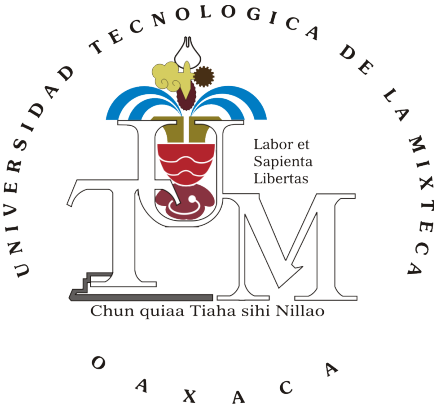


UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

Implementación de un Sistema de Visión Mono-Cámara Basado en el Registro de Imágenes 2D.

T E S I S

Para obtener el título en:

INGENIERO EN MECATRÓNICA

Presenta:

ARMANDO LEVID RODRÍGUEZ SANTIAGO

Director de Tesis:

DR. ROSEBET MIRANDA LUNA

Asesor de Tesis:

DR. ANTONIO ORANTES MOLINA

Huajuapan de León, Oaxaca, México, Julio de 2013

Tesis presentada el 5 de Julio de 2013
ante los siguientes sinodales:

Dr. Enrique Guzmán Ramírez.
Dr. José Aníbal Arias Aguilar.

Director de Tesis:
Dr. Rosebet Miranda Luna.
Asesor de Tesis:
Dr. Antonio Orantes Molina.

A mi Madre con la mayor gratitud por todos sus esfuerzos para que yo pudiera terminar mi carrera profesional. Por haberme dado todo y por enseñarme a luchar por lo que se quiere.Gracias por guiar mi camino y estar siempre junto a mi.

Mi triunfo es tuyo.

Agradecimientos

Gracias a mis mamas Alberta y Lourdes, quienes con su esfuerzo, desvelo, esmero e ilusión me han dado una carrera profesional, la mejor herencia que se le puede dar a un hijo. Pero sobre todo porque han hecho de mi una gran persona. **Gracias Mamas!**.

Agradezco a mis tíos y tías, Elías, su esposa Magdalena, Miguel Ángel y Guadalupe, Erasto y Teresa. Alberto y Gabriela. Quienes con su ejemplo y apoyo me han enseñado a siempre seguir adelante a pesar de los obstáculos que la vida presenta.

Gracias a mis hermanos Oscar, Evelyn, Emmanuel y a todos mis primos, quienes siempre me han dado motivos para sonreír y no olvidar lo más importante de la vida, la familia. Que este triunfo les motive a seguir adelante y hacer realidad sus sueños.

Gracias al Dr. Rosebet Miranda Luna, director de esta tesis, por su dedicación, tiempo y los conocimientos que me deja ya que han hecho de mi un mejor estudiante y una mejor persona.

A mi asesor y sinodales el Dr. Antonio Orantes, Dr. Enrique Guzmán y Dr. José Aníbal. Por sus valiosos consejos, observaciones y exigencias para que esta tesis se haya realizado con gran calidad, gracias.

A todos mis amigos que siempre me dan la fuerza para continuar a pesar de las piedras en el camino durante toda mi carrera profesional, en el desarrollo de esta tesis y en mi vida, muchas gracias.

Gracias a dios por darme la oportunidad de terminar mi carrera profesional y a todas esas personas que de forma directa o indirecta participaron con sus valiosos consejos, apoyo y motivación durante mis estudios universitarios y en especial en esta tesis, sin todos ustedes no habría sido posible este éxito.

A Todos...**GRACIAS!**

Resumen

En esta tesis se presenta la implementación de un sistema de visión por computadora para la estimación de la ubicación y la aproximación a una referencia deseada respecto a un objetivo en un entorno estructurado de una plataforma móvil. Este sistema se basa en el procesamiento digital de imágenes 2D en niveles de gris, utilizando un algoritmo de búsqueda exhaustiva optimizado con una estrategia piramidal, la suma de la diferencia de intensidades como medida de similitud e implementado en MATLAB.

Se considera el caso en el que la plataforma móvil se encuentra cercana y de frente al objetivo, que en este caso es una escena plana y una distancia de referencia respecto al objetivo de *100cm*. Además se presenta un método sencillo de calibración para la cámara web utilizada.

Índice

Índice de figuras

Índice de Tablas

Capítulo 1

Introducción

La robótica es una rama de la tecnología que ha presentado un auge sumamente considerable en las últimas décadas. Una de sus áreas de aplicación es la navegación autónoma de robots móviles, tema de investigación muy amplio y muy abordado por investigadores en diferentes partes del mundo [Auat et al., 2011][Raguraman et al., 2009][MacMillan et al., 2011]. Muchas veces se requiere que un robot, móvil o manipulador, realice diversas tareas, para lo cual, es preciso conocer la ubicación actual del robot y la posición a la cual se desea llegar para ejecutar las tareas deseadas, una posible solución es mediante un sistema de visión artificial basado en el registro de imágenes.

El registro de imágenes consiste en determinar las transformaciones geométricas que ponen en correspondencia las estructuras respectivas de dos imágenes de una misma escena adquiridas ya sea en diferentes instantes de tiempo o, desde posiciones distintas. Las aplicaciones del registro de imágenes son amplias y en cada caso se requiere una metodología bien adaptada [Li et al., 1995][Posada et al., 2004][Sarrut, 2000][Vassal, 1998].

1.1. Planteamiento del Problema

Para que un robot, móvil o manipulador, realice tareas como tomar y/o transportar objetos, realizar ensambles, abrir o cerrar puertas, entre muchas otras tareas de servicio, es necesario que se ubique en una posición especifica de manera que pueda ejecutar las tareas requeridas. Para lograr el posicionamiento en el lugar preciso y de forma autónoma dentro de un entorno estructurado, es necesario conocer su posición actual con respecto a un marco de referencia, la ubicación en la cual se desea posicionar, así como la trayectoria de desplazamiento.

La Universidad Tecnológica de la Mixteca cuenta con la plataforma Zagros Max 99. Se trata de un móvil que cuenta con cuatro ruedas, dos de ellas son de tracción y dos más para su estabilidad [Robotics, 2002]. Esta plataforma carece de sensores que le permitan conocer su ubicación en el espacio, de manera que los desplazamientos a partir de una posición de referencia se determinan mediante los tiempos de activación de sus motores de tracción, sin embargo, debido a su arquitectura diferencial, a la imperfección de su sistema de tracción

influenciado por las características de la superficie por la cual se desplaza, para iguales tiempos de activación de cada motor, el vehículo describe trayectorias curvas en lugar de rectas, resultando prácticamente imposible programar los tiempos para llevar al móvil con precisión a una posición final deseada.

Con este trabajo de tesis se desea dotar a la plataforma Zagros Max 99 con un sistema de visión mono-cámara de bajo costo, basado en el registro de imágenes 2D, que le permita referenciarse con respecto a un objetivo (escena plana) y eventualmente reducir su distancia con respecto a una posición final deseada. Para limitar la complejidad de este trabajo se considerará únicamente el caso en el que el móvil se encuentra cercano y de frente al objetivo, de manera que se pueda considerar despreciable cualquier efecto de perspectiva en las imágenes.

1.2. Justificación

La plataforma Zagros Max 99 constituye una opción a nuestro alcance para el estudio de estrategias de navegación y posicionamiento, siempre que sea equipada con los sensores necesarios.

De entre las diversas alternativas posibles y como tema de investigación, se desea experimentar y determinar los alcances del registro de imágenes 2D en niveles de gris, empleando una transformación rígida y una búsqueda exhaustiva de los parámetros de transformación en combinación con la estrategia piramidal como herramientas para la ubicación y posicionamiento autónomo de un vehículo con las características de dicha plataforma. Esta alternativa de registro de imágenes fue considerada ya que se desea cubrir un rango relativamente amplio para los parámetros de transformación involucrados, además de minimizar el tiempo de cómputo.

1.3. Objetivos

1.3.1. Objetivo General

Dotar a la plataforma móvil Zagros Max 99 de un sistema de visión basado en el registro de imágenes 2D, utilizando una cámara web de bajo costo, que permita evaluar la factibilidad de determinar su ubicación con respecto a un objetivo (escena de referencia plana) ubicado frente a él y de disminuir el error en su posicionamiento.

1.3.2. Objetivos Específicos

Para cumplir con el objetivo establecido, se proponen los siguientes objetivos específicos:

- ✓ Calibrar la cámara web Perfect Choise modelo PC 320425.
- ✓ Aprender el funcionamiento del Vehículo Móvil Zagros Max 99.

- ✓ Implementar el sistema de adquisición de imágenes mediante MatLab.
- ✓ Implementar en MatLab un algoritmo para el registro de imágenes 2D, basado en la suma diferencia de intensidades, en una estrategia de optimización piramidal, y utilizando un modelo de transformación rígido.
- ✓ Implementar rutinas de desplazamientos y giros en base a la estimación de su posición para la disminución del error de posicionamiento.
- ✓ Desarrollar una interfaz de usuario e integrar las diferentes etapas del software en un solo algoritmo y acoplarlo al vehículo móvil.
- ✓ Determinar con precisión el radio de acción del sistema de visión implementado.
- ✓ Realizar pruebas y evaluar el desempeño del sistema en la estimación y en la corrección de la ubicación del móvil para la navegación autónoma del robot Zagros Max 99.

1.4. Propuesta de solución

Los sistemas de visión y esta propuesta de trabajo se desarrolla en tres etapas principales, la primer etapa consta de la calibración de la cámara a utilizar, donde se determina la relación ***pixeles-centímetros*** es decir, la relación que guarda los pixeles desplazados en una imagen y las distancias del mundo real. Se emplea un método sencillo de calibración para el factor de escala y de desplazamientos, además este método puede ser extendió a otros parámetros de transformación.

La segunda etapa corresponde al registro de imágenes. Considerando como referencia a una escena plana, se opta por una metodología de registro de imágenes 2D y un modelo de transformación proyectivo (matriz de transformación homogénea). Se obtienen los parámetros de transformación para realizar la correspondencia entre la imagen de referencia y la capturada por el sistema en su posición actual.

La tercer etapa corresponde a la aplicación, donde se conocen los alcances del software de registro de imágenes implementado y además con la información obtenida en las etapas anteriores reducir el error de posicionamiento de una plataforma móvil a una distancia de referencia deseada respecto a un objetivo el cual se considera como una escena plana.

1.4.1. Calibración

La etapa de calibración consiste en encontrar la relación matemática entre los pixeles desplazados en una imagen y las distancias en el mundo real. Antes de iniciar con el proceso de calibración es necesario realizar la alineación del eje óptico de la cámara localizando el centro geométrico de la imagen capturada por la cámara mediante el método desarrollado por Willson.

El proceso realizado para encontrar la relación ***pixeles-centímetros***, consiste en la capturan de una serie de imágenes a distancias y desplazamientos conocidos respecto al objetivo, posteriormente se determinan los parámetros de transformación geométrica que ponen en correspondencia dos imágenes, los cuales pueden ser utilizados para estimar la posición relativa desde donde fue adquirida una de las imágenes con respecto a la otra, por ejemplo, un acercamiento de la cámara con respecto a una escena de referencia que se pueda considerar plana, se traduce en un cambio en el factor de escala de la matriz de transformación.

Con estos datos se realizan las aproximaciones lineales Y y D (Ecuaciones 3.1 y 3.5), con la cual estimar la posición real del sistema de visión.

1.4.2. Registro de imágenes

El registro de imágenes consiste en la puesta en correspondencia de la imagen capturada en la posición actual del sistema y la imagen de referencia utilizada.

El registro de imágenes se realiza mediante el algoritmo de búsqueda exhaustiva y una estrategia de optimización piramidal. El algoritmo de registro de imágenes implementado realiza una transformación polar logarítmica sobre la imagen de referencia y las imágenes transformadas con los parámetros de la matriz de transformación homogénea y utilizando la suma de diferencia de intensidades como medida de similitud para garantizar el registro de imágenes correspondiente.

Una vez realizado el registro de imágenes se obtiene el factor de escala A y la cantidad de pixeles correspondiente al desplazamiento P_x para poner en correspondencia las dos imágenes, los cuales son suministrados a las aproximaciones lineales obtenidas en la etapa de calibración (Y y D) y así, poder estimar la distancia real a la cual el sistema se encuentra ubicado.

1.4.3. Aplicación

Para determinar los alcances del software de registro de imágenes se trabaja tanto en condiciones ideales como reales.

Primeramente el software es puesto a prueba con imágenes ideales, donde no existen perturbaciones que pudieran influir en la estimación de los parámetros de transformación, y así determinar el rango de operación del software implementado. En seguida se realizan pruebas con imágenes capturadas por la cámara utilizada y de igual manera se determina el rango de operación del software para estimar la posición de la cámara.

Una vez estimada la ubicación de la cámara y dado que se encuentra fija a la plataforma móvil, se procede a compensar la diferencia entre la distancia actual y la de referencia. Esto se realiza manipulando la plataforma móvil bajo un algoritmo empírico, basado en las distancias obtenidas y realizando tres iteraciones. Obteniendo así, el rango de operación del software para determinar y eventualmente reducir el error de posicionamiento de una plataforma móvil.

La Figura 1.1, muestra la interacción de estas tres etapas, que en su conjunto realizan un sistema de visión capaz de reducir el error de posicionamiento de la plataforma móvil respecto a una distancia de referencia deseada.

El software ordena a la cámara capturar la escena actual de su posición y le proporcione además esta imagen. También ordena a la plataforma móvil que ejecute las rutinas de desplazamiento necesaria y así poder reducir el error de su posicionamiento respecto a la distancia de referencia deseada.

Figura 1.1: Etapas de la Propuesta de Solución

1.5. Contenido del Documento

En este documento de tesis encontramos en el Capítulo 2 las bases teóricas necesarias para el desarrollo de este trabajo de investigación, donde se incluyen el estudio de la geometría proyectiva, el registro de imágenes y la estrategia piramidal, además también la revisión del estado del arte.

El Capitulo 3 esta dedicado a la implementación del sistema, donde se describe el método de calibración desarrollado para la cámara, el software de registro de imágenes implementado, el conocimiento de los distintos elementos del sistema y la manera de integración de todos ellos.

Los resultados obtenidos son presentados en el Capítulo 4, donde se muestran los alcances del software de registro de imágenes implementado y los resultados obtenidos del funcionamiento del sistema.

Por ultimo en el Capitulo 5 se presentan las conclusiones generadas después del análisis de los resultados obtenidos y se plantean algunos posibles trabajos futuros para el sistema desarrollado.

Capítulo 2

Marco Teórico Y Estado del Arte

2.1. Marco Teórico

Para la realización de esta tesis son necesarias las bases teóricas que a continuación se presentan.

2.1.1. Robótica

La robótica es una rama de la tecnología que, en los últimos años, ha presentado un auge sumamente considerable.
La real academia de la lengua española define a la robótica como [RAE, 2012]:

"Técnica que aplica la informática al diseño y empleo de aparatos que, en sustitución de personas, realizan operaciones o trabajos, por lo general en instalaciones industriales"

Estos aparatos son comúnmente llamados robots, pero, ¿qué es un robot? Algunas definiciones:

- ✓ Del ingl. robot, y este del checo robota, trabajo, prestación personal. Máquina o ingenio electrónico programable, capaz de manipular objetos y realizar operaciones antes reservadas solo a las persona.
- ✓ Dispositivo multifuncional reprogramable diseñado para manipular y/o transportar material a través de movimientos programados para la realización de tareas variadas. (Robot Institute of America, 1979).
- ✓ Un sistema que existe en el mundo físico y que autónomamente censa su medio ambiente y actúa sobre él (Maja Mataric/USC)

Las definiciones anteriores son muy parecidas entre sí, pues coinciden en que los robots son aparatos autónomos que realizan diversas tareas y son reprogramables.

Podemos encontrar que existen diversos tipos de robots dependiendo del propósito de su aplicación, cuya complejidad va desde aquellos que se utilizan en tareas industriales de ensamblado (e. g. ensambladoras de autos), móviles o de locomoción (e. g. exploración), hasta llegar a los de aspecto humano ("humanoides").

2.1.2. Robots Móviles

Un robot es móvil mediante ruedas o algún tipo de extremidad que le permita desplazarse. El análisis de móviles con ruedas, es de menor complejidad en comparación con los que cuentan con extremidades. Centrando nuestra atención en los móviles con ruedas encontramos que existen diversas configuraciones que hacen posible la eficiencia y estabilidad del robot.

✓ **Configuracion Diferencial:**
Consiste en dos ruedas con sus respectivos motores, además de una o más rueda de estabilidad. Esta arquitectura se caracteriza por ser fácil de construir y fácil de controlar, pero presenta la gran dificultad de trazar líneas rectas, pues una pequeña perturbación, ya sea en las llantas o en el terreno, ocasionará que pierda la línea que estaba trazando [Vicente Lober, 2003].

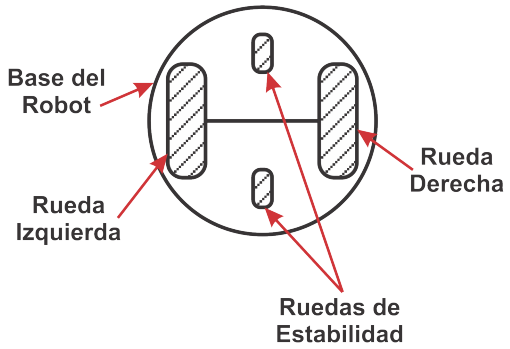


Figura 2.1: Configuración Diferencial

✓ **Configuración Triciclo:**
Consiste en tres ruedas donde una de ellas es la directriz y las dos restantes son de tracción. Figura 2.2.

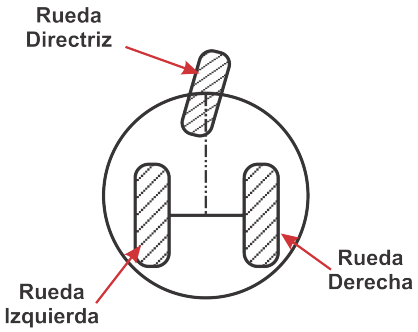


Figura 2.2: Configuración Triciclo

Cuenta con una cinemática complicada y puede presentar problemas al girar, pero no suele presenta problemas al avanzar en lineas rectas [Vicente Lober, 2003].

✓ **Configuración Ackerman:**
Es una configuración muy frecuente en robótica móvil. Se trata de un sistema de cuatro ruedas, dos de ellas como directrices y dos de tracción(Figura 2.3), es un sistema de fácil implementación, pero posee una cinemática muy complicada[Vicente Lober, 2003].

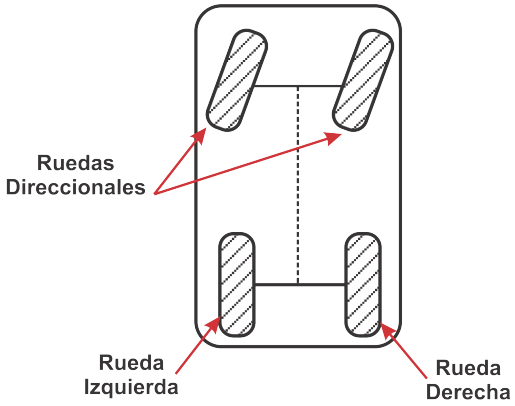


Figura 2.3: Configuración Ackerman

Estos son solo algunas de las arquitecturas existentes en robótica móvil. En nuestro tema de investigación se hará uso de un robot de arquitectura diferencial con dos ruedas de estabilidad.

2.1.3. Visión por Computadora o Visión Artificial

Según Aristóteles: *Visión es saber que hay y donde mediante la vista*, la real academia española define a la visión como: *Acción y efecto de ver*, entonces, *Ver* es: *Reconocer con cuidado y atención algo, leyéndolo o examinándolo"*. Gibson nos dice: *Visión es recuperar información de los sentidos (vista), propiedades validas del mundo exterior*[Sucar, 2012].

Existen muchas definiciones para la visión, pero una de las más acertadas y hasta cierto punto aceptada por la comunidad científica para definir la vision artificial es la de Marr[Marr, 1982].

"Visión es un proceso que produce a partir de imágenes del mundo exterior una descripción que es útil para el observador y que no tiene información irrelevante".

De la definición de Marr, podemos deducir que la visión es:

✓ Un ***proceso*** computacional.

✓ La ***descripción*** depende del observador.

✓ Contiene ***información relevante***, por lo tanto hay que eliminar información no relevante.

Algunas otras definiciones para la visión artificial son:

- ✓ La visión artificial se puede considerar como el conjunto de todas aquellas técnicas y modelos que nos permitan el procesamiento, análisis y explicación de cualquier tipo de información espacial obtenida a través de imégenes digitales[Armendariz., 2003].
- ✓ La visión artificial es un campo de la inteligencia artificial cuyo propósito es conseguir programar un ordenador para que sea capaz de analizar e interpretar una escena.
- ✓ Programar un computador para que entienda una escena o las características de una imagen[Mateo Ingelmo, 2009].
- ✓ En robótica la visión artificial es un sistema que los robots pueden utilizar para inferir el estado de su entorno. Se trata de un sistema complejo que entrega un flujo continuo de información, que debe de ser procesado para intervenir directamente en la acción o comportamiento de un robot[Alvarado Legaria, 2007].

De las definiciones anteriores podemos destacar que *la visión artificial es un sistema encargado de obtener información del entorno mediante el procesamiento de imágenes, la información que se obtiene depende del propósito para el cual el sistema de visión haya sido diseñado.*

La visión artificial está muy relacionada con el procesamiento de imágenes, las cuales son capturadas mediante una o múltiples cámaras. A través del procesamiento de imágenes se pretende mejorar la calidad de la imagen para obtener la mejor información[Sucar, 2012]. En la práctica se utilizan diferentes tipos de imágenes tales como: Imagen binaria, Imagen en tonos de gris o monocromática, Imágenes a color, Imagen multi- espectral[Armendariz., 2003].

Utilizar la visión artificial representa numerosas ventajas tales como>

- ✓ Mayor precisión en las medidas, tanto en 2D como 3D.
- ✓ Alta velocidad de respuesta.
- ✓ Verificación en lugares inaccesibles para las personas.
- ✓ Verificación objetiva y uniforme a lo largo del tiempo.
- ✓ Ahorro económico.

Gracias a las tecnologías de visión artificial, una aplicación innovadora bien diseñada e implementada puede ser capaz de:

- ✓ Optimizar al máximo el proceso de fabricación.
- ✓ Maximizar la productividad y la calidad del producto.
- ✓ Dar un valor agregado a procesos de producción.

Geometría de Proyección

Las transformaciones geométricas básicas son traslación, rotación y escala.

✓ Traslación: Cambio en la Posición.
Aplicar traslación a un objeto es cambiar su posición a lo largo de la trayectoria en una línea recta. Al agregar las distancias, de traslación, t_x y t_y a la posición de coordenadas originales (x, y) para mover el punto a una nueva posición (x', y') se obtiene:

$$\begin{aligned} x' &= x + t_x \\ y' &= y + t_y \end{aligned} \tag{2.1}$$

El par de distancias de traslación *(t_x, t_y)* es llamado vectores de traslación o vectores de cambio [Donald Hear and Baker., 1997]. Podemos expresar las ecuaciones de traslación como una sola ecuación matricial de columna para representar las posiciones de coordenadas, quedando de la siguiente manera:

$$P = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, P' = \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix}, T = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{2.2}$$

Lo que nos permite expresar las dos ecuaciones de traslación bidimensional en la forma de la siguiente matriz:

$$P' = P + T \tag{2.3}$$

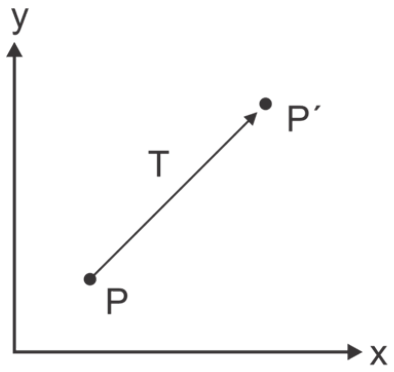


Figura 2.4: Traslación

✓ Rotación: Cambio en la Orientación.
Para generar una rotación, especificamos un ángulo de rotación θ y la posición (x_r, y_r) del punto de rotación o punto pivote, en torno al cual se gira el objeto. El punto pivote es generalmente el centro geométrico de la imagen.

Se define entonces la matriz de rotación R como [Donald Hear and Baker., 1997]

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{2.4}$$

En la Figura 2.5 se muestra la rotación de un objeto respecto a su centro geométrico.

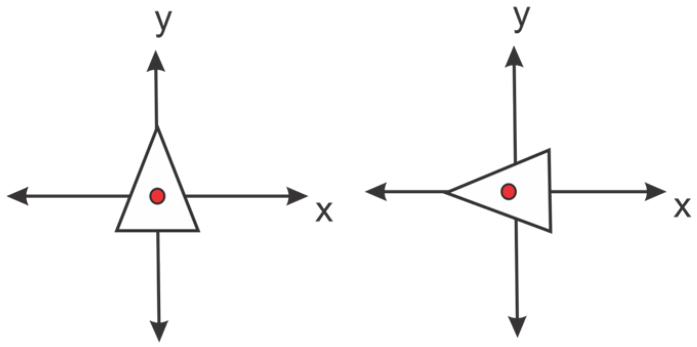


Figura 2.5: Rotación de un objeto, con punto pivote en su centro geométrico

✓ Escalado: Cambio en el Tamaño:
Una transformación de escala o escalamiento altera el tamaño de un objeto o polígono. Por lo tanto para aplicar el escalamiento es necesario multiplicar los valores de coordenadas *(x, y)* de cada vértice por los factores de escala S_x y S_y respectivamente, para producir las coordenadas transformadas (x', y'), teniendo el par de coordenadas siguiente:

$$\begin{aligned} x' &= x \cdot S_x \\ y' &= y \cdot S_y \end{aligned} \tag{2.5}$$

Dónde S_x escala los objetos en la dirección del eje x, mientras que el factor S_y lo hace en la dirección del eje y. De forma matricial el escalamiento se expresa como sigue:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

$$P' = S \cdot P \tag{2.6}$$

Donde S es la matriz de escala de 2x2. Es posible asignar valores numéricos positivos cualesquiera a los factores de escala S_x y S_y. Los valores menores a 1 reducen el tamaño de los objetos y los mayores a 1 producen una ampliación. Cuando se asigna el mismo valor a S_x y S_y se genera una escala uniforme que mantiene las proporciones relativas de los objetos (Figura 2.6). Cuando S_x y S_y tiene valores distintos se obtiene una escala diferencial (Figura 2.7).

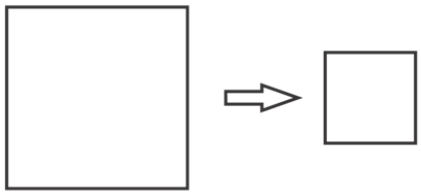


Figura 2.6: Escalado Uniforme

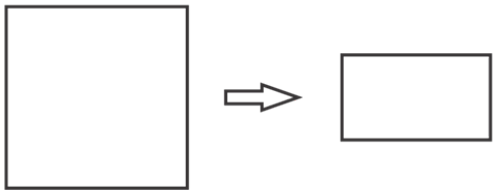


Figura 2.7: Escalado Diferencial

✓ Representación Homogénea.
Para expresar cualquier transformación bidimensional (en coordenadas cartesianas (x, y)) como una multiplicación de matices, representamos a cada transformación en coordenadas homogéneas (x_h, y_h, h), donde:

$$x = \frac{x_h}{h}$$
$$y = \frac{y_h}{h} \tag{2.7}$$

Por lo que, una representación general de coordenadas homogéneas se puede expresar como $(h \cdot x, h \cdot y, h)$ [Donald Hear and Baker., 1997]. Seleccionando el parámetro homogéneo h como cualquier valor diferente a cero. Es conveniente establecer $h = 1$, ya que otos valores son utilizados por ejemplo en la formulación de matrices de transformación de vistas tridimrensionales. Por lo tanto, las coordenadas homogéneas se representan en coordenadas $(x, y, 1)$ y las operaciones de transformación se expresan como matrices de 3x3 [Donald Hear and Baker., 1997]. Teniendo así las matrices correspondientes a cada transformación:

Traslación

$$T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

Escala

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

Rotación

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

Expresar las transformaciones en coordenadas homogéneas nos permite representar todas las ecuaciones de transformación geométrica como multiplicaciones de matrices, con lo cual obtendremos la siguiente forma general para la matriz de transformación M:

$$M = T \cdot S \cdot R$$

$$M = \begin{bmatrix} S_x \cos\theta & -S_y \sin\theta & t_x \\ S_x \sin\theta & S_y \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

Aplicando las restricciones mencionadas en la Sección 1.1, consideramos el ángulo $\theta = 0$ debido a la perspectiva ya que el móvil no puede girar alrededor del eje perpendicular al plano imagen. $S_x = S_y = A$, es decir escalado uniforme u homogéneo. De modo que la matriz de transformación, utilizada en este trabajo se considera como sigue:

$$M = \begin{bmatrix} A & 0 & t_x \\ 0 & A & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

2.1.4. Registro de Imágenes

El registro de imágenes (Registration en inglés) puede ser considerado como un procedimiento de alineamiento espacial de imágenes pudiendo ser de la misma o de diferente modalidad [Li et al., 1995].

Los métodos de registro de imágenes son necesarios cuando buscamos la fusión de imágenes que contienen información complementaria (fusión de datos), o simplemente para comparar el contenido de las imágenes (puesta en correspondencia de objetos) [Li et al., 1995]. Geométricamente esto supone alinear una de las imágenes con otra imagen. Este tema tiene multitud de aplicaciones, tales como, detección de rostros, codificación de vídeo, imágenes médicas, predicción del tiempo, codificación de bases de datos, etc. Las diferencias entre las dos imágenes pueden ser de diversa naturaleza: desplazamientos, giros, deformaciones, distinto contraste, etc.; así se puede comprobar cómo en ocasiones este tipo de cálculos llega a ser realmente complejo. [García Capel, 2007].

Principio Matemático del Registro de Imágenes

Sean dos imágenes I_r (referencia) e I_T (a transformar). Superponer las dos imágenes implica determinar los parámetros de la transformación $\widetilde{T}$ (Ecuación 2.13) mediante un método de optimización ($argopt$) que maximice la semejanza (medida de similitud S) de las estructuras homólogas extraídas de las imágenes. Las funciones f_1 y f_2 representan los algoritmos de cálculo o de segmentación de las estructuras homólogas de las imágenes I_r e I_T.

$$\widetilde{T} \quad = \quad arg \quad opt_T S(f_1(I_r), T(f_2(I_T))) \tag{2.13}$$

Las estructuras homólogas, la medida de similitud, el tipo de la transformación geométrica T y el método de optimización a emplear se definen en función de la aplicación, de la naturaleza de las imágenes, de los requerimientos en términos de precisión, de robustez, etc.

Componentes del Esquema Matemático Clásico del Registro de Imágenes

✓ Información en Común:
La información en común (estructuras homólogas) es extraída de las imágenes gracias a algoritmos de segmentación (funciones f_1 y f_2 de la Ecuación 2.13), o calculada directamente con los valores de los niveles de gris. De manera general, la información que se extrae de las imágenes es de dos tipos.

- Imágenes Primitivas:
  Las imágenes primitivas son objetos geométricos que pueden ser extraídos de las imágenes. Las primitivas son por ejemplo puntos particulares, contornos con forma de segmentos de rectas o de curvas, superficies, etc. En fin, es posible utilizar varios tipos de primitivas al mismo tiempo, por ejemplo contornos y puntos para volver más robusto al procedimiento de registro de imágenes [Li et al., 1995].

- Niveles de Grises:
  Los niveles de gris de las imágenes son con frecuencia utilizados para calcular una medida de similitud en el caso en el que ninguna primitiva significativa pueda ser extraída de las imágenes. En tal caso, la medida de similitud es calculada gracias a una relación común entre los niveles de gris de las imágenes, o a partir de criterios estadísticos.

✓ Medida de Similitud:
En el contexto del registro de imagines, la similitud es evaluada mediante la medida de una relación entre las intensidades de los píxeles de las imagines. La hipótesis que se hace es la siguiente: La relación es máxima cuando las imágenes están registradas, y conforme decrece también lo hace la calidad del registro de las imágenes. Se trata de un concepto que mide un cierto tipo de dependencia entre las distribuciones de intensidad que caracterizan las imágenes, pudiendo considerarse diferentes medidas de similitud de acuerdo a la hipótesis que se pueda establecer entre tales distribuciones.

Consideremos por ejemplo un caso ideal de registro de imágenes en el que las dos imágenes son adquiridas con la misma cámara. Supongamos que los puntos de vista de las dos adquisiciones son casi idénticos, que las condiciones de iluminación permanecen relativamente constantes y que el ruido que afecta a las imágenes es despreciable. Bajo tales condiciones, un píxel de una imagen y su correspondiente en la segunda imagen deberían tener niveles de grises idénticos o muy parecidos si los dos píxeles representan un mismo punto de la escena. A partir de esta hipótesis de correlación (relación = identidad) es fácil deducir una medida de similitud:*la suma de la diferencia entre píxeles correspondientes es nula o mínima cuando las imágenes se encuentran registradas correctamente*. En este caso concreto se trata de un criterio de disimilaridad que hay que minimizar.

✓ Tipo de Transformación:
Las transformaciones Geométricas T modifican la relación espacial entre píxeles. En términos del procesamiento de imágenes digitales una transformación geométrica consiste en una transformación espacial que define la reubicación de los píxeles en el plano imagen.

En la Tabla 2.1 se observan algunas transformaciones básicas. Las coordenadas (x, y) de un punto son transformadas en (x', y'), φ es el ángulo de rotación, t_x y t_y son los parámetros de translación, k, k_x y k_y corresponden a factores de escala isotropicos y en las direcciones x y y respectivamente. S_x y S_y son los factores responsables del efecto *"shiring"* en las direcciones x y y respectivamente. a_{ij} son parámetros de valor real, w representa la tercera coordenada del punto.

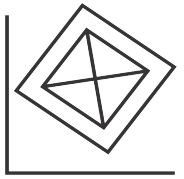
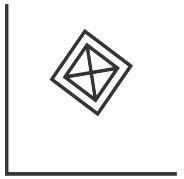
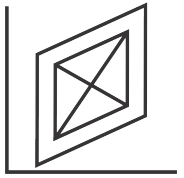
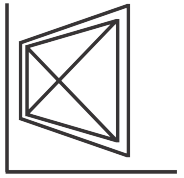
Según la aplicación, se emplean transformaciones T más o menos complicadas entre imágenes. Por ejemplo, una transformación de tipo euclidiana será utilizada cuando la puesta en correspondencia de dos imágenes implican únicamente translaciones y rotaciones. Una transformación de este tipo puede resultar insuficiente en la mayoría de las aplicaciones, en cuyo caso es necesario considerar una relación geométrica mas compleja entre las imágenes a registrar. La naturaleza de la T a utilizar está ligada en general al sistema de adquisición de las imágenes, a los tipos de escenas adquiridas y a sus posiciones relativas.

En cualquier caso, es necesario determinar el tipo de transformación que modela mejor las diferencias entre imágenes.

✓ Método de Optimización:
El objetivo de toda optimización es encontrar de manera rápida, robusta y precisa una solución. El tipo de método de optimización se selecciona de acuerdo a diferentes criterios:

- Tiempo de cálculo disponible,
- Ubicación de la posición inicial en el espacio de parámetros,
- Naturaleza de los datos (discretos o continuos),
- Forma del espacio de parámetros (extremo global, con o sin, numerosos extremos locales), etc.

Tabla 2.1: Clasificación de Modelos de Transformación Según los Grados de Libertad.

Transformación	Grados de Libertad	Modelo de Transformación	Ejemplo
Euclideana	Traslaciones + Rotaciones	$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\varphi & \pm\sin\varphi \\ \mp\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$	
Rigida	Euclideana + Factor de Escala Uniforme	$\begin{bmatrix} x' \\ y' \end{bmatrix} = k \begin{bmatrix} \cos\varphi & \pm\sin\varphi \\ \mp\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$	
Afin	Rigida + Factor de Escala No Uniforme	$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} k_x\cos\varphi & \pm S_x\sin\varphi \\ \mp S_y\sin\varphi & k_y\cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$	
Proyectiva	Afin + Proyeccion Perspectiva	$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} u/w \\ v/w \end{bmatrix} \Lambda \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	

En el caso en el que el espacio de parámetros contiene además del extremo global buscado numerosos extremos locales bien pronunciados, y que las condiciones iniciales están relativamente alejadas de la solución buscada resulta difícil, y a veces imposible, encontrar la solución con una técnica de optimización. En estos casos extremos, una búsqueda exhaustiva en el espacio de parámetros es necesaria. Para ello el algoritmo debe realizar la correspondencia entre imágenes haciendo una búsqueda de todos parámetros (escala, desplazamientos en x y y, rotación, etc.) involucrados en el registro.

2.1.5. Búsqueda Exhaustiva

Considérese un arreglo de elementos en el cual los objetos se han colocado en cierto orden y que queremos encontrar un elemento particular en él. El proceso usado para encontrar esta entrada se denomina ***búsqueda***. Dado que buscar es una actividad común en la computación, es preferible encontrar métodos eficientes para ejecutarla, como lo son los algoritmos de búsqueda por fuerza bruta, combinatoria o exhaustiva[Langsam et al., 1997].

Los algoritmos exhaustivos son aquellos que analizan todo el espacio de búsqueda para encontrar una o todas las soluciones y garantizan que pueden encontrar una solución óptima. Este tipo de estrategias tiene una premisa fundamental que afirma que si un problema tiene solución esta la encuentra.

En el contexto del registro de imágenes la búsqueda exhaustiva consiste en probar todos y cada uno de los valores que en forma combinada pueden tomar los parámetros del modelo de transformación que se este considerando, para determinar cual es la que registra mejor el par de imágenes. Por ejemplo, cuando la puesta en correspondencia de dos imágenes implica únicamente un desplazamiento en pixeles D_x en la dirección del eje x, se prueban todas las translaciones posibles a lo largo de este eje, en donde el espacio de búsqueda para el desplazamiento se encuentra acotado ($D_{min} \leq D_x \leq D_{max}$), ver Figura 2.8. Evidentemente esta búsqueda sería imposible de realizar en la practica si consideramos que nuestro espacio de búsqueda es continuo es decir, el desplazamiento (en píxeles) que separa dos imágenes puede tomar cualquier valor no entero. En la practica se discretiza el espacio de búsqueda obteniéndose ND_x valores para D_x. En particular cuando se trata de desplazamientos es común considerar incrementos para D_x de un píxel ($\triangle x = 1$), de manera que $D_x^{i+1} = D_x^i + \triangle x$, con $i = 1, 2, 3, ...ND_x$ el indice del desplazamiento y $D_x^1 = D_{min}$. $\triangle x$ pueden reducirse de acuerdo a la exactitud deseada. De esta manera se evalúan únicamente ND_x valores para D_x dentro del espacio de búsqueda.

Para cada valor de translación se evalúa una función de similaridad (medida de similitud) entre las dos imágenes, y al final se toma el desplazamiento que maximice la función. Conforme se consideran mas grados de libertad, el tiempo de calculo crece rápidamente, por ejemplo, si además de buscar desplazamientos D_x, también se buscan desplazamientos D_y a lo largo del eje y, con ND_y posibles valores para D_y, en este caso la función de similaridad es evaluada ND_xxBD_y veces es decir, para cada valor posible de D_x se evalúan todos los valores posibles para D_y.

La busqueda exhaustiva es sencilla de implementar, sin embargo su coste de ejecución es proporcional al número de soluciones posibles, el cual es exponencialmente proporcional al tamaño del problema. Este método es prohibitivo debido a que el coste inherente al cálculo de los parametros de transformación buscados en las imágenes puede ser elevado.[Micó, 1996].

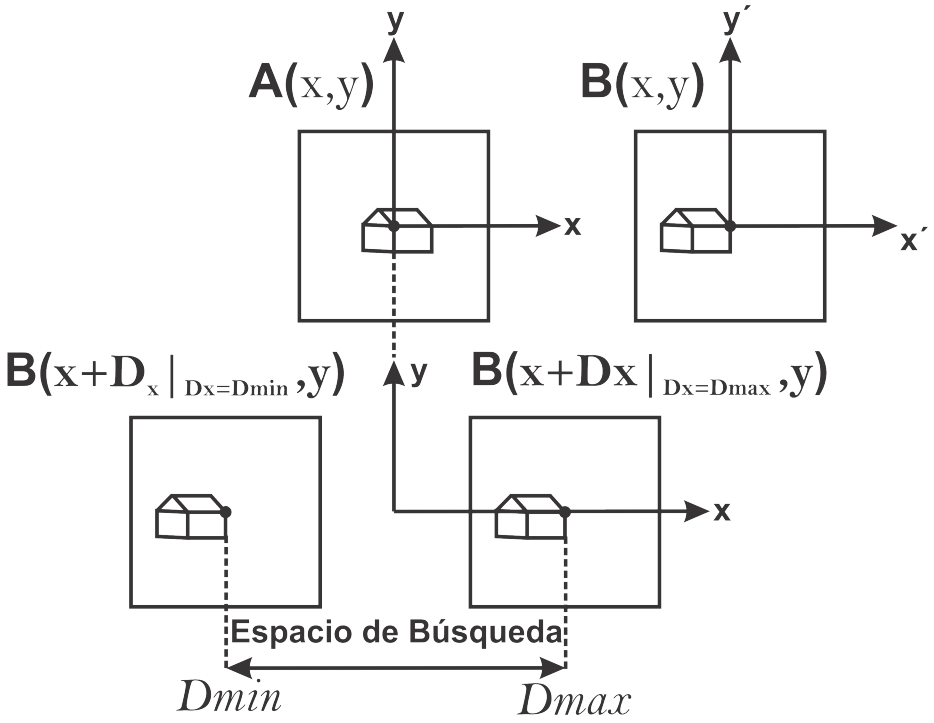


Figura 2.8: Búsqueda Exhaustiva

2.1.6. Método Piramidal

Sin duda la búsqueda exhaustiva es un algoritmo que garantiza encontrar los parámetros requeridos para el registro de imágenes, desafortunadamente también es un proceso que consume una gran cantidad de tiempo y recursos computacionales, en especial cuando se trabaja con imágenes de gran tamaño. Para solucionar tal problema se aplica una estrategia de optimización piramidal, con la cual se obtiene una rápida convergencia entre las iteraciones.

La estrategia piramidal como su nombre lo dice, consiste en formar una pirámide, en donde cada nivel de la pirámide contiene una imagen de tamaño menor a la original (Figura 2.9), así, la imagen original se ubica en la base de la pirámide y la imagen de tamaño más pequeño se ubica en la punta de la pirámide, cada nivel de la pirámide suele nombrarse sub-espacio.

Las imágenes de cada sub-espacio representan un 50 % del tamaño de la imagen del nivel anterior, es decir, las dimensiones de la imagen del sub-espacio k_i son la mitad de las dimensiones del nivel $k_{(i-1)}$ (con $i = 1, 2, 3...s$, y s el número de sub-espacios).

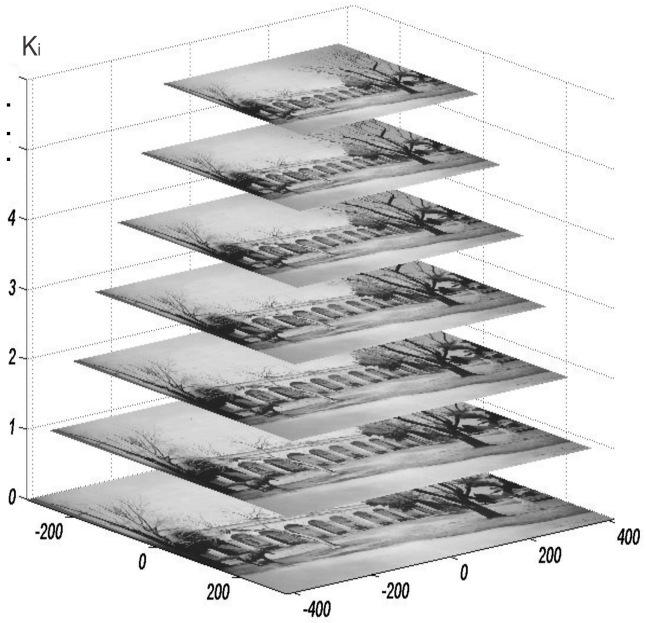


Figura 2.9: Estructura Piramidal

En el proceso de reducción de la resolución de las imágenes que se realiza mediante diezmado en el dominio espacial, intervienen filtros anti-aliasing con el propósito de retirar de manera efectiva los detalles de altas frecuencias y el ruido de las imágenes.

Aplicando una estrategia piramidal la búsqueda de los parámetros se realiza de forma más rápida, para entender el por qué, a continuación se explica la estrategia piramidal.

En el nivel superior k_s (imágenes mas pequeñas) se efectúa una primera optimización de los parámetros de transformación. A pesar de que no se conocen los valores de los parámetros de las transformaciones que registran el par de imágenes y que pudieran estar lejos de los valores con los que se inicializa su optimización, el proceso es relativamente rápido ya que se trabaja con imágenes pequeñas. Una vez determinados los parámetros de transformación que registran las imágenes se desciende un nivel en la pirámide (nivel k_{s-1}), a partir de los parámetros de transformación determinados en el nivel k_s se estiman los parámetros de transformación aproximados para este nuevo nivel con los que se comienza un nuevo proceso de optimización. En el caso de los parámetros de translación, al pasar de un nivel al siguiente sus magnitudes se duplican ya que las imágenes son del doble del tamaño del nivel anterior, así por ejemplo, si en un nivel se encuentra que una imagen esta desplazada con respecto a otra 10 *pixeles* con una incertidumbre de $\pm 0{,}5$ *pixeles*, el desplazamiento entre estas dos imágenes deberá estar entre 19 y 21 *pixeles* en el siguiente nivel.

Para el caso del factor de escala, su valor se mantiene entre los diferentes niveles, por ejemplo, si en el nivel se determina un factor de escala de 0,9 con una incertidumbre de ±0,05, en el nuevo nivel se espera que el factor de escala que relaciona ambas imágenes este entre 0,85 y 0,95, de manera que en este nuevo nivel el proceso de optimización se inicializa con el valor de 0,9. De esta manera es posible acotar el espacio de búsqueda de los parámetros en los niveles siguientes, es decir que solamente en el primer nivel de la pirámide se efectúa una búsqueda amplia de los parámetros de transformación, y en todos los niveles subsiguientes se realiza una proceso de refinamiento de los resultados obtenidos en el nivel previo.

Este proceso se repite hasta terminar con todos los niveles de la pirámide. Se espera que en la última iteración del algoritmo, con la imagen original, los parámetros buscados sean los correctos. Al continuar con la iteración a sub-espacios de mayor tamaño se conduce a una estimación más precisa [Hellier et al., 2001]. Gracias a que al iniciar el cálculo de la aproximación con parámetros cercanos a los reales, los cálculos del algoritmo son pocos y en consecuencia el tiempo de procesamiento es mínimo [Adelson et al., 1984].

La estrategia piramidal reduce el tiempo de optimización, ya que los parámetros de transformación son obtenidos con pocos cálculos en cada iteración, además los resultados son más finos de forma progresiva [Thévenaz et al., 1998].

2.2. Estado del Arte

Como se mencionaba, la necesidad de conocer la posición de un robot es de gran importancia cuando se quiere realizar tareas de servicio o realizar navegación autónoma. A continuación se revisan algunos trabajos relacionados con nuestro tema de investigación así como las diferencias con cada uno de ellos.

Krotkov en [Krotkov, 1989] equipa un móvil con una sola cámara y un mapa con marcas especiales en puntos estratégicos tales como (puertas, escritorios y uniones entre paredes). El robot se sitúa en una superficie plana y adquiere una imagen, enseguida extrae características como bordes verticales y calcula las direcciones entre puntos de referencia visibles. El problema principal que destaca es determinar la posición y la orientación del robot aplicando correspondencia entre las marcas conocidas y las capturadas por la cámara. Para la solución del problema se hace uso de un árbol de interpretación donde se implementa un algoritmo de búsqueda hasta encontrar las marcas especiales.

Este trabajo presenta la solución para determinar las mediciones sobre su ubicación, no se requiere de reconstrucción tridimensional, pero se tiene un procesamiento de imágenes imperfecto lo que podría provocar la no detección de algunos puntos o marcas especiales además se presentan diversos problemas de tiempo, y como mejora esperan encontrar la correspondencia de la información sin utilizar el árbol de interpretación propuesto, con lo que esperan reducirá los tiempos requeridos para encontrar la ubicación del robot.

La diferencia entre el trabajo de Krotkov y el nuestro es que nosotros no buscaremos formas geométricas predeterminadas, sino una textura correspondiente a una escena previamente almacenda en la LapTop, además de utilizar una cámara distinta y solo nos limitaremos al posicionamiento del robot.

Ferruz y Olleron [Ferruz and Olleron, 1995] estudian la estimación del movimiento de un vehículo con respecto a su entono empleando sensores pasivos y sin necesitar ninguna estructura del entorno. Utilizan una secuencia de imágenes tomadas con una cámara convencional de video. La estimación se basa en el seguimiento en la secuencia de imágenes de una serie de rasgos simples (ventanas) que se identifican mediante contraste de los niveles de gises.

El método proporciona la suficiente información para estimar el movimiento, se emplean estrategias heurísticas que facilitan el establecimiento de correspondencias fiables entre las ventanas de la secuencia de imágenes. Con la correspondencia y el algoritmo de ocho puntos se calculan los ángulos de orientación y la dirección del vector de desplazamiento. La estimación del movimiento empleando una secuencia de imágenes es un problema computacionalmente complejo que presenta importantes dificultades para su aplicación en tiempo real, el vehículo utilizado es un Romero 3R.

Hernández presenta [Hernández et al., 2003] un sistema de bajo costo para medir la posición y orientación de robos móviles en interiores. El sistema (ver Figura 2.10)se compone de un emisor localizado en una pared del entorno y un receptor en la parte superior del robot. El emisor es un puntero láser que actúa como un faro giratorio y el receptor es un arreglo de 32 fotocélulas que forman un cilindro. La posición del Robot y su orientación se obtiene tomando los tiempos en los que la luz del láser impacta en cada una de las fotocélulas. La precisión obtenida es de 5 cm en la posición y 1 grado en la orientación. Se obtienen medidas cada 0.2 segundos.

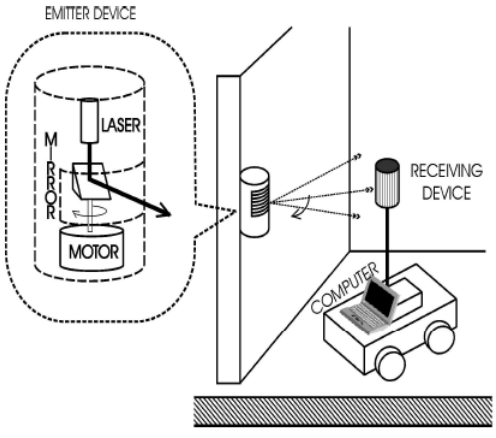


Figura 2.10: Representación esquemática del Sistema

Este sistema ofrece buena precisión, tanto en la posición y la orientación, además muy buena velocidad de censado.

En [Vázquez Jiménez, 2005] se presenta una tesis que tiene como objetivo auto localizar un robot (Figura 2.11) en un ambiente cerrado, particularmente en el Departamento de Ciencias de la Computación del instituto de investigaciones en matemáticas aplicadas y en sistemas de la UNAM. El sistema requiere de la captura de varias imágenes por medio de una cámara de video montada sobre el móvil. La estimación de la posición del robot se realiza mediante la comparación de las características de la imagen actual y las ya almacenadas con anterioridad.

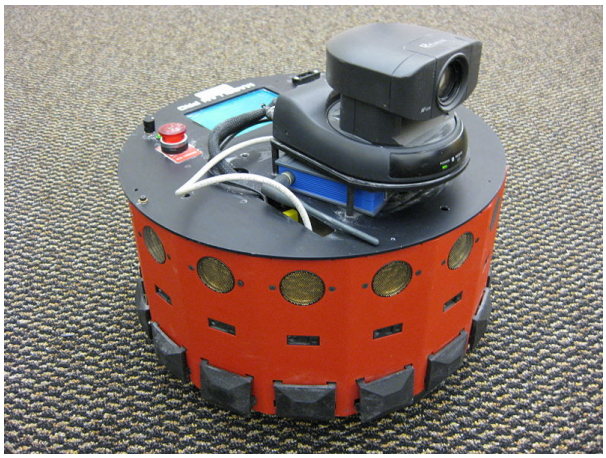


Figura 2.11: Robot Golem de la IIMAS UNAM

El sistema presentado requiere de un entrenamiento previo, es decir, de la captura de imágenes en distintas poses del ambiente además que estas imágenes deben ser capturadas con mucha precisión y cuidado, ya que de lo contrario no podrá estimar su ubicación en el ambiente. A pesar de que el sistema es implementado en un móvil bastante sofisticado, no se obtiene buenos resultados para la ubicación del mismo, el tener una base de datos de diversas imágenes para la ubicación del móvil, vuelve a este sistema sumamente complejo.

Viramontes y González presentan la ubicación del efector final (herramienta) de un brazo robótico (Figura 2.12), el sistema tiene muy buena precisión ya que logra ubicar la herramienta de trabajo con un error en milímetros, se ayuda de un puntero laser el cual es buscado mediante algoritmos de procesamiento de imágenes, además utiliza LEDś para mantener una iluminación uniforme sobre la superficie plana donde se ubicara la herramienta [Viramontes and González, 2007].

En el artículo presentado en [Bjerknes et al., 2007] se describe un sistema de bajo costo para posicionamiento en 2D y 3D de robots móviles, el sistema hace uso de sensores ultrasónicos y de señales de radiofrecuencia y logran una precisión del 1 %, utilizan la herramienta Player/Stage para combinar robots simulados y robots reales.

La ventaja de este sistema en comparación con, por ejemplo, sistemas de visión, es que el sistema presentado puede seguir fácilmente un gran número de robots, y el aumento en el número de robots no incrementa la carga computacional en el sistema. Las mediciones de seguimiento de posición muestran una variación menor de $\pm 1cm$ para los ejes (x, y) y de $\pm 4cm$ para el eje z, en 3D.

En conclusión, este nivel de rendimiento es muy meritorio dado los bajos costos de hardware empleado. La desventaja que el sistema presenta es que es sensible a los cambios de temperatura, ya que para algún cambio de temperatura será necesaria una re-calibración del sistema.

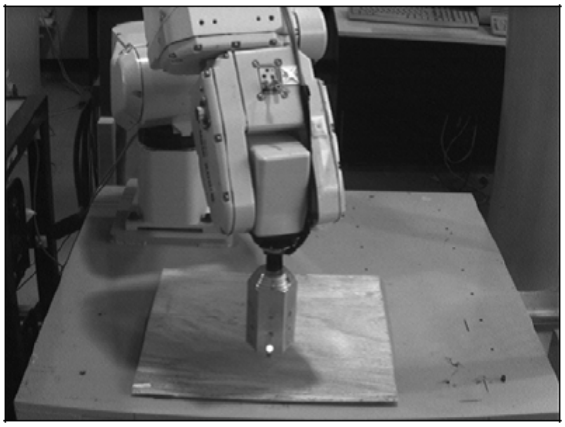


Figura 2.12: Brazo Robótico

En [Cruz Hernández, 2011] se presenta un sistema tanto para ubicación como para navegación de móviles, en este caso son robots humanoides (Figura 2.13) en un ambiente estructurado. El principal sensor que utilizan los robots para adquirir información del entorno es un sistema de visión basado en una cámara que captura imágenes del ambiente. En la investigación presentada se desarrolla un sistema de visión capaz de adquirir y procesar varias imágenes por segundo, de manera que se asegura que el sistema robótico cuente con información oportuna en un ambiente dinámico. El objetivo es encontrar el elipsoide que envuelva la distribución de las diferentes tonalidades para un color dado de un ambiente real.
Se emplea un método no probabilístico y una localización por triangulación, y se determinan las distancias hacia su objetivo a partir del análisis del número de pixeles.

Los resultados que obtiene son adecuados para su aplicación ya que el error entre la posición real y la estimada por el robot es de ±15cm además de su gran capacidad de procesar múltiples imágenes en poco tiempo.

Este sistema determina las distancias con respecto a su objetivo en base a la cantidad de pixeles en una imagen, algo muy parecido a lo que proponemos realizar.

Figura 2.13: Robot Humanoide BOGOBOT

Gifford en [Gifford, 2009] utiliza registro de imágenes para determinar con precisión giros y desplazamientos de un móvil, usan una cámara web enfocada hacia el suelo, como se muestra en la Figura 2.14. El registro lo basan en la FFT. El plano imagen de la cámara se desplaza siempre paralelo al suelo.

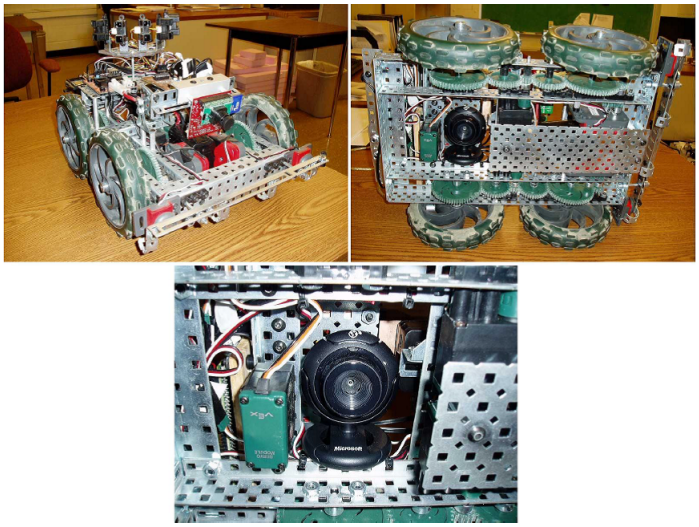


Figura 2.14: Móvil y Cámara Web

Este método es muy rápido y muy sencillo de implementar, pero solo se puede trabajar con translaciones, rotaciones y escala [Anuta, 1970][Reddy and Chatterji, 1996][Chen et al., 1994], es muy sensible a la perspectiva, incluso con perspectivas pequeñas.

La búsqueda exhaustiva es robusta por naturaleza, la estrategia piramidal permite acelerar el algoritmo, y la suma de diferencia de intensidades (SDI) como medida de similitud representa un costo computacional pequeño, es por ello que optamos por una metodología que incluye búsqueda exhaustiva, estrategia piramidal y la suma de diferencia de intensidades.

Capítulo 3

Implementación

En este capítulo se describe la implementación del sistema, el cual se encuentra conformado por 4 etapas principales. En la primer etapa se describe todo lo relacionado con la cámara web, esto incluye la descripción de sus características, la manipulación de la webcam a través de MATLAB y la calibración realizada para conocer la relación entre las distancias del mundo real y la cantidad de pixeles en la imagen capturada. La etapa dos se refiere al hardware utilizado, en una tercera etapa se describe el software de registro de imágenes implementado y por último se describe la integración de las etapas anteriores. Las etapas mencionadas se muestran en la Figura 3.1.

3.1. Cámara

3.1.1. Adquisición de Imágenes

La cámara utilizada para la captura de imágenes es una webcam Perfect Choice modelo PC-320425, (Figura 3.2). Esta webcam posee las siguientes características:

- ✓ Voltaje de alimentación: 5Vcc.
- ✓ Corriente de operación: 150mA.
- ✓ Conector USB.
- ✓ Sensor CMOS SVGA(48Kpixeles).
- ✓ Resolución óptica: 800x600 pixeles.
- ✓ Rango de muestreo: 30fps(cuadros por segundo).
- ✓ Enfoque de imagen de 3cm a infinito.

La webcam Perfect Choice PC-320425 es una cámara de bajo costo, su instalación se realiza de manera sencilla siguiendo las indicaciones del manual de usuario [PerfectChoise, 2012].

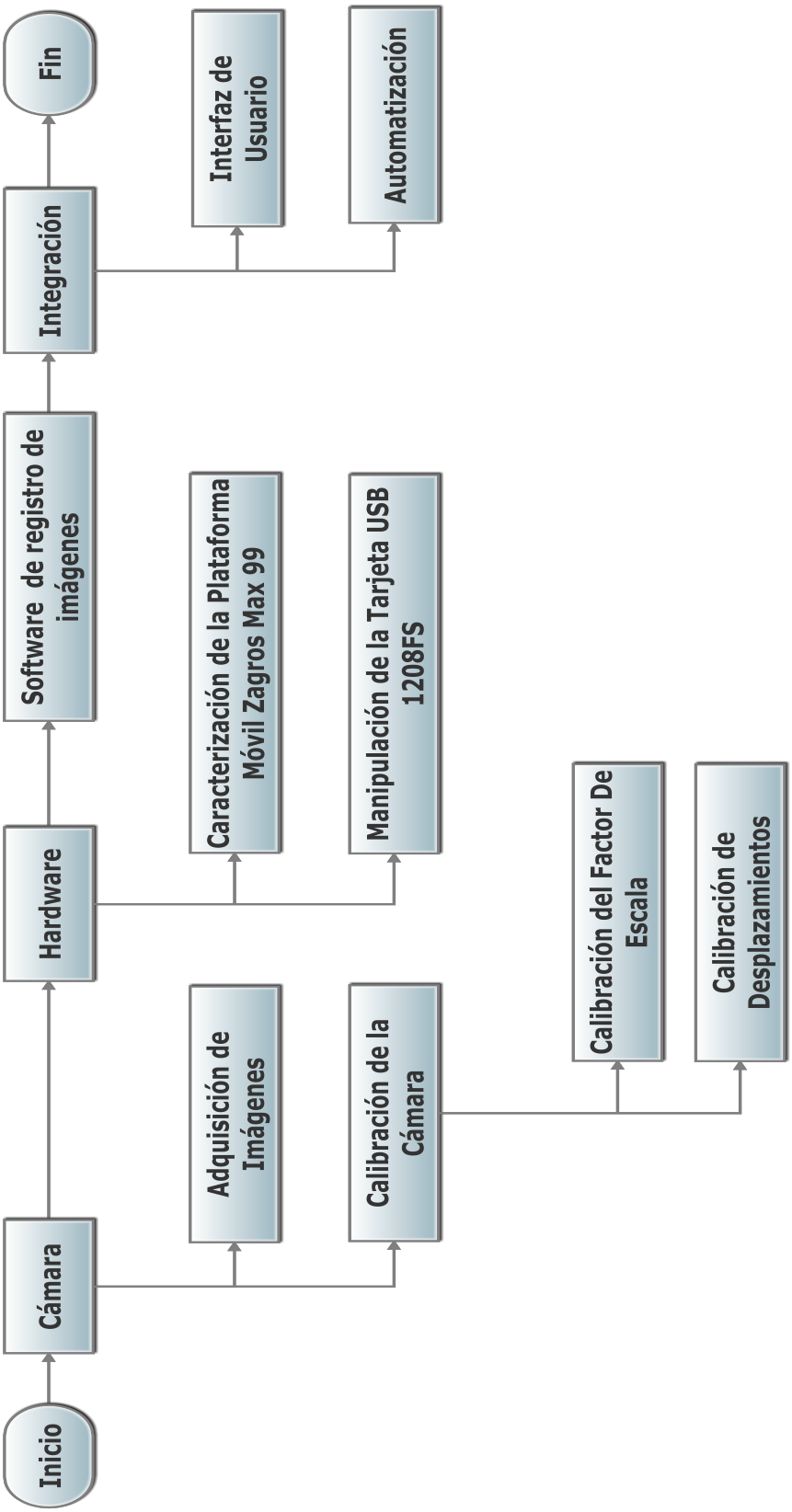


Figura 3.1: Etapas de la Implementación

Figura 3.2: Perfect Choice modelo PC-320425

Para conocer la relación entre los pixeles de la imagen capturada por esta cámara y las distancias del mundo real en la Sección 3.1.2 se describe la calibración respectiva.

Para manipular la webcam mediante MATLAB[MathWorks, 2012] se hace uso del ***Image Acquisition Toolbox***, la inicialización se realiza como se muestra a continuación:

```
1        obj = videoinput('winvideo');
2        preview(obj);
```

Donde *videoinput* crea un objeto del tipo video, el cual representa un enlace entre MATLAB y el dispositivo de video para capturar imágenes mientras que *'winvideo'* es el adaptador correspondiente para el tipo de dispositivo a utilizar.

El comando *preview* crea una ventana de vista previa del video transmitido por el dispositivo *obj*, muestra además fecha y hora, resolución de la imagen y el estado en el que se encuentra el dispositivo (Figura 3.3).

La captura de imágenes se realiza mediante el comando *getsnapshot* como se muestra a continuación:

```
1        img = getsnapshot(obj);
```

Este comando proporciona una imagen adquirida directamente del dispositivo *obj* y la almacena en *img*, a esta captura es posible aplicar procesamiento digital de imágenes, lo que implica el algoritmo de registro de imágenes (Sección 3.3).

3.1.2. Calibración de la Cámara

La calibración de la cámara es parte esencial dentro de un sistema de visión por computadora ya que nos permite conocer la relación existente entre los pixeles de la imagen capturada por la webcam y las distancias en el mundo real.

Figura 3.3: Componentes de la ventana de video proporcionada por Preview

Algunos métodos de calibración que se han desarrollado como el de Tsai descrito en el libro de [De la Escalera Hueso, 2001] permiten determinar los parámetros intrínsecos (parámetros internos de la cámara), y extrínsecos que son aquellos que relacionan los sistemas de referencia del mundo real y la cámara, siendo estos últimos los que nos proporcionan la información que necesitamos, es decir, la posición y orientación respecto del entorno, sin embargo requieren del uso de plataformas de calibración. Algunos otros no necesitan utilizar plataformas especiales para tal efecto, y solamente requieren la adquisición de al menos una imagen 2D [Miranda-Luna et al., 2008][Zhang, 1999].

Considerando el hecho de que nuestra referencia será una escena plana (bi-dimensional), podemos optar por una metodología alternativa que nos permita determinar de forma sencilla la relación entre los píxeles de la imagen y las distancias en el mundo real y por lo tanto, la ubicación del móvil sobre el cual estará sujeta la cámara. Para realizar la calibración nos apoyamos en un algoritmo de registro de imágenes basado en la información mutua y el gradiente descendente estocástico implementado en C++ [Miranda-Luna et al., 2008]. El software registra pares de imágenes, en donde una imagen es la de referencia y la otra es la capturada por la cámara desde una ubicación diferente. El software optimiza los parámetros de la matriz de transformación perspectiva (Ecuación 2.12) que relacionan o ponen en correspondencia ambas imágenes.

El método de optimización implementado (gradiente descendente estocástico) solo permite determinar los parámetros de transformación siempre y cuando las transformaciones relativas entre las dos imágenes sean relativamente pequeñas, por ejemplo, el factor de escala debe de estar entre 0.85 y 1.15 para que el algoritmo garantice la convergencia hacia el óptimo global, y en el caso de los desplazamientos, no deben exceder al 10 % del tamaño de la imagen en las direcciones de los ejes x y y. Bajo estas condiciones, el algoritmo es capaz de estimar los parámetros de transformación incluso con un error subpixel.

Los desplazamientos que consideramos en cada una de las etapas de calibración representan situaciones reales para el móvil y se traducen como transformaciones geométricas grandes. Por lo tanto para poder hacer uso del software de registro de imágenes de apoyo [Miranda-Luna et al., 2008], es necesario inicializar los parámetros de la matriz de transformación con valores cercanos a los reales de manera que el software pueda optimizarlos sin que quede atrapado en un optimo local.

A continuación se describe el procedimiento que se siguió para determinar la relación que existe entre los parámetros de translación y escala (de la matriz de transformación) con los desplazamientos del móvil (en metros). Se utilizó como objetivo la escena plana con textura no homogénea (póster) mostrado en Figura 3.4 a una distancia de referencia de *100cm* entre cámara y objetivo.

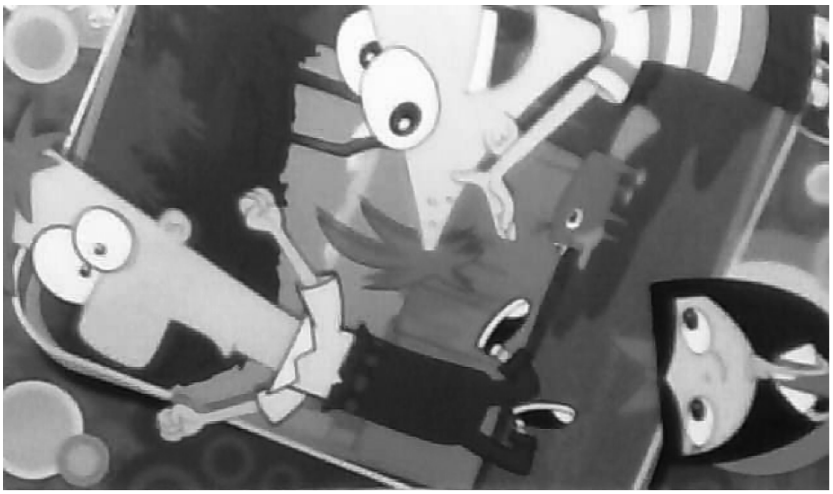


Figura 3.4: Escena plana de referencia

Cabe señalar que se buscó (en la medida de lo posible y tomando en cuenta nuestra restricción en cuanto a equipo) trabajar bajo condiciones de iluminación y posicionamiento las más cercanas a las ideales, lo cual es conveniente aunque no necesariamente indispensable [Miranda-Luna et al., 2008] para garantizar los mejores resultados en la calibración.

Alineación del Eje óptico de la cámara

Para alinear el eje óptico de la cámara es necesario conocer el centro de la imagen que no necesariamente coincide con su centro geométrico. Para ello nos apoyamos en uno de los método propuesto por Willson y descrito en el libro[De la Escalera Hueso, 2001]. En nuestro caso particular se capturaron tres imágenes de una misma escena a diferentes distancias (*50cm*, *100cm* y *150cm*) con respecto a ésta. Nuestra escena capturada contiene 6 puntos de referencia y de acuerdo a Willson al observar sus trayectorias en las imágenes el punto intersección corresponde al centro de la imagen. En nuestro caso el centro de la imagen se localiza en (350, 320) pixeles.

Una vez determinado el centro de la imagen comenzamos con la alineación del eje óptico de la cámara, para ello el objetivo es situado en una pared, se traza una línea perpendicular al plano del objetivo (sobre el suelo). Enseguida se hace coincidir el eje óptico de la cámara con la línea perpendicular al plano del objetivo y que pasa por su centro geométrico. Para esto se hizo coincidir la proyección (sobre el plano imagen) del centro geométrico del objetivo con el centro de la imagen capturada por la webcam realizando alejamientos y acercamientos de la cámara, ajustando en cada ocasión su ángulo de inclinación con respecto al plano del objetivo hasta que, pese a los desplazamientos la ubicación de la proyección del centro del objetivo en la imagen adquirida no se modifique.

Calibración del Factor De Escala

Una vez alineado el eje óptico de la cámara y ubicado el centro de la imagen, se procede a determinar la relación entre el factor de escala y la distancia entre el objetivo y la cámara, para ello se realiza la adquisición de una serie de imágenes a diferentes distancias con respecto a la posición de referencia. Primero se captura una imagen a una distancia de *100cm* entre cámara y objetivo, esta imagen servirá como referencia.

Después de la captura de la imagen de referencia, la cámara se desplaza desde una distancia de *50cm* hasta *150cm* del objetivo a pasos de *10cm*, capturando una imagen en cada una de estas posiciones (ver Figura 3.5).

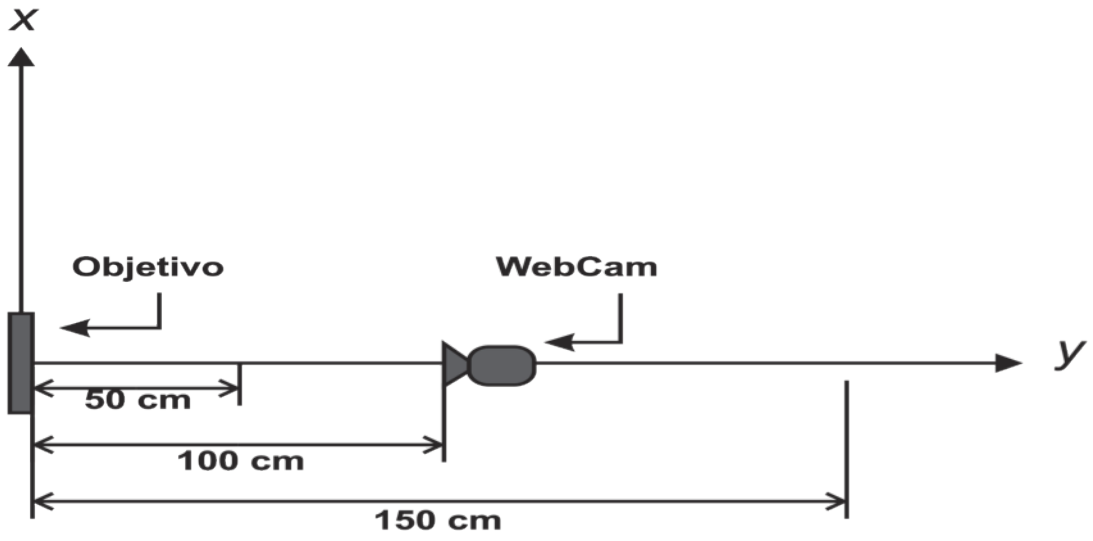


Figura 3.5: Diagrama de Movimientos para el Factor de Escala, vista superior.

Posterior a la captura de las imágenes, se procede a encontrar el factor de escala. Para determinar este parámetro se hace uso del software de registro de imágenes implementado en C++ [Miranda-Luna et al., 2008] descrito al inicio de este capítulo.

Los resultados obtenidos son mostrados en la Tabla 3.1, y con ayuda de MATLAB y el comando ***polyfit*** se encuentra la expresión matemática (Ecuación 3.1) que relaciona la distancia, entre el objetivo y la cámara, con el parámetro de transformación de escala.

$$Y = 106{,}4639A - 5{,}7088 \tag{3.1}$$

Donde $\boldsymbol{Y}$ representa la estimación de la distancia perpendicular al plano del objetivo (en *cm.*) a la cual se encuentra la cámara, y $\boldsymbol{A}$ corresponde al factor de escala.

Tabla 3.1: Datos experimentales obtenidos para el Factor de escala

Distancia real [cm.]	Factor De Escala A
150	1.45
140	1.37
130	1.28
120	1.18
110	1.09
90	0.9
80	0.8
70	0.71
60	0.61
50	0.52

La gráfica de la Figura 3.6 muestra los datos experimentales (puntos) de la Tabla 3.1 y la aproximación (línea continua) obtenida (Ecuación 3.1).

Calibración de Desplazamientos

Conservando la distancia de referencia (*100cm*) respecto al objetivo y ubicando la cámara sin desplazamientos a izquierda o derecha, es capturada una imagen de prueba, posteriormente se capturan varias imágenes desde diferentes pocisiones a lo largo de la línea paralela al plano del objetivo. Para la adquisición de las imagenes se tuvo cuidado de mantener el plano imagen paralelo al plano del objetivo. Los desplazamientos maximos contemplados permiten conservando un mínimo del 50 % del contenido común entre la imagen de prueba y las adquiridas por la cámara al desplazarse. Este mismo procedimiento se repite para las distancias de *50cm* y *150cm*. Se tiene entonces, un rango de desplazamientos de $\pm 15cm$, $\pm 30cm$ y $\pm 45cm$ para *50cm*, *100cm* y *150cm* respectivamente. El procedimiento antes descritos se puede visualizar en la Figura 3.7.

Se consideró un mínimo de área en común entre las 2 imágenes de 50 %, con el propósito de cubrir las transformaciones y por lo tanto los desplazamientos más amplios que pueden ser estimados sin error por algoritmos de registro de imágenes como el reportado en [Zokai and Wolberg, 2005].

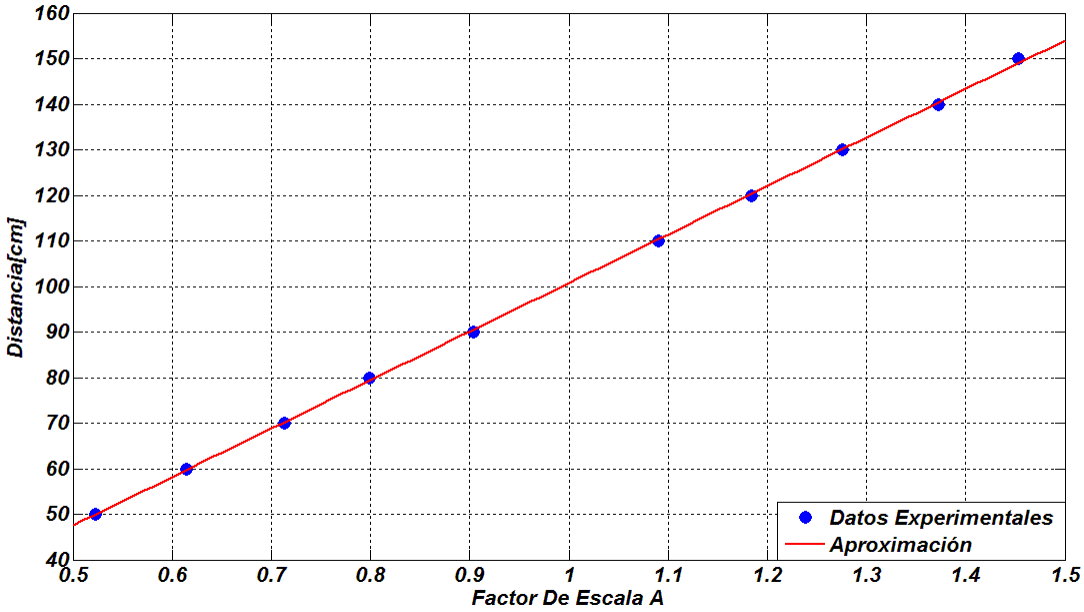


Figura 3.6: Gráfica del Factor De Escala

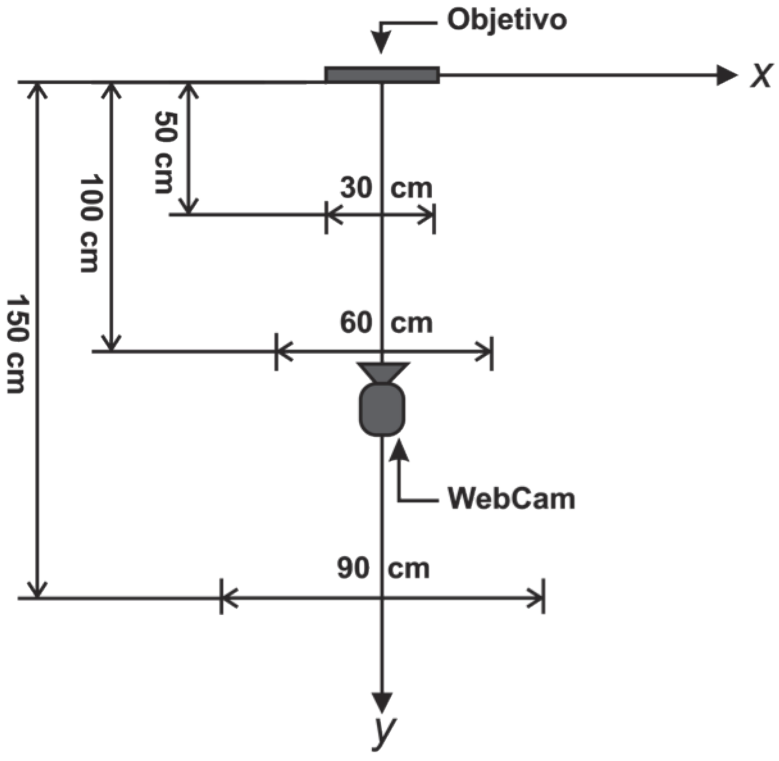


Figura 3.7: Diagrama de Movimientos para Desplazamientos

Después del procedimiento mencionado anteriormente y con ayuda del mismo software de registro de imágenes utilizado para la calibración del factor de escala, los datos experimentales son optimizados (Tabla 3.2).

Tabla 3.2: Datos Experimentales de desplazamientos

Distancia [cm]	Desplazamiento[cm]	Desplazamiento [Pixeles]
50	-12	-110.13
	-9	-83.65
	-6	-54.14
	-3	-24.88
	3	31.28
	6	61.41
	9	91.55
	12	120.22
100	-24	-239.25
	-18	-166.5
	-12	-112.69
	-6	-68.09
	6	55.4
	12	118.46
	18	175.67
	24	237.6
150	-45	-420.75
	-36	-342
	-27	-256.77
	-18	-169.8
	-9	-86.33
	9	87.02
	18	170.1
	27	259.02
	36	348.83
	45	431.27

Nuevamente con ayuda de MATLAB se obtienen las ecuaciones matemáticas para las diferentes distancias.

$$P_{x1} = 9{,}595 D_{50cm} + 4{,}1793 \tag{3.2}$$

$$P_{x2} = 9{,}8123 D_{100cm} + 0{,}3954 \tag{3.3}$$

$$P_{x3} = 9{,}5191 D_{150cm} + 1{,}9071 \tag{3.4}$$

Las ecuaciones anteriores cuentan con pendientes similares, por lo que, si se realiza un promedio, se obtiene la Ecuación 3.5 .

$$\begin{aligned} P_x &= 9{,}6421D + 2{,}1606 \\ \therefore D &= 0{,}1P_x - 0{,}22 \end{aligned} \tag{3.5}$$

Donde $D(cm)$ corresponde a la ubicación paralela respecto al plano del objetivo, en función de la cantidad P_x de pixeles desplazados en la imagen capturada por la webcam.

En la Figura 3.8 se grafican, en líneas punteadas, verde, azul y negra, las aproximaciones anteriores (Ecuación 3.2, 3.3 y 3.4 respectivamente) y, en línea continua color rojo la gráfica correspondiente a la Ecuación 3.5.

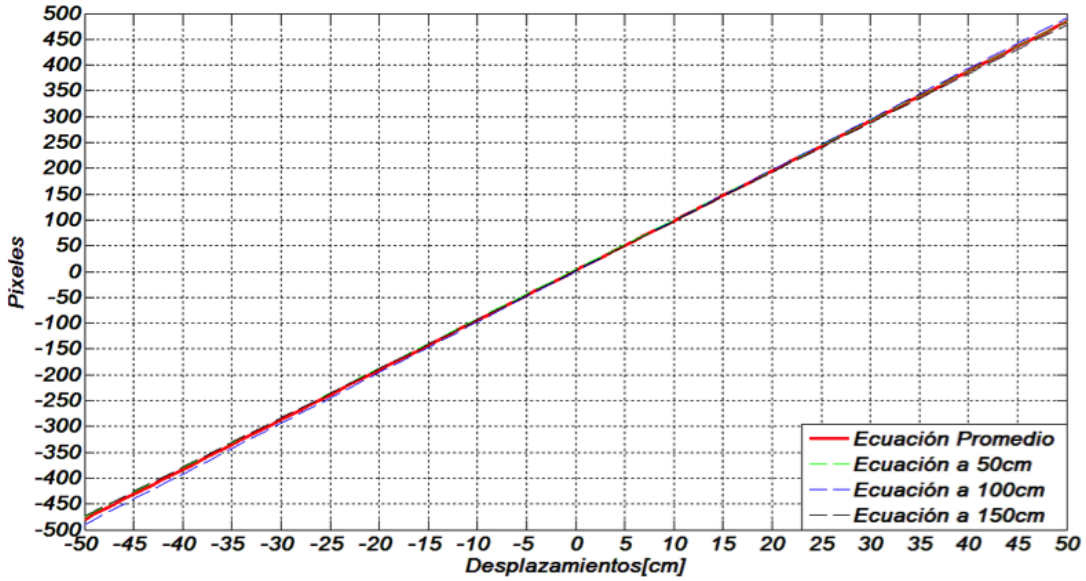


Figura 3.8: Gráfica Promedio

El método de calibración desarrollado relaciona los parámetros de transformación y las coordenadas de la cámara con respecto a una referencia, este método ha sido respaldado en la publicación del artículo[Rodríguez Santiago et al., 2012].

3.2. Hardware

La etapa correspondiente al hardware, se refiere al conocimiento del funcionamiento de los diversos dispositivos que incluirá el sistema completo, los cuales se describen a continuación.

3.2.1. Caracterización de la Plataforma Móvil Zagros Max 99

La Universidad Tecnológica de la Mixteca cuenta con la plataforma Zagros Max 99, Figura 3.9, el móvil está diseñado para ser una solución práctica y económica en diferentes aplicaciones. Plataformas similares son utilizadas en diversos trabajos de investigación ya que representan una buena opción para el estudio de estrategias de navegación y posicionamiento. La plataforma cuenta con las siguientes características [Robotics, 2002].

- ✓ Cuenta con 2 motores de 12 volts en la base con un torque máximo de 20 in-lb (con carga máxima, consumiendo 1.3 amperes por motor).
- ✓ Velocidad máxima de 12 metros por minuto.
- ✓ 2 ruedas de tracción de 15 cm de diámetro.
- ✓ 2 ruedas de estabilidad es de 7.5 cm de diámetro.
- ✓ Los motores son controlados con un chip T1 SN754410.
- ✓ La carga máxima recomendada es de 13.6 Kg.

Figura 3.9: Plataforma Zagros Max 99

El funcionamiento de esta plataforma no es complicado, ya que al ser de arquitectura diferencial, tenemos el control de cada uno de los motores de tracción. Los motores, son controlados por el chip T1 SN754410. Todo el módulo de potencia de la plataforma se muestran en Figura 3.10.

Caracterización de Desplazamientos

La plataforma carece de sensores que le permitan conocer su ubicación en el espacio, de manera que los desplazamientos a partir de una posición de referencia se determinan mediante los tiempos de activación de sus motores. El driver para el control de los motores es interconectado como se muestra en la Figura 3.11.

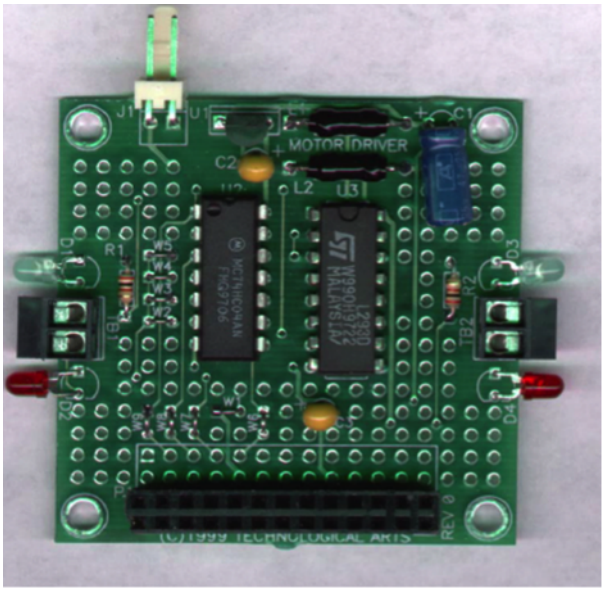


Figura 3.10: Modulo de Potencia y Chip T1 SN754410

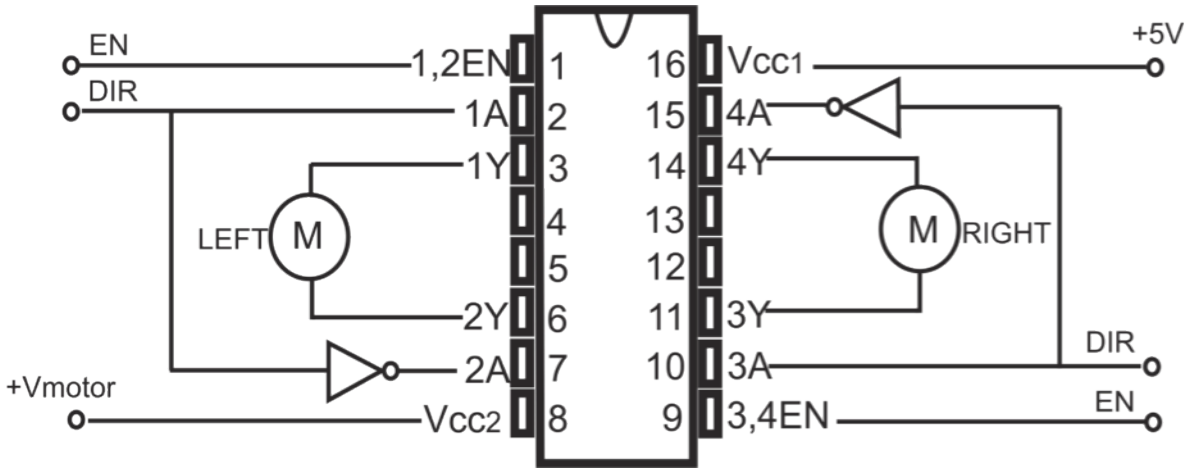


Figura 3.11: Conexión del SN754410

Con dicha configuración cada uno de los motores es activado y configurado de acuerdo a las equivalencias de la Tabla 3.3, en la cual se muestra la función del motor de acuerdo a la configuración de sus terminales, hay que resaltar que las equivalencias son aplicables para ambos motores. En la Tabla 3.3, H = Alto (High), L = Bajo (Low) y X = Condición no importa (Don't Care). Con dichas combinaciones es posible controlar los movimientos de la plataforma móvil.

Tabla 3.3: Activación de los motores

EN	1A	2A	Función
H	L	H	Turn Right
H	H	L	Turn Left
L	X	X	Motor Stop

La plataforma alcanza una velocidad ideal de *12 m/min*, es decir que *100cm* son recorrido en 5 segundos, por lo tanto, si se desea saber el tiempo de activación t(en segundos) para recorrer una distancia cualquiera d (en centímetros), se tiene la siguiente relación:

$$t = d * 0{,}05 \tag{3.6}$$

Donde el factor *0.05* será calibrado, es decir ajustado, ya que se considera una velocidad ideal, además es necesario contemplar el peso total del móvil (plataforma más accesorios que incluyen fuente de alimentación, DAQ, LapTop y cargador). Después de realizar una serie de pruebas, ajustando en cada una de ellas el tiempo para recorrer *100cm*, se obtiene la Ecuación 3.7, la cual determina el tiempo t de activación de los motores para recorrer cualquier distancia d (en centímetros).

$$t = d * 0{,}045 \tag{3.7}$$

Caracterización de Giros

Después de determinar la relación *tiempo-distancia* para recorrer distancias rectas, a continuación se determinan los tiempos para que el móvil gire $\pm 90°$ respecto a una posición, para lo cual, el móvil es ubicado sobre una superficie cubierta por mosaicos los cuales son de forma cuadrada. Esta caracterización se hizo contemplando todo el peso del móvil.

Tomando como referencia las equinas de uno de los mosaicos, se asigna un tiempo a uno de los motores y se deja fijo el otro, después se regresa a la posición de referencia y el tiempo es ajustado, continuando así hasta percibir que el móvil ha girado 90°, el mismo procedimiento es aplicado al otro motor. Después de diversas pruebas se encuentran los tiempos para que el móvil gire en sentido horario y anti horario (Tabla 3.4).

Tabla 3.4: Tiempos para girar 90°

Tiempo **t**(s)	Sentido de Giro
0.89	Horario
0.9	Anti-Horario

3.2.2. Manipulación de la Tarjeta USB 1208FS

La USB-1208FS (Figura 3.12) es una tarjeta de adquisición de datos compatible con el sistema operativo Microsoft Windows, es capaz de trabajar con LabVIEW y MATLAB. Esta tarjeta cuenta con ocho entradas y dos salidas de 12-bits analógicas, 16 terminales digitales I/O y un contador de eventos externo de 32-bits. El dispositivo es alimentado por +5 volts adquiridos del mismo puerto USB por lo que no requiere de alimentación externa. Las entradas analógicas pueden ser configuradas mediante software ya sea de modo simple teniendo ocho entradas analógicas de 11-bits o en modo diferencial con cuatro entradas analógicas de 12-bits. Dieciséis líneas digitales I/O que pueden ser configuradas como entradas o salidas independientes en dos puertos de 8-bits [Computing, 2010][Cuadrado Vaca, 2005].

Figura 3.12: Tarjeta USB-1208FS

La instalación de la tarjeta resulta bastante sencilla, ya que solamente es necesario conectarla al puerto USB e instalar los drivers requeridos y el software ***InstaCal***, que el fabricante proporciona en un CD, con el cual se configura la tarjeta de modo simple o diferencial.

Para trabajar en MATLAB con esta tarjeta, se requiere del ***Data Acquisition Toolbox (DAQ)***, puede ser mediante diagramas a bloques con MATLAB *Simulink* o directamente con código **.m* utilizando los comandos *analoginput*, *analogoutput* y *digitalio*. Se utiliza el comando *digitalio* ya que el módulo de potencia de la plataforma solo requiere de pulsos de activación. A continuación se presenta el código utilizado para la inicialización y manipulación de la tarjeta a través de MATLAB.

```
1        dio = digitalio('mcc',0);
2        lines = addline(dio,0:3,'out');
3        putvalue(dio,[0 0 0 0]);
```

Donde *'mcc'* corresponde al nombre del fabricante del dispositivo, en este caso *Measurement Computing*, *0* indica el número ID (identificador) del dispositivo, el cual puede variar dependiendo del número de dispositivos conectados al ordenador.

El comando *addlines* configura los puertos del dispositivo *dio*, de manera que, en este caso, se utiliza desde el puerto 0 al 3 (0:3), en forma de salida (*out*). Por último el comando *putvalue* envía las ordenes al dispositivo, en este ejemplo envía a *dio* la configuración de los cuatro puertos en bajo ([0 0 0 0]).

Conforme a la configuración del módulo de potencia de la plataforma, se presentan las siguientes configuraciones para las diferentes rutinas de desplazamientos del móvil (Tabla 3.5).

En el vector de configuración ***A3*** y***A2*** indican con *1* la *activación* y con *0* la *desactivación* de los motores A y B respectivamente, mientras que *A1* y *A0* indican el sentido de giro. Con las configuraciones de las de la Tabla 3.5 es posible controlar la plataforma y realizar la navegación de acuerdo con la información proporcionada por el algoritmo de registro de imágenes.

La conexión entre la tarjeta de adquisición y el módulo de potencia de la plataforma, no requiere de alguna etapa de acondicionamiento ya que solo basta con conectar las terminales correspondientes del módulo de potencia directamente a la tarjeta y conectar la terminal GND de la fuente de alimentación de la plataforma con la terminal correspondiente de la tarjeta.

Tabla 3.5: Rutinas de Desplazamiento

Rutina	A3	A2	A1	A0	Configuración DAQ
Avanza	1	1	0	0	[1 1 0 0]
Retrocede	1	1	1	1	[1 1 1 1]
Gira Izquierda(Motor B)	0	1	0	1	[0 1 0 1]
Gira Derecha(Motor A)	1	0	1	0	[1 0 1 0]

3.3. Software de registro de imágenes

El software de apoyo utilizado en la sección 3.1.2 no resulta de utilidad práctica en nuestra aplicación de robótica móvil, en donde esperamos factores de escala y translaciones relativamente grandes para los cuales el algoritmo no garantiza resultados correctos.

Por tanto se elige utilizar un software de registro de imágenes basado en el algoritmo de búsqueda exhaustiva y el método de optimización piramidal (Sección 2.1.6), con la suma de la diferencia de intensidades como medida de similitud (Sección 2.1.4).

El cálculo de la medida de similitud de suma de diferencia de intensidades se efectúa rápidamente ya que implica solamente $M \times N$ restas y una suma, (con M y N el tamaño en pixeles del área superpuesta de las imágenes en las direcciones x y y respectivamente), lo que resulta conveniente para nuestra aplicación. La búsqueda exhaustiva garantiza que el algoritmo encontrará de forma autónoma los parámetros de transformación que minimizan nuestra función (Ecuación 2.13) sin embargo, esto implica calcular un gran número de veces la medida de similitud. La cantidad de veces que se tiene que calcular depende de la amplitud del espacio de parámetros (que en nuestra aplicación es muy amplio), de la cantidad de parámetros a optimizar (en este caso son dos: translación en eje x y *escala*) y de la resolución requerida para cada parámetro, de manera que una búsqueda exhaustiva suele resultar computacionalmente ineficiente.

La solución a este problema es la estrategia piramidal, ya que permite explorar un amplio espacio de parámetros al tiempo que reduce el número de operaciones que se tienen que realizar (ver sección 2.1.6).

Además de garantizar velocidad de cálculo, un algoritmo basado en ésta metodología puede trabajar de forma automática sin necesidad de inicializar los parámetros manualmente, algo indispensable en nuestra aplicación.

El software utilizado para este propósito fue implementado en MATLAB, y se basa en el algoritmo clásico de búsqueda exhaustiva-estrategia piramidal, pero realiza una transformación geométrica (de coordenadas rectangulares a coordenadas polares-logarítmicas) tal como se hace en [Zokai and Wolberg, 2005], esto permite representar como translaciones la rotación y la escala. De este modo el algoritmo mediante la búsqueda únicamente de translaciones puede estimar la translación la rotación y la escala.

Este software recibe como entrada dos imágenes en niveles de gris, una es de referencia, mientras que la segunda es la que será transformada para hacerla corresponder con la primera y así determinar (en presencia de perspectiva moderada) la traslación a lo largo del eje x y la *escala*, ya que consideramos despreciables tanto rotaciones como translación a lo largo del eje y.

El código se incluye en el apéndice A, un diagrama de flujo de este algoritmo se muestra en la Figura 3.13, y sus alcances prácticos se verifican en la sección 4.1.

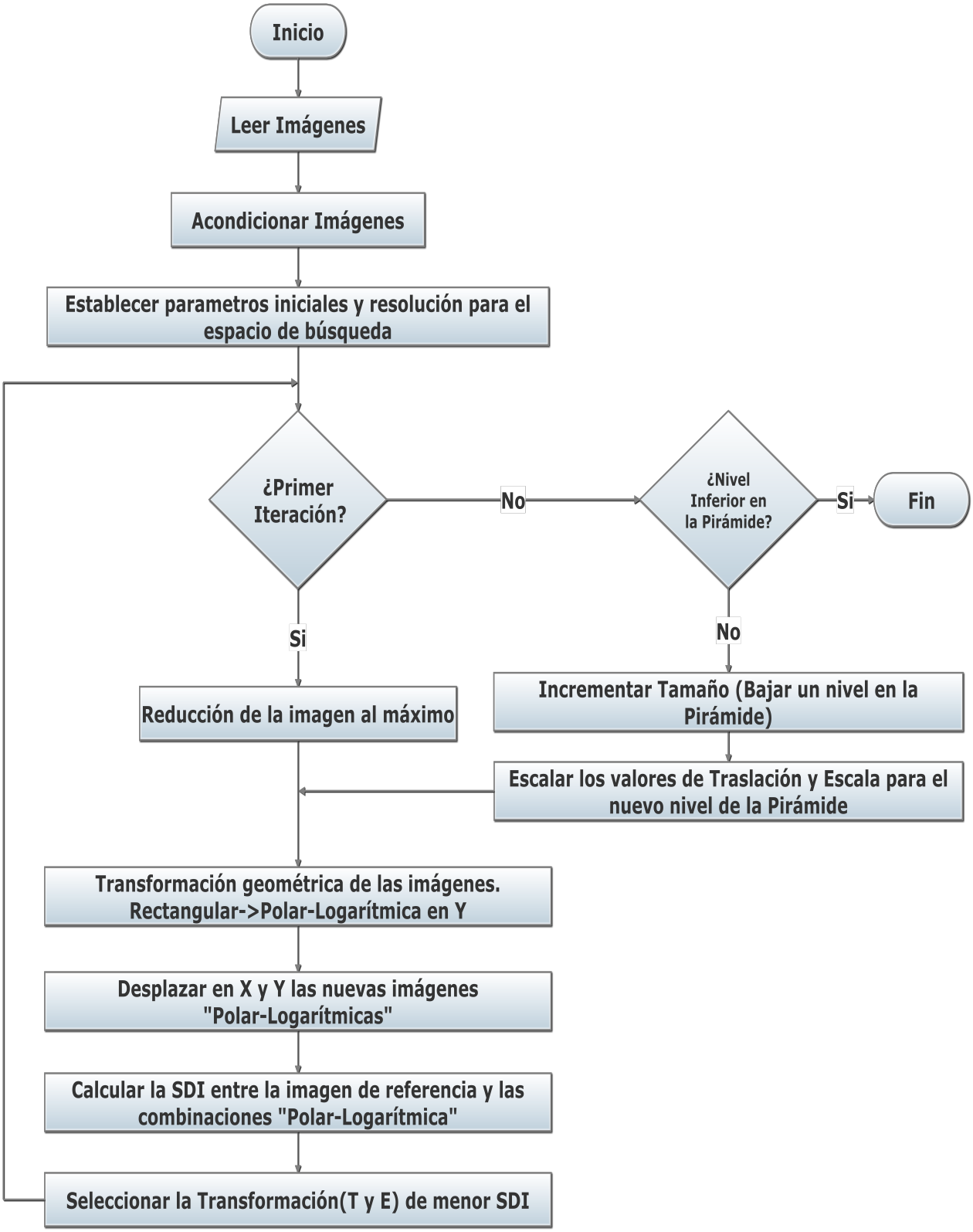


Figura 3.13: Diagrama de Flujo del Software de Registro de Imágenes

3.4. Integración

Después de conocer las partes del sistema de visión, éstas se unen para formar uno solo. El sistema final se compone de la plataforma móvil Zagros Max 99 dotada de una fuente de poder HP E3611A, que será utilizada para la alimentación, tanto de el modulo de potencia como de los motores del móvil, la tarjeta USB1208FS, la webcam y una LapTop (Ver Figura 3.14).

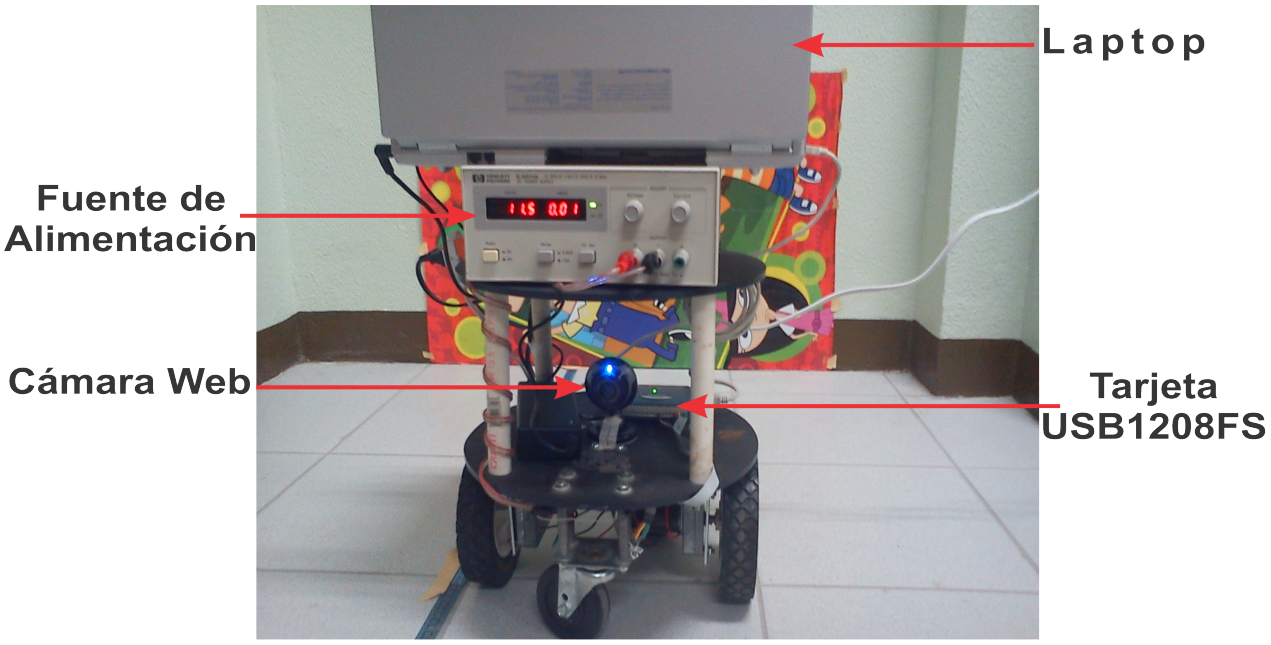


Figura 3.14: Sistema completo

El funcionamiento cosiste principalmente en la captura de una imagen mediante la cámara, misma que se envia a la LapTop, para que el software de registro de imágenes realice los cálculos necesarios y, posteriormente la plataforma móvil a través de la tarjeta de adquisición de datos (DAQ), realice los movimientos correspondientes para aproximarse a la referencia deseada.

3.4.1. Interfaz de Usuario

La interfaz de usuario es realizada mediante la herramienta de diseño **GUIDE** que **MATLAB** proporciona. En la interfaz de usuario (Figura 3.15) se presenta la **secuencia de video en tiempo real** que la webcam está capturando, **Snapshot** indica la imagen capturada en tiempo de ejecución, **Referencia** muestra la imagen que el sistema utiliza como referencia . En el cuadro de **Resultados** se muestran los parámetros obtenidos a partir de estas imágenes (factor de escala y desplazamientos) y las distancias que el móvil tiene que recorrer para llegar a la referencia deseada.

Al presionar el botón **Start** el sistema comenzará a trabajar realizando 3 iteraciones y cambiará a **Processing** para indicar que el sistema está en operación y, en el cuadro de **Movimientos** se indican los movimientos que el sistema está ejecutando iluminándose de acuerdo a la secuencia que el móvil está ejecutando.

Dentro del código de la interfaz (Apéndice A) se ha incluido el código correspondiente para operar de manera autónoma a la webcam, la tarjeta de adquisición de datos y el software de registro de imágenes. En la Figura 3.15 se presenta la interfaz de usuario en funcionamiento y en Figura 3.16 se muestra el sistema trabajando en conjunto con la interfaz.

Figura 3.15: Funcionamiento de la Interfaz de Usuario

Figura 3.16: Sistema de Visión e Interfaz de Usuario

3.4.2. Automatización

La automatización del sistema consiste en la toma de decisiones por parte del sistema de forma autónoma, es decir, sin la intervención del usuario.

El sistema trabaja de acuerdo al algoritmo mostrado en la Figura 3.17 para aproximarse a la referencia deseada.

Dónde:

- ✓ D_Ref: Distancia de referencia del sistema, en este caso *100cm*.
- ✓ D_Actual: Distancia a la cual el sistema se encuentra respecto a la referencia.
- ✓ D_x: Desplazamiento del móvil, es decir la distancia de traslación a izqueirda o derecha.
- ✓ N_it: Numero de iteración.

Al decir, Avanza, Retrocede, Gira Izquierda o Gira Derecha, se refiere a las configuraciones (Ver Tabla 3.5) que el software tiene que enviar a la tarjeta de adquisición de datos para que el móvil realice los movimientos correspondientes, y en consecuencia aproximarse a la referencia deseada.

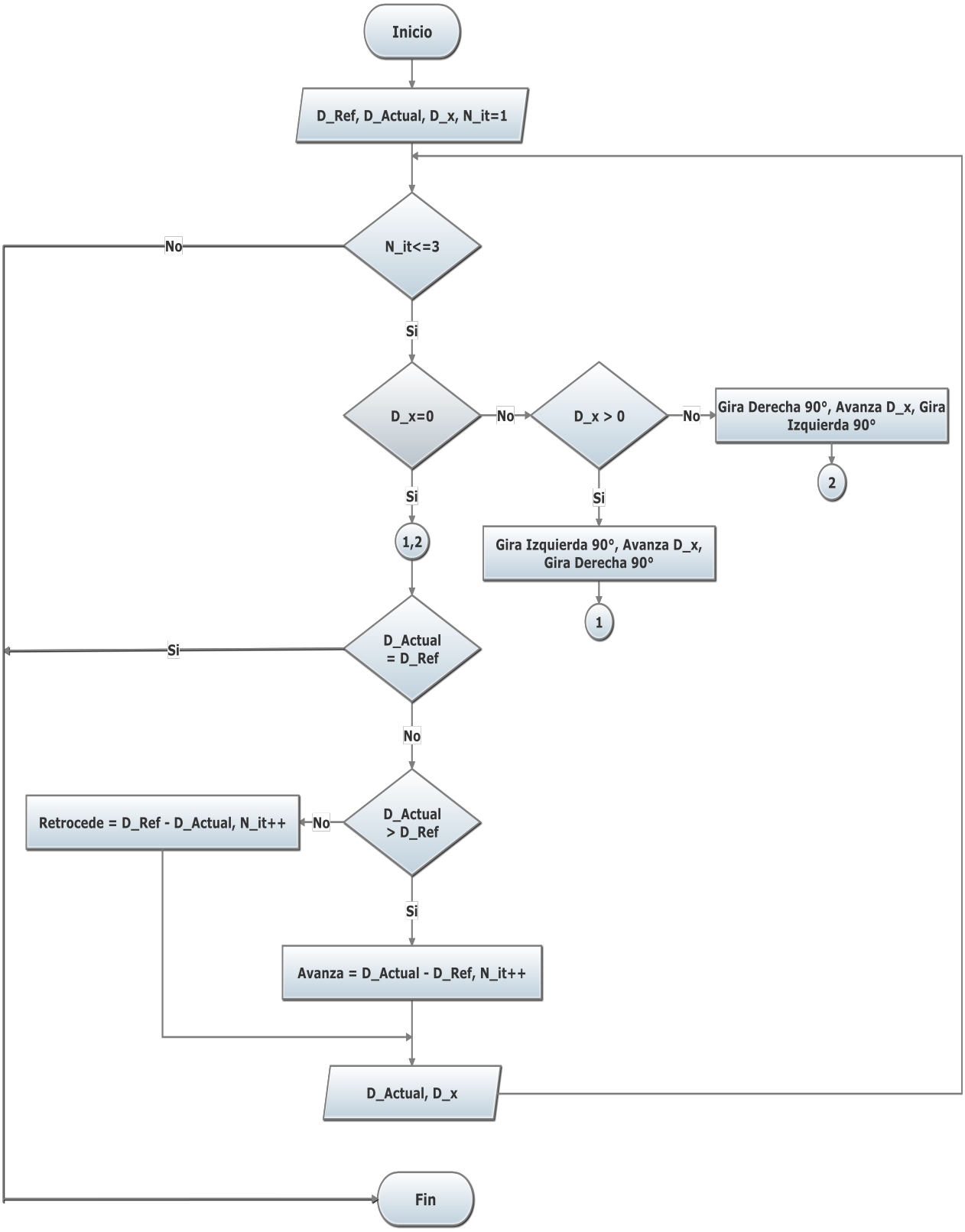


Figura 3.17: Diagrama de Flujo para Automatizar el Sistema de Visión

Capítulo 4

Pruebas y Resultados

En este capítulo se describen las pruebas realizadas y los resultados obtenidos del sistema de visión puesto en funcionamiento.

4.1. Alcance del software de registro de imágenes bajo condiciones ideales

Lo primero que necesitamos es verificar el alcance del software de registro de imágenes implementado, en relación a los desplazamientos máximos que es capaz de manejar, para cerciorarnos que será de utilidad en nuestra aplicación. Para ello se realizan pruebas de distancias y desplazamientos, es decir, del factor de escala $\boldsymbol{A}$ y del desplazamiento (a izquierda y derecha) t_x respecto al plano del objetivo, ya que se considera que la cámara se encuentra fija a la plataforma y sin posibilidad de realizar algún movimiento con respecto a ella que afecte el resto de los parámetros de la matriz de transformación (Ecuación 2.11). En cuanto a la perspectiva, se espera que sea pequeña en todo momento y que la posición inicial de partida es de frente al objetivo.

Se considera una imagen similar a la escena de referencia real y de las mismas dimensiones que las imágenes capturadas por la cámara web, en este caso una imagen de *640*x*480* pixeles, ver Figura 4.1. Posteriormente se crean copias de la imagen con modificaciones conocidas del factor de escala y de desplazamiento. Las copias de la imagen serán evaluadas con el software y se determinaran sus alcances al trabajar con imágenes ideales, en secciones posteriores se realizarán pruebas con imágenes reales capturadas por la webcam.

Para estas pruebas modificamos el factor de escala en un intervalo desde *0.2* a *3.0* con incrementos de *0.2* y el desplazamiento desde *-240* a *240* con incrementos de *40* pixeles. Lo que representa distancias aproximadas desde *16cm* hasta *300cm* entre cámara y objetivo para el factor de escala y $\pm 25cm$ para desplazamientos. El registro de imágenes se realiza en una LapTop que cuenta con un Procesador Intel Centrino Core 2 Duo a 2.20GHz, 4GB de memoria RAM y sistema operatico Microsoft Windows 7 Ultimate.

Figura 4.1: Imagen utilizada para validación del software

Realizado el registro de imágenes, se obtienen los datos mostrados en la Tabla 4.1, la cual, muestra el factor de escala estimado por el software en una copia de la imagen de referencia, por ejemplo, para una imagen con una amplificación de *1/0.6* y *120* pixeles de desplazamiento se debe aplicar un factor de reducción de *0.6* mientras que el software estima un factor de *0.61*.

Los datos son mostrados de acuerdo a una escala de colores en donde, el color verde es asignado a los datos con un error máximo del $\pm 5\,\%$ en comparación con el valor real esperado, el color amarillo es asignado a los datos con $\pm 10\,\%$ de error y los datos fuera de estos porcentajes son mostrados en color rojo. De forma similar en la Tabla 4.2 se presentan los datos correspondientes a los desplazamientos, los porcentajes son del $\pm 20\,\%$, $\pm 50\,\%$ para el color verde y amarillo respectivamente y el color rojo es asignado a los datos con un error mayor al $\pm 50\,\%$.

Con la escala de colores anterior, en la Tabla 4.3 se indican los límites del software de registro de imágenes implementado. Podemos encontrar que el rango para el factor de escala se encuentra entre ***(0.8, 1.8)*** para casi todo el rango de desplazamientos considerado, lo que equivaldría aproximadamente a distancias ***(40, 220)cm***.

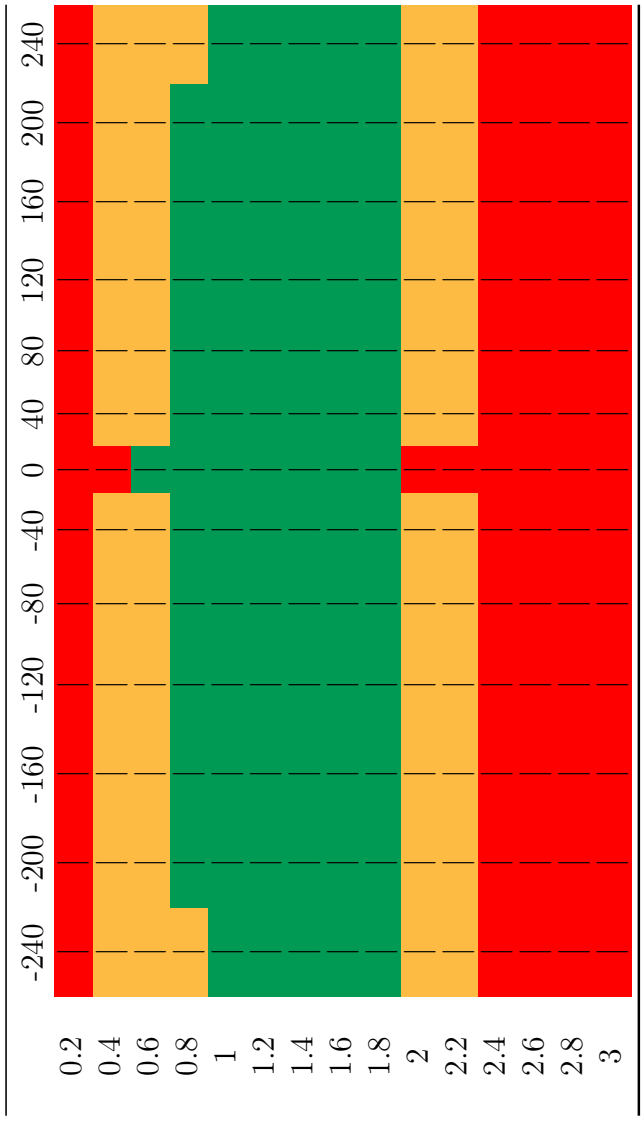
Tabla 4.1: Factor de Escala

Escala	Desplazamientos												
	-240	-200	-160	-120	-80	-40	0	40	80	120	160	200	240
0.2	**4.47**	**0.37**	**4.47**	**33**	**6.93**	**3.49**	**1.28**	**1.7**	**6.93**	**33**	**4.47**	**0.37**	**4.47**
0.4	**0.54**	**12.54**	**0.1**	**33**	**0.37**	**25.71**	**1.65**	**25.71**	**0.37**	**33**	**0.1**	**12.54**	**0.54**
0.6	**3.49**	**0.61**	**0.63**	**0.61**	**0.61**	**0.61**	**0.61**	**0.61**	**0.61**	**0.61**	**0.63**	**0.61**	**3.49**
0.8	**27.36**	**0.8**	**0.8**	**0.8**	**0.8**	**0.8**	**0.8**	**0.8**	**0.8**	**0.8**	**0.8**	**0.8**	**27.36**
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
1.2	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**	**1.21**
1.4	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**	**1.41**
1.6	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**	**1.6**
1.8	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**	**1.81**
2	**1.99**	**2.12**	**1.99**	**2.12**	**1.99**	**2.12**	**33**	**2.12**	**1.99**	**2.12**	**1.99**	**2.12**	**1.99**
2.2	**2.4**	**2.4**	**2.4**	**2.4**	**33**	**2.4**	**33**	**2.4**	**33**	**2.4**	**2.4**	**2.4**	**2.4**
2.4	**33**	**2.72**	**33**	**2.72**	**33**	**2.72**	**33**	**2.72**	**33**	**2.72**	**33**	**2.72**	**33**
2.6	**3.08**	**3.08**	**3.08**	**3.08**	**33**	**3.08**	**3.49**	**3.08**	**33**	**2.89**	**3.49**	**3.49**	**3.49**
2.8	**3.49**	**3.49**	**3.49**	**3.49**	**3.49**	**3.49**	**29.13**	**4.47**	**3.49**	**3.49**	**3.49**	**3.49**	**3.49**
3	**6.93**	**4.47**	**3.95**	**3.49**	**3.95**	**3.95**	**6.93**	**4.47**	**3.95**	**3.49**	**3.95**	**6.11**	**6.93**

Tabla 4.2: Desplazamientos

Escala	Desplazamientos												
	-240	-200	-160	-120	-80	-40	0	40	80	120	160	200	240
0.2	**-205**	**255**	**239**	**187**	**223**	**-196**	**-213**	**196**	**-223**	**254**	**-225**	**-255**	**205**
0.4	**-223**	**-102**	**-243**	**239**	**225**	**193**	**1**	**-193**	**201**	**191**	**243**	**102**	**223**
0.6	**-239**	**-255**	**-255**	**-200**	**221**	**143**	**0**	**-143**	**133**	**200**	**255**	**255**	**239**
0.8	**-176**	**-250**	**-200**	**-150**	**-100**	**-50**	**0**	**50**	**100**	**150**	**200**	**250**	**176**
1	**-240**	**-200**	**-160**	**-120**	**-80**	**-40**	**0**	**40**	**80**	**120**	**160**	**200**	**240**
1.2	**-200**	**-167**	**-133**	**-100**	**-67**	**-33**	**0**	**33**	**67**	**100**	**133**	**167**	**200**
1.4	**-171**	**-143**	**-114**	**-86**	**-57**	**-29**	**0**	**29**	**57**	**86**	**114**	**143**	**171**
1.6	**-150**	**-125**	**-100**	**-75**	**-50**	**-25**	**0**	**25**	**50**	**75**	**100**	**125**	**150**
1.8	**-133**	**-111**	**-89**	**-67**	**-44**	**-22**	**0**	**22**	**44**	**67**	**89**	**111**	**133**
2	**-120**	**-100**	**-80**	**-60**	**-40**	**-20**	**42**	**20**	**40**	**60**	**80**	**100**	**120**
2.2	**-109**	**-131**	**-73**	**-55**	**-37**	**-18**	**-46**	**18**	**37**	**54**	**73**	**131**	**109**
2.4	**-142**	**-84**	**-103**	**-50**	**-79**	**-17**	**-46**	**16**	**79**	**50**	**103**	**83**	**142**
2.6	**-92**	**-77**	**-62**	**-46**	**-31**	**-15**	**-1**	**15**	**31**	**46**	**62**	**77**	**92**
2.8	**-86**	**-71**	**-57**	**-43**	**-29**	**-15**	**-34**	**14**	**29**	**43**	**57**	**71**	**86**
3	**-87**	**-67**	**-54**	**-40**	**-33**	**-14**	**-7**	**13**	**26**	**40**	**53**	**61**	**87**

Tabla 4.3: Área de Convergencia entre Factor de Escala y Desplazamientos



4.2. Alcance del software de registro de imágenes con imágenes reales.

Conociendo los alcances del software de registro de imágenes, se procede a validar estos alcances con imágenes reales capturadas por la cámara web.

La webcam se ajusta a una base y se establece la referencia a una distancia de *100cm*, posteriormente la cámara es ubicada a distancias de *50cm*, *100cm* y *150cm*, realizando desplazamientos arbitrarios sobre una linea paralela al plano imágen dentro del rango de $\pm 25cm$. Los resultados son mostrados en la Tabla 4.4.

Se puede apreciar que el rango de desplazamientos se reduce conforme nos acercamos al objetivo, mientras que a la distancia de referencia y distancias mayores los límites se conservan.

Tabla 4.4: Datos Reales en comparación con Datos Obtenidos por software

Datos Reales		Datos por Software	
Distancia[cm]	Desplazamiento[cm]	Distancia[cm]	Desplazamiento[cm]
50	-15	50.56	-13.28
	-11		-9.61
	-5		-3.72
	11		10.39
	13		10.72
100	-30	100	-
	-22		-20.02
	-13		-10.79
	13		11.82
	20		17.11
	30		-
150	-40	148.38	-
	-30		-
	-20		-15.84
	-13		-11.92
	20		17.95
	30		-
	40		-

Estos datos son graficados en la Figura 4.2 donde, en color rojo se muestran los datos aproximados por software, mientras que en color azul se distinguen los datos de la ubicación real de la cámara.

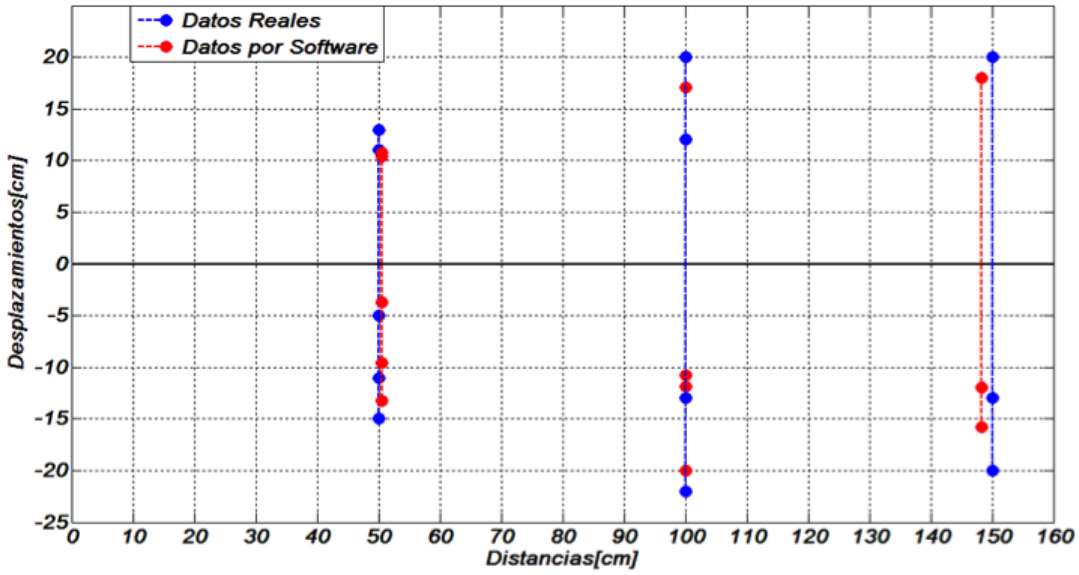


Figura 4.2: Resultados de la cámara

Como se aprecia, en la Tabla 4.4 y en la Figura 4.2, los datos obtenidos por software presentan un error máximo de *1.6 cm.* en distancias perpendiculares al objetivo. Mientras que en los desplazamientos paralelos al plano del objetivo, el error máximo es de *4.16cm.* Estas cantidades se encuentran dentro de los porcentajes establecidos en la Sección 4.1.

4.3. Sistema de visión en operación

4.3.1. Validación del Factor de Escala

Después de la comprobación de la efectividad del algoritmo de registro de imágenes para la estimación de la ubicación de la cámara en un entorno real estructurado, se procede con la puesta en marcha del sistema de visión completo, esperando que éste sea capaz de estimar su posición actual y eventualmente reducir su distancia con respecto a la posición de referencia.

Primeramente se realizan pruebas para validar las distancias respecto al plano del objetivo, es decir verificando el factor de escala. El sistema se programa para que realice tres iteraciones de aproximación con el fin de encontrar la máxima distancia a la cual el sistema será capaz de estimar su ubicación y aproximarse a la referencia deseada.

Con la misma distancia de referencia (*100cm*), y con ayuda de la interfaz de usuario desarrollada en la Sección 3.4.1. El sistema es ubicado a una distancia de *20cm* respecto del objetivo, posteriormente será alejado con intervalos de *10cm*, hasta una distancia de *250cm*.

Después de que el sistema ha sido ubicado sin desplazamientos a izquierda o derecha, el sistema es puesto en marcha, y para poder aproximarse a la distancia de referencia deseada realiza las siguientes etapas:

1. Captura una imagen de su posición actual.
2. Estimar la posición en la cual se encuentra.
3. Toma de decisiones, es decir, la rutina a realizar (avanzar o retroceder).
4. Ejecuta la rutina seleccionada.

Este procedimiento se repite en tres ocasiones, con lo cual se espera que el sistema alcance la posición de referencia deseada (de *100cm* respecto del objetivo). Los resultados obtenidos son mostrados en Tabla 4.5, donde la primer columna corresponde al número de prueba realizada, la segunda corresponde a la distancia inicial del sistema. La siguiente columna muestra la distancia que el software de registro de imágenes ha estimado en una primera iteración, en la cuarta columna se observa la estimación de la posición final que el software obtiene después de tres aproximaciones realizando los movimientos necesarios en cada una de ellas para aproximarse a la posición de referencia deseada. La última columna muestra la distancia final real del móvil, es decir, la distancia real respecto al objetivo a la cual se encuentra el sistema.

Realizado un análisis se observa que es posible estimar la posición del sistema en el rango establecido en la Sección 4.1 aun cuando se tiene un error de *6.62cm*, partiendo de una distancia de *220cm*. Para distancias fuera de este rango, el sistema no garantiza la aproximación a la distancia de referencia ya que eventualmente el sistema supera el $\pm 10\,\%$ el error en su posición.

Tabla 4.5: Resultados de la estimación de distancias respecto al objetivo

$N°$ Prueba	Distancia	Primer Estimación	Estimación Final	Posición Final Real
1	10	-	-	-
2	20	-	-	-
3	30	-	-	-
4	40	28.14	96.73	95
5	50	50.56	106.86	105
6	60	56.16	106.86	105
7	70	66.74	106.86	106
8	80	79.1	106.86	105
9	90	87.5	106.86	107
10	100	100	100	100
11	110	110.45	93.56	98
12	120	121.93	90.48	97
13	130	134.54	93.56	100
14	140	143.62	90.48	97
15	150	148.38	93.56	98
16	160	163.58	90.48	97
17	170	174.54	93.56	96
18	180	186.2	93.56	96
19	190	198.62	93.56	95
20	200	218.75	103.38	108
21	210	233.26	103.38	110
22	220	248.7	103.38	109
23	230	248.7	103.38	109
24	240	-	-	-
25	250	-	-	-

Cabe destacar también que en distancias muy pequeñas, es decir muy cercanas al objetivo, el área de captura de la imagen es muy reducida, es decir que no se cuenta con la información necesaria y suficiente para realizar el registro de imágenes correspondiente (imagen **B** en Figura 4.3). En distancias grandes (distancias mayores a *220cm*), aunque se cuenta con imágenes donde se puede apreciar toda la escena de referencia, la imagen posee, en su mayoría, información irrelevante para el registro correspondiente, además en distancias grandes la resolución de la cámara carece de una buena calidad en la imagen capturada (imagen **C** en Figura 4.3), por lo que en los dos casos descritos, el sistema no es capaz de garantizar la estimación de la distancia real a la cual se encuentra el sistema.

Esto se puede observar en la Figura 4.3 donde la imagen **A** es la imagen de referencia y las imágenes **B** y **C** ha sido capturada a distancias de *20cm* y *240cm* del objetivo respectivamente.

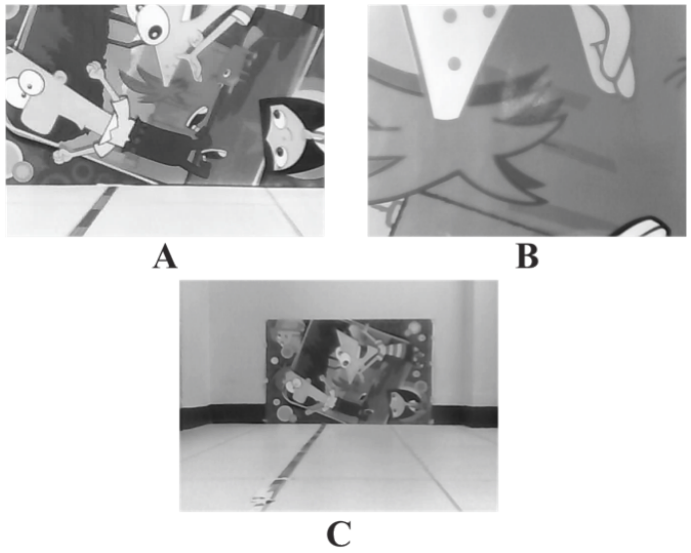


Figura 4.3: Imágenes en comparación con la imagen de referencia

La Figura 4.4 muestra las gráficas correspondientes a las aproximaciones realizadas por el sistema, la gráfica en color azul representa la posición original del sistema, en color negro se aprecia la primer estimación de la posición, mientras que en color verde se aprecia la estimación final que el sistema ha alcanzado después de tres aproximaciones y haber realizado los movimientos necesarios para aproximarse a la posición de referencia deseada, en tanto que, en color rojo podemos observar la distancia real medida. Se observa que el error de aproximación a la posición real deseada disminuye considerablemente desde la primer hasta la última estimación, de forma similar ocurre con la posición real alcanzada por el sistema.

En la Figura 4.5 se observa con mayor detalle la distancia final estimada en color negro, y en color rojo la distancia final real alcanzada por el sistema.

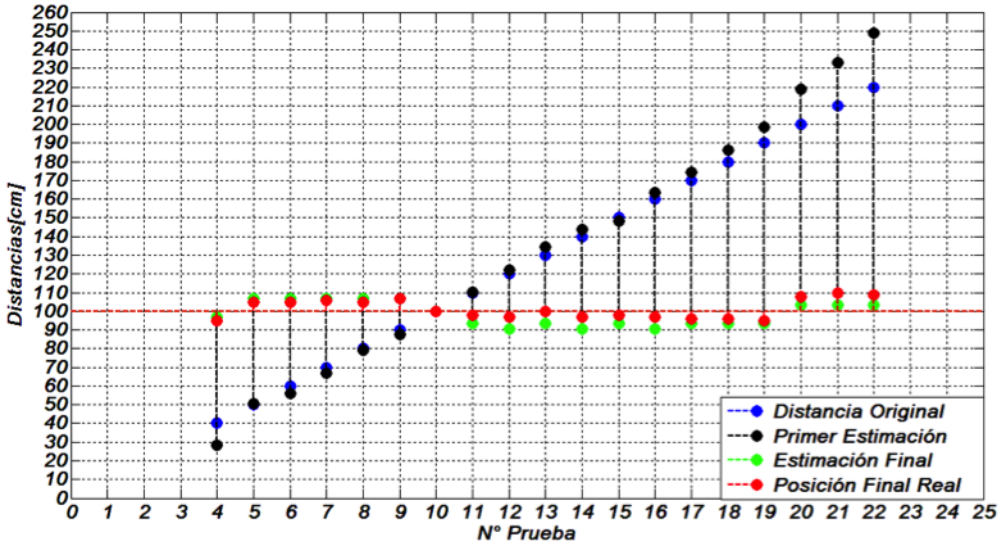


Figura 4.4: Primer aproximación por software

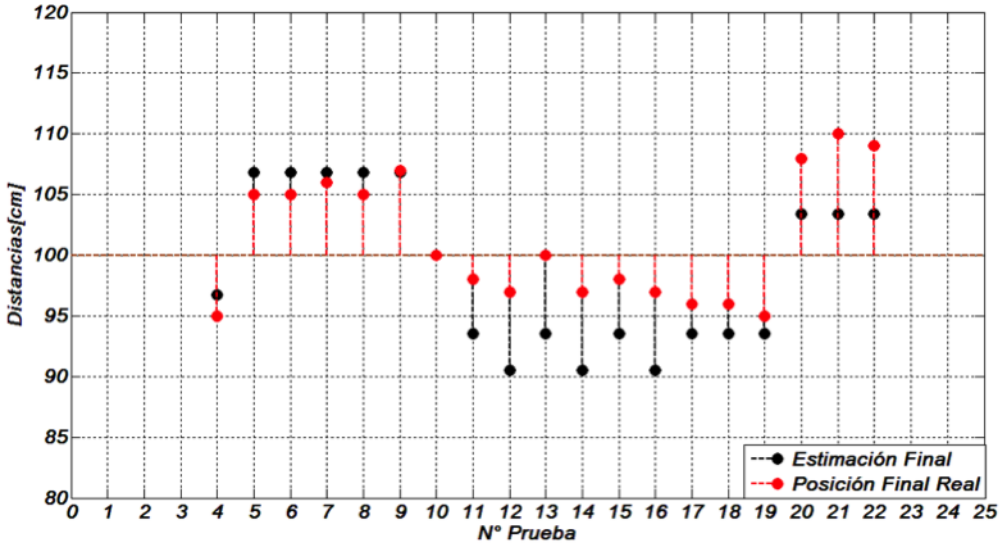


Figura 4.5: Comparación de las Aproximaciones por software

4.3.2. Validación del Factor de Escala y Desplazamientos

Conservando la distancia de referencia, el sistema es ubicado en diferentes posiciones con respecto al objetivo que incluyen distancias que van desde *20cm* hasta *250cm* y desplazamientos arbitrarios a izquierda y derecha en el rango de $\pm 25cm$. Después de ser ubicado el sistema es puesto en operación siguiendo la rutina utilizada en la Sección 4.3.1, con una diferencia en el paso 3, ya que en esta ocasión, además de tomar decisiones en avanzar o retroceder, primeramente tomará las decisiones correspondientes para avanzar hacia la izquierda o hacia la derecha dependiendo del desplazamiento estimado. De igual manera esta rutina es repetida en tres ocasiones.

Los resultados son mostrados en la Tabla 4.6, donde en primer lugar se muestra la ubicación original del sistema, enseguida la primer y última estimación de su posición, así mismo, muestra la ubicación final real que el sistema alcanza después de realizar tres aproximaciones y ejecutar las rutinas de desplazamiento necesarias.

Se aprecia que la diferencia entre la posición inicial y la final del sistema de visión en su mayoría disminuye, mientras que en ocasiones el sistema no es capaz de aproximarse a la referencia deseada. Los errores presentes son debidos, muy probablemente, a la presencia de perspectiva, la cual se hace presente después de que el móvil ha ejecutado las rutinas (giros y desplazamientos) para aproximarse a la posición de referencia deseada. Esto debido a que el móvil tiene que realizar un número de movimientos mayor para reducir su distancia con respecto a la referencia deseada, es decir, el móvil tiene que compensar distancias y también desplazamientos. Una de las causas es debido a la arquitectura de la plataforma móvil y la imperfección de sus sistema de tracción, lo que hace prácticamente imposible que el móvil trace trayectorias rectas y, mientras más movimientos realice mayor será el desvió de la trayectoria recta trazada, con esto el móvil genera perspectiva y en ocasiones llegar a perder de "vista" al objetivo.

Tabla 4.6: Ubicación del sistema

Ubicación Original		Primer Estimación		Estimación Final		Ubicación Final Real	
Distancia	Desplazamiento	Distancia	Desplazamiento	Distancia	Desplazamiento	Distancia	Desplazamiento
10	-	-	-	-	-	-	-
20	-15	24.08	-7.59	110.45	-	110	-
20	15	37.96	-2.47	106.86	-8.5	100	-
30	-15	24.08	-6.22	96.73*	-16.18	170	-
30	15	43.86	7.89	110.45	-10.14	110	5
41	1.5	32.74	3.7	100	-1.78	108	5.5
44	3.5	37.96	3.51	110.45	0.91	100	0
47	-4.5	24.08	-1.49	114.16	-5.88	110	12
52	5.5	48.81	7.59	110.45	5.58	105	5.5
61	-2.5	37.96	6.01	71.45	-3.04	109	0
69	0	64.49	1.52	106.86	-8.61	106	12
70	1.5	66.74	2.57	110.45	-5.47	106	15
78	-7	50.56	-5.22	100.00*	-9.85	165	12
96	2	62.3	-3.28	100.00*	-8.82	156	6
111	2	110.45	2.51	90.48	-4.34	100	9
114	0	114.16	0	96.73	-0.97	97	9.5
120	-3	69.06	-5	96.73	-5.13	92	7
126	3.5	126	2.32	96.73	-6.8	97	-1
142	4	148.38	5.88	90.48	-5.86	93	-14
170	0	174.54	-1.76	93.56	-2.05	100	7
170	-4	168.98	-36.4	90.48	-5.19	95	5
180	3	186.2	1.5	69.06*	-1.84	111	9
200	0	218.75	-35.32	103.38	-0.96	110	28.5
200	-2.5	218.75	-35.32	103.38	-1.07	110	0

∗Datos obtenidos con la presencia de gran perspectiva

Capítulo 5

Conclusiones y Trabajos Futuros

A continuación se describen las conclusiones obtenidas y algunos trabajos futuros a implementar en el sistema de visión desarrollado.

5.1. Conclusiones

El procedimiento propuesto para la calibración de la cámara permitió establecer la relación entre los parámetros del modelo de transformación 2D y las coordenadas de la cámara con respecto a una referencia, además de que dicho procedimiento es sencillo de implementar, y puede ser extendido a otros tipos de parámetros de transformación como lo es la perspectiva.

Se pudo comprobar la importancia del software de registro de imágenes para la determinación de la ubicación de la cámara con respecto a una referencia. Este debe presentar una buena exactitud (del orden de milimetros) ya que la reducción en el error de posicionamiento del móvil depende de ello. Además el software debe ser útil en situaciones reales de navegación autónoma, en donde, como hemos comprobado, esperamos transformaciones geométricas grandes y sobre todo combinadas, es decir, translaciones a lo largo del eje x, el factor de escala y la perspectiva.

La importancia de la estrategia piramidal se demuestra en la obtención de los parámetros de transformación de forma relativamente rápida en comparación con el gradiente descendente estocástico que tarda mas de *50 segundos* en realizar el registro de imágenes correspondiente, por el contrario hemos obtenido un tiempo de *6 segundos* con el software que hemos implementado para el registro de imágenes correspondiente, además de que no necesita ser inicializado con parámetros cercanos a los reales a diferencia del implementado en [Miranda-Luna et al., 2008].

El sistema que hemos desarrollado, a pesar de la arquitectura del móvil que no permite trazar desplazamientos estrictamente rectos, es capaz de estimar su ubicación entre los *40cm* y *220cm* de distancia respecto al objetivo con un error máximo de *6.62cm* en distancias superiores a *200cm* y desplazamientos de $\pm 25cm$ respecto a la línea perpendicular que pasa por el centro del objetivo. Este rango es relativamente amplio en comparación con el desarrollado en [Cruz Hernández, 2011] donde realizan la navegación y posicionamiento de un robot humanoide con un error aproximado de *10cm*.

Con esto podemos demostrar que los objetivos han sido alcanzados, ya que, es posible determinar la ubicación de la plataforma móvil en un entorno 3D estructurado y además aproximarse a la ubicación de referencia reduciendo su error de posicionamiento. Demostrando así que el software de registro de imágenes 2D basado en un modelo de transformación rígido,y una búsqueda exhaustiva en combinación con la estrategia de optimización piramidal, es útil para la estimación de la ubicación de robots móviles.

5.2. Trabajos Futuros

Sin duda este sistema requiere de mejoras, por lo que se propone los siguientes trabajos futuros, con los cuales poder realizar un mejor y mas eficiente sistema de visión.

Como se mencionaba en la sección anterior el proceso de calibración es sencillo, por lo que se propone trabajar en la perspectiva, misma que se debe incluir para darle mayor aplicabilidad a nuestro sistema de visión. Así mismo incluir la perspectiva en el algoritmo de procesamiento de imágenes desarrollado para obtener mejores resultados de la estimación de la ubicación del sistema.

Se propone también incluir un sistema de control, que permita una mejor navegación de la plataforma móvil. Una propuesta es incluir una cámara extra ubicada en la base de la plataforma enfocada hacia el suelo como en [Gifford, 2009] , que pemita al móvil no perder la trayectoria trazada. En caso contrario optar por la implementación del sistema con una plataforma móvil de arquitectura no diferencial.

Una mejora más es la implementación de este sistema con un diseño empotrado, de manera que demande menos energía y de esta manera se puede pensar en independizar el sistema de la línea de alimentación de 120v, utilizada actualmente para alimentar la LapTop.

Bibliografía

[Adelson et al., 1984] Adelson, E. H., Anderson, C. H., Bergen, J. R., Burt, P. J., and Ogden, J. M. (1984). Pyramid Methods in Image Processing. *RCA Engineer*, 29(6):33–41.

[Alvarado Legaria, 2007] Alvarado Legaria, S. (2007). Diseño y Modelado de una Arquitectura VLSI para el Reconocimiento de Imágenes en un Sistema de Visión Artificial con base en La Transformada Discreta Wavelet y Memorias Asociativas Morfológicas. Tesis para obtener el título de ingeniero en electrónica, Universidad Tecnológica de la Mixteca (UTM), Huajuapan De León.

[Anuta, 1970] Anuta, P. E. (1970). Spatial registration of multispectral and multitemporal digital imagery using fast fourier transform techniques. *IEEE Transactions on Geoscience Electronics*, 8(4):353–368. ISSN: 0018-9413.

[Armendariz., 2003] Armendariz., M. A. M. (2003). *Visión Artificial Estéreo Con Aplicación Al Control De Un Brazo De Robot*. PhD thesis, Instituto Politécnico Nacional, México, D.F.

[Auat et al., 2011] Auat, F. A., De la Cruz, C., Carelli, R., and Bastos Filho, T. F. (2011). Navegación autonoma asistida basada en slam para una silla de ruedas robotizada en entornos restringidos. *Revista Iberoamericana de Automática e Informática Industrial (RIAI)*, 8(2):81–92. ISSN: 1697-7912.

[Bjerknes et al., 2007] Bjerknes, J. D., Liu, W., Winfield, A. F., Melhuish, C., and Lane, C. (2007). Low cost ultrasonic positioning system for mobile robots. In *Proceesiding of Towards Autonomous Robotics Systems (TAROS 2007)*, pages 107–114, Aberystwyth.

[Chen et al., 1994] Chen, Q.-s., Defrise, M., and Deconinck, F. (1994). Symmetric phase-only matched filtering of fourier-mellin transforms for image registration and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(12). ISSN: 0162-8828.

[Computing, 2010] Computing, M. (2010). *User's Guide*. Measurement Computing Corporation, Norton, Massachusetts 02766. Manual de Usuario.

[Cruz Hernández, 2011] Cruz Hernández, E. R. (2011). Desarrollo de un sistema de visión para la localización y navegación de robots humanoides. Tesis para obtener el grado de maestro en ciencias de la ingeniería, Instituto Tecnológico Y De Estudios Superiores De Monterrey, Atizapan de Zaragoza, Edo. De México.

[Cuadrado Vaca, 2005] Cuadrado Vaca, A. L. (2005). Automatización del Sistema de Entretenimiento en Control de Procesos CPTS-1 Mediante la Tarjeta de Adquisición de Datos USB PMD-1208LS. Proyecto de grado para la obtencion del título en ingeniero de telecomunicación, Escuela Politécnica del Ejército, Sangolquí-Ecuador.

[De la Escalera Hueso, 2001] De la Escalera Hueso, A. (2001). *Visión por Computador, Fundamentos y Métodos*. ISBN: 84-205-3098-0. Prentice Hall, Madrid.

[Donald Hear and Baker., 1997] Donald Hear, M. and Baker., P. (1997). *Computer Graphics: C Version*. ISBN: 0-13-530924-7. Prentice Hall, Upper Saddler River, N. J.

[Ferruz and Olleron, 1995] Ferruz, J. and Olleron, A. (1995). Estimación de movimientos de vehículos en entornos no estructurados empleando una secuencia de imágenes. In *XVI Jornadas de Automática.*, pages 40–50.

[García Capel, 2007] García Capel, L. E. (2007). Aplicación de Medidas Estadísticas de Similitud al Registro de Imagen Multimodo. Proyecto de fin de carrera. ingeniero de telecomunicación, Universidad Politécnica de Cartagena.

[Gifford, 2009] Gifford, C. M. (2009). Low-cost mobile robot localization using only a downward-facing webcam. Technical report, University of Kansas, Lawrence.

[Hellier et al., 2001] Hellier, P., Barrillot, C., Memin, E., and Perez, P. (2001). Hierarchical Estimation of a Dense Deformation Field for 3-d Robust Registration. *Medical Imaging, IEEE Transactions on*, 20(5):388–402. ISSN: 0278-0062.

[Hernández et al., 2003] Hernández, S., Torres, J. M., Morales, C. A., and Acosta, L. (2003). A new low cost system for autonomus robot heading and position localization in a closed area. *Autonomus Robots*, 15(2):99–110. ISSN: 0929-5593.

[Krotkov, 1989] Krotkov, E. (1989). Mobile robot localization using a single image. In *Robotics and Automation,1989. Proceedings., 1989 IEEE International Conference on*, volume 2, pages 978–983, Scottsdale, AZ. IEEE. ISBN: 0-8186-1989-4.

[Langsam et al., 1997] Langsam, Y., Augenstein, M. J., and Tenenbaum, A. M. (1997). *Estructuras de Datos con C y C++*. Prentice Hall, Mexico, segunda edition. ISBN: 0-13-036997-7.

[Li et al., 1995] Li, H., Manjunath, B., and Mitra, S. K. (1995). A contour-based approach to multisensor image registration. *IEEE Transaction on Image Processing*, 4(3):320–334. ISSN: 1057-7149.

[MacMillan et al., 2011] MacMillan, N., Allen, R., Marinakis, D., and Whitesides, S. (2011). Ranged-based navigation system for mobile robots. In John's, S., editor, *Canadian Conference on Computer and Robot Vision (CRV)*, pages 16–23, 25–27, St. Johns, NL.

[Marr, 1982] Marr, D. (1982). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. ISBN: 0716712849. New York: Freeman and Company, San Francisco: W. H.

[Mateo Ingelmo, 2009] Mateo Ingelmo, A. (2009). *Estudio De Técnicas De Caracterización De La Figura Humana,Para Su Posible Aplicación A Problemas De Reconocimiento De Género*. PhD thesis, Universitat Jaume I.

[MathWorks, 2012] MathWorks (2012). Mathworks. Online: [http://www.mathworks.com/].

[Micó, 1996] Micó, M. L. (1996). *Algoritmos de Búsqueda de Vecinos Más Próximos en Espacios Métricos*. PhD thesis, Universidad Politécnica de Valencia, Valencia, España.

[Miranda-Luna et al., 2008] Miranda-Luna, R., Daul, C., Blondel, W. C. P. M., Hernandez-Mier, Y., Wolf, D., and Guillemin, F. (2008). Mosaicing of Bladder Endoscopic Image Sequences: Distortion Calibration and Registration Algorithm. *IEEE Transaction on Biomedical Engineering*, 55(2):541–553. ISSN: 0018-9294.

[PerfectChoise, 2012] PerfectChoise (2012). *PerfectChoise Tu vida Digital - Manual de Usuario*. Manual de Usuario.

[Posada et al., 2004] Posada, R., Daul, C., and Miranda, R. (2004). Towards a fractioned treatment in conformational radiotherapy using 3d-multimodal data registration. In *Image Processing, 2004. ICIP'04. 2004 International Conference on*, volume 3, pages 1911–1914, Vandoeuvre-lés-Nancy, France. ISSN: 1522-4880.

[RAE, 2012] RAE (2012). Real academia española. Online: [http://www.rae.es/rae.html].

[Raguraman et al., 2009] Raguraman, S., Tamilselvi, D., and Shivakumar, N. (2009). Mobile robot navigation using fuzzy logic controller. In *International Conference on Control, Automation,Communication and Energy Conservation (INCACEC 2009)*, pages 1–5, Perundurai, Tamilnadu. ISBN: 978-1-4244-4789-3.

[Reddy and Chatterji, 1996] Reddy, B. S. and Chatterji, B. (1996). An fft-based technique for translation, rotation, and scale-invariant image registration. *IEEE Transactions on Image Processing*, 5(5):1266–1271. ISSN: 1057-7149.

[Robotics, 2002] Robotics, Z. (2002). *Zagros Robotics Max/Rex*. Zagros Robotics, St. Louis, MO. Manual de Usuario.

[Rodríguez Santiago et al., 2012] Rodríguez Santiago, A. L., Miranda Luna, R., Arias Aguilar, J. A., and Antonio García, A. (2012). Calibración de una Cámara para un Sistema de Visión Mono-Cámara Económico. *VIII Semana Nacional de Ingeniería Electónica SENIE 12*, pages 348–356. ISBN: 978-607-477-902-8.

[Sarrut, 2000] Sarrut, D. (2000). *Recalage Multimodal et Plate-Forme d' Imagerie Médicale à Accès Distantimagerie Médicale à Accès Distant*. PhD thesis, Université Lumière Lyon 2.

[Sucar, 2012] Sucar, L. E. (2012). Visión computacional. Online: [http://ccc.inaoep.mx/ esucar/Libros/vision-sucar-gomez.pdf].

[Thévenaz et al., 1998] Thévenaz, P., Ruttimann, U. E., and Unser, M. (1998). A Pyramid Approach To Subpixel Registration Based on Intensity. *IEEE Transactions On Image Processing*, 7(1):27–41. ISSN: 1057-7149.

[Vassal, 1998] Vassal, P. (1998). *Fusion d' images Multi-Modales pour la Radiothérapie Conformationnelle : Application au Repositionnement du Patient*. PhD thesis, Université Joseph Fourier, Grenoble I.

[Vázquez Jiménez, 2005] Vázquez Jiménez, I. (2005). Autolocalización de un robot móvil por medio de visión computacional en espacios cerrados. Tesis para obtener el título de ingeniero en computación, Universidad Nacional Autónoma De México, San Juan de Aragón, Estado de México.

[Vicente Lober, 2003] Vicente Lober, J. A. (2003). *Técnicas de Inteligencia Artificial en la Construcción de Robots Móviles Autónomos*. PhD thesis, Universidad De Salamanca, Salamanca.

[Viramontes and González, 2007] Viramontes, J. L. and González, E. J. (2007). Visión para posicionar un manipulador utilizando un sistema de visión mono-ocular. In *6to. Congreso Nacional de Mecatrónica*, San Luis Potosí.

[Zhang, 1999] Zhang, Z. (1999). Flexible Camera Calibration by a Viewing a Plane From Unknown Orientations. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision (iccv99)*, volume 1, pages 666–673, 20–27, Kerkyra. IEEE. ISBN: 0-7695-0164-8.

[Zokai and Wolberg, 2005] Zokai, S. and Wolberg, G. (2005). Image Registration Using Log-Polar Mappings for Recovery of Large-Scale Similarity and Projective Transformations. *IEEE Transactions on Image Processing*, 14(10):1422–1434. ISSN: 1057-7149.

Apéndice A

Codigo Matlab

A.1. Inicialización de GUIDE, DAQ y Cámara

Inicialización de GUIDE

```
function varargout = Recalage_User(varargin)
% RECALAGE_USER M-file for Recalage_User.fig
%      RECALAGE_USER, by itself, creates a new RECALAGE_USER or raises the existing
%      singleton*.
%
%      H = RECALAGE_USER returns the handle to a new RECALAGE_USER or the handle to
%      the existing singleton*.
%
%      RECALAGE_USER('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in RECALAGE_USER.M with the given input arguments.
%
%      RECALAGE_USER('Property','Value',...) creates a new RECALAGE_USER or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Recalage_User_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Recalage_User_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Recalage_User

% Last Modified by GUIDE v2.5 21-Nov-2012 16:55:55

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Recalage_User_OpeningFcn, ...
                   'gui_OutputFcn',  @Recalage_User_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before Recalage_User is made visible.
function Recalage_User_OpeningFcn(hObject, eventdata, handles, varargin)
% Choose default command line output for Recalage_User
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
clc;
handles.output = hObject;
% Update handles structure
handles.rgb = [];
handles.noback = [];
guidata(hObject, handles);

scrsz = get(0, 'ScreenSize');
pos_act=get(gcf,'Position');
xr=scrsz(3) - pos_act(3);
xp=round(xr/2);
yr=scrsz(4) - pos_act(4);
yp=round(yr/2);
set(gcf,'Position',[xp yp pos_act(3) pos_act(4)]);
```

Inicialización de la DAQ

```
global i time dio lines D_X X D_Y Y A R rgbBN;
i=0;
%***************DAQ**************************
dio = digitalio('mcc',0);
lines = addline(dio,0:3,'out');
putvalue(dio,[0 0 0 0]);
set(handles.text19,'Enable','off');
set(handles.text20,'Enable','off');
set(handles.text21,'Enable','off');
set(handles.text22,'Enable','off');
set(handles.text23,'Enable','off');
```

Inicialización de la Cámara

```
% This sets up the video camera and starts the preview
% Only do this when it's invisible
if strcmp(get(hObject,'Visible'),'off')
try
```

```
handles.vidobj = videoinput('winvideo');
% Update handles structure
start(handles.vidobj);
guidata(hObject, handles);
vidRes = get(handles.vidobj, 'VideoResolution');
nBands = get(handles.vidobj, 'NumberOfBands');
hImage = image(zeros(vidRes(2), vidRes(1), nBands),'Parent',handles.axes1);
        preview(handles.vidobj,hImage);
        %preview(handles.vidobj);
catch
msgbox('NO HAY CÁMARA CONECTADA. Cargando Profile.jpg.')
hImage = image(imread('profile.jpg'), 'Parent',handles.axes1);
end

end
% Choose default command line output for video
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% Get default command line output from handles structure
varargout{1} = handles.output;
```

Función para Poner en Funcionamiento el Sistema

```
% --- Executes on button press in start.
function start_Callback(hObject, eventdata, handles)
% hObject    handle to start (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns start state of start
global D_X X D_Y Y A R time dio name_img;
disp('Iniciamos');
handles.ref = imread('ref.bmp');
image(handles.ref,'Parent',handles.axes3);
axes(handles.axes3);
axis off;
set(handles.start,'ForegroundColor',[1,0.0,0.0]);
set(handles.start,'String','Processing');
% while(get(handles.start,'Value'))
for i=1:3
    captura(hObject, eventdata, handles);
    disp('Tomada');
    procesa();
```

```
    X=round(X);
    set(handles.text5,'String',D_X);
    set(handles.text7,'String',D_Y);
    set(handles.text9,'String',A);
    set(handles.text11,'String',R);
    set(handles.text13,'String',Y);
    set(handles.text15,'String',X);
    Camina(handles, X, Y, dio);
    set(handles.text19,'Enable','off');
    set(handles.text21,'Enable','off');
    set(handles.text22,'Enable','off');
    set(handles.text20,'Enable','off');
    set(handles.text23,'Enable','off');
    captura(hObject, eventdata, handles);
    disp('Tomada');
    procesa();
    X=round(X);
    set(handles.text5,'String',D_X);
    set(handles.text7,'String',D_Y);
    set(handles.text9,'String',A);
    set(handles.text11,'String',R);
    set(handles.text13,'String',Y);
    set(handles.text15,'String',X);
    get(handles.start,'Value');
    set(handles.start,'ForegroundColor',[0.0,0.498,0.0])
    set(handles.start,'String','Start');
    writefile(name_img,A,Y,D_X,X,'Posicion_Final',e_x);
    disp('Successful');
```

Función para realizar el procesamiento

```
function procesa()
global X Y D_X D_Y time A R rgbBN;
    tic
    disp('Procesa activo');
    im1=imread('ref.bmp');
    im2 = rgbBN;
    [Dcx, Dcy, A1, R] = recalage(im1, im2);
    D_X=Dcx; D_Y=Dcy;
    A=1/A1;
    m=106.4639;
b=-5.7088;
  Y=((m*A)+b)-0.7551;
    m2=9.6421;
```

```
X=(Dcx/A1)/m2;
disp('Procesa finalizado');
toc
```

A.2. Función de Registro de Imágenes

Registro de Imágenes

```
function [Dcx, Dcy, A, R] = recalage(im1, im2)
tic
% Normaliza imagenes
im1d=double(im1);
im2d=double(im2);

im1d=filtPasBa(im1d,0.5);
im2d=filtPasBa(im2d,0.5);

m=min(min(double(im1d)));
M=max(max(double(im1d)));
im1d=(im1d-m)*(255/(M-m));
m=min(min(double(im2d)));
M=max(max(double(im2d)));
im2d=(im2d-m)*(255/(M-m));
im1=uint8(im1d);
im2=uint8(im2d);

D_A=32;
h=12;
DeltaCent=1;
DYi_=-4;
DYs_= 4;
DeltaDY=1;  % Siempre debe ser entero
DXi_=-5;
DXs_= 6;
DeltaDX=1;  % Siempre debe ser entero

conta=1;
fin1=0;
fin2=0;

NumeroIteracion=0;
while fin1~=1 | fin2~=1
    NumeroIteracion=NumeroIteracion+1;

% ..............................................................1
if NumeroIteracion==1
    % Calculo del factor de reducción para la primera iteración
    [zy,zx]=size(im1);
    m=min(zy,zx); m=min(m);
```

```
    m2=m;
    i=0;
    while m2>14
        m2=m2/2;
        i=i+1;
    end
    factor_reduccion=0.5.^(i-1);
elseif factor_reduccion<1
    factor_reduccion=factor_reduccion*2;
    multiplicaDcx=1;
    if factor_reduccion==1
        fin1=1;
    end
end
% ..................................................................1

im1p=funreduce(im1, factor_reduccion);
im2p=funreduce(im2, factor_reduccion);

zim1p=size(im1p);
m3=min(zim1p);
radiomin=floor(m3/2);

% ..................................................................2
if NumeroIteracion==1
    % Calculo de Dcxi, Dcxs, Dcyi, Dcys para la primera iteración
    [zy2,zx2]=size(im1p);
    Dcxi=-ceil(zx2/3);
    Dcxs=-Dcxi;
    Dcyi=-ceil(zy2/3);
    Dcys=-Dcyi;
elseif multiplicaDcx==1;
    % Calculo de Dcxi, Dcxs, Dcyi, Dcys para la enesima iteración
    Dcxi=(2*Dcx)-1;
    Dcxs=(2*Dcx)+1;
    Dcyi=(2*Dcy)-1;
    Dcys=(2*Dcy)+1;
end
    multiplicaDcx=0;

if NumeroIteracion==1
    DAng=D_A; %360/11.25;
    DXi=DXi_;
    DXs=DXs_;
```

```
elseif DAng>1
    DAng=DAng/2;
    DXi=(2*DX)-1;
    DXs=(2*DX)+1;
    if DAng<=1
        fin2=1;
    end
else
    DXi=DX-1;
    DXs=DX+1;
end

NVA=floor(360/DAng);

if NumeroIteracion==1
    altura=h;
    DYi=DYi_;   % Desplazamiento para determinar la escala
    DYs=DYs_;
elseif altura<192
    altura=altura*2;
    DYi=(2*DY)-3;
    DYs=(2*DY)+3;
else
    DYi=DY-3;
    DYs=DY+3;
end
% ..............................................................2

if NumeroIteracion<5
    cantidad=floor((altura/2.5)*(NVA/2.5));
else
    cantidad=floor((altura/2.5)*(NVA/ (2.5*conta) ));
    conta=conta+0.5;
end

[IM1, Base]=
funrecapolaloginterp_BaseN_para_imagen_coarse_to_fine(im1p,...
 DAng, altura, 0, 0, 0, 0, 1);
[IM2, Base]=
funrecapolaloginterp_BaseN_para_imagen_coarse_to_fine(im2p,...
 DAng, altura, Dcxi, Dcxs, Dcyi, Dcys, DeltaCent);

IM2b=fun_circshiftea(IM2, DYi, DYs, DeltaDY, DXi, DXs, DeltaDX);
```

```
SD=fun_SD_(IM1, IM2b);
[c Ic]=min(SD);
[d,Id]=min(c);
[e,Ie]=min(d);
[f,If]=min(e);

w=If;               % w esta relacionada con DX
v=Ie(1,1,1,w);      % v esta relacionada con DY
z=Id(1,1,v,w);      % z esta relacionada con Dcy
x=Ic(1,z,v,w);      % x esta relacionada con Dcx
% y=Ib(1,x,z,v,w);
HubicacionMinimo=[x,z,v,w];

Dcx=   Dcxi+((x-1)*DeltaCent);
Dcy=   Dcyi+((z-1)*DeltaCent);
DY =   DYi +((v-1)*DeltaDY);
DX =   DXi +((w-1)*DeltaDX);
% [Desplazamiento en x, Desplazamiento en y, Escala Rotación]
[Dcx, Dcy, Base.^(-DY), -DX];
% [Dcx, Dcy, DY, -DX*DAng]
% figure; imshow(IM1,[]);
% figure; imshow(IM2b(:,:,x,z,v,w),[]);
end
A = Base.^(-DY);
R = -DX;
toc
```

A.2.1. Funciones Auxiliares para el Registro de Imágenes

Filtro Pasa Bajas

```
function [imf] = filtroPasaBajas(im,r)
% Esta función filtra (en el dominio de la frecuencia) una imagen de tamaño
% MxN en donde M y N pueden o no ser numeros pares.
% im=imagen de entrada que sera filtrada de tipo double
% r=frecuencia de corte (normalizada de 0 a 1)

[sy,sx]=size(im);
csx=sx/2;
csy=sy/2;

W=zeros(sy,sx);

if csx-floor(csx)==0    %N es numero par
    cx=csx+1;
else
    cx=csx+0.5;
end

if csy-floor(csy)==0    %M es numero par
    cy=csy+1;
else
    cy=csy+0.5;
end

for i=1:sx
    for j=1:sy
        x=i-cx;
        y=j-cy;
        a=(sx*r)/2;
        b=(sy*r)/2;
        if ((x/a).^2)+((y/b).^2)<=1
            W(j,i)=1;
        end
    end
end
F=fftshift(fft2(im));
Ff=F.*W;
imf=ifft2(ifftshift(Ff));
```

Calcula la Diferencia de Intensidades

```
function [SD] = fun_SD(IM1, IM2)
% Calcula la diferencia de intensidades entre IM1 y cada una de IM2

[a1, b1, c1, c2, c3, c4]=size(IM2);
SD=zeros(c1, c2, c3, c4);

for k1=1:c1
    for k2=1:c2
        for k3=1:c3
            for k4=1:c4
                IM1_IM2=(IM1-IM2(:,:,k1,k2,k3,k4)).^2;
                sombra=IM1.*IM2(:,:,k1,k2,k3,k4);
                M=max(sombra); M=max(M);
                sombra=sombra/M;
                sombra=ceil(sombra);
                n=sum(sum(sombra));
                SD(k1, k2, k3, k4)= sum(sum(IM1_IM2.*sombra));
                SD(k1, k2, k3, k4)=SD(k1, k2, k3, k4)/n;
            end
        end
    end
end
```

Transformación Geométrica − > Polar-Logarítmica

```
function [IMP,Base] =
        fun_Rec_a_Pol_a_Log_BaseN_para_imagen_coarse_to_fine(im, DAng,...
                                    altura, Dcxi, Dcxs, Dcyi, Dcys, DeltaCent)
DAng=floor(DAng);
[Ny,Nx]=size(im);
Mx=360;
My=ceil((sqrt((Nx*Nx)+(Ny*Ny))/2)+1);
cx=floor(Nx/2)+0.5;
cy=floor(Ny/2)+0.5;
Base=My.^(1/altura);
t=1:altura-1;
h=Base.^t;
t2=0:DAng:Mx-1;
cosdit=cosd(t2);
sindit=sind(t2);

NCentX=uint8(((Dcxs-Dcxi)/DeltaCent)+1);
NCentY=uint8(((Dcys-Dcyi)/DeltaCent)+1);
IMP=zeros(altura, uint8(Mx/DAng), NCentX);% , NCentY);

k2=0;
for a23=Dcyi:DeltaCent:Dcys
    k2=k2+1;
    k1=0;
    for a13=Dcxi:DeltaCent:Dcxs
        k1=k1+1;
        k0=0;
        for i=0:DAng:Mx-1
            k0=k0+1;
            for j=1:altura-1 %My-1
                x=h(j)*cosdit(k0);
                y=h(j)*sindit(k0);
                yy=cy - y + a23; % + 1.0;
                xx=x + cx - a13; % + 0.5;
                if xx>=1 & xx<=Nx & yy>1 & yy<=Ny
                    xi=floor(xx);
                    xs=ceil(xx);
                    yi=floor(yy);
                    ys=ceil(yy);
                    dxi=xx-xi;
                    dxs=xs-xx;
                    dyi=yy-yi;
                    dys=ys-yy;
```

```
                    if xi==xs & yi==ys
                        IMP(altura-j+1,k0,k1,k2)=im(yy,xx);
                    elseif xi==xs & yi~=ys
                        IMP(altura-j+1,k0,k1,k2)=(im(yi,xx)*dys) + (im(ys,xx)*dyi);
                    elseif xi~=xs & yi==ys
                        IMP(altura-j+1,k0,k1,k2)=(im(yy,xi)*dxs) + (im(yy,xs)*dxi);
                    else
                        IMP(altura-j+1,k0,k1,k2)=(im(yi,xi)*dys*dxs) +
                         (im(yi,xs)*dys*dxi) + (im(ys,xi)*dyi*dxs) +
                         (im(ys,xs)*dyi*dxi);
                    end
                end
            end
        end
    end
end
```

Cálculo del Factor de Reducción

```
function [im2] = fun_reduce(im, factor_reduccion)
% Unicamente las hace mas pequeñas

im=double(im);
[TamInY,TamInX]=size(im);

TamFinX=floor(TamInX*factor_reduccion);
TamFinY=floor(TamInY*factor_reduccion);

quita_en_x=floor((TamInX-TamFinX)/2);
quita_en_y=floor((TamInY-TamFinY)/2);

IM=fftshift(fft2(im));
IM2=IM(quita_en_y+1:TamInY-quita_en_y, quita_en_x+1:TamInX-quita_en_x);
im2=ifft2(ifftshift(IM2));
im2=(real(im2));
m=min(im2); m=min(m);
M=max(im2); M=max(M);
im2=(im2-m)*(255/(M-m));
im2=uint8(im2);
```

A.3. Distancias y Desplazamientos

Determina y Corrige Distancias y Desplazamientos

```
function Camina(handles, X, Y, dio)
t_right_right=0.95;
t_left_right=0.93;
t_right_left=0.93;
t_left_left=0.9;
t_sleep=1.0;
X = round(X);
Y = round(Y);
if(X == 0)
    chek_Y(Y, dio,handles);
elseif (X > 0)
    disp('left 90°');
        putvalue(dio,[1 1 0 1]);
        set(handles.text19,'Enable','off');
        set(handles.text20,'Enable','on');
        set(handles.text21,'Enable','off');
        set(handles.text22,'Enable','off');
        set(handles.text23,'Enable','off');
        pause(t_left_left);
        set(handles.text20,'Enable','off');
        putvalue(dio,[0 0 0 0]);
    pause(t_sleep);
    disp('forward_X_time');
        putvalue(dio,[1 1 0 0]);
        set(handles.text19,'Enable','on');
        set(handles.text20,'Enable','off');
        set(handles.text21,'Enable','off');
        set(handles.text22,'Enable','off');
        set(handles.text23,'Enable','off');
        pause(X*0.045);
        set(handles.text19,'Enable','off');
        putvalue(dio,[0 0 0 0]);
        pause(t_sleep);
    disp('right 90°');
        putvalue(dio,[1 1 1 0]);
        set(handles.text19,'Enable','off');
        set(handles.text20,'Enable','off');
        set(handles.text21,'Enable','off');
        set(handles.text22,'Enable','on');
        set(handles.text23,'Enable','off');
        pause(t_right_left);
```

```
        set(handles.text22,'Enable','off');
        putvalue(dio,[0 0 0 0]);
        pause(t_sleep);
        chek_Y(Y, dio,handles);
else
    disp('right 90°');
        putvalue(dio,[1 1 1 0]);
        set(handles.text19,'Enable','off');
        set(handles.text20,'Enable','off');
        set(handles.text21,'Enable','off');
        set(handles.text22,'Enable','on');
        set(handles.text23,'Enable','off');
        pause(t_right_right);
        set(handles.text22,'Enable','off');
        putvalue(dio,[0 0 0 0]);
        pause(t_sleep);
    disp('forward_X_time');
        putvalue(dio,[1 1 0 0]);
        set(handles.text19,'Enable','on');
        set(handles.text20,'Enable','off');
        set(handles.text21,'Enable','off');
        set(handles.text22,'Enable','off');
        set(handles.text23,'Enable','off');
        pause(abs(X*0.045));
        set(handles.text19,'Enable','off');
        putvalue(dio,[0 0 0 0]);
        pause(t_sleep);
    disp('left 90°');
        putvalue(dio,[1 1 0 1]);
        set(handles.text19,'Enable','off');
        set(handles.text20,'Enable','on');
        set(handles.text21,'Enable','off');
        set(handles.text22,'Enable','off');
        set(handles.text23,'Enable','off');
        pause(t_left_right);
        set(handles.text20,'Enable','off');
        putvalue(dio,[0 0 0 0]);
        pause(t_sleep);
        chek_Y(Y, dio, handles);
end
```

Comprueba la Distancia Respecto del Objetivo

```
%   dref = Distancia de Referencia del Objetivo
function chek_Y (Y, dio, handles)
dref = 100;
    if(Y == dref)
        disp('Origen')
        set(handles.text19,'Enable','off');
        set(handles.text20,'Enable','off');
        set(handles.text21,'Enable','off');
        set(handles.text22,'Enable','off');
        set(handles.text23,'Enable','on');
        putvalue(dio,[0 0 0 0]);
    elseif (Y > dref)
        disp('Adelante');
            putvalue(dio,[1 1 0 0]);
            set(handles.text19,'Enable','on');
            set(handles.text20,'Enable','off');
            set(handles.text21,'Enable','off');
            set(handles.text22,'Enable','off');
            set(handles.text23,'Enable','off');
            pause((Y-100)*0.045);
            set(handles.text19,'Enable','off');
            putvalue(dio,[0 0 0 0]);
    else
        disp('Atras');
            putvalue(dio,[1 1 1 1]);
            set(handles.text19,'Enable','off');
            set(handles.text20,'Enable','off');
            set(handles.text21,'Enable','on');
            set(handles.text22,'Enable','off');
            set(handles.text23,'Enable','off');
            pause((100-Y)*0.045);
            set(handles.text21,'Enable','off');
            putvalue(dio,[0 0 0 0]);
    end
end
```