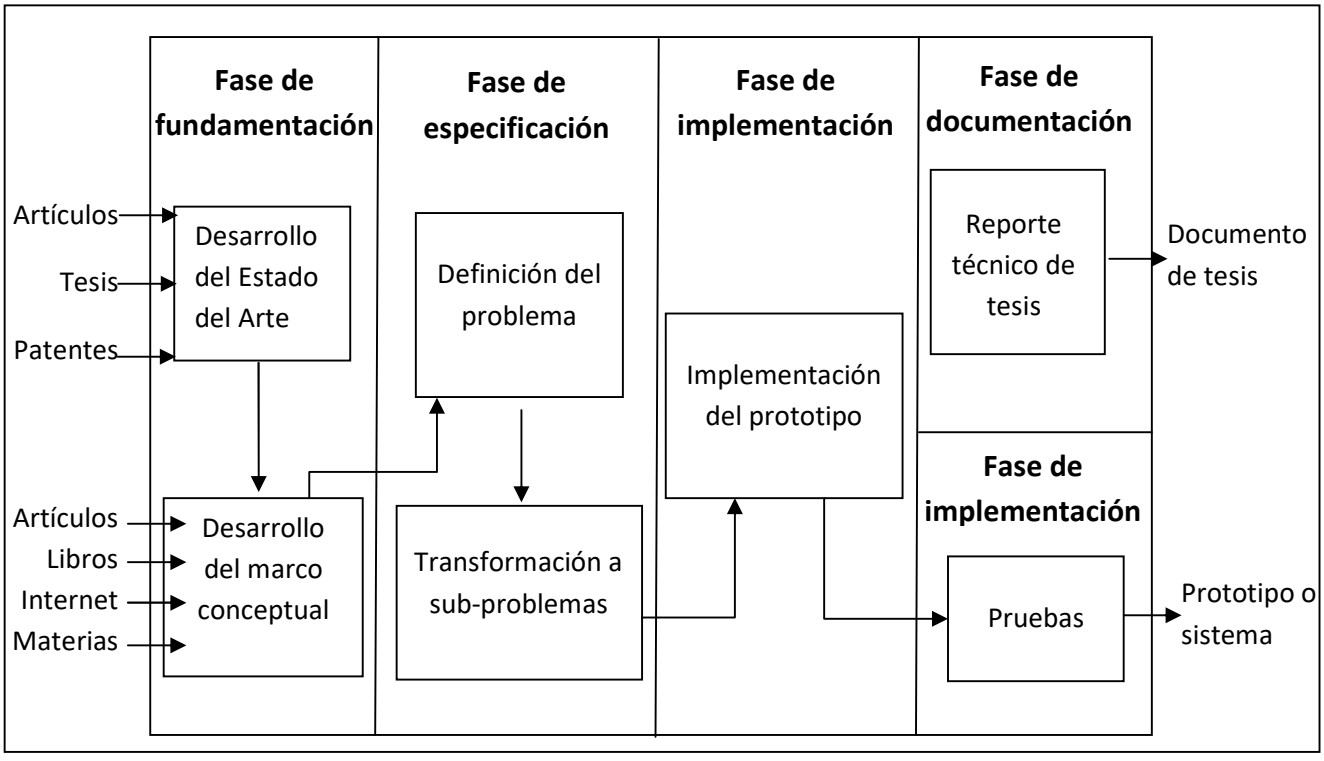


Figura 3.3 Metodología de solución.

2.8. Discusión

La odometría visual promete mejores resultados que la odometría mecánica, ya que la odometría visual reduce el error de localización cuando se genera una trayectoria esto se debe a que la odometría mecánica principalmente utiliza la información de las vueltas de las ruedas pero esto genera error en terrenos irregulares, con base en los métodos de odometría visual que se analizaron se opto por el método de Correspondencias 3D ya que este modelo permite utilizar una sensor RGB-D.

Capítulo 4 Análisis, diseño e implementación del sistema

En este capítulo se presenta el análisis del sistema así como las tareas que se realizaron en cada fase. Posteriormente, dentro del diseño del sistema se describen las técnicas y algoritmos que se implementaron.

4.1. Método de odometría visual implementado

Después de analizar algunos de los métodos de odometría visual se tomó la decisión de implementar el algoritmo de correspondencias 3D a 3D.

El Algoritmo 1 muestra el pseudocódigo para establecer la pose de cámara [Arun 1987], tomado en cuenta que la información de puntos destacados de x, y, z es conocida al utilizar una cámara RGB-D, este proceso suele ser mejor ya que este tipo de cámaras proporcionan información en 3D.

Algoritmo 1 Obtención de traslación de odometría visual.

1 Captura imagen I_{i-1}
2 Establecer nuevo *KeyFrame* $I_{k-1} \leftarrow I_{i-1}$
3 Extraer características $IC_{k-1} \leftarrow I_{k-1}$
4 Calcular centroides 3D $\bar{X}_{k-1} \leftarrow IC_{k-1}$
5 **for** Captura imagen $\leftarrow I_i$
6 Extraer características $IC_i \leftarrow I_i$
7 Obtener correspondencias de IC_{k-1}, IC_i
8 **Si** (IC_i != IC_{k-1}) en un 75%
9 Establecer nuevo *KeyFrame* $I_k \leftarrow I_i$
10 Calcular centroides 3D $\bar{X}_k \leftarrow IC_i$
11 Obtener Traslación de pose $t_k = \bar{X}_k - \bar{X}_{k-1}$
12 Obtener Rotación de pose $USV^T = svd((X_{k-1} - \bar{X}_{k-1})(X_k - \bar{X}_k)^T)$
13 **Fin Si**
14 **Fin for**

donde, I_{k-1}, I_k son dos *Key-Frames*, se extraen puntos destacados para cada *Key-Frame* cada uno de estos conjuntos de puntos están contenidos en IC_{k-1}, IC_k, la cantidad de puntos en las imágenes es el mismo pero no necesariamente estos tendrán coincidencia en ambas imágenes. $\bar{X}_{k-1}$, $\bar{X}_k$ son los centroides de los puntos 3D de cada imagen, IC_{k-1}, IC_k son los puntos destacados de cada imagen.

El Algoritmo 1 se divide en dos partes, la primera obtiene la traslación de la cámara, y la segunda parte se encarga de calcular la rotación de la cámara. Para el cálculo de la traslación se toman dos conjuntos de puntos destacados 3D; para obtener el centroide de cada conjunto se restan los centroides obtenidos lo cual dará como resultado el desplazamiento que el cuadro actual atenido con respecto al cuadro anterior línea 11 del Algoritmo 1.

Por otro lado para calcular la rotación entre 2 imágenes, es necesario calcular primero los centroides utilizados en la traslación, después estos valores son restados a cada uno de los datos del conjunto de puntos, estas diferencias se multiplican para obtener una matriz de 3x3, finalmente se utiliza SVD [Arun 1987] para obtener la rotación de la cámara para ello se hace uso del modulo de OpenCV disponible [OpenCV 2016].

4.2. Sistema de trayectorias 3D

El sistema que se implementó para mapeo de trayectorias 3D se describe en cada una de las etapas de la Figura 4.1.

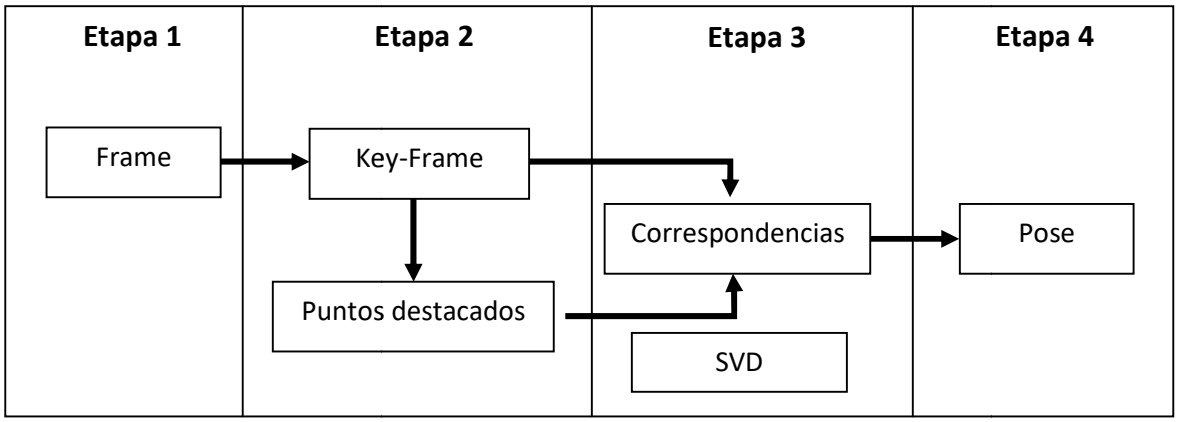


Figura 4.1 Etapas del sistema de trayectorias 3D.

Frame: Obtiene las imágenes de la cámara RGB-D tanto de color como de profundidad.

Key-Frame: Selecciona los *Key-Frame*s de acuerdo a la cantidad de puntos existentes en las imágenes, un nuevo *Key-Frame*s se establece cuando el 25% de los puntos del *Key-Frame* al cuadro actual han desaparecido.

Puntos destacados: Utilizando el módulo *Features_extration* de OpenCV [OpenCV 2016] se obtienen los puntos destacados de cada *Frame* con sus respectivas descripciones, para evitar que los puntos no se concentren en alguna región de la imagen, estos son dispersados dividiendo la imagen en cuatro regiones, así cada región contiene el 25% de los puntos totales de la imagen.

Correspondencias: Utilizando el método *Flann* de OpenCV se toman dos *Key-Frames* consecutivos para genera correspondencias, usando las descripciones de los puntos destacados.

Pose: De las correspondencias obtenidas de dos *Key-Frames* consecutivos se calcula el centroide de cada *Key-Frame*, aplicando una resta entre los centroides se obtiene la traslación de la cámara, la rotación es calculada utilizando la clase *SVD* de OpenCV (línea 12 del Algoritmo 1).

Mapa: Se genera un mapa utilizando la información de los *Key-Frame*, este proceso se realiza fuera de línea una vez que el mapeo de trayectoria ha concluido, esto debido al costo computacional.

4.3. Proceso para creación de trayectoria 3D

En esta sección se describen cada una de los puntos que fueron necesarios para obtener la pose de la cámara, utilizando el método de correspondencias 3D.

Para construir las trayectorias se utilizó PCL y un método implementado de Correspondencias 3D a 3D los cuales fueron comparados con la ruta real (*Ground truth*).

Uno de los primeros experimentos fue generar una trayectoria en 3D desplazando únicamente la cámara en el eje x. El proceso completo de la implementación se muestra de la Figura 4.2 hasta la Figura 4.7.

Al iniciar el programa se ignoran las imágenes de los primeros dos segundos esto para evitar tomar un *Frame* que pudiera estar desenfocado o alguna variación de la iluminación, en el segundo tres se establece el primer *Key-Frame*, inmediatamente se comienzan a extraer puntos de las imágenes, esto para el proceso de seguimiento de puntos, cuando el *Frame* actual está por debajo del 75% de coincidencias en puntos destacados con el *Key-Frame* anterior, este cuadro es establecido como *Key-Frame*.

Cuando se tienen dos *Key-Frames* consecutivos Figura 4.2, se comienza el proceso de obtención de pose. Cabe mencionar que los puntos destacados de cada *Key-Frame* son guardados y estos pueden ser utilizados en cálculos futuros.

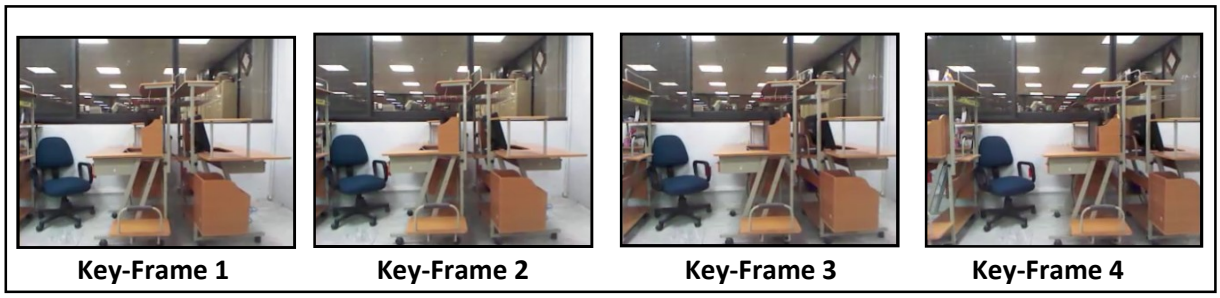


Figura 4.2 *Key-Frames* seleccionados.

Conforme las imágenes son capturadas por el sensor se aplica el método ORB con el cual se extraen puntos destacados de las imágenes, ver Figura 4.3. Estos puntos son comparados con los puntos del último *Key-Frame*, y de acuerdo al porcentaje de correspondencias entre ambos se establece un nuevo *Key-Frame*.

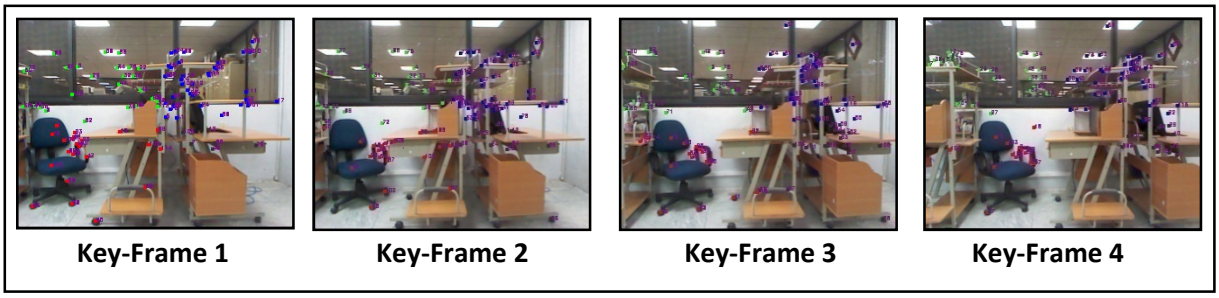


Figura 4.3 Extracción de puntos destacados.

Una secuencia de *Key-Frames* es generada, la cual será usada para definir una nueva pose de la cámara. Al tener establecidos dos *Key-Frames* consecutivos se extraen correspondencias de puntos destacados Figura 4.4.

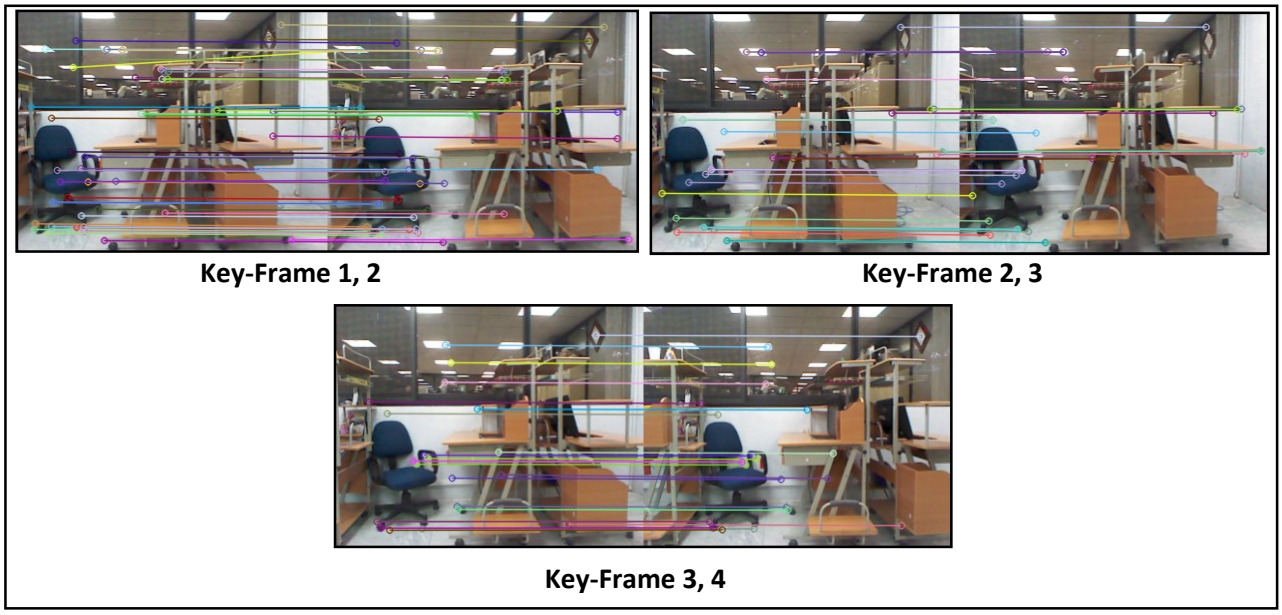


Figura 4.4 Correspondencias de puntos destacados.

Utilizando los puntos destacados de los *Key-Frames* consecutivos se calculan los centroides paso 4 del Algoritmo 1, en la Figura 4.5 muestra tres marcos de referencia, los dos ubicados en la parte superior son las *Key-Frames* consecutivos, si se alinean en un mismo punto los orígenes de las dos imágenes se obtiene algo como se muestra en la parte inferior de la Figura 4.5.

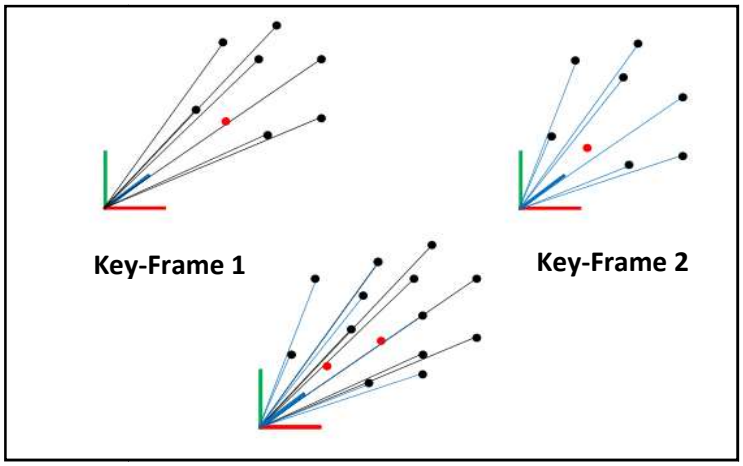


Figura 4.5 Centroides de *Key-Frames.*

Entre los conjuntos se genera una distancia en 3D de los centroides, parte superior izquierda de la Figura 4.6, si los centroides se alinean se obtiene un desplazamiento entre los orígenes de los *Key–Frames* parte superior derecha Figura 4.6.

La alineación entre los puntos destacados de los *Key-Frames* no es perfecta ya que aunque las descripciones de los puntos sean exactas estos cambian su posición conforme la cámara avanza en un entorno.

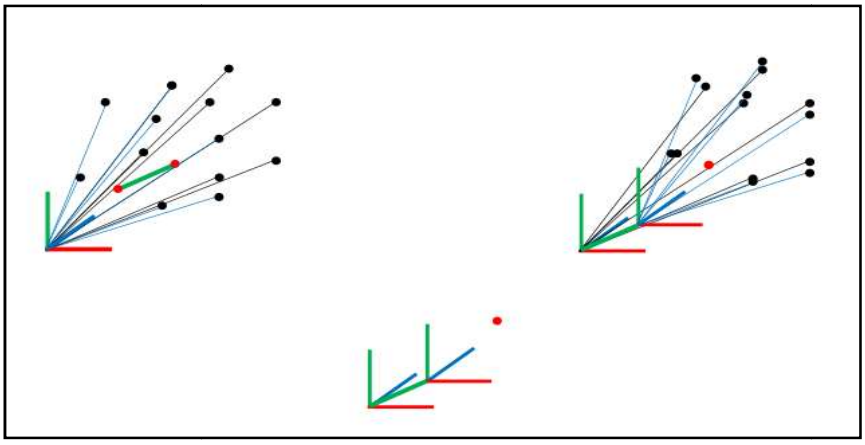


Figura 4.6 Alineación de centroides.

Los puntos destacados están descritos en el espacio 3D utilizando la posición de alto, ancho en unidades de pixeles mientras que la profundidad está dada en unidades de milímetros.

Para llevar esta traslación a un espacio 3D es necesario convertir las unidades de pixeles a milímetros, esto se logra utilizando los parámetros obtenidos de la calibración, ya que

estos parámetros nos ayudan a identificar que tan alto y ancho es un pixel en unidades de milímetros, así también estos se ven afectados por un factor radial al aplicar lo anterior se llevan las coordenadas de pixeles a un espacio en milímetros.

Así cada que se calcula la pose de la cámara se obtienen seis valores de los cuales tres corresponden al desplazamiento en 3D y los tres restantes a la rotación Figura 4.7.

```
[0.1301, 0.0031, -0.0012][0.0021, 0.0016, 0.0028]
[0.3112, 0.0025, -0.0023][0.0028, 0.0011, 0.0015]
```

Figura 4.7 Datos de pose.

Una vez obtenidos los valores de cada pose utilizando información de dos *Key-Frames* consecutivos se agregan a un archivo, generando así las coordenadas de traslación y rotación que se han calculada para cada pose en una trayectoria.

La distancia entre Kinect y el centroide dentro de la imagen es utilizada para desplazar la pose obtenida a las coordenadas de la cámara para cada *Key-Frame* Figura 4.8. Al realizar este desplazamiento de coordenadas se obtiene el desplazamiento de la cámara, si esto no se realizara la pose estaría adelantada a la cámara.

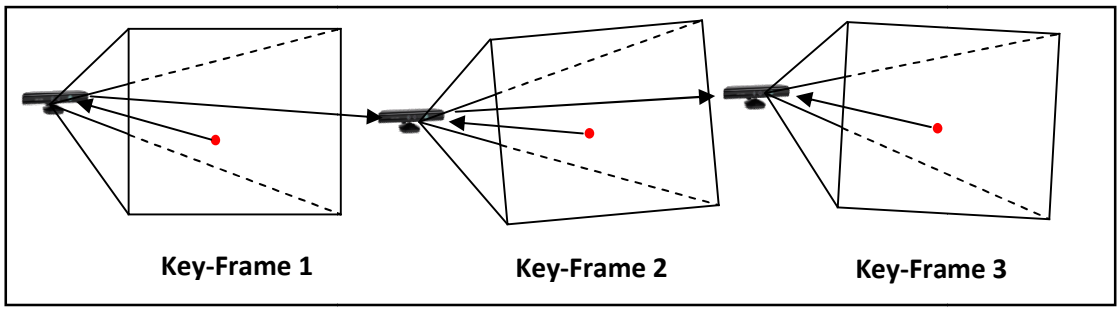


Figura 4.8 Centroides de *Key-Frames.*

4.3.1. Calibración de cámara

En este trabajo la idea de calibrar la cámara es: poder obtener parámetros que permitan interpretar la información de una imagen dada en unidades de pixeles a medición del mundo real (ejemplo centímetros). Existen algunos programas que han sido desarrollados para obtener los parámetros de calibración de la cámara Kinect. En este trabajo se utiliza el módulo de *kinect_node* incluido en ROS [Conley 2011] para obtener los parámetros de calibración, otro paquete disponible es MRPT [mrpt 2016] el cual no es exclusivo de Kinect, pues este implementa distintas técnicas de calibración para distintas cámaras.

Para obtener la pose de la cámara en coordenadas del mundo real es necesario conocer los parámetros intrínsecos de la cámara, ya que estos contienen relaciones con los datos de la imagen de una cámara por ejemplo la distorsión radial, la distancia en milímetros de un pixel relacionado al CCD de la cámara entre otros.

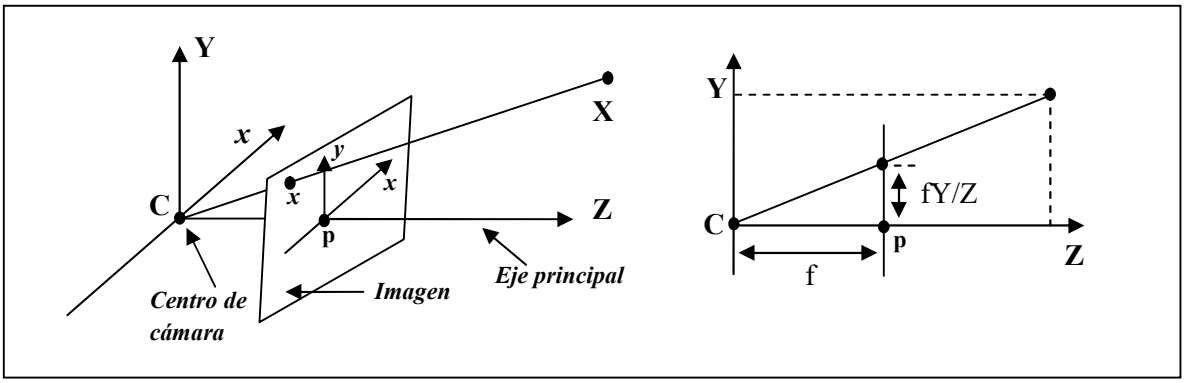


Figura 4.9 Calibración de cámara *pinhole*.

Un ejemplo común es la calibración *pinhole* Figura 4.9, donde $\boldsymbol{C}$ es el centro de la cámara y $\boldsymbol{p}$ el punto principal.

Para obtener los valores de calibración de cámara se utilizan las siguientes ecuaciones:

$$X = f\frac{x_c}{z_c} \tag{1}$$

$$Y = f\frac{y_c}{z_c} \tag{2}$$

donde (X, Y, Z) son las coordenadas absolutas o universales de cualquier punto de una imagen en tres dimensiones, según se observa en la Figura 25. Se supone que $Z < f$, esto es, todos los puntos de interés se encuentran enfrente de la lente. En primer lugar nos interesa obtener una relación que nos de las coordenadas (x, y) de la proyección del punto (X, Y, Z) en el plano de la imagen, para ello se utilizan las ecuaciones 1 y 2 [Martinsanz 2001].

Utilizando el modulo de *kinect_node* incluido en ROS los datos que se han obtenido de la calibración se muestran en la Tabla 4.

Tabla 4. Parámetros de calibración.

f_x	5.2921e+02
f_y	5.2556e+02
c_x	3.2894e+02
c_y	2.6748e+02
k_1	2.6451e-01
k_2	-8.3991e-01

donde, f_x, f_y son la distancia focal,c_x, c_y denotan el centro de la cámara y k_1, k_2 son parámetros de distorsión radial, todos los valores anteriores están dados en milímetros.

Estos parámetros serán utilizados para trasformar coordenadas de la cámara a coordenadas del mudo real como se muestra en la Figura 4.9.

4.3.2. Puntos destacados

Dentro de la odometría visual la extracción de puntos de una imagen es un paso fundamental para establecer la pose de la cámara o definir un nuevo *Key-Frame*.

OpenCV contiene métodos de extracción de puntos algunos de ellos son: SIFT, SURF, ORB, *GoodFeatures* y MSER.

De los métodos de extracciones de puntos destacados se decide utilizar ORB, debido a la descripción de cada uno de los puntos que es más representativa que de los demás métodos [Rublee 2011] esto sin duda impacta en una mejor correspondencia de puntos destacados.

Para minimizar el tiempo de procesamiento se restringe la cantidad de puntos extraídos a 120 como en [Scaramuzza 2014].

Uno de los problemas al restringir la cantidad de puntos es que estos pueden concentrarse en una región de la imagen, para ayudar a dispersar los puntos en toda la imagen, cada imagen es dividida en 4 regiones asegurando que esa región cuente con el 25% de los puntos, dispersando así los puntos en la imagen, Figura 4.10.

Figura 4.10 Dispersión de puntos.

4.3.3. Descripción de puntos destacados

Después de evaluar los distintos métodos de extracción de características se tomo la decisión de utilizar el método de ORB, esto debido a distintas ventajas que presenta contra los demás métodos. ORB obtiene 31 descripciones de cada punto [Rublee 2011], con las cuales se realizan las correspondencias de los puntos destacados.

4.3.4. Establecer nuevo *Key-Frame*

Existen distintas técnicas para establecer un *Key-Frame*, las más comunes son: tomar en cuenta un porcentaje de puntos que no aparecen en las dos imágenes, cuando el porcentaje de la escena actual es menor que el porcentaje establecido, se define la imagen actual como *Key-Frame*, otra forma es establecer *Key-Frame*s cada n tiempo. También se puede considerar la distancia, esto sería cada ciertos metros o centímetros, tomar un *Key-Frame*, se debe tomar en cuenta la velocidad de desplazamiento ya que si la velocidad del robot o cámara es rápida no se podrán encontrar correspondencias entre las imágenes pues podrían no ser consecutivas de la escena.

En este sistema se utilizó la técnica de porcentaje de puntos. Donde la consulta de puntos se realiza siempre y cada vez que se desea establecer un *Key-Frame*se toma esta consulta para analizar cuantos puntos corresponden en la imagen actual con el *Key-Frame* anterior, por ejemplo si el 25% de los puntos no aparecen en el cuadro actual este se establece como *Key-Frame*. Esto permitirá que independientemente de la velocidad de la cámara se garantice que hay información de correspondencias entre la imagen actual y el *Key-Frame* anterior.

4.3.5. Correspondencia de puntos destacados

De los métodos disponibles en OpenCV de correspondencias de puntos destacados, se implementaron: *FlannMatcher* y *Brute-Force*. En estas pruebas no se considerará una restricción de puntos para que el método *Brute-Force* encuentre correspondencias a la cantidad de puntos totales.

4.4. Discusión

De los distintos métodos que se analizaron se opto por el método de Correspondencias 3D debido a sus ventajas por ejemplo que permite utilizar un sensor RGB-D. Una vez seleccionado el método a implementar se abordaron los distintos puntos que componen el sistema de Odometría Visual de esta tesis.

También se explico la importancia de dividir la imagen en 4 secciones iguales para dispersar los puntos destacados en toda la imagen, esto permite que los puntos destacados no se concentren en una zona de la escena, con el fin de garantizar que ante desplazamientos bruscos no se pierda la totalidad de los puntos destacados.

En el siguiente capítulo se explican mas a detalla los experimentos resaltando los puntos fuertes de cada uno de los métodos implementados.

Capítulo 5 Experimentos y Resultados

En este capítulo se presentan las principales pruebas realizadas al sistema. Para la validación se realizaron pruebas independientes de cada módulo del sistema, y al final se probó el sistema completo. Una vez que el software de desarrollo se instaló y configuró correctamente, se desarrolló un sistema de odometría visual, el cual es explicado en detalle.

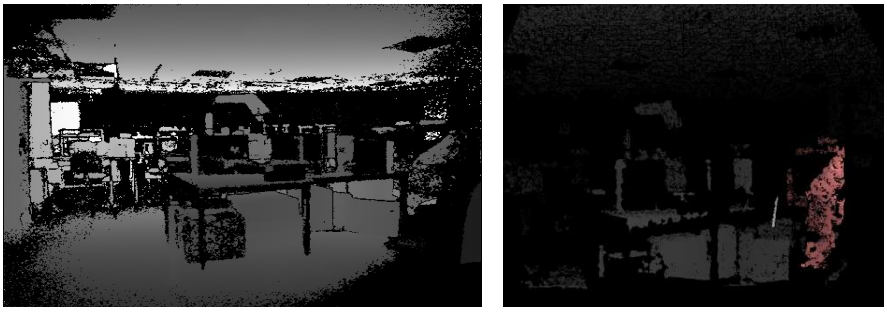
5.1. Pruebas del sensor

- Mostrar la información captura por Kinect en una venta de la PC.

Se evaluaron los sensores disponibles Kinect Xbox360 y Kinect One, la Figura 5.1 se muestran las imágenes de color de los distintos dispositivos, las dimensiones disponibles de la imagen RGB para Kinect Xbox360 es 640x480, mientras que para Kinect One 1920x1080 [Smisek 2011].

a) Kinect One. **b) Kinect Xbox360.**

Figura 5.1 Imágenes RGB de los distintos dispositivos.



a) Kinect One. **b) Kinect Xbox360.**

Figura 5.2 Imágenes de profundidad de los distintos dispositivos.

Ambas cámaras proporcionan información de profundidad Figura 5.2, en las pruebas realizadas entre las distintas librerías el Kinect de Xbox 360 no tuvo problemas con la comunicación con las distintas librerías de las herramientas auxiliares, mientras que el Kinect One presenta algunas incompatibilidades, por esos motivos se opto por utilizar la cámara de sensor Kinect 360.

- Guardar imágenes RGB tanto del Kinect Xbox360 y Kinect One en la PC.

Existen algunas aplicaciones que permiten guardan imágenes RGB-D tales como *PCL*, *Autocad*, *3DReshaper*, *Blender*, *3D Studio Ma*, *OPENGL*, *PDAL* y *MATLAB*, en este trabajo se opto por utilizar la librería de *PCL* que es software libre, debido a que muchas de estas

aplicaciones no se pueden fusionar con un sistema embebido para realizar un sistema autómata o necesitan de costosas licencias de aplicación.

5.2. Pruebas de extracción de puntos destacados

Las técnicas de puntos destacados con las cuales se experimentó están disponibles como software libre en la herramienta OpenCV, en la cual se pueden encontrar: *GoodFeatures*, MSER, ORB, SIFT y SURF entre otros.

Para realizar la extracción de características las primeras pruebas se realizaron con dos imágenes en Figura 5.3 la Tabla 5 muestra la cantidad total de puntos extraídos en cada imagen, utilizando distintos métodos de extracción de puntos.

Detector	Imagen 1	Imagen 2
GoodFeatures	459	473
MSER	218	217
ORB	380	382
SIFT	1522	1556
SURF	1083	1097

Tabla 5. Extracción de características.

Para ejemplificar la extracción de puntos destacados la Figura 5.3 muestra dos imágenes consecutivas a las que se les aplico el método ORB los puntos destacados obtenidos se agregan a la imagen como pequeños círculos de color. Los distintos métodos de extracción de características que se utilizaron para los experimentos se encentran en el Anexo A.1.

Figura 5.3 Puntos destacados utilizando ORB.

5.3. Pruebas de Correspondencia de puntos

Utilizando la información de los puntos destacados de la Tabla 5 se realizó la correspondencia de puntos entre dos imágenes consecutivas Figura 5.4, para ello se usaron distintos métodos tabla 6. Las imágenes con las cuales se realizaron los experimentos de correspondencias están contenidas en el Anexo A.2.

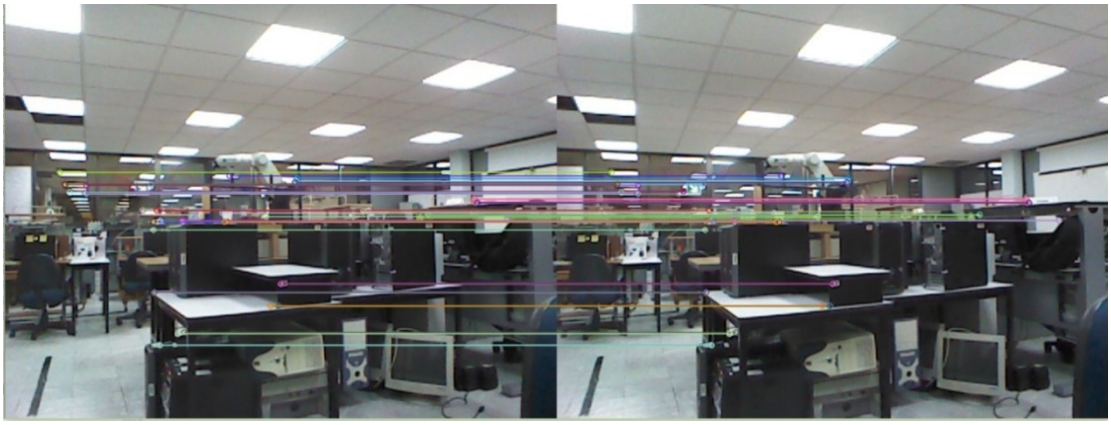


Figura 5.4 Correspondencia de puntos destacados.

Los resultados obtenidos de correspondencias de puntos utilizando *GoodMatches* son escasos pero la mayoría de estas correspondencias de las imágenes son correctas, mientras que *Brute-Force* obtiene más correspondencias pero notoriamente existen más malas correspondencias.

Tabla 6. Correspondencia de puntos.

Descriptor	Goodmatches	Brute-Force
GoodFeatures	24	421
MSER	6	218
ORB	49	245
SIFT	9	1426
SURF	26	153

5.4. Pruebas de selección de *Key-Frames*

En este sistema se utilizó la técnica de porcentaje de puntos, La consulta de puntos se hace siempre, y cuando la prueba es positiva, se establece *Key-Frame* para esto se consulta cuantos puntos corresponden en la imagen actual con el *Key-Frame* anterior, por ejemplo si el 50% de los puntos no aparecen en el cuadro actual, este se establece como

Key-Frame; esto permitirá que independientemente de la velocidad de la cámara se garantice que hay información de correspondencias entre la imagen actual y el *Key-Frame* anterior.

Se realizaron pruebas de selección de *Key-Frame*s con distintos porcentajes de puntos destacados. La Tabla 7 muestra las correspondencias correctas e incorrectas que se obtuvieron para cada porcentaje.

Tabla 7. Porcentaje de correspondencias.

%	**90**	**85**	**80**	**75**	**70**	**65**	**60**	**55**	**50**	**45**	**40**	**35**	**30**	**25**	**20**
Si	80	82	81	87	85	89	92	91	93	92	95	98	90	89	86
no	20	18	19	13	15	11	8	9	7	8	5	2	10	11	14

El sistema extrae 120 puntos destacados a cada imagen, de estos puntos existe un porcentaje mínimo que son descartados automáticamente debido a que no se cuenta con la información de profundidad de dichos puntos. Los resultados obtenidos de los experimentos mostrados en la tabla 7 toman como el 100% de los restantes descartando los que no tienen profundidad, ejemplo si de los 120 puntos no se obtiene información de profundidad de 5 puntos, los 115 puntos restantes serán el 100%.

Cuando la cámara comienza a capturar imágenes, una de las primeras imágenes se establece como *Key-Frame 1.* Se realizaron varias pruebas para definir que cuadro es conveniente tomar como *Key-Frame*, llegando a la conclusión de que las primeras 60 imágenes son inestables, los dos primeros segundos aproximadamente ya que presentan problemas de cambios de intensidad obteniendo imágenes que varían tendiendo a hacer oscuras o claras.

5.5. Pruebas de generación de trayectorias

Se generaron distintas trayectorias 3D conforme se realizaron los experimentos e incrementó la cantidad de movimientos y distancias de las trayectorias.

El procesamiento se realiza en tiempo real; obteniendo la posición y orientación de la cámara en un espacio 3D, las gráficas de errores tanto de traslación y rotación se generaron fuera de línea.

En la Figura 5.5 se muestran las trayectorias reales que fueron trazadas en el laboratorio de Inteligencia Artificial de Cenidet, las cuales fueron trazadas en un espacio de 7, 7, 2 metros aproximadamente con respecto a ***x, y, z***.

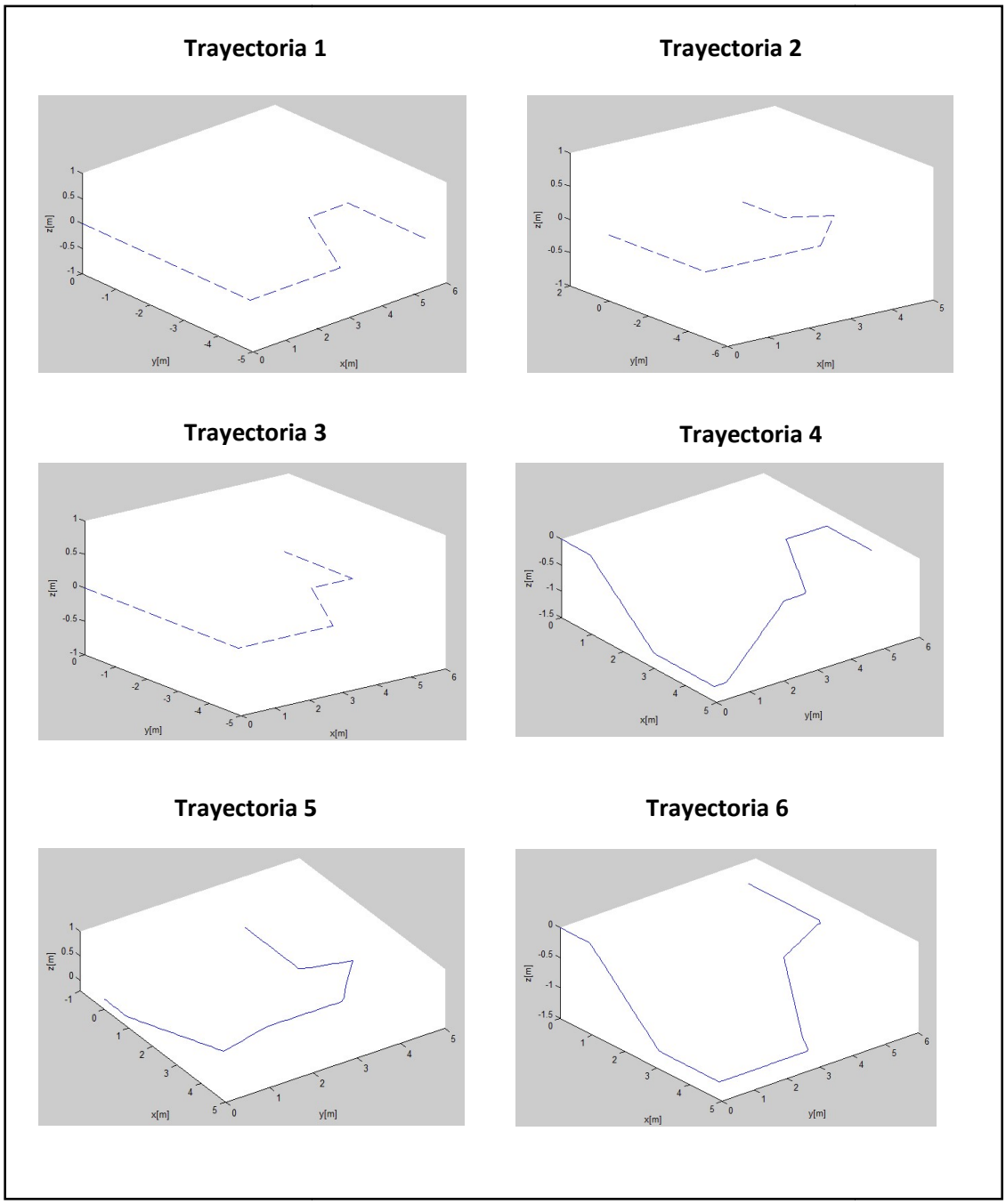


Figura 5.5 *Groundtruth* de las distintas trayectorias.

En la Figura 5.6 se muestran las trayectorias en 3D obtenidas de los métodos de PCL y correspondencias 3D, donde se generó un desplazamiento en el eje x.

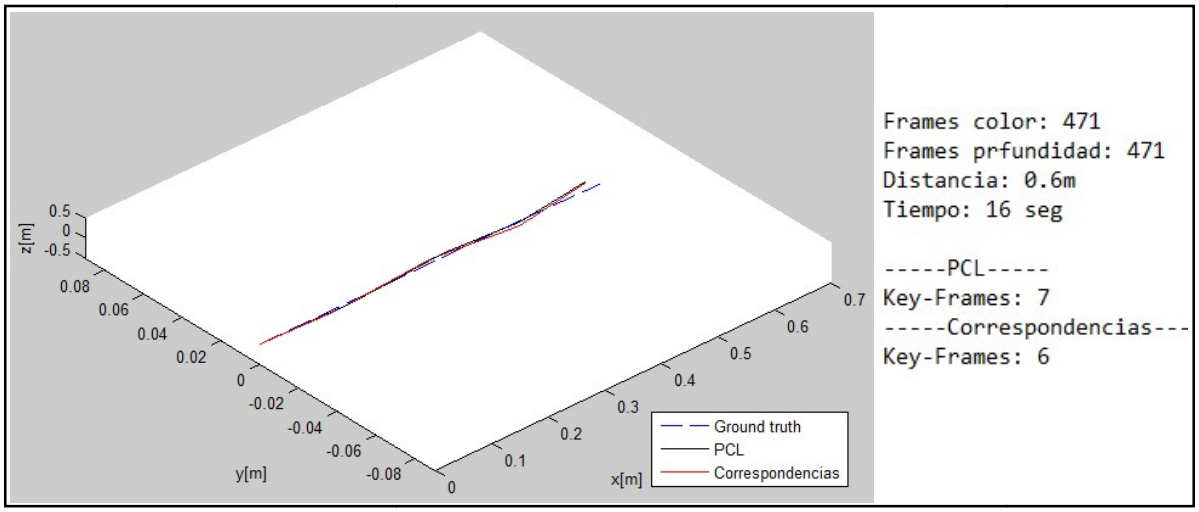


Figura 5.6 Trayectoria en un eje.

Los errores de traslación de la trayectoria desplazada por un eje son obtenidos en metros, en la Figura 5.6 y la Figura 5.7 muestra los errores de la rotación expresados en unidades de radianes.

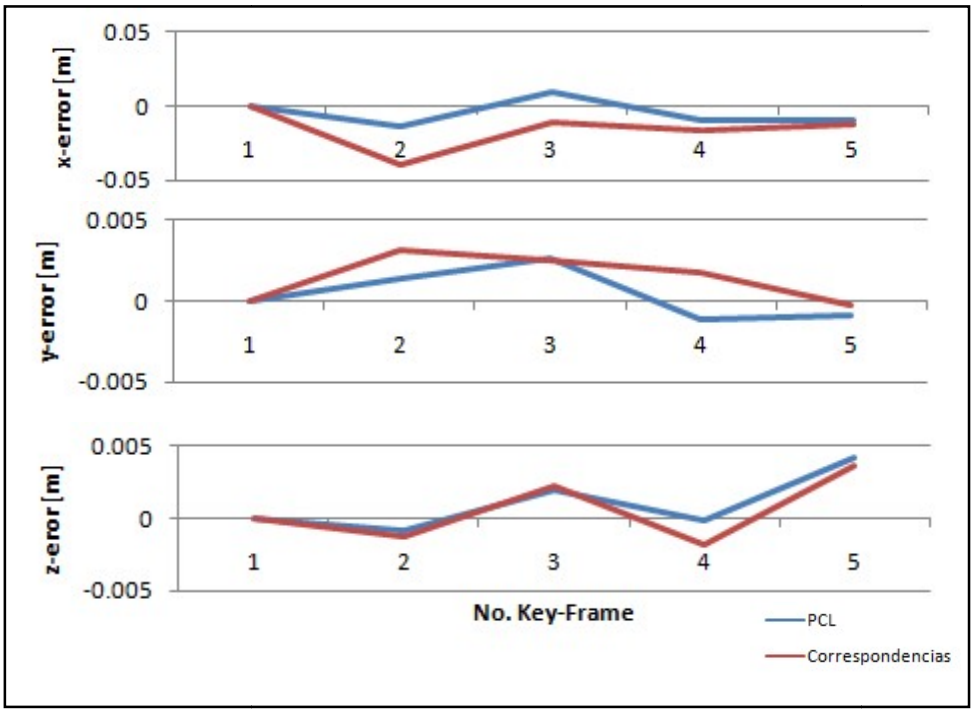


Figura 5.7 Error de traslación.

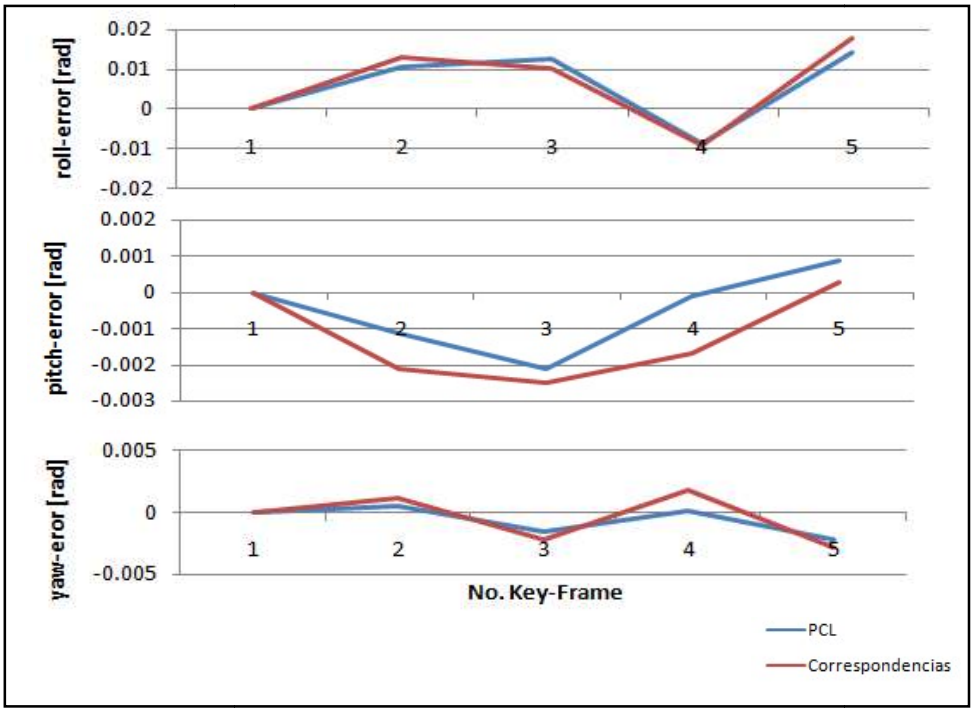


Figura 5.8 Error de rotación.

5.5.1. Trayectorias desplazadas en x, y

En la trayectoria 1 se utilizan desplazamiento hacia adelante en x, y rotaciones en Yaw *derecha-izquierda* Figura 5.9.

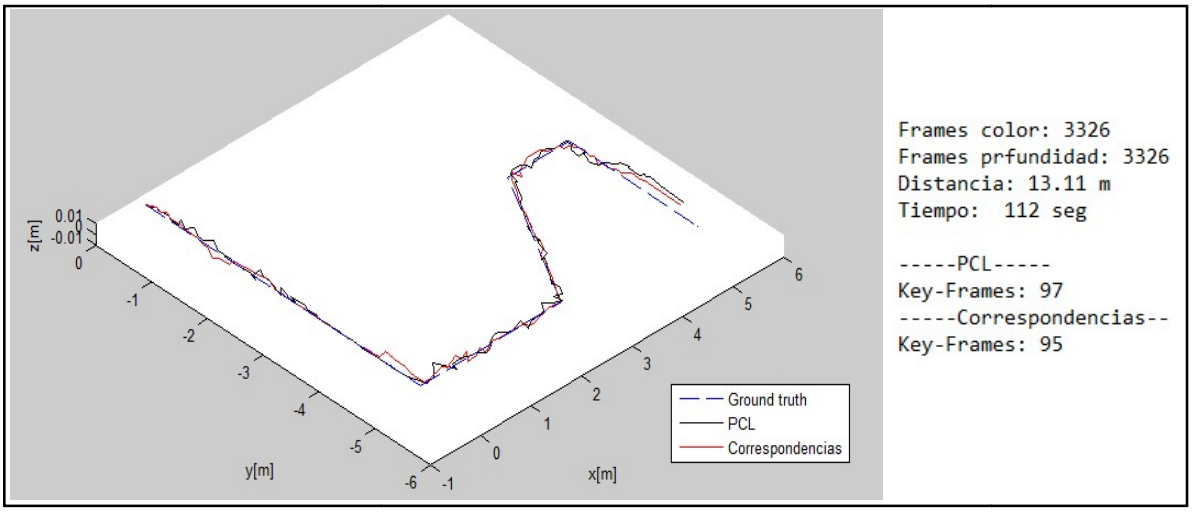


Figura 5.9 Trayectoria 1.

En la Figura 5.10 se realiza un acercamiento a las trayectorias generadas donde se aprecian con más detalle las rutas que se generan.

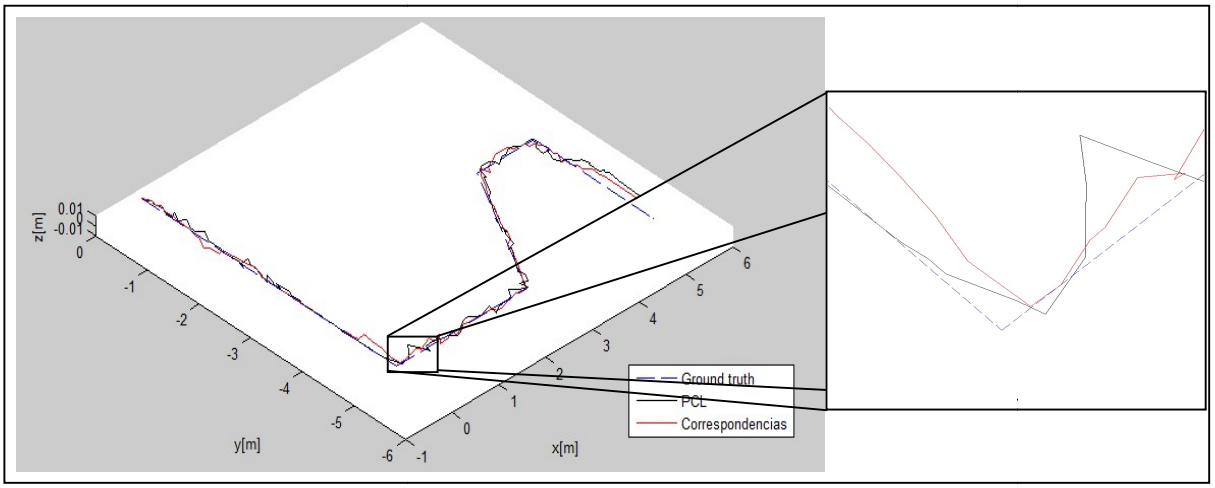


Figura 5.10 Ampliación de trayectoria.

La Figura 5.11 muestra el error de traslación y la Figura 5.12 el error de rotación que pertenecen a la trayectoria 1.

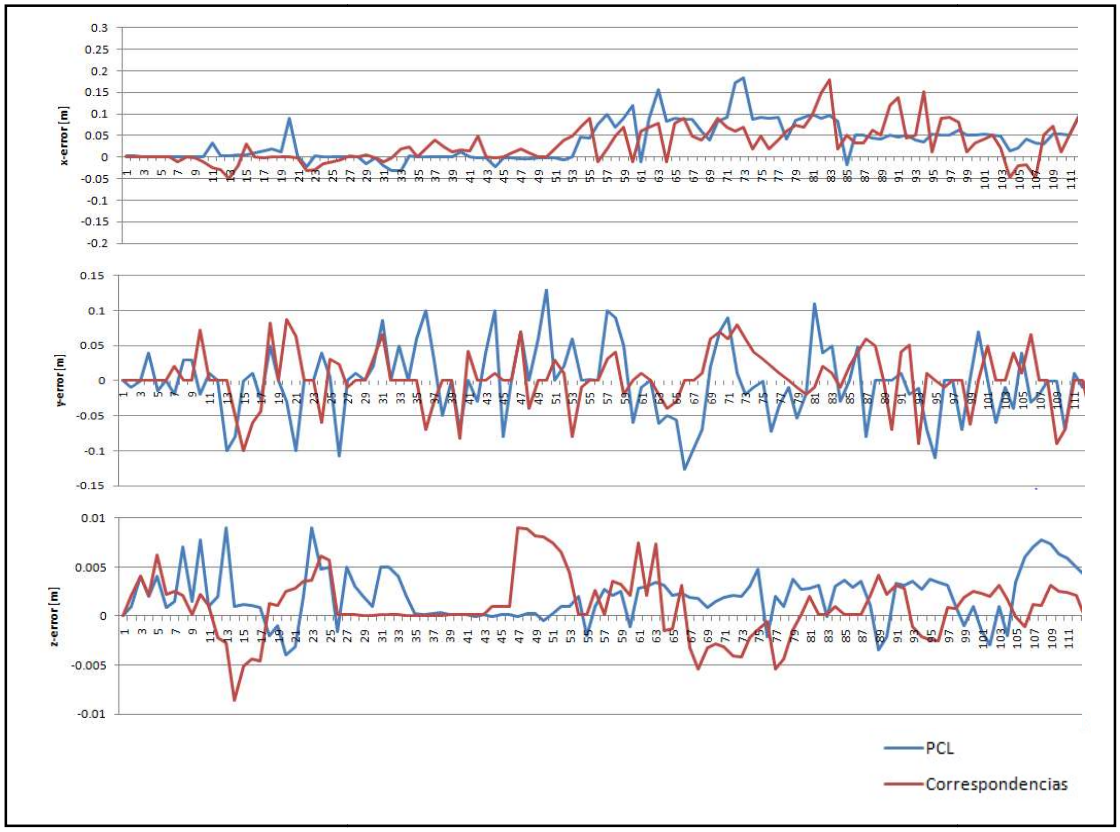


Figura 5.11 Error de traslación de trayectoria 1.

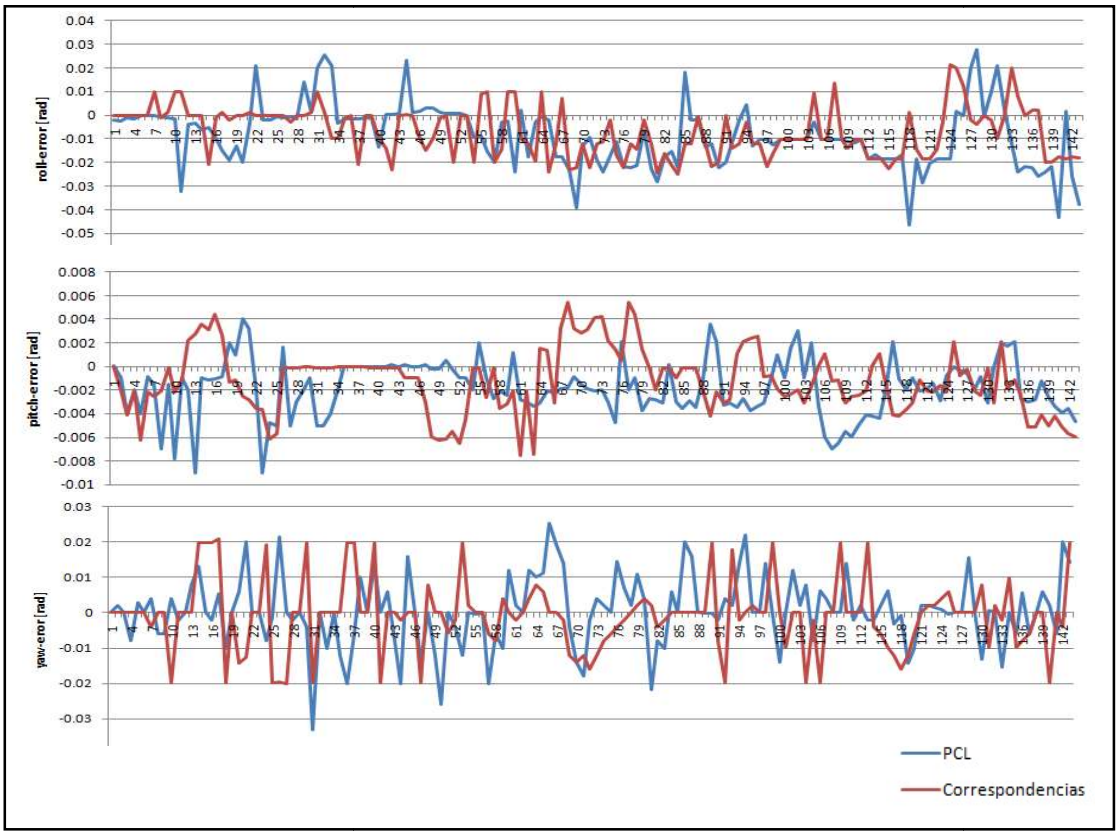


Figura 5.12 Error de rotación de trayectoria 2.

En la trayectoria 2 se utiliza desplazamiento hacia adelante en x, y rotaciones en Yaw *derecha-izquierda* Figura 5.13.

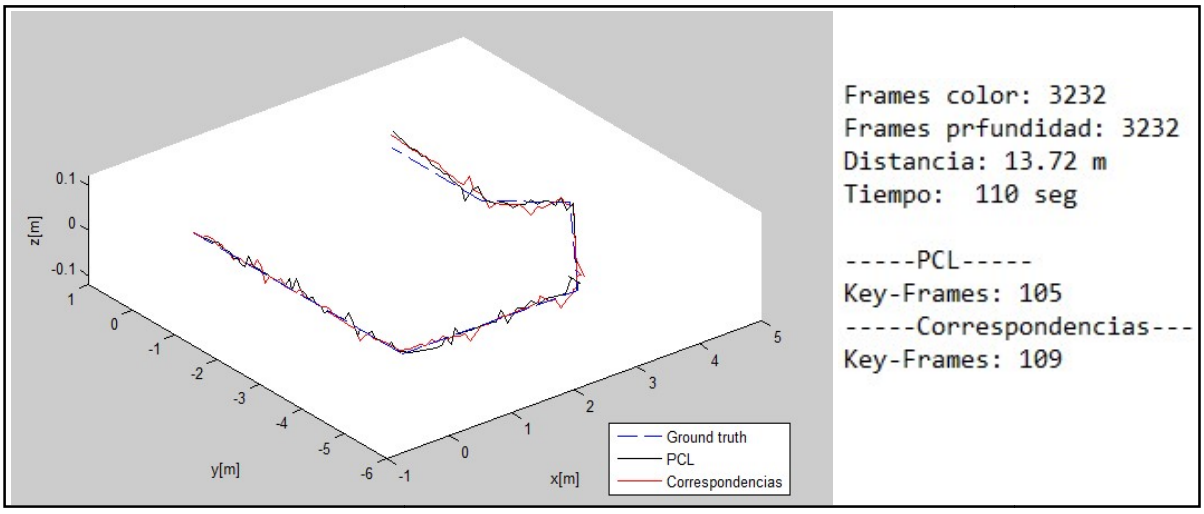


Figura 5.13 Trayectoria 2.

La Figura 5.14 muestra el error de traslación y la Figura 5.15 el error de rotación que pertenecen a la trayectoria 2.

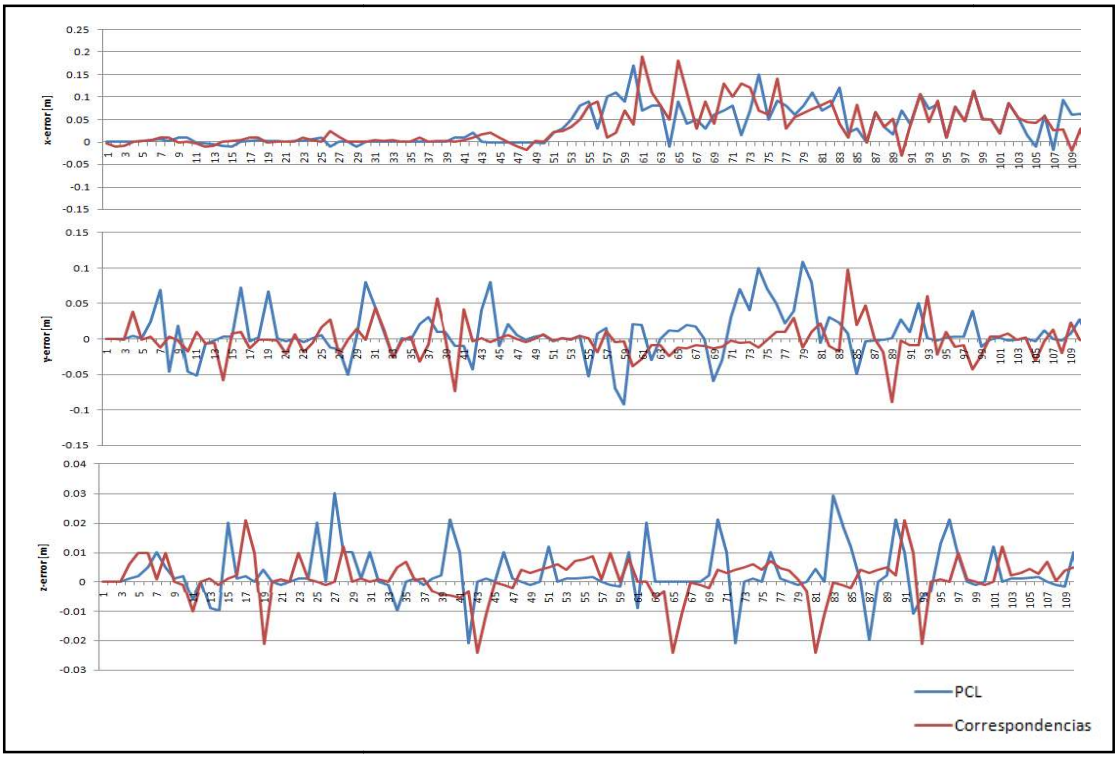


Figura 5.14 Error de traslación de trayectoria 2.

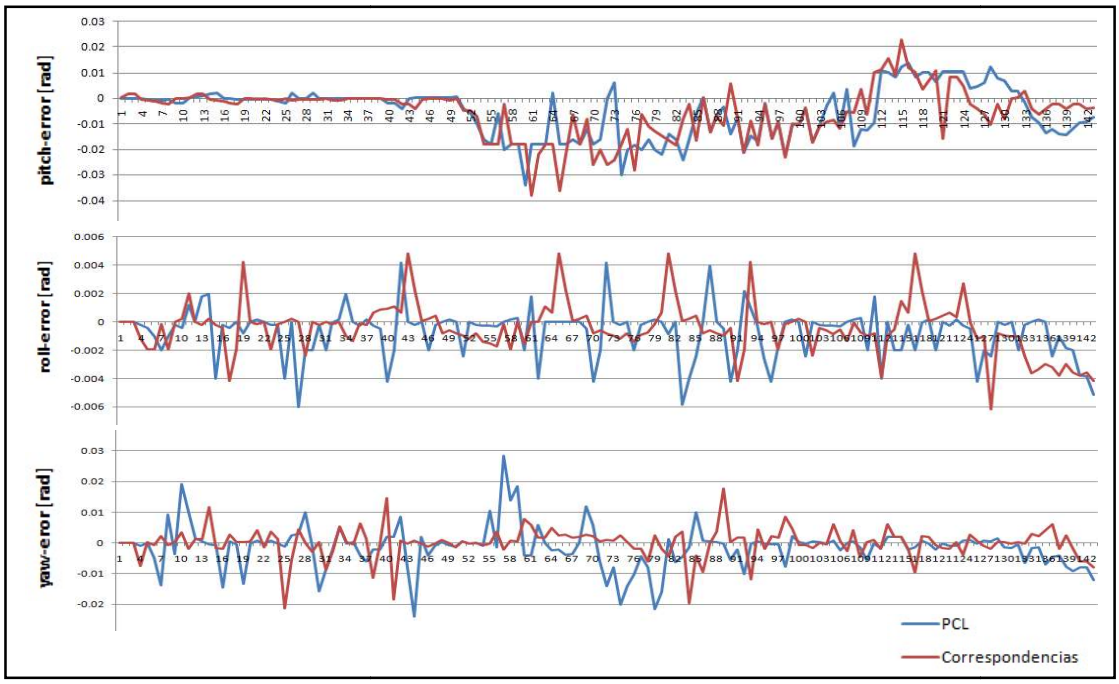


Figura 5.15 Error de rotación de trayectoria 2.

En la trayectoria 3 se utiliza desplazamiento hacia adelante en x, y rotaciones en Yaw *derecha-izquierda* Figura 5.16.

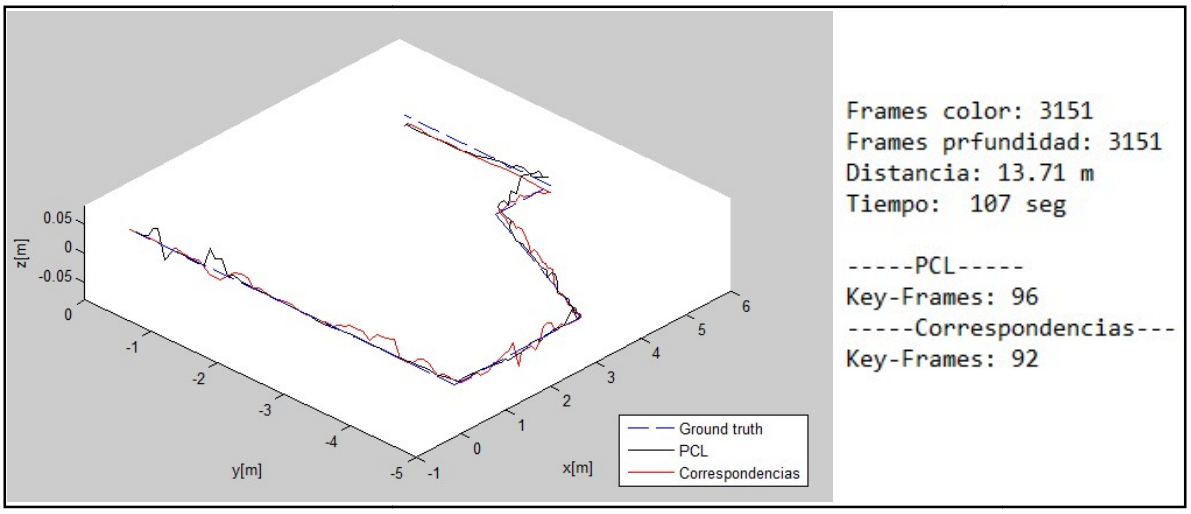


Figura 5.16 Trayectoria 3.

La Figura 5.17 muestra el error de traslación y la Figura 5.18 el error de rotación que pertenecen a la trayectoria 3.

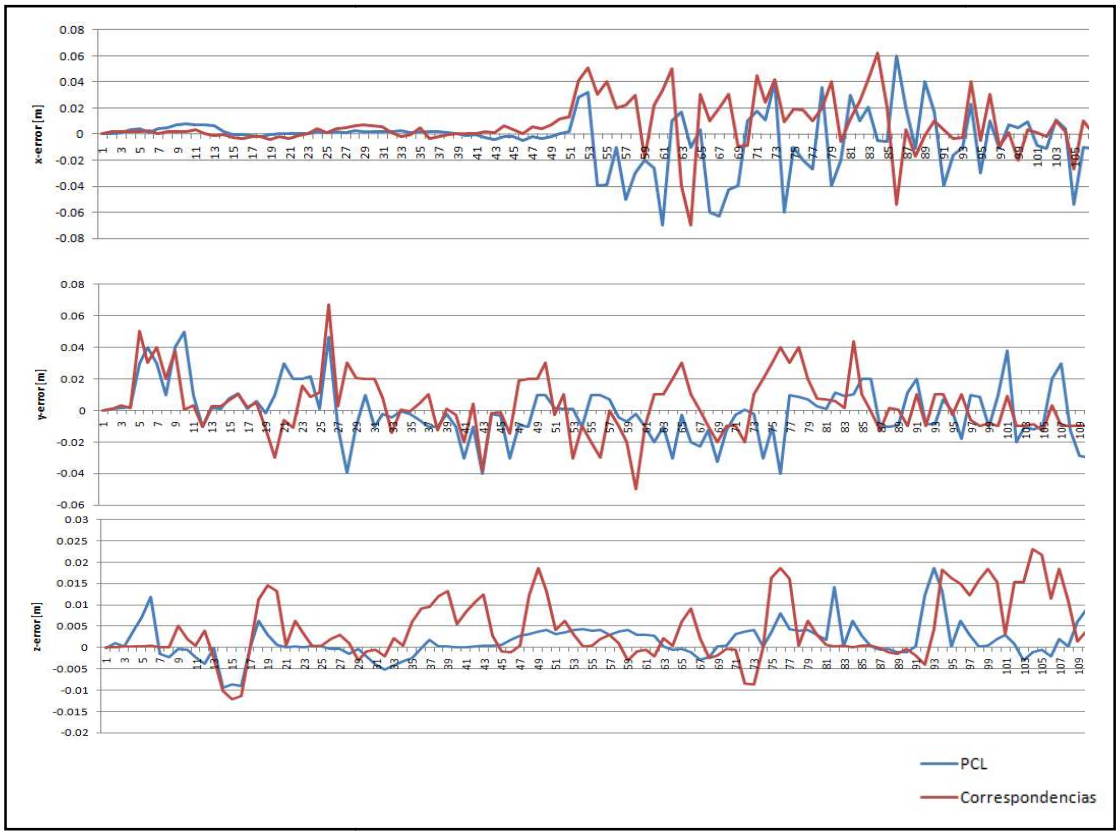


Figura 5.17 Error de traslación de trayectoria 3.

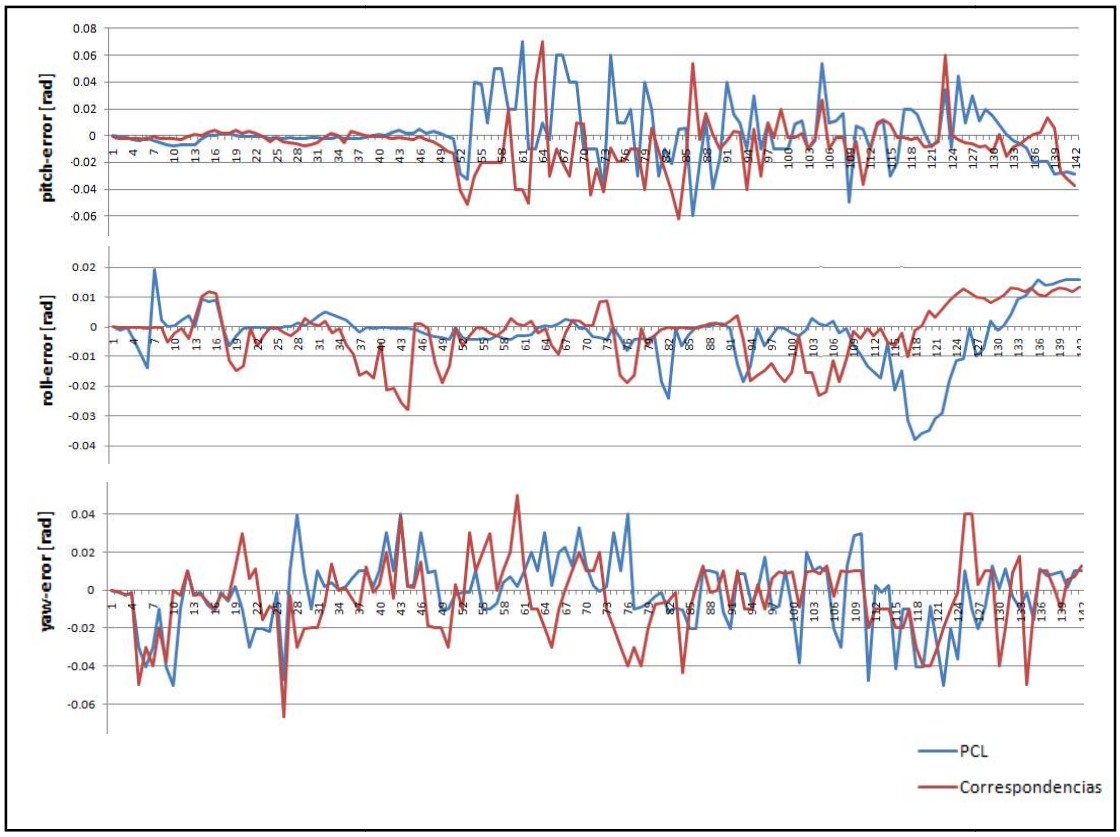


Figura 5.18 Error de rotación de trayectoria 3.

5.5.2. Trayectorias desplazadas en x, y, z

En la trayectoria 4 se utiliza desplazamiento hacia adelante en x, rotaciones en Yaw *derecha-izquierda* y rotaciones en $Pitch$ *arriba-abajo* Figura 5.19.

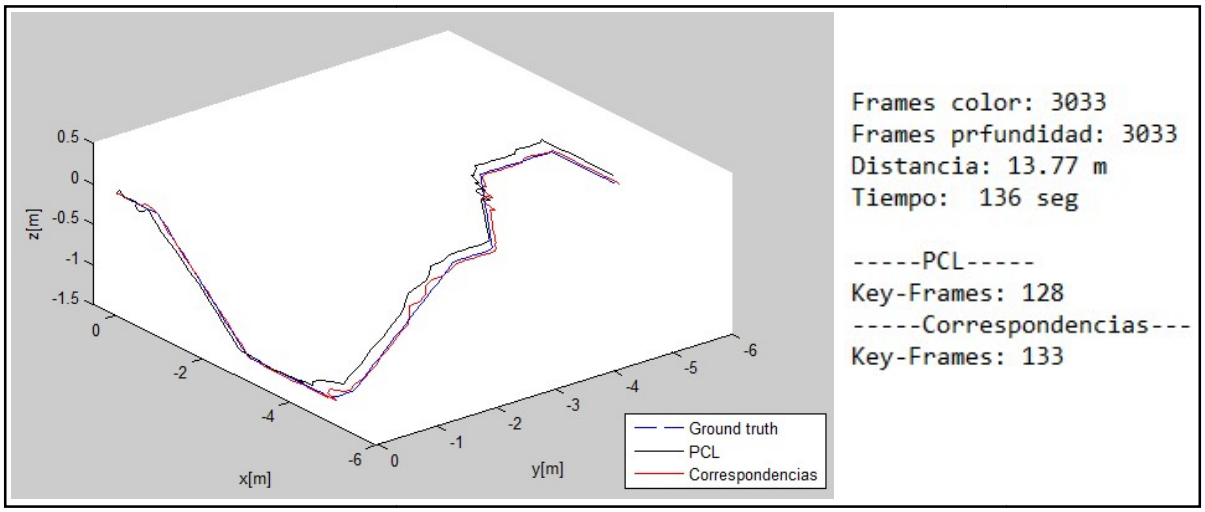


Figura 5.19 Trayectoria 4.

La Figura 5.20 muestra el error de traslación y la Figura 5.21 el error de rotación que pertenecen a la trayectoria 4.

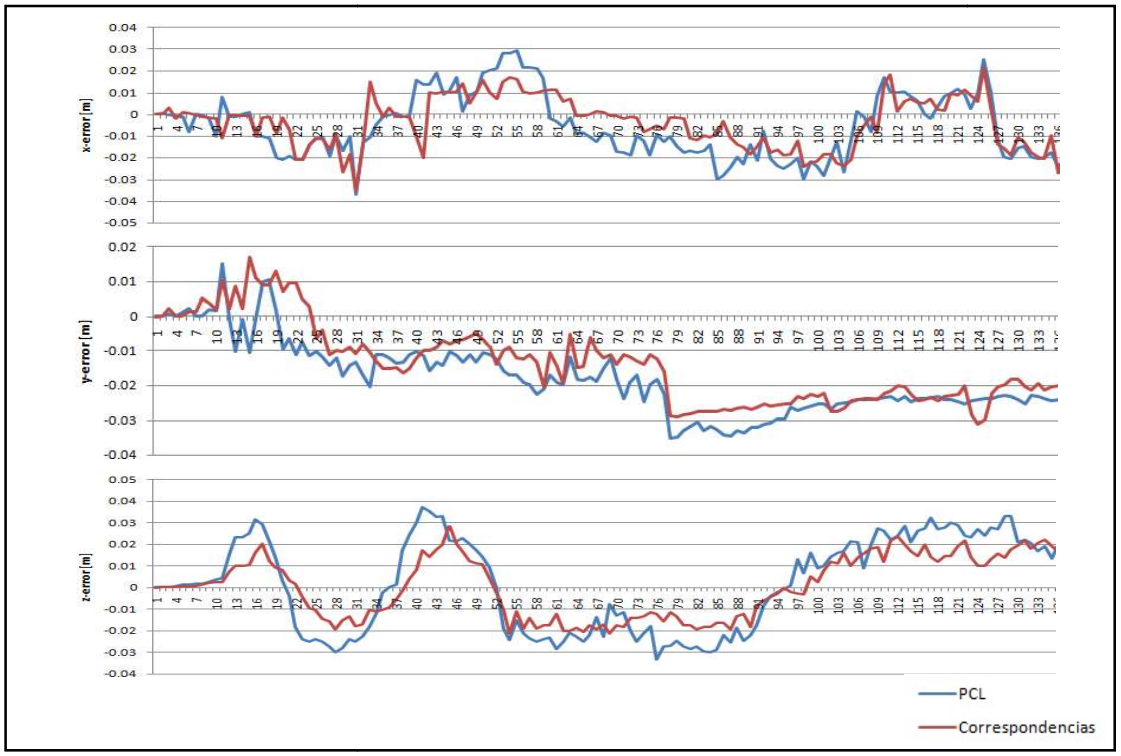


Figura 5.20 Error de traslación de trayectoria 4.

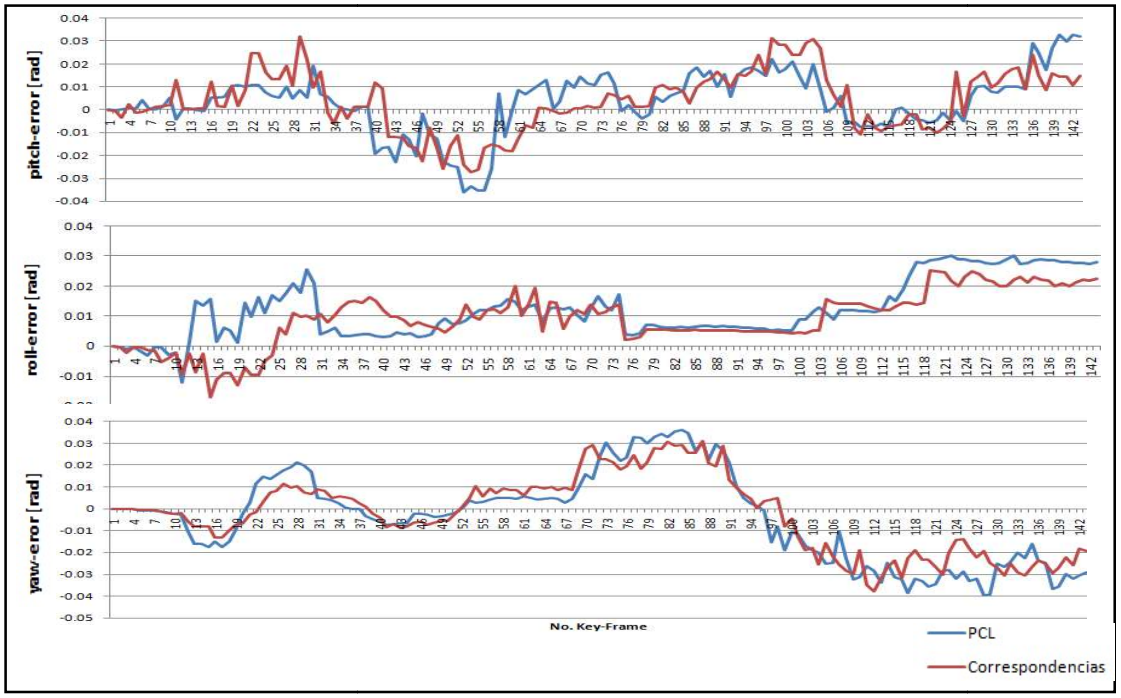


Figura 5. 21 Error de rotación de trayectoria 4.

En la trayectoria 5 se utiliza desplazamiento hacia adelante en x, rotaciones en Yaw *derecha-izquierda* y rotaciones en $Pitch$ *arriba-abajo* Figura 5.22.

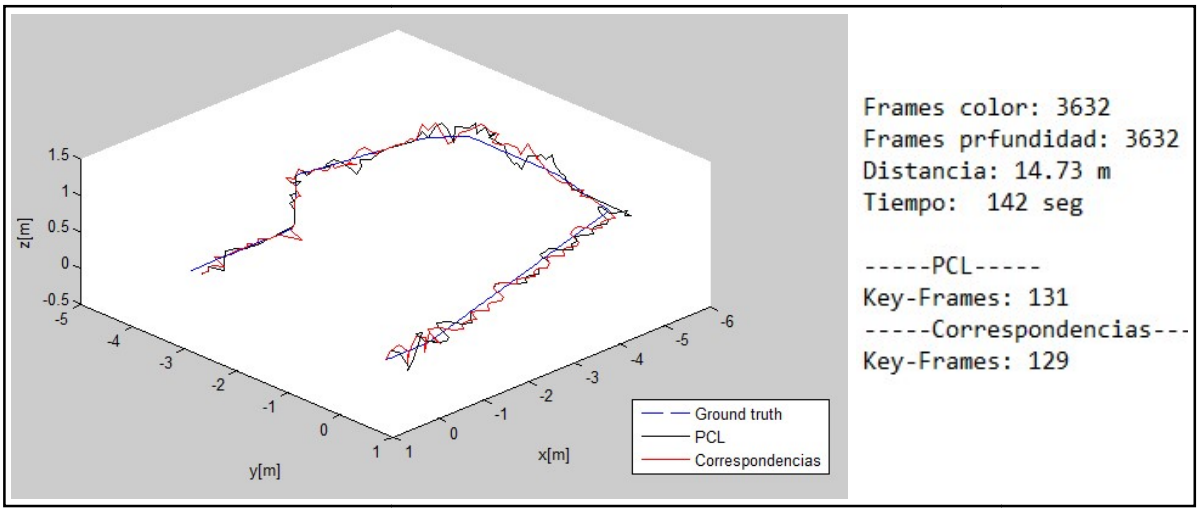


Figura 5.22 Trayectoria 5.

La Figura 5.23 muestra el error de traslación y la Figura 5.24 el error de rotación que pertenecen a la trayectoria 5.

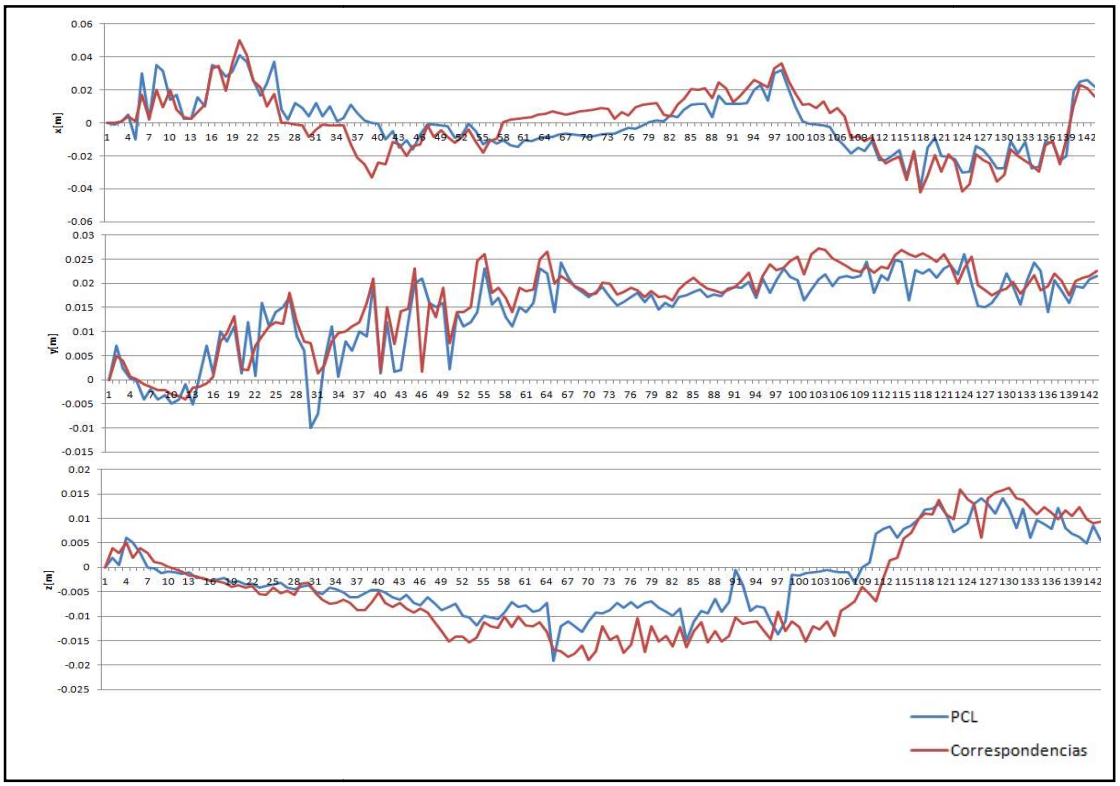


Figura 5.23 Error de traslación de trayectoria 5.

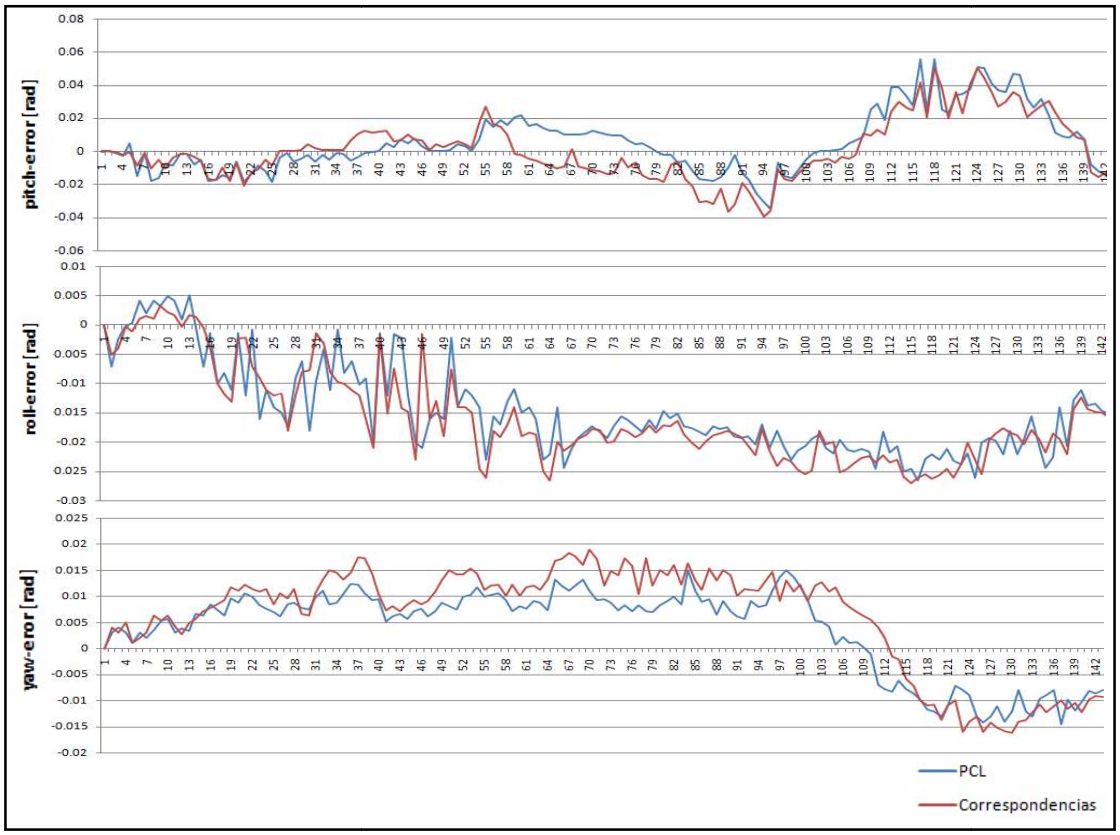


Figura 5.24 Error de rotación de trayectoria 5.

En la trayectoria 6 se utiliza desplazamiento hacia adelante en x, rotaciones en Yaw *derecha-izquierda* y rotaciones en $Pitch$ *arriba-abajo* Figura 5.25.

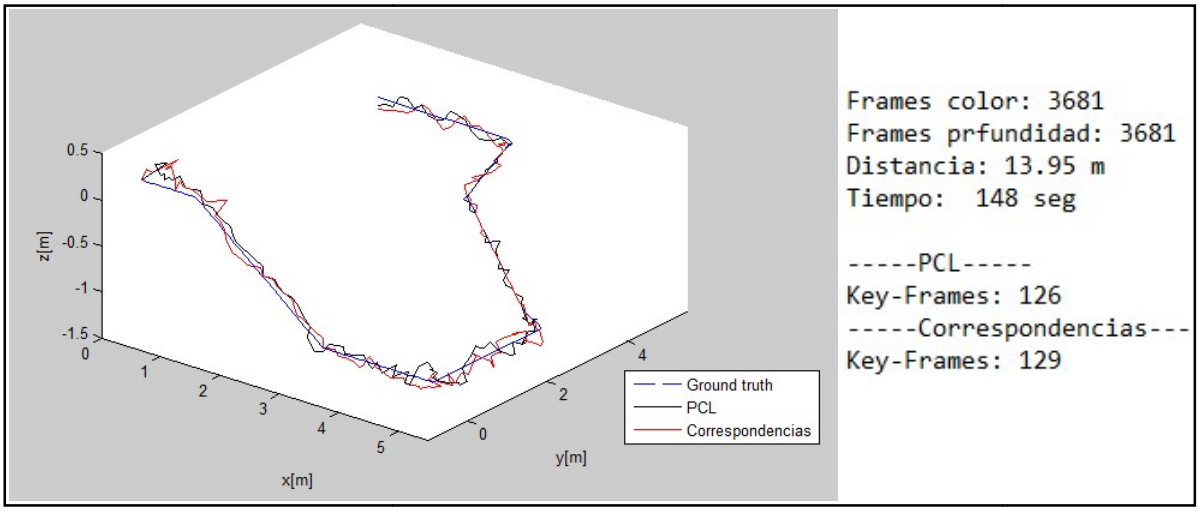


Figura 5.25 Trayectoria 6.

La Figura 5.26 muestra el error de traslación y la Figura 5.27 el error de rotación que pertenecen a la trayectoria 6.

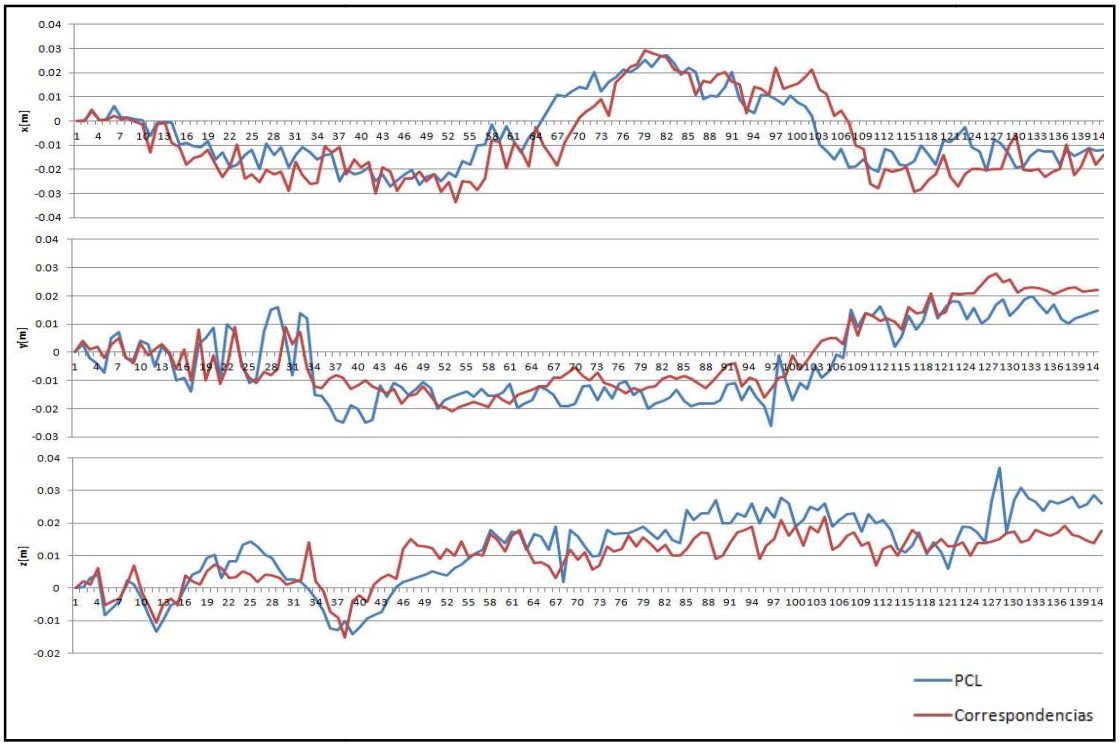


Figura 5. 26 Error de traslación de trayectoria 6.

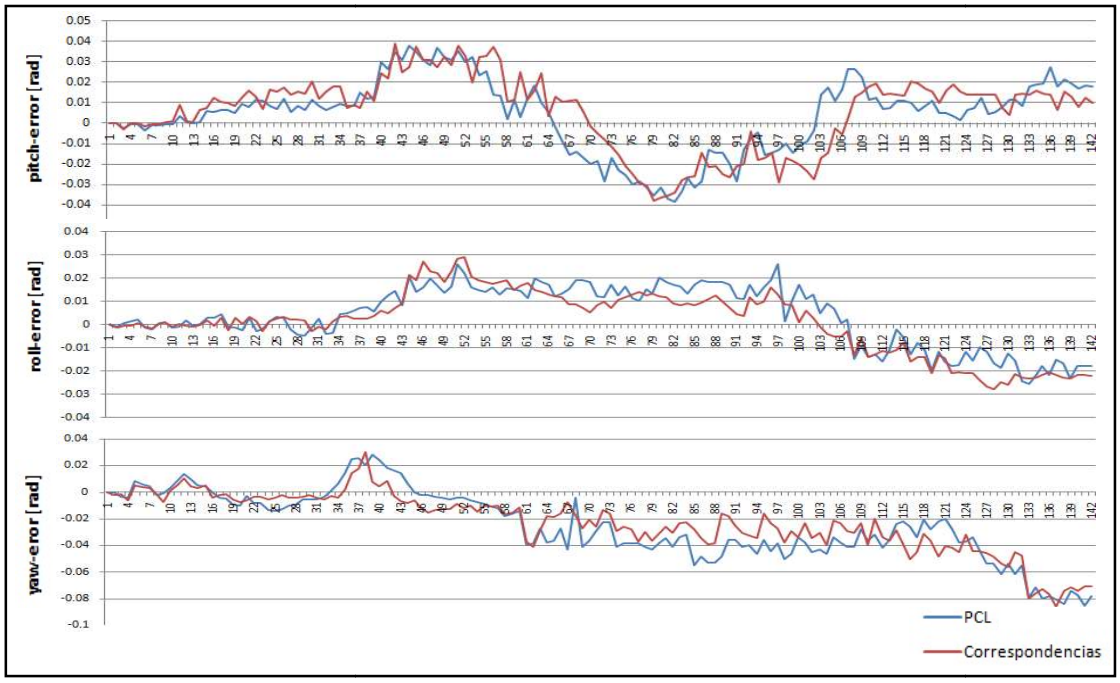


Figura 5.27 Error de rotación de trayectoria 6.

Los errores máximos obtenidos de traslación de las trayectorias se muestran en la Tabla 8. La Tabla 9 muestra los errores máximos de rotación.

En la Tabla 8 el desplazamiento esta en unidades de metros y los valores de la tabla 9 en radianes para describir la rotación, las celdas sombreadas representan el error mínimo.

Tabla 8. Errores máximos de desplazamiento obtenidos.

	Error máximo x[m]		Error máximo y[m]		Error máximo z[m]	
No.	**Correspondencias**	**PCL**	**Correspondencias**	**PCL**	**Correspondencias**	**PCL**
Trayectoria 1	0.0149	0.0142	0.1057	0.1328	0.0973	0.0958
Trayectoria 2	0.1721	0.1932	0.1232	0.1072	0.0392	0.0336
Trayectoria 3	0.0642	0.0622	0.0381	0.0281	0.0526	0.0672
Trayectoria 4	0.0295	0.0216	0.1528	0.1694	0.3712	0.2883
Trayectoria 5	0.0413	0.0581	0.2614	0.2736	0.1428	0.1625
Trayectoria 6	0.2734	0.2926	0.2291	0.2783	0.3741	0.2216

Tabla 9. Errores máximos de rotación obtenidos.

	Error máximo x[rad]		Error máximo y[rad]		Error máximo z[rad]	
No.	**Correspondencias**	**PCL**	**Correspondencias**	**PCL**	**Correspondencias**	**PCL**
Trayectoria 1	0.0278	0.0214	0.0042	0.0054	0.0253	0.0212
Trayectoria 2	0.0138	0.0228	0.0049	0.0057	0.0285	0.0177
Trayectoria 3	0.0753	0.0729	0.0194	0.0135	0.0411	0.0531
Trayectoria 4	0.0325	0.0327	0.0254	0.0301	0.0362	0.0328
Trayectoria 5	0.0504	0.0557	0.0042	0.0576	0.0152	0.0194
Trayectoria 6	0.0387	0.0392	0.0266	0.0293	0.0281	0.0304

5.6. Discusión

El sistema implementado, permitió crear 6 trayectorias con dos diferentes métodos, que fueron trazadas en el laboratorio de Inteligencia Artificial de Cenidet, en un espacio de 7, 7, 2 metros aproximadamente con respecto a ***x, y, z***. El método de Correspondencias 3D reporto un mejor desempeño debido a que el error de traslación Tabla 8 y rotación Tabla 9 es menor con respecto al método de PCL.

Capítulo 6. Conclusiones generales y trabajo futuro

En este capítulo se presentan los objetivos alcanzados, logros, las conclusiones finales y los trabajos futuros del presente trabajo así como las oportunidades de mejora.

6.1. Objetivos alcanzados

A continuación se presenta el objetivo general de esta tesis, el cual se definió al inicio de la investigación y su cumplimiento se mostró en la sección de resultados.

El objetivo primordial de esta tesis fue construir o mapear la trayectoria del movimiento en tres dimensiones de una cámara usando información de un sensor RGB-D.

Este sistema se limito a determinar la ubicación respecto a la pose inicial de la cámara con información visual y construir en un espacio tridimensional la trayectoria considerando el factor del tiempo mapeando una trayectoria.

Los detalles de los objetivos específicos alcanzados se explican a continuación y los alcanzados son señalados con una (✓) a la izquierda.

6.1.1. Objetivo general

✓ Construir o mapear la trayectoria del desplazamiento en tres dimensiones de una cámara usando información de un sensor RGB-D.

6. 1.2. Objetivos Específicos

✓ Determinar la ubicación respecto a la pose inicial de la cámara con información visual.

✓ Construir en un espacio tridimensional la trayectoria considerando el factor del tiempo.

✓ Mapear una trayectoria que genera un robot si se cuenta con él, de lo contrario será la trayectoria del sensor desplazada a mano.

6.2. Alcances

✓La trayectoria deberá ser tridimensional.

✓ Se usarán videos de interiores tomados con Kinect con una duración de 4 minutos. Una longitud de 16 metros y dimensiones del entorno 7, 7, 2 metros aproximadamente con respecto a ***x, y, z***.

✓Funcionamiento en tiempo real, aunque limitado por el hardware (computadoras notebook).

✓En caso de usar un robot móvil con algún otro sensor de distancia, podrá fusionar mediciones para propósitos odométricos.

✓El funcionamiento del sistema estará orientado a entornos interiores.
✓El sistema no considera la presencia de obstáculos fijos o móviles en la trayectoria.

6.3. Logros

Se desarrollo un sistema de odometría visual en lenguaje C++ con el cual se genera una trayectoria del desplazamiento del sensor RGB-D Kinect para trayectorias 3D en interiores. Se crearon seis trayectorias con dos métodos: método de PCL, método de correspondencia 3D. El movimiento de la cámara es de aproximadamente 0.09 m/seg. y la cámara funciona a 30 cuadros/seg.

6.4. Conclusiones

Se analizaron distintos métodos de odometría visual destacando ventajas y desventajas entre ellos. Se evaluaron distintos métodos de extracción de puntos destacados así como de correspondencias de puntos destacados. Se generaron trayectorias 3D utilizando una cámara RGB-D en nuestro caso la cámara de Kinect 360, con esta cámara se evita el cálculo de la profundidad como lo hacen otros métodos de odometría visual. Los compromisos establecidos en la propuesta de tesis se cumplieron en un 100%.

6.5. Otros resultados

Se genero un manual detallado de la instalación y configuración de OpenNI 2.1.0.4, OpenCV 3.0 y PCL 1.6.0 para el correcto funcionamiento en conjunto utilizando el IDE Visual Studio 2012. También se reportan errores de compatibilidad entres las librerías y su solución.

Trabajos futuros

La experimentación fue realizada con una cámara RGB-D en escenario de interiores, esta tendría problemas si se utilizara en escenarios de exteriores ya que si los puntos que se detectan en la imagen no están a menos de4 metros estos no se podrían utilizar debido a que carecen de información de profundidad, por este motivo se podrían explorar otros tipos de cámaras para trazar trayectorias en 3D.

El método desarrollado en esta tesis está restringido a generar trayectorias en 3D en interiores, donde hay ausencia de objetos en movimiento dentro de la escena por ello se podrían agregar algoritmos extras para la detección de objetos y que estos no influyan en el cálculo de la pose del robot.

Bibliografía

[Achtelik 2014] M. Achtelik, D. Scaramuzza, L. Doitsidis, F. Fraundorfer, E. Kos-matopoulos, A. Martinelli, M. Achtelik, M. Chli, S. Chatzichristofis,L. Kneip, D. Gurdan, L. Heng, G. Lee, S. Lynen, L. Meier, M. Polle-feys, A. Renzaglia, R. Siegwart, J. Stumpf, P. Tanskanen, C. Troiani,and S. Weiss, "Vision-Controlled Micro Flying Robots: from SystemDesign to Autonomous Navigation and Mapping in GPS-denied En-vironments," IEEE Robotics and Automation Magazine, 2014.

[Agarwal 2014] Aditya Agarwal, Daniel Maturana, and Sebastian Scherer, "Visual Odometry in Smoke Occluded Environments," doctoral dissertation, tech. report CMU-RI-TR-15-07, Robotics Institute, Carnegie Mellon University, July, 2014.

[Andrés 2012] Andrés Felipe Calvo Salcedo et al. "Procesamiento de nubes de puntos por medio de la librería PCL" Scientia et Technica Año XVII, No 52, Universidad Tecnológica de Pereira. ISSN 0122-1701, 2012.

[Arun 1987] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," IEEE Trans. Pattern Anal. Machine Intell., vol. 9, no. 5, pp.698–700,1987.

[Bay 2006] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In European Conference on Computer Vision May 2006.

[Burger 2010] Burger, W.; Bhanu, B. "Estimating 3D egomotion from perspective image sequence". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12 (11): 1040–1058. doi:10.1109/34.61704. 2010.

[Casagrande 2013] David S. Casagrande, "REAL-TIME FEATURELESS VISUAL ODOMETRY FOR SEA FLOOR IMAGING WITH A LAGRANGIAN FLOAT", Open Access Master's Theses. Paper 159, 2013.

[Conley 2011] Ken Conley, Documentation ROS, Official website, http://wiki.ros.org/kinect_node, Ultima consulta, 30-05-2016.

[Durrant 2006] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part I. The essential algorithms," Robot. Automat. Mag., vol. 13, no. 2, pp. 99–110, 2006.

[Engel 2012] J. Engel, J. Sturm, and D. Cremers, "Accurate Figure Flying with a Quadrocopter Using Onboard Visual and Inertial Sensing," in Proc.ViCoMoR Workshop at IEEE/RJS IROS, 2012.

[Engel 2013] J. Engel, J. Sturm, and D. Cremers, "Semi-Dense Visual Odometry for a Monocular Camera," in Proc. IEEE Int. Conf. on Computer Vision, 2013.

[Gómez 2013] Gómez Cruz Lázaro, Fusión de información de pose en robots móviles con visión, (Tesis de Maestría). Centro Nacional de Investigación y Desarrollo Tecnológico. Cuernavaca, México, 2013.

[Gonzáles 2011] Gonzáles, Sergio. Visión Binocular para Robot Móvil en Exteriores Simulados. (Tesis de Maestría). Centro Nacional de Investigación y Desarrollo Tecnológico. Cuernavaca, México. Febrero, 2011.

[Harris 1988] Chris Harris, Mike Stephens, A combined corner and edge detector, Plessey Research Roke Manor, United Kingdom, pp. 147-151, 1988.

[Irani 2010] Irani, M.; Rousso, B.; Peleg S. "Recovery of Ego-Motion Using Image Stabilization". *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*: 2010.

[Jürgen 2012] Felix Endres, Jürgen Hess et al. An Evaluation of the RGB-D SLAM System, IEEE International Conference on Robotics and Automation, 2012.

[Klein 2007] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 1–10, Nov. 2007.

[Lowe 2004] D. G. Lowe., Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91-110, 2004.

[Martinsanz 2001] Martinsanz, G.P., de la Cruz García, J.M., "Visión por computador: imágenes digitales y aplicaciones" RA-MA, ISBN: 9788478974726, 297-311, segunda edición, 2001.

[Meyer 2000] Carl D. Meyer, Matrix analysis and applied linear algebra, SIAM, ISBN: 978-0-898714-54-8, Primer edición, 2000.

[Mrpt 2016] Mobile Robot Programming Toolkit,http://www.mrpt.org, Ultima consulta, 30-05-2016.

[Nister 2004] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in Proc. Int. Conf. Computer Vision and Pattern Recognition, pp. 652–659, 2004.

[OpenCV 2016] Última actualización 1 Septiembre 2016, Clase SVD de OpenCV, http://docs.opencv.org/trunk/df/df7/classcv_1_1SVD.html

[PCL 2016] Última actualización 1 Septiembre 2016, Codigo para obtener la pose de la cámara, http://docs.pointclouds.org/trunk/common_2include_2pcl_2point__cloud_8h_source.html

[Peñaloza 2014] Karla Margarita Peñaloza Membrila. Navegación Visual en Trayectorias Cerradas. (Tesis de Maestría), Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Morelos, 2014.

[Peter 2012] Peter Henry, Michael Krainin, et al., RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments, The International Journal of Robotics Research 2012.

[Pizzoli 2014] M. Pizzoli, C. Forster, and D. Scaramuzza, "REMODE: Probabilistic, Monocular Dense Reconstruction in Real Time," in Proc. IEEE Int.Conf. on Robotics and Automation, 2014.

[Radu 2011] Radu Bogdan Rusu Steve Cousins, 3D is here: Point Cloud Library (PCL) IEEE International Conference on Robotics and Automation (ICRA), Rusu ICRA2011 PCL, Shanghai, China, mayo ,2011.

[Rosten 2006] E. Rosten and T. Drummond. Machine learning for highspeed corner detection. In European Conference on Computer Vision, volume I, 2006.

[Rublee 2011] Ethan Rublee, Vincent Rabaud, Kurt Konolige Gary Bradski , "ORB: an efficient alternative to SIFT or SURF" IEEE International Conference on Computer Vision 978-1-4577-1102, 2011.

[Scaramuzza 2011] D. Scaramuzza and F. Fraundorfer, "Visual Odometry, Part I: The First 30 Years and Fundamentals [Tutorial]," IEEE RAM, 2011.

[Scaramuzza 2014] Christian Forster, Matia Pizzoli, Davide Scaramuzza∗, SVO: Fast Semi-Direct Monocular Visual Odometry, ICRA, 2014.

[Shaojie 2012] Shaojie Shen; Michael, Nathan; Kumar, V., "Autonomous indoor 3D exploration with a micro-aerial vehicle," *Robotics and Automation (ICRA), 2012 IEEE International Conference on* , vol., no., pp.9,15, 14-18 May 2012.

[Smisek 2011] J. Smisek, M. Jancosek and T. Pajdla, "3D with Kinect," *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, Barcelona, 2011, pp. 1154-1160.

[Usenko 2016] Usenko, Vladyslav C. et al. "Direct visual-inertial odometry with stereo cameras." *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 1885-1892, 2016.

[Vergara 2015] Andres Vergara Bahena. Navegación, Localización y Mapeo de Robots Móviles para Trayectorias Pre-especificadas por Imágenes. (Tesis de maestría), Centro Nacional de investigación y Desarrollo Tecnológico, Cuernavaca, Morelos, 2015.

[Vogiatzis 2011] G. Vogiatzis and C. Hernández, "Video-based, Real-Time Multi View Stereo," Image and Vision Computing, vol. 29, no. 7, 2011.

[Xingshuai 2016] Xingshuai Dong1 & Bo He 1 & Xinghui Dong2 & Junyu Dong3, Monocular visual-IMU odometry using multi-channel image patch exemplars, Springer, Multimed Tools Appl DOI 10.1007/s11042-016-3927-8, 2016.

[Zárate 2016] Alida Esmeralda Zárate Jiménez. Detección de objetos en movimiento para un robot móvil con sensor RGB-D. (Tesis de Maestría). Centro Nacional de Investigación y Desarrollo Tecnológico. Cuernavaca, México. Febrero, 2016.

Anexo A

A.1. Detección de características

Imágenes obtenidas utilizando distintos métodos de extracción de características, las imágenes corresponden a *Frames* distintos.

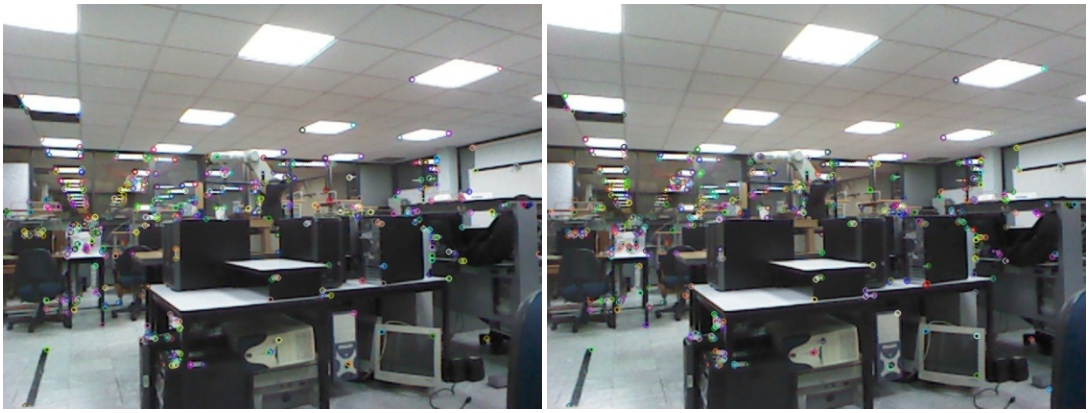


Figura A1. Imágenes usando GoodFeaturesToTrackDetector
Este método es una variación del método Harris.

Figura A2. Imágenes usando MSER.

FiguraA3. Imágenes usando ORB.

Figura A4. Imágenes usando SIFT.

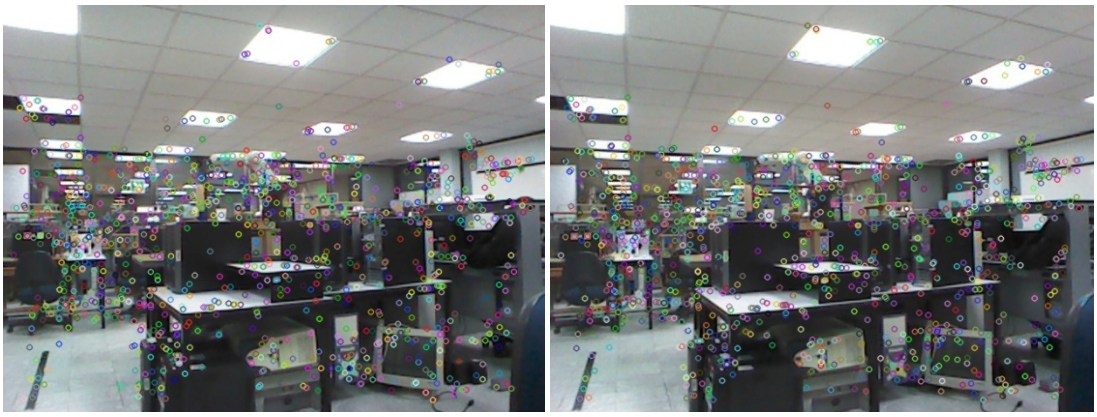


Figura A5.Imágenes usando SURF.

A.2.Correspondencia de puntos

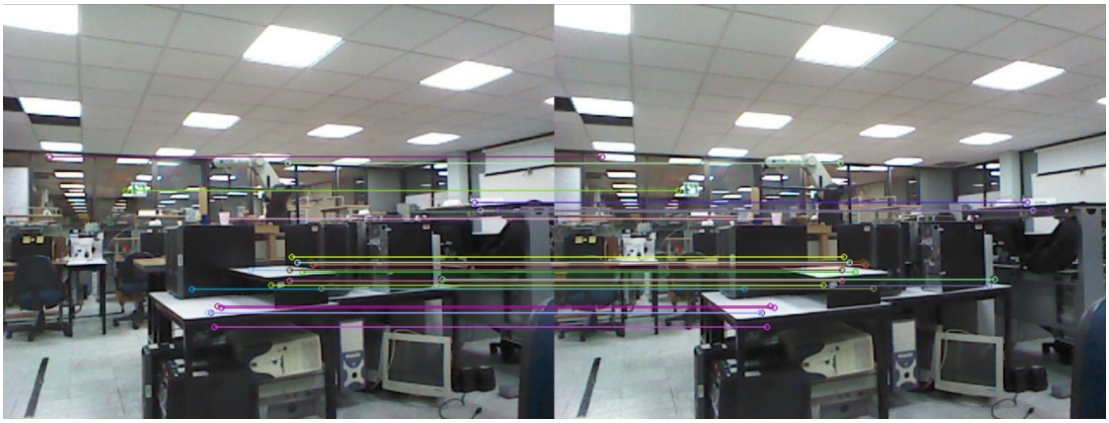


Figura A6. Imágenes usando SURF.

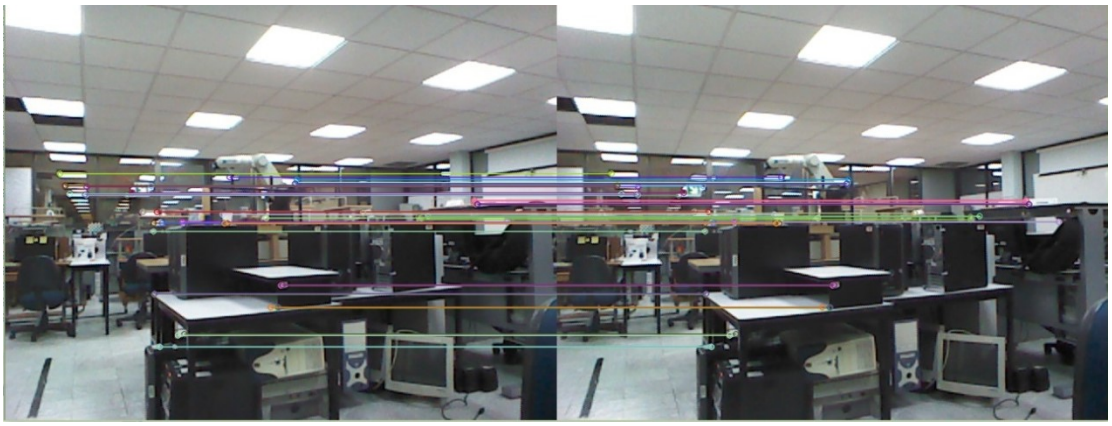


Figura A7. Imágenes usando ORB.

Figura A8. Imágenes usando SIFT.

Figura A9. Imágenes usando MSER.

Figura A10. Imágenes usando GoodFeaturesToTrackDetector.

A.3.Correspondencia de puntos con ORB

Se experimentó cambiando distintos parámetros del método ORB obteniendo distintos resultados.

A continuación se muestran algunos resultados de correspondencias usando ORB como descriptor: En la primera prueba se extrajeron 120 puntos destacados de cada imagen y 120 correspondencias. Para la siguiente se extraen 120 puntos destacados en ambas imágenes pero solo 83 correspondencias. Para la última prueba se extrajeron 120 puntos destacados de los cuales solo 39 tienen correspondencias.

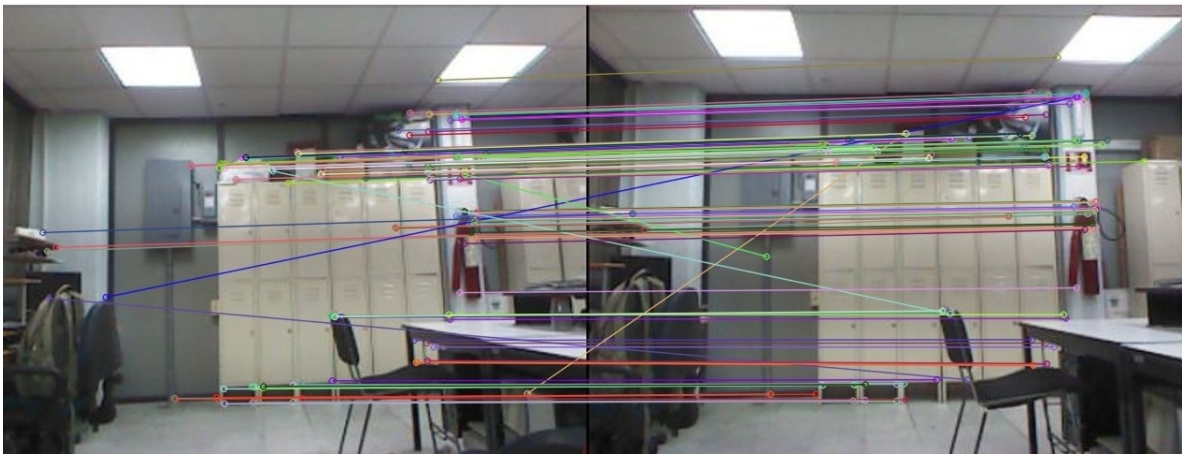


Figura A11. 120 correspondencias.

Figura A12. 83 correspondencias.

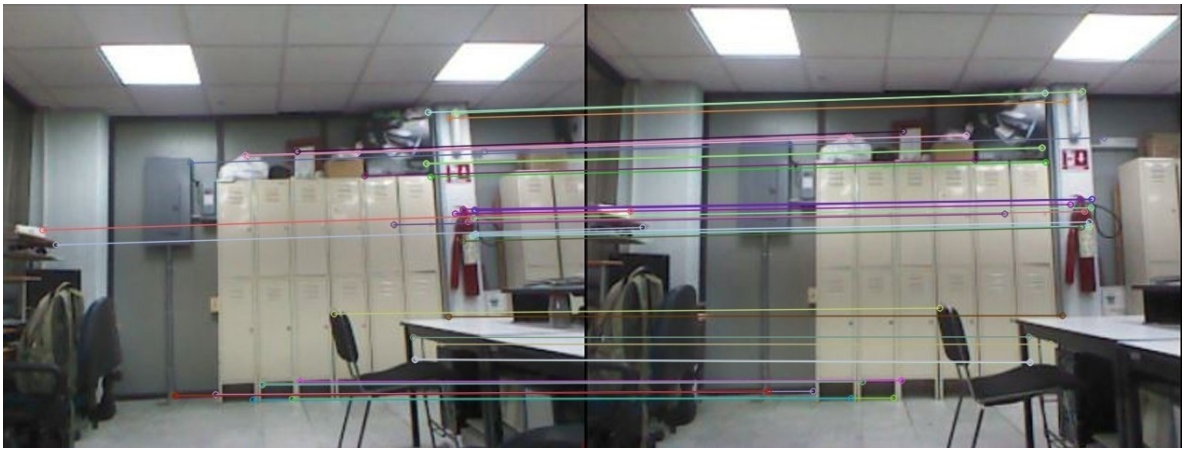


Figura A13. 39 correspondencias.

Anexo B

Herramienta de SVD que ofrece OpenCV.

SVD::SVD()

C++: *static* void SVD::**compute**(InputArray **src**, OutputArray **w**, OutputArray **u**, OutputArray **vt**, int **flags**=0)
C++: *static* void SVD::**compute**(InputArray **src**, OutputArray **w**, int **flags**=0)

Parámetros:
src – Matriz descompuesta
w – Valores singulares calculados
u – Vector izquierdo singular calculado
V – Vector derecho singular calculado
vt – Matriz transpuesta de valores singulares derechos
flags –Banderas de operación.

Operadores de banderas

SVD::MODIFY_A, utiliza el algoritmo para modificar la matriz descompuesta; Puede ahorrar espacio y acelerar el procesamiento.
SVD::NO_UV, indica que sólo se procesará un vector de valores singulares ***w***, mientras que ***u*** y ***vt*** se establecen en matrices vacías.
SVD::FULL_UV, cuando la matriz no es cuadrada, por defecto el algoritmo produce matrices ***u*** y ***vt***para la reconstrucción posterior de **A**; Sin embargo, si se especifica **FULL_UV**, ***u***y***vt*** serán matrices ortogonales cuadradas de tamaño completo.

Anexo C

Interface de usuario

La interface del sistema está compuesta por 4 principales funciones Figura D1, la primera es: iniciar el sistema al pulsar el botón *iniciar,* la cámara se enciende desplegando el video obtenido de Kinect, imagen de color y profundidad, también se despliega una gráfica en 3D donde se visualizan los datos obtenidos de pose, los cuales también son desplegados.

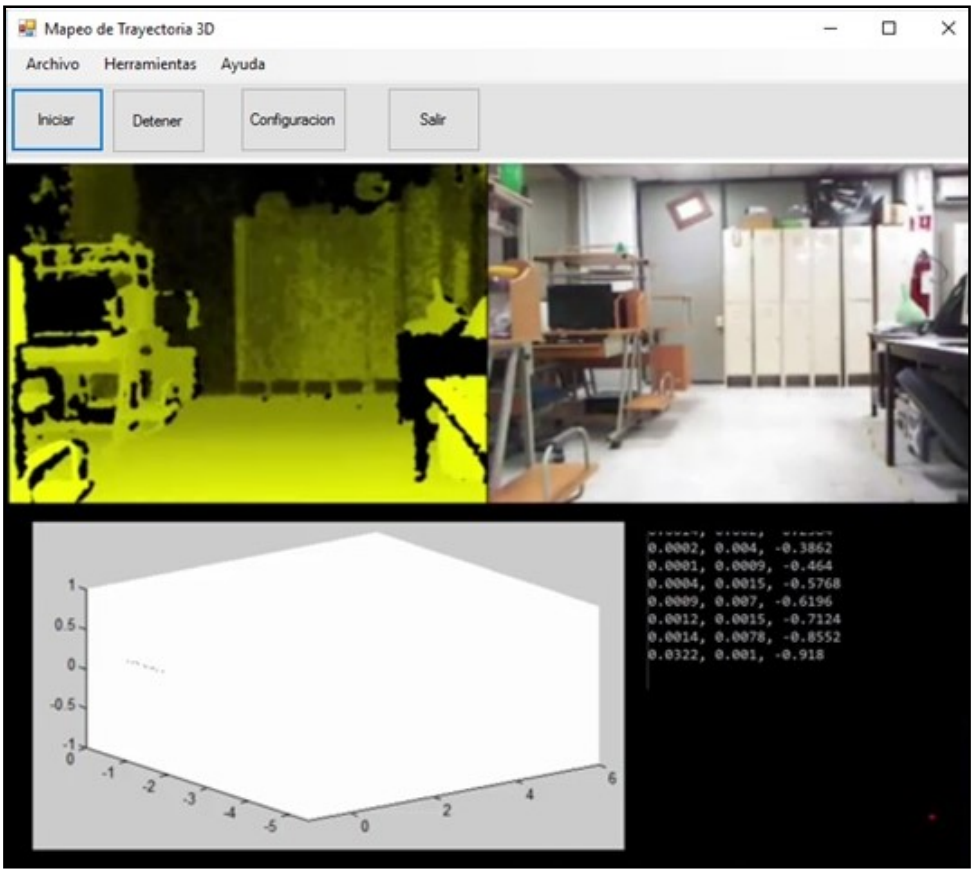


Figura D1. Interface principal del sistema.

Las configuraciones del sistema se muestran en la Figura D2, donde se puede seleccionar el tipo de extractor de puntos, el meto de correspondencia de puntos, las coordenadas iníciales donde estará posicionado el robot dentro del mapa.

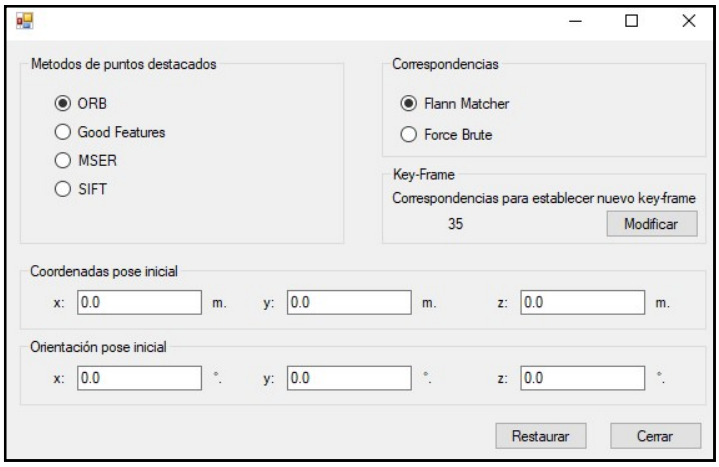


Figura D2. Configuraciones del sistema.

La Figura D3 muestra una herramienta que permite establecer un umbral para la selección de un nuevo *key-Frame*. La cantidad de 35 puntos es el valor por defecto. Entre más grande es el número que se seleccione los *key-Frame* estarán más cerca.

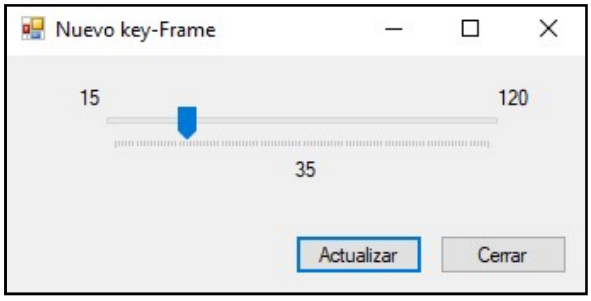


Figura D3. Porcentaje de puntos para establecer nuevo *Key-Frame*.

Anexo D

Se realizaron algunos experimentos utilizando varios paquetes de odometría visual *OpenSource*así como de SLAM, los cuales presentaron algunos problemas de compilación debido a incompatibilidades de librerías y conflictos en el llamado de clases.

SVO (Semi-Direct Visual Odometry)

El sistema de odometría visual semi-directa se encuentra disponible en *https://github.com/uzh-rpg/rpg_svo.* Se descargaron los archivos pero al momento de compilar estos generaban errores advirtiendo que el programa no podía ser compilado debido a que faltaban algunas librerías o no se encontraban en la ruta establecida.

El desarrollador de este software ofrece los módulos del sistema por separado de los cuales algunos se pudieron compilar pero otros no.

MRPT (Mobile Robot Programming Toolkit)

MRPT proporciona aplicaciones y bibliotecas portátiles y probadas que cubren estructuras de datos y algoritmos empleados en áreas comunes de investigación robótica; es de código abierto, publicado bajo la licencia BSD.

Es una aplicación disponible en *http://www.mrpt.org/* es sencilla de instalar el problema se presenta al configurar la cámara Kinect.