Construcción de un modelo de memoria visual para la navegación de robots humanoides

Presenta:
Ing. Alan López Martínez

Como requisito para obtener el título en:
MAESTRO EN OPTOMECATRÓNICA

Asesor:
Dr. Francisco Cuevas de la Rosa
Coasesor:
Dr. Jean Bernard Hayet

León Gto, a 10 de noviembre de 2014

Agradecimientos

Quiero expresar mi gratitud al CONACYT por el apoyo económico brindado. De igual forma agradezco al CIO por la oportunidad de ingreso al programa de maestría y al CIMAT por el espacio ofrecido para hacer uso del laboratorio de robótica.

Agradezco también a mis asesores: el Dr. F.J. Cuevas y el Dr. J.B. Hayet. Ellos contribuyeron con su experiencia y sugerencias a que lograra terminar el proyecto de tesis. También agradezco de especial forma al Dr. H. Becerra, quien compartió de su tiempo y experiencia para el desarrollo de mi tesis.

De igual forma quiero expresar mi gratitud a mis revisores: el Dr. Asención Guerrero y la Dra. Ma. del Socorro Hernández. Su tiempo y ayuda permitieron mejorar mi tesis.

Resumen

El problema de navegación de robots consiste en el desplazamiento seguro de éstos dentro de un ambiente, sea éste estructurado o no, lo cual implica la solución a tres problemas diferentes: 1. Determinación de una representación eficiente y adecuada del ambiente a navegar. Esto a través de un mapa que describa el espacio de navegación; 2. Auto-localización. Es necesario que el robot logre localizarse en el mapa; 3. Diseño de leyes de control. Debe ser posible el desplazamiento del robot de un punto A a un punto B con leyes de control adecuadas. La presente tesis expone el trabajo realizado con el objetivo de proveer una solución a la primera de las problemáticas ya expuestas: el modelado del ambiente de navegación. Existen varios trabajos que proponen soluciones para la adecuada representación de un ambiente [26, 8, 17]. Tales representaciones pueden ser divididas en dos familias: las geométricas y las topológicas. En las primeras, el espacio de navegación se representa en un marco de referencia euclidiano, mientras que en las segundas el ambiente es descrito de forma cualitativa. El proyecto se orienta a la construcción de un mapa topológico, es decir, un mapa sin información métrica del ambiente, en la forma de una memoria visual. Dicha memoria puede estar constituida por tres partes. La primera es un grafo donde cada imagen que se considera clave bajo ciertos criterios y que modela topológicamente el ambiente, es un nodo. La segunda mantiene los mismos nodos que la primera pero asigna una arista a cada nodo. Y la tercera disminuye el número de nodos estructurando el grafo incluyendo aristas en caminos visuales.

Índice general

Índice de figuras

1 Introducción

Existen diversos trabajos de investigación que proponen métodos para dotar de autonomía a los robots al momento de conducirse por un ambiente. Google^{TM} por ejemplo, ha desarrollado un automóvil sin conductor que es capaz de desplazarse en calles y avenidas detectando señales de tránsito, peatones y carriles para vehículos entre otros. Sin embargo, la navegación autónoma de un robot es imposible sin algún conocimiento sobre el "mundo" en el que navega. Es por ello que surge la necesidad de una descripción del entorno que sea lo suficientemente robusta para la navegación del robot.

Actualmente existen diversos enfoques para la construcción de mapas que sirven al robot para su navegación autónoma. Es posible describir estos mapas de forma cuantitativa y/o cualitativa. El proyecto de tesis, aquí presentado, se enfoca en la construcción de un modelo topológico o cualitativo del ambiente de navegación; para ello podemos imitar en cierta medida a la forma en que los seres humanos creamos este tipo de representaciones.

Cuando una persona camina por primera vez en un entorno desconocido, lo que comunmente hace es memorizar algunas imágenes clave del camino recorrido para después usarlas como referencia en un recorrido posterior por ese entorno. Por ejemplo, un ser humano no almacena en su memoria datos métricos del entorno, sino información perceptual como imágenes que le parecen clave [41].

Lo mencionado en el párrafo anterior plasma la idea central de una memoria visual, es decir, la representación de un ambiente con una colección mínima de imágenes que conforman el mapa de navegación. El paradigma de la navegación basada en una memoria visual ha sido utilizado en su mayoría para ser aplicado en robots móviles tipo carro, y pocos son aquellos que buscan una descripción del entorno de navegación dirigida a robots humanoides; las complicaciones asociadas a este tipo de robots yacen en el movimiento brusco de éste al caminar, provocando así la captura de imágenes borrosas. Por lo que el proyecto busca una solución en particular para la construcción de una memoria visual usando imágenes capturadas por una cámara a bordo de un robot humanoide.

Memoria Visual

Esta tesis expone la representación de un ambiente para la posterior navegación de un robot humanoide; tal representación es construida como una Memoria Visual

(MV). El sensor es una cámara proyectiva a bordo del robot, y las trayectorias por las cuales el robot tiene permitido navegar se describen con un conjunto de imágenes (imágenes clave) que caracterizan el ambiente. A esta colección de imágenes la llamamos memoria visual.

La metodología para la construcción de la memoria visual es exhibida en la Fig. 1.1. La construcción de ésta se divide en tres etapas: la primera etapa inicia con una fase de aprendizaje, en ésta el robot es conducido por un humano a través del entorno, mientras el robot captura imágenes de entrenamiento; la captura de estas imágenes es a una tasa constante. Después, este conjunto de imágenes se procesa en la etapa dos, que busca reducir el número de imágenes que conformarán la MV. Para lograr tal reducción, se seleccionan sólo algunas de las imágenes del conjunto, que son llamadas imágenes clave. Para ello se analiza la similitud entre imágenes, o su traslape, y se desechan las imágenes que no cumplen ciertos criterios. Tal selección también toma en cuenta el cumplimiento de algunas hipótesis necesarias para el control del robot durante la navegación.

Una vez que se tiene un conjunto de imágenes clave, el cual denominamos conjunto M1, se organizan la imágenes pertenencientes a éste para formar una representación topológica del ambiente de navegación, es decir, un grafo. Esto último ocurre en la etapa tres de la construcción de la memoria visual.

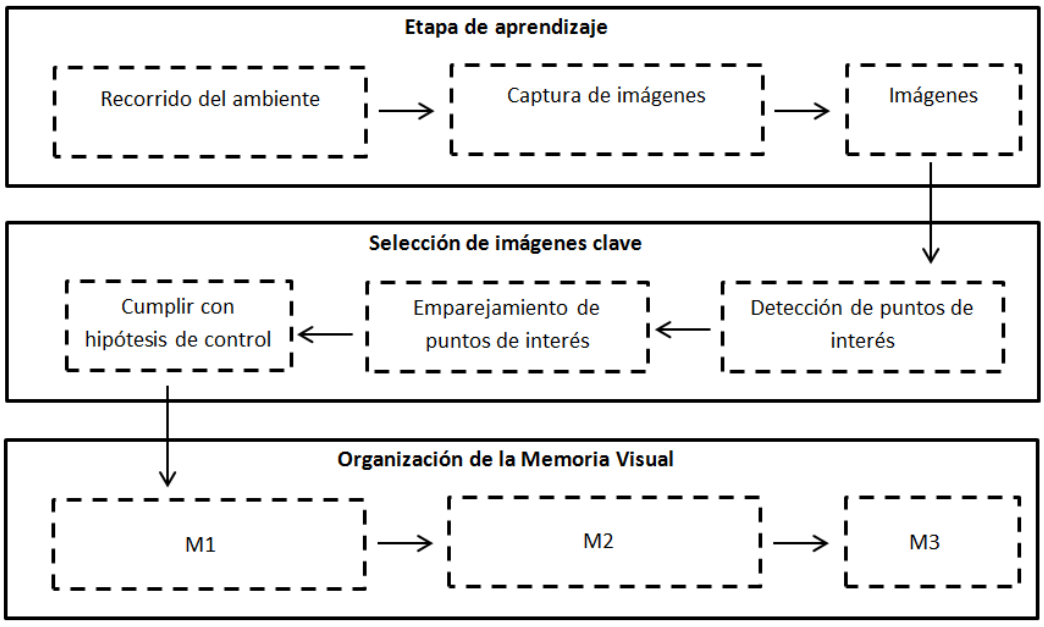


Figura 1.1: *Memoria Visual. Metodología.*
La metodología para la construcción de la Memoria Visual puede ser dividida en tres etapas diferentes. La primera etapa implica la captura de imágenes de entrenamiento. La segunda selecciona sólo las imágenes necesarias para modelar el ambiente. La última etapa organiza estas imágenes clave en un grafo.

1.1. Motivación

Actualmente existen diversas técnicas que buscan dotar a los robots de autonomía en cuestiones de navegación [26, 5], esto es, que el robot sea capaz tanto de conocer su localización en el entorno, como de planear el camino por el cual desplazarse para alcanzar una ubicación objetivo, todo sin ayuda externa.

Se busca conceder autonomía a los robots comerciales e industriales que son ampliamente utilizados, pues realizan tareas de forma más exacta o a menor costo que los seres humanos. También se les utiliza en trabajos demasiado sucios, peligrosos o tediosos para los humanos. Los robots son muy utilizados en plantas de manufactura, montaje y embalaje; en transporte; en exploraciones en la Tierra y en el espacio; en cirugía, armamento, investigación en laboratorios y en la producción en masa de bienes industriales o de consumo. Como ejemplo, la construcción de la MV puede ser utilizada para la navegación autónoma de robots en almacenes industriales. Esta aplicación es considerada como estudio de mercado en [23].

Para lograr autonomía en navegación, se necesita un modelo del entorno por el cual el robot debe desplazarse. De lo contrario la navegación autónoma es imposible. Se ha recurrido a la reconstrucción en 3D del entorno de navegación, por ejemplo con ayuda de láseres, aunque tal reconstrucción puede ser demasiado costosa computacionalmente. Esta es una de las razones por las que en esta tesis se propuso construir un mapa topológico que sea capaz de modelar apropiadamente el ambiente de navegación. Este tipo de mapas se ha hecho popular para robots móviles tipo carro, pero pocas son las contribuciones al modelo topológico del ambiente para la navegación de robots humanoides.

Es por ello que en este trabajo de tesis se busca la *Construcción de un Modelo de Memoria Visual para la Navegacón de Robots Humanoides*. Y para lograr esta meta se propusieron los siguientes objetivos.

1.2. Objetivos

1.2.1. Objetivo general y objetivos particulares

El objetivo de esta tesis es generar soluciones a problemas de Visión por Computadora (CV) generados en el proceso de modelado topológico de un ambiente en forma de memoria visual. Esto mediante el uso de herramientas de Computación Evolutiva (CE). Para lograr dicho objetivo principal, se han establecido los siguientes objetivos particulares:

1. Generar una solución para la optimización de funciones de más de una variable con algoritmos genéticos que pueda ser aplicada para mejorar la construcción del modelo de memoria visual.

2. Generar una solución que sea capaz de encontrar buenos emparejamientos o correspondencias de puntos de interés entre dos imágenes. Tal solución debe ser robusta a diferentes escalas, iluminación y rotaciones.
3. Calcular una solución que estime apropiadamente la matriz fundamental entre un par de imágenes. Esto para asegurar la correcta selección de imágenes clave en función de algunas hipótesis de control del robot.
4. Obtener una solución adecuada y eficiente que resuelva el problema de la organización topológica de la memoria visual.

1.3. Contribuciones

A continuación se exponen las principales contribuciones de esta tesis:

1. Se propuso una solución eficiente al problema de optimización de funciones de más de una variable mediante el uso de algoritmos genéticos. Particularmente en relación a la estimación de la matriz esencial, que codifica la posición de las cámaras, y en consecuencia, el traslape entre imágenes.
2. Se resolvió, eficientemente, el problema de la búsqueda de emparejamientos entre imágenes con una combinación del algoritmo genético ya mencionado y de otros métodos.
3. Se implementó el algoritmo de 8 puntos para la estimación de la matriz fundamental y así contribuir al análisis del cumplimiento de las hipótesis de control.
4. Se desarrolló una interfaz gráfica para la visualización de la memoria visual en un grafo.

1.4. Estructura del documento

De acuerdo con la literatura, existen diversas condiciones que tienen que ser respetadas para la construcción de una memoria visual, y para el buen funcionamiento de un algoritmo genético. Estas condiciones así como el trabajo relacionado al proyecto de tesis son presentadas en el capítulo 2.

En el capítulo 3, se exponen los fundamentos básicos para entender el funcionamiento de los algoritmos genéticos y de la memoria visual.

En el capitulo 4, se presenta la metodología seguida para el desarrollo del proyecto de tesis. Los resultados del proyecto son expuestos en el capítulo 5. Y por último, en el capítulo 6, se analizan los resultados y se comentan las conclusiones y el trabajo futuro.

2 Estado del arte

Las contribuciones de varios investigadores han generado una gama de algoritmos que permiten realizar la construcción de un modelo de algún ambiente para la navegación de robots. Lo mismo para el área de la computación evolutiva, donde se ha contribuído con algoritmos genéticos para la resolución de problemas de visión por computadora que pueden tratarse como problemas de optimización.

En este capítulo se presentan algunas contribuciones al estado del arte de los temas antes mencionados. Además, a partir de esto se justifica la elección del paradigma de memoria visual y su uso en la navegación de robot humanoides.

2.1. Representación del ambiente de navegación

Desde los años setenta se ha considerado que un robot puede construir un mapa de su entorno de navegación, y a partir de los años ochenta se han propuesto enfoques para la reconstrucción en tres dimensiones del entorno en el contexto de navegación de robots [16].

2.1.1. Representaciones 3D

Un enfoque para la representación del entorno de navegación son los mapas 3D densos que son construidos con información visual [28]. Este enfoque presenta una desventaja: es computacionalmente costoso.

Una forma para reducir el costo computacional es extraer parte de la información visual. En lugar de construir un modelo 3D denso con millones de voxeles, una representación 3D no densa es construida con sólo características especiales. Ejemplos de trabajos que hacen uso de esta técnica son [35, 14]. Aún cuando es menos denso que el mapa presentado en [28], su enfoque sigue siendo computacionalmente costoso para ambientes grandes y complejos.

2.1.2. Representaciones topológicas

Cuando no estamos interesados en describir el ambiente cuantitativamente sino cualitativamente, recurrimos a representaciones topológicas. En este tipo de representaciones, los diferentes lugares forman un grafo donde cada lugar es un nodo.

Memoria visual

En [26], Matsumoto et al. proponen un modelo para representar un ambiente de navegación denominado VSRR (View-Sequenced Route Representation). Esta representación es construida a partir de imágenes en secuencia capturadas por un robot móvil tipo carro a lo largo de corredores. Esta representación se obtiene mediante el almacenamiento de imágenes de vista frontal en resolución reducida a lo largo de una ruta durante una etapa de navegación supervisada. En este trabajo el robot se mueve con una primitiva de movimiento conocida hasta que la imagen actual cambia un cierto grado desde la última imagen memorizada. La diferencia entre dos imágenes se obtiene mediante el emparejamiento de una plantilla (template matching) por correlación. Como resultado se obtiene el desplazamiento horizontal de la plantilla y un valor de correlación, el cual se compara con un umbral para determinar el grado de cambio de la imagen actual y entonces guardar una nueva imagen clave cuando la correlación es baja. La desventaja del sistema propuesto es su limitación a corredores, ya que no se construye una memoria para la navegación de un cuarto completo por ejemplo.

En [7, 8], Courbon et al. presentan un modelo de navegación autónoma basado en imágenes. El método propuesto se divide en tres etapas: 1) construcción de la memoria visual; 2) localización, y 3) navegación autónoma. En la primera etapa, el robot tipo carro captura una secuencia de imágenes del ambiente cuando es dirigido por un humano; después esta secuencia se reduce y organiza en caminos visuales. En la segunda etapa, el robot se localiza comparando la imágen que "observa" la cámara con las imágenes de la MV. Por último, la tercera etapa consiste en el diseño de leyes de control basada en imágenes, con el propósito de guiar al robot de la ubicación actual a la ubicación objetivo.

En [17], Goedeme presenta una MV para la navegación autónoma de una silla de ruedas. La construcción de la MV parte de una etapa de aprendizaje donde se capturan imágenes a una tasa constante de un ambiente variado. Después, estas imágenes son agrupadas por "lugares". A diferencia de Courbon, Goedeme construye *clusters* de imágenes con la teoría Dempster-Shafer para después realizar comparaciones globales y locales con el fin de encontrar la similitud entre imágenes.

En [21], Ido et al. proponen la navegación de interiores para un robot humanoide HRP-2. Para lograr este objetivo y hacer frente al *blur* y *swing* carácteristicos de las caminatas de los humanoides, se contruye una representación topológica del ambiente de navegación basada en la detección del flujo óptico.

El proyecto de tesis busca una representación topológica del ambiente de navegación en forma de una memoria visual. Esto reduce el costo computacional de reconstruir la escena en tres dimensiones. Además de hacer el cálculo de trayectorias más fácil y rápido.

Tradicionalmente se han usado GPS o láseres para la navegación. Una desventaja de un GPS es su dependencia a satélites, por lo que no puede ser usado en interiores.

Por otro lado, los láseres pueden ser voluminosos y más caros que una cámara digital. Por eso, se propone una solución únicamente basada en visón por computadora. Las cámaras son compactas y baratas, además de estar ya embebidas en la plataforma robótica que se usará, que es el robot humanoide Nao [2].

2.2. Algoritmos genéticos en visión por computadora

Existen varios trabajos que usan Algoritmos Genéticos (AG) para el procesamiento de imágenes. Por ejemplo, en [3] se aplican en la búsqueda de círculos en imágenes, codificando 3 puntos como cromosomas de círculos candidatos. Después, la función de aptitud evalúa si tales círculos candidatos se encuentran en la imagen. El método de selección utilizado en este trabajo es la rueda de ruleta.

En [15], los AG son usados para la búsqueda de las mejores características visuales para la detección de rostros. Para esto se codifica como cromosoma al vector del descriptor SIFT de 128 elementos. Mientras que la función de aptitud es evaluada con las distancias euclideanas de los puntos de interés emparejados.

En [22], los AG son usados para la detección en imágenes de matrículas vehiculares. Y en [25] se compara el uso de algoritmos genéticos con la transformada de Hough para la detección de primitivas en imágenes, es decir, figuras geométricas sencillas.

En [39], se hace uso de un AG y del algoritmo ICP para superponer nubes de puntos en el espacio. Se obtienen diferentes nubes de puntos de un objeto encontrando la superficie del mismo en diferentes vistas. Se hace la reconstrucción 3D completa del objeto empalmando las nubes de puntos de las diferentes vistas. En [40] también se hace la reconstrucción 3D. Se presenta un método para la reconstrucción completa de objetos con la ayuda de un digitalizador. Las constribuciones de este método se extienden a la ingeniería inversa.

Las contribuciones de los AG también se extienden a la metrología óptica [10]. Por ejemplo, en [9, 11], se resuelve el problema de demodulación de patrones de franjas. Para solucionar el problema de forma más rápida y eficiente, el algoritmo divide el patrón en subimágenes.

3 Fundamento Teórico

Para fundamentar la propuesta de un método para construir una Memoria Visual (MV) que represente un entorno de navegación, en este apartado se detallan los elementos teóricos necesarios para entender el proceso de la construcción de la MV. Se aborda con particular detalle el marco teórico de MV y de algoritmo genético, pues son el núcleo del proyecto de tesis.

3.1. Teoría de grafos

Como se comentó en el capítulo uno, la MV es representada en términos de grafos. De forma que previo a explicar las definiciones de cada uno de los elementos que componen la MV, se exponen de forma breve algunas de las definiciones de la teoría de grafos.

Un grafo $G = (N, A)$ es un conjunto finito de nodos $N = \{N_1, N_2, \ldots, N_n\}$ asociado a un conjunto de aristas $A = \{A_1, A_2, \ldots, A_{na}\}$. Cada arista puede ser definida por dos nodos como sigue: $A_r = A_{s,t} = \{N_s, N_t\}$.

Consideremos las siguientes definiciones:

- El tamaño de un grafo es definido por el número n_a de aristas.
- G es un grafo vacío cuando $n_a = 0$.
- G es un grafo orientado si todos lo elementos de A son aristas orientadas, esto es $A_{s,t} \neq A_{t,s}$.
- Una cadena Γ de un grafo G es un conjunto de nodos adyacentes.
- Un grafo es conexo cuando todo par de nodos distintos es unido por una cadena Γ.
- Un grafo ponderado es un grafo G provisto de una función que asocia un peso a las aristas, $\gamma : A \to R^+$ ó a los nodos, $\gamma : N \to R^+$.

3.2. Memoria visual

3.2.1. Definición

Una memoria visual es una representación topológica de un entorno, y es construida a partir de una secuencia de imágenes organizadas. Esta representación

es utilizada como referencia para permitir y guiar la navegación del robot, la cual debe asegurar un camino de una posición inicial a una final donde el desplazamiento del robot pueda ser controlado. El modelo de la MV puede representarse como un grafo donde cada nodo representa una ubicación del robot en el ambiente de navegación.

Supongamos que queremos construir la MV para el ambiente mostrado, como un ejemplo, en la Fig. 3.1. Se propone la construcción de la memoria compuesta por tres elementos, MV = {M1, M2, M3}; donde M1, M2 y M3 son descritos en términos de grafos, y son definidos más adelante.

La primera etapa para la construcción de la MV consiste en la etapa de aprendizaje, en la que el robot es guiado por un humano a través del entorno por cada pasillo y almacén, mientras el robot captura una secuencia de imágenes $S = \{F_i | i \in \{1, 2, \ldots, m\}\}$.

En una segunda etapa fuera de línea, el número de imágenes de la secuencia capturada en la primera etapa se reduce, descartando algunas imágenes de la secuencia bajo ciertos criterios. Las imágenes que no fueron desechadas se definen como imágenes clave, y son las suficientes y necesarias para una adecuada representación del ambiente. A este conjunto de imágenes clave le llamaremos M1.

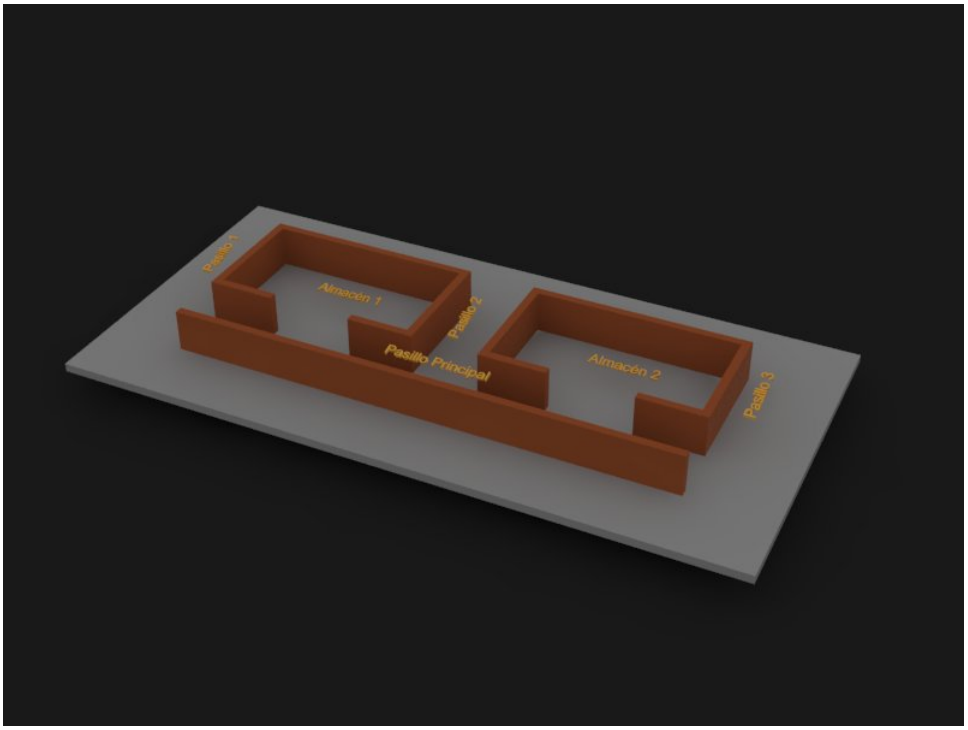


Figura 3.1: *Ejemplo de ambiente a modelar mediante una memoria visual.*

En términos de grafos, $M1 = (N_{M1}, A_{M1})$, con un conjunto de nodos N_{M1} igual

al número de imágenes clave y un conjunto de aristas A_{M1} vacio:

$$A_{M1} \quad \text{Conjunto vacío}$$
$$N_{M1} = \{I_i | i \in \{1, 2, \ldots, n\}\}$$

donde I_i es la íesima imagen clave y n es el número de imágenes clave.

M2 se define como un mapa topológico de bajo nivel. Para construir M2, hacemos que cada imagen clave sea un nodo, y cada nodo es a su vez unido por una arista al nodo siguiente, como se muestra en la Fig. 3.2. Cabe resaltar que las aristas que unen cada nodo pueden ser ponderadas con una función γ.

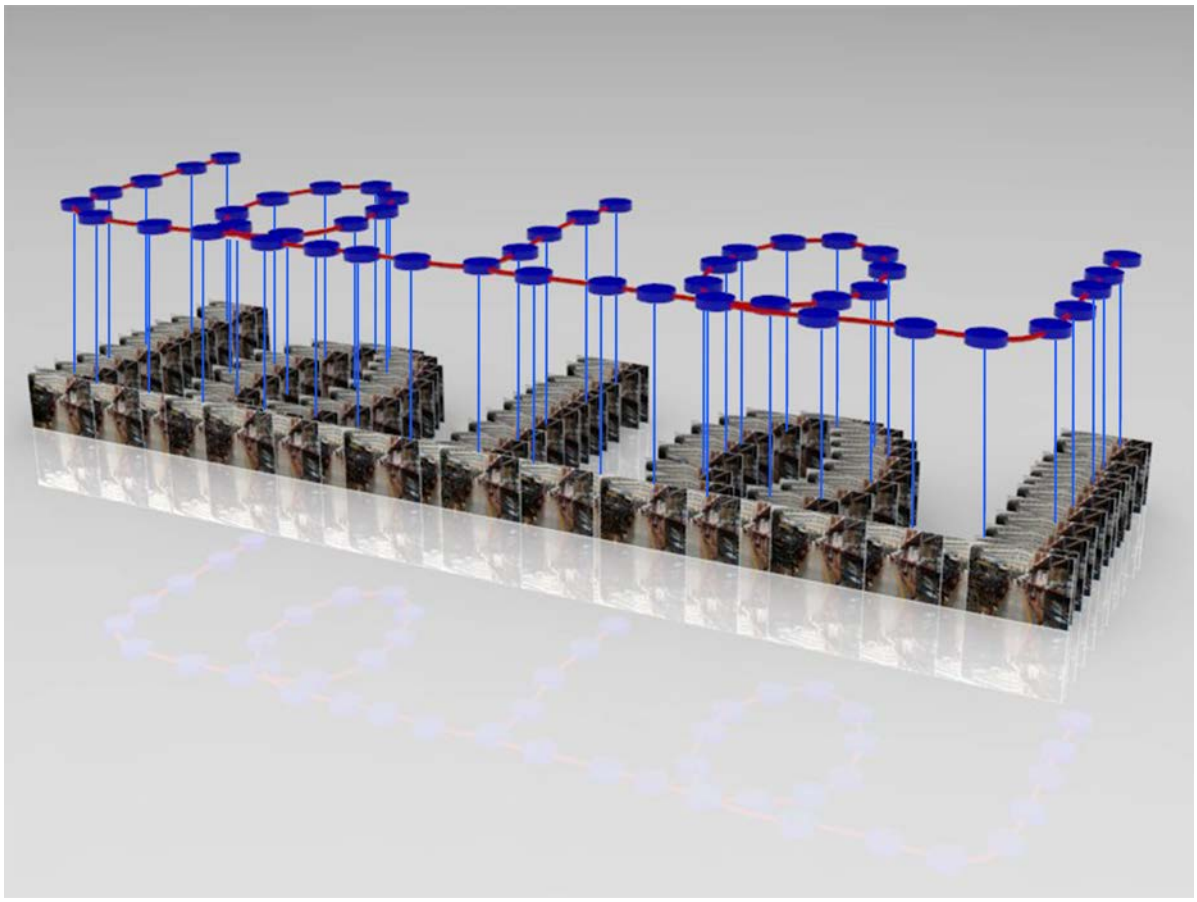


Figura 3.2: *M2.*
Para M2 se hace que cada imagen clave sea un nodo. A diferencia de M1, M2 une cada nodo con una arista para crear un mapa topológico de bajo nivel.

Así el grafo que representa a M2 queda definido como, $M2 = (N_{M2}, A_{M2})$:

$$N_{M2} = \{N_i | i \in \{1, 2, \ldots, n\}\}$$

donde el nodo N_i corresponde al lugar de adquisición de la imagen I_i. Y existe un conjunto de aristas A_{M2} que une cada uno de los nodos. Se busca además que el

grafo M2 sea conexo, es decir, que garantice la existencia de un camino de cualquier nodo N_a a cualquier otro nodo N_b.

Por último, M3 de MV es un mapa topológico de alto nivel, el cual se construye dividiendo a M2 en caminos visuales como se aprecia en la Fig. 3.3. M3 es definido como $M3 = (N_{M3}, A_{M3})$:

$$N_{M3} = \{\Gamma^1, \Gamma^2, \ldots, \Gamma^{n_\Gamma}\},$$

donde

$$\Gamma^i = \{N_j^i \,|\, j \in \{1, 2, \ldots, n_{\Gamma^i}\}\}.$$

Las aristas A_{M3} son las que unen el nodo final $N^i_{n_{\Gamma^i}}$ del camino visual i con el primer nodo N_1^j del camino visual j.

La tarea de navegación supone grandes desplazamientos para el robot. Así, la representación de la MV de la forma descrita es conveniente, pues permite dividir la tarea de navegación en objetivos intermedios.

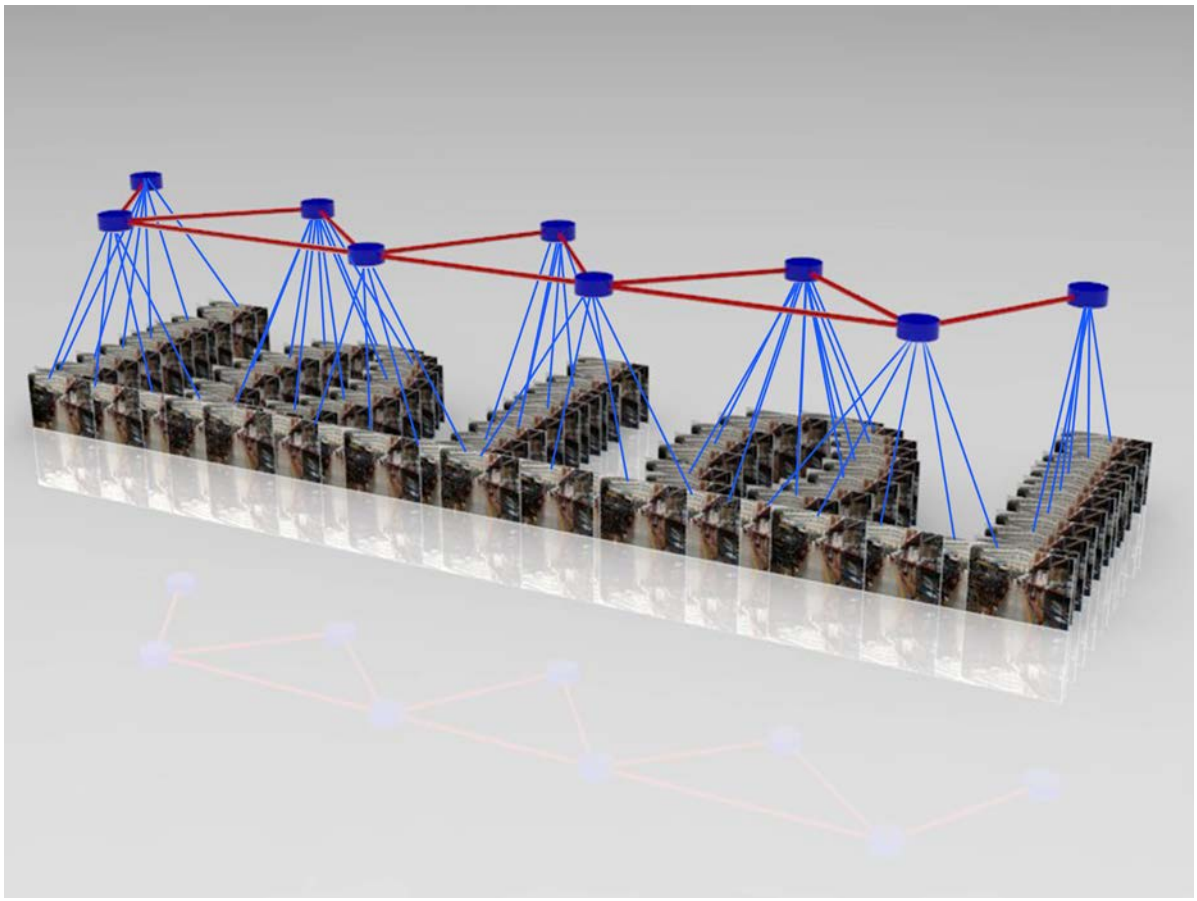


Figure 3.3: *M3.*
En M3 el conjunto de nodos y aristas es disminuido, pues se elijen caminos visuales concatenando aristas de M2. Se crea entonces un mapa de alto nivel, pues se agrupan los nodos por elementos con etiquetas como: "pasillo" y "almacén".

3.2.2. Hipótesis de control

Como se mencionó en la sección anterior, la MV es construida enteramente a partir de imágenes clave; no se incluye información adicional a diferencia del trabajo en [21]. A continuación se expone cómo es la selección de las imágenes clave de la secuencia de entrenamiento S. Para esto, es necesario respetar dos hipótesis propuestas en [7, 8].

- *Hipótesis 1*: Dadas dos imágenes clave I_i e I_{i+1}, existe algún camino factible para el robot que lo lleve de la posición de captura de I_i a la posición de captura de I_{i+1}.

- *Hipótesis 2*: Dos imágenes clave sucesivas I_i e I_{i+1} contienen un conjunto P_i de caracteristicas visuales emparejadas suficientes para calcular las leyes de control. Permitiendo hacer llevar al robot desde la ubicación asociada a I_i hasta la ubicación asociada a I_{i+1}.

Asegurar la navegación del robot es la principal motivación de tales hipótesis, puesto que es necesario que el desplazamiento de I_i a I_{i+1} sea factible, y pueda ser controlado. El control es basado únicamente en las imágenes, por lo que la hipótesis 2 es necesaria. El modelo de homografía, la geometría epipolar y el tensor trifocal son algunas de las herramientas posibles para el control [5], y en éstas y otras es necesario un mínimo de puntos de interés emparejados entre I_i e I_{i+1}.

Para la construcción de la MV, se busca que, de la posición donde el robot capturó una imagen I_i, éste se desplaze la mayor distancia posible a la segunda imagen clave I_{i+1}. Es decir, se pretende que la distancia entre imágenes clave consecutivas sea la máxima posible. Esto asegura que la MV contenga un mínimo de imágenes necesarias para la representación del ambiente (lo que supondría un ahorro en memoria), pero conservando el traslape indispensable entre imágenes necesario para tener emparejamientos, y en consecuencia para el diseño de leyes de control.

Es evidente entonces que para la selección de imágenes clave una comparación entre un par de imágenes debe ser realizada; una comparación entre una imagen clave I_i y una imagen clave potencial I_{Pi}. Es importante determinar la relación que existe entre ese par de imágenes para saber la proporción en la que ambas registran la misma escena. Para esto se aplican algoritmos que determinen las correspondencias entre imágenes a partir de puntos de interés. Tanto los puntos de interés como los algoritmos para la búsqueda de correspondencias son explicados a continuación.

3.3. Detección y emparejamiento de puntos de interés

En visión por computadora, el concepto de puntos de interés, o también llamados *keypoints*, *features points* y características visuales, es frecuentemente usado para resolver problemas de reconocimiento de objetos, reconstrucción 3D, búsqueda de imágenes correspondientes, entre otros. Los puntos de interés tal como vértices y contornos, son ventajosos, pues en lugar de trabajar con cada píxel de la imagen, se seleccionan algunos puntos especiales para llevar a cabo análisis locales. Este enfoque es adecuado, siempre y cuando existan suficientes puntos de interés y puedan ser localizados fácilmente. Además, el uso de puntos de interés es robusto respecto a oclusiones, en comparación con esquemas que usan correlación de regiones (templates) en la imagen.

Actualmente, existen diversos algoritmos que demuestran ser robustos en la localización de puntos de interés. Llamamos a tales algoritmos *detectores* de puntos de interés, e incluyen detectores como: *HARRIS, FAST, STAR, SIFT, SURF, ORB, BRISK, MSER, GFTT, Dense y SimpleBlob.*

Estos algoritmos son útiles para localizar puntos correspondientes entre un par de imágenes. Esto se logra al buscar los puntos de interés correspondientes o conjugados en cada una de las imágenes, pero, ¿cómo saber que un punto de interés encontrado en una imágen es el correspondiente punto de interés encontrado en una segunda imagen? La respuesta está en los *descriptores*, que son algoritmos aplicados a los puntos de interés que buscan caracterizar a los puntos. Es decir, un descriptor asigna un "número de identidad" único para el punto de interés.

Para realizar el emparejamiento de puntos de interés, es necesario comparar los descriptores asociados a los puntos de interés de ambas imágenes. Existen diversos algoritmos para asociar descriptores a puntos de interés, como por ejemplo *SIFT, SURF, BRIEF, BRISK, ORB y FREAK*. Los descriptores y detectores son explicados brevemente en la siguiente subsección del documento.

3.3.1. Detectores y descriptores de características locales

A continuación se presentan los métodos para la selección de puntos de interés y/o el cálculo de los descriptores:

HARRIS [19]. Es un detector de esquinas invariante a la rotación y a cambios de iluminación, pero no a la escala. Se considera que un píxel es una esquina de Harris cuando las variaciones de intensidad en dos direcciones ortogonales entre sí son altas. Para encontrar la variación de intensidad se define una ventana con centro en el píxel candidato (u, v), que se compara con otra ventana que es trasladada en diferentes direcciones, y cuyo centro está en $(u+\delta_u, v+\delta_v)$. La comparación se hace con la SSD

(Sum-of-Squared-Difference) ponderada con una función Gaussiana para reducir el ruido

$$s(u,v,\delta_u,\delta_v) = \sum_{(i,j)} \mathbf{W}[i,j] \left(\underbrace{\mathbf{I}[u+\delta_u+i, v+\delta_v+j]} - \mathbf{I}[u+i, v+j] \right)^2, \quad (3.1)$$

donde $\mathbf{I}$ es la intensidad del píxel y $\mathbf{W}$ es una función Gaussiana. Si el término indicado se aproxima con series de Taylor se obtiene lo siguiente:

$$s(u,v,\delta_u,\delta_v) \approx \sum_{(i,j)} \mathbf{W}[i,j] \left(\delta_u^2 \mathbf{I}_u^2[u+i,v+j] + 2\delta_u\delta_v \mathbf{I}_u[u+i,v+j]\mathbf{I}_v[u+i,v+j] + \delta_v^2 \mathbf{I}_v^2[u+i,v+j] \right),$$

que podemos expresar en forma matricial de la siguiente forma:

$$s(u,v,\delta_u,\delta_v) = (\delta_u\, \delta_v)\mathbf{A} \begin{pmatrix} \delta_u \\ \delta_v \end{pmatrix}, \quad (3.2)$$

donde

$$\mathbf{A} = \begin{pmatrix} \sum \mathbf{W}[i,j]\mathbf{I}_u^2[u+i,v+j] & \sum \mathbf{W}[i,j]\mathbf{I}_u[u+i,v+j]\mathbf{I}_v[u+i,v+j] \\ \sum \mathbf{W}[i,j]\mathbf{I}_u[u+i,v+j]\mathbf{I}_v[u+i,v+j] & \sum \mathbf{W}[i,j]\mathbf{I}_v^2[u+i,v+j] \end{pmatrix}.$$

Un punto (u,v) es detectado como esquina de Harris cuando $s(u,v,\delta_u,\delta_v)$ es alto para cualquier dirección del vector (δ_u,δ_v), en otras palabras, cuando los valores singulares de la matriz $\mathbf{A}$ son altos. Se demuestra que esta condición se cumple cuando

$$C_H = det(\mathbf{A}) - kTraza^2(\mathbf{A}), \quad (3.3)$$

siendo C_H la función de puntuación que determina el peso de un píxel para ser considerado esquina, y k un escalar.

GFTD (*Good Features To Track Detector*) [36]. Se usa para detectar esquinas. Propone una modificación al detector HARRIS al momento de evaluar la función de puntuación. En lugar de $C_H = det(\mathbf{A}) - kTraza^2(\mathbf{A})$ se propone:

$$C_{GFTD} = min(\lambda_1, \lambda_2), \quad (3.4)$$

donde λ_i son los valores propios de la matriz $\mathbf{A}$. Cuando este valor es alto se considera que el punto candidato es un *good feature to track*.

FAST (*Features from Accelerated Segment Test*) [33]. Este algoritmo elige puntos candidatos aplicando una prueba a un segmento. Si P es un píxel candidato, la decisión de aceptarlo como esquina se realiza mediante el examen de un círculo de píxeles centrados en P. Si se encuentra que la intensidad de los píxeles del círculo difiere significativamente de la intensidad del punto central, entonces se declara a P como esquina. Para aplicar la evaluación anterior se define un círculo de Bresenham con radio r y un umbral t (en la práctica un círculo de 16 píxeles tiene mejores resultados según [33]. Ver Fig. 3.4).

La comparación de cada píxel $x \in \{1, \ldots, 16\}$ del círculo, en relación con P, lo clasifica en uno de los tres estados posibles:

$$S_{P\to x} = \begin{cases} d, & I_{P\to x} \leq I_p - t \quad (más\, oscuro) \\ s, & I_p - t < I_{P\to x} < I_p + t \quad (similar) \\ b, & I_p + t \leq I_{P\to x} \quad (más\, brillante) \end{cases} \tag{3.5}$$

Para definir al punto candidato P como esquina, se tiene que comparar entonces a cada punto del círculo con P según los criterios expuestos en 6. El algoritmo reduce el tiempo de ejecución comparando primero los píxeles 1,5,9 y 13 del círculo; si el estado de dos o más de estos píxeles es s se puede rechazar al píxel candidato sin necesidad de comparar los otros 12 píxeles restantes del círculo. FAST es más rápido en comparación con Harris.

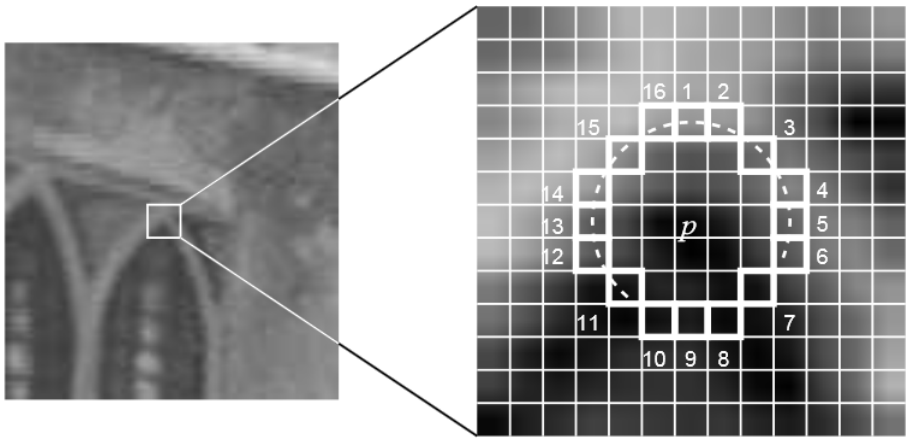


Figura 3.4: *Detector FAST.*
Los cuadros resaltados son los píxeles usados en la detección, el píxel P es el centro de una esquina candidata. Imagen tomada de [33].

SIFT (*Scale-Invariant Feature Detector*) [24]. El algoritmo se realiza en cuatro pasos: 1. Construcción de Pirámides del espacio de escalas: Se representa la imagen en diferentes escalas y tamaños. Se lleva a cabo de manera eficiente mediante el uso de una función de diferencia gaussiana, para identificar los posibles puntos de interés que son invariables a escala y orientación. 2. Localización de puntos clave: Se buscan aquellos puntos que se mantienen en cuanto a cambios de escala. Para ello se debe estudiar cada píxel y realizar una comparación con los pixeles vecinos. Los puntos clave son seleccionados en base a las medidas de su estabilidad. 3. Asignación de orientación: En este paso se asigna a cada punto clave una dirección de acuerdo a las direcciones del gradiente y a la zona que rodea dicho punto. 4. Descriptor de Puntos clave: consiste en calcular un descriptor para la región de la imagen local que sea fácilmente identificable, y sin embargo tan invariable como sea posible a los cambios en la iluminación o el punto de vista 3D.

SURF (*Speeded-Up Robust Features*) [4]. El descriptor SURF emplea una distribución de respuestas Haar-wavelet en el entorno del punto de interés y utiliza imágenes integrales eficientemente. El desempeño es parecido al SIFT, como se ve en los resultados (ver capítulo 4), pero el tiempo computacional es menor. Esto se debe a que el SIFT realiza un escalado de imágenes pero el descriptor SURF hace uso de la matriz Hessiana, más concretamente, del valor del determinante de la matriz, para la localización y la determinación de la escala de los puntos.

STAR, se deriva del detector CenSurE (*Center Surrounded Extrema*) [1]. En comparación con SIFT, reduce el costo computacional al usar imágenes integrales y filtros de caja (*box filters*). Estos últimos sirven para aproximar el LoG (Laplaciano de Gaussianas) para la etapa del análisis en el espacio de escalas, mientras que el SIFT usa una función de diferencia de Gaussianos. Para construir el descriptor se propone el uso de una versión particular del descriptor SURF.

BRIEF (*Binary Robust Independent Elementary Features*) [6]. Este descriptor está constituido por una cadena de bits, a diferencia de SIFT y SURF que representan vectores de flotantes, la cual se genera como resultado de un conjunto de tests binarios, representados por τ, y calculados a partir de una imagen integral. Para construir el descriptor, se define un parche alrededor de un punto de interés; los elementos del descriptor son el resultado de una comparación de intensidades entre dos píxeles del parche donde previamente se ha reducido el ruido con un kernel Gaussiano. Cada elemento del descriptor está en función del test τ, que es definido por:

$$\tau(p) = \begin{cases} 1 & si\, I(p_1) < I(p_2) \\ 0 & de\, lo\, contrario \end{cases}, \tag{3.6}$$

donde p_1 y p_2 son dos puntos elegidos aleatoriamente del parche. El tamaño del descriptor está en función de la cantidad de tests τ. Generalmente, un descriptor BRIEF de 256 bits (32 bytes) es suficiente para lograr resultados similares a los que se obtienen mediante un descriptor SURF estándar de 64 elementos flotantes (256 bytes). El descriptor cumple varias características deseables, que son simples de generar. Pueden compararse en forma eficiente, y su representación es extremadamente concisa, economizando memoria y acelerando el proceso de emparejamiento de puntos de interés.

ORB (*ORiented Binary robust independent elementary features*) [34]. El algoritmo usa FAST en pirámides para detectar puntos estables, encuentra su orientación con el primer momento, y calcula los descriptores con BRIEF. La idea del descriptor ORB es combinar la rapidez del detector de puntos de interes FAST con las propiedades que aporta el descriptor BRIEF.

Lo anterior resume los métodos más utilizados para la detección y emparejamiento de puntos de interés, paso fundamental para la selección de imágenes clave en el esquema propuesto. Para tal fin, es necesario encontrar características visuales con cualquiera de los detectores y comparar sus descriptores. Sin embargo, la eficiencia de tales algoritmos es falible pues está expuesta a errores. Por lo que se considera que los emparejamientos con estos métodos son potencialmente correctos, y es necesario evaluar su autenticidad. Para esta evaluación, la geometría epipolar es de importancia, por lo que es explicada a continuación.

3.4. Geometría epipolar

La geometría epipolar relaciona dos imágenes (I e I') de una misma escena; ésta establece dependencia a partir de los parámetros internos de la cámara y de su pose relativa. La geometría epipolar establece como referencia la intersección del plano de cada imagen con una familia de planos que contienen la línea base, esto es, la línea que une los centros ópticos $\mathbf{C}$ y $\mathbf{C}'$ asociados a las imágenes I e I'.

Empezaremos por explicar las relaciones geométricas que existen entre un punto 3D denotado por X, el cual se observa desde distintas posiciones de cámara como puede apreciarse en la Fig. 3.5. Esta configuración corresponde a dos cámaras (o una misma pero en diferente posición) que observan la misma escena.

El centro óptico de cada cámara, denotados como $\mathbf{C}$ y $\mathbf{C}'$, y el punto en el mundo X, definen un plano epipolar π en el espacio, tal como se muestra en la Fig. 3.6. El punto X es proyectado a los planos imagen de cada cámara, y tales proyecciones son $\mathbf{x}$ para la cámara 1 y $\mathbf{x}'$ para la cámara 2. Estos puntos ($\mathbf{x}$ y $\mathbf{x}'$) son llamados puntos conjugados. También definimos e_1 y e_2 que son llamados epipolos y son las proyecciones de cada centro de cámara en el plano imagen de la otra. Las líneas $\overline{\mathbf{x}e_1}$

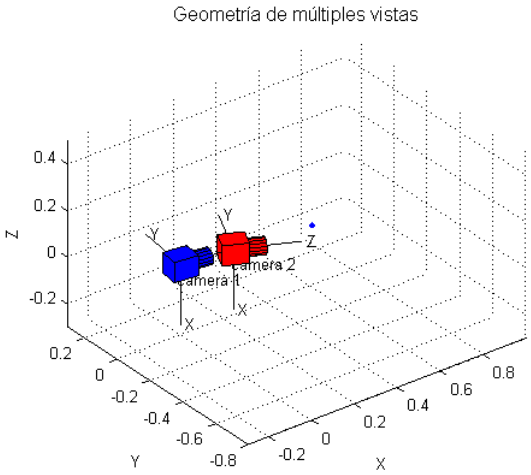


Figura 3.5: *Geometría de dos vistas.*

y $\overline{\mathbf{x}'e_2}$ son llamadas líneas epipolares, que a su vez son la proyección de las líneas $\overline{X\mathbf{C}'}$ y $\overline{X\mathbf{C}}$ en el plano imagen 1 y en el plano imagen 2 respectivamente.

Por definición la proyección del punto X en el plano imagen 2, es decir $\mathbf{x}'$, debe encontrarse en la línea epipolar l', y el punto $\mathbf{x}$ a su vez en la línea epipolar l. Ésta relación geométrica es importante; se llama restricción epipolar y puede ser usada para evaluar emparejamientos potenciales. Tal restricción puede ser expresada en términos de una matriz especial llamada matriz fundamental.

3.4.1. Matriz Fundamental

Cuando se considera un par de imágenes de una misma escena, se tiene que para cada punto $\mathbf{x}$ de la primera imagen, existe su correspondiente línea epipolar l' en la segunda imagen. De tal forma, que cualquier punto $\mathbf{x}'$ en la segunda imagen que corresponde al punto $\mathbf{x}$ debe yacer en la línea epipolar l'. La línea epipolar l' es la proyección, en la segunda imagen, de la línea que parte del punto $\mathbf{x}$ hacia el centro óptico de la primera cámara.

La matriz fundamental $\mathbf{F}$ es la transformación proyectiva de un punto en la primera imagen a su línea epipolar correspondiente en la segunda imagen. En otras palabras, $\mathbf{F}$ es una transformación de puntos a líneas. La matriz $\mathbf{F}$ es una matriz de tamaño 3×3 y tiene un rango igual a 2.

Si se tiene un punto X en el espacio cuya proyección en dos imágenes diferentes es $\mathbf{x}$ en el plano imagen 1 y $\mathbf{x}'$ en el plano imagen 2, la geometría epipolar relaciona los puntos $\mathbf{x}$ y $\mathbf{x}'$ de la siguiente forma:

$$\mathbf{x'^T F x} = 0. \tag{3.7}$$

La expresión en la ecuación anterior es una de las propiedades más importantes de la matriz **F**. Esta propiedad, conocida como restricción epipolar, es una buena alternativa para evaluar si dos puntos son correspondientes entre dos imágenes.

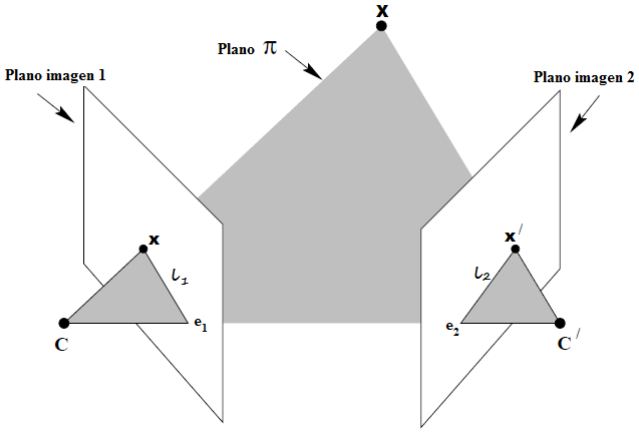


Figura 3.6: *Geometría epipolar.*

Por otro lado, la definición de la linea epipolar, para el segundo plano imagen, está dada por:

$$\mathbf{Fx} \propto l' \tag{3.8}$$

que podemos expresar como

$$\mathbf{F}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \propto l' \tag{3.9}$$

donde **F** es la matriz fundamental, **x** es el punto en el plano imagen 1 en coordenadas homogéneas y l' es la línea epipolar correspondiente en el plano imagen 2.

3.4.2. Cálculo de la Matriz Fundamental

3.4.2.1. Considerando puntos correspondientes

Es posible encontrar la matriz fundamental **F** con el llamado algoritmo de 8 puntos descrito a continuación.

Para pares emparejados $\mathbf{x} \longleftrightarrow \mathbf{x}'$ en dos imágenes. Si tenemos suficiente cantidad de puntos emparejados (al menos 8) $\mathbf{x}_i \longleftrightarrow \mathbf{x}'_i$, entonces es posible calcular $\mathbf{F}$. Cada par de puntos emparejados $\mathbf{x} = (x, y, 1)^T$ y $\mathbf{x}' = (x', y', 1)^T$ da lugar a la siguiente ecuación lineal con los coeficientes de $\mathbf{F}$

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0, \quad (3.10)$$

la cual podemos escribir como producto escalar

$$(x'x, x'y, x', y'x, y'y, y', x, y, 1)\mathbf{f} = 0. \quad (3.11)$$

Tenemos entonces un sistema de ecuaciones lineales en la forma

$$\mathbf{Mf} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = 0. \quad (3.12)$$

Lo anterior es un sistema de ecuaciones homogéneo, por lo que $\mathbf{f}$ sólo puede ser encontrada hasta un factor de escala. Podemos considerar el problema como mínimos cuadrados y encontrar $\mathbf{f}$ con SVD, por ejemplo. La solución para $\mathbf{f}$ será el vector singular correspondiente al más pequeño valor singular de $\mathbf{M}$, es decir, la última columna de $\mathbf{V}$ en la descomposición $SVD(\mathbf{M}) = \mathbf{UDV^T}$. Esta solución para $\mathbf{f}$ es la que minimiza $\| \mathbf{Mf} \|$ sujeta a la condición $\| \mathbf{f} \| = 1$. Para mejorar el condicionamiento del problema y así la estabilidad del resultado, se sugiere aplicar una sencilla transformación a los pares de puntos antes de formar el sistema de ecuaciones. Se aplica una transformación que haga que el centroide de los puntos sea el origen y que la distancia promedio del origen a cada punto sea igual a $\sqrt{2}$.

3.4.2.2. Considerando las posiciones de cámaras

Es posible estimar una matriz $\mathbf{F}$ con el conocimiento a priori de la posición de las cámaras. La deducción algebraíca siguiente está basada en el trabajo de Xu y Zhang [42]. El conjunto de puntos en el espacio que se proyecta como $\mathbf{x}$ en el plano imagen 1 de la Fig. 3.6 está definido como

$$X(\lambda) = \mathbf{P}^+\mathbf{x} + \lambda\mathbf{C}. \quad (3.13)$$

Donde $\mathbf{P}^+$ es la pseudo inversa de la matriz de proyección $\mathbf{P}$, de forma que $\mathbf{P}^+\mathbf{P} = \mathbf{I}$, y $\mathbf{C}$ es el centro de cámara definido como $\mathbf{PC} = 0$. El rayo es

parametrizado por el escalar λ. Dos puntos en particular pertenecientes a este rayo son $\mathbf{P}^{+}\mathbf{x}$ (en $\lambda = 0$) y el punto $\mathbf{C}$ (en $\lambda = 0$). Estos dos puntos son proyectados en el plano imagen dos como sigue: $\mathbf{P}'\mathbf{P}^{+}\mathbf{x}$ y $\mathbf{P}'\mathbf{C}$. La línea epipolar contendrá entonces a estos dos puntos de forma que $l' = (\mathbf{P}'\mathbf{C}) \times (\mathbf{P}'\mathbf{P}^{+}\mathbf{x})$.

Sabemos de la ecuación 3.8 que $l' = \mathbf{F}\mathbf{x}$, entonces

$$l' = (\mathbf{P}'\mathbf{C}) \times (\mathbf{P}'\mathbf{P}^{+})\mathbf{x} \tag{3.14}$$

$$\mathbf{F} = [\mathbf{P}'\mathbf{C}]_{\times}\mathbf{P}'\mathbf{P}^{+} \tag{3.15}$$

3.4.3. Matriz Esencial

La restricción epipolar puede ser expresada también en términos de coordenadas normalizadas al plano imagen. El resultado es una matriz $\mathbf{E}$ llamada matriz esencial. De la ecuación 3.15, deducida en la subsección anterior, podemos encontrar una expresión para $\mathbf{E}$. Supongamos dos cámaras, una de ellas ubicada en el origen,

$$\mathbf{P} = \mathbf{K}[\mathbf{I}\,|\,\mathbf{0}] \quad y \quad \mathbf{P}' = \mathbf{K}'[\mathbf{R}\,|\,\mathbf{t}],$$

donde $\mathbf{R}$ es la matriz de rotación entre los sistemas de referencia asociados a cada centro óptico de las cámaras, y $\mathbf{t}$ el vector de traslación asociado a la posición de la segunda cámara. Se puede mostrar que $\mathbf{E}$ es expresada en términos de $\mathbf{R}$ y $\mathbf{t}$, sustituyendo $\mathbf{P}$ y $\mathbf{P}'$ en 3.15 y considerando como la identidad a $\mathbf{K}$ y $\mathbf{K}'$. De forma que la expresión para $\mathbf{E}$ es:

$$\mathbf{E} = [\mathbf{t}]_{\times}\mathbf{R}. \tag{3.16}$$

La matriz esencial puede ser calculada de igual forma en función de la matriz fundamental cuando las matrices de calibración son conocidas,

$$\mathbf{E} = \mathbf{K}'^{\mathbf{T}}\mathbf{F}\mathbf{K}, \tag{3.17}$$

donde $\mathbf{K}$ es la matriz de calibración y $\mathbf{F}$ la matriz fundamental.

De esta sección se deduce que la restricción epipolar es útil para evaluar la autenticidad de emparejamientos de puntos de interés. Evaluar tal autenticidad con la geometría epipolar implica encontrar una matriz fundamental $\mathbf{F}$ tal que para cada par de puntos emparejados $\mathbf{x} = (x, y, 1)^T$ y $\mathbf{x'} = (x', y', 1)^T$ se cumpla $\mathbf{x'^T F x} = 0$.

A partir del conjunto de emparejamientos potenciales (correctos e incorrectos), encontrados mediante la comparación de descriptores de puntos de interés, sólo elegimos como buenos emparejamientos a aquellos que cumplan con la restricción epipolar. Pero para comprobar tal restricción debemos conocer la matriz $\mathbf{F}$, y necesitamos tener un conjunto con únicamente emparejamientos adecuados para estimarlas (suponiendo que no contamos con las posiciones de las cámaras).

El problema es que a partir de puntos correspondientes, donde existen emparejamientos correctos e incorrectos, necesitamos encontrar una matriz $\mathbf{F}$. Para resolver este problema y eliminar todos los emparejamientos incorrectos, podemos valernos del método RANSAC explicado a continuación.

3.5. RANSAC

RANSAC (Random Sample Consensus) es un método para estimar un modelo matemático a partir de un conjunto de datos ruidosos con valores atípicos que no se ajustan al modelo. La idea es elegir en varias ocasiones del conjunto de datos una muestra de valores para estimar el modelo matemático hasta encontrar la mejor estimación.

En nuestro caso, el modelo matématico a estimar es la matriz fundamental, por lo que necesitamos una muestra de 8 valores para estimarla con el algoritmo descrito arriba. Una vez estimada la matriz fundamental, se procede a comprobar la restricción epipolar derivada de esta matriz para cada par de puntos emparejados. Cada uno de los emparejamientos que cumplen tal restricción *votan* por el modelo estimado con la muestra de valores elegidos al azar. El proceso anterior es repetido varias veces, y se elige a la matriz fundamental con más *votos* como la más probable de ser la correcta.

El problema de encontrar una matriz $\mathbf{F}$ a partir de un conjunto con valores atípicos o *outliers* puede tratarse de igual forma como un problema de optimización. La computación evolutiva es una alternativa al RANSAC para la solución a problemas de este tipo. En la siguiente sección se explican los algoritmos genéticos como herramientas para la solución a problemas de optimización. Y en el capítulo 4 se aborda el uso de algoritmos genéticos para la estimación de la geometría epipolar.

3.6. Optimización

Optimizar es buscar la mejor solución a un problema bajo ciertas restricciones. Los problemas de optimización se plantean de tal forma que se minimice el esfuerzo requerido, o bien se maximicen los beneficios deseados. En problemas científicos y/o de ingeniería, el esfuerzo requerido o los beneficios deseados pueden ser expresados como una función o funciones que actúan sobre ciertas variables de decisión. Por lo tanto, la optimización puede ser definida como el proceso de encontrar las soluciones que proporcionen el valor máximo o mínimo de una función [38].

Algunos problemas en ingeniería, ciencia y producción tienen varias soluciones y algunos tienen infinitas soluciones. De forma que estos problemas se plantean como una función objetivo a ser minimizada o maximizada (optimización) según sea el caso. El objetivo es encontrar o identificar la mejor solución posible, entre todas las soluciones potenciales, en términos de algún o algunos criterios de efectividad o desempeño.

Las funciones objetivo modelan un problema de optimización, ya que sus valores más extremos representan la meta de la optimización. Cuando su valor mínimo es buscado, la función objetivo es frecuentemente referida como función de costo. En el caso especial donde el valor mínimo a buscar es cero, la función objetivo es conocida como función de error. Una de las herramientas usadas para la solución a problemas de optimización es la computación evolutiva descrita en la siguiente sección.

3.6.1. Computación Evolutiva, Algoritmos Genéticos (AG)

La Computación Evolutiva (CE) engloba diversos algoritmos que están inspirados biológicamente con base en la teoría Neo-Darwiniana de la evolución natural. Ésta es vista como un proceso de optimización, en el cual los individuos de una población gradualmente se adaptan a su ambiente. En los algoritmos pertenecientes a la CE, un individuo es una solución potencial del problema y el ambiente al que pertenece es la función o funciones objetivo y sus restricciones. Este ambiente determinará la capacidad de supervivencia del individuo.

Un algoritmo genético es una clase de algoritmo perteneciente a la CE. Es un algoritmo de búsqueda basado en la mecánica de la selección natural. Son llamados así porque se inspiran en la evolución biológica. Estos algoritmos hacen evolucionar una población de individuos sometiéndola a acciones aleatorias semejantes a las que actúan en la evolución biológica (mutaciones y recombinaciones genéticas), así como también a una selección de acuerdo con alguna función de aptitud, que decide cuáles son los individuos más adaptados que sobreviven, y cuáles los menos aptos, que son descartados. El algoritmo evolutivo simple usa tres operadores básicos según [12]:

1. Reproducción

2. Cruza
3. Mutación

En un algoritmo genético, los individuos son codificados como cadenas binarias, que representan el cromosoma o genotipo del individuo. Por otra parte, el valor real al que codifica el genotipo es llamado fenotipo. A continuación se explican cada uno de los ya mencionados operadores básicos.

La *reproducción* es un proceso de selección de cromosomas para el posterior apareamiento e intercambio de genes, con base en el valor de su función de aptitud (llamada así en biología). En otras palabras, se copian o seleccionan con mayor probabilidad aquellos individuos cuya función de aptitud es mayor y así tienen mayor probabilidad de contribuir con descendencia a la próxima generación. La función de aptitud determina entonces la supervivencia o muerte de algún individuo. Uno de los métodos más usado para la selección de individuos es la llamada rueda de ruleta. Este método asigna un porcentaje de la ruleta, es decir una probabilidad, a cada individuo según su aptitud; donde el 100 % de la ruleta es la suma de las aptitudes. En la Fig. 3.7, se observa la ruleta que se construye para una población de cuatro individuos con aptitudes: 25, 784, 441 y 961.

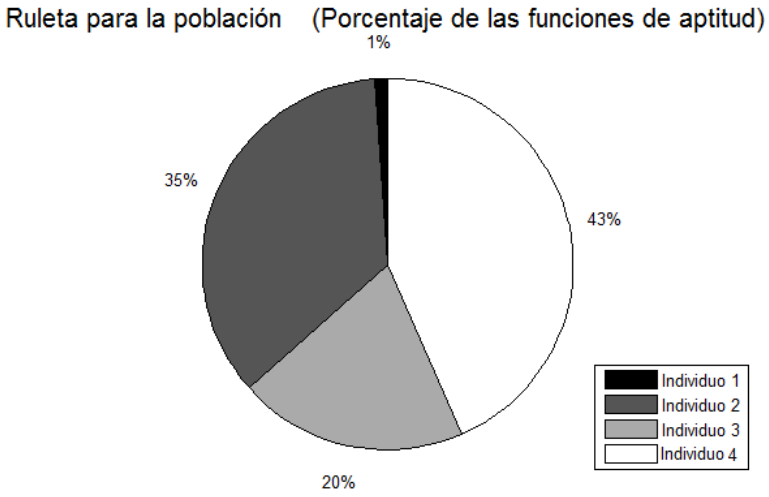


Figura 3.7: *Ejemplo de Ruleta.*

La *cruza* es un operador que lleva a cabo el intercambio de información genética de dos individuos de la población, a los que llamaremos padres, en el cual se combinan sus genes, que son los bits de las cadenas binarias, para generar descendencia o hijos.

La *mutación* modifica bits de la cadena binaria con una probabilidad P_m con el fin de generar nuevos individuos o hijos que formarán parte de la nueva población.

Este operador permite la variabilidad en la población, con la finalidad de no quedar estancado en un máximo o mínimo local. Eso contribuye a la exploración de todo el espacio de búsqueda evitando así la convergencia prematura.

Los distintos algoritmos genéticos que se pueden formular responden a un esquema básico común, y comparten una serie de propiedades:

- Procesan simultáneamente, no una solución al problema, sino todo un conjunto de ellas. Estos algoritmos trabajan con una codificación de soluciones potenciales al problema, que se denominan individuos o cromosomas. El conjunto de todos ellos forman la población con la que trabaja el algoritmo.
- La composicion de la población se va modificando a lo largo de las iteraciones del algoritmo que se denominan generaciones. De generación en generación, además de variar el número de copias de un mismo individuo en la población, también pueden aparecer nuevos individuos generados mediante operaciones de transformación sobre individuos de la población anterior. Dichas operaciones se conocen como operadores genéticos, que ya han sido descritos.
- Cada generación incluye un proceso de selección, que da mayor probabilidad de permanecer en la población y participar en las operaciones de reproducción a los mejores individuos. Los mejores individuos son aquellos que dan lugar a los mejores valores de la función de adaptación del algoritmo. Es fundamental para el funcionamiento de un algoritmo genético que este proceso de selección tenga una componente aleatoria, de forma que individuos con bajo valor de adaptación también tengan oportunidades de sobrevivir, aunque la probabilidad asociada a éstos sea menor. Es esta componente aleatoria la que dota a los algoritmos genéticos de capacidad para escapar de óptimos locales y de explorar distintas zonas del espacio de búsqueda.

3.6.2. Ejemplo de un AG simple

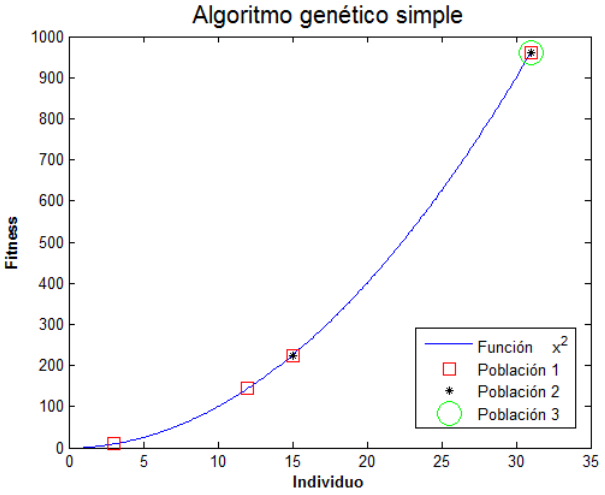
Consideremos el problema de maximizar la función $f(x) = x^2$, donde x varia entre 0 y 31. Se codifican los individuos con una cadena de 5 elementos siendo $00000 = 0$ y $11111 = 31$. Para empezar se elige aleatoriamente una población de 4 indiviudos.

El algoritmo elige de forma aleatoria una población inicial con individuos: 15, 31, 3 y 12, cuya codificación, valor de aptitud y porcentaje en la ruleta se muestran en la Tabla 1. En la primera iteración del algoritmo se encuentra una nueva población con individuos: 15, 31, 31 y 31. Se observa que la aptitud máxima se mantuvo y que los mejores individuos fueron seleccionados para la cruza.

Individuo	Cadena / Cromosoma	Aptitud	% en ruleta
1	01111	225	16.8
2	11111	961	71.76
3	00011	9	0.67
4	01100	144	10.77

Tabla 1. Cromosomas y valores de aptitud del ejemplo.

En la Fig. 3.8 se presenta una gráfica con la evolución de la población, en cada iteración de los algoritmos de reproducción y cruza la función de aptitud mejoró en términos generales. Sin embargo al tratarse de un algoritmo basado en la probabilidad, la población no siempre converge al máximo en la tercera iteración.



Población 1	Población 2	Población 3
15	15	31
31	31	31
3	31	31
12	31	31

Figura 3.8: *Algoritmo genético. Ejemplo.*
Evolución de la población del ejemplo. En la tercera iteración la población converge y se ubica en el máximo de la función. Se puede observar que la población mejora en cada iteración.

4 Metodología

Este capítulo está dedicado al análisis de los procedimientos que se llevaron a cabo para la construcción de la MV. Como se explicó en la introducción la memoria se construye en tres etapas diferentes, como se exhibe en la Fig. 1.1. A continuación se explica a detalle cada una de las etapas.

4.1. Etapa de aprendizaje

En esta etapa el robot recorre el ambiente a navegar al mismo tiempo que captura una secuencia de imágenes de entrenamiento $S = \{F_i | i \in \{1, 2, \ldots, m\}\}$, ésto con asistencia humana. Se resalta el hecho de que tal adquisición de imágenes puede realizarse sin el robot, pues es posible que un humano recorra el ambiente con una cámara en mano.

La calidad de la MV para describir el ambiente depende mucho de esta etapa. Pues la representación topológica dependerá mucho del recorrido que el humano "enseñe" al robot. Para los experimentos se capturaron diferentes secuencias: una simulación en un ambiente sintético; una con la camara abordo del robot humanoide NAO; también se capturaron imágenes con la cámara de un teléfono inteligente en el almacén de una fabrica de calzado de la región. La resolución de las imágenes para las secuencias de simulación y del robot humanoide Nao fue de 640×480. La resolución para la secuencia capturada con la cámara del teléfono inteligente fue de 1920×1080.

4.2. Selección de imágenes clave

En la Fig. 4.1 podemos ver la representación de un camino visual de M3 (ver capítulo 2), donde las imágenes clave forman el conjunto I_i del camino visual Γ_i. Para la selección de estas imágenes se debe buscar la mayor distancia posible entre I_i e I_{i+1} pero conservando un mínino de traslape entre ellas.

Para resolver este problema necesitamos entonces saber qué tanta correspondencia existe entre I_i e I_{i+1}, y también necesitamos saber qué tan bien condicionada está la geometría epipolar entre estas imágenes. Cada vez que se evalúa esta información para las imágenes de la secuencia S, se puede rechazar o aceptar una imagen como imagen clave en el conjunto I_i.

Explicamos a continuación cómo se encuentra la correspondencia entre imágenes y cómo se evalúa la geometría epipolar. Después se muestra cómo se hace uso de esta información para encontrar imágenes clave consecutivas con la mayor distancia entre ellas y el mínimo de correspondencia razonable para el uso posterior de un algoritmo de control como en [5].

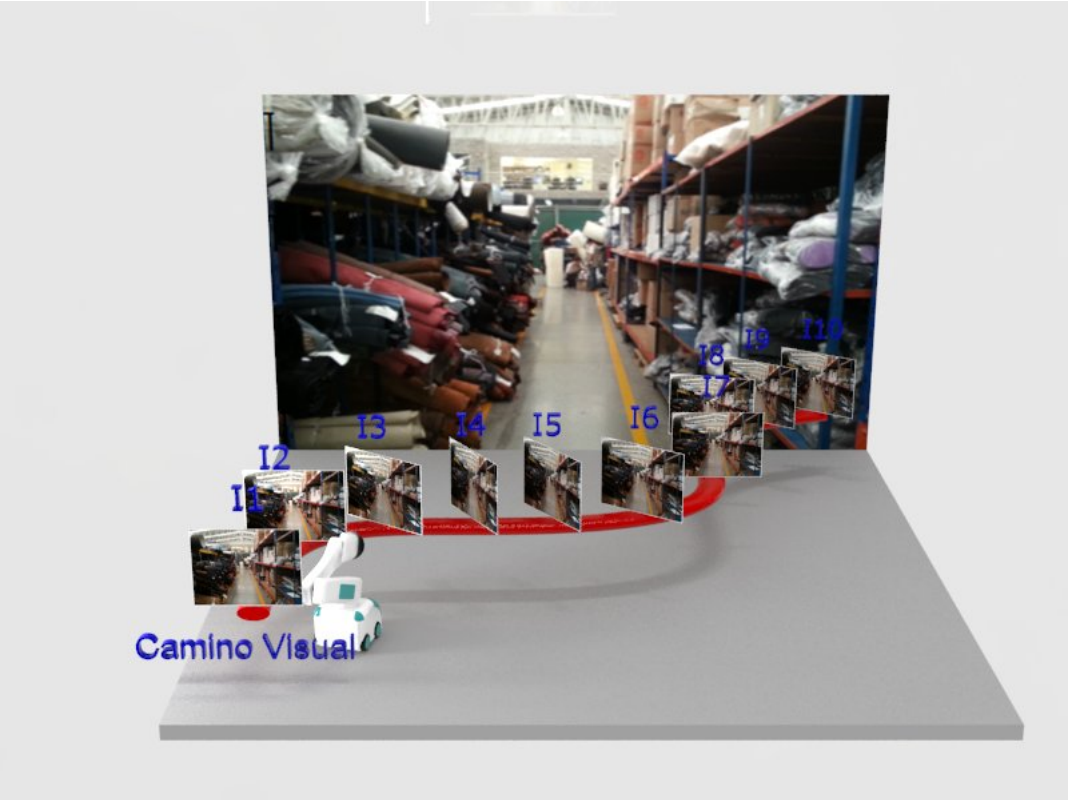


Figura 4.1: *Ejemplo de un camino visual de M3.*
Las imágenes fueron capturadas en el almacén de una empresa de calzado de León Gto. Ilustran el camino que seguiría un robot móvil para el acomodo de materia prima.

4.2.1. Búsqueda y emparejamiento de puntos de interés

Como se ha mencionado, el núcleo del algoritmo para la construcción de la MV es un buen emparejamiento de suficientes puntos de interés, que son puntos correspondientes. El mayor problema que se encuentra son las imágenes con efecto *blur* debido al movimiento brusco de la cámara abordo, provocado por la caminata de un robot tipo humanoide. En [30, 31] se propone una modificación de SIFT para un mejor emparejamiento cuando las imágenes son capturadas por un robot humanoide. En tales trabajos, se propone encontrar la dirección del *blur* en una imagen previamente seccionada y modificar la manera de calcular los puntos SIFT.

Este problema en particular, del manejo del *blur*, se considera trabajo futuro y no se aborda en esta tesis. A continuación, se comenta la solución propuesta para la búsqueda de correspondencias.

La detección y emparejamiento de puntos de interés se hace con una clase de C++ que se ha propuesto, denominada *geneticMatcher*, la cual encuentra buenos emparejamientos de los puntos de interés en las imágenes validando cada emparejamiento con la restricción epipolar evaluada con ayuda de un algoritmo genético. Esto reduce errores y aumenta la confianza de que un emparejamiento sea adecuado. Una técnica alterna es la homografía, sin embargo ésta exige que los puntos de interés se encuentren en un plano o que el cambio de movimiento de la cámara sea sólo de rotación. De forma que es más conveniente validar los emparejamientos con la restricción epipolar para nuestro caso.

Los pasos para la detección de emparejamientos adecuados son los siguientes:

1. Se detectan los puntos de interés y se caracterizan por sus descriptores en las dos imágenes consecutivas de la secuencia.

2. Se comparan y emparejan los descriptores entre ambas imágenes, encontrando dos emparejamientos candidatos por punto de interés.

3. Se remueven los emparejamientos para los cuales la razón NN (Nearest Neighbour) es mayor a un umbral. NN es la división de la distancia euclidiana entre el punto a emparejar a y el mejor emparejamiento candidato b_1 por la distancia entre el punto a emparejar a y el segundo mejor emparejamiento b_2.

4. Se remueven los emparejamientos no simétricos, es decir, aquellos para los cuales el mejor emparejamiento de $a \longleftrightarrow b$ no corresponde en la dirección contraria para $b \longleftrightarrow a$.

5. Se validan los emparejamientos con la matriz Esencial (restricción epipolar) encontrada con un algoritmo genético.

Para la implementación de los pasos anteriores se hace uso de algunas funciones de la librería OpenCV de C++ [29]. El primer paso detecta puntos de interés y encuentra sus descriptores. Después, en el segundo paso se usa el método de la clase de OpenCV *cv::BruteForceMatcher::knmatch* con k=2, que encuentra los 2 mejores emparejamientos con base a la distancia entre sus descriptores para cada punto en ambas imágenes. La búsqueda de estos emparejamientos se hace en dos direcciones, de la imagen 1 a la imagen 2 y viceversa, con lo que para cada punto

en la primera imagen se encuentran los 2 mejores emparejamientos en la segunda, y para cada punto de la segunda imagen se encuentran los 2 mejores emparejamientos en la primera. De forma que para cada punto de interés se tienen 2 emparejamientos candidatos.

Figura 4.2: *Ejemplo de emparejamientos resultando de la clase geneticMatcher. En la imagen, se aíslan algunos emparejamientos para un par de imágenes. Cada par de puntos emparejados se resalta con un color diferente. Se hace notar que una buena cantidad de puntos no fueron emparejados, sin embargo los que sí lo fueron son buenos emparejamientos.*

Si esta distancia es pequeña para el primer mejor emparejamiento y relativamente grande para el segundo mejor emparejamiento podemos elegir con confianza al primer emparejamiento como adecuado, pues es característico. Por otra parte cuando las distancias del primer mejor emparejamiento y del segundo son muy cercanas, elegir uno u otro puede llevarnos a errores, de forma que rechazamos ambos emparejamientos. Esto se hace verificando la razón NN que es la división de la distancia del primer mejor emparejamiento por la del segundo mejor emparejamiento.

Los dos mejores emparejamientos se basan en la distancia entre sus descriptores. Entonces en el paso 3 se rechazan los emparejamientos para los cuales la razon NN entre las distancias de ambos emparejamientos es mayor que un umbral. Esto sirve para descartar los emparejamientos candidatos que tienen "competidores" de nivel similar.

En el paso 2 se encuentran emparejamientos en ambas direcciones y el paso 3 filtra con las distancias de igual forma en ambas direcciones, de forma que se tienen 2 conjuntos de buenos emparejamientos. Después en el paso 4 se descartan los emparejamientos que no se encuentran en ambos conjuntos.

La última prueba para calificar a un emparejamiento como bueno se hace en el paso 5, donde se rechazan los emparejamientos que no cumplen con la restricción epipolar. Para eso se usa la función *geneticAlgorithm* de la clase desarrollada *geneticMatcher* que describiremos a continuación. En la figura Fig. 4.2 se muestra

un ejemplo de los resultados de emparejamientos encontrados con la clase *geneticMatcher*.

En la sección siguiente se describe el algoritmo genético usado en el paso 5 para encontrar la matriz esencial que permite validar los emparejamientos.

4.2.2. Descripción del AG usado

Se hace uso de un AG para evaluar la restricción epipolar en cada uno de los emparejamientos candidatos y así rechazar aquellos que no cumplan con esta restricción. De forma que el algoritmo ayuda en la búsqueda de buenos emparejamientos.

La matriz esencial es la matriz fundamental normalizada considerando la matriz de calibración $\mathbf{K}$ como la identidad. Analíticamente podemos encontrar la matriz esencial con la siguiente ecuación

$$\mathbf{E} = [\mathbf{t}]_{\times}\mathbf{R}, \tag{4.1}$$

siendo $\mathbf{R}$ la matriz de rotación y $\mathbf{t}$ el vector de traslación. El teorema de rotación de Euler establece que cualquier rotación puede ser representada por no más de tres rotaciones con respecto a los ejes de coordenadas, así que podemos representar $\mathbf{R}$ como sigue,

$$\mathbf{R} = \mathbf{R}_z(\phi)\mathbf{R}_y(\theta)\mathbf{R}_z(\psi), \tag{4.2}$$

lo que reduce el número de variables a usar en el algoritmo genético. Por otro lado podemos considerar al vector de traslación como sigue,

$$\mathbf{t} = [t_x, t_y, 1]^T. \tag{4.3}$$

Se sabe que la matriz esencial codifica la posición de la cámara hasta una escala, por lo que considerar $t_z = 1$ es perfectamente válido. Así, teniendo en cuenta lo anterior cada individuo es formado como sigue:

$$Ind = [\phi, \theta, \psi, t_x, t_y], \tag{4.4}$$

donde

- $\phi \in [0, 2\pi]$
- $\theta \in [0, 2\pi]$
- $\psi \in [0, 2\pi]$
- $t_x \in [-5, 5]$
- $t_y \in [-5, 5]$

4.2.2.1. Representación de los individuos

Como ya se ha comentado, en un AG, los individuos pueden ser representados con cadenas binarias, por ejemplo. Es importante que cada posición de la cadena codifique una variable para el problema, ya que de esta forma se favorece que los genes que dan alta calidad de aptitud a un individuo sigan dando lugar a características de calidad en un nuevo individuo obtenido por alguna operación genética a partir del primero. También es importante buscar una codificación a partir de la cual se pueda llegar de forma eficiente al fenotipo, ya que la decodificación será una operación muy frecuente a lo largo de la evolución. Tal decodificación es explicada a continuación.

Cuando trabajamos con una función sobre números reales, asignando un intervalo con valores mínimo y máximo, necesitamos discretizar dicho intervalo para hacer corresponder posiciones del intervalo real considerado con números enteros representados por nuestra cadena binaria. Así mismo, un parámetro de precisión es usado para indicar el número de posiciones decimales en las que nos interesa que el resultado de esta correspondencia sea exacto. Este parámetro condiciona la longitud de la cadena binaria.

Con una cadena binaria de longitud l podemos representar 2^l números enteros, así que podemos hacer el tamaño de la discretización como sigue:

$$tamaño\_discretización = \frac{x_{max} - x_{min}}{2^l - 1} < precisión. \tag{4.5}$$

Para obtener la longitud l de la cadena binaria que necesitamos para que nuestro algoritmo pueda alcanzar la precisión requerida, se aplica la siguiente ecuación:

$$l = \left\lceil log_2 \left(1 - \frac{x_{max} - x_{min}}{precisión} \right) \right\rceil. \tag{4.6}$$

Una vez que la longitud de la cadena se ha calculado, el genotipo (cadena binaria) codifica al fenotipo (valor real). Para decodificar este valor, se considera que cada uno de los 2^l números que puede representar la cadena binaria representa un punto en el intervalo $[x_{min}, x_{max}]$ donde buscamos optimizar la función. Entonces el valor codificado puede encontrarse según la siguiente ecuación:

$$valor = x_{min} + cadena_{10} \left(\frac{x_{max} - x_{min}}{2^l - 1} \right), \tag{4.7}$$

donde $cadena_{10}$ es el valor decimal que corresponde a la cadena binaria considerada. Esto entonces lo aplicamos para cada uno de nuestros individuos.

4.2.2.2. Población inicial

Es imprescindible para el buen funcionamiento del AG dotar a la población de suficiente variedad para poder explorar todas las zonas del espacio de búsqueda. Para esto se crea una población inicial de tamaño n (individuos), los cuales son generados de la siguiente forma:

- Se genera un numero aleatorio $a \in [min, max]$ donde min y max son el mínimo y el máximo valor aleatorio generado por la computadora respectivamente.
- Si $a < \frac{max}{2}$ se hace el primer gen del genotipo 1, de lo contrario se hace 0.
- Se asignan l genes para cada individuo con el punto anterior, donde l es calculada con la ecuación 4.6.
- Se generan n individuos repitiendo los 3 puntos anteriores.

Este método asigna probabilidades iguales a los ceros y unos usados para generar el genotipo.

Como alternativa se consideró iniciar la población con individuos estimados con el algoritmo de 8 puntos:

- Se eligen al azar 8 pares de puntos para estimar la matriz fundamental.
- Se hace $\mathbf{E} = \mathbf{K^T F K}$.
- Se encuentra la rotación y traslación a partir de $\mathbf{E}$.
- Se codifica la rotación y traslación en una cadena binaria.
- Se generan n individuos repitiendo los puntos anteriores.

Para las pruebas se consideró $n = 800$.

4.2.2.3. Función de aptitud de los individuos

La función de adaptación evalúa la cantidad de inliers para la matriz esencial que modela cada individuo. De cada uno de los individuos se calcula una matriz esencial candidata con 4.1. Se encuentran las líneas epipolares y se define un inlier cuando la distancia del punto a considerar es menor que un umbral U_{ga} de su línea epipolar correspondiente. Si $\mathbf{P^{Nb}} = \mathbf{K^{-1}P^b}$ son las coordenadas normalizadas en el segundo plano imagen, la línea epipolar correspondiente se encuentra como sigue:

$$l_i = (\mathbf{K}^{-1})^T \mathbf{E P^{Nb}}. \tag{4.8}$$

Así, consideramos como un inlier a un punto cuya distancia a esta línea es menor que el umbral.

4.2.2.4. Proceso de selección

La población se somete a un proceso de selección que favorece la cantidad de copias de los individuos más adaptados pero que no converga prematuramente. Para tal efecto se ha elegido la selección de Boltzmann, que controla la presión de selección usando una función de "temperatura", y que es descrita como sigue.

La aptitud de cada individuo se modifica de la siguiente forma.

$$f_{modificada} = e^{f/T} \tag{4.9}$$

En la literatura [13], se aclara que la temperatura T puede hacerse disminuir de forma lineal, escalonada o exponencial; siendo la última la que mejores resultados informa, por ello se eligió que la temperatura T en el algoritmo disminuyera exponencialmente. Teniendo la aptitud modificada de cada individuo se procede a la selección, donde la probabilidad de selección p_i de un individuo i es proporcional a su aptitud relativa:

$$p_i = \frac{f(i)_{modificada}}{\overline{f}_{modificada}} \tag{4.10}$$

siendo $\overline{f}_{modificada}$ la aptitud media de la población.

Luego, necesitamos generar un número aleatorio de acuerdo con la distribución de probabilidad dada por los p_i, y así generando un número aleatorio a podemos seguir el siguiente procedimiento:

- Se definen "puntuaciones" acumuladas de la siguiente forma:

  $$P_{acum_0} = 0$$

  $$P_{acum_i} = p_1 + \ldots + p_i \qquad (i = 1, \ldots, n) \tag{4.11}$$

- Se genera el número aleatorio $a \in [0, 1]$.
- Se selecciona al individuo i que cumpla con:

  $$P_{acum_{i-1}} < a < P_{acum_i}$$

  Este proceso se repite para cada individuo que se desee seleccionar.

4.2.2.5. Operador de Cruce

Se propone el cruce de dos puntos, que consiste en seleccionar al azar dos posiciones en la cadena de ambos padres e intercambiar las partes de los padres divididas por dichas posiciones. Para realizar este operador genético, se hizo lo siguiente:

- Se compara una realización de una distribución de probabilidad uniforme $p_a \in [0, 1]$ aleatoria con la tasa de cruce p_c. Si $p_a > p_c$ no se aplica el operador, de lo contrario se continúa con los siguietes puntos.
- Se generan dos números aleatorios $k_1 \in [1, l]$ y $k_2 \in [1, l]$ restringidos a la condición siguiente, $k1 < k2$. Estos dos números representan los puntos de cruce.
- De una pareja de padres $padre_1$ y $padre_2$, se genera una pareja de hijos $hijo_1$ y $hijo_2$.
- Se asigna el genotipo del $padre_1$ al $hijo_1$ desde la posición 1 hasta la posición k_1.
- Se asigna el genotipo del $padre_2$ al $hijo_2$ desde la posición 1 hasta la posición k_1.
- Se asigna el genotipo del $padre_1$ al $hijo_2$ desde la posición $k_1 + 1$ hasta la posición k_2.
- Se asigna el genotipo del $padre_2$ al $hijo_1$ desde la posición $k_1 + 1$ hasta la posición k_2.
- Se asigna el genotipo del $padre_1$ al $hijo_1$ desde la posición $k_2 + 1$ hasta la posición l.
- Se asigna el genotipo del $padre_2$ al $hijo_2$ desde la posición $k_2 + 1$ hasta la posición l.

Como ejemplo, consideremos dos cadenas que representan al $padre_1$ y al $padre_2$. Consideremos también $k_1 = 3$ y $k_2 = 9$,

$$padre_1 = 1\,1\,0\,|_{k_1}\,1\,0\,0\,1\,1\,0\,|_{k_2}\,1$$

$$padre_2 = 0\,0\,1\,|_{k_1}\,1\,1\,1\,1\,0\,0\,|_{k_2}\,0.$$

De acuerdo a los pasos ya mencionados, la nueva generación queda como sigue:

$$hijo_1 = 1\,1\,0\,|_{k_1}\,1\,1\,1\,1\,0\,0\,|_{k_2}\,1$$

$$hijo_2 = 0\,0\,1\,|_{k_1}\,1\,0\,0\,1\,1\,0\,|_{k_2}\,0.$$

Lo anterior se realiza hasta llegar a un número n de hijos correspondiendo al tamaño inicial de la población. La tasa de cruce p_c es otro de los parametros esenciales del AG y se consideró igual a 1.

4.2.2.6. Operador de Mutación

Se propone la mutación aleatoria bit a bit, con probabilidad p_m, que consiste en cambiar el valor de una de las posiciones de la cadena: si es cero pasa a uno y si es uno pasa a cero. Tal como el operador de cruce, para que el operador de mutación sea aplicado o no es necesario proveer una tasa de mutación p_m. La implementación de este algoritmo se lleva a cabo como sigue:

- Para la posición k_i donde $i \in [1, l]$; se compara una realización de una distribución de probabilidad uniforme $P_a \in [0, 1]$ aleatoria con la tasa de mutación P_m. Si $P_a > P_m$ no se aplica el operador, de lo contrario se continúa con el siguiente punto.
- Se cambia el valor en la posición k_i. Si es cero pasa a uno y si es uno pasa a cero.

En muchos casos la mutación produce individuos con peor adaptación que los individuos originales, ya que la mutación puede romper las posibles correlaciones entre génes que se hayan formado con la evolución de la población. La razón de existir del operador se justifica, ya que la diversidad de individuos que proporciona el operador de mutación, es fundamental para evitar estancamientos del método en mínimos o máximos locales.

4.2.2.7. Proceso de reemplazo

El método evolutivo implementado es un algoritmo genético generacional, es decir, la población se renueva por completo de una generación a otra, donde los hijos sustituyen a los padres. Sin embargo, se considera un conjunto de élite que contiene a los indivuos con las mejores aptitudes. Este conjunto es de tamaño m igual al 1 % de la población, y sobrevive y actualiza en cada iteración.

4.2.2.8. Resumen de las características del AG

Parámetros:

1. Valor del error permitido para la discretización del intervalo $err = 0.001$.
2. Tamaño de la población $n = 800$.
3. Número límite de iteraciones del algoritmo 800.
4. Probabilidad de cruces $p_c = 1$.
5. Probabilidad de mutaciones $p_m = 0.0001$.
6. Tamaño de élite 1% de la poblacion.

Características del algoritmo implementado:

1. Representación de los individuos: Se representan mediante cadenas binarias que corresponden con los puntos del espacio de búsqueda. $Ind = [\phi, \theta, \psi, t_x, t_y]$.
2. Función de aptitud: Cantidad de inliers para la matriz esencial modelada por el individuo.
3. Selección: Boltzmann.
4. Operador de cruce: operador de dos puntos que selecciona puntos de cruce aleatoriamente e intercambia los segmentos resultantes.
5. Operador de mutación: aleatoria bit a bit que cambia el valor del punto considerado para mutación.

4.2.3. Selección de imágenes clave

Una vez que tenemos buenos puntos emparejados, la clase *isKeyImage*, que se desarrolló e implementó en C++, hace la selección de las imágenes clave respetando las hipótesis expuestas en sec. 3.2.2. Esas hipótesis exigen contar con un número suficiente de puntos de interés para lograr el control, además de considerar la buena estimación de la geometría epipolar. Para esto, la clase *isKeyImage* usa un umbral para el tamaño del conjunto P_i de puntos de interés emparejados. Además de implementar el algoritmo de 8 puntos para el cálculo de la matriz fundamental y así evaluar el condicionamiento de la misma.

El umbral es calculado como sigue. La primera imagen F_1 de la secuencia $S = \{F_i | i \in \{1, 2, \ldots, m\}\}$ es la imagen clave I_1 de $M1$. Se encuentran entonces sus puntos de interés y sus descriptores correspondientes. Después se encuentra esto mismo para la imagen F_2 y se hace un emparejamiento de los puntos de interés entre las dos imágenes para generar el conjunto $P_1 = \#BuenosEmparejamientos$. El tamaño de este conjunto se guarda y define un umbral U que será usado posteriormente. Después se compara la imagen I_1 con las imágenes consecutivas de S, es

decir F_2 hasta la imagen F_n, donde n es el número de imagen para la cual el conjunto $P_n \leq \frac{1}{2}U$. Entonces la imagen F_n es seleccionada como la imagen clave I_2 que es comparada de forma parecida que en el caso anterior con F_{n+1}, y el umbral se actualiza a $U = P_{n+1}$. Después se sigue comparando la imagen clave I_2 con $F_{n+1,\ldots,}F_m$ donde m es el número de imagen para el cual $P_m \leq \frac{1}{2}U$. Y el proceso continua hasta llegar al final de la secuencia S.

Además de un mínimo número de buenos emparejamientos, una relativa distribución uniforme de los puntos de interés emparejados en la imagen asegura el cumplimiento de las hipótesis expuestas, referentes al control y a la factibilidad de navegación del robot. Una adecuada distribución de puntos permitirá utilizar de forma segura alguna restricción geométrica para el control, como el tensor trifocal o la matriz fundamental utilizados por ejemplo en [5].

Para evaluar la uniformidad de la distribución se proponen dos consideraciones. Uno, dividir la imagen en una rejilla, donde se busca que las imágenes clave tengan una apropiada distribución de puntos de interés en cada sección de la rejilla. Para las pruebas se eligió que cada sección contenga al menos un número d, establecido por el usuario, de puntos de interés. Dos, el cálculo de la matriz fundamental con el algoritmo de 8 puntos. Tal algoritmo crea un sistema de ecuaciones que se resuelve por mínimos cuadrados con SVD. Se verifica el correcto condicionamiento del problema con los valores singulares encontrados con SVD. Se sabe que cuando el último valor singular es similar al penúltimo valor singular, existen múltiples soluciones al problema [18]. Se define entonces una razón R_v igual a la división del penúltimo valor singular por el último valor singular. Cuando R_v esta dentro de un umbral consideramos que existe una única solución, y en consecuencia el problema está bien condicionado, o en otras palabras, hay una buena dispersión de puntos.

El método propuesto para la selección de imágenes clave se puede sintetizar en los siguientes 9 pasos:

1. Dada una secuencia de imágenes $S = \{F_i | i \in \{1, 2, \ldots, m\}\}$, se elige $F_{i=1}$ y F_{i+1} como la imagen 1 e imagen 2 respectivamente. Se hace que la imagen 1 sea la primera imagen clave.
2. Para las imágenes 1 y 2 se encuentran los puntos de interés y se calculan sus descriptores.
3. Se emparejan los descriptores de ambas imágenes, para encontrar un conjunto P de buenos emparejamientos.
4. Se divide la imagen 1 en una rejilla de 2×2.
5. Se establece el umbral U igual al tamaño del conjunto P.

6. Se evalua el condicionamiento de la matriz fundamental.
7. Se hace la imagen 2 igual a la siguente en la secuencia S, y se repiten los pasos 2 a 4.
8. Se verifican 3 condiciones. Uno, que $P \geq \frac{1}{2}U$. Dos, que existan al menos $d = 4$ puntos emparejados de la rejilla creada en el paso 4. Y tres, que el radio del penúltimo valor singular y el último del cálculo de la matriz fundamental esté dentro del umbral. Si se cumplen, ir al paso 7, de lo contrario seguir al paso 9.
9. Se establece la imagen 1 como la siguiente imagen clave. Se actualiza el umbral U haciéndolo igual al tamaño del último conjunto P. Ir al paso 7.

El paso 3 se realiza usando la clase *geneticMatcher* que se desarrollo, descrita anteriormente.

4.2.4. Alternativa a la selección de imágenes clave

El método de selección anterior asegura que una imagen clave I_i tendrá el traslape suficiente para el control con la siguiente imagen clave I_{i+1}. Sin embargo, este método no asegura siempre que la distancia recorrida entre la posición de captura de I_i e I_{i+1} sea la máxima.

Ésto último debido a las imágenes borrosas de la secuencia. Es posible que una imagen borrosa en F_i se haya capturado cerca de la posición de captura de I_i; con lo que esperaríamos una buena cantidad de traslape. No obstante, debido al *blurring* presente, la cantidad de puntos de interés emparejados será poca, y el método que interpretará poco traslape, la elegirá como I_{i+1}. Tener una imagen clave borrosa ocasionará que la siguente imagen clave se elija prematuramente por las mismas razones.

Otra desventaja del primer método es que éste considera que el traslape que existe entre una imagen clave I_i con la siguiente en la secuencia de entrenamiento es el máximo posible (umbral U en el paso 5 de la sec. 4.2.3). De esta manera, se elige como la siguiente imágen clave a aquella que tenga menos de la mitad de este traslape. El problema se torna más complejo cuando la imagen usada para fijar el umbral U es borrosa.

Normalmente en una secuencia capturada por un robot humanoide o por un humano con una cámara, es natural que se capturen imágenes borrosas; debido al movimiento brusco al desplazarse. Sin embargo, puede ocurrir en ocasiones que cuando se tiene una imagen borrosa, las inmediatas siguientes no presenten éste efecto.

En este tipo de secuencias, lo normal es tener una tasa de captura alta (la mínima que se probó fue de 5 $frames$ por segundo); por lo que el desplazamiento del robot desde la posición de captura de una imagen F_i a la posición de captura

de la imagen F_{i+5}, por ejemplo, puede ser muy pequeño. También es muy probable que aunque la imagen F_i sea borrosa algunas de las siguientes no lo sean.

Por lo anterior se propone una alternativa sencilla que arroja mejores resultados para la selección de imagenes clave. Teniendo en cuenta las condiciones de selección del primer método, se elige un conjunto de imágenes clave *potenciales* $IP = \{IP_i | i \in \{1, 2, \ldots, n\}\}$ como se aprecia en la Fig. 4.3. Después, del conjunto IP se elige como imágen clave a la que tenga mayor traslape. El umbral U también es modificado en caso de encontrar un mejor traslape.

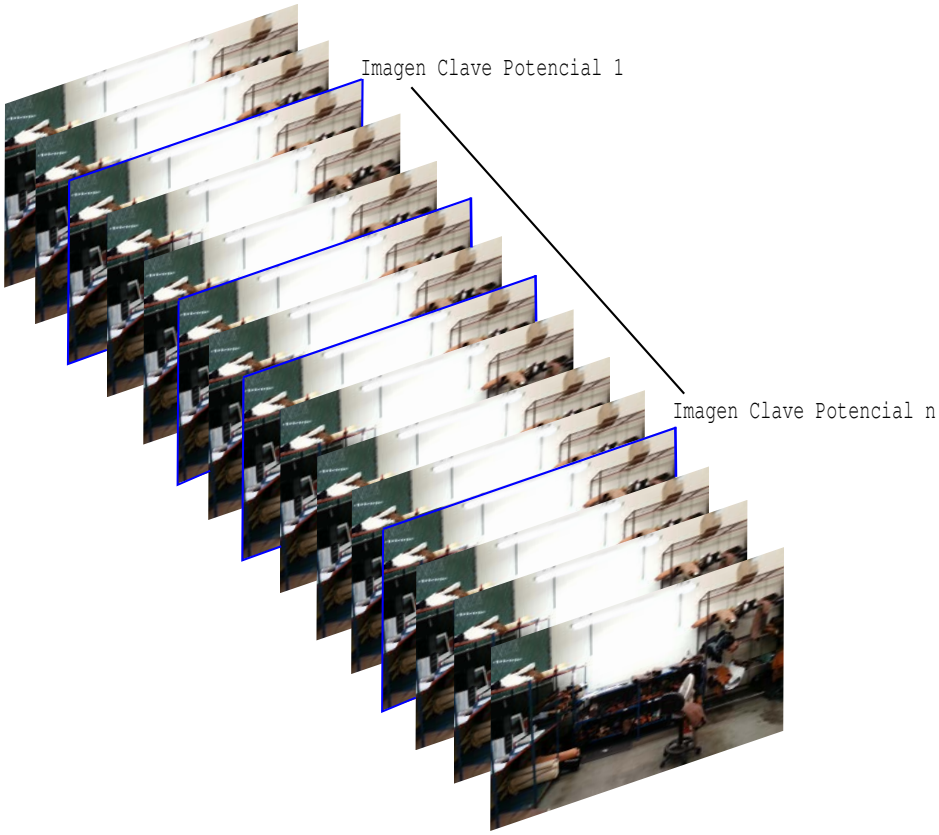


Figura 4.3: *Imagenes Clave Potenciales.*
La distancia en frames para la búsqueda de imágenes clave potenciales varía en función del incremento del umbral U(cantidad de puntos emparejados). Cuando en un frame F_i, U aumenta, significa que F_i hace mayor traslape con I_i que los anteriores, aumentando la probabilidad de que los frames siguientes a F_i tengan buen traslape.

Por defecto, se eligió que $n = 3$. Sin embargo, el valor de n puede aumentar en función al cambio en el umbral U. Como ejemplo, véase la Fig. 4.4. Este ejemplo parte de la imagen clave I_1, la primera en la secuencia de entrenamiento.

Como se ve en la figura, se fija el umbral $U = 162$ comparando la primera imagen clave con la inmediata en la secuencia de entrenamiento. Sin embargo este umbral no

es del todo confiable pues F_2 presenta *blurring*. En cambio, la imagen F_3 no presenta este efecto y se actualiza U a 517 pues se encontró un mayor traslape. También se actualiza n para que permita una búsqueda de imágenes clave potenciales en mayor cantidad de $frames$.

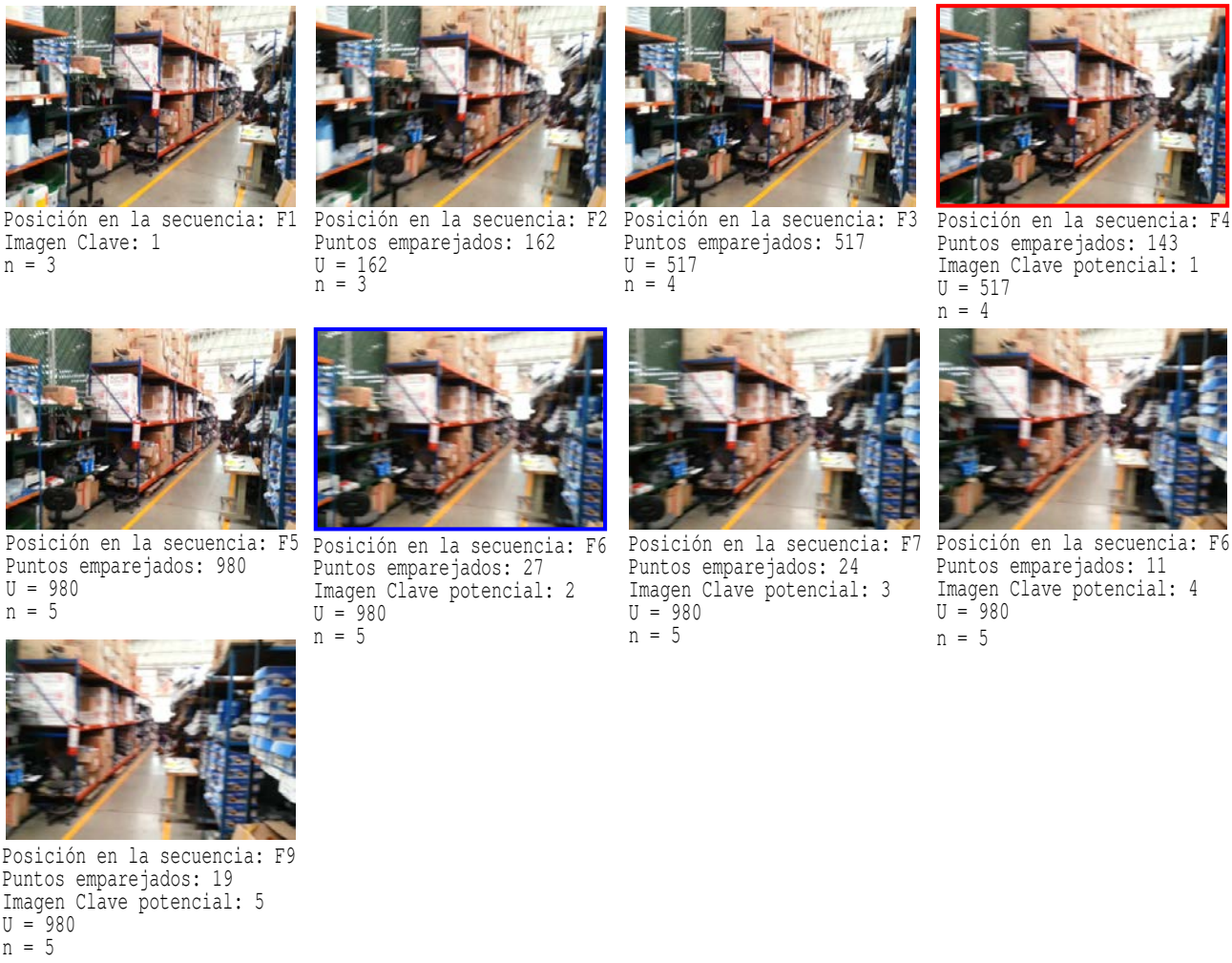


Figura 4.4: *Alternativa a la selección de imágenes clave.*
El ejemplo ilustra la selección de la imagen clave I_2. La cantidad de imágenes clave potenciales a buscar aumenta cuando U aumenta, pues se considera que hay mayor probabilidad de encontrar imagenes con suficiente traslape en la secuencia después de cada incremento de U. En este caso en particular, la secuencia giró por lo que después de U=980 no se encontró mejor traslape. La imagen resaltada con azul es la que el primer método expuesto seleccionaría; la roja es la seleccionada con el segundo método propuesto.

La cantidad de puntos emparejados entre I_i y F_5 es todavía mayor, fijando $U = 980$. Por lo que se considera que la probabilidad de encontrar una buena

cantidad de traslapes después de F_5 aumente, así que se incrementa n para que busque imágenes clave potenciales en más cantidad de $frames$ de la secuencia.

El segundo método de selección propuesto da oportunidad de buscar una imagen clave bien condicionada, es decir, alejada lo máximo posible de la anterior imagen clave, pero con el mayor traslape necesario entre ellas. Es importante que el algoritmo propuesto logre una mayor cantidad de puntos emparejados a pesar de tener imágenes desenfocadas. Por eso se hizo una comparativa entre varios detectores y descriptores para elegir al que mejor respuesta tiene para éste y otros problemas.

4.2.5. Comparativas entre diferentes combinaciones detector-descriptor

Para elegir el mejor detector y descriptor a usar en la clase *geneticMatcher* se llevó a cabo una comparativa y selección a partir de diferentes combinaciones detector-descriptor. Se corrió el algoritmo de la clase propuesta *isKeyImage* para encontrar esta combinación. En seguida se muestran los resultados.

En [30, 31, 37] se encuentra que el descriptor con mejor desempeño es SIFT seguido por el SURF, pero como veremos a continuación una modificación al parametro del SURF ofrece mejores resultados para nuestro caso particular en el problema de selección de imágenes clave. Estas pruebas fueron realizadas considerando el primer método de selección expuesto. Las pruebas se hicieron con una secuencia de 445 imágenes, capturada con el robot tipo humanoide Nao. Tal secuencia contiene varias imágenes con blur. Se usaron los parámetros por defecto implementados en las funciones de OpenCV.

En la figura Fig. 4.5, se muestran los resultados con el detector FAST, y como descriptores a comparar, SIFT y SURF. La gráfica a) muestra la razón que hubo entre inliers y outliers, y en la gráfica b) se ve como prácticamente cada imagen fue seleccionada como imagen clave. La gráfica c) muestra cúal fue el número de puntos encontrados y emparejados. En las tres gráficas se observa que el comportamiento del método es el mismo usando como descriptor SIFT y SURF. En las gráficas se aprecia como los puntos graficados se empalman al no haber diferencia dada por el descriptor. Como puede observarse en las figuras, usando este detector, sólo se llegó a la imagen 34 de la secuencia.

La poca cantidad de puntos de interés detectados significa menor cantidad de buenos emparejamientos y esto provoca que prácticamente cada frame fuera elegido como imagen clave por el algoritmo. Además, no tiene una buena distribución de puntos en la imágen. La ejecución convergió en la imagen 34 de la secuencia al encontrar menos de 8 emparejamientos necesarios para calcular la matriz fundamental y validarlos con la restricción epipolar. Se tiene buena persistencia en los puntos de interés pero en términos generales el desempeño del detector no es bueno para la aplicación, debido a la poca cantidad de puntos detectados.

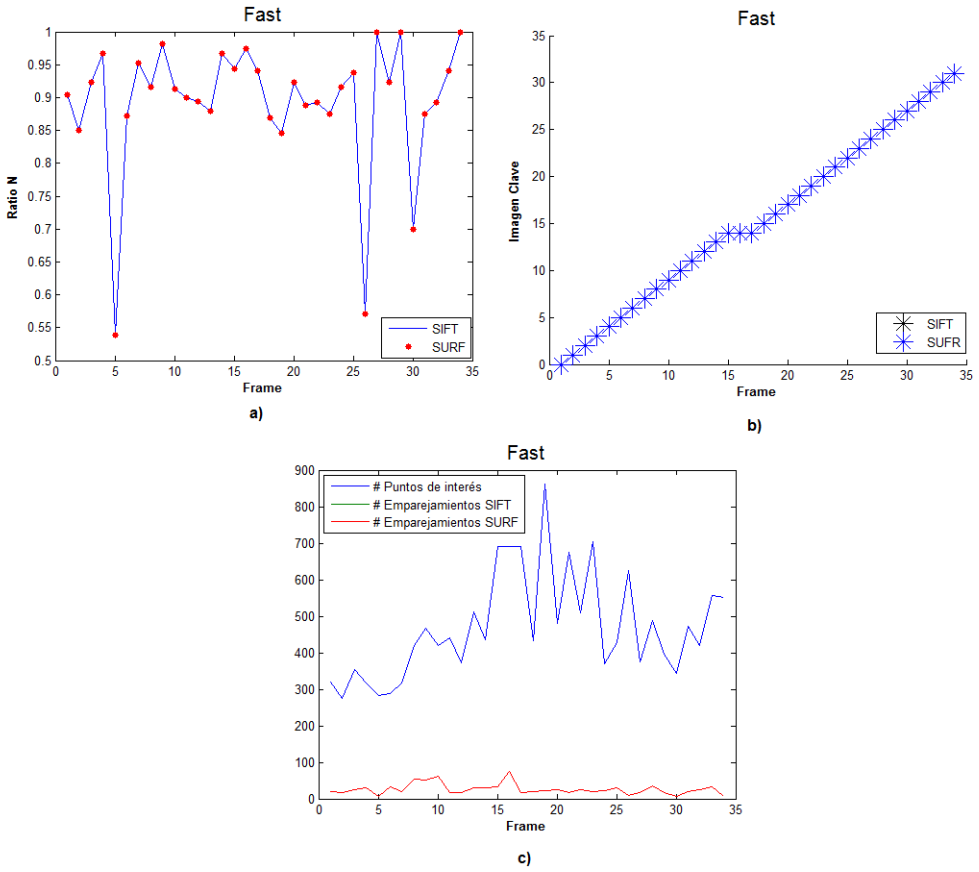


Figura 4.5: *Resultados detector FAST.*
De arriba a abajo: La primera gráfica muestra la razón inliers/outliers. La segunda el número de imágenes clave conforme se avanza en la secuencia. Para este caso se observa que prácticamente todas las imágenes fueron seleccionadas como imagen clave. La última gráfica muestra la cantidad de puntos de interés detectados (en azul) y el número de buenos emparejamientos (en rojo y verde).

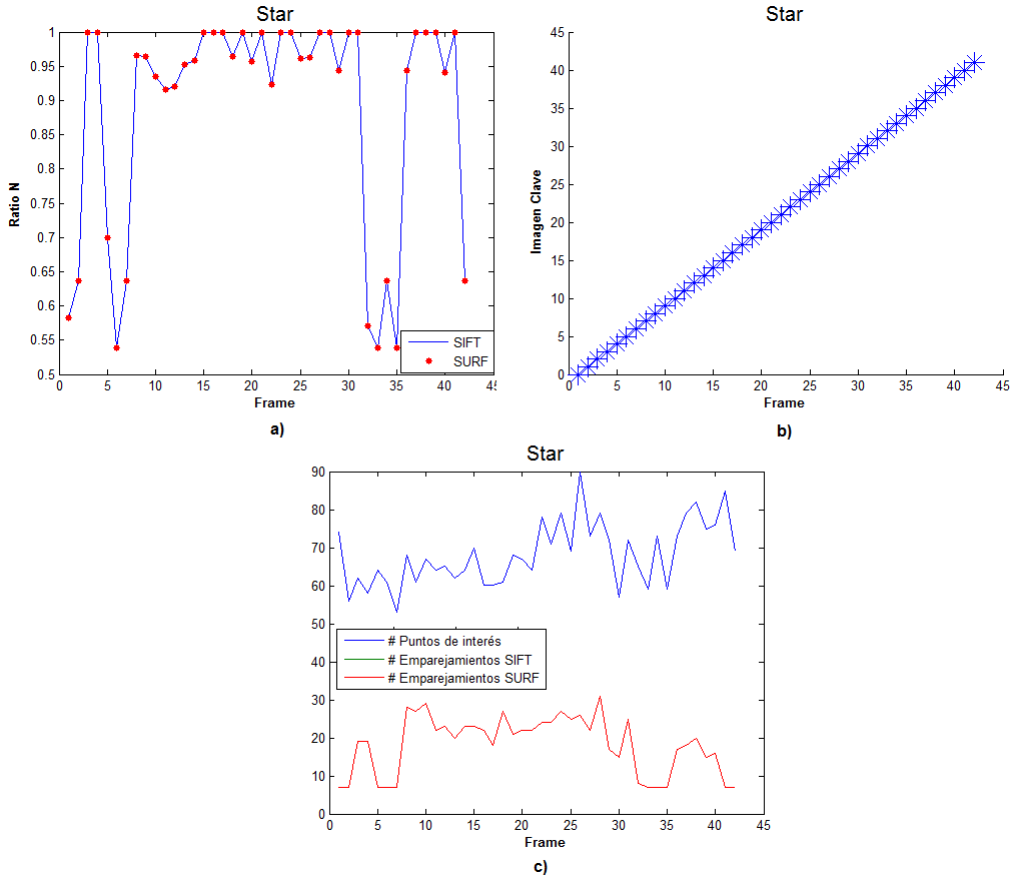


Figura 4.6: *Resultados detector STAR.*
Se observa en las tres gráficas un menor desempeño en comparación con el detector FAST.

Los resultados para el detector STAR se muestran en la Fig. 4.6. En la gráfica a) se muestra la razón $\frac{inliers}{outliers}$ que fue ligeramente menor que con el detector FAST con un promedio de 0.88, mientras que en el caso anterior fue de 0.89. En la gráfica b) se observa un menor desempeño, pues más imágenes clave fueron seleccionadas que en el caso anterior. Esto se explica con la gráfica c) pues poca cantidad de puntos de interés fue encontrada, aunque se alcanzó a procesar hasta la imagen número 42, la primera en la secuencia que presenta blurring. Nuevamente el uso de descriptor SIFT o SURF no marcó ninguna diferencia en el comportamiento del método, como se aprecia en las gráficas.

El detector SIFT tiene mejores resultados que los anteriores, tal como se observa en la Fig. 4.7. En la figura se presentan las gráficas de la combinación detector-descriptor con mejores resultados. Usando el detector SIFT se llegó a la imagen 95 de la secuencia, y menos imágenes clave fueron seleccionadas. Y el promedio de la razón $\frac{inliers}{outliers}$ fue de 0.93. En este caso también fue indistinto el uso del descriptor SIFT o SURF.

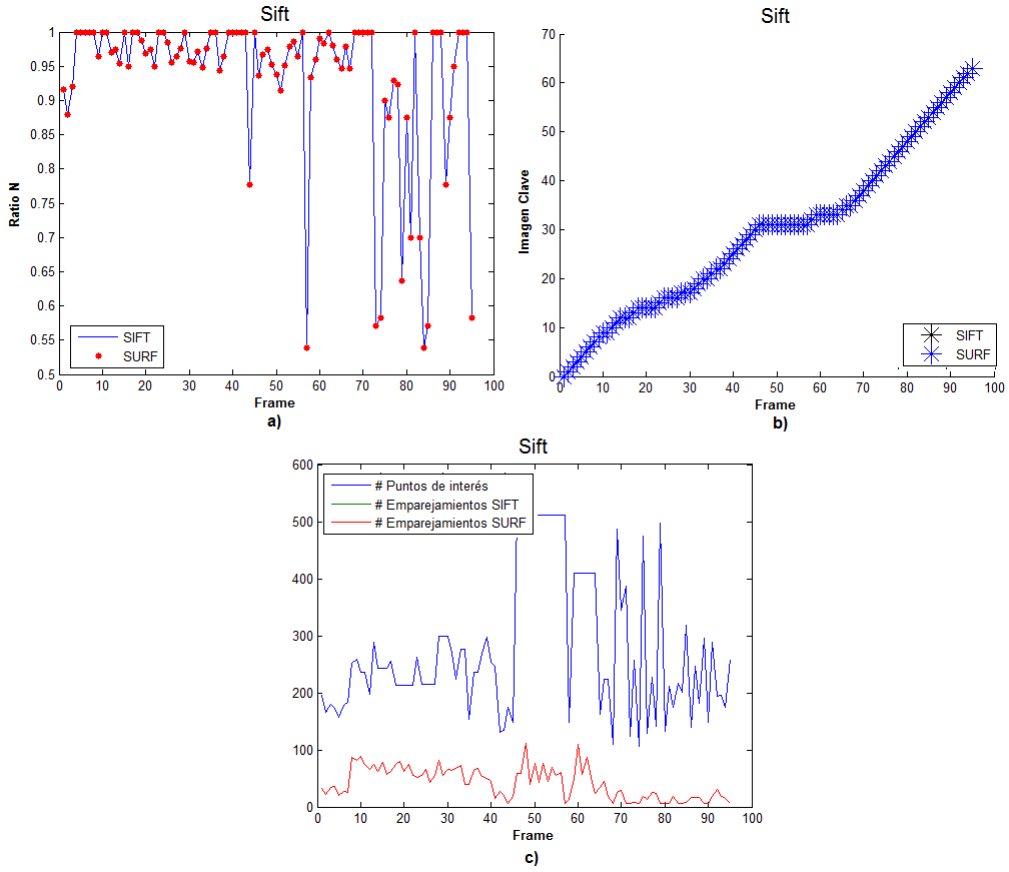


Figura 4.7: *Resultados detector SIFT.*
Uno de los detectores que presentó mejores resultados. Gráfica a) La razón inliers/outliers es buena. Gráfica b) Un menor número de imágenes clave fueron seleccionadas respecto al uso de detectores FAST y STAR. Gráfica c) un buen número de puntos de interés fueron detectados y emparejados.

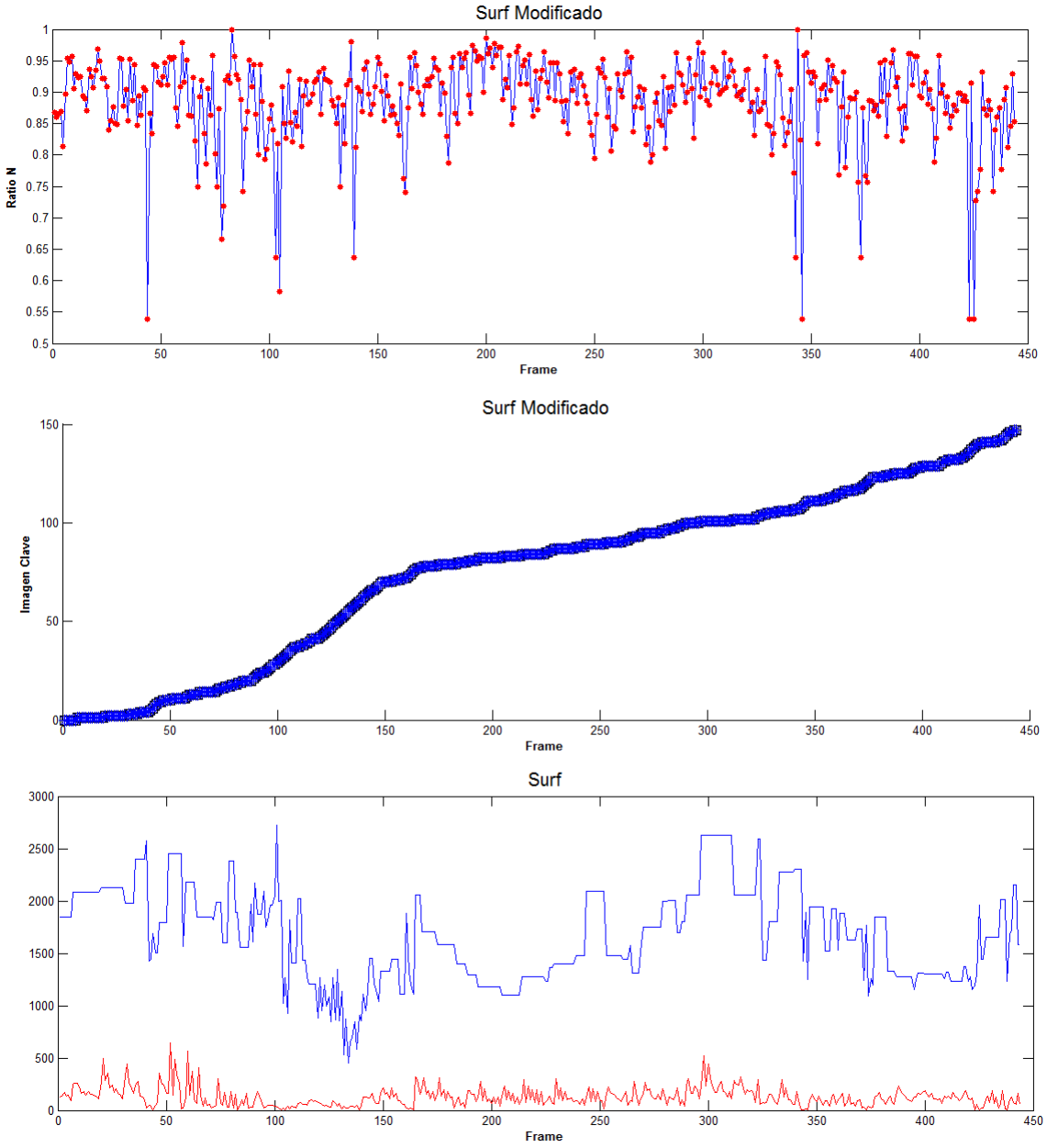


Figura 4.8: *Resultados detector SURF.*
Es modificado en uno de sus parametros, el umbral que es aplicado al Hessiano. Con este cambio se observa en cada una de las gráficas un mejor desempeño en comparación con los casos anteriores.

En la Fig. 4.8 se muestra la combinación que fue elegida como la mejor. Se usa el detector SURF donde el primer parámetro, referente al umbral que es aplicado al Hessiano de una imagen, es fijado en 10. La respuesta con el descriptor SIFT o SURF es similar a la hora de realizar el emparejamiento. Se logran detectar suficientes puntos de interés.

En la tabla 2 se presenta un resumen de las comparaciones. La primera columna de la tabla contiene la combinación detector-descriptor a evaluar. La segunda columna incluye el promedio de puntos detectados. La tercera columna posee el promedio de buenas correspondecias encontradas. La cuarta columna contiene la información referente a la razon$\frac{inliers}{outliers}$. La quinta y última columna muestra la cantidad de imágenes que lograron ser procesadas menos la cantidad de imágenes seleccionadas como clave, para encontrar el número de imágenes descartadas. De la tabla podemos concluir que el detector SURF obtuvo mejores resultados, independientemente del descriptor usado. La razón es la mayor cantidad de puntos encontrados; cuando hay mayor cantidad de puntos hay una mayor probabilidad de tener más emparejamientos adecuados.

Detector-Descriptor	*Puntos*	*Correspondencias*	*inliers/outliers*	*Imágenes descartadas*
FAST-SIFT	470.1471	26.1765	0.8933	34-32 = 2
FAST-SURF	470.1471	26.1765	0.8933	34-32 = 2
FAST-ORB	470.1471	26.1765	0.8933	34-32 = 2
FAST-BRIEF	470.1471	26.1765	0.8933	34-32 = 2
STAR-SIFT	68.1905	18.2143	0.8864	42-42 = 0
STAR-SURF	68.1905	18.2143	0.8864	42-42 = 0
STAR-ORB	68.1905	18.2143	0.8864	42-42 = 0
STAR-BRIEF	68.1905	18.2143	0.8864	42-42 = 0
SIFT-SIFT	274.8421	41.9263	0.9321	95-64 = 31
SIFT-SURF	274.8421	41.9263	0.9321	95-64 = 31
SIFT-ORB	274.8421	41.9263	0.9321	95-64 = 31
SIFT-BRIEF	274.8421	41.9263	0.9321	95-64 = 31
SURF-SIFT	1672.6	127.7725	0.8865	444-148 = 296
SURF-SURF	1672.6	127.7725	0.8865	444-148 = 296
SURF-ORB	1672.6	127.7725	0.8865	444-148 = 296
SURF-BRIEF	1672.6	127.7725	0.8865	444-148 = 296
GFTD-SIFT	431.2667	9.3333	0.7501	15-15 = 0
GFTD-SURF	431.2667	9.3333	0.7501	15-15 = 0
GFTD-ORB	431.2667	9.3333	0.7501	15-15 = 0
GFTD-BRIEF	431.2667	9.3333	0.7501	15-15 = 0

Tabla 2. Comparativas. Estos resultados muestran que cada uno de los descriptores tuvo el mismo resultado. Fue la cantidad de puntos detectados y su distribución en la imagen lo que marcó la diferencia.

4.3. Organización de la Memoria Visual

Como se ha comentado, se propuso la organización de la MV en un grafo. Éste se presenta mediante una interfaz gráfica de usuario que es expuesta en el siguiente capítulo. Tal interfaz permite al usuario explorar la MV y visualizar información importante. La construcción de caminos visuales Γ para formar M3 es hecha manualmente concatenando aristas de M2. Lo que permite construir un mapa conceptual de alto nivel con conceptos que tienen sentido para los humanos como: cuarto, puerta, almacén entre otros. Esto último para permitir una posible interaccion robot-humano.

Cabe menciona que la MV construida solo está formada por el conjunto de imágenes clave organizadas topológicamente, no incluye información de comandos del robot asociados a las imágenes clave durante la etapa de aprendizaje, a diferencia del trabajo [21]. Incluir esta información adicional facilitaría el control del robot durante la etapa de navegación autónoma, sin embargo, resulta más interesante no contar con esta información.

5 Resultados

Para demostrar el desempeño de las soluciones propuestas se realizaron varias pruebas. En este capítulo se presentan los resultados obtenidos al ejecutar y evaluar los diferentes algoritmos para la construcción de la MV.

5.1. Solución propuesta para la estimación de la Matriz Esencial (AG)

Tal como se detalló en el capítulo 4. El algoritmo genético arroja como resultado una matriz esencial que codifica la posición relativa entre las ubicaciones de cámara desde donde se capturaron dos imágenes. Se hicieron diferentes pruebas para conocer la precisión del algoritmo en la estimación de esta posición. Para una de estas pruebas se utilizaron imágenes sintéticas generadas a partir de un ambiente simulado en *Webots* [27] donde un robot móvil se desplaza en este ambiente y captura fotos del mismo.

Figura 5.1: *Ejemplo de imágenes generadas en la simulación.*

Los algoritmos genéticos son de tipo aleatorio, por lo cual no siempre encuentran la misma solución y aproximación al resultado. Sin embargo, lo ideal es que estadísticamente lo hagan. Así que se hicieron un total de 125 corridas del algoritmo para encontrar 125 matrices esenciales. Cada una de estas matrices se descompuso para encontrar otra matriz en la forma

$$\begin{bmatrix} \mathbf{R}_{3\times3} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix},$$

que representa la posición de la cámara. La rotación y la traslación estimadas se compararon con los datos de localización real disponibles para la secuencia obtenida en simulación y con los datos de odometría disponibles para la secuencia capturada por el robot humanoide Nao.

Para poder comparar las rotaciones la submatriz **R** de la odometría y de la descomposición de **E** se expresaron en términos de Rodrigues. Para una mejor visualización de los errores, éstos se muestran en gráficas de cajas en la Fig. 5.2. En la Tab. 5.1 se muestran los resultados de la comparación entre el promedio de las ejecuciones del AG y RANSAC. Los parámetros del AG son los presentados en la metodológia; los parámetros para RANSAC son un umbral de 1.1 píxeles a la línea epipolar y una confianza de 99 %. Esta confianza está relacionada con el número de iteraciónes para el RANSAC. La probabilidad de encontrar un modelo con únicamente emparejamientos adecuados está dada por $1 - (1 - n8)k$, que es la confianza c; $8n$ es la probabilidad de seleccionar 8 emparejamientos buenos y k el número de iteraciones. La función de OpenCV para RANSAC recibe como parámetro la confianza c y hace el cálculo para n y k.

	Algoritmo Genético	RANSAC (8 puntos)
Error del ángulo de rotación	1.18	1.34
Error del ángulo entre el vector de Rodrigues	0.24	0.28
Error del ángulo entre el vector de traslación	89.99	83.76
Cantidad de puntos emparejados	875 (promedio)	821

Cuadro 5.1: *Resultados estadísticos en grados.*

El caso presentado es para un par de imágenes clave de la memoria visual. Es conveniente, sin embargo, evaluar el AG para una mayor cantidad de pares de imágenes. Por esta razón se hicieron pruebas para cada par de la memoria visual, y se calcularon los errores correspondientes. Los resultados se muestran en la Fig. 5.3.

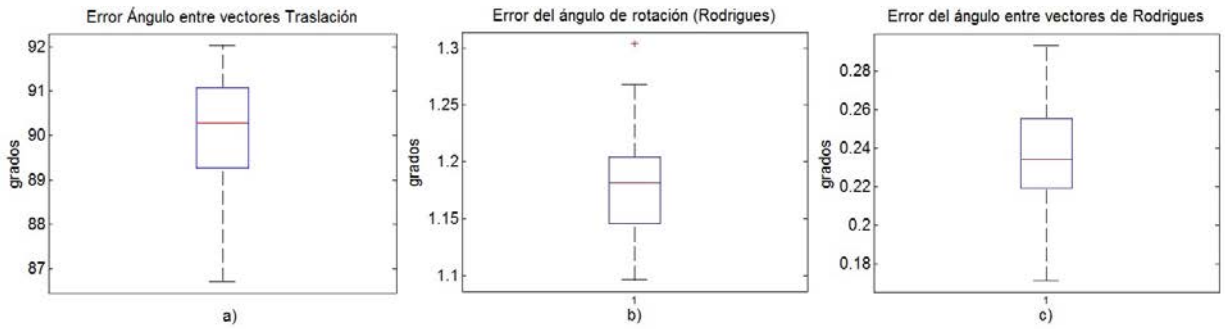


Figura 5.2: *Gráficas de Caja.*
Las gráficas muestran los errores obtenidos para el caso de un par de imágenes.

Para ambos casos los errores son admisibles. Por ejemplo, en el primer caso, para un sólo par comparado con RANSAC, se demostró que la eficiencia del AG es ligeramente superior. Con el AG se logró encontrar una mayor cantidad de puntos emparejados que satisfacen la restricción epipolar. Esto debido al mayor número de iteraciones que efectúa el AG, logrando así un refinamiento de la Matriz Esencial. Para el caso de varios pares de imágenes el error se mantiene admisible también.

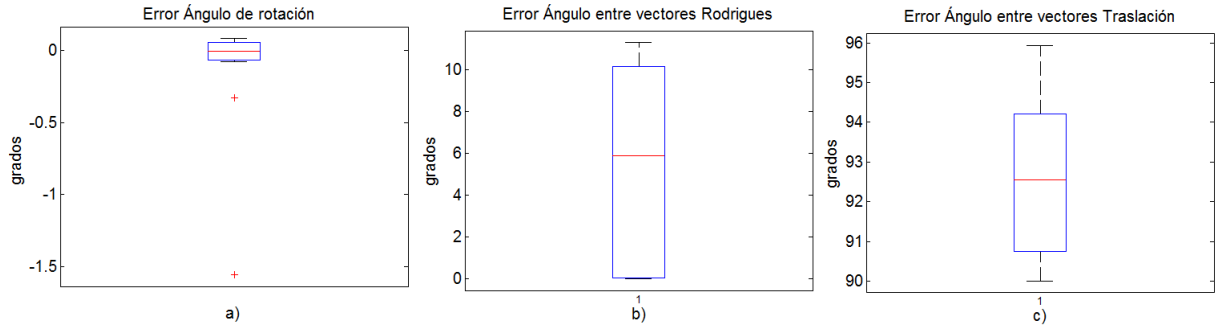
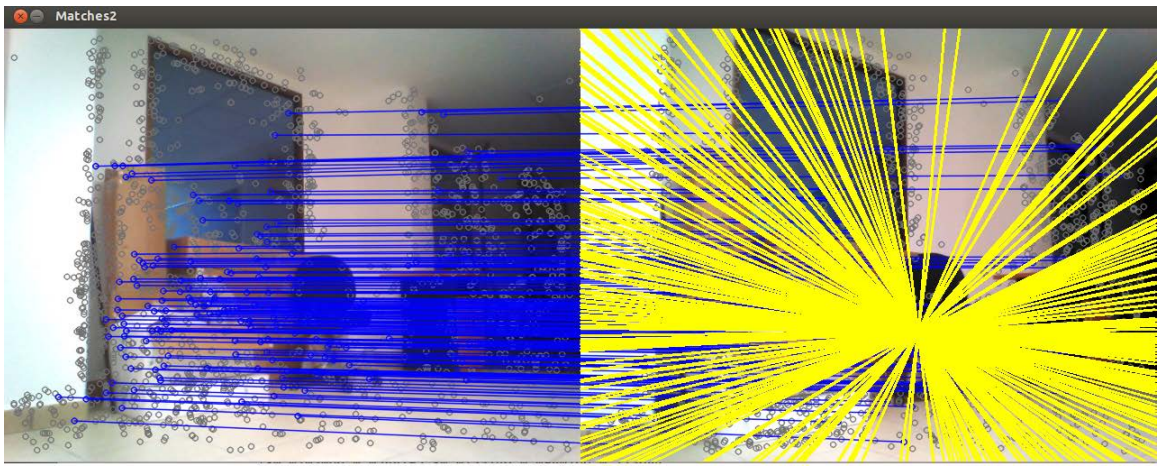


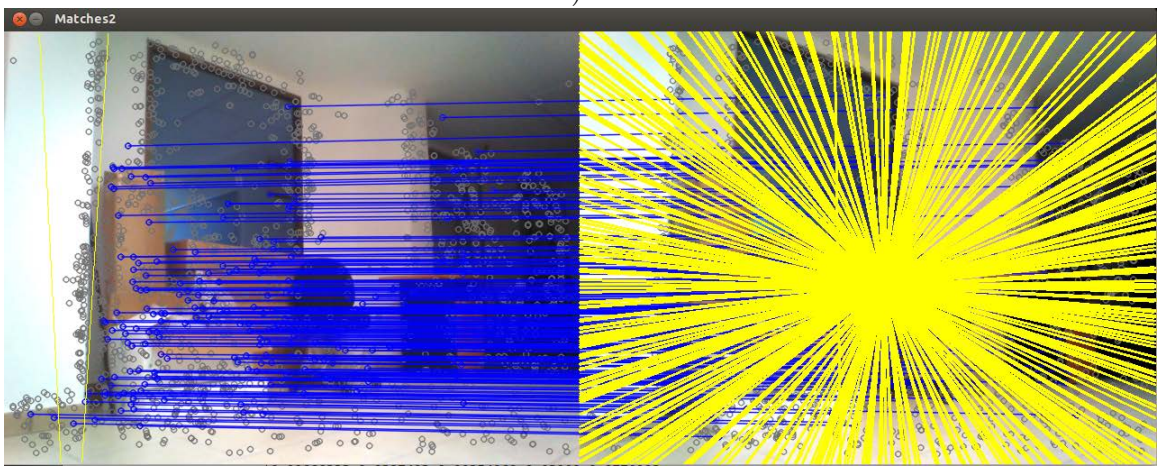
Figura 5.3: *Gráfica de Cajas.*
Las gráficas contienen información de los errores obtenidos para todos los pares de la memoria visual de imágenes sintéticas.

De igual forma se probó el método propuesto para la construcción de la MV con imágenes capturadas por el robot humanoide en un laboratorio. Otra secuencia de imágenes se capturó con la cámara de un teléfono celular en un almacén de una fábrica de calzado. Para ambas secuencias se construyó la MV con buenos resultados pues se usó alrededor del 25 % de las imágenes de la secuencia de entrenamiento para modelar el ambiente con la MV. Para la secuencia del humanoide Nao, de 445 imágenes de entrenamiento se encontró la MV correspondiente con 130 imágenes. Para la secuencia del almacén, de 416 imágenes de entrenamiento se construyó una MV con 93 imágenes.

En la Fig. 5.4, se observan dos resultados para la Matriz Esencial encontrada con el AG, para el caso de imágenes reales. Se observan resultados satisfactorios.



a)



b)

Figura 5.4: *Ejemplos de la restricción epipolar para imágenes reales. Las imágenes de muestra se tomaron de una secuencia capturada por el robot humanoide Nao. En esta parte del recorrido el robot describe un movimiento prácticamente recto de la primera a la segunda imagen. Se observa que el epipolo se encuentra en en el centro de la segunda imagen para ambos casos; algo esperado debido a la naturaleza del movimiento del robot.*

5.2. Solución propuesta para el emparejamiento de puntos de interés

Se probaron varias corridas de la clase *geneticMatcher* con buenos resultados, como se mostró en la sección anterior. Por otro lado, el algoritmo de 8 puntos que se implementó para la estimación de la matriz fundamental y como consecuencia evaluar el condicionamiento de la geometría epipolar en función al condicionamiento del sistema de ecuaciones obtuvo resultados satisfactorios.

5.3. Memoria visual resultado

En esta subsección se presentan los grafos resultantes de los dos métodos propuestos para la selección de imágenes clave. El grafo de la Fig. 5.5 es el grafo que corresponde a la MV construida con el primer método para la selección de imágenes clave. En general, este método logra eliminar una cantidad considerable de imágenes. Por otro lado, en la Fig. 5.6 se muestra la MV que resultó de la alternativa a la selección de imágenes clave que se presentó en la sec. 4.2.4. De un total de 445 imágenes de entrenamiento, el primer método logró eliminar 258 imágenes y el segundo método 315.

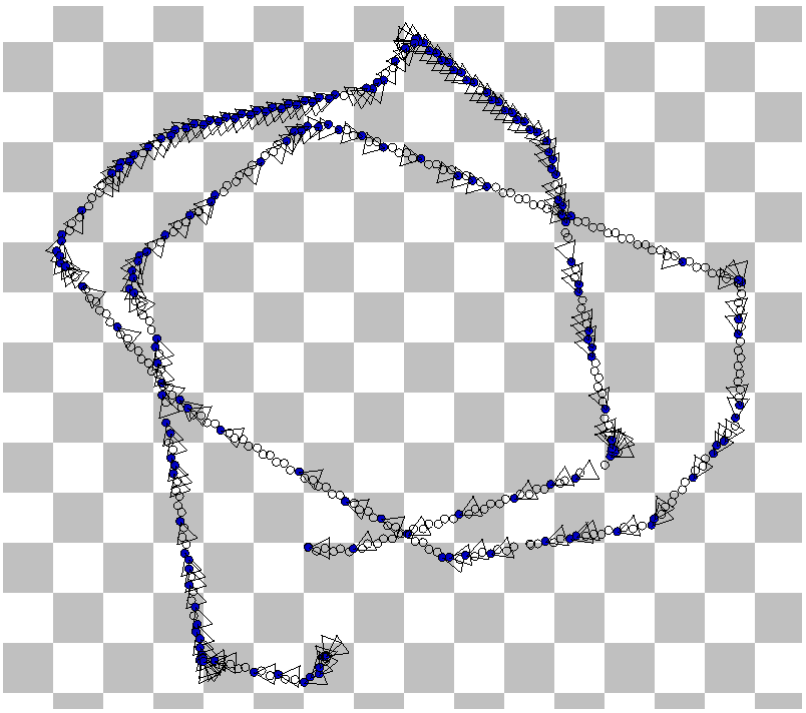


Figura 5.5: *Grafo 1.*
Los resultados que se presentan son para una secuencia capturada por el robot humanoide Nao. La secuencia contiene 445 imágenes y se dispone de la odometría para poder dibujar las posiciones de la cámara.

Como puede observarse, el segundo método logra reducir la cantidad de imágenes redundantes. En el primer grafo se observa cómo en dos tramos de la MV se eligió prácticamente cada imagen de la secuencia como imagen clave. La causa de ésto es la cantidad de imágenes borrosas en ese tramo, además de la falta de textura en esa parte del ambiente. El método alternativo que busca imágenes potenciales evita este tipo de problemas.

El trabajo de tesis logró diferentes contribuciones: se logró implementar una solución efectiva al problema de emparejamiento de imágenes mediante el uso de

algorimtos genéticos; se implementó satisfactoriamente el algoritmo de 8 puntos para evaluar hipótesis de control; una interfaz gráfica fue implementada para una mejor visualización de la MV. En el siguiente capítulo se exponen los alcances de tales contribuciones así como las conclusiones y trabajo futuro.

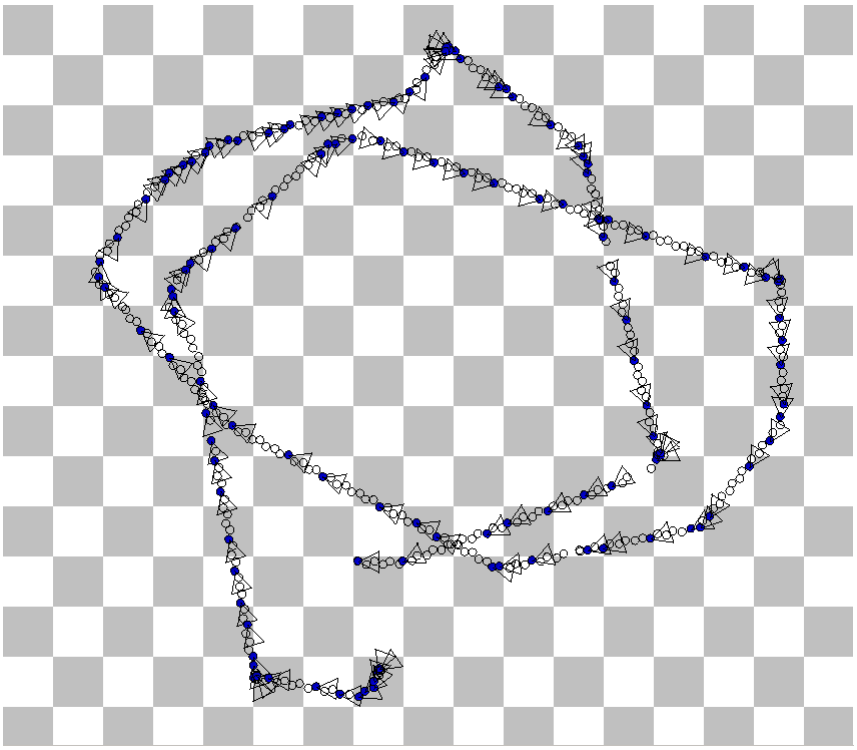


Figura 5.6: *Grafo 2.*
El grafo corresponde a la memoria visual que fue construída con el segundo método propuesto para la elección de imágenes clave. Se logra solucionar el problema de imágenes clave borrosas, además de reducir aún más el número de imágenes de la memoria visual.

6 Discusión de los resultados, conclusiones y perspectivas

Las contribuciones técnicas del proyecto de tesis son varias: el desarrollo y diseño de un algoritmo genético utilizado para realizar la reconstrucción 3D de la escena, reconocimiento de objetos, estimacion de la posición de cámara, *visual tracking* entre otros; la MV contribuye al estado del arte de la construcción de mapas para la navegación, en particular, para robots humanoides.

El proyecto de tesis tiene contribuciones también en el aspecto industrial. El proyecto fue presentado a 2 empresas de la región con buenos resultados. Ambas expresaron interés en el proyecto propuesto. Por ejemplo, en una fábrica de calzado se observó que la organización y el manejo del almacén podrian optimizarse. Se presentó un sistema autónomo para la organización de almacenes en la industria. Para tal sistema se propuso un robot móvil tipo carro con un brazo robótico embebido; el robot se conduciría con ayuda de la MV. Este trabajo es presentado en [23].

La representación del ambiente de navegación propuesta tiene varias ventajas:

- Su construcción es más fácil para ambientes complejos en comparación con reconstrucciones 3D.
- Uso mínimo de memoria, pues no se almacena una gran cantidad de información.
- A diferencia de modelos métricos, basados en fusión sensorial (visón, IMUs, odometría), no es vulnerable a errores métricos ocasionados por el error en el modelo cinemático del robot por ejemplo.
- La localización es más rápida.
- La planificación de rutas es más fácil.

6.1. Conclusiones

En este trabajo se presentó una propuesta para la construcción automática de un mapa topológico a partir de imágenes. El enfoque establecido en esta tesis parte de una gran colección de imágenes capturadas en diferentes posiciones dentro de un ambiente. Primero se construyó un mapa de bajo nivel que consiste en un grafo

donde cada nodo corresponde a la posición de captura de cada imagen clave unido por aristas, después un mapa de alto nivel se construyó a partir del primer mapa.

Se presentó un modelo de MV para contribuir a la navegación autónoma de robots, en particular de robots humanoides. Para cumplir con este objetivo se propusieron varios enfoques, por ejemplo, se presentó un algoritimo genético como alternativa al método RANSAC para verificar la geometría epipolar entre un conjunto de puntos emparejados con información ruidosa. Cuando se emplea el algoritmo genético para estimar la matriz esencial en lugar de RANSAC se logró encontrar un conjunto confiable de puntos emparejados. Se hicieron varias pruebas para verificar la confiabilidad del método propuesto.

El algoritmo genético fue usado para verificar el emparejamiento entre dos imágenes. La cantidad de traslape entre imágenes se determinó para decidir incluir una imagen dada dentro de la MV. El primer procedimento seguido para la construcción de la MV fue satisfactorio, pues reducía considerablemente la cantidad de imágenes clave, sin embargo este procedimiento tenía problemas con las imágenes borrosas y con poca textura de la secuencia. Un método alternativo a la selección de imágenes clave permitió reducir aún más la cantidad de imágenes clave conservando el traslape necesario para el control.

Se implementó el algoritmo de 8 puntos para encontrar la matriz fundamental con el fin de evaluar el condicionamiento del sistema de ecuaciones y en consecuencia el condicionamiento de la geometría epipolar. Verificar tal condicionamiento fue fundamental para elegir una imagen clave pues el adecuado condicionamiento de la geometría epipolar es importante para un posible control visual que dirija al robot a navegar de forma autónoma por la ruta establecida en la MV.

6.2. Trabajo futuro

Teniendo en cuenta los resultados del proyecto de tesis, el proyecto puede extenderse a lo siguiente: usar un algoritmo genético multiobjetivo para encontrar la matriz fundamental; aprovechar la representacion de grafo de la MV para la creación de un mapa de alto nivel de forma automática. En cuanto al proyecto con la industria, el trabajo podría extenderse a la implementación del sistema de control del robot.

7 Apéndice I

Memoria visual resultado (interfaz gráfica)

Para facilitar la construcción de la MV, así como su análisis, se propuso la implementación de una interfaz gráfica. La Fig. 7.1 muestra esta interfaz. La barra de herramientas permite al usuario lo siguiente: (1) Seleccionar un archivo con los nombres de las imágenes de entrenamiento a ser procesadas. (2) Correr el algoritmo para la construcción de la MV. (3) Dibujar el grafo resultante. Es posible también introducir algunos de los parámetros del algoritmo. Como el número de filas y columnas de la rejilla que divide la imágen para examinar la dispersión de puntos de interés. Además es posible elegir entre la matriz fundamental evaluada con RANSAC y la matriz esencial encontrada con un algoritmo genético para evaluar los emparejamientos de características visuales.

Figura 7.1: *Interfaz Gráfica.*

El grafo que resulta del algoritmo describe las posiciones de la cámara desde donde se tomó cada imagen, y toma como referencia la información de la odometría cuando ésta existe. Cuando una imagen pertenece al conjunto de imágenes clave ésta se resalta de las demás. Se dibuja así mismo el campo de vista de la cámara para poder visualizar mejor la pose de la cámara. En la Fig. 5.5, se exhibe un ejemplo de la visualización de la MV que ofrece la interfaz. Hacer *click* en alguna de las imágenes clave permite al usuario el acceso a ciertas características de esa imagen, como son el número de imágen en el conjunto. Como se aprecia en la Fig. 7.2, la ventana despliega la cantidad de puntos de interés entre esa imagen clave y la anterior imagen clave, así como las diferentes características que llevaron a considerar esa imagen en particular como imagen clave.

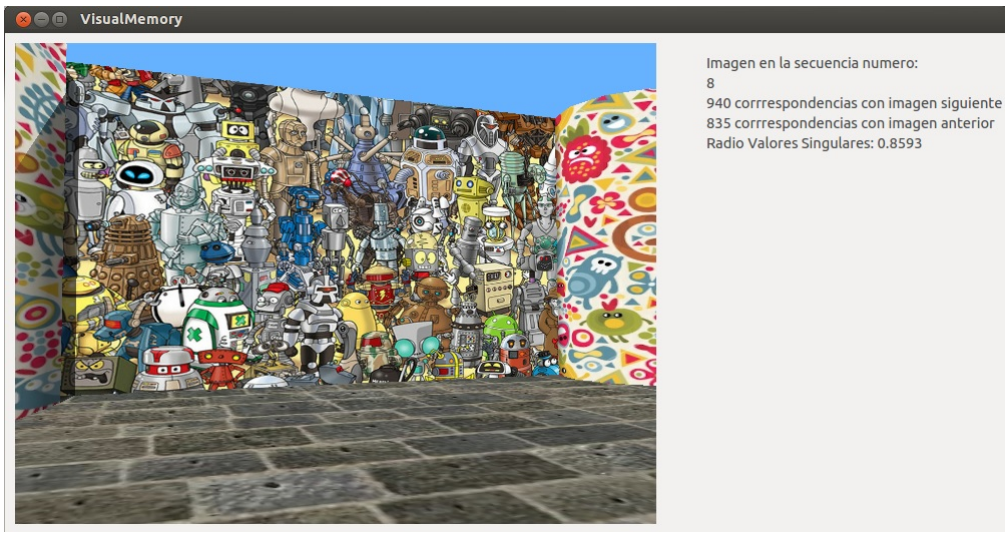


Figura 7.2: *Ejemplo de Características.*
Ejemplo de la visualización de las características de la Imagen Clave. Este ejemplo es para la memoria visual construida a partir de las imágenes sinteticas.

Bibliografía

[1] Agrawal, M., Konolige, K., and Blas, M. R. (2008). CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching. *Lecture Notes in Computer Science. Computer Vision ECCV*. Vol. 5305, No., pp 102-115.

[2] Aldebaran Robotics. (2008). `http://aldebaranrobotics.com/`.

[3] Ayala-Ramirez V., Garcia-Capulin C., Perez-Garcia A. and Sanchez-Yanez R. (2009). Circle detection on images using genetic algorithms. *Pattern Recognition Letters*. Vol. 27, No. 6, pp 652-657.

[4] Bay H., Tuytelaars T. and Van Gool L. (2006). "SURF: Speeded Up Robust Features". *Computer Vision and Image Understanding*. Vol. 110, No. 3, pp 346-359.

[5] Becerra H. M., Sagüés C., Mezouar Y. and Hayet J.B. (2014). Visual navigation of wheeled mobile robots using direct feedback of a geometric constraint. *Autonomous Robots*. Vol. 37, No. 2, pp 137-156.

[6] Calonder M., Lepetit V., Strecha C. and Fua, P. (2010). BRIEF: Binary robust independent elementary features. *In Proc. of ECCV*. Vol. 6314, No., pp 778-792.

[7] Courbon J. Navigation de Robots Mobiles par Mémoire Sensorielle. (2009). Thése de doctorat. *Université Blaise Pascal - Clermont-Ferrand II (08/12/2009), Philippe Martinet (Dir.)*.

[8] Courbon J., Mezouar Y. and Martinet P. (2009). Autonomous Navigation of Vehicles from a Visual Memory Using a Generic Camera Model. *IEEE Transactions on Intelligent Transportation Systems*. Vol. 10, No. 3, pp 392-402.

[9] Cuevas F.J., Mendoza F. and Servin M. (2006). Window Fringe Pattern Demodulation by Multi-Functional Fitting Using a Genetic Algorithm. *Optics Communications*. Vol. 261, No., pp 231-239.

[10] Cuevas F.J., Gonzales 0., Susuki Y., Hernandez D., Rocha M. and Alcala N. (2006). Genetic Algorithms Applied to Optics and Engineering. *In Proc. of SPIE Fifth Symposium Optics in Industry*. Vol. 6046, No., pp 154-160.

[11] Cuevas F.J., Sossa-Azuela J.H. and Servin M. (2002). A Parametric Method Applied to Phase Recovery From a Fringe Based on a Genetic Algorithm. *Optics Communications*. Vol. 203, No., pp 213-223.

[12] David E. G. (1998). Genetic Algorithms in search, optimization and machine learning. pp 1-25.

[13] David E. G. (1990). A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. *Complex Systems*. Vol., No., pp 445-460.

[14] Davison A. (2003). Real-time simultaneous localisation and mapping with a single camera. *In Proc. of Ninth IEEE International Conference on Computer Vision*. Vol., No., pp. 1403,1410. Vol.2, pp. 13-16.

[15] Derya O. (2006). Feature Selection for Face Recognition Using a Genetic Algorithm. *Department of Computer Engineering*. Bilkent University.

[16] DeSouza G.N. and Kak A.C. (2002). Vision for mobile robot navigation: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 24, No. 2, pp 237-267.

[17] Goedeme T. (2006). Visual Navigation. Katholieke Universiteit Leuven.

[18] Golub G.H. and Van Loan C.F. (1989). Matrix Computations. *Baltimore, Md.: Johns Hopkins Univ. Press.*.

[19] Harris C. and Stephens M. (1988). A combined corner and edge detector. *In Proc. of Fourth Alvey Vision Conference*. Vol., No., pp 147-151.

[20] Hartley R. and Zisserman A. (2000). Multiple view geometry in computer vision. *Cambridge University Press*. ISBN: 0521623049, 2000.

[21] Ido J., Shimizu Y., Matsumoto Y. and Ogasawara T. (2009). Indoor navigation for a humanoid robot using a view sequence. *The International Journal of Robotics Research*. Vol. 28, No. 2, pp 315–325.

[22] Kim S., Kim D. and Kim H. (1996). A Recognition of Vehicle License Plate Using a Genetic Algorithm Based Segmentation. *In Proc. of Third IEEE International Conference on Image Processing*. Vol 2, No., pp 661-664.

[23] López A. (2014). ViCoMex, Soluciones en Visión por Computadora. Plan de Negocios. *Centro de Investigaciones en Óptica*.

[24] Lowe D.G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*. Vol. 60, No. 2, pp 91-110.

[25] Lutton E. and Martinez P. (1994). A Genetic Algorithm for the Detection of 2D Geometric Primitives in Images. *In Proc. of IEEE 12th IAPR International Conference on Pattern Recognition. Conference A: Computer Vision & Image Processing*. Vol. 1, No., pp 526-528.

[26] Matsumoto Y., Ikeda K., Inaba M. and Inoue H. (1999). Visual Navigation using Omnidectional view Sequence. *In Proc. of IEEE/RSJ International Conference on Intelliget Robots and Systems*. Vol. 1, No., pp 317-322.

[27] Michael. O. (2004). Webots: Professional Mobile Robot simulation. *International Journal of Advanced Robotic Systems*. Vol. 1, No. 1, pp 39-42.

[28] Nistér D., Naroditsky O. and Bergen, J. (2004). Visual Odometry. *In Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 1, No., pp 652-659.

[29] OpenCV. (2014). `http://opencv.org/`.

[30] Pretto A., Menegatti E. and Pagello E. (2007). Reliable features matching for humanoid robots. *7th IEEE/RAS International Conference on Humanoid Robots*. Vol., No., pp 532-538.

[31] Pretto A., Menegatti E., Bennewitz M., Burgard W., Pagello E. (2009). A Visual Odometry Framework Robust to Motion Blur. *IEEE International Conference on Robotics and Automation*. Vol., No., pp 2250-2257.

[32] Robert L. (2011). OpenCV 2 Computer Vision Application Programming Cookbook. *BIRMINGHAM - MUMBAI*. ISBN: 9781849513241.

[33] Rosten E. and Drummond T. (2006). Machine learning for high-speed corner detection. *In Proc. of ECCV*. Vol. 3951, No., pp 430-443.

[34] Rublee E., Rabaud V., Konolige K. and Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. *IEEE International Conference on Computer Vision*. Vol., No., pp 2564-2571.

[35] Se S., Lowe D., and Little J. (2001). Local and global localization for mobile robots using visual landmarks. *In Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 1, No., pp 414-420.

[36] Shi J. and Tomasi C. (1994). Good Features to Track. *In Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol., No., pp 593-600.

[37] Song Z. and Klette R. (2013). Robustness of Point Feature Detection. *Lecture Notes in Computer Science. Computer Analysis of Images and Patterns*. Vol. 8048, No., pp 91-99.

[38] S.S. Rao. (1996). Engineering optimization theory and practice. *New York: Wiley*, ISBN: 0-471-55034-5, 3rd. edition.

[39] Torres D. and Cuevas F.J. (2012). Three-Dimensional Point Cloud Registration using a Genetic Algorithm and the Iterative Closest Point Algorithm. *SciTePress Digital Library*. Vol., No., pp 547-552.

[40] Torres D. and Cuevas F.J. (2011). A Genetic Algorithm Applied in the Three-Dimensional Reconstruction of Digitalized Objects. *In Proc. of SPIE Eighth Symposium Optics in Industry*. Vol. 8287, No., pp 870-875.

[41] Van Veen H., Distler H., Braun S., and Bulthoff H. (1998). Navigating through a virtual city: Using virtual reality technology to study human action and perception. *Journal, Future Generation Computer Systems*. Vol. 14. No. 3-4, pp 231-242.

[42] Xu and Z. Zhang. (1996). Epipolar Geometry in stereo, Motion and Object Recognition. *Kluwer Academic Publishers.*