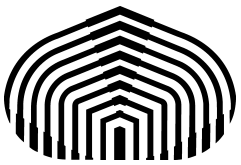


UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERIA ELECTRÓNICA

SISTEMA DE GENERACIÓN DE MOSAICOS 2D PARA ROBOTS MÓVILES A PARTIR DE VIDEO MONOCULAR

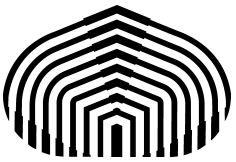
Por:
Victor Yovanni Garcia Carmona

Realizado con la asesoría de:
José de la Cruz Cappelletto Fuentes

PROYECTO DE GRADO

Presentado ante la Ilustre Universidad Simón Bolívar
como requisito parcial para optar al título de
Ingeniero Electrónico

Sartenejas, Marzo de 2018



UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA ELECTRÓNICA

ACTA FINAL PROYECTO DE GRADO

SISTEMA DE GENERACIÓN DE MOSAICOS 2D PARA ROBOTS MÓVILES A PARTIR DE VIDEO MONOCULAR

Presentado por:
Victor Yovanni Garcia Carmona

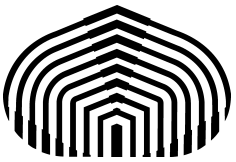
Este Proyecto de Grado ha sido aprobado por el siguiente jurado examinador:

José de la Cruz Cappelletto Fuentes

Novel Antonio Certad H.

Gerardo Fernandez López

Sartenejas, @día de Mayo de 2018



UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA ELECTRÓNICA

SISTEMA DE GENERACIÓN DE MOSAICOS 2D PARA ROBOTS MÓVILES A PARTIR DE VIDEO MONOCULAR

PROYECTO DE GRADO

PRESENTADO POR:
Victor Yovanni Garcia Carmona, Carnet: 12-10738

RESUMEN

Al realizar tareas de exploración para el análisis del suelo en espacios aereos o en fondo marino, es muy comun emplear sistemas de adquisición basados en captura de videos para su posterior análisis. En la actualidad, el incremento de la tecnologia sobre el procesamiento de datos, ha permitido que los algoritmos de vision por computadora coloquen a la camara como principal sensor para la reconstruccion de entornos recorridos por vehiculos móviles. El presente trabajo se encuentra enfocado al análisis e implementación de distintos algoritmos para la reconstrucción de un mosaico 2D (dos dimensiones) del suelo que recorre un robot, a partir de la información proveniente de una cámara monocular ubicada en la parte inferior de este. El robot en cuestión puede realizar recorridos aéreos para realizar la adquisición del vídeo, o incluso trayectorias mas desafiantes como serian las aplicaciones subacuáticas. Para esto, se implementarán distintos algoritmos usando técnicas de procesamiento de imágenes y visión por computadora, para la elaboración de un sistema automatizado que permita generar un mapa en dos dimensiones de la trayectoria recorrida por el vehículo móvil, mejorando la detección de puntos clave y optimizando el calculo de las matrices de transformación para la alineación de imágenes en el mosaico, logrando así un mapa con la menor distorsión posible. Además de realizar análisis sobre el error de proyección de dichas imágenes en el mapa del suelo generado.

Palabras clave: mosaico, video monocular, puntos clave, matriz de transformación.

Agradecimientos

Índice general

Índice de figuras

Índice de Tablas

Acrónimos y Símbolos

Dedicatoria

A @personasImportantes, por @razonesDedicatoria.

Capítulo 1

Introducción

La navegación y exploración en áreas de difícil acceso mediante el uso de robots, es una tarea que se ha venido desarrollando en el Grupo de Investigación y Desarrollo en Mecatrónica de la USB *(GIDM)* desde hace mu tiempo. Donde una de las aplicaciones mas demandantes, es la tarea de reconstruir un mapa 2D de la superficie mapeada por los robots utilizados. En el presente capitulo se pretende introducir los trabajos previos y avances que se han tenido en el desarrollo de este tipo aplicaciones, específicamente en el *GIDM*, que dieron origen y motivación para la realización del proyecto. Además, de postular un serie de problemas que el presente trabajo busca solucionar.

1.1. Antecedentes

En el *GIDM* se han realizado grandes avances en el desarrollo de equipos y plataformas robóticas, para actividades de investigación, exploración e inspección de ambientes no estructurados. Usualmente cuando se opera en este tipo de ambientes, en busca de realizar exploraciones mas eficientes y a mayor escala, se emplean vehículos operados remotamente ROV (del inglés: Remotely Operated Vehicles) equipados con cámaras de vídeo, o bien en el caso de aplicaciones subacuáticas también se suelen emplear vehículos autónomos submarinos AUV (del inglés: Automated Underwater Vehicles), mientras que para exploraciones aéreas son empleados vehículos aéreos no tripulados UAV (del ingles: Unmanned Aerial Vehicle).

En este sentido, se tienen proyectos como el presentado por *Danilo, D.* [1], cuyo trabajo de grado consistió en el desarrollo en un sistema de operación remota para un robot submarino (ROV), con la finalidad de implementarlo en tareas de exploración. Con objetivos similares, *Said, A.* [2] basó su proyecto de grado en la instrumentación y control de un robot cuadricóptero volador(AUV).

Adicional a los proyectos antes mencionados, en el el *GIDM* se cuenta con un vehículo submarino llamado OpenROV[1], el cual es un robot maniobrado remotamente de baja envergadura diseñado especialmente para operar bajo el agua.

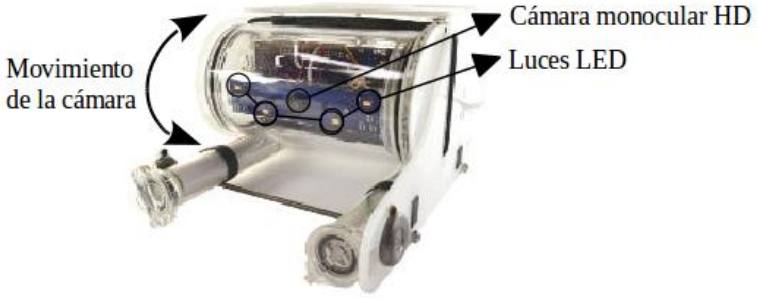


FIGURA 1.1: Robot móvil OpenROV

Si bien se cuenta con un conjunto de plataformas robóticas adaptadas para tareas de exploración, hasta el momento no se han desarrollados sistemas basados en visión que permitan incluirse en la etapa de navegación y mapeo de dichos robots, siendo la presente investigación la primera en abordar la tarea de la reconstrucción del suelo recorrido haciendo uso únicamente de una cámara de vídeo, sensor presente en todas las plataformas robóticas entes mencionados.

1.2. Justificación y planteamiento del problema

Cuando se habla de construir un mosaico 2D, se hace referencia al proceso de recortar y alinear imágenes, de tal forma que puedan ser representadas todas juntas en una sola gran imagen. Es importante considerar que las imágenes para este tipo de aplicaciones son capturadas desde diferentes ubicaciones de la cámara, a diferencia del proceso para elaborar imágenes panorámicas, en las cuales esta

[1] `https://www.openrov.com/products/openrov28/`

ubicación es una constante. Esta característica trae consigo uno de los principales problemas en la construcción de mosaicos, y se debe al efecto paralaje. Este efecto está asociado a la diferencia entre las posiciones aparentes de los objetos, según el punto desde donde se observa.

En la figura 1.2 se aprecia un ejemplo ilustrativo de esta definición, en la cual, si nos fijamos en el punto de vista **A**, se observa la estrella a la izquierda del circulo, mientras que desde le punto **B** este orden se encuentra invertido.

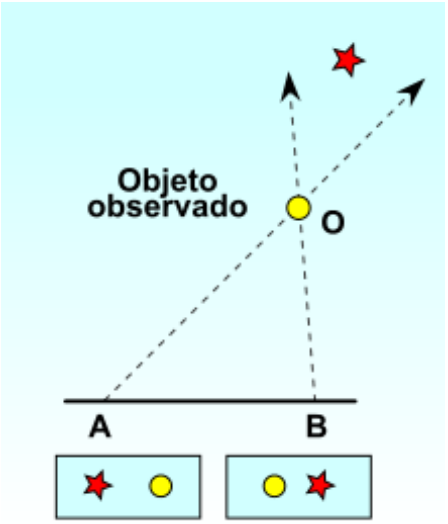


FIGURA 1.2: Efectos en el cambio del punto de vista.

Si bien, este problema afecta en gran medida la construcción del mosaico, no es el único presente, y se intensifican en aplicaciones de mapeo submarino, en las cuales, se evidencian efectos de distorsión de los objetos, absorción y cambios en la dirección de la luz, producto de pequeñas partículas suspendidas en el agua.

En el *GIDM* actualmente se utilizan mecanismos manuales para la elaboración de estos mapas, en especifico, se hace uso de *softwares* como Hugin[2]. Este es un programa de código abierto y gratuito bajo licencia GPL[3], el cual esta dedicado a la generación de imágenes panorámicas, incluyendo funciones para el recorte, alineación y corrección de color; además de algoritmos

FIGURA 1.3: Logo del software Hugin

[2] `hugin.sourceforge.net/`
[3] `http://www.gnu.org/copyleft/gpl.html`

para la optimización de parámetros en la cámara, y co-
rrección de distorsión. Si bien este software esta diseñado para la creación de imágenes panorámicas, permite el uso de varios tipos de proyecciones cartográficas, entre estas la rectangular, proyectando las imágenes sobre un plano recto.

Esta practica manual, además de limitar el alcance de los sistemas embebidos para el uso en navegación automática, requiere de una inversión de tiempo importante por medio del usuario en el proceso de selección y alineación de imágenes.

Atendiendo a esta necesidad, es necesario contar con un sistema que permita realizar la reconstrucción del suelo con la menor interacción posible del ser humano. Asimismo, con el fin de poder realizar operaciones de mapeo y localización simultanea *SLAM* (del ingles: Simultaneous Localization and Mapping), haciendo uso de las herramientas y robots existentes en el laboratorio, se requiere contar con un sistema basado en visión, que genere de forma automática un mapa 2D de la superficie sobre la que navega o sobrevuela el vehículo remoto, y que logre lidiar de manera efectiva ante los problemas previamente planteados.

1.3. Objetivos

1.3.1. Objetivo General

Analizar e implementar un sistema automatizado que permita la reconstrucción de un mapa en dos dimensiones, del suelo recorrido por robot, aéreo o submarino, a través de la información capturada por una cámara ubicada en su parte inferior.

1.3.2. Objetivos Específicos

- Análisis comparativo de métodos vigentes en la reconstrucción de mosaicos 2D, a partir de imágenes y videos de entrada.
- Implementación de modulo de pre-procesamiento y corrección de entrada.

- Análisis comparativo de métodos de detección y descripción de puntos característicos.
- Implementación de módulo de alineación de imágenes mediante la detección de puntos característicos.
- Cuantificar el error de proyección y distorsión en los modelos 2D generados.

1.4. Estructura del trabajo

Luego de presentar el planteamiento del problema y la descripción del proyecto, la presente investigación se encuentra dividida en 5 capítulos, organizados de la siguiente manera:

En el ***Capitulo 2*** se presenta una revisión del estado del arte sobre los algoritmos de generación de mosaico, en el cual se exponen los trabajos recientes y avances importantes en esta área de investigación. En ese mismo sentido, se describen los módulos principales que componen este tipo de sistemas.

El ***Capitulo 3*** inicia una revisión teórica en la cual se describe el funcionamiento de los algoritmos detectores, descriptores y emparejadores de características, y posteriormente se presentan resultados de pruebas comparativas entre los mas usados para este tipo de aplicaciones.

El módulo encargado de la alineación de imágenes en el mosaico, es descrito en el ***Capitulo 4***. Al igual que el capitulo anterior, se presenta una revisión teórica de los conceptos necesarios para su implementación. Luego, se introduce el modelo de sub mosaicos, y la implementación de un conjunto de correcciones geométricas sobre este nuevo modelo. Finalmente se muestran los resultados de los algoritmos aplicados en esta sección, seguidos de sus respectivos análisis.

En el ***Capitulo 5*** se describe el modulo final del sistema, en donde se explica el funcionamiento de los algoritmos que corrigen visualmente el mosaico final. De igual forma se muestran los resultados de su implementación, seguidos de una conclusión final sobre estos.

Finalmente, en el ***Capitulo 6*** se presentan las conclusiones finales, además de propuestas sobre recomendaciones y posibles implementaciones que pueden aportar mejoras y/o permitir la continuación del proyecto aquí planteado.

Capítulo 2

Sistemas de generación de mosaico

En este capítulo se presenta una revisión teórica del estado actual de las aplicaciones e investigaciones que se han desarrollado en el área de procesamiento de imágenes, aplicado a la construcción de mosaicos, además de una reseña histórica de la evolución de dichos métodos. Con esto se pretende recuperar y trascender el conocimiento acumulado en esta área de estudio, además de familiarizar al lector con los conceptos básicos, necesarios para la comprensión del presente trabajo. Para finalizar, se presenta un modelo del sistema de generación de mosaico a implementar en base a los algoritmos y técnicas que presentan las investigaciones mas recientes en esta área.

2.1. Estado del arte

La elaboración de mosaicos para la construcción de mapas del suelo, se ha desarrollado incluso antes desde la era digital de la computadoras. Desde que el proceso de registrar fotografías ha existido, se comenzaron a usar para elaborar mapas topográficos [3], donde imágenes adquiridas a partir de globos aerostáticos o altas colinas eran unidas manualmente. Posteriormente, producto de los avances en materia de aeronáutica, el interés por la aerofotografía se incrementó en gran medida. En este mismo sentido se utilizaban aviones para el registro de imágenes a

mayores altitudes, y se cubrían mayores áreas en menor cantidad de tiempo. Pero debido a que no se alcanzaban suficiente altura, y se mantenía la necesidad de registrar grandes áreas, era requerido que los mapas se construyan mediante fotografías que se superpongan, de igual forma esta tarea se llevaba a cabo mediante técnicas manuales por medio de expertos.

La necesidad de registrar áreas aun mas grandes siguió avanzando, motivado por la llegada de los satélites que eran capaces de enviar a tierra la información que obtenían de las cámaras. Los avances tecnológicos en materia de computación, y el creciente aumento de datos para esta aplicación, promovieron el desarrollo de técnicas de procesamiento digital de imágenes para dar solución a este tipo de problemas. En este sentido, distintos centros de investigación en el área de la física, robótica y visión por computadora, han aplicados sus esfuerzos en desarrollar algoritmos para la realización de estos mapas en ambientes mas desafiantes como lo son el fondo marino [4, 5, 6, 7].

Tal y como se ha mencionado el proceso de generación de mosaico involucra varios pasos principales, que los podemos definir como sigue: registro de imágenes, Alineación y fusión. Este proceso se ilustra gráficamente en la figura 2.1.

- **Registro:** De su termino en inglés *image-registration*, consiste en establecer la correspondencia geomántica entre las imágenes que componen la misma escena. Para esto, es necesario estimar la transformación geométrica que logra alinear dichas imágenes en el mismo plano.
- **Alineación:** También llamada proyección, consiste en alinear las imágenes registradas en un sistema de referencia común, es decir, con respecto a un plano re referencia. En este caso se utiliza la transformación geométrica calculada en el paso anterior.
- **Fusión:** En este paso se busca corregir los errores fotométricos o discontinuidades presentes en el mosaico luego del proceso de alineación. Estos errores aparecen, producto de errores en la estimación de las transformaciones o a cambios en la perspectiva de los objetos observados

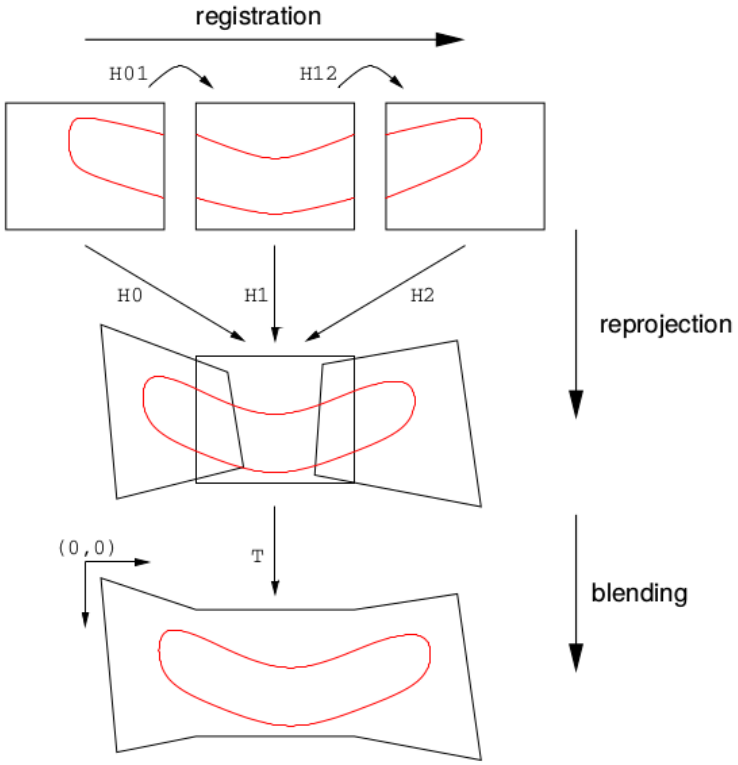


FIGURA 2.1: Proceso básico para la generación de mosaico

2.2. Esquema propuesto

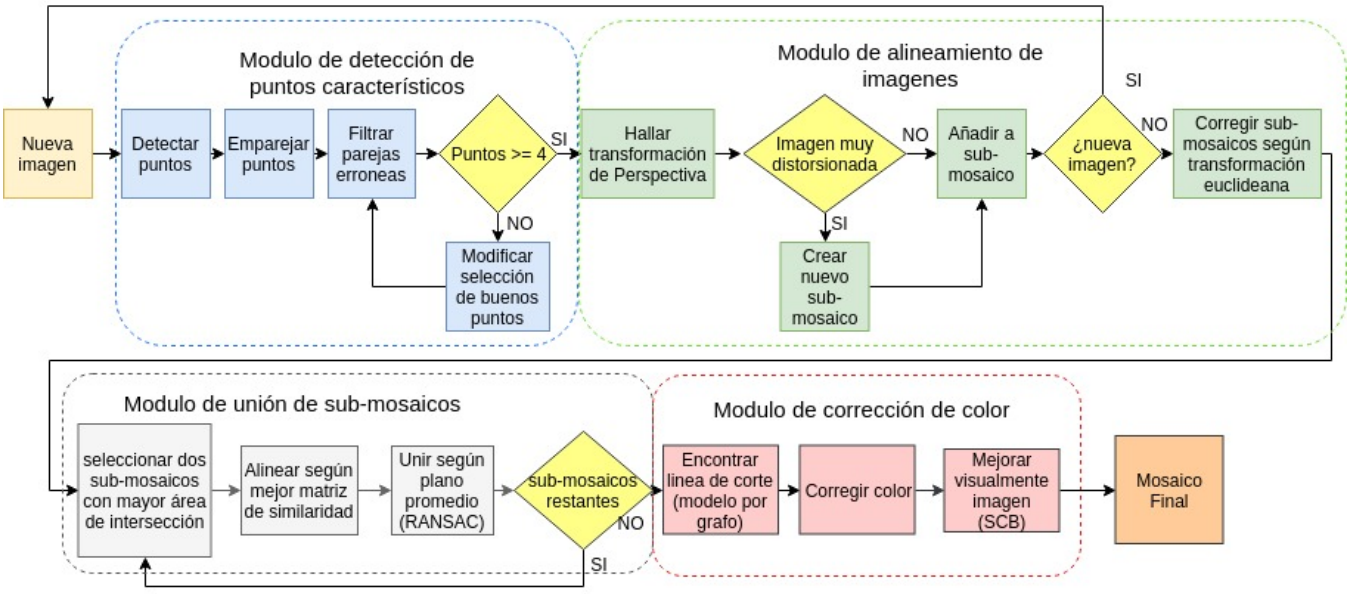


FIGURA 2.2: Esquema propuesto para la construcción del mosaico

2.3. OpenCV

Capítulo 3

Detección de puntos de interés

3.1. Introducción

3.2. Revisión teórica

3.2.1. Detectores y descriptores de características

Antes mencionar la evolución de los algoritmos de detección de puntos de interés, primero es necesario definir que son estos. Los puntos de interés, puntos clave, o *"features"* (en español: características) como son comúnmente llamados, son regiones en una imagen que contienen patrones específicos, lo que hace que puedan ser fácilmente seguidos o ubicados en otra imagen. Tuytelaars y Mikolajczyk [8] definen un punto característico local como *"un patrón en la imagen que difiere de su vecindario directo"*. De esta forma, se considera que los puntos característicos deben proporcionar la posibilidad de ser identificados en diferentes imágenes con el objetivo de emparejarlos.

Para alcanzar este objetivo los detectores y extractores de puntos característicos deben cumplir con ciertas propiedades que les permita funcionar bajo distintas condiciones, en concreto se busca que estos algoritmos cumplan con las siguientes propiedades:

- **Robustez:** El algoritmo debe ser capaz de detectar la misma ubicación del punto característico independientemente ante cambios en la escala, rotación, traslación, iluminación, transformaciones geométricas, artefactos de compresión y ruido.
- **Repetibilidad:** El algoritmo debe ser capaz de detectar el mismo punto característico de la misma escena bajo cambios en el punto de vista.
- **Exactitud:** El detector debe localizar el punto característico de manera precisa (misma ubicación de pixel). Especialmente para tareas de alineación de imágenes.
- **Generalidad:** El algoritmo debe ser capaz de detectar puntos que pueden ser usadas en distintas aplicaciones, es decir, que detecte varios tipos de características (esquinas, burbujas, etc.)
- **Eficiencia:** El algoritmo debe ser capaz de detectar puntos característicos en nuevas imágenes a gran velocidad, para soportar aplicaciones en tiempo real.
- **Cantidad:** El algoritmo debe detectar todos, o casi todos los puntos característicos presentes en la imagen.

Figura 3.1: Caracterización de regiones en una imagen

Llegados a este punto es necesario definir el funcionamiento de los algoritmos detectores, y que consideran estos como puntos característicos, basados en la definición previamente planteada. Atendiendo a la imagen 3.1, se puede observar que

se caracterizan seis áreas de interés. Analizando estos segmentos, vemos que ***A*** y ***B*** corresponden con superficies planas, lo que hace que sea muy difícil identificar la ubicación exacta de estas superficies en la imagen original. Por otro lado, tenemos las regiones ***C*** y ***D***, las cuales corresponden con bordes en la imagen, si bien, se puede limitar en gran medida el área de búsqueda hacia toda las regiones del mismo bordes, sigue siendo difícil acertar con la ubicación correcta. Por ultimo, analizando las regiones ***E*** y ***F*** tenemos que corresponden a esquinas de la imagen original, en este caso se puede identificar fácilmente la ubicación exacta de la región en la imagen.

A partir de esta idea, en la cual se consideran las esquinas como regiones fácilmente identificables en una imagen, en *1988* nace el primer algoritmo de detección de puntos de interés llamado Detector de esquinas de Harris [9] (nombre original en inglés: Harris Corner Detector), y como su nombre lo indica está basado en la detección de esquinas.

Retomando el concepto planteado previamente, este detector busca la diferencia de intensidad de una región con su entorno directo, es decir, se detectará una esquina para aquellas regiones que presenten una alta variación de intensidad, al desplazar la ventana estudiada en cualquier dirección. En la figura 3.2 se puede apreciar visualmente como funciona esta ventana de búsqueda.

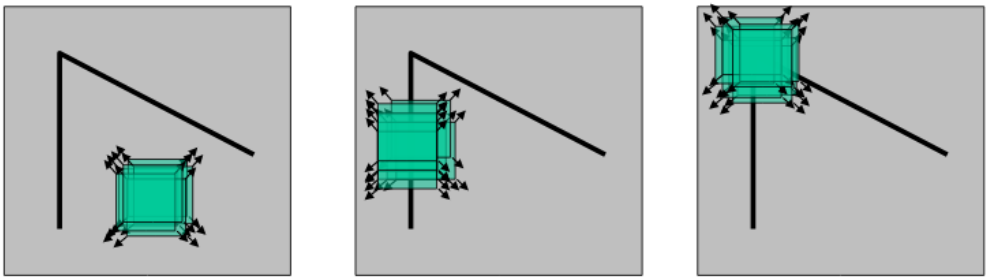


FIGURA 3.2: Ventana de búsqueda para de detección de esquinas

Cuando se trabajan con detectores de características, se desea que estos sean invariantes ante la mayor cantidad de variables posibles, tal y como se menciona en la propiedad de robustez que deben tener estos algoritmos. Si bien el detector presentado anteriormente es invariante ante la traslación y la rotación (ya que las esquinas se mantienen como esquinas si son rotadas o desplazadas), no funciona

de la misma forma ante cambios de escala. Como se observa en la figura 3.3, una región considerada como esquina, se podría considerar plana si es ampliada.

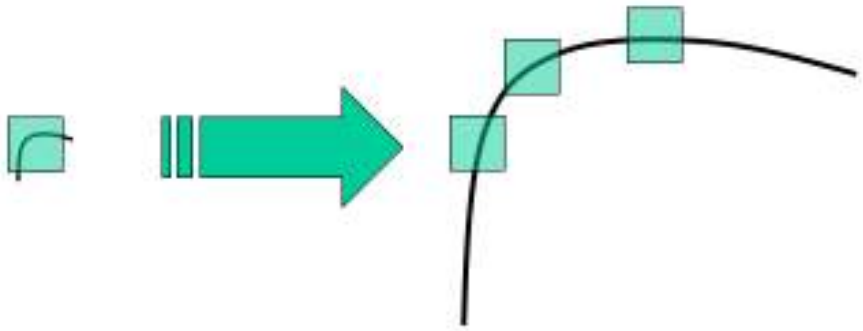


FIGURA 3.3: Efecto del escalado sobre las esquinas

Con el fin de conseguir detectar los mismos puntos ante cambios en la escala de la imagen, Lindeberg, T. [10] propone un algoritmo detector de "manchas" multi-escala a través de la búsqueda de un máximos en el espacio de escala, el cual se crea utilizando un operador laplaciano. El Laplaciano de Gaussianas LoG (del ingles: Laplacian-of-Gaussian), es una combinación lineal de segundas derivadas utilizado para detectar burbujas o manchas en una imagen. El funcionamiento es el siguiente: Dada una imagen de entrada, la representación para cada escala $-s-$ de la imagen se define como la convolución de la imagen con un filtro Gaussiano con desviación estándar s.

Este resultado brinda una fuerte respuesta positiva para burbujas oscuras y respuestas fuertes negativas para burbujas claras, ambas de un tamaño $2s$, donde s es la escala. De esta forma las caracteristicas detectadas presentan una fuerte relación entre el tamaño de las estructuras en la imagen y el grado de difusión del filtro gaussiano. Donde la desviación estándar del filtro se usa para controlar la escala cambiando que tanto se difumina la imagen.

Llegados aquí, una vez se haya detectado la ubicación de los puntos característicos en la imagen, la información de la localidad de este debe ser codificada y almacenada, y de esta forma lograr tener un descriptor único de la región con el objetivo final de ubicarlo en otra imagen. Con este fin se desarrollaron los algoritmos descriptores, los cuales una vez tengan la ubicación de los puntos característicos se encargan de convertir la información de su alrededor en una serie de números, o

un vector que permita diferenciar un punto clave de otro. Esta información también es necesaria que sea invariante ante las variable mencionadas previamente, para lograr una identificación eficiente del mismo punto en distintas imágenes bajo distintas condiciones.

Partiendo de estos problemas, y del hecho que el calculo del operador LoG es computacionalmente costoso, en *2004 D. Lowe* crea el detector y descriptor *SIFT* [11] (del inglés: Scale Invariant Feature Transform), en el cual el espacio de escala es construido en forma piramidal con la diferencia de gaussianas DoG (del inglés: Difference of Gaussians). En este sentido, El operador DoG ofrece una aproximación al LoG, donde se calcula sin convolución restando niveles de escala adyacentes de una pirámide gaussiana. El proceso para la detección y descripción de puntos de interés de este algoritmo, consta de cuatro pasos principales:

En primer lugar, realiza una detección de máximos en el espacio de la escala aplicando la diferencia gaussiana *DoG*. Para esto, se aplica el filtro gaussiano con distintos tamaños de media (se tienen distintas escalas), luego restando estas imágenes para distintos pares de escalas se logra la diferencia de gaussiana. Posteriormente se buscan los máximos locales a lo largo del espacio (coordenadas X,Y) para cada correspondiente escala. Este proceso de detección se puede visualizar en la figura 3.4.

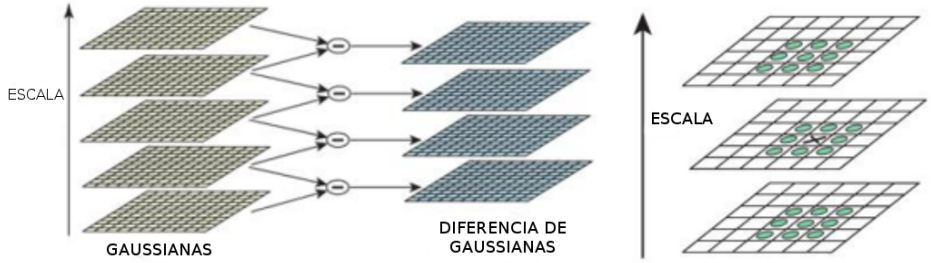


Figura 3.4: Detección de máximos en el espacio de la escala DoG

En segundo lugar para la localización de puntos de interés, se descartan los puntos encontrados en el paso anterior que no superen cierto valor de umbral, es decir, que no estén lo suficientemente contrastados con su entorno. Con esta etapa el algoritmo solo toma en cuenta los puntos claves mas fuertes por cada escala. Además, con el objetivo de eliminar los bordes suficientemente contrastados que no correspondan con esquinas, el algoritmo usa una matriz hessiana para calcular las curvaturas principales, y así quedarse solo con esquinas.

Para garantizar la invarianza con respecto a la rotación, se toman los píxeles vecinos al punto clave y se calcula la magnitud y dirección del gradiente en esa región. Con esto se hace un histograma de la magnitud del gradiente en cada dirección, donde el pico mayor del histograma indica la orientación. En el caso que exista un pico mayor al 80 % del pico principal, este se utiliza para crear otro punto de interés en la misma posición pero con la distinta rotación.

Finalmente para crear el vector descriptor por cada punto clave se crea una matriz de 16x16 alrededor de este, dividida en 4 subregiones de 4x4 píxeles con un histograma de orientaciones para cada uno. Seguidamente, el descriptor del punto será el vector con los valores de los histogramas de las regiones 4x4 concatenados. La figura 3.5 la representación del descriptor de SIFT.

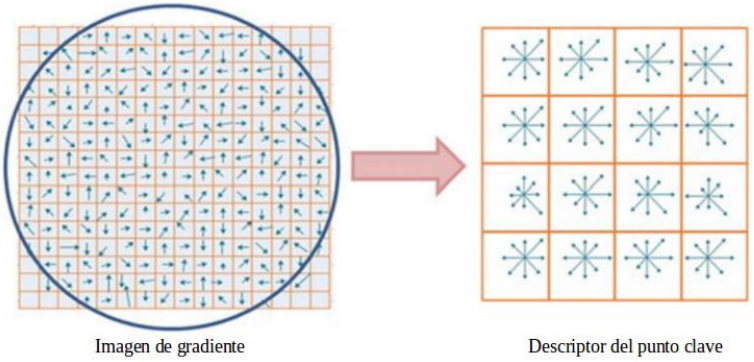


FIGURA 3.5: Descriptor del algoritmo SIFT

En el año 2006, un grupo de tres personas Bay, H., Tuytelaars, T. and Van Gool, L. desarrollan *SURF* [12], el cual es un detector y descriptor de características basado en SIFT, pero con modificacioes que aumentan su velocidad de detección. Si bien, sacrifica un poco de rendimiento y precisión, lo hace mas provechoso para aplicaciones embebidas que demanden mayor velocidad de computo y menor uso de recursos, como por ejemplo *SLAM*. El proceso para la extracción de características por parte de este algoritmo se compone de los siguientes pasos:

Como primer paso, en lugar de aproximar el laplaciano de Gauss *LoG* (del inglés: Laplacian of Gaussians) con la diferencia de Gaussianas (DoG) como lo hace SIFT, este algoritmo aproxima LoG con cuadrados para promediar la imagen. La ventaja de aplicar filtros con cuadrados es que con la ayuda de imágenes integrales el cálculo computacional se reduce en gran medida.

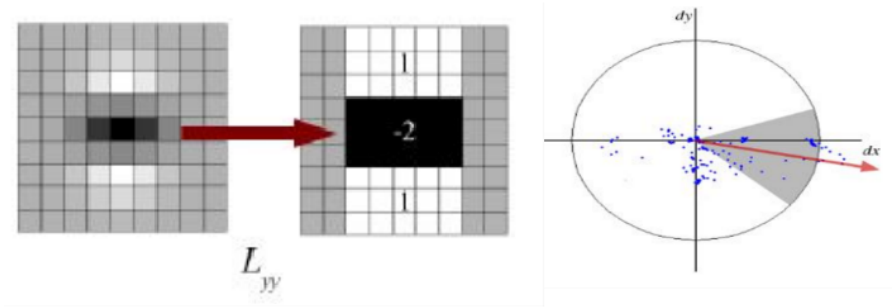


FIGURA 3.6: Deteccion y descripción mediante el algoritmo SURF

En función de identificar la orientación, el algoritmo utiliza la respuesta wavelet Haar en horizontal y vertical en un vecindario de $6s$ (donde s es la escala evaluada) píxeles al rededor del punto de interés, Luego estas respuestas son representadas como puntos en el espacio, para luego calcular la orientación dominante con la suma de todos los resultados dentro de una ventana deslizante de apertura 60°. En la figura 3.6 se puede visualizar en el lado izquierdo, la aproximación que realiza de la derivada de segundo orden del filtro gaussiano, y su aproximación con un filtro cuadrado. Del lado derecho se ilustra el vector de orientación en función a la distribución de puntos estudiados.

El siguiente avance importante en los algoritmos de detección aparece en el año 2011 con *ORB* [13] (del inglés: Oriented FAST and Rotated BRIEF), este utiliza una combinación del detector FAST (del inglés: Features from Accelerated Segment Test) y del descriptor BRIEF (del inglés: Binary Robust Independent Elementary Features), este nuevo algoritmo esta caracterizado por su alta velocidad de procesamiento manteniendo un buen rendimiento, gracias al uso de un descriptor binario.

Como se mencionó utiliza el algoritmo FAST, el cual consiste en encontrar esquinas evaluando los píxeles en un perímetro circular, de esta forma, un punto será detectado como esquina si la cantidad de píxeles de color opuesto al evaluado, supera cierto valor de umbral (ver izquierda en la figura 3.7), posteriormente con el fin de aumentar la robustez, es aplicado el algoritmo de clasificación de esquinas de *Harris*. De igual forma se realiza con una estructura piramidal evaluando varias escalas (al igual que SIFT).

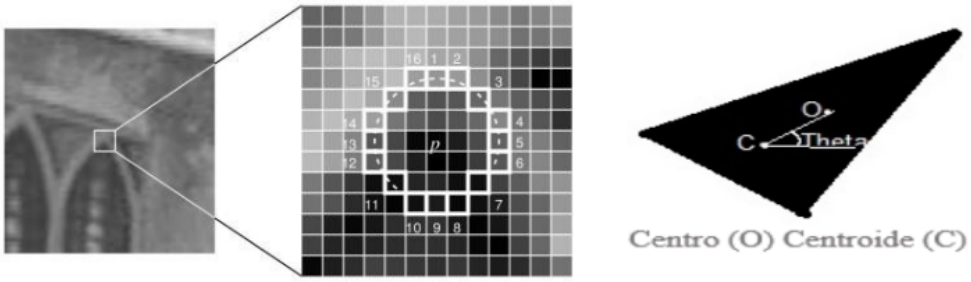


FIGURA 3.7: Deteccion y descripción mediante el algoritmo SURF

Como el algoritmo FAST no toma en cuenta la orientación, en el ORB se modificó para que calculara la orientación de la siguiente forma: Se considera una región ubicada en el centro del punto estudiado, luego se calcula el centroide de la región en función a la intensidad de los puntos. De esta forma, la dirección del vector desde el punto central hasta el centroide es asignado como vector de orientación. Observando a la derecha en 3.7 se aprecia un ejemplo del lugar del centroide *(C)* y del centro *(O)* par auna región en particular.

Para el descriptor utiliza BRIEF, a diferencia de los anteriores (SIFT y SURF) este es un descriptor binario y no vectorial. El descriptor BFIEF produce una palabra de n-bits usando el algoritmo *Local Binay Tests* (LBT), el problema de esta representación es que no es muy robusta ante cambios en la rotación. Para resolver esto ORB utiliza la información de la orientación previamente calculada en el paso de detección para aplicar LBT en esa orientación.

Los algoritmos de detección que se mencionaron hasta este momento tienen una caracteristica en común, y es que cuando trabajan con el esquema piramidal lo hacen bajo el espacio de escala Gaussiano, el cual es una instancia particular de difusion lineal. De esta forma, al utilizar este filtro no se respetan los limites naturales de los objetos y se difumina del mismo nivel toda la region de la imagen cuando se avanza entre nieveles de escala.

Enfocándose en esta característica, en el año de *2012* se desarrolla el detector y descriptor llamado KAZE [14] por parte de *Pablo Fernández Alcantarilla*. Este novedoso algoritmo opera completamente en un espacio de escala no lineal, y para ello utilizan un esquema de división de operadores aditivos (*AOS*, del inglés:

Additive Operator Splitting), que les permite obtener espacios de escala no lineales de forma eficiente. De este modo se puede realizar un difuminado localmente adaptativo, posibilitando que se remueva el ruido en las imágenes, manteniendo información importante sobre los bordes de los objetos al avanzar en el espacio de escala. En la figura 3.8 se puede observar como afecta en los bordes de los objetos el aplicar un filtro de difusión lineal, y uno que no lo es, bajo el esquema propuesto por este algoritmo.

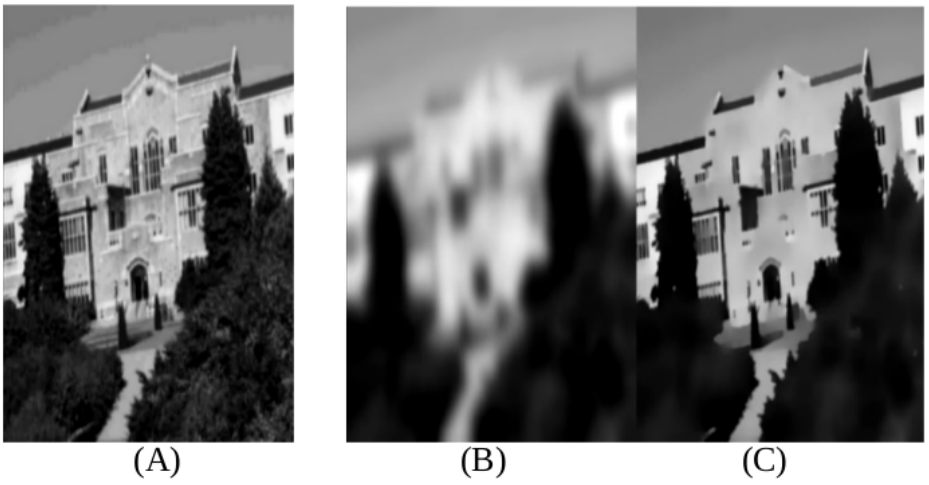


FIGURA 3.8: *(A)*: imagen original, *(B)* filtro lineal Gaussiano, *(C)* filtro no lineal usado en KAZE

Bajo este mismo esquema de difusión no lineal, el mismo autor en el año *2013* desarrolla la versión acelerada de este algoritmo que recibe el nombre de *A-KAZE* [15] (del ingles: Accelerated KAZE). Esta mejora se utiliza un esquema basado en difusión explícita rápida *FED* (del ingles: Fast Explicit Difussion) en lugar de *AOS*, el cual es un nuevo esquema piramidal que incrementa en gran medida la velocidad de computo para construir el espacio de escala no lineal.

Para el calculo de la orientación el primer algoritmo *KAZE* utiliza un descriptor para la orientación similar al que emplea SURF. Este encuentra la orientación dominante en un área circular de radio $6s$ (s corresponde con la escala), y para cada muestra del círculo se calcula la derivada de primer orden en las direcciones X e Y, y se ponderan con una gaussiana centrada en el punto de interés. Luego, las respuestas de estas derivadas son representadas como puntos en un espacio vectorial, donde la orientación dominante se haya sumando las respuestas dentro de un segmento de circulo deslizante con apertura de 60°.

Por otro lado, la versión acelerada *A-KAZE* emplea un descriptor basado en una versión modificada del algoritmo de diferencia local binaria *LDB* [16] (del ingles: Local Difference Binary), llamado M-LBD (del ingles: Modified Local Difference Binary), el cual aprovecha al máximo la información del espacio de escala no lineal. La modificación consiste en hacer un sub-muestreo de cada región que divide la zona del descriptor, en lugar de calcular el promedio de todos los píxeles de la región, es decir, se tienen muestras de cada subdivisión para distintas escalas.

3.2.2. Emparejadores de puntos característicos

Emparejadores

3.3. Implementación y resultados de modulo comparativo

3.4. Conclusiones

Resumen

Capítulo 4

Alineación de imágenes

4.1. Introducción

Introducción

4.2. Revisión teórica

Prueba

4.2.1. Transformaciones geométricas

4.3. Generación de sub-mosaicos

Prueba

4.3.1. Selección de imagen de referencia

Prueba

4.3.2. Matriz de transformación promedio

Prueba

4.4. Corrección euclidiana

Prueba

4.5. Resultados

Resumen

4.6. Conclusiones

Resumen

4.7. Algoritmos (temporal)

Algoritmo 1: Calculo de matriz de homografia promedio

mientras *no se alcanza el maximo de iteraciones* **hacer**
    seleccionar 4 puntos aleatorios del primer sub-mosaico;
    seleccionar los 4 puntos correspondientes en el segundo sub-mosaico;
    calcular el punto medio para cada par de puntos correspondientes;
    calcular la transformacion desde los puntos del primer sub-mosaico hasta los puntos medios;
    aplicar transformacion en el primer sub-mosaico;
    calcular error de distorsión en el primer sub-mosaico;
    si *el error es menor que el mas bajo obtenido* **entonces**
        guardar el error como el mas bajo;
        guardar la matriz de transformación como la mejor;
    en otro caso
        restaurar valores del primer sub-mosaico;
    fin
fin

Algoritmo 2: Encontrar matriz de transformacion

$I_{i+1} \equiv$ Imagen nueva;
$I_i \equiv$ ultima imagen anadida al mosaico;
$V_i \equiv$ vecinos de I_i;
mientras *puntos emparejados* $\geq$ *4* **hacer**
    Emparejar puntos de I_{i+1} con I_i;
    si I_i *tiene vecinos* **entonces**
        para cada *vecinos de* I_i **hacer**
            Emparejar puntos de I_{i+1} con V_i;
        fin
    fin
    descartar malos emparejamientos; aplicar busqueda sectorizada; **si** *puntos totales emparejados* $\leq$ *3* **entonces**
        modificar criterio para descartar;
        si *criterio para descartar llega al minimo* **entonces**
            terminar ; `// no es posible emparejar imagen`
        fin
    fin
fin

Capítulo 5

Unión de imágenes

5.1. Introducción

Prueba

5.2. Linea de costura

Prueba

5.2.1. Mapa mas cercano

Prueba

5.2.2. Corte por grafo

Prueba

5.3. Corrección de color

Prueba

5.3.1. Ajuste de ganancia

Prueba

5.3.2. Método de Reinhard

Prueba

5.4. Fusión de imágenes

5.4.1. Fusión ponderada

Prueba

5.4.2. Fusión piramidal

Prueba

5.5. Resultados

5.6. Conclusiones

Capítulo 6

Conclusiones y trabajos futuros

Conclusiones

Bibliografía

[1] Danilo Díaz Tarascó. Desarrollo de sistema de telemetria para teleoperación de robot submarino, 2016.

[2] Said Alexander Alvarado Marín. Caracterización, instrumentación y control de robot cuadricóptero volador, 2017.

[3] Paul R. Wolf. *Elements of Photogrammetry (Second Edition)*. McGraw-Hill Higher Education, 1983.

[4] N. Gracias y J. Santos-Victor. Underwater mosaicing and trajectory reconstruction using global alignment. *MTS/IEEE Conference and Exhibition*, 2001.

[5] H Pizarro, O. y Singh. Seabed video mosaicking with matisse: a technical overview and cruise results. *International Offshore and Polar Engineering Conference*, 2004.

[6] Pizarro O. Singh-H. y Howland J. Eustice, R. Underwater image toolbox for optical image processing and mosaicking in matlab. *International Symposium on Underwater Technology*, 2002.

[7] Borgetto M.-Opderbecke J. Pessel N. y Rigaud V Allais, A. G. Toward large-area mosaicing for underwater scientific applications. *IEEE Journal of Oceanic Engineering*, 2003.

[8] Tinne Tuytelaars y Krystian Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 2008.

[9] Hans P. Morevec. Towards automatic visual obstacle avoidance. 1977.

[10] T Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 1998.

[11] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Computer Science Department, University of British Columbia*, 2004.

[12] Tuytelaars T. y Van Gool L Bay, H. Surf: Speeded up robust features. *European Conference on Computer Vision*, 2006.

[13] Kurt Konolige y Gary Bradski Ethan Rublee, Vincent Rabaud. Orb: an efficient alternative to sift or surf. *International Conference on Computer Vision*, 2011.

[14] y Andrew J. Davison Pablo Fernandez Alcantarilla, Adrien Bartoli. Kaze features. *European Conference on Computer Vision*, 2012.

[15] Pablo Fernández Alcantarilla. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *British Machine Vision Conference (BMVC)*, 2013.

[16] Kwang-Ting Cheng Xin Yang. Local difference binary for ultrafast and distinctive feature description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

Apéndice A

Instalación de librería OpenCV y dependencias

A.1. @sección

A.1.1. @subsección

"Saludo".