VNIVERSIDAD NACIONAL AVTÓNOMA DE MEXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

ADECUACIÓN DE TÉCNICAS DE DESCRIPCIÓN VISUAL UTILIZANDO INFORMACIÓN 3D Y SU APLICACIÓN EN ROBÓTICA

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA

P R E S E N T A:

ABEL PACHECO ORTEGA

DIRECTOR DE LA TESIS: DR. JESUS SAVAGE CARMONA
DR. WALTERIO W. MAYOL CUEVAS

MÉXICO, D.F. **2011.**

Agradecimientos

Doy gracias:

A mis padres Eva Ortega, Gregorio Pacheco y hermana Magaly Pacheco quienes son siempre sinónimo de apoyo e inspiración.

A Luz Yazmín Brito Hernandez quien me ha acompañado, apoyado y aconsejado como pocos.

A mis tutores Dr. Jesus Savage Carmona y Dr. Walterio W. Mayol Cuevas, que me ayudaron a darle forma a este trabajo con sus aportaciones, consejos y dirección. Así también a los doctores Boris Escalante, María Elena Martínez, Pablo Pérez y Fernando Arámbula por sus observaciones y comentarios sobre este trabajo.

A CONACyT por el financiamiento otorgado.

A todos lo que intervinieron en la realización de esta nueva meta.

Resumen

Los datos obtenidos desde cámaras RGB-D (capaces de brindar información espacial) permiten realizar una extracción de características más estables, éstas son utilizadas para determinar relaciones de localización entre diferentes posiciones en las que se esté observando una misma escena.

En este desarrollo se ha realizado la construcción de un detector que permite identificar puntos de interés con características visuales estables, este método es llamado FAST+3D. La descripción fue implementada de manera que continuará siendo discriminante a pesar de cambios de perspectiva, iluminación, rotaciones y escala; todo a partir de normalizaciones determinadas por la información tridimensional obtenida.

Las relaciones visual y de localización entre dos imágenes dadas es determinada por medio de árboles de busqueda y metodologías de fotogrametría también a partir de la información 3D.

Se ha planetado el uso de las métodologías de detección, descripción y localización desarrolladas en conjunto con la construcción de mapas topológicos para la navegación de robots móviles.

Índice general

Índice de figuras

Capítulo 1

Introducción

El neurocientifico y psicólogo británico David Marr estableció que existen tres niveles en los cuales un dispositivo de percepción debe ser conceptualizado. El primer nivel es aquel en el que se encuentra la teoría computacional del mismo, donde se establece el desempeño en la conversión de un tipo de información a otro y las propiedades abstractas del mapeo se definen de manera precisa. El segundo nivel determina la representación de la entrada, la salida y el algoritmo a ser usado en la transformación de una a otra. En el tercer nivel Marr ubica los detalles de cómo son realizadas las representaciones físicamente.

De manera particular, respecto a la visión Marr define en [Mar82] que "la visión humana es una tarea de procesamiento de información, que describe con las imágenes lo que está presente en el mundo real y dónde se encuentra".

La visión por computadora es una disciplina cuyo objetivo es realizar decisiones a partir de características del mundo real extraídas a través de imágenes digitales del mismo. Dicho con otras palabras, la visión por computadora construye descriptores de

una escena basándose en las características relevantes de una imagen o una secuencia de ellas, de tal manera que la caracterización esté libre de información irrelevante. Estudiar la visión no sólo implica la extracción de información, sino también el almacenamiento de la misma para tenerla disponible como base de conocimiento para futuras tomas de decisiones.

Otro de los desarrollos importantes en la últimas 5 décadas, es la construcción de robots. Según la Real Académica Española un robot es una "máquina o ingenio electrónico programable, capaz de manipular objetos y realizar operaciones antes reservadas sólo a las personas" [Aca01]. Las personas con capacidad visual, saben que de los mecanismos de percepción que el ser humano tiene, el sistema de visión es el más utilizado para las actividades diarias. Así que no es de sorprenderse que los desarrollos en visión computacional tengan un campo de aplicación muy significativo en la robótica.

El uso de robots en la generación de productos comenzó a realizarse de manera incierta en los años 60, pero, poco a poco fue ganando popularidad y ha tenido un incremento considerable en los últimos años. El ejemplo más claro está en la industria automotriz, donde el uso de robots se ha hecho menester para la producción en serie de los mismos. Los robots en la industria permiten que costos y tiempos de producción se vean reducidos en una proporción importante, permitiendo con ello que productos de buena calidad son accesibles a una mayor cantidad de personas; por lo tanto, el desarrollo de tecnologías que puedan ser aplicadas a diferentes gamas de procesos implica un beneficio considerable, más si ésto se hace en países que buscan un impulso en su desarrollo.

Dicho éxito industrial ha sido en gran medida gracias a que los entornos de trabajo son construídos de tal manera que el robot no tenga dificultades al realizar sus actividades; las cuales son repetitivas y ejecutadas en tiempos precisos y coordinados con las demás etapas de producción.

Hoy en día los robots han sido aplicados en una gran cantidad de actividades no relacionadas con la generación de productos (robots industriales), sino en la ejecución de actividades fuera de ella como en medicina, agricultura, limpieza, espacio, construcción, domestico, etc., este tipo de robots son llamados "de servicio".

Realizar las actividades antes mencionadas, implica que el robot se desplace en entornos que no fueron construidos especialmente para la ejecución de las mismas. Esto ha llevado a que uno de los puntos hito en robótica de servicios sea el desarrollo de técnicas con las que el robot extrae información del entorno y la utiliza para ubicarse en él, con lo cual se logra lo que se conoce como *Mapeo* y *Localización*.

Saber la localización de un agente en un ambiente determinado es de suma importancia para el desempeño del mismo. Con este propósito primero se necesita realizar un mapeo del ambiente con el cual se pueda realizar la localización.

Los mapas pueden ser dados como conocimiento *a priori* a los agentes o generados de manera dinámica por los mismos mientras realizan una exploración del ambiente (también llamada *Simultaneous Localization and Mapping*, SLAM). Las construcciones de modelo del ambiente pueden ser clasificadas en modelos métricos y topológicos. En los modelos topológicos la información acerca del ambiente es representada por medio de "posiciones referencia" y las relaciones que existen entre ellas, el agente por tanto tiene tantos sistemas de referencia como posiciones tenga el mapa . En los modelos métricos se tiene información detallada y precisa sobre el ambiente, con ellos la posición del agente es determinada con respecto a un sistema de referencia absoluto.

La decisión entre el uso de modelos topológicos o métricos en la construcción de mapas depende de la tarea que el agente realizará, si ésta requiere de una localización exacta dentro del ambiente es mejor utilizar modelos métricos; si la tarea requiere únicamente

del reconocimiento de lugares específicos y no una precisa ubicación en cada instante de la ejecución, entonces es mejor optar por la utilización de modelos topológicos.

Existen dos tipos de localización, local y global. Las técnicas de localización local se encarga de trabajar y corregir los cálculos hechos sobre la posición del agente, por lo tanto requiere saber la ubicación inicial del mismo. Las técnicas de localización global son más robustas, pueden localizar el robot sin ninguna información previa acerca de su posición, este tipo de técnicas de localización pueden resolver el "problema del secuestro de robot", en donde el robot es colocado en una posición desconocida.

A lo largo del desarrollo de esta línea de investigación se han creado y modificado algoritmos y elementos de percepción para la construcción de mapas y la localización de los agentes. En los primero trabajos se han utilizado lásers y sonares como sensor primario (véase [LSKE11] para detallar al respecto), posteriormente fueron introducidos elementos de percepción visual como cámaras monoculares ([DRMS07, CPMC07, CPMcC06, MCC10]) , estéreo ([ML00, SLL05]) y últimamente cámaras RGB-D ([HKH$^+$10]).

Cuando se usan elementos de percepción visual los inconvenientes presentes en la construcción del mapa y la localización son principalmente: movimientos rápidos, ruido, cambio de intensidades de luz y oclusiones.

En el ámbito académico se están realizando una gran cantidad de desarrollos en robótica, en los cuales participan personas especializadas en diferentes disciplinas, un ejemplo de ello es el Laboratorio de Biorrobótica de la Facultad de Ingeniería (UNAM). Ahí se desarrollan, entre otros, robots de servicios domésticos; algunas de las tareas que realizan dichos robots son: detección de objetos, navegación, reconocimiento de personas, interacción auditiva con seres humanos, etc.

El objetivo de esta tesis es generar una aplicación que realice la extracción de caracte-

rísticas visuales robustas ante cambios de orientación, iluminación, perspectiva y escala, con una buena relación *tiempo de ejecución-resultados*, dicha metodología será utilizada para proponer un sistema que realice el mapeo y localización visual del entorno en el cual se desplaza la cámara.

Para dicho propósito se realizó una división del trabajo de la siguiente manera:

Capítulo 2. Marco teórico. En este capítulo estarán las bases teóricas y los trabajos relacionados con este desarrollo.

Capítulo 3. Arquitectura del sistema. Aquí se presentará la metodología de extracción de características visuales desarrollada.

Capitulo 4. Aplicación en Robótica. En el se mostrará que el uso de las información obtenida con el método de planteado puede ser utilizado para complementar o sustituir el método de localización básica de robots.

Capítulo 5. Pruebas. En este capítulos se presentarán los resultados obtenidos con las metodologías de descripción y localización visual.

Capítulo 6. Conclusiones. Aquí se desarrollarán las conclusiones obtenidas de la pruebas, junto con las debilidades y fortalezas de la aplicación.

Capítulo 2

Marco teórico

En [GW01] definen *imagen* como una función bidimensional, $f(x, y)$, donde x y y son coordenadas espaciales (plano) y la amplitud de f en cualquier par coordenado (x, y) es llamada la intensidad o nivel de gris de la imagen en ese punto. Cuando x, y y f son discretas y finitas, entonces se habla de una *imagen digital*. Las imágenes digitales entonces se encuentran constituídas de un número finito de elementos con una localización y un valor definido. Cada uno de estos elementos es llamado pixel[1].

El tipo de imagen definida anteriormente es referida como *monocromática* o *en escala de grises*. Las imágenes digitales a color son generadas a partir de la unión de imágenes monocromáticas individuales. Estas componentes son también llamadas *bandas*, por ejemplo, una imagen a color en un sistema RGB consiste de tres imágenes monocromáticas del mismo tamaño, donde cada una de ellas representa una componente del

[1]Debido al origen inglés de la palabra se tienen discusiones sobre su correcta escritura en español, al respecto la Real Acedemía Española en [Rea06] indica:
píxel o pixel. La voz inglesa pixel ('elemento más pequeño de los que componen una imagen digital') se ha incorporado al español con dos acentuaciones, ambas válidas. La forma llana *píxel* refleja la pronunciación etimológica —mayoritaria en el conjunto del ámbito hispánico— y debe escribirse con tilde por acabar en consonante distinta de -n o -s. Su plural es píxeles: "El monitor brinda mayor resolución debido a que la pantalla presenta más puntos o píxeles". En algunos países como México se usa exclusivamente la forma aguda pixel (pron. [piksél]), cuyo plural es pixeles.

sistema de color (rojo, verde y azul). En este tipo de imágenes, el pixel puede ser visto como un vector cuya dimensión es igual al número de bandas que conforman la imagen.

Las camaras digitales convencionales realizan una discretización de las intensidades de luz reflejadas sobre una escena, dicho en otra palabras, generan una representación bidimensional discretizada de un mundo tridimensional analógico. La manera usual de representar este proceso es llamado *proyección central* (véase figura 2.1), donde un rayo va desde un punto en el mundo tridimensional hacia un punto en el espacio llamado *centro de proyección* y continua hasta intersecarse con el *plano de la imagen*, en él se hace el promedio de intensidades de acuerdo a un arreglo bidimensional, generándose una imagen digital.

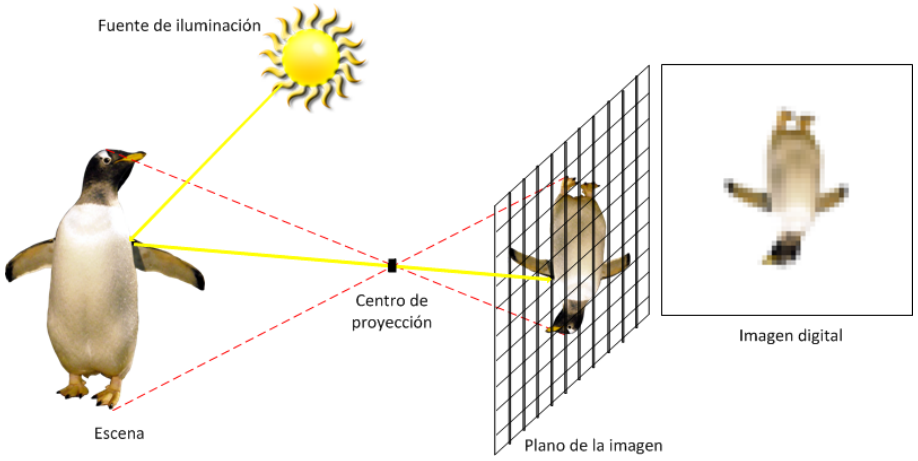


Figura 2.1: Discretización de imágenes

La visión por computadora extrae características de una imagen digital o una secuencia de ellas, estas características son analizadas para formular conclusiones y realizar una toma de decisiones. Muchas han sido las técnicas desarrolladas para la extracción y descripción de características en imágenes digitales, el número de disciplinas aplicadas en el área es bastante grande, abarcando desde la biología hasta la geometría y el álgebra lineal.

En este trabajo se presenta una breve introducción de los conceptos, algoritmos y bibliotecas de funciones utilizadas. De manera particular se realizó una revisión sobre las transformaciones 2D en imágenes digitales, los detectores de características, la manera en que dichos características son descritas, se abordarán las técnicas utilizadas para el empatado de descriptores de dos imágenes diferentes y las implicaciones reportadas que conlleva el uso de alguno de los detectores antes mencionados. También se hablará sobre cámaras RGB-D, la forma en que trabajan y las posibilidades que ellas han abierto en el desarrollo del campo. Al final del capítulo haré mención de las bibliotecas de funciones que serán utilizadas y/o modificadas en el desarrollo de la tesis.

2.1. Transformaciones 2D de imágenes

Las líneas en el plano se representan matemáticamente como $ax + by + c = 0$, donde diferentes valores de a, b y c representan líneas diferentes. Las líneas en un plano pueden ser representadas entonces con el vector $(a, b, c)^T$. Nótese que $(a, b, c)^T$ y $k(a, b, c)^T$, para cualquier $k \neq 0$, son la misma línea, por lo que a estos vectores se les llama vectores homogéneos.

Un punto $p = (x, y)^T$ en un plano se encuentra en una línea $l = (a, b, c)^T$ únicamente si cumple con la ecuación $ax + by + c = 0$, que en términos de producto interno de vectores se escribe como $(x, y, 1)\cdot(a, b, c)^T$; entonces, un punto en $\mathbb{R}^2$ puede ser representado como un vector de tres elementos agregándole un 1 más. Esta representación también es llamada *punto en coordenadas homogéneas*. Así, $(x, y, 1)^T$ y $(kx, ky, k)^T$ representan el mismo punto y un vector homogéneo (r, s, t) representativo de un punto, en realidad mapea el punto $(r/t, s/t)$ en $\mathbb{R}^2$. Los puntos presentados como vectores homogéneos pertenecen al conjunto P^2.

Teorema 1. *Un mapeo* $h: P^2 \rightarrow P^2$ *es una proyección si, y solo si, existe una matriz singular de 3x3,* H*, tal que para cualquier punto dado en* P^2*, representado por un vector* x*, es cierto que* $h(x) = H(x)$*.*

En muchas de las aplicaciones de visión por computadora se pretende encontrar la relación entre dos imágenes en las que está presente la misma escena, pero, tomadas bajo diferentes condiciones. Estas dos imágenes tienen diferentes aspectos que, en términos generales, son provocados por deformaciones, cambios de posición u orientación ya sea de la escena o la cámara. Los cambios de este tipo pueden ser traducidos como transformaciones de la imagen. Estas transformaciones son llamadas *homografías*. Una homografía, en términos generales, es una proyección de un plano sobre otro plano.

El proceso de estimación de una homografía es definido por Hartley y Zisserman en [HZ03] como: *Dado un conjunto de puntos* x_i *en* P^2 *y un correspondiente conjunto de puntos* $x_i^{'}$ *también en* P^2*, calcule la transformación que relaciona cada* x_i *con* $x_i^{'}$. En el caso que compete este desarrollo, los puntos x_i y $x_i^{'}$ son puntos en dos imágenes, que están relacionados por la matriz de transformación presentada en la ecuación 2.1.

$$x' = \mathbf{H}x \tag{2.1}$$

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

Sin pérdida de generalidad es posible fijar el valor h_{33} en 1. Por ejemplo, las figuras 2.2(a)

y 2.2(b) se encuentran relacionadas por la homografía $H = \begin{bmatrix} 1.2945 & -0.2024 & 225.28 \\ 0.4584 & 0.9493 & 0 \\ 0.0009 & 0 & 1 \end{bmatrix}$.

(a) (b)

Figura 2.2: Homografía.

2.2. Extractores de características

Encontrar similitudes entre dos imágenes diferentes es un problema común en visión por computadora, para cumplir este propósito, de las imágenes a comparar, se extraen características que en pasos posteriores pueden ser comparadas.

Las características extraídas pueden ser globales o locales. Las globales son una representación de la imagen completa, mientras que las locales lo son de regiones específicas de la misma. Las características locales son mejores si cumplen con ser localizadas, significativas y robustas; lo que significa que tienen una ubicación relacionada con áreas específicas y que son detectadas a pesar de cambios en escala, orientación y/o iluminación.

Una vez se tienen ubicados los puntos de interés de una imagen es importante construir vectores que los representen para poder realizar comparaciones con otros conjuntos de puntos extraídos desde diferentes imágenes, a estos vectores se les llama descriptores.

En los siguientes puntos mostraré algunos de los detectores y los descriptores más importantes que han sido desarrollados.

2.2.1. Detectores de puntos de interés

El primero de los pasos para la extracción de características locales es determinar la ubicación de aquellos pixeles que dada su ubicación y las características de la zona de la imagen en que se encuentran son fáciles de ubicar y sus características están bien determinadas. Muchos han sido los detectores creados, cada uno de ellos hace uso de operadores diferentes para determinar la existencia de puntos de interés; la evaluación de algunos de ellos puede ser consultada en [SMB00, MGBR08, MM06]. Los métodos de detección más utilizados se presentan a continuación.

Harris: Detector presentado en [HS88], basado en el calculo de los eigenvalores λ_1 y λ_2 de la matriz

$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{bmatrix} \tag{2.2}$$

donde I_x e I_y son gradientes de la imagen en las direcciones horizontal y vertical respectivamente[2] y la sumatoria se hace sobre una región cuadrada alrededor de cada posición (x, y). Dada la función de autocorrelación $R = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)$, el punto (x, y) asociado es seleccionado como esquina si el valor de R es positivo y localmente máximo; los valores asignados al parametro k en la literatura se fijan entre 0.04 y 0.15.

Shi-Tomasi: Este detector fue presentado en [ST93], está basado en el detector de esquinas de Harris. La función que determina la presencia de un punto de interés (esquina)

[2]Los gradientes en el punto (x, y) se calculan como $I_x = \sqrt{(i(x+1, y) - i(x-1, y))^2}$, $I_y = \sqrt{(i(x, y+1) - i(x, y-1))^2}$ donde $i(u, v)$ representa la intensidad de la imagen en el punto (u, v)

está dada por $R = min(\lambda_1, \lambda_2)$. Si R es mayor que una valor predefinido, entonces el punto (x, y) asociado es considerado como punto de interés.

Detector SIFT: El algoritmo *Scale Invariant Feature Transform* (SIFT), realiza la detección y descripción de puntos de interés en imágenes. Se trata de uno de los métodos más utilizados, su nombre es debido a que transforma una imagen en descriptores invariantes a escala, rotación y parcialmente invariante a cambios de iluminación y punto de vista. Este algoritmo fue presentado en [Low04].

En la primera de las etapas el algoritmo identifica la posición y escala de puntos candidatos (puntos máxima). Utiliza una aproximación a el *Laplaciano de Gaussiana* (*LoG*), que en [Lin94] y [MS02] demostró ser un buen detector de puntos invariantes a escala, mediante la *Diferencia de Gaussianas* (*DoG*) aplicada en el espacio-escala de la imagen. El espacio-escala (ecuación 2.4) es una familia de imágenes asociadas que se construye a partir de un suavizado progresivo de la imagen original, el cual se logra a partir de la convolución de un filtro gaussiano (ecuación 2.3) con la imagen.

$$g(x, y; \sigma) = \frac{1}{2\pi t} \exp^{-\frac{x^2+y^2}{2\sigma}} \tag{2.3}$$

$$L(x, y; \sigma) = g(x, y; \sigma) * I(x, y) \tag{2.4}$$

donde I es la imagen a partir de la cual se construirá el espacio escala y σ es el parametro que será modificado progresivamente, pondera el suavizado de la imagen.

Los puntos candidato son aquellos que se encuentran bien identificados entre DoG adyacentes, es decir tienen un valor máximo o mínimo con respecto a los 26 puntos alrededor del mismo en la estructura generada (véase figura. 2.3b).

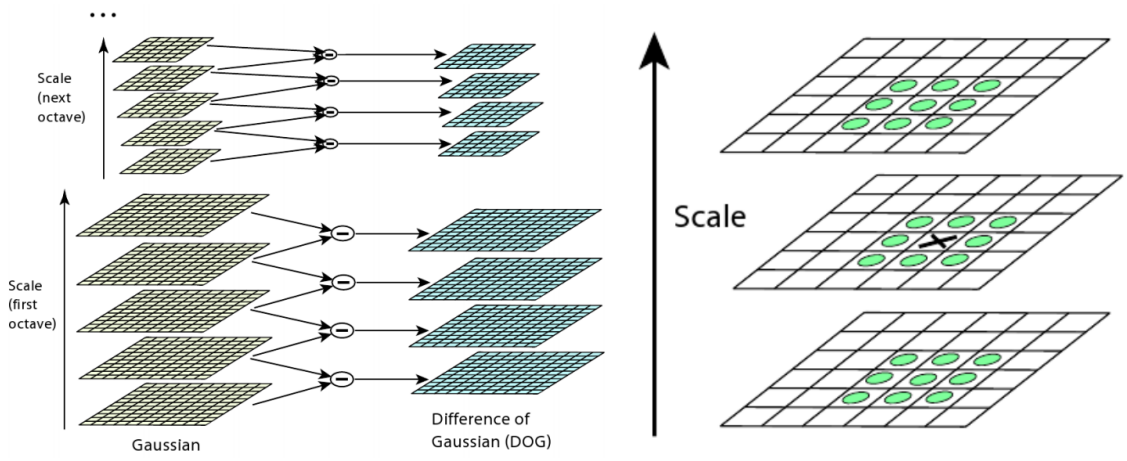


Figura 2.3: Análisis espacio-escala. (a) Diferencia de gaussianas aplicada a espacio-escala. (b) Detección de máximos y mínimos locales.[Low04].

Una vez encontrados los puntos candidato en cada escala, se hace un filtrado de los mismos para eliminar puntos en regiones de bajo contraste y que no sean esquinas, pues no destacan ante cambios de iluminación pequeños y presencia de ruido. SIFT también realiza el calculo de la matriz presentada en la ecuación 2.2 y sus eigenvalores λ_1 y λ_2, se considera el punto finalmente

$$\frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \lambda_2} < \frac{(r+1)^2}{r} \tag{2.5}$$

siendo r un umbral cuyo valor más común es 10.

Detector FAST: *Features from Accelerated Segmented Test (FAST)* es un detector de esquinas presentado en [RD05], cuyo criterio de selección de un pixel p dentro de una imagen esta basado en el análisis de 16 pixeles (circulo de radio 3) alrededor del punto. Siendo I_p el valor de intensidad del punto p, éste es considerado como esquina si existen un conjunto de n pixeles contiguos en el perímetro de un circulo de radio 3 alrededor del mismo que tengan una intensidad superior a $I_p + t$ o inferior a $I_p - t$, donde t es

un umbral determinado (véase figura 2.4). Generalmente n tiene valores entre 9 y 12, para eliminar puntos candidato de manera mucho más rápida es realizado un chequeo preliminar en los pixeles 1, 9 , 5 y 13. La rápida eliminación de puntos candidatos hace que FAST tenga un gran rendimiento. Análisis en las relaciones de rendimiento con las variaciones en los valores t y n pueden ser encontrados en [RD05] y [RD06].

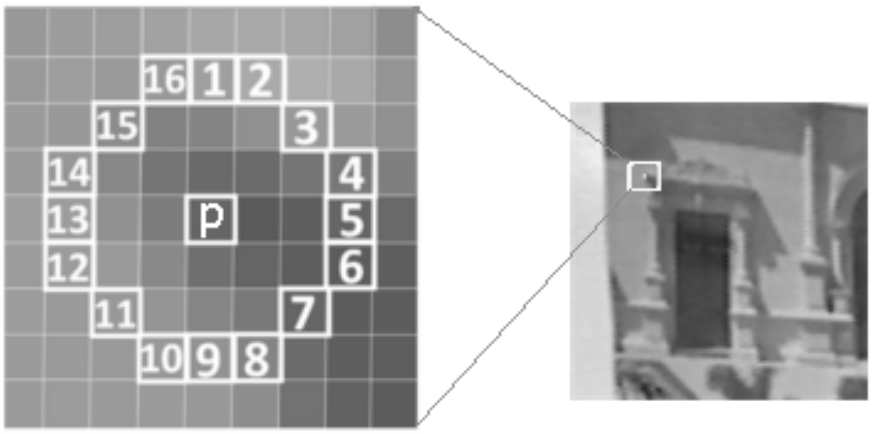


Figura 2.4: Prueba FAST en el punto p

Sin embargo FAST detecta como esquinas a muchos puntos adyacentes, por ello se necesita de una supresión de no máximos donde la puntuación de un punto p dado es la sumatoria de las diferencias de cada pixel contiguo con los que fue superada la detección FAST sin supresión.

2.2.2. Descriptores

Hasta este punto lo que tenemos es la localización de los puntos de interés dentro de una imagen; sin embargo lo que se desea es realizar la asociación de cada uno de estos puntos a atributos que típicamente se encuentran en las variaciones de los pixeles alrededor de él. Al igual que con los detectores, muchas han sido las metodologías

aplicadas para realizar la descripción de cada ubicación en el conjunto L arrojado por algunos de los detectores antes mencionados. En los siguiente párrafos mencionaré las principales características de los metodos de descripción más utilizados. Evaluaciones sobre el desempeño de los mismos pueden ser consultadas en [MS05, GMBR10].

Escala de grises de parche: Este descriptor es el más simple, para construirlo solo se toman los valores de intensidad de los pixeles en una región de nxn alrededor del punto de interés a describir.

Descriptor SIFT: El descriptor generado por SIFT esta basado en un análisis de gradientes y orientación de la regiones alrededor de los puntos de interés. El tamaño de la región a analizar alrededor de cada punto depende de la escala a la cual fue detectado y cuanto mayor sea la escala mayor será la región de análisis. Las magnitudes (m) y orientaciones (θ) de los descriptores en cada pixel son calculadas como:

$$m(x,y) = \sqrt{\left(L\left(x+1,y\right) - L\left(x-1,y\right)\right)^2 + \left(L\left(x.y+1\right) - L\left(x,y-1\right)\right)^2} \tag{2.6}$$

$$\theta(x,y) = \tan^{-1}\frac{L\left(x.y+1\right) - L\left(x,y-1\right)}{L\left(x+1,y\right) - L\left(x-1,y\right)}$$

Después de realizar el calculo de magnitud y orientación en la region alrededor de cada punto de interés, se genera un histograma con 36 contenedores, aquellas direcciones que tengan un 80 % de la magnitud máxima serán consideradas también orientaciones representativas del punto de interés y con ellas se formarán nuevos descriptores, es decir, un mismo punto de interés puede tener uno o varios descriptores asociados. Las orientaciones de los puntos de interés son interpoladas con los contenederos vecinos para obtener una mejor exactitud.

Finalmente, para construir el descriptor se analiza la región que rodea al máximo o mínimo local detectado. Para dicho descriptor este vecindario es dividido 4 regiones de

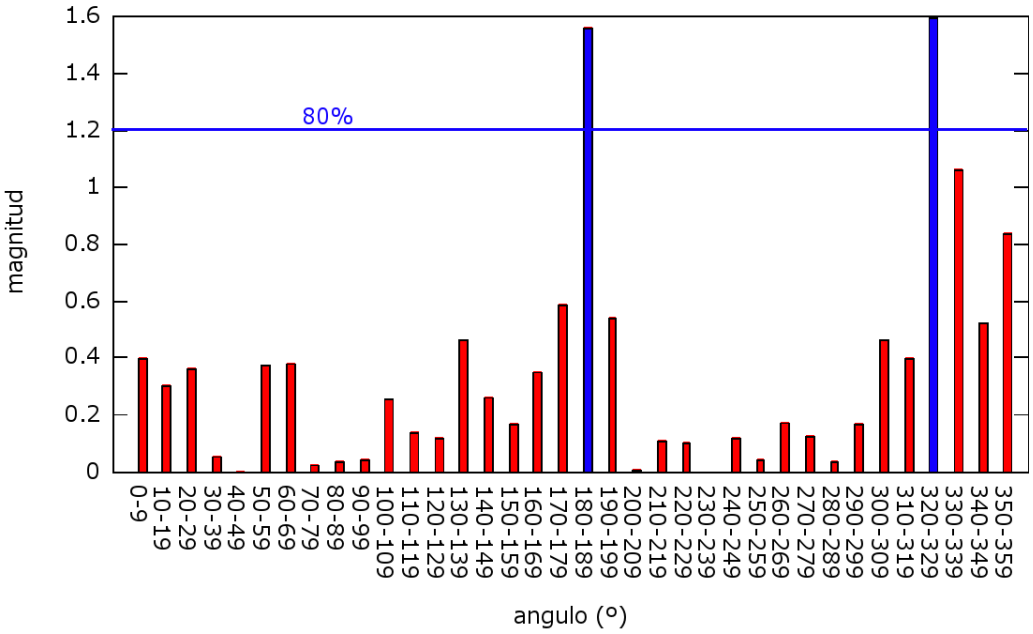


Figura 2.5: Calculo de orientación de descriptor. Histograma de gradientes.

4x4 zonas cada uno donde se genera un historial de orientaciones relativas a la del punto de interés (8 direcciones para cada zona), teniéndose en total 4x4x8=128 elementos en el descriptor (véase figura. 2.6) .

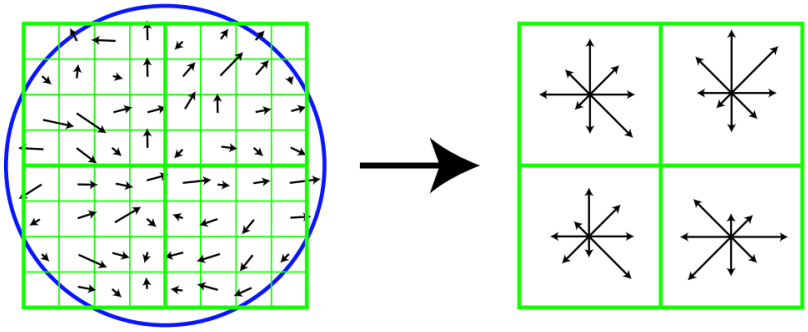


Figura 2.6: Generación del descriptor SIFT. Solo se muestra una división de 2x2 regiones, pero SIFT utiliza en realidad una división de 4x4, [Low04].

SIFT es un algoritmo que nos brinda características muy estables, pero el costo computacional es más significativo que el que otros algoritmos nos pueden brindar. Ejemplos de la salida del algoritmo SIFT se encuentra en la figura 2.7.

Figura 2.7: Ejemplos de algoritmo SIFT, escena observada (izquierda) y descriptores calculados (derecha)

GLOH: el *Gradient location-orientation histogram,* presentado en [MS05], es una mo-

dificación al descriptor SIFT realizada para incrementar la robustez del mismo. La diferencia radica en que este descriptor es construido a partir de generar una división regiones de tipo log-polar alrededor del punto de interés, tiene tres contenedores en dirección radial y 8 en dirección angular (véase figura 2.8). Con ello se obtiene un conjunto descriptores de dimensión 272, el cual es interpretado por medio de una técnica de análisis multivariado llamada Análisis de Componentes Principales (PCA[3], por sus siglas en ingles) para reducirlo a las 128 dimensiones más significativas.

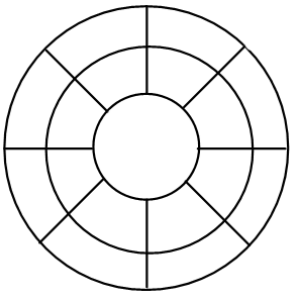


Figura 2.8: División de regiones en GLOH.

PCA-SIFT: Este detector fue presentado en [KS04], en él después de determinadar la posición, escala y orientación de un punto de interés, como en SIFT, se realiza un análisis de gradiente en una región de 41x41 adecuada para la escala, se gira la ventana según la orientación dominante y se forma un vector que concatena los mapas de gradientes en dirección horizontal y vertical (se formarán tantos vectores como orientaciones dominantes tenga el punto de interés); la dimensión de los vectores obtenidos es 2x39x39 = 3042. Dicha dimensión del conjunto de vectores es reducida hasta bajas dimensiones (16,20,28,36,etc.) por medio de los eigenvectores y eigenvalores arrojados por un Análisis de Componentes Principales previamente realizado sobre una base de 21 000 vectores calculados en fase de entrenamiento.

[3]Para mayor detalle sobré Análisis de Componentes Principales véase [Jol02]

2.3. Empatado de descriptores

El conjunto de descriptores F_O de una imagen patrón O puede ser usado como base de conocimiento para determinar existencias en otro conjunto de descriptores F_I de una imagen I dada. Puesto que los descriptores asociados a las características de la imagen son vectores, la forma más sencilla de realizar comparaciones entre dos descriptores de imágenes diferentes es la distancia euclidiana.

$$d(f_{O,i}, f_{I,j}) = \sqrt{\sum_{k=1}^{n} (f_{O,i.k} - f_{I,j,k})^2} \tag{2.7}$$

donde:

n: dimensión del vector

$f_{O,i.k}$,$f_{I,j,k}$ son los valores del vector $f_{O,i}$ y $f_{I,j}$ en la posición k

Con esta medida de similitud es posible determinar que el punto x con descriptor $f_{O,i}$ de la imagen O se encuentra en la imagen I si el descriptor $f_{I,j}$ con menor distancia a $f_{O,i}$, es menor que un umbral establecido. Aquellos descriptores que superen el umbral son almacenados en el conjunto $M = \{m_1, m_2, ..., m_N\}$ donde cada elemento tiene la forma $m_i = \{f_{O,i}, f_{I,j}\}$.

La dimensión de los descriptores que genera SIFT es 128. El problema con el manejo de alta dimensionalidad es que la busqueda del descriptor con menor distancia implica que cada uno de los descriptores en F_O sea comparado con cada descriptor en F_I, luego entonces el proceso de empatado básico necesita una cantidad importante de operaciones que lo vuelven muy ineficiente. Para resolver el inconveniente planteado se hace uso de árboles de búsqueda.

2.3.1. Árbol de búsqueda k-d

Un árbol de búsqueda k-d es una estructura de datos de partición de espacio para organizar puntos en un espacio de k dimensiones presentada en [Ben75], es muy útil cuando se necesita realizar búsquedas multidimensionales. Un árbol de busqueda k-d divide y organiza recursivamente un espacio en conjuntos utilizando hiperplanos, lo que se logra con ello es realizar una división del conjunto vectorial en conjuntos cada vez más reducidos.

El algoritmo 2.1 muestra el método de construcción del árbol de un conjunto de vectores L.

Algorithm 2.1 Construcción de árbol k-d

1. $d = 1$
2. Determinar el vector v con la mediana m del conjunto L en la dimensión d
3. Dividir el conjunto con respecto al vector v en los conjuntos LI y LD con valores en la dimensión d menores y mayores que m respectivamente
4. $d++$
5. Realizar desde el punto 2 si el conjunto LI tiene más de dos componentes
6. Realizar desde el punto 2 si el conjunto LD tiene más de dos componentes

Realizar la búsqueda de un vector x en el árbol es una operación de orden $O(log\, n)$, donde n es el número de vectores en la estructura, pues cada nivel del árbol realiza una división binaria del conjunto (conjuntos LI y LD del algoritmo 2.1), en la busqueda no se examinan todos los vectores, pues se discriminan aquellos que pertenecen a conjuntos que se encuentran a una distancia mayor que una supuesta mínima ya encontrada.

Con vectores de muchas dimensiones su desempeño se ve significativamente reducido,

por ello se tiene una construcción alternativa que habilita uso de la busqueda *Best Bin First* (véase Sección 2.3.1.1) .

2.3.1.1. Algoritmo de busqueda Best Bin First

Se trata de un algoritmo utilizado para búsquedas en conjuntos de muchas dimensiones presentado en [BL97]. Para su utilización se necesita de una variante del árbol de busqueda k-d mejorada para conjuntos de altas dimensiones, a ésta me referiré como *árbol BBF*.

El algoritmo 2.2 muestra el método de construcción del árbol BBF a partir de un conjunto de vectores L.

Algorithm 2.2 Construcción de árbol k-d para busqueda *Best Bin First*

1. Si el conjunto L tiene 2 o más componentes.
   - *a*) Encontrar la dimensión d del conjunto de vectores con mayor varianza
   - *b*) Determinar la media m de la dimensión d del conjunto de vectores
   - *c*) Dividir el conjunto con respecto m en la dimensión d en los conjuntos LI con valores menores y LD con valores mayores
   - *d*) Realizar desde el punto 1 con el conjunto LI
   - *e*) Realizar desde el punto 1 con el conjunto LD
2. Si no, finalizar

El algoritmo 2.3 muestra como se realiza la búsqueda *Best Bin First* con un vector v. Las búsquedas se realizan por proximidad espacial y es fijado límite de puntos visitados N_{max}.

La figura 2.9 muestra el resultado de realizar un empatado entre dos imágenes con árbol BBF. Como primer paso fueron obtenidos los descriptores SIFT de ambas imágenes,

Algorithm 2.3 Busqueda *Best Bin First*

1. $n = 0$ *(número de puntos visitados)* , $d_{min} = \infty$ *(distancia mínima encontrada)*
2. $n++$
3. Si $n > N_{max}$ finalizamos la búsqueda y verificamos el vector asociados a d_{min}
4. Determinar el subconjunto T al cual pertenece el vector v buscado
5. Memorizar distancia y referencia a la rama que no será accesada en esta iteración en *cúmulo (vector ordenado por distancias)*
6. Acceder a la rama de T
7. Si se trata de divisor de subconjunto
   a) realizar desde el punto 2.
8. Si no
   a) medir distancias entre vectores d
   b) si $d < d_{min}$
      1) $d_{min} = d$ y almacenar la referencia del vector asociado
9. Acceder a la primera referencia en *cúmulo* y ejecutar desde 1.

posteriormente, con el conjunto de descriptores de la imagen de la izquierda fue generado un árbol BBF, finalmente, la búsqueda de cada descriptor de la imagen derecha se realizó en el árbol construido. Los puntos amarillos marcan las ubicaciones de los descriptores SIFT de cada imagen; en color cian y rojo se encuentran enlazados aquellos descriptores empatados.

2.3.2. Empatados atípicos

Nótese que en la figura 2.9 existen correspondencias equivocadas (líneas rojas). Este tipo de correspondencias atípicas aparece cuando las regiones de los dos puntos tienen

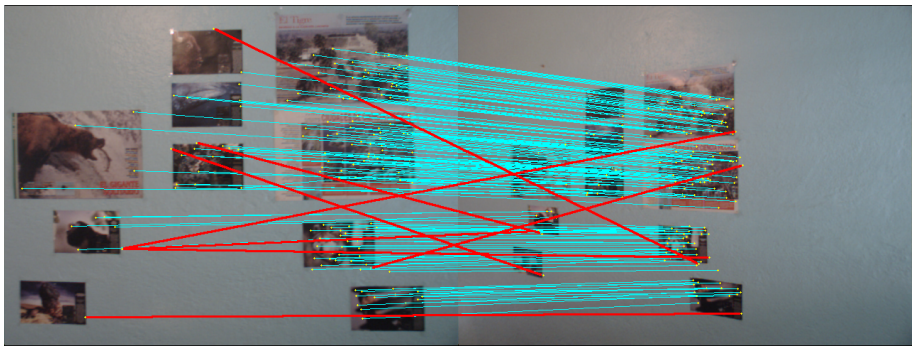


Figura 2.9: Empatado con busqueda *Best Bin First*

descriptores muy parecidos a pesar de no tener correspondencia real en la escena observada. La transformación de homografía es el modelo de la relación mas simple que existe entre dos imágenes de la misma escena. Realizar una estimación de la misma (ecuación 2.1) permitiría eliminar aquellos empatados atípicos.

En procedimientos clásicos la estimación se realiza con el ajuste de la descripción que mejor represente todos los datos, sin embargo, en ellos no se tienen mecanismos internos para eliminación de errores atípicos, además, están basados en el supuesto de que la cantidad de información a ajustar es considerablemente grande, de tal manera que la aparición de errores graves no perturba en gran medida dicho ajuste.

En el caso de empatado de descriptores estos errores son muy comunes. En el ejemplo mostrado en la figura 2.9 fueron empatados 135 pares de descriptores y 7 de ellos eran atípicos, es decir, 5 % de las correspondencias eran datos espurios. Realizar la estimación directa de la matriz de homografía acarrearía un gran error acumulado al sustituir los puntos en la ecuación 2.1. En la siguiente sección es presentado uno de los métodos de estimación más utilizados en visión, robusto ante conjuntos con un gran porcentaje de datos que no pertenecen al modelo a estimar.

2.3.2.1. *Random Sample Consensus* (RANSAC)

El método RANSAC, presentado en [FB81], usa un conjunto inicial de datos tan pequeño como sea posible, con él realiza la estimación de parámetros de un modelo dado y lo hace cada vez más grande utilizando los datos que son consistentes. La operación descrita es realizada en varias ocasiones y se determina que la mejor estimación realizada es aquella en la que el conjunto de datos consistentes es mayor.

El algoritmo 2.4 muestra los pasos para realizar una estimación sin tomar en cuenta datos anómalos por medio de *RANSAC*.

Algorithm 2.4 Random Sample Consensus (RANSAC)

Dado un modelo que puede ser estimado con mínimo n datos y un conjunto de datos P donde $n \geq \#(P)$

1. Repetir N veces:

   a) Escoger de manera aleatoria un subconjunto S de n elementos de P

   b) Determinar los parámetros del modelo que se ajustan a S.

   c) Generar el subconjunto de datos $S*$ que contenga todos aquellos datos compatibles con el modelo calculado con un margen de error e. $S*$ es llamado el consenso de S.

   d) Si $\#(S^*)$ es mayor que un valor r, calculado a partir de una estimación del número datos anomalos esperados. Utilizar $S*$ para realizar una nueva estimación del modelo y de error de ajuste.

   e) Si no, realizar la siguiente iteración

2. Seleccionar el modelo con el menor error de ajuste.

El número de repeticiones **N** se calcula a partir de la ecuación 2.8

$$N = \frac{log(1-z)}{log(1-w^n)} \tag{2.8}$$

donde

w proporción de datos que empatan con el modelo con una tolerancia de error **e**

z probabilidad de escoger un conjunto S de n elementos sin datos anomalos

El margen de error e para datos que empatan con el modelo estimado y el número mínimo de ellos que validan el modelo r**,** son parámetros del algoritmo y su valor depende de la aplicación que se esté realizando; el número N de estimaciones a realizar depende de la proporción de datos w y la probabilidad de éxito p escogidas.

Aplicar este método de estimación para determinar la transformación de homografía entre dos imágenes de una misma escena permite además obtener el subconjunto de descriptores empatados que excluye a aquellos atípicos. Obsérvese la figura 2.10, que muestra el resultado de utilizar RANSAC para eliminar empatados anomalos a partir de la estimación de la transformación de homografía sobre el conjunto de datos mostrados en la figura 2.9.

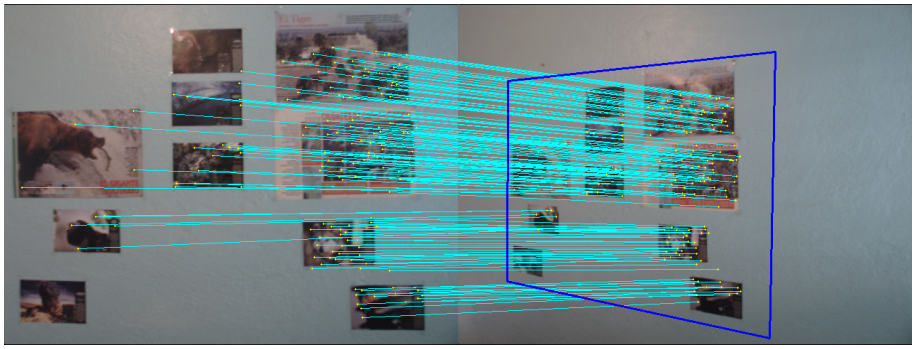


Figura 2.10: Empatado depurado con RANSAC

2.4. Bibliotecas libres: OpenCV, Open-Source SIFT Library, OpenNI

El desarrollo de los últimos años en el área de procesamiento de imágenes y visión por computadora ha motivado el surgimiento de bibliotecas de desarrollo gratuitas que permiten tener una base solida a partir de la cual realizar nuevos desarrollos. A continuación presento aquellas que utilizaré en el desarrollo de esta tesis.

Open Source Computer Vision (OpenCV, http://opencv.willowgarage.com) es una biblioteca de funciones escritas en C++, C y Python, desarrollada originalmente por Intel, que permite realizar una variedad de operaciones relacionadas con imágenes como captura, análisis, modificaciones, almacenamiento y manejos matriciales en general.

La *Open-Source SIFT Library* presentada en [Hes10] es una biblioteca desarrollada en C que permite realizar el cálculo de las características SIFT de una imagen. Esta biblioteca fue encapsulada en una clase C++ para su uso en este lenguaje.

Open Natural Interaction (OpenNI, http:\\http://www.openni.org/) es un *framework* multilenguaje disponible para diferentes plataformas que define una biblioteca de funciones estándar que se encargan de las comunicaciones con sensores de audio y video. La principal característica de este *framework* es su capacidad para extraer información de cámaras RGB-D, como las desarrolladas por la compañía *Primesense* o el sensor Kinect de Microsoft que esta basado en ellas.

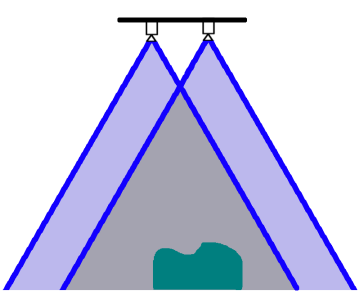
Capítulo 3

Arquitectura del sistema

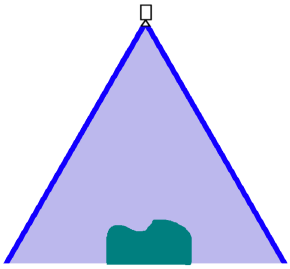
Los desarrollos tecnológicos tienen un gran impacto en el desarrollo de la visión por computadora, el aumento en la rapidez con la que las computadoras realizan calculos y las nuevas metodologías para el análisis de datos han permitido realizar tareas que en otros momentos tenían características cuyo desempeño era muy lento al grado de ser en ocasiones prohibitivo para su ejecución.

Uno de los principales avances es la aparición de cámaras capaces de obtener información visual, ya sea a color o en escala de grises, y la profundidad a la que se encuentran los pixeles en cada una de las tomas que la cámara realiza. Algunas de las tecnologías con las que trabajan este tipo de cámaras son: visión estereoscópica, *time of flight* y las basadas en luz estructurada; detalles y ejemplos de ellas los presento a continuación (véase figura 3.1).

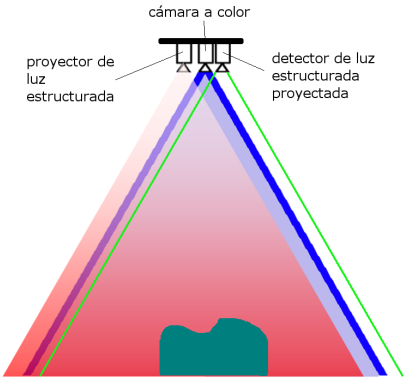
Cámaras estereoscópicas: Tratan de emular la manera en que los seres humanos percibimos imágenes, éste tipo de cámaras están compuestas de dos objetivos. Capturando imágenes desde distintas posiciones y con las relaciones que existen entre ambas



(a) Cámara estereoscópica



(b) Cámara *Time of Flight* (ToF)



(c) Cámara 3D a base de luz estructurada infrarroja

Figura 3.1: Tecnologías para obtención de imágenes con información 3D

(matrices obtenidas previa calibración, texturas de la escena) son capaces de determinar la profundidad de pixeles que se presentan en ambas imágenes.

La cámara estereoscópica STH-DCSG-VAR de *Videre* tiene una separación variable entre sus objetivos (de 5 a 20 cm) que le permite una mayor flexibilidad de configuraciones, captura imágenes a color a una velocidad de 30 cuadros por segundo, con una buena sincronización y baja cantidad de ruido, lo que la hace un dispositivo con alto rendimiento y bajo consumo de energía. El costo de esta cámara es de 1500 dolares.

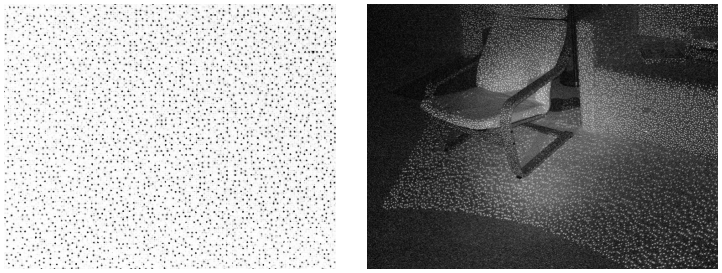
Cámaras Time-of-Flight (ToF) : Estás cámaras calculan la distancia a la que se encuentra cada pixel en la imagen midiendo el tiempo de retardo de una señal de luz lanzada desde la cámara al objetivo. Este tipo de cámaras aparecieron en el mercado en el año 2004 y generalmente tienen una baja resolución.

La SwissRanger 4000 (SR4000) de *Mesa* es una de las cámaras ToF más conocidas. Trabaja en el rango de distancia de 10 centímetros a 5 metros, con una resolución de 176 x 144 pixels, un campo de visión horizontal de 43.6° y vertical de 34.6°, operando a 54 cuadros por segundo. El costo de esta cámara es de 9000 dolares.

Cámaras basadas en luz estructurada: Se basan en la proyección de un patrón lineal sobre la escena a observar, las deformaciones que este patrón tenga en la escena determinan la profundida de los elementos que componen la imagen. Están compuestas por un proyector de luz estructurada, un detector especial para la luz proyectada y una cámara digital normal.

La Primesensor de *Primesense* proyecta un patrón infrarrojo conocido en la escena (véase figura 3.2), que permite determinar medidas de profundida de cada pixel en una imagen a color con el análisis de la deformación de dicho patrón (cámara que detecta luz infrarroja). La resolución de esta cámara es de 640x480 pixeles a 30 cuadros por segundo, con un campo de visión de horizontal de 57^{o}, vertical de 43^{o} y un rango de distancia de 40 centímetros a 6 metros. Las cámaras Kinect de Microsoft se encuentran basadas en este dispositivo, el costo es de 150 dolares.

Las cámaras RGB-D brindan información de profundidad que puede ser analizada para determinar no solo la distancia a la que se encuentra los elementos de la imagen, sino también la posición en coordenadas tridimensionales con referencia a la posición



(a) Patrón (b) Deformación sobre escena

Figura 3.2: Patrón usado por cámara Kinect

de la cámara. OpenNI permite realizar la extracción de esta información de manera automática en cámaras basada en la tecnología de proyección de luz estructurada. Esta información es utilizada en este trabajo para adecuar técnicas de descripción visual ya conocídas en visión computacional de manera que les brinde mayor robustez y/o rapidez en su ejecución. La cámara utilizada para el desarrollo de este trabajo es la Kinect.

El proceso de descripción de imágenes utilizando información 3D propuesto puede ser observado en la figura 3.3. En las siguientes secciones describiré cada uno de los pasos que lo constituyen.

Figura 3.3: Proceso de descripción propuesto

3.1. *Features from Accelerated Segmented Test* (FAST) con análisis de vecindario en 3D

Como he descrito en la sección 2.2.1, FAST es un detector de esquinas que examina 16 pixeles alrededor del punto candidato. El número de puntos extraídos depende de la

resolución, el umbral de diferencia de intensidades y la escena que está siendo analizada. Sin embargo debido a que FAST trabaja únicamente con información de imágenes 2D algunos de los puntos detectados se encuentran en regiones inestables, como aquellas que se encuentran en los contornos de los objetos donde un cambio de posición desde la que es muestreada la escena implica un cambio significativo del descriptor de la región donde se ubica el punto de interés (véase figura 3.5b).

El uso de la cámara RGB-D ha permitido agregar una etapa más de filtrado a FAST para obtener putos de interés aún más estables, esta versión modificada será llamada de aquí en adelante FAST+3D. En los procesos de descripción visual presentado en el capitulo anterior, el objetivo de los detectores es determinar los puntos de interés con un vecindario que cumpla con características estables. En SIFT el tamaño del vecindario a analizar es determinado por la escala a la que los puntos de interés son ubicados en el análisis espacio-escala, es decir, la escala es determinada por las propiedades naturales de la imagen y no de la escena. En este desarrollo la información de profundidad brindada por el sensor RGB-D es utilizada para determinar la escala (en escena) asociada a la región alrededor de un determinado pixel en la imagen.

Para establecer la relación escala-profundidad se hizo uso de de un objeto de dimensiones conocidas como entrada. El objeto es colocado a diferentes profundidades desde la cámara e imagenes de este son capturadas. Utilizando esta información se ha obtenido una relación lineal, que en nuestro caso es:

$$y = 0.001716x + 0.2735 \tag{3.1}$$

donde y (mm/pixel) representa la dimensión del objeto a una profundidad x (mm).

Por lo tanto, determinar el factor de escala de una imagen referencia a una determinada

profundidad esta dado por:

$$escala = \frac{0.001716 * depth_0 + 0,2735}{0.001716 * depth_i + 0,2735} \tag{3.2}$$

donde $depth_0$ es la profundida referencia; $depth_i$ es la profundidad a la que se busca el factor de escala.

Gracias al factor de escala es posible realizar una análisis 3D sobre regiones proporcionalmente "equivalentes", sobre puntos de interés detectados. Se determinó experimentalmente que el vecindario patrón a utilizar tiene un tamaño (w_0) de 25 pixeles por lado a una profundidad de 1700 milímetros. Sustituyendo en la ecuación 3.2 se tiene que las escalas estarán determinadas por $escala = \frac{0.001716*1700+0,2735}{0.001716*depth_i+0,2735}$, siendo $depth_i$ la profundidad a la que se encuentra en punto de interés (véase figura 3.4); el tamaño del vecindario está dado por:

$$w_i = int(w_0 * scale + 0.5) \tag{3.3}$$

$$w_i = int(25 * scale + 0.5) \tag{3.4}$$

Chekhlov *et al.* en [CPMcC06] han demostrado que predecir la escala a la que se encontrará el punto característico incrementa la eficiencia en etapa de empatado (busqueda) pues se evita el gasto computacional de calcular el espacio-escala en cada imagen de la escena capturada. Las cámaras RGB-D y el análisis profundidad-escala descrito permite determinar el tamaño del vecindario a analizar y evita realizar predicciones.

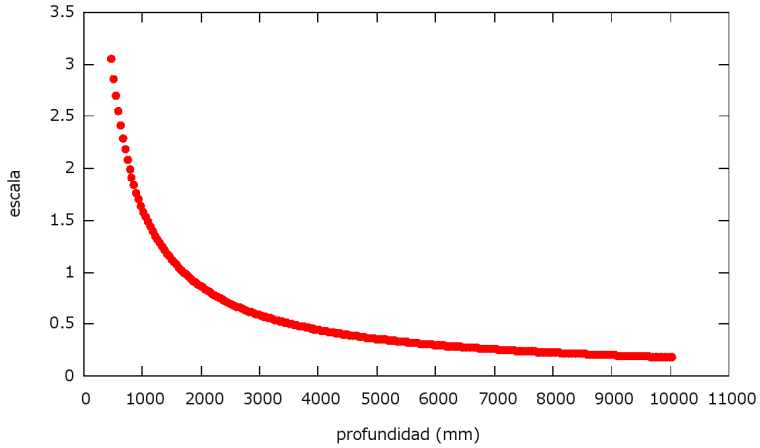


Figura 3.4: Cambio de escala con respecto a profundidad de pixel medida

El conjunto de puntos detectados por FAST+3D está definido como

$$
\begin{aligned}
P_{FAST+3D} &= \{p_i \epsilon P_{FAST} | gneig(p_i) = 1\} \\
gneig(p_i) &= \begin{cases} 0 & si\ diag(p_i) \geq \tau \\ 1 & otros \end{cases} \\
diag(p_i) &= \frac{r}{2w_i + 1}
\end{aligned}
\tag{3.5}
$$

donde:

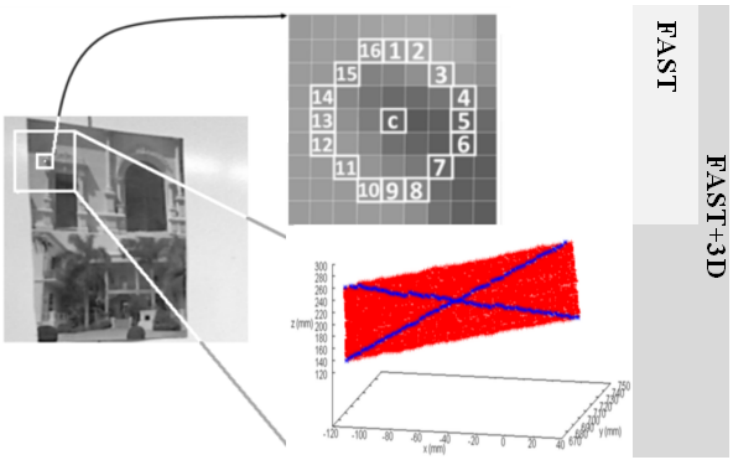
P_{FAST} es el conjunto de puntos detectados por FAST.

r es el número de pixeles en las diagonales del vecindariodonde la diferencia de pro-

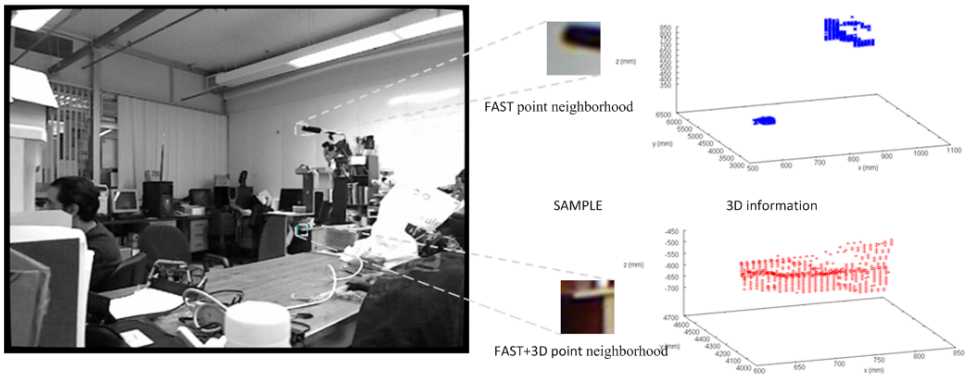
fundidades con el punto de interés (pixel central del vecindario) no es mayor que un umbral ρ determinado.

τ es un umbral por medio del cual se determina cual es el porcentaje máximo de puntos en las diagonales que serán tolerados para determinar que un punto de interés sea eliminado del conjunto final.

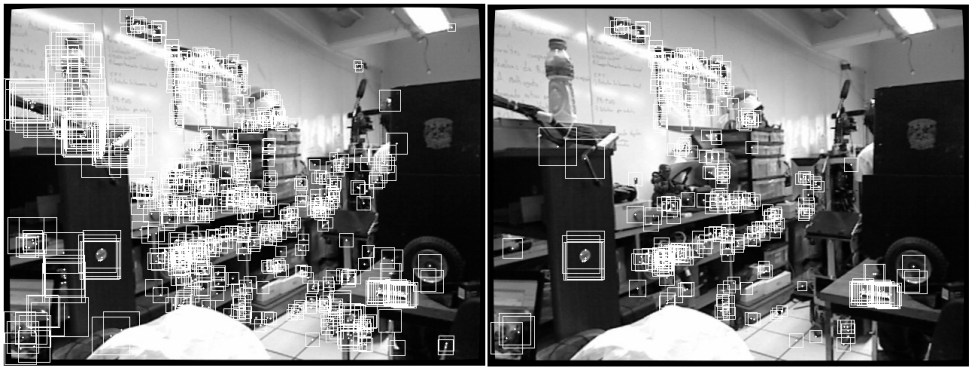
Para el desarrollo de este trabajo $\rho = 200$ milímetros, $\tau = 5\,\%$, el número de puntos a evaluar en FAST $n = 9$ y el umbral FAST $t = 40$. La figura 3.5c muestra los puntos detectados por los algoritmos FAST y FAST+3D en una escena determinada, una gran proporción de los puntos detectados por FAST no tienen un vecindario estable, lo que significa que realizar la descripción de los mismos no es recomendable; el número de puntos detectados por FAST+3D es menor pero son más estables.



(a)



(b)



(c)

Figura 3.5: FAST *vs* FAST+3D.(a) Detección de puntos de interés. (b) Ejemplos de punto detectado por FAST y FAST+3D, con sus respectivos vecindarios e información tridimensional. (c) Puntos detectados y sus vecindarios asociados FAST (izquierda) FAST+3D (derecha)

3.2. Cambios de perspectiva en muestras de escena

SIFT ha demostrado ser un algoritmo con un excelente desempeño en términos de desempeño de descripción. Sus descriptores han demostrado ser robustos ante un gran número de condiciones, sin embargo presenta algunos problemas cuando los cambios de perspectiva son cercanos o mayores a 45 grados. En los siguientes párrafos mostraré la técnica usada para normalizar descriptores ante cambios de perspectiva con los que SIFT no es capaz de lidiar.

Hasta este punto se ha determinado la escala de cada punto de interés y el tamaño en pixeles del vecindario que ha de ser caracterizado. La cámara RGB-D puede brinda información sobre la posición en coordenadas tridimensionales (con respecto a la posición de la cámara) de cada uno de los elementos que se encuentran en el vecindario, analizando esta información es posible determinar cuales son las características que dicho vecindario guarda con respecto a la cámara.

Como se ha descrito FAST+3D tiene una cierta tolerancia τ ante elementos que difieren en profundidad con el pixel central (punto de interés); realizar la estimación directa del plano que mejor describa al vecindario de un punto dado arrojaría muy probablemente un resultado erroneo, por ello, se realizó un análisis con RANSAC para eliminar dichos elementos atípicos usando como modelo la ecuación del plano $Ax + By + Cz + D = 0$ y el sistema de referencia mostrado en la figura 3.6 .

Se aplica un análisis de componentes principales al conjunto resultado, el eigenvector asociado con el eigenvalor de menor valor será considerado el vector normal de dicho plano (componentes A, B y C de la ecuación del plano). Para obtener el parámetro D nos auxiliamos de la siguiente deducción.

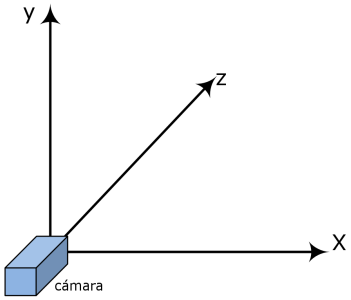


Figura 3.6: Sistema de referencia de la cámara RGB-D

Siendo $n = (n_x, n_y, n_z)$ y $r_0 = (x_0, y_0, z_0)$ la normal y un punto en el plano respectivamente, un punto$r = (x, y, z)$ pertenece al plano sí

$$\begin{aligned} n(r - r_0) &= 0 \\ n_x(x - x_0) + n_y(y - y_0) + n_z(z - z_0) &= 0 \\ n_x x - n_x x_0 + n_y y - n_y y_0 + n_z z - n_z z_0 &= 0 \\ n_x x + n_y y + n_z z - (n_x x_0 + n_y y_0 + n_z z_0) &= 0 \end{aligned} \tag{3.6}$$

de la ecuación del plano entonces

$$\begin{aligned} A &= n_x \\ B &= n_y \\ C &= n_z \\ D &= -(n_x x_0 + n_y y_0 + n_z z_0) \\ &= -n \bullet r_0 \end{aligned} \tag{3.7}$$

donde r_0 es la coordenada tridimensional del punto de interés en el cual se este realizando el análisis de vecindario.

Una vez obtenida la ecuación del plano que mejor represente al vecindario, los ángulos

α y γ son calculados con respecto al eje Z del sistema de referencia de la cámara (véase figura 3.7), los valores obtenidos pueden tener las siguientes variaciones.

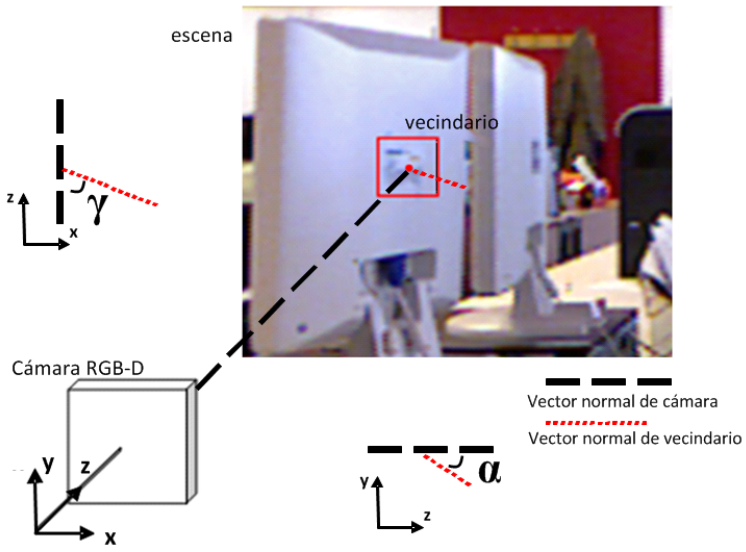
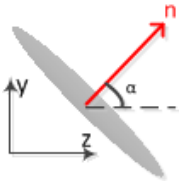
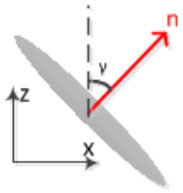
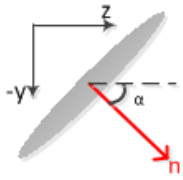
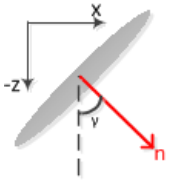
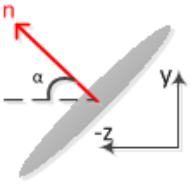
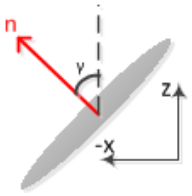
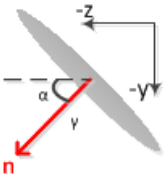
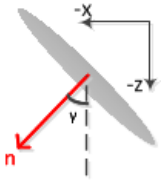


Figura 3.7: Análisis de vecindario

	α: giro en el eje X (plano ZY)	γ: giro en el eje Y (plano XZ)
caso 1:		
caso 2:		
caso 3:		
caso 4:		

resumen

z	y	giro
+	+	negativo
+	-	positivo
-	+	positivo
-	-	negativo

x	z	giro
+	+	positivo
+	-	negativo
-	+	negativo
-	-	positivo

Los ángulos calculados de cada vecindario son utilizados para realizar la transformación de perspectiva necesaria para colocar el vector normal del vecindario paralelo al vector

del eje Z del sistema de referencia de la cámara. La transformación es realizada con:

$$
\begin{aligned}
R_{yx} &= R_y R_x \qquad (3.8)\\
R_{yx} &= \begin{bmatrix} cos\gamma & 0 & -sen\gamma \\ 0 & 1 & 0 \\ sen\gamma & 0 & cos\gamma \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\alpha & sen\alpha \\ 0 & -sen\alpha & cos\alpha \end{bmatrix} \\
R_{yx} &= \begin{bmatrix} cos\gamma & sen\gamma sen\alpha & -sen\gamma cos\alpha \\ 0 & cos\alpha & sen\alpha \\ sen\gamma & -cos\gamma sen\alpha & cos\gamma cos\alpha \end{bmatrix}
\end{aligned}
$$

Luego es calculada la ubicación en la imagen de cada elemento del vecindario transformado. OpenNI tiene implementado un método que se encarga de manera automática de esta transformación cuyo nombre es *ConvertRealWorldToProjective*. En seguida calculo la matriz de transformación (ecuación 2.1) que mapee de las ubicaciones en la imagen del vecindario original a las localizaciones del vecindario después de la transformaciones. Ejemplos de cambios de perspectiva realizados por este método se ilustran en la figura 3.8.

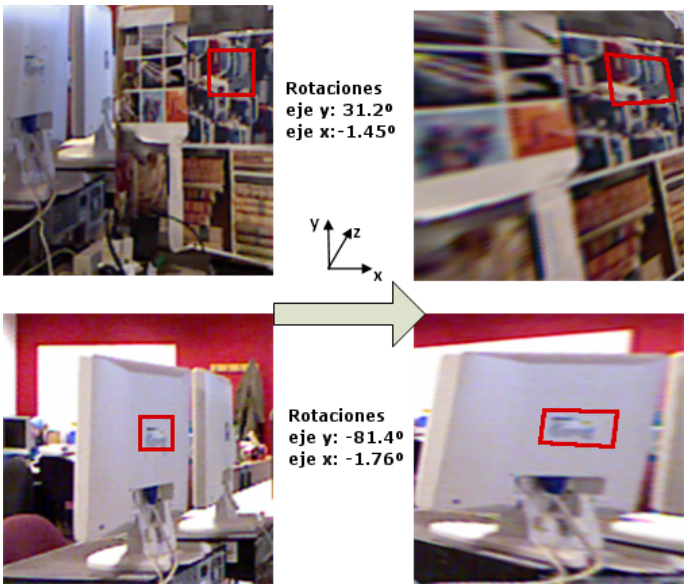


Figura 3.8: Cambios de perspectiva

3.3. Descriptor tipo *Scale Invariant Feature Transform* (SIFT)

Cada uno de los elementos del conjunto de vecindarios deformados es redimensionado a 25x25 pixeles y pasan por un proceso de descripción como como en SIFT [Low04] (véase sección 2.2.2). Las regiones de descripción tienen en conjunto un tamaño de 17x17 pixeles, de tal manera que ante la alineación con orientaciones dominantes de 45^{9} (peor caso), éstas se mantengan dentro de los limites del vecindario. Algunos de los ejemplos de estas zonas de descripción pueden ser observados en la figura 3.9.

Cada vector generado en los puntos de interés está comprendido finalmente de 133 elementos: 2 que indican las coordenadas de ubicación en imagen, 3 que almacenan las coordenadas de ubicación espacial con respecto a la cámara y 128 del descriptor tipo SIFT realizado para describir el vecindario.

3.4. Resumen de metodología de descripción

Las metodología de descripción planteada queda con los siguientes pasos bien definidos

1. Detectar puntos de interés con FAST.
2. Asociar un tamaño de vecindario (en pixeles) a cada punto de interés detectado a partir de una análisis profundidad-escala.
3. Depurar puntos detectados eliminando aquellos con vecindarios inestables.
4. Normalización ante cambios de perspectiva por medio del análisis de información 3D del vecindario de cada punto
5. Describir los puntos de interés detectados y sus vecindario normalizado como en SIFT.

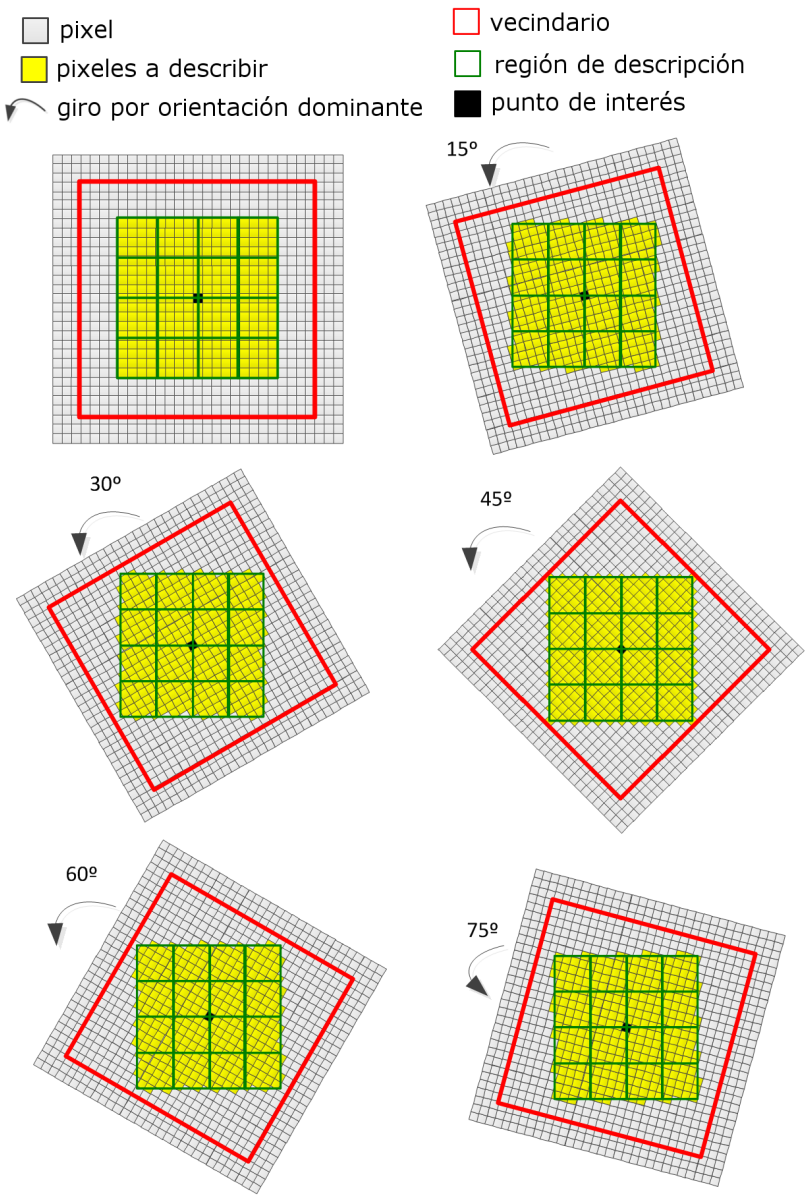


Figura 3.9: Ejemplos de rotaciones con respecto a orientación dominante

Capítulo 4

Aplicación en Robótica

La navegación es una de las tareas más complejas y necesarias en el desempeño de un robot. Implica 4 etapas bien definidas: *percepción*, donde el robot debe generar información de datos provenientes los elementos de percepción, *localización*, el robot debe responder la(s) pregunta(s) ¿donde me encuentro? ¿cuales es mi posición con respecto al sistema de referencia usado (x, y, θ)?, *toma de decisiones*, el robot determina los siguientes movimientos y *control de movimientos*, el robot realiza el control de sus actuadores para seguir la trayectoria determinada.

El uso de modelos topológicos para la descripción del entorno donde se desplazará un agente se percibe como la manera natural en la que los seres vivos se desplazan. Es decir, cuando nosotros llegamos a un lugar desconocido tratamos de encontrar puntos de referencia y las relaciones que existen entre ellos, a partir de las cuales podremos localizarnos en posteriores incursiones, además, no es natural pensar: "estoy en la posición 17,3 y necesito llegar a la 9,12"; sino más bien: "estoy en sala y necesito ir a la cocina que se encuentra a la izquierda".

Así también, es natural pensar que el sistema de navegación por excelencia en los seres humanos es el sistema visual, éste se basa en la percepción y aprendizaje de elementos hito del ambiente que posteriormente pueden ser utilizados para establecer relaciones e inferir localizaciones.

Hasta ahora se ha descrito una metodología de descripción y almacenaje de elementos visuales que puede ser utilizada para realizar empatados entre muestras de una escena dada, con esta información además, es posible determinar las relaciones que guardan las posiciones desde las cuales fueron realizadas dichas muestras. En las siguientes secciones se hablará un poco sobre un método de localización básica de robots y la manera en que un sistema de localización visual puede ser aplicado sustituir o complementar las inferencias arrojadas por dicho método.

4.1. Localización básica de robot por odometría

La odometría son aquellas técnicas que utilizan información proveniente de elementos de percepción (láser, sonar, cámara, etc.) para estimar la posición actual de un robot móvil en un instante determinado. El tipo más básico de odometría es aquel basado en las lecturas de elementos ubicados en las ruedas del robot (*encoders*), dichas lecturas determinan el número de giros o fracciones de él, que ha realizado las ruedas desde un momento determinado.

La odometría esta basada en que las lecturas de los *encoders* tiene una relación lineal con el desplazamiento real, lo cual es incorrecto pues existen una gran cantidad de elementos que llevan a la acumulación de diferencias en la relación lecturas-distancia, algunos de ellos incluso impredecibles. Algunas de las características que producen errores en la estimación y que pueden ser predichos son: los diámetros de las ruedas no son iguales,

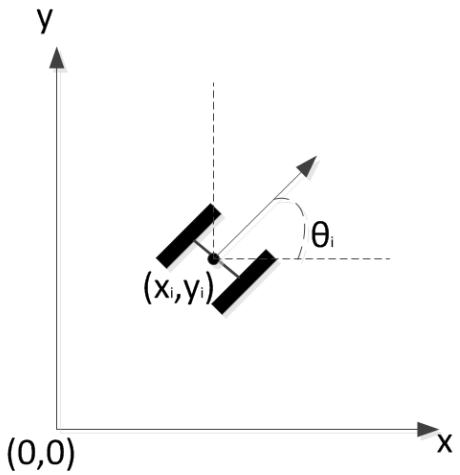


Figura 4.1: Parámetros de odometría

no se encuentran bien alineadas, la resolución de los encoders, entre otras. Lamentablemente existen algunos factores de estimación que no pueden ser predecidos, algunos de los que destacan son: patinaje de las ruedas, fuerzas externas, suelos irregulares, etc.

Aquellos errores predecibles se les denominan sistemáticos y pueden ser modelados por medio de una análisis de comportamiento del robot ante desplazamientos, estos análisis arrojan que la posición calculada está rodeada por una elipse de error característica que crece mientras el robot continué desplazándose debido a que este tipo de errores se acumulan constantemente (para mayor detalle al respecto puede consultarse en [TBF05, SN04, CK97]).

El problema con los errores no sistemáticos (no predecibles) es que agregan un gran error en la estimación de posición del robot. Estos errores al igual que las incertidumbres acumuladas por los errores sistemáticos pueden ser corregidos con la ayuda de sistemas de posicionamiento alternativos. Por ejemplo, dado el mapa topológico de figura 4.2, donde el objetivo del robot es desplazarse de la posición marcada con el circulo verde al circulo morado, los cuadrados azules son las posiciones referencia previamente registrada en el mapa y los círculos azules son aquellas posiciones que el planeador de movimientos

determinó como metas parciales antes del destino final, es posible realizar una corrección de los errores si el robot es capaz de determinar la posición relativa con respecto al siguiente destino parcial por medio de la metodología de localización que será mostrado en la siguiente sección.

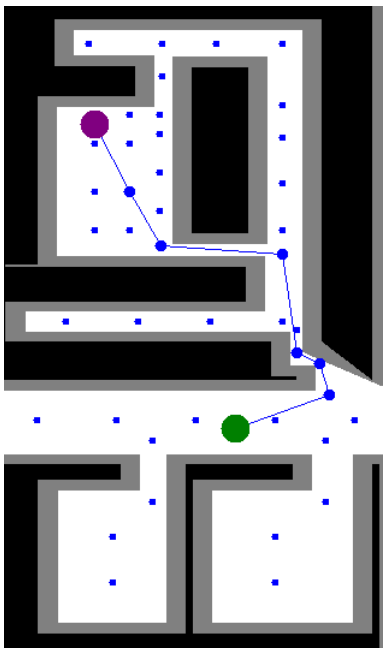


Figura 4.2: Mapa topológico

4.2. Empatado y Algoritmo de Orientación Absoluta

La metodología propuesta para realizar localización a base de empatado visual presupone una etapa de entrenamiento en la que se construye un mapa topológico donde cada ubicación establecida es muestreada en diferentes direcciones, obteniendo las características visuales y las relaciones entre ellas (véase figura 4.3).

Los descriptores de la imagen y su información tridimensional son generados utilizando la metodología mostrada en la figura 3.3 y desarrollada completamente en el Capítulo

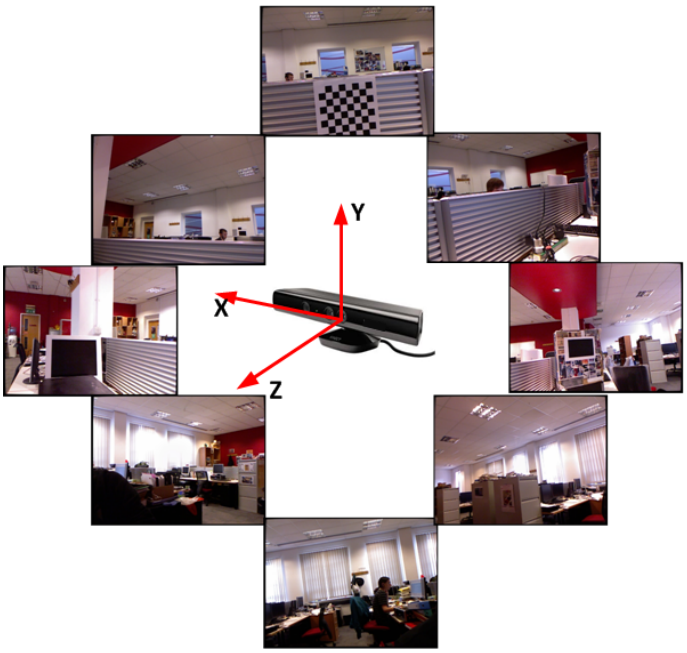


Figura 4.3: Muestreo en ubicaciones de mapa topológico

3, las características de cada imagen muestra son almacenadas con un árbol k-d, se tienen tantos arboles k-d como direcciones en las que se haya realizado el muestreo en cada ubicación.

En el esquema planteado, durante la ejecución de movimientos el sistema de localización visual recibe información sobre la posición a la que se esta dirigiendo el robot, con ello se comienza a comparar lo que el robot visualiza y lo que se encuentra almacenado en la base de conocimiento. El empatado de imágenes es realizado por medio de comparación de descriptores con distancia euclidiana, la eficiencia de este proceso es mejorada con el uso del algoritmo de busqueda BBF, posteriormente el conjunto de descriptores empatados es depurado por medio de RANSAC; si el número de elementos del conjunto depurado es menor a 4 se considera que el empatado no existe.

Una vez que se ha establecido el empate visual entre la escena observada y alguna de las direcciones muestreadas de una ubicación dada en etapa de entrenamiento, se

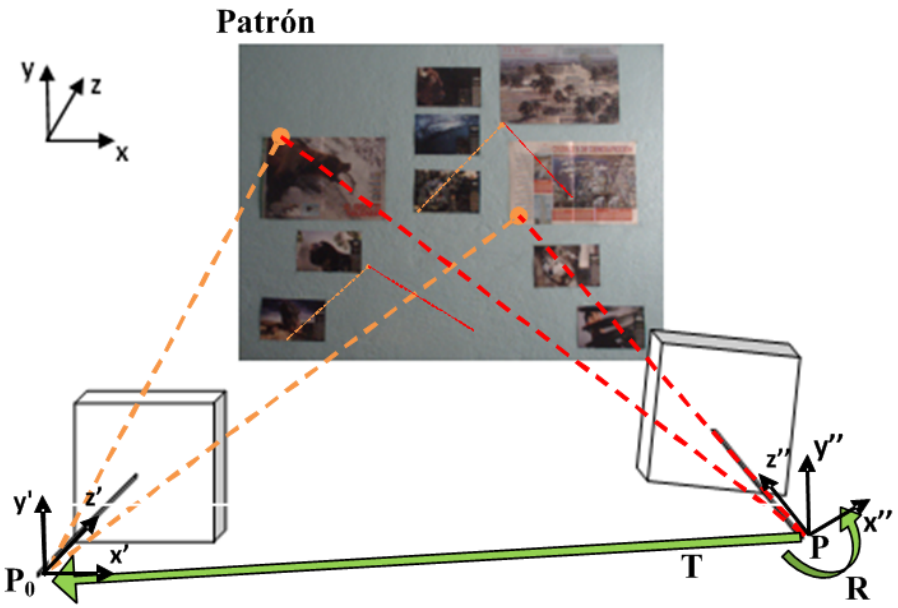


Figura 4.4: Orientación absoluta de un patrón

tienen dos conjuntos de puntos con localización espacial en dos sistemas coordenados diferentes; el encontrar las relaciones de translación T y rotación R entre estos sistemas es un problema clásico de fotogrametría llamado orientación absoluta. .

Siendo $\{p_{1,i}\}$ y $\{p_{2,i}\}$ los conjuntos de coordenadas tridimensionales de los puntos correctamente empatados en el sistema coordenado 1 y 2 respectivamente, donde i va de*1* hasta n (número de puntos empatados), ahora se busca la transformación que mejor describa dicha relación en términos de coordenadas espaciales. Dicha transformación debe satisfacer para par de coordenadas la siguiente estructura:

$$p_{2,i} = \lambda R(p_{1.i}) + T \tag{4.1}$$

donde λ es un factor de escala

R y T son las matrices de rotación y translación que relacionan los sistemas

Debido a que las medidas de las coordenadas tridimensionales no son exactas la ecuación

4.1 no satisface en todos los pares de puntos, por ello se tendrá un valor residual dado por:

$$e_i = p_{2,i} - \lambda R(p_{2,i}) - T \tag{4.2}$$

El objetivo es encontrar los valores de R, T y λ que minimicen la suma de los errores cuadrados $\sum_{i=1}^{n} \|e_i\|^2$.

Encontrar el mejor conjunto de parametros (6, de la matriz de rotación, 3 del vector de translación y el factor de escala) no es una tarea fácil de realizar y muchos métodos han sido planteados para encontrarlos, la mayoría de ellos iterativos; ejemplos de estas metodologías pueden ser encontrados en [Kra00, EMM01, PW00]. Horn en [Hor87] plantea una metodología de forma cerrada (no basada en iteraciones) para la estimación de los parametros. A continuación los resultados de su análisis matemático.

Siendo los centroides de cada conjunto de puntos calculados como

$$\begin{aligned} \bar{p}_1 &= \frac{1}{n}\sum_{i=1}^{n} p_{1,i} \\ \bar{p}_2 &= \frac{1}{n}\sum_{i=1}^{n} p_{2,i} \end{aligned} \tag{4.3}$$

el **vector de translación** que minimiza la suma de errores cuadrados es

$$T = \bar{p}_2 - \lambda R(\bar{p}_1) \tag{4.4}$$

el **factor de escala** correspondiente es

$$\lambda = \sqrt{\frac{\sum_{i=1}^{n} \|\bar{p}_2\|^2}{\sum_{i=1}^{n} \|\bar{p}_1\|^2}} \tag{4.5}$$

para el calculo de la **matriz de rotación** primero se define que una rotación de ángulo β sobre un eje definido por el vector $\hat{w} = (w_x, w_y, w_z)^T$ puede ser representada por el *cuaternión unitario*[1]

$$\begin{aligned} \dot{q} &= [q_0\ q_x\ q_y\ q_z] \\ \dot{q} &= [\cos\frac{\beta}{2}\ \sin\frac{\beta}{2}w_x\ \sin\frac{\beta}{2}w_y\ \sin\frac{\beta}{2}w_z] \end{aligned} \tag{4.6}$$

el cuaternión que minimiza la suma de errores cuadrados es el eigenvector cuyo eigenvalor correspondiente sea el mayor de la matriz N

$$N = \begin{bmatrix} (S_{xx}+S_{yy}+S_{zz}) & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & (S_{xx}-S_{yy}-S_{zz}) & S_{xy}+S_{yx} & S_{yz}+S_{zy} \\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & (-S_{xx}+S_{yy}-S_{zz}) & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{yz}+S_{zy} & S_{yz}+S_{zy} & (-S_{xx}-S_{yy}+S_{zz}) \end{bmatrix} \tag{4.7}$$

donde $S_{xx} = \sum_{i=1}^{n} p_{1,i}(x)\, p_{2,i}(x)$, $S_{xy} = \sum_{i=1}^{n} p_{1,i}(x)\, p_{2,i}(y)$ y así sucesivamente.

la matriz de rotación estará dada entonces por

$$R = \begin{bmatrix} q_0^2+q_x^2-q_y^2-q_z^2 & 2(q_xq_y-q_0q_Z) & 2(q_xq_z+q_0q_y) \\ 2(q_yq_x+q_0q_z) & q_0^2-q_x^2+q_y^2-q_z^2 & 2(q_yq_z-q_0q_x) \\ 2(q_zq_x-q_0q_y) & 2(q_zq_y+q_0q_x) & q_0^2-q_x^2-q_y^2+q_z^2 \end{bmatrix} \tag{4.8}$$

Para los propósitos de este trabajo únicamente es necesario determinar la rotación que ha sido realizada en el eje y, es decir, solo nos interesa el giro θ en el plano XZ (según

[1] *vector de cuatro elementos con norma igual a 1,* $\dot{q} = [q_0\ q_x\ q_y\ q_z]$

el sistema coordenado de la figura 3.6).

$$\theta_{XZ} = acos\left((R[0\ 0\ 1]^T) \bullet [0\ 0\ 1]^T\right) \tag{4.9}$$

si $2(q_x q_z + q_0 q_y) > 0$ el giro es positivo, sino el giro es negativo.

4.3. Résumen de métodología de localización visual

La metodología de localización visual planteada consta de dos etapas, en la primera se realizar la caracterización del entorno. Los pasos asociados son:

1. Describir cada ubicación en el mapa topológico con la metodología planteada en el capitulo 3. La descripción es realizada en tantas direcciones como convenga en la ubicación a describir.

2. Construir de árboles k-d para almacenar los descriptores obtenido y la información 3D asociada a ellos.

En la segunda etapa se realiza el empatado y el cálculo de posiciones relativas a partir de los siguientes pasos:

1. Extraer características desde el punto de vista actual del agente.

2. Buscar descriptores obtenidos en los árboles k-d con uso del algoritmo BBF (empatado puntual).

3. Eliminar puntos empatados anómalos con RANSAC y el modelo de cambio de perspectiva en imágenes 2D (empatado de imágenes).

4. Estimar posición relativa entre patrón almacenado y escena observada con Algoritmo de Orientación Absoluta.

Capítulo 5

Experimentos

El método de descripción visual y su aplicación en robots móviles descritos en los capítulos anteriores son evaluados usando un sensor MS Kinect como cámara RGB-D en resolución de 640x480 pixeles. Todas las pruebas están compuestas de fase de entrenamiento y fase de prueba, efectuando en la primera la extración y almacenamiento de características visuales de una escena dada y en la segunda fase se intenta realizar el empatado visual (busqueda de coincidencia) y la generación de conclusiones pertinentes.

La presentación visual del software generado se observa en la figura 5.1.

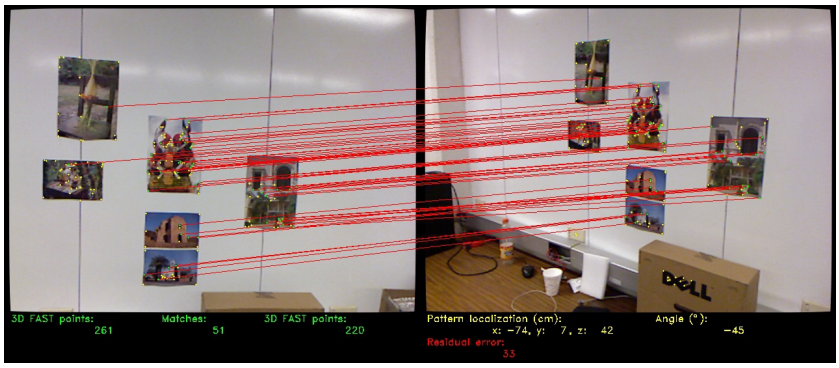


Figura 5.1: Interfaz de software

5.1. Posición real vs posición estimada

En la etapa de entrenamiento, un patrón visual es colocado a una distancia de 1 metro de la cámara y los descriptores del mismo son calculados y almacenados como he descrito anteriormente.

En la fase de prueba, la cámara RGB-D es colocada en distintas posiciones cada 33 centímetros, la cámara siempre es colocada de manera que el plano de la imagen y el patrón observado sean paralelos en toda ejecución de la prueba. Estas posiciones están marcadas con pequeños rombos azules en la figura 5.2. Una vez colocada la cámara en una de las posiciones determinadas, los metodos de descripción, busqueda y localización son ejecutados en 200 ocasiones. El umbral de detección usado en FAST+3D (t) en esta fase de entrenamiento es colocado en 50 y en prueba es 60, con lo que el número de puntos descritos en la primera etapa siempre es mayor.

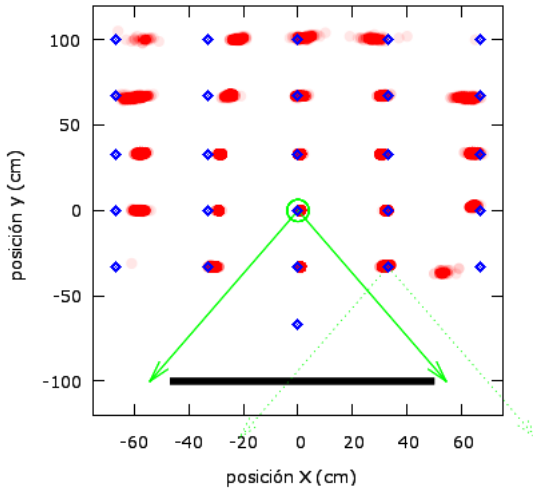


Figura 5.2: Comparación entre posición estimada (circulos rojos) y real (cuadros azules)

Las posiciones calculadas por el método de localización están marcadas como círculos rojos en la figura 5.2. De manera general, el rendimiento de la metodología brindo localizaciones adecuadas, donde los errores tienden a incrementarse proporcionalmente

con la distancia de la cámara a la localización donde fue realizada la muestra patrón. El tiempo promedio que la metodología utilizó para finalizar cada ejecución es 574(ms), con 449 descriptores del patrón almacenados en el árbol k-d respectivo y un promedio de 157 descriptores en promedio generados en cada ejecución en etapa de prueba. El error promedio en términos del eje coordenado horizontal es 3.75 (mm) y el vertical con 0.39 (mm).

El peor caso se presentó en la posición (67,-33) con un error promedio de 17 (mm), probablemente porque el número de pares de puntos empatados es poco significativo (solo 6) y su localización en la imagen está concentrada en una zona pequeña de la imagen. Existen algunas ubicaciones donde el porcentaje de localizaciones positivas fue mínimo (1 % de las ejecuciones en dicha posición) o nulo, ambas debido a que la distancia hacia el patrón era grande y la información devuelta por la cámara es insuficiente o por que la cámara se encuentra demasiado cerca del patrón, en la zona donde la cámara no es capaz de devolver información espacial (distancias menores a 40 cm).

5.2. Ángulo de vista real vs estimado

La etapa de entrenamiento en este experimento es la misma que en la prueba anterior.

En la etapa de prueba, el patrón observado fue colocado a una distancia de 1.5 metros, a diferentes ángulos con respecto al eje Y del sistema coordenado de la cámara. El conjunto de ángulos en los que se realizó la prueba es {-60, -45, -30, 0, 30, 60}. El algoritmo estándar SIFT fue ejecutado de manera paralela para visualizar y comparar su robustez bajo cambios de perspectiva y compararlo con el método propuesto (véase figura 5.3). La tabla 5.1 muestra los resultados obtenidos en la comparación.

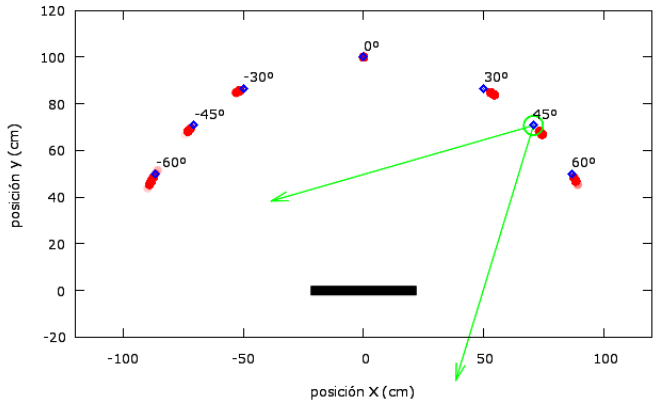


Figura 5.3: Comparación entre ángulo estimado(circulos rojos) y real (cuadros azules)

	SIFT			Detector FAST+3D y descriptor SIFT		
Ángulo	% empatado	Puntos detectados	Tiempo de ejecución [ms]	% empatado	Puntos detectados	Tiempo de ejecución [ms]
60	2	779	2077.51	100	225	705.53
45	100	860	2210.41	100	290	735.59
30	100	889	2331.35	100	297	728.80
0	100	846	2582.45	100	265	706.31
-30	100	867	2442.66	100	193	560.79
-45	100	880	2429.34	100	152	735.59
-60	3	826	2254.08	94	181	632.03
promedio		**850**	**2332.54**		**228**	**686.38**

Tabla 5.1: Comparación entre SIFT y la descripción con cambio de perspectiva

La tabla 5.1 muestra que SIFT trabaja bien con cambios de perspectiva menores a 45^{o} (porcentaje de empatado es 100 %), pero pierde precisión cuando el ángulo es incrementado, mientras que la metodología de descripción propuesta se mantiene realizando empatados positivos en la mayor parte de las ejecuciones (94 %). El tiempo de ejecución necesario para realizar SIFT es muy superior debido a que el análisis espacio-escala es computacionalmente caro. En la metodología propuesta, sin embargo, su cálculo no es necesario ya que se hace uso de la información devuelta por la cámara RGB-D, para determinar el tamaño de las zonas a describir.

La media del error de la metodología propuesta en la medida del ángulo es de 1.71^{o}. La

peor estimación se dio en una ejecución en el punto marcado con -60º, donde el error de estimación es de 4º, sin embargo la precisión del ángulo continua considerándose buena.

5.3. Robustez ante cambios de ángulo

La etapa de entrenamiento en este experimento es la misma que en las prueba anteriores.

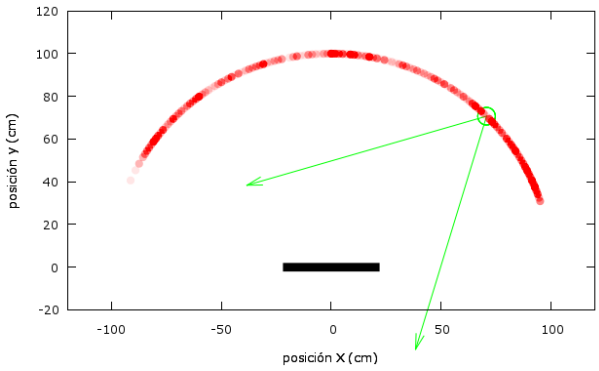
En la etapa de prueba, el patrón fue puesto a una distancia de 1.5 metros y el ángulo fue variado de manera continua con objeto de evaluar los ángulos en los cuales el empatado es realizado correctamente (véase 5.4a).

La metodología propuesta es capaz de realizar empatados correctos con cambios de perspectiva de hasta 72º. El hecho de que sea posible realizar empatados con ángulos mayores de 60º es sin duda uno de los puntos fuertes de la metodología, mostrando que la normalización implementada para cambios de perspectiva está arrojando los resultados esperados.

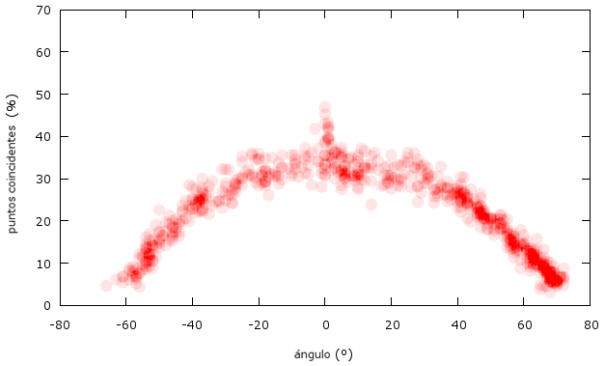
Como resultado adicional, la figura 5.4b muestra como los cambios de perspectiva causan una reducción en el porcentaje de puntos coincidentes entre la imagen patrón y la observada en en la prueba, hasta el momento en que el número de puntos es insuficiente para realizar los calculos necesarios en la metodología.

5.4. Ambiente con condiciones no controladas

La metodología de localización fue ejecutada en un corredor de oficina donde las condiciones ambientales no fueron controladas, el mapa del mismo puede observarse en la



(a) Ángulos de empatado



(b) Porcentaje de puntos empatado

Figura 5.4: Respuesta ante variaciones de ángulo

figura 5.5.

En la etapa de entrenamiento, un conjunto de patrones fue extraído de determinadas posiciones. Las ubicaciones desde las cuales fue muestreado el entorno y los patrones están marcados en color azul y rojo, respectivamente, en la figura 5.5. Las condiciones de luz y el número de puntos descritos en cada escena patrón analizada tienen una gran cantidad de variaciones, lo que permitió observar el comportamiento de la metodología ante condiciones muy diferentes.

Figura 5.5: Mapa, ubicaciones y vecindarios descritos de prueba

Figura 5.5: Mapa, ubicaciones y vecindarios descritos de prueba (continuación)

En la fase de prueba, la metodología es ejecutada en promedio 168 veces por cada localización, realizando movimientos de manera manual con la cámara, dirigiéndola siempre hacia el patrón analizado y de manera continua. Siempre que se diera un empatado

de imágenes correcto, la localización relativa a la ubicación patrón es calculada. Los porcentajes de localización pueden ser observados en la figura 5.6.

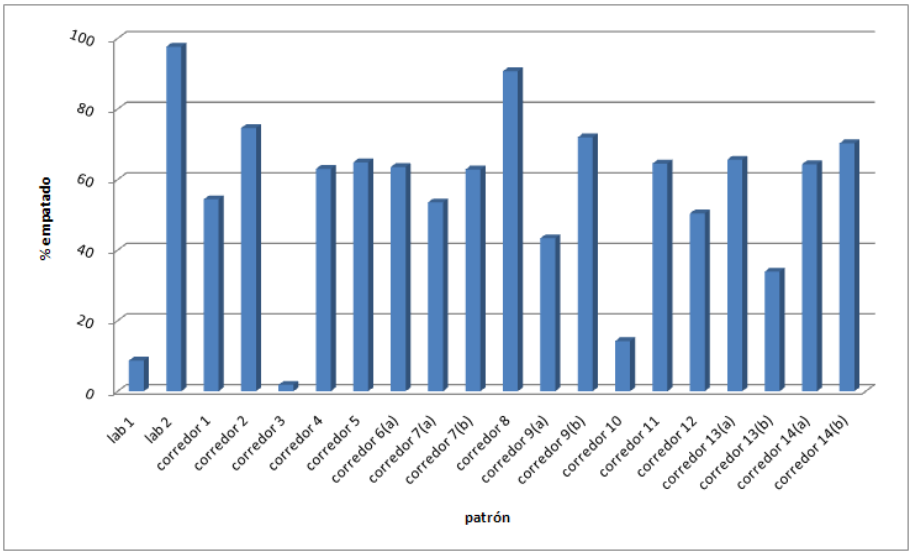


Figura 5.6: Porcentajes de localizaciones exitosas por ubicación

En alguna ubicaciones, donde las condiciones de luz eran pobres, el porcentaje de empatado visual fue muy bajo, debido a que con esas condiciones el número de puntos detectados por FAST+3D es insuficiente. Algunas imágenes no son lo suficientemente estables, pues la cámara siempre estuvo en movimiento y los puntos de interés no estaban bien marcados; además, por las características del ambiente de trabajo, las imágenes presentaban una gran cantidad de reflejos, lo cual también tuvo influencia en el desempeño. La tabla 5.2 presenta un resumen del desempeño.

Localización	# ejecuciones	tiempo de ejecución (ms)	% empatado	puntos FAST+3D	# puntos empatados
lab 1	368	180.08	8.70	155	7
lab 2	127	496.34	97.64	215	48
corredor 1	160	245.10	54.38	128	30
corredor 2(a)	87	57.43	0.00	0	0
corredor 2(b)	181	235.36	74.59	105	23
corredor 3	387	73.81	1.81	26	6
corredor 4	200	184.07	63.00	42	10
corredor 5	148	384.65	64.86	195	25
corredor 6(a)	181	211.62	63.54	87	16
corredor 6(b)	83	70.84	0.00	0	0
corredor 7(a)	170	230.36	53.53	92	17
corredor 7(b)	70	402.02	62.86	146	17
corredor 8	86	746.56	90.70	373	51
corredor 9(a)	60	161.10	43.33	63	18
corredor 9(b)	107	204.22	71.96	62	17
corredor 10	225	157.92	14.22	73	7
corredor 11	265	303.05	64.53	148	21
corredor 12	256	285.50	50.39	171	25
corredor 13(a)	32	1832.60	65.63	904	84
corredor 13(b)	233	135.49	33.91	43	8
corredor 14(a)	216	394.76	64.35	200	37
corredor 14(b)	74	272.34	70.27	115	21

Tabla 5.2: Detalle de resultados

Capítulo 6

Conclusiones

La metodología de descripción desarrollada presenta dos etapas bien definidas, la primera realiza la detección de puntos de interés y la segunda realiza la descripción de la información generada y la almacena. En el desarrollo de este trabajo fue utilizada información tridimensional generada por una cámara RGB-D.

En la etapa de detección se agregó un análisis extra de puntos candidato devueltos por el detector FAST para eliminar aquellos que tienen un vecindario con altas variaciones de profundidad (inestables). Esta modificación es llamada FAST+3D. El tamaño del vecindario analizado se determina a partir del análisis profundidad-escala.

En la etapa de descripción se realizó un análisis de vecindario más extenso utilizando la información espacial que la cámara RGB-D brinda, lo cual permitió realizar una normalización ante cambios de perspectiva. Los vecindarios normalizados son descritos por medio de descriptores tipo SIFT, obteniéndose invarianza a cambios de iluminación, orientación y escala.

La información generada es alamacenada con arboles k-d y las busquedas son realizadas

con el algoritmo Best Bin First. La metodología de descripción propuesta ha demostrado tener una buena robustez ante cambios de perspectiva, el empatado de escenas continua realizándose con cambios de hasta $70^{\underline{o}}$.

También fue generada una metodología de localización por medio de la cual es posible determinar la ubicación relativa entre las posiciones desde las cuales fueron capturadas la imágenes empatadas.

Los tiempos de ejecución de los procesos de descripción, busqueda y localización dependen de la cantidad de puntos detectados por FAST+3D y el tamaño del vecindario que corresponde a cada uno. Las pruebas realizadas muestran que éstos van desde las cuatro ejecuciones por segundo. La metodología de localización ha mostrado tener una precisión adecuada, la cual depende del acercamiento que se tenga a la posición patrón.

Este trabajo además plantea el uso de la metodología de localización en el campo de la robótica, la construcción de mapas topológicos visuales se adecua a la metodología de navegación de robots como el construido en el laboratorio de Biorrobótica de la UNAM. Aunque las pruebas no fueron realizadas directamente en un robot, fueron emuladas las condiciones que podría tener la ejecución en el mismo, incluso la manera en que fueron ejecutadas plantea un grado de dificultad más complejo.

En términos generales las metodologías han mostrado tener un buen desempeño, sin embargo aún es posible incrementarlo por medio de las paralelización de los algoritmos. Así también, como trabajo futuro, podrían probarse algunas técnicas de detección y descripción diferentes a las que se plantearon. En el caso de la detección podrían ser evaluadas técnicas que a pesar de ser más lentas que FAST tienen una mayor repitibilidad (véase [MS05, SMB00, MGBR08]) o construir un nuevo detector que haga un uso más extensivo de la información tridimensional. Una versión del descriptor SIFT que haga uso de una análisis de componente principales (como en PCA-SIFT) para reducir

la dimensión de los descriptores, cambiar la forma de las regiones de descripción (como en GLOH) son alternativas que se vislumbran viables para la mejora del desempeño y robustez.

Bibliografía

[Aca01] Real Académia Española, *Diccionario de la lengua española*, 22 ed., Espasa, 2001.

[Ben75] Jon Louis Bentley, *Multidimensional binary search trees used for associative searching*, Commun. ACM **18** (1975), pp. 509–517.

[BL97] Jeffrey S. Beis and David G. Lowe, *Shape indexing using approximate nearest-neighbour search in high-dimensional spaces*, Proceedings of the 1997 Conference on Computer Vision and Pattern, CVPR '97, IEEE Computer Society, 1997.

[CK97] Kok Seng Chong and Chong Lindsay Kleeman, *Accurate odometry and error modelling for a mobile robot*, IEEE International Conference on Robotics & Automation, 1997, pp. 2783–2788.

[CPMC07] Denis Chekhlov, Mark Pupilli, Walterio Mayol, and Andrew Calway, *Robust real-time visual slam using scale prediction and exemplar based feature description*, Proc. Int Conf on Computer Vision and Pattern Recognition, 2007, pp. 1–7.

[CPMcC06] Denis Chekhlov, Mark Pupilli, Walterio Mayol-cuevas, and Andrew Calway, *Real-time and robust monocular slam using predictive multi-resolution*

descriptors, 2nd International Symposium on Visual Computing, 2006, pp. 276–285.

[DRMS07] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse, *Monoslam: Real-time single camera slam*, IEEE Trans. Pattern Anal. Mach. Intell. **29** (2007), 1052–1067.

[EMM01] J. Chris McGlone Edward M. Mikhail, James S. Bethel, *Introduction to modern photogrammetry*, John Wiley & Sons, Inc, 2001.

[FB81] Martin A. Fischler and Robert C. Bolles, *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*, Commun. ACM **24** (1981), 381–395.

[GMBR10] Arturo Gil, Oscar Mozos, Monica Ballesta, and Oscar Reinoso, *A comparative evaluation of interest point detectors and local descriptors for visual slam*, Machine Vision and Applications **21** (2010), 905–920.

[GW01] Rafael C. Gonzalez and Richard E. Woods, *Digital image processing*, 2nd ed., Addison-Wesley Longman Publishing Co., Inc., 2001.

[Hes10] Rob Hess, *An open-source siftlibrary*, Proceedings of the international conference on Multimedia (New York, NY, USA), MM '10, ACM, 2010, pp. 1493–1496.

[HKH+10] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox, *Rgbd mapping: Using depth cameras for dense 3d modeling of indoor environments*, Proc. of International Symposium on Experimental Robotics, 2010.

[Hor87] Berthold K. P. Horn, *Closed-form solution of absolute orientation using unit quaternions*, Journal of the Optical Society of America A **4** (1987), pp. 629–642.

[HS88] C. Harris and M. Stephens, *A combined corner and edge detector*, Proceedings of the 4th Alvey Vision Conference, 1988, pp. 147–151.

[HZ03] Richard Hartley and Andrew Zisserman, *Multiple view geometry in computer vision*, 2 ed., Cambridge University Press, New York, NY, USA, 2003.

[Jol02] I. T. Jolliffe, *Principal Component Analysis*, second ed., Springer, 2002.

[Kra00] Karl Kraus, *Photogrammetry: Geometry from images and laser scans*, second ed., Walter de Gruyter, 2000.

[KS04] Yan Ke and Rahul Sukthankar, *Pca-sift: a more distinctive representation for local image descriptors*, Proceedings of the 2004 IEEE computer society conference on Computer vision and pattern recognition, CVPR'04, IEEE Computer Society, 2004, pp. 506–513.

[Lin94] Tony Lindeberg, *Scale-space theory: A basic tool for analysing structures at different scales*, Journal of Applied Statistics **21** (1994), pp. 224–270.

[Low04] David G. Lowe, *Distinctive image features from scale-invariant keypoints*, Int. J. Comput. Vision **60** (2004), pp. 91–110.

[LSKE11] Adalberto Llarena, Jesus Savage, Angel Kuri, and Boris Escalante-Ramírez, *Odometry-based viterbi localization with artificial neural networks and laser range finders for mobile robots*, Journal of Intelligent & Robotic Systems (2011), pp. 1–35.

[Mar82] David Marr, *A computacional investigation into the human representation and processing of visual information*, W. H. Freeman and Company, 1982.

[MCC10] Jose Martinez-Carranza and Andrew Calway, *Unifying planar and point mapping in monocular slam*, Proceedings of the British Machine Vision Conference, BMVA Press, 2010.

[MGBR08] Oscar Martinez Mozos, Arturo Gil, Monica Ballesta, and Oscar Reinoso, *Interest point detectors for visual SLAM*, 12th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2007 (Daniel Borrajo, Luis Castillo, and Juan Manuel Corchado, eds.), Lecture Notes in Artificial Intelligence, vol. 4788, Springer, Germany, 2008, pp. 170–179.

[ML00] Don Murray and Jim Little, *Using real-time stereo vision for mobile robot navigation*, Autonomous Robots, vol. 8, 2000, pp. 161–171.

[MM06] Farzin Mokhtarian and Farahnaz Mohanna, *Performance evaluation of corner detectors using consistency and accuracy measures*, Comput. Vis. Image Underst. **102** (2006), pp. 81–94.

[MS02] K. Mikolajczyk and C. Schmid, *An affine invariant interest point detector*, Proceedings of the 7th European Conference on Computer Vision-Part I (London, UK, UK), ECCV '02, Springer-Verlag, 2002, pp. 128–142.

[MS05] Krystian Mikolajczyk and Cordelia Schmid, *A performance evaluation of local descriptors*, IEEE Trans. Pattern Anal. Mach. Intell. **27** (2005), 1615–1630.

[PW00] Bon DeWitt Paul Wolf, *Elements of photogrammetry*, third edition ed., McGraw-Hill, 2000.

[RD05] Edward Rosten and Tom Drummond, *Fusing points and lines for high performance tracking*, Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2, ICCV '05, IEEE Computer Society, 2005, pp. 1508–1515.

[RD06] Edward Rosten and Tom Drummond, *Machine learning for high-speed corner detection*, European Conference on Computer Vision, 2006, pp. 430–443.

[Rea06] Real Académia Española, *Diccionario panhispanico de dudas*, 2 ed., Santillana, 2006.

[SLL05] Stephen Se, David G. Lowe, and James J. Little, *Vision-based global localization and mapping for mobile robots*, IEEE Transactions on Robotics **21** (2005), pp. 364–375.

[SMB00] Cordelia Schmid, Roger Mohr, and Christian Bauckhage, *Evaluation of interest point detectors*, Int. J. Comput. Vision **37** (2000), pp. 151–172.

[SN04] Roland Siegwart and Illah R. Nourbakhsh, *Introduction to autonomous mobile robots*, Bradford Company, Scituate, MA, USAv, 2004.

[ST93] Jianbo Shi and Carlo Tomasi, *Good features to track*, Tech. report, Ithaca, NY, USA, 1993.

[TBF05] Sebastian Thrun, Wolfram Burgard, and Dieter Fox, *Probabilistic robotics (intelligent robotics and autonomous agents)*, The MIT Press, 2005.