CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE MECATRÓNICA

Implementación de un algoritmo de localización y mapeo simultáneo en un cuatrirotor

Tesis que presenta el
Ing. Cruz Sánchez Victor Hugo

Para obtener el grado de:
Maestro en Ciencias

En la especialidad de:
Ingeniería Eléctrica

Director de Tesis:
Dr. Hugo Rodríguez Cortés

Ciudad de México Febrero de 2017

Dedicatoria

Dedicado a
mi familia y a mis amigos de la Maestría

Agradecimientos

Especiales agradecimientos a mi familia, a mi asesor el Doctor Hugo Rodríguez y a mi amigo Francisco Albarrán.

Resumen

El problema de localización de vehículos no tripulados normalmente ha sido resuelto mediante sensores del tipo de posicionamiento satelital, sensores inerciales o cámaras estacionarias. Una alternativa que puede ser ventajosa en algunas condiciones es la implementación de cámaras a bordo para realizar una visión por computadora. En este trabajo se aborda el problema de la localización para un cuatrirotor. La solución a este importante problema se plantea mediante localización y mapeo simultaneo (SLAM). La técnica puede abordarse desde un enfoque de visión por computadora. Sin embargo, el SLAM sólo plantea el problema de obtener posiciones por lo que se puede incluir algún tipo de controlador al cuatrirotor con el objetivo de brindarle autonomía. Para este trabajo se plantea como etapa de control un enfoque del tipo predictivo. Se presentan simulaciones con el controlador del tiempo predictivo, implementación experimental de un sistema SLAM y finalmente la conjunción del SLAM junto con el controlador a un cuatrirotor.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Este capítulo describe los problemas de localización de un robot en un ambiente no estructurado y de construcción de un mapa de tal ambiente. Además, se presenta una introducción a la técnica de Control Predictivo Basado en Modelos (MPC), específicamente al Control Predictivo Generalizado (GPC).

1.1. El problema de la localización y la construcción de mapas

1.1.1. Localización y construcción de mapas simultáneos

Cuando un robot se mueve en un entorno sobre el cual tiene poco o ningún conocimiento previo, no basta con estimar el movimiento de la cámara con respecto a una imagen (*egomotion*) por medio de navegación por estima (*dead-reckoning*). En mediciones de movimiento relativo existen errores que se incrementan con el tiempo y por lo tanto la localización del robot es errónea. Es entonces necesario que el robot utilice sensores para identificar puntos de referencia y calcular su posición relativa a estos puntos desde ubicaciones futuras sobre su trayectoria [3].

Cualquier elemento, cuya ubicación relativa al robot sea medible repetidamente puede considerarse como una "característica" que puede ubicarse en un mapa. Normalmente se utilizan sensores del tipo sonar, de visión o láser para identificar características geométricas tales como puntos, líneas y planos de una escena. Con fusión de sensores se pueden obtener diversos tipos de características para la construcción de mapas, todas las características contribuyen a la localización [3].

Lo más importante a considerar al formular un algoritmo de construcción de mapa es que todas las mediciones son inciertas. Los mapas deben reflejar este hecho para que sean útiles. Existen dos enfoques para la construcción de mapas, dependiendo de la aplicación. Uno consiste en construir un mapa basado en los datos adquiridos durante una visita

guiada preliminar a un entorno, procesando todas las mediciones obtenidas fuera de línea para producir un mapa para uso futuro. En algunas aplicaciones a menudo no es posible mapear todas las áreas que el robot recorrerá de forma previa, por lo que se debe considerar el problema más desafiante de la localización secuencial y la construcción de mapas, esto es el segundo enfoque. El desafío en la construcción de mapas secuenciales es que a medida que cada nuevo conjunto de datos llega, debe ser posible incorporar la nueva información en el mapa en un tiempo de procesamiento limitado, ya que el robot sensa información y el próximo conjunto de datos pronto llegará. Esto a su vez requiere que la representación de todos los conocimientos obtenidos hasta el tiempo actual deben estar representados por una cantidad limitada de datos. Esta cantidad no puede crecer con el tiempo [3].

La construcción de mapas secuenciales es, por lo tanto, el proceso de propagación a través del tiempo de una estimación probabilística del estado actual de un mapa y la ubicación del robot en relación con él [3]. A continuación, se describen propiedades generales del proceso de construcción de mapas.

1.1.2. El proceso de construcción de mapas

Un mapa que es hecho por un robot que no tiene medidas externas de *egomotion* está fundamentalmente limitado en exactitud. El problema es causado por los errores compuestos de mediciones sucesivas. Considere, por ejemplo, a un ser humano que tiene la tarea de dibujar una línea recta muy larga equipado sólo con una regla de 30 cm, sin usar referencias externas. Los primeros metros serían fáciles, ya que sería posible mirar atrás, al inicio de la línea y alinear la regla para dibujar una nueva sección. A una distancia grande del punto de partida, la incertidumbre acumulativa será importante, y será imposible saber si las partes de la línea que se están dibujando son paralelas a la dirección original. Cambiar el proceso de medición podría mejorar las cosas. Si, por ejemplo, se pudieran colocar banderas a intervalos regulares a lo largo del camino, entonces la alineación correcta podría lograrse a distancias más largas. Eventualmente, las banderas originales desaparecerían de la vista y los errores se acumularían, a un ritmo más lento que antes. Algo similar ocurre en un sistema de construcción de mapas robótico. En un momento determinado se pueden realizar mediciones de únicamente el conjunto de características que son visibles desde la posición actual. Se puede confiar en la posición del robot en relación con las características que se pueden ver actualmente, pero de manera decreciente, tanto como se consideren las características que se han medido en el pasado más lejano. Un algoritmo de construcción de mapas correctamente formulado debe reflejar esto para que los mapas generados sean consistentes y útiles para períodos prolongados de navegación [3].

1.1.3. Acoplamiento de estimaciones del mapa

La construcción de mapas autónomos es un proceso que se debe realizar cuidadosamente, ya que los procesos de construcción del mapa y el cálculo de la ubicación con respecto a

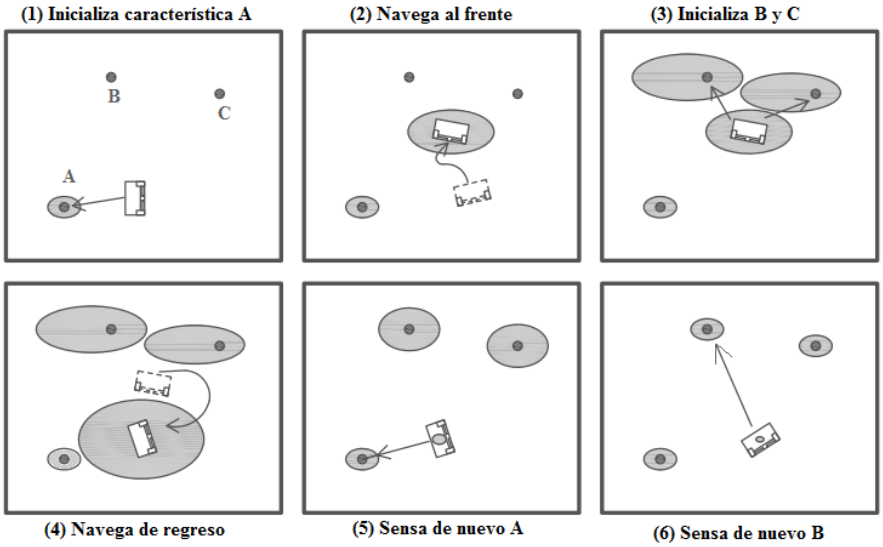

Figura 1.1: Ejemplo de construcción de mapas secuenciales, donde un robot moviéndose en dos dimensiones tiene un sensor preciso que le permite detectar la ubicación relativa de las características del punto (A, B o C) y menos precisa, odometría para estimación de movimiento por *dead-reckoning*. Los puntos negros son las localizaciones reales de características del entorno y las áreas grises representan estimaciones aproximadas de dichas características y de las posiciones del robot.

este, están inherentemente acoplados. Muchos enfoques iniciales en la construcción de mapas en línea tomaron enfoques simples para representar el estado y su incertidumbre [4], [5]. Las ubicaciones del robot en movimiento en el mundo y las características se almacenaron y actualizaron de forma independiente, utilizando múltiples filtros Kalman. Sin embargo, para movimientos a largo plazo, estos métodos resultan ser deficientes. Aunque producen buenas estimaciones del movimiento instantáneo, no toman en cuenta la interdependencia entre puntos del mapa y la localización, los mapas tienen deriva de manera sistemática. Por lo tanto, estos mapas no son capaces de producir estimaciones para largos recorridos, donde las características previamente vistas pueden revisarse después de períodos de abandono, una acción que permite corregir estimaciones [3].

Para dar una idea de la interdependencia de las estimaciones en la construcción de mapas secuenciales y la posición del robot, los pasos de un escenario simple se representan en la Figura 1.1. El punto es que un algoritmo de construcción de mapas debe ser capaz de hacer frente a movimientos arbitrarios y hacer uso de toda la información que obtiene.

En la Figura 1.1(1), un robot opera en un ambiente del cual no tiene conocimiento

previo. Definiendo el origen de coordenadas en esta posición inicial, utiliza un sensor para identificar la característica A, y medir su posición respecto a esta. El sensor es preciso, pero hay una cierta incertidumbre en esta medida que está representada por el área gris.

El robot avanza, en la Figura 1.1(2), haciendo una estimación de su movimiento usando dead reckoning (por ejemplo, contando las vueltas de sus ruedas). Este tipo de estimación de movimiento es impreciso y causa incertidumbres de movimiento que crecen sin límite en el tiempo. Esto se refleja en el tamaño de la región de incertidumbre alrededor del robot.

En la Figura 1.1(3), el robot hace una medición inicial de las características B y C. Dado que la estimación de la posición del robot es incierta, sus estimaciones de las ubicaciones de B y C tienen regiones grandes de incertidumbre, equivalentes a la incertidumbre de la posición del robot más la menor incertidumbre de medición del sensor. Sin embargo, aunque no puede representarse en el diagrama, las estimaciones de las ubicaciones del robot, B y C están acopladas en este punto. Estas posiciones relativas son bastante conocidas. Lo que es incierto es la posición del grupo en su conjunto.

En la Figura 1.1(4) el robot gira y vuelve a acercarse a su posición inicial. Durante éste movimiento la estimación de su posición, actualizada con *dead reckoning,* tiene mayor incertidumbre. En la Figura 1.1(5), sin embargo, la remedición de la característica A, cuya ubicación absoluta es bien conocida, permite al robot mejorar drásticamente la estimación de su posición. Lo importante es notar que esta medida también mejora la estimación de las ubicaciones de las características B y C. Ya que a pesar de que el robot había avanzado con respecto a la primera medición, las estimaciones de la posición de las característica todavía estaban parcialmente acopladas al estado del robot. La estimación del robot también mejora las estimaciones de características. Las estimaciones de características se mejoran adicionalmente, en la Figura 1.1(6), donde el robot remide directamente la característica B. Esta medida también mejora C debido a su interdependencia (las ubicaciones relativas de B y C son bien conocidas, dado que se midieron desde el mismo sitio).

En esta etapa, todas las estimaciones son bastante buenas y el robot ha construido un mapa útil. Es importante entender que esto ha ocurrido con un número bastante pequeño de mediciones porque se ha hecho uso del acoplamiento entre estimaciones [3].

1.2. Control predictivo basado en modelos (MBPC o MPC)

MPC designa un amplio conjunto de métodos de control los cuales hacen uso del modelo del proceso para obtener la señal de control a partir de la minimización de una función objetivo. Las ideas principales en las familias de controladores predictivos son las siguientes. Uso explicito de un modelo para predecir las salidas del proceso en instantes de tiempo futuro (horizonte), el calculo de una secuencia de control con la minimización de una función objetivo y una estrategia de retroceso. Esto es, en cada instante, el horizonte

se desplaza hacia el futuro, lo cual implica la aplicación de la primera señal de control de la secuencia calculada en cada paso [6].

Algorítmos de MPC difieren en el modelo usado para representar el proceso, el modelo del ruido y las funciones de costo a ser minimizadas. Existen muchas aplicaciones del control predictivo exitosas en el presente. En procesos que implican a robots manipuladores, por ejemplo, en la anestesia clínica [7], en la industria del cemento, torres de secado y brazos robóticos [8]. También existen aplicaciones en columnas de destilación, plantas de policloruro de vinilo (PVC) y generadores de vapor [9]. En [10] se presenta un enfoque de control en tiempo real para un cuatrirotor donde se combina retroalimentacion completa de estados y control predictivo basado en modelos (MPC). En [11] se presenta una estrategia MPC para la estabilizacion y seguimiento de trayectorias de un cuatrirotor con empuje restringido para considerar condiciones de operación reales. Los autores utilizan linealización aproximada alrededor de puntos de operación.

MPC presenta una serie de ventajas sobre otros métodos, entre las cuales destacan las siguientes. Puede utilizarse para controlar una gran variedad de procesos, desde aquellos con dinámica relativamente simple hasta aquellos complejos, incluyendo sistemas con retardo grande o de fase no mínima. Introduce control hacia adelante de forma natural para compensar perturbaciones. El controlador resultante es una ley de control fácil de implementar. Es muy útil cuando se conocen referencias futuras. Es una metodología basada en principios básicos los cuales permiten extensiones futuras [6].

Lógicamente, también existen inconvenientes. Uno de estos es que la ley de control es más difícil de derivar que los controladores *PID* clásicos. Cuando se consideran restricciones, el gasto computacional es alto. El mayor inconveniente es la necesidad de que un modelo apropiado del proceso este disponible [6].

En la práctica, MPC ha probado ser una estrategia razonable para el control en la industria. A pesar de la falta de resultados teóricos en algunos puntos cruciales como la estabilidad o robustez [6].

El método GPC, propuesto por Clarke en [12] es uno de los más populares métodos MPC tanto en la industria como en la investigación. Ha sido implementado en aplicaciones industriales mostrando buen rendimiento [8]. Este puede lidiar con gran variedad de problemas de control en plantas con un razonable número de variables de diseño [6].

La idea básica del GPC es calcular una secuencia de señales de control futuras de forma tal que se minimice una función de costo definida sobre un horizonte de predicción. El índice a ser optimizado es la evolución de una función cuadrática que mide la distancia entre la salida predicha del sistema y alguna secuencia de referencia predicha sobre el horizonte mas una función cuadrática que mide el esfuerzo de control [6].

El Control Predictivo Generalizado provee una solución analítica e incorpora el concepto de horizonte de control además de considerar la ponderación de los incrementos de control en la función de costo [6].

1.3. Objetivos

1.3.1. Objetivo General

Regular la posición, respecto a un eje Cartesiano, de un cuatrirotor utilizando un esquema de localización y mapeo simultáneo visual y un control predictivo generalizado.

1.3.2. Objetivos Específicos

- Implementación del SLAM visual con una cámara conectada a una computadora.
- Envío de datos de posición obtenidos por el SLAM visual mediante comunicación serial inalámbrica.
- Implementación de un controlador predictivo generalizado en el vehículo.
- Cerrar el lazo del controlador predictivo generalizado utilizando la información del SLAM visual

1.4. Organización de la Tesis

En el Capítulo dos se muestra el modelo del cuatrirotor utilizado, se presenta la explicación del desarrollo de un controlador del tipo predictivo y se describe la implementación del controlador GPC. En el Capítulo tres se aborda la adquisición de imágenes y su procesamiento así como la forma en que se puede obtener una posición a partir de estas. Además se describe el sistema de SLAM visual que se implementó. Finalmente en el último Capítulo se muestra el hardware utilizado y resultados en simulación, resultados experimentales, conclusiones y trabajo a futuro propuesto.

Capítulo 2

Modelado y Control

En este capítulo se describe el modelo del cuatrirotor que se considera para diseñar el controlador predictivo y la estrategia para implementar este controlador.

2.1. Modelo del cuatrirotor

Para describir el modelo dinámico del cuatrirotor es necesario definir dos marcos de referencia (Figura 2.1), el marco de referencia del cuerpo $0x_by_bz_b$ y el marco de referencia inercial $0x_ey_ez_e$.

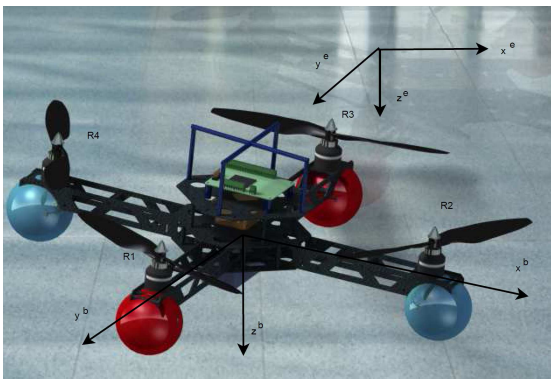

Figura 2.1: Marcos de referencia.

La dinámica rotacional del cuatrirotor está dada por [13]

$$J\dot{\Omega} + \Omega \times J\Omega = M_e \tag{2.1}$$

donde $\Omega = \begin{bmatrix} p & q & r \end{bmatrix}^T$ es el vector de velocidad angular alrededor de los ejes x^b, y^b y z^b.

J es la matriz de inercia, M_e es el vector momentos de entrada, esto es,

$$M_e = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} (T_2 - T_4)\, l \\ (T_1 - T_3)\, l \\ -Q_1 + Q_2 - Q_3 + Q_4 \end{bmatrix} \tag{2.2}$$

donde l es la distancia del centro de masa del cuatrirotor a los ejes de los rotores, Q_i son los momentos reactivos y T_i los empujes producidos por cada motor definidos como

$$T_i = C_{T_i} \pi \bar{r}_i^4 \rho \omega_i^2 \tag{2.3}$$

$$Q_i = C_{Q_i} \pi \bar{r}_i^5 \rho \omega_i^2 \tag{2.4}$$

con C_{T_i} y C_{Q_i} los coeficientes aerodinámicos propios de cada hélice, $\bar{r}_i$ el radio de cada hélice, ρ la densidad del aire y ω_i son las velocidades de cada uno de los motores.

La dinámica traslacional en un marco de referencia inercial es [13]

$$\ddot{X} = g e_3 - \frac{1}{m_a} T_T R\left(\Phi\right) e_3 \tag{2.5}$$

con $X = \begin{bmatrix} x & y & z \end{bmatrix}^T$ el vector de posición traslacional, $e_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$, m_a es la masa del cuatrirotor, g es la constante de gravedad, T_T es el empuje total de los motores del cuatrirotor dado por

$$T_T = \sum_{i=1}^{4} T_i \tag{2.6}$$

La matriz de rotación $R\left(\Phi\right)$, en la secuencia de rotación Z-Y-X, que describe la orientación el términos de los ángulos de Euler alabeo ϕ, cabeceo θ y guiñada ψ es

$$R\left(\Phi\right) = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ c_\theta s_\psi & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{2.7}$$

La relación entre los empujes T_i producidos por cada motor con el empuje total T_T y el vector de momentos M_e tiene la forma

$$\begin{bmatrix} T_T \\ L \\ M \\ N \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 0 & l & 0 & -l \\ l & 0 & -l & 0 \\ -\bar{r}_i \frac{C_Q}{C_T} & \bar{r}_i \frac{C_Q}{C_T} & -\bar{r}_i \frac{C_Q}{C_T} & \bar{r}_i \frac{C_Q}{C_T} \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \tag{2.8}$$

Por lo tanto los empujes T_i que deben generar cada uno de los motores para aplicar las señales de control T_T y M_e se obtienen por medio de la expresión

$$\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} = \begin{bmatrix} -\frac{1}{4} & 0 & \frac{1}{2l} & -\frac{C_T}{4 C_Q \bar{r}_i} \\ -\frac{1}{4} & \frac{1}{2l} & 0 & \frac{C_T}{4 C_Q \bar{r}_i} \\ -\frac{1}{4} & 0 & -\frac{1}{2l} & -\frac{C_T}{4 C_Q \bar{r}_i} \\ -\frac{1}{4} & -\frac{1}{2l} & 0 & \frac{C_T}{4 C_Q \bar{r}_i} \end{bmatrix} \begin{bmatrix} T_T \\ L \\ M \\ N \end{bmatrix}. \tag{2.9}$$

2.2. Control predictivo generalizado (GPC)

El enfoque de GPC utiliza un modelo CARIMA (Controlador Auto Regresivo de Promedio Móvil Integrado) con n salidas y m entradas [14] de la forma

$$\mathbf{A}\left(z^{-1}\right) y\left(t\right) = \mathbf{B}\left(z^{-1}\right) u\left(t-1\right) + \frac{1}{\Delta}\mathbf{C}\left(z^{-1}\right) e\left(t\right) \tag{2.10}$$

donde $\mathbf{A}\left(z^{-1}\right)$ y $\mathbf{C}\left(z^{-1}\right)$ son matrices polinomiales mónicas de orden $n \times n$ y $\mathbf{B}\left(z^{-1}\right)$ es una matriz polinomial de orden $n \times m$. El operador Δ queda definido como $\Delta = 1 - z^{-1}$. Las variables $y\left(t\right)$, $u\left(t\right)$ y $e\left(t\right)$ son vectores de salida, de control y de ruido, con dimensión $n \times 1$, $m \times 1$ y $n \times 1$, respectivamente. Es común considerar por simplicidad $e\left(t\right)$ como ruido blanco.

El objetivo del GPC es encontrar la secuencia de control que minimice el siguiente criterio con horizonte finito

$$\begin{aligned} J_c\left(N_p, N_u\right) = &\sum_{j=1}^{N_p} \left\| \hat{y}\left(t+j|t\right) - w\left(t+j\right) \right\|_R^2 \\ &+ \sum_{j=1}^{N_u} \left\| \Delta u\left(t+j-1\right) \right\|_Q^2 \end{aligned} \tag{2.11}$$

donde $\hat{y}\left(t+j|t\right)$ es una predicción óptima j pasos adelante de la salida del sistema calculada en el tiempo t. Es posible calcular a $\hat{y}$ si se conocen los vectores de la entrada y salida pasados y la secuencia de referencia futura. N_p y N_u son los horizontes de predicción y de control, respectivamente. $w\left(t+j\right)$ es el vector de la secuencia de referencia para el vector de salida. R y Q son matrices de ponderación positivas definidas.

Es común modelar $\mathbf{C}\left(z^{-1}\right)$ como la matriz identidad. La predicción para la salida se obtiene como

$$\begin{aligned} y\left(t+j\right) = &\mathbf{F}_j\left(z^{-1}\right) y\left(t\right) + \mathbf{E}_j\left(z^{-1}\right) e\left(t+j\right) \\ &+ \mathbf{E}_j\left(z^{-1}\right) \mathbf{B}\left(z^{-1}\right) \Delta u\left(t+j-1\right) \end{aligned} \tag{2.12}$$

donde $\mathbf{E}_j\left(z^{-1}\right)$ y $\mathbf{F}_j\left(z^{-1}\right)$ se obtienen al resolver en forma recursiva la ecuación Diofantina matricial

$$I_{n\times n} = \mathbf{E}_j\left(z^{-1}\right) \tilde{\mathbf{A}}\left(z^{-1}\right) + z^{-j}\mathbf{F}_j\left(z^{-1}\right) \tag{2.13}$$

donde $\tilde{A}(z^{-1}) = \Delta A(z^{-1})$. Haciendo $\mathbf{E}_j\left(z^{-1}\right)\mathbf{B}\left(z^{-1}\right) = \mathbf{G}_j\left(z^{-1}\right) + z^{-j}\mathbf{G}_{jp}\left(z^{-1}\right)$ con $grado\left(\mathbf{G}_j\left(z^{-1}\right)\right) < j$, la ecuación de predicción puede escribirse en la forma siguiente

$$\begin{aligned} \hat{y}\left(t+j|t\right) = &\mathbf{G}_j\left(z^{-1}\right) \Delta u\left(t+j-1\right) \\ &+ \mathbf{G}_{jp}\left(z^{-1}\right) \Delta u\left(t-1\right) + \mathbf{F}_j\left(z^{-1}\right) y\left(t\right) \end{aligned} \tag{2.14}$$

La ecuación (2.14) puede expresarse en forma compacta como

$$\mathbf{y} = \mathbf{G}\mathbf{u} + \mathbf{f} \tag{2.15}$$

con $\mathbf{f}_j = \mathbf{G}_{jp}\left(z^{-1}\right)\Delta u\left(t-1\right) + \mathbf{F}_j\left(z^{-1}\right)y\left(t\right)$ donde no se consideran los términos que aparecen con adelanto en el tiempo.

La ecuación (2.11) se puede reescribir como

$$J_c = \left(\mathbf{G}\mathbf{u} + \mathbf{f} - w\right)^T \bar{R}\left(\mathbf{G}\mathbf{u} + \mathbf{f} - w\right) + \mathbf{u}^T\bar{Q}\mathbf{u} \tag{2.16}$$

donde $\bar{R} = diag\left(R, \cdots, R\right)$ y $\bar{Q} = diag\left(Q, \cdots Q\right)$. Si no hay restricciones, el control óptimo puede expresarse como [6]

$$\Delta\mathbf{u}\left(t\right) = K\left(\mathbf{w} - \mathbf{f}\right) \tag{2.17}$$

donde $K = \left(\mathbf{G}^T\bar{R}\,\mathbf{G} + \bar{Q}\right)^{-1}\mathbf{G}^T\bar{R}$. Debido a que la estrategia GPC utiliza el enfoque de horizonte móvil, solamente es necesaria la componente $\Delta\mathbf{u}\left(t\right)$ en el instante t y se descartan las demás componentes de $\Delta\mathbf{u}\left(t\right)$

2.3. Estrategia de control

En [13] se implementa una estrategia de control en donde se controla la dinámica rotacional y traslacional para un cuatrirotor por separado. La rotación es controlada por un control de orientación en espacio SO(3) y la traslación por el controlador TECS. En este trabajo se implementó el GPC en lugar del TECS. El diagráma de la Figura 2.2 muestra la estructura de los controladores.

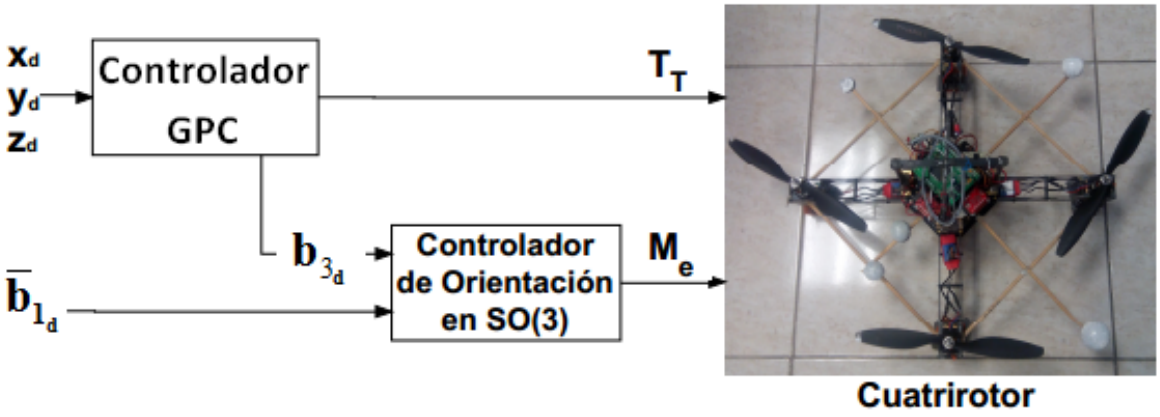

Figura 2.2: Diagrama de control.

Los vectores b_{i_d} con $i = 1, 2, 3$, hacen referencia a la matriz de rotación deseada, la cual se especifica en la siguiente subsección.

2.3.1. Implementación del GPC

La ecuación (2.5) puede expresarse como

$$m_a\ddot{X} = m_a g e_3 - u - \Gamma \tag{2.18}$$

donde u es una señal de control virtual

$$\Gamma = \frac{T_T}{e_3^T R_d^T R e_3}\left(\left(e_3^T R_d^T R e_3\right) b_3 - b_{3_d}\right) \tag{2.19}$$

Si se considera que la dinámica rotacional es mucho más rápida que la dinámica traslacional entonces el término $\Gamma \to 0$ y el sistema puede 2.18 expresarse como

$$m_a\ddot{X} = m_a g e_3 - u \tag{2.20}$$

Entonces el diseño del control predictivo para la dinámica traslacional se diseña con la estructura lineal (2.20)

$$\begin{array}{ccc} \dot{x} = v_x & \dot{y} = v_y & \dot{z} = v_z \\ \dot{v}_x = -\frac{1}{m_a}u_x & \dot{v}_x = -\frac{1}{m_a}u_y & \dot{v}_z = g - \frac{1}{m_a}u_z \end{array} \tag{2.21}$$

de donde se obtienen las funciones de transferencia

$$\frac{X(s)}{U_x(s)} = \frac{Y(s)}{U_y(s)} = \frac{Z(s)}{U_z(s)} = -\frac{1}{m_a}\frac{1}{s^2} \tag{2.22}$$

La fuerza de gravedad g se considera como una perturbación desconocida que será compensada por el controlador.

Para el diseño del control predictivo se han tomado los parámetros $m_a = 1.12$ y $g = 9.81$ de donde resultan las funciones de transferencia

$$\begin{aligned} \frac{X(z^{-1})}{U_x(z^{-1})} &= \frac{Y(z^{-1})}{U_y(z^{-1})} = \frac{Z(z^{-1})}{U_z(z^{-1})} \\ &= \frac{-0.00003098z^{-1} - 0.00003098z^{-2}}{1 - 2z^{-1} + z^{-2}} \end{aligned} \tag{2.23}$$

Se obtienen ahora las matrices polinomiales

$$\mathbf{A}\left(z^{-1}\right) = diag\left[1 - 2z^{-1} + z^{-2}\right] \tag{2.24}$$

$$\mathbf{B}\left(z^{-1}\right) = diag\left[-0.00003098z^{-1} - 0.00003098z^{-2}\right] \tag{2.25}$$

donde $\mathbf{A}(z^{-1})$ y $\mathbf{B}(z^{-1})$ son matrices polinomiales de orden 3×3. Con lo que se tienen los elementos necesarios del modelo CARIMA.

En [15] se propone un esquema de control para la rotación del cuatrirotor utilizando las características del espacio $SO(3)$. Se definen los errores de ángulo e_R y velocidad angular e_Ω como

$$e_R = \frac{1}{2}\left(R_d^T R - R^T R_d\right)^\vee \tag{2.26}$$

$$e_\Omega = \Omega - R^T R_d \Omega_d \tag{2.27}$$

donde R_d y Ω_d son los valores deseados para la matriz de rotación y el vector de velocidad angular, respectivamente. El mapeo $^\vee : \mathbf{so}(3) \to \mathbb{R}^3$ es el mapeo inverso del mapa $\hat{\cdot} : \mathbb{R}^3 \to \mathbf{so}(3)$ definido como $\hat{x}\ y = x \times y$. Entonces se propone el control como

$$M_e = -k_R e_R - k_\Omega e_\Omega + \Omega \times J\Omega - J\left(\hat{\Omega} R^T R_d \Omega_d - R^T R_d \dot{\Omega}_d\right)$$

con k_R, $k_\Omega > 0$ La matriz de rotación deseada se obtiene como $R_d = [b_{2_d} \times b_{3_d}, b_{2_d}, b_{3_d}] \in SO(3)$. Donde

$$b_{2_d}(t) = \frac{b_{3_d} \times b_{1_d}}{\|b_{3_d} \times b_{1_d}\|} \tag{2.28}$$

b_{3_d} puede obtenerse a partir del control de la dinámica traslacional como

$$b_{3_d} = \frac{u}{\|u\|} \tag{2.29}$$

donde $u = \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}$.

Se tiene entonces el controlador listo para implementarse al cuatrirotor. Resta obtener las posiciones del cuatrirotor que alimentaran al controlador. El siguiente capitulo trata este problema.

Ejemplo numérico

Debido a la naturaleza numérica del método GPC, no es posible escribir una expresión analítica para la secuencia de control u. Por lo tanto, se presenta un ejemplo de cálculo con un horizonte de predicción de $N_p = 3$, con horizonte de control $N_u = 2$, peso para el error de $R = diag(0.1, 0.1, 0.1)$ y para el control de $Q = diag(1, 1, 1)$, donde se obtiene una expresión analítica.

Al resolver de manera iterativa la ecuación (2.13) se obtienen las matrices E_j y F_i de

orden 3×3 con $j = 1, ..., 3$ dadas por

$$\begin{aligned} E_1 &= diag\,(1) \\ E_2 &= diag\left(1 + 3z^{-1}\right) \\ E_3 &= diag\left(1 + 3z^{-1} + 6z^{-2}\right) \\ F_1 &= diag\left(1 - 3z^{-1} + 3z^{-2}\right) \\ F_2 &= diag\left(3 - 8z^{-1} + 6z^{-2}\right) \\ F_3 &= diag\left(6 - 15z^{-1} + 10z^{-2}\right) \end{aligned}$$

Al utilizar E_j y F_j en la ecuación (2.12) se llega a la forma de la ecuación (2.14) de donde se observa que

$$\mathbf{G} = \begin{bmatrix} G_1 & 0 \\ G_2 & G_1 \\ G_3 & G_2 \end{bmatrix} \quad \mathbf{G}_{jp} = \begin{bmatrix} G_{1p} \\ G_{2p} \\ G_{3p} \end{bmatrix}$$

con

$$\begin{aligned} G_1 &= diag\left(-0.0289 \times 10^{-3}\right) \\ G_2 &= diag\left(-0.1156 \times 10^{-3}\right) \\ G_3 &= diag\left(-0.2602 \times 10^{-3}\right) \\ G_{1p} &= diag\left(-0.0289 \times 10^{-3}\right) \\ G_{2p} &= diag\left(-0.0867 \times 10^{-3}\right) \\ G_{3p} &= diag\left(-0.1735 \times 10^{-3}\right) \end{aligned}$$

$G_{1...N_p}$ y $G_{1p...N_pp}$ son matrices cuadradas de orden 3×3.

Finalmente al utilizar la expresión (2.17) se obtiene la secuencia de control

$$\mathbf{u} = \begin{bmatrix} \Delta u\,(t) & \Delta u\,(t+1) & \Delta u\,(t+2) \end{bmatrix}$$

de donde solo se utiliza la primera componente.

Capítulo 3

Visión por computadora

En este capítulo se describen conceptos y herramientas de visión por computadora con la finalidad de implementarlos en un sistema de SLAM visual.

3.1. Formación de una imagen

En está sección se analiza la forma en que las imágenes se forman y capturan, el primer paso en la percepción humana y robótica del mundo. A partir de las imágenes podemos deducir el tamaño, la forma y la posición de los objetos en el mundo, así como otras características tales como el color y la textura. Todos los vertebrados tienen lentes que forman una imagen invertida en la retina cuyos conos y bastones son sensibles a la luz. Una cámara digital es similar en principio; un lente de vidrio o plástico forma una imagen en la superficie de un chip semiconductor con un arreglo de dispositivos sensibles a la luz para convertir la luz en una imagen digital [1].

El proceso de formación de una imagen en un ojo o en una cámara involucra una proyección del mundo tridimensional sobre una superficie bidimensional. La información de profundidad se pierde y no se puede concluir si se trata de un objeto grande a la distancia o uno pequeño cercano. Esta transformación de tres a dos dimensiones es conocida como proyección en perspectiva [1].

3.1.1. Transformación de la perspectiva

Los aspectos elementales de la formación de una imagen con un lente delgado se muestran en la Figura 3.1. El eje z positivo es el eje óptico de la cámara. La coordenada z del objeto y su imagen están relacionadas por la ley del lente

$$\frac{1}{z_0} + \frac{1}{z_i} = \frac{1}{f} \tag{3.1}$$

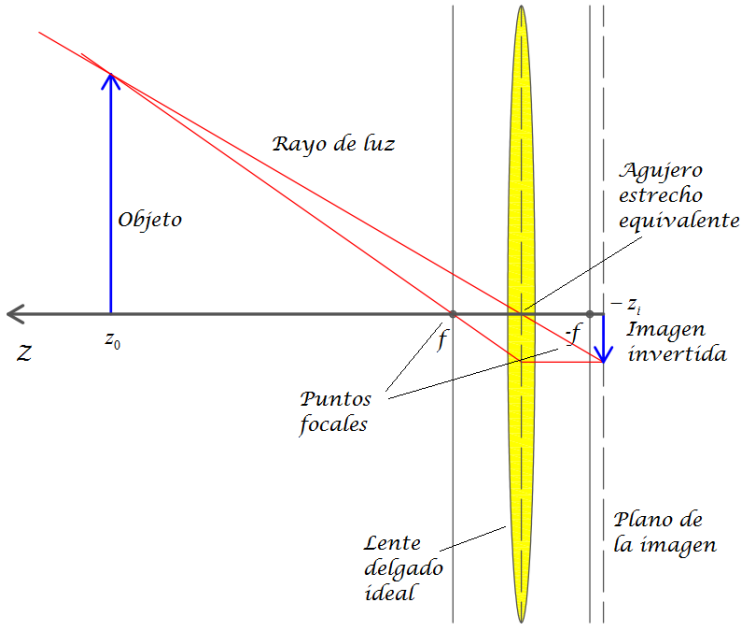

Figura 3.1: Geometría de la formación de una imagen para un lente delgado convexo mostrado en una sección transversal bidimensional. Un lente tiene dos puntos focales a una distancia f a cada lado del lente. Por convención el eje óptico de la cámara es z.

donde z_0 es la distancia al objeto, z_i es la distancia a la imagen y f es la longitud focal de los lentes. Para $z_0 > f$ se forma una imagen invertida sobre el plano de la imagen en $z < -f$. En una cámara, el plano de la imagen se fija a la superficie del chip sensor, por lo que el anillo de enfoque de la cámara mueve al lente a lo largo del eje óptico siendo z_i la distancia al plano de la imagen. Para un objeto en el infinito $z_i = f$. Una cámara estenopeica no necesita centrarse. La necesidad de enfocar es la compensación por mayor capacidad de recolección de luz del lente [1].

En visión por computadora es común utilizar el modelo de imagen de perspectiva central mostrado en la Figura 3.2. Los rayos convergen en el origen del marco de la cámara C y se proyecta una imagen no invertida sobre el plano de la imagen localizado en $z = f$. Utilizando triángulos similares se puede demostrar que un punto en las coordenadas del mundo $P = (X, Y, Z)$ se proyecta al plano de la imagen $p = (x, y)$ por

$$x = f\frac{X}{Z}, y = f\frac{Y}{Z} \tag{3.2}$$

Se puede escribir el punto del plano de la imagen en una forma homogénea $\tilde{p} = (x', y', z')$ donde

$$x' = f\frac{X}{z'},\ y' = f\frac{Y}{z'},\ z' = z \tag{3.3}$$

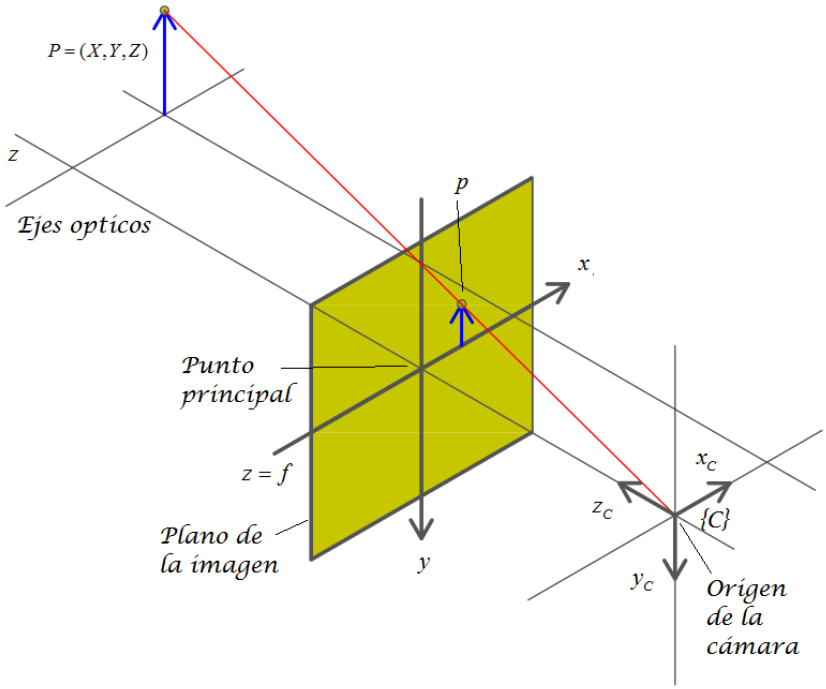

Figura 3.2: Modelo de proyección central. El plano de imagen es f estando enfrente del origen de la cámara y sobre el cual una imagen no invertida es formada.

en forma matricial

$$\tilde{p} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.4}$$

donde las coordenadas no homogéneas del plano de la imagen son

$$x = \frac{x'}{z'}, \; y = \frac{y'}{z'} \tag{3.5}$$

Si se escriben las coordenadas del mundo en forma homogénea como ${}^c\tilde{P} = (X, Y, Z, 1)^T$ entonces la proyección de perspectiva puede escribirse de forma lineal como

$$\tilde{p} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}^c\tilde{P} \tag{3.6}$$

ó

$$\tilde{p} = C\,{}^c\tilde{P} \tag{3.7}$$

donde C es una matriz de 3×4 conocida como matriz de la cámara. Se debe notar que se ha escrito ${}^c\tilde{P}$ para resaltar el hecho de que ésta es la coordenada del punto con respecto

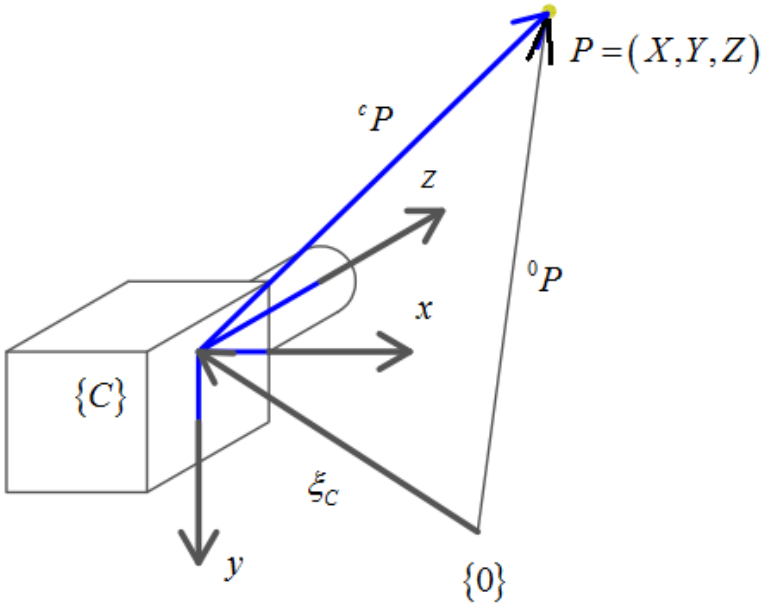

Figura 3.3: Marco de coordenada de la cámara.

al marco de la cámara C. Las tildes indican cantidades homogéneas. La tercera columna de C es un vector paralelo al eje óptico de la cámara en las coordenadas del mundo. La matriz de la cámara puede factorizarse como

$$\tilde{p} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}^{c}\tilde{P} \tag{3.8}$$

donde la segunda matriz, es la matriz de proyección.

En general, la cámara tiene una *pose* ξ_C con respecto al marco de coordenadas del mundo como se muestra en la Figura 3.3. La posición de un punto en coordenadas homogéneas es

$${}^{c}\tilde{P} = {T_C}^{-1}\, {}^{0}P \tag{3.9}$$

donde T_C es la matriz de transición homogénea [1].

En una cámara digital el plano de la imagen es una cuadricula de $W \times H$ elementos sensibles a la luz que corresponden directamente a elementos de una imagen o pixeles. Las coordenadas de un pixel son vectores de dos dimensiones (u, v). Por convención el origen está ubicado en la parte superior izquierda (figura 3.4). La coordenada de un pixel está relacionada a la coordenada del plano de la imagen por

$$u = \frac{x}{\rho_w} + u_0, v = \frac{y}{\rho_h} + v_0 \tag{3.10}$$

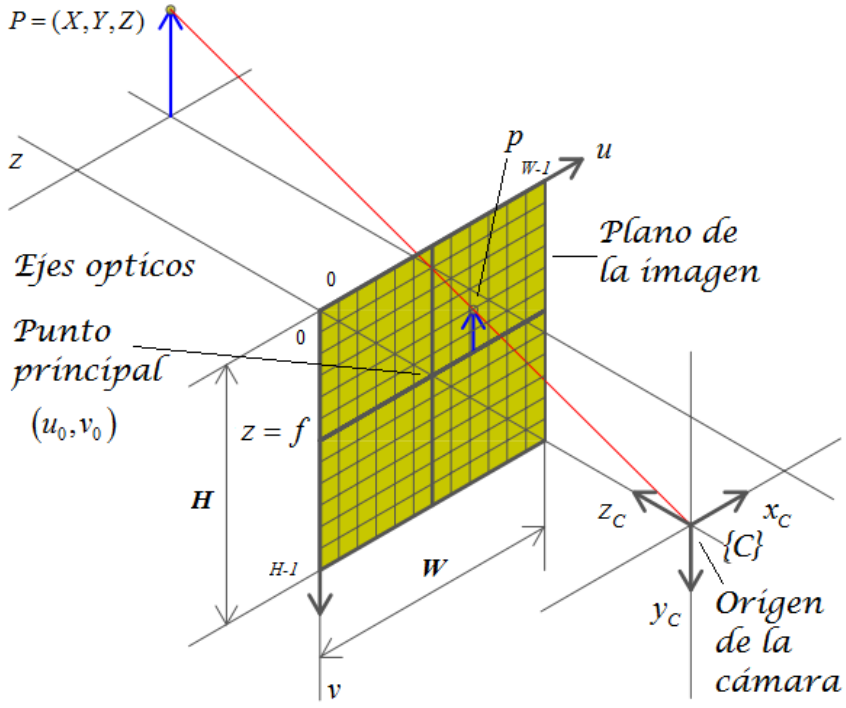

Figura 3.4: Modelo de proyección central mostrando el plano de la imagen y los pixeles discretos.

donde ρ_w y ρ_h son el ancho y la altura de cada pixel respectivamente y (u_0, v_0) es el punto principal (coordenada del punto donde el eje óptico intersecta el plano de la imagen). Se puede escribir la ecuación (3.6) en coordenadas de pixeles como

$$\tilde{p} = \begin{bmatrix} \frac{1}{\rho_w} & 0 & u_0 \\ 0 & \frac{1}{\rho_w} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 0 \end{bmatrix} {}^c\tilde{P} \tag{3.11}$$

donde $\tilde{p} = (u', v', w')$ es la coordenada homogénea del punto del mundo P en coordenadas de pixeles. Las coordenadas no homogéneas de los pixeles del plano de la imagen son

$$u = \frac{u'}{w'}, v = \frac{v'}{w'} \tag{3.12}$$

Combinando las ecuaciones (3.9) y (3.11) se obtiene

$$\begin{aligned}\tilde{p} = \begin{bmatrix} \frac{f}{\rho_w} & 0 & u_0 \\ 0 & \frac{f}{\rho_w} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} T_c^{-1} \tilde{P} \\ = K P_0 T_c^{-1} \tilde{P} \\ = C \tilde{P}\end{aligned} \tag{3.13}$$

donde todos los términos están dentro de la matriz de la cámara C. Esta es una transformación homogénea de 3×4 que realiza escalamiento, traslación y proyección de perspectiva. También se denomina matriz de proyección o matriz de calibración de la cámara [1].

Los parámetros intrínsecos son características innatas de la cámara y del sensor $(f, \rho_w, \rho_h, u_0, v_0)$. Los parámetros extrínsecos describen la *pose* de la cámara, seis para orientación y traslación en el espacio especial euclidiano en tres dimensiones SE(3). Por lo tanto, hay un total de 11 parámetros. La matriz de la cámara tiene 12 elementos, por lo que un grado de libertad (el factor de escala global) no está únicamente determinado [1].

3.1.2. Distorsión en lentes

Las imperfecciones de la lente producen una variedad de distorsiones, tales como aberración cromática (franja de color), aberración esférica o astigmatismo (variación en el enfoque a través de la escena) y distorsiones geométricas en las que los puntos del plano de la imagen están desplazados en valores acordes con la ecuación (3.4) [1].

La distorsión geométrica es generalmente el efecto más problemático que se encuentra en aplicaciones robóticas y comprende dos componentes, radial y tangencial. La distorsión radial hace que los puntos de la imagen sean trasladados a lo largo de las líneas radiales desde el punto principal. El error radial se aproxima correctamente por el polinomio

$$\delta r = k_1 r^3 + k_2 r^5 + k_3 r^7 + \cdots \tag{3.14}$$

donde r es la distancia entre el punto de la imagen y el punto principal. La distorsión de barril se produce cuando la ampliación disminuye con la distancia desde el punto principal, lo que hace que las líneas rectas cerca del borde de la imagen se curven hacia fuera. La distorsión de cojín ocurre cuando la ampliación aumenta con la distancia desde el punto principal y hace que las rectas cercanas al borde de la imagen se curven hacia dentro. La distorsión tangencial, o distorsión de descentramiento, ocurre en ángulos rectos a los radios pero es menos significativa que la distorsión radial. Las coordenadas del punto (u, v) después de la distorsión están dadas por

$$u^d = u + \delta_u, v^d = v + \delta_v \tag{3.15}$$

donde el desplazamiento es

$$\begin{bmatrix}\delta_u \\ \delta_v\end{bmatrix} = \begin{bmatrix}u(k_1r^2 + k_2r^4 + k_3r^6 + \dots) \\ v(k_1r^2 + k_2r^4 + k_3r^6 + \dots)\end{bmatrix} + \begin{bmatrix}2p_1uv + p_2(r^2 + 2u^2) \\ p_1(r^2 + 2v^2) + 2p_1uv\end{bmatrix} \tag{3.16}$$

Este vector de desplazamiento se puede trazar para diferentes valores de (u, v). Los vectores indican el desplazamiento necesario para corregir la distorsión en diferentes puntos de la imagen, es decir $(-\delta_u, -\delta_v)$, y muestran distorsión radial dominante. Típicamente son suficientes tres coeficientes para describir la distorsión radial, el modelo de distorsión está parametrizado por $(k_1, k_2, k_3, p_1, p_2)$ que se consideran como parámetros intrínsecos adicionales [1].

3.1.3. Calibración de la cámara

El modelo de proyección de cámara dado por la ecuación (3.13) tiene una serie de parámetros que en la práctica son desconocidos. En general, el punto principal no está en el centro de la matriz de pixeles. La longitud focal de una lente es precisa al 4 % y es correcto si la lente se centra en el infinito. También es común que los parámetros intrínsecos cambien si un lente se desprende y se vuelve a colocar o se ajusta por enfoque o apertura. Los únicos parámetros intrínsecos que se pueden obtener son las dimensiones de los foto sitios ρ_w y ρ_h de la hoja de datos del fabricante del sensor. Los parámetros extrínsecos, la *pose* de la cámara, plantea la cuestión de dónde está exactamente el punto central de la cámara [1].

La calibración de la cámara es el proceso de determinar los parámetros intrínsecos de la cámara y los parámetros extrínsecos con respecto al sistema de coordenadas del mundo. Las técnicas de calibración se basan en conjuntos de puntos con respecto al mundo cuyas coordenadas relativas son conocidas y cuyas correspondientes coordenadas plano-imagen también son conocidas. Técnicas de vanguardia como el *toolbox* de calibración de Bouguet para MATLAB® [16] requieren simplemente una serie de imágenes de un plano de tablero de ajedrez como se muestra en la Figura 3.5. A partir de esta imagen se pueden estimar los parámetros intrínsecos (incluyendo los parámetros de distorsión) así como la *pose* relativa del tablero de ajedrez en cada imagen. Las técnicas de calibración clásica requieren una vista única de los objetivos de calibración tridimensional, pero no pueden estimar el modelo de distorsión. Estas se explican a continuación [1].

3.1.4. Enfoque de transformación homogénea

El método de transformación homogénea permite la estimación directa de la matriz C de la cámara dada en la ecuación (3.13). Los elementos de esta matriz son funciones de los parámetros intrínsecos y extrínsecos. Ajustando $\tilde{p}=(u, v, 1)$, expandiendo la ecuación

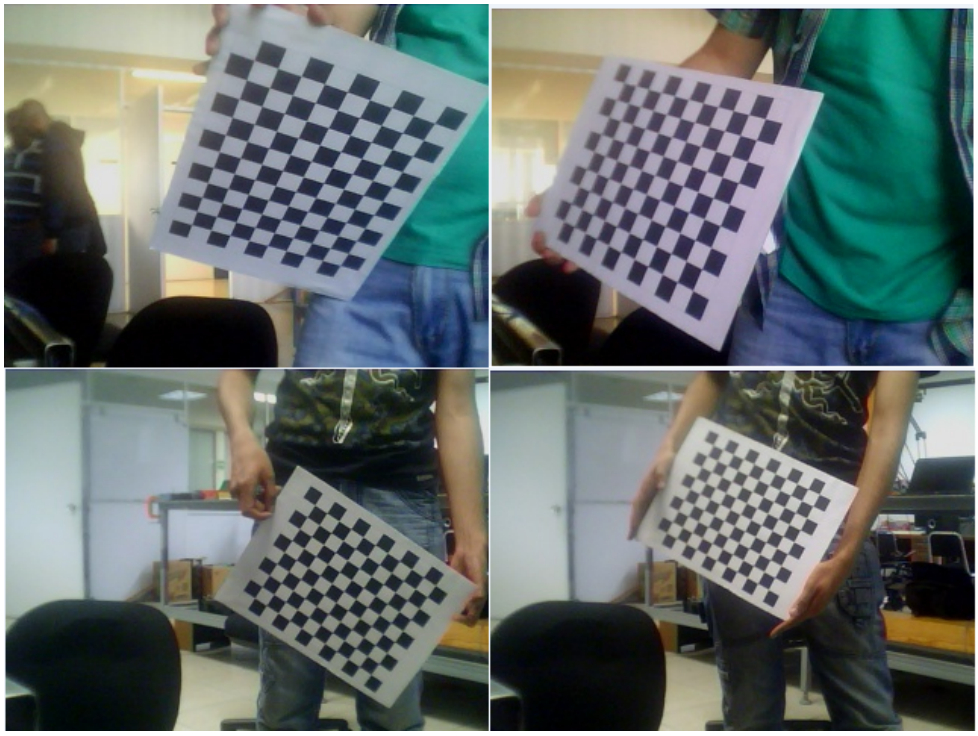

Figura 3.5: Ejemplos de cuadros para calibración con un tablero.

(3.13) y sustituyendo en la ecuación (3.12) se puede escribir

$$\tilde{p} = \begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} C_{1,1} & C_{1,2} & C_{1,3} & C_{1,4} \\ C_{2,1} & C_{2,2} & C_{2,3} & C_{2,4} \\ C_{3,1} & C_{3,2} & C_{3,3} & C_{3,4} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.17}$$

$$\begin{aligned} u' &= uw' \\ v' &= vw' \\ C_{11}X + C_{12}Y + C_{13}Z + C_{14} - C_{31}uX - C_{32}uY - C_{33}uZ - C_{34}u &= 0 \\ C_{21}X + C_{22}Y + C_{23}Z + C_{24} - C_{31}vX - C_{32}vY - C_{33}vZ - C_{34}v &= 0 \end{aligned} \tag{3.18}$$

donde (u, v) son las coordenadas de los pixeles correspondientes al punto con respecto al mundo (X, Y, Z) [1].

La calibración requiere un objetivo tridimensional. Se debe conocer la posición del centro de cada marcador (X_i, Y_i, Z_i), $i \in [1, N]$ con respecto al marco objetivo T, siendo este desconocido. Se captura una imagen y se determinan las correspondientes coordenadas plano-imagen (u_i, v_i). Para cada uno de los N marcadores se apilan las dos ecuaciones de

la ecuación (3.18) para formar la ecuación matricial

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 \\ & & & & & \vdots & & & & & \\ X_N & Y_N & Z_N & 1 & 0 & 0 & 0 & 0 & -u_NX_N & -u_NY_N & -u_NZ_N \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -v_NX_N & -v_NY_N & -v_NZ_N \end{bmatrix} \begin{bmatrix} C_{11} \\ C_{12} \\ \vdots \\ C_{33} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_N \\ v_N \end{bmatrix} \tag{3.19}$$

La ecuación (3.19) se puede resolver para los elementos de matriz de la cámara $C_{11} \cdots C_{33}$. La solución puede determinarse dentro de un factor de escala desconocido y por convención C_{34} se establece igual a uno. La ecuación (3.19) tiene 11 incógnitas y para su solución requiere que $N \geq 6$. A menudo se utilizan más de seis puntos conduciendo a un conjunto sobredeterminado de ecuaciones que se resuelve utilizando mínimos cuadrados [1].

Si los puntos son coplanares entonces la matriz de la izquierda de la ecuación (3.19) pierde rango. Esta es la razón por la cual el objetivo de calibración debe ser tridimensional [1].

Los elementos de la matriz de la cámara son funciones de los parámetros intrínsecos y extrínsecos. Sin embargo, dada una matriz de la cámara, la mayoría de los valores de los parámetros pueden recuperarse [1].

Si el objetivo de calibración es un cubo, los marcadores son sus vértices y su marco de coordenadas T es paralelo a las caras del cubo con su origen en el centro del cubo. La *pose* de la cámara se estima con respecto al objetivo de calibración T y por lo tanto es ${}^{T}\hat{\xi}_{C}$. La posición verdadera del objetivo con respecto a la cámara es ${}^{C}\xi_{T}$. Si la estimación es exacta entonces ${}^{C}\xi_{T} \oplus {}^{T}\hat{\xi}_{C}$ será cero. Donde el operador $\oplus$ indica la composición de *poses* relativas [1].

3.2. Extracción de características de una imagen

Las imágenes pueden interpretarse como grandes matrices con valores de pixeles, pero para aplicaciones robóticas tienen demasiados datos sin información útil. Se necesita tener información sobre aspectos concisos tales como la *pose* del objeto, el tipo de objeto, la velocidad con que se mueve. Las respuestas a estos aspectos pueden obtenerse a partir de la información que se obtiene de la imagen, las características de la imagen. Estas son la esencia de la escena y la materia prima necesaria para la localización y el mapeo [1].

La extracción de características opera en una imagen y devuelve una o más características de la imagen. Son típicamente escalares (por ejemplo, área o relación de aspecto) o vectores cortos (por ejemplo, la coordenada de un objeto o los parámetros de una línea). La extracción de características es una etapa de concentración de información que reduce la velocidad de datos de $10^6 - 10^8$ bytes por segundo a la salida de una cámara a datos del orden de decenas de características por cuadro [1].

3.2.1. Características de región

La segmentación de imagen es el proceso de particionar una imagen en regiones significativas. El objetivo es segmentar o separar los píxeles que representan objetos de interés. Un requisito clave es la robustez, ya que el método se degrada a medida que se violan los supuestos, por ejemplo cambio de iluminación o punto de vista de la escena [1].

La segmentación de imágenes se divide en tres problemas. La primera es la clasificación, que es un proceso de decisión aplicado a cada píxel y asignándolo a una de las clases C_l, $c \in 0 \cdots C_l - 1$. Comúnmente se usa $C_l = 2$ y se conoce como clasificación binaria o binarización. Los pixeles se pueden clasificar como objeto ($c_l = 1$) o no-objeto ($c = 0$) y se muestran como píxeles blancos o negros, respectivamente. En la práctica se acepta que esta etapa es imperfecta y que los píxeles pueden ser clasificados erróneamente. Los pasos de procesamiento posteriores tendrán que lidiar con esto [1].

El segundo paso en el proceso de segmentación, es la representación, donde píxeles adyacentes de la misma clase están conectados para formar conjuntos espaciales $S_1 \cdots S_m$. Los conjuntos se pueden representar asignando una etiqueta de conjunto a cada píxel o mediante una lista de coordenadas de píxeles que define el límite del conjunto conectado. En la tercera y última etapa, los conjuntos S_i se describen en términos de características escalares o de valor vectorial tales como tamaño, posición y forma [1].

Clasificación en escala de grises

Una regla común para la clasificación binaria de niveles es:

$$c[u, v] = \begin{cases} 0 & I[u, v] < tr \\ 1 & I[u, v] \geq tr \end{cases} \quad \forall (u, v) \in I \tag{3.20}$$

donde la decisión esta basada simplemente en el valor del píxel. $I[u, v]$ es la imagen compuesta de pixeles. A este enfoque se le conoce como de umbral y tr es el umbral. El análisis necesario para elegir el umbral de manera adecuada se describe a continuación [1].

Dentro de un histograma (gráfica que muestra como se distribuyen en cantidad los pixeles a través de una escala de grises entre cero y uno) se pueden observar dos picos (Figura 3.6). A esta distribución se le conoce como bimodal. El pico más pequeño se compone de los pixeles brillantes y tiene un intervalo muy pequeño de variación de valor. El pico más grande se compone de pixeles más obscuros y tiene una variación mayor en

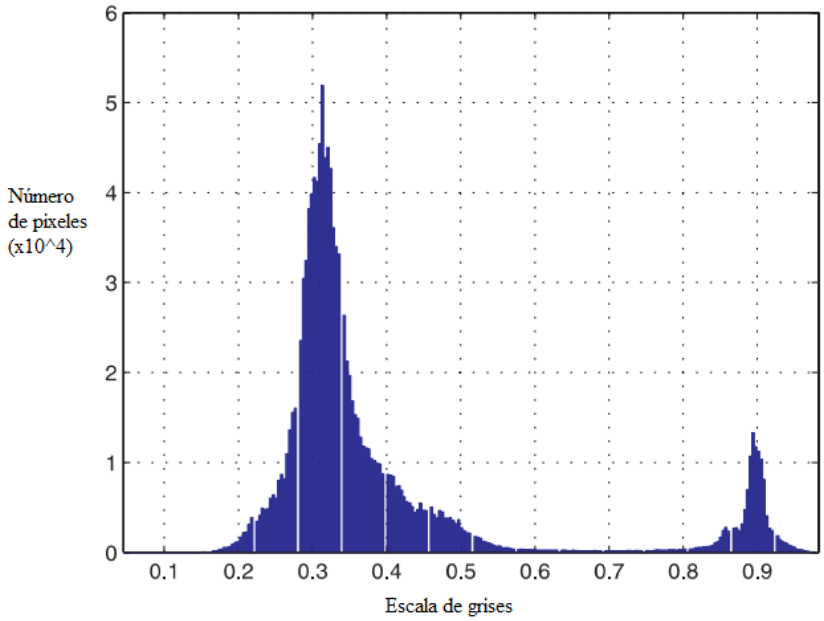

Figura 3.6: Histograma de una imágen [1].

el brillo. El umbral referido está en el valle entre los picos. El valor óptimo se obtiene con el método de Otsu. Este método separa los pixeles en dos clases de forma que minimiza la varianza de valores en cada clase y maximiza la varianza de valores entre clases. Este método asume que sólo existen dos picos. Es un método débil, ya que un cambio en iluminación de la escena significa que el umbral elegido podría ya no ser adecuado [1].

Una alternativa es usar un umbral local. El algoritmo Niblack calcula un umbral local (usado ampliamente en sistemas de reconocimiento de características)

$$tr[u,\, v] = \mu(W) + k\sigma(W) \tag{3.21}$$

donde W es la región sobre en punto (u, v), μ es la media y σ es la desviación estándar. El tamaño de la ventana W es un parámetro crítico y debe ser de tamaño similar al objeto observado [1].

3.2.2. Características de puntos

Las características de puntos son visualmente distintas en la imagen y a menudo se llaman puntos de interés, *keypoints* (puntos clave) o comúnmente, pero menos precisamente, puntos de esquina. A continuación se mencionan algunas técnicas clásicas para encontrar puntos de interés [1].

Detectores de esquina clásicos

Un punto en una línea tiene un valor alto de gradiente en una dirección normal a la línea. Sin embargo, el valor del gradiente a lo largo de la línea es bajo, lo que significa que un píxel en la línea se verá muy similar a sus vecinos en la línea. Por el contrario, un punto de interés es un punto que tiene un gradiente de imagen alto en direcciones ortogonales. Puede ser un solo píxel que tiene una intensidad significativamente diferente a todos sus vecinos o podría ser literalmente un píxel en la esquina de un objeto. Dado que los puntos de esquina son bastante distintos, tienen una probabilidad mucho mayor de ser detectados de forma fiable en diferentes vistas de la misma escena. Por lo tanto, son clave para las técnicas de multiple vista como la estéreo [1].

El primer detector de puntos de esquina fue el operador de interés de Moravec. Se basó en la intuición de que si una región de imagen W debe estar situada sin ambigüedad en otra imagen, debe ser suficientemente diferente para todas las regiones adyacentes superpuestas. Moravec definió la similitud entre una región centrada en (u, v) y una región adyacente, desplazada por $(\delta u, \delta v)$, como

$$s(u, v, \delta_u, \delta_v) = \sum_{(i,j)\in W} (I[u + \delta_u + i, v + \delta_v + j] - I[u + i, v + j])^2 \tag{3.22}$$

donde W es alguna región de la imagen local, normalmente una ventana cuadrada de $N \times N$. Esta se evalúa para desplazamientos en ocho direcciones cardinales (norte, noreste, ..., noroeste) representadas como $(\delta_u, \delta_v) \in D$ y el valor mínimo es

$$C_M(u, v) = \min_{(\delta_u,\delta_v)\in D} s(u, v, \delta_u, \delta_v) \tag{3.23}$$

La función $C_M(\cdot)$ se evalúa para cada pixel en la imagen y los puntos de interés son aquellos donde C_M es alto. La principal limitación del detector de Moravec es que es no isotrópico ya que examina el cambio de imagen, esencialmente el gradiente, en un número limitado de direcciones. En consecuencia, el detector puede dar una salida grande para un punto en una línea, lo cual no es deseable [1].

Se puede generalizar el enfoque definiendo la similitud como la suma ponderada de las diferencias cuadradas entre la región de la imagen y la región desplazada como

$$s(u, v, \delta_u, \delta_v) = \sum_{(i,j)\in W} \mathbf{W}[i, j](I[u + \delta_u + i, v + \delta_v + j] - I[u + i, v + j])^2 \tag{3.24}$$

donde $\mathbf{W}$ es una matriz de ponderación que enfatiza los puntos más próximos al centro de la ventana W. El término de la imagen más grande puede aproximarse por una serie truncada de Taylor y de forma compacta se tiene

$$s(u, v, \delta_u, \delta_v) = \begin{pmatrix} \delta_u & \delta_v \end{pmatrix} A \begin{pmatrix} \delta_u \\ \delta_v \end{pmatrix} \tag{3.25}$$

donde

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \tag{3.26}$$

con

$$\begin{aligned} a_{11} &= \sum \mathbf{W}[i,j] I_u^2[u+i, v+j] \\ a_{12} &= \sum \mathbf{W}[i,j] I_u[u+i, v+j] I_v[u+i, v+j] \\ a_{21} &= \sum \mathbf{W}[i,j] I_u[u+i, v+j] I_v[u+i, v+j] \\ a_{22} &= \sum_{\mathbf{W}} [i,j] I_v^2[u+i, v+j] \end{aligned}$$

donde I_u e I_v son los gradientes de la imagen vertical y horizontal respectivamente. Antes de continuar se define un operador espacial lineal importante, la convolución. Definida como

$$O[u,v] = K \otimes I = \sum_{(i,j) \in W} I[u+i, v+j] K[i,j] \quad \forall (u,v) \in I \tag{3.27}$$

donde $K \in \mathbb{R}^{W \times W}$ es el núcleo de la convolución. Para cada pixel de salida, la ventana correspondiente de pixeles de la imagen de entrada W se multiplica con el núcleo K. Un núcleo adecuado para suavizar una imagen (hacerla borrosa o desenfocarla) es la función Gaussiana bidimensional

$$G(u,v) = \frac{1}{2\pi\sigma^2} e^{\frac{u^2+v^2}{2\sigma^2}} \tag{3.28}$$

que es simétrica sobre el origen y el volumen bajo la curva es la unidad. La propagación es controlada por el parámetro de desviación estándar σ [1].

Si la matriz de ponderación es un núcleo gaussiano $W = G(\sigma_I)$ y se remplaza la sumatoria por una convolución entonces

$$A = \begin{pmatrix} G(\sigma_I) \bigotimes I_u^2 & G(\sigma_I) \bigotimes I_u I_v \\ G(\sigma_I) \bigotimes I_u I_v & G(\sigma_I) \bigotimes I_v^2 \end{pmatrix} \tag{3.29}$$

siendo una matriz 2×2 simétrica referida varias veces como el tensor de la estructura o matriz de autocorrelación. La cual contiene la estructura de la intensidad del vecindario local y sus valores propios proporcionan una descripción invariante rotacional del vecindario. Los elementos de la matriz A se calculan a partir de los gradientes de la imagen, cuadrados o multiplicados, y luego se suavizan mediante una matriz de ponderación. Esto reduce el ruido y mejora la estabilidad y fiabilidad del detector [1].

Un punto de interés (u, v) es uno para el cual $s(\cdot)$ es alto para todas las direcciones del vector (δ_u, δ_v). Es decir, en cualquier dirección que se mueve la ventana rápidamente se vuelve diferente a la región original. Si se considera la imagen original I como superficie, los valores propios de A son las principales curvaturas de la superficie en ese punto. Si ambos valores propios son pequeños entonces la superficie es plana, es decir, la región de la imagen tiene una intensidad local aproximadamente constante. Si un valor propio es alto y el otro bajo, entonces la superficie es en forma de cresta, lo que indica un borde. Si

ambos valores propios son altos, la superficie se eleva bruscamente, lo que consideramos una esquina [1].

El detector de Shi-Tomasi considera una esquina más importante, como el valor propio mínimo

$$C_{ST}(u, v) = \text{mín}(\lambda_1, \lambda_2) \tag{3.30}$$

donde λ_i son los valores propios de A. Los puntos en la imagen para los que esta medida es alta se denominan "buenas características para el seguimiento". El detector de Harris se basa en esta misma visión pero define la importancia de la esquina como

$$C_H(u, v) = \det(A) - k\mathsf{tr}(A) \tag{3.31}$$

De nuevo, un valor alto representa una mejor y distintiva esquina. Dado que $\det(A) = \lambda_1\lambda_2$ y $\mathsf{tr}(A) = \lambda_1 + \lambda_2$ el detector de Harris responde cuando ambos valores propios son grandes y evita elegantemente el cálculo de los valores propios de A lo cual tiene un costo computacional alto. Un valor comúnmente utilizado para k es 0.04. Otra variante es el detector de Noble:

$$C_N(u, v) = \frac{\det(A)}{\mathsf{tr}(A)} \tag{3.32}$$

el cual es aritméticamente simple pero potencialmente singular.

Normalmente, la intensidad de la esquina se calcula para cada píxel, dando como resultado una imagen de la importancia de la esquina. Después se aplica la supresión de máximos no locales para retener únicamente valores que son mayores que sus vecinos inmediatos. Una lista de estos puntos se crea y se clasifica de acuerdo a la importancia de la esquina de forma descendente. Puede considerarse un umbral para aceptar esquinas de una importancia particular o una fracción particular de la esquina más importante o las más importantes N esquinas [1].

Detectores de esquina escala-espacio

El detector de Harris de la sección anterior funciona bien en la práctica pero responde mal a los cambios de escala. El cambio de escala, debido al cambio de la cámara a la distancia de la escena, es común en muchas aplicaciones reales [1].

Antes de continuar se introducen los siguientes conceptos. Un medio para encontrar el punto de gradiente máximo es calcular la segunda derivada y determinar dónde es cero. El operador Laplaciano

$$\bigtriangledown^2 I = \frac{\partial^2 I}{\partial u^2} + \frac{\partial^2 I}{\partial v^2} = I_{uu} + I_{vv} \tag{3.33}$$

es la suma de la segunda derivada espacial en las direcciones horizontal y vertical. Para una imagen discreta esto se puede calcular por convolución con el núcleo Laplaciano, el cual es isotrópico (responde igualmente a los bordes en cualquier dirección). La segunda

derivada es aún más sensible al ruido que la primera derivada y se usa comúnmente en conjunción con una imagen suavizada gaussiana

$$\bigtriangledown^2 I = L \otimes (G(\sigma) \otimes I) = (L \otimes G(\sigma)) \otimes I \tag{3.34}$$

la cual se combina en el laplaciano del núcleo gaussiano (LoG) y L es el núcleo laplaciano dado en la ecuación (3.33). Esto puede escribirse analíticamente como

$$LoG(u,v) = \frac{\partial^2 G}{\partial u^2} + \frac{\partial^2 G}{\partial v^2} = \frac{1}{\pi\sigma^4}\left(\frac{u^2+v2}{2\sigma^2} - 1\right) e^{\frac{u^2+v^2}{2\sigma^2}} \tag{3.35}$$

el cual es conocido como el operador de Marr-Hildreth o el núcleo de sombrero mexicano [1].

La convolución de la imagen original con un núcleo gaussiano incrementando σ da como resultado el tamaño del núcleo y por lo tanto la cantidad de cálculo crece en cada paso de la escala. Una propiedad de un gaussiano es que un gaussiano convolucionado con otro gaussiano resulta en un gaussiano más amplio. En lugar de convolucionar la imagen original con gaussianos cada vez más amplios, se puede aplicar repetidamente el mismo gaussiano al resultado anterior. Teniendo en cuenta que el núcleo LoG se pude aproximar por la diferencia de dos gaussianos, se puede escribir

$$(G(\sigma_1) - G(\sigma_2)) \otimes I = G(\sigma_1) \otimes I - G(\sigma_2) \otimes I \tag{3.36}$$

donde $\sigma_1 > \sigma_2$. La diferencia del operador gaussiano aplicada a la imagen es equivalente a la diferencia de la imagen en dos niveles diferentes de suavizado. Si se realiza el suavizado mediante aplicación sucesiva de un gaussiano se tiene una secuencia de imágenes a niveles aumentados de suavizado. La diferencia entre los pasos sucesivos de la secuencia es, por tanto, una aproximación al laplaciano de gaussiano. La Figura 3.7 muestra esto en forma esquemática [1].

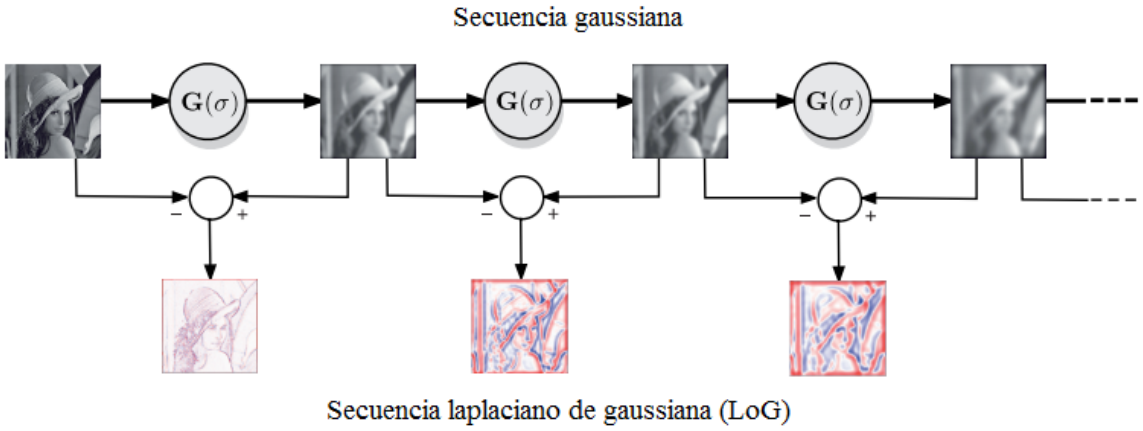

Figura 3.7: Esquema para el calculo de la secuencia gaussiana y laplaciano de gaussiana.

Se observa que una secuencia unicamente gaussiana de imágenes se vuelve cada vez más borrosa. En el laplaciano de la secuencia gaussiana los ojos oscuros son bloques fuertemente positivos a baja escala y el sombrero de color claro se convierte en una gota fuertemente negativa a gran escala [1].

Característica de punto escala-espacio

Los conceptos de escala-espacio sustentan una serie de detectores de características populares que encuentran puntos sobresalientes dentro de una imagen y determinan su escala y también su orientación. La Transformada de Característica de Escala-Invariante (SIFT) se basa en los máximos en una diferencia de secuencia gaussiana. La Característica Robusta Acelerada (SURF) se basa en los máximos en una secuencia aproximada gaussiana de Hessian [1].

El algoritmo SURF es más que un detector de características invariantes de escala, sino que también calcula un descriptor robusto. El descriptor es un vector de 64 elementos que codifica el gradiente de la imagen en subregiones de la región de soporte de una manera que es invariante al brillo, la escala y la rotación. Esto permite que los descriptores de características se acomplen adecuadamente con un descriptor del mismo punto del mundo en otra imagen incluso si su escala y orientación son muy diferentes. La diferencia de posición, escala y orientación de las características acopladas da alguna indicación del movimiento relativo de la cámara entre las dos vistas [1].

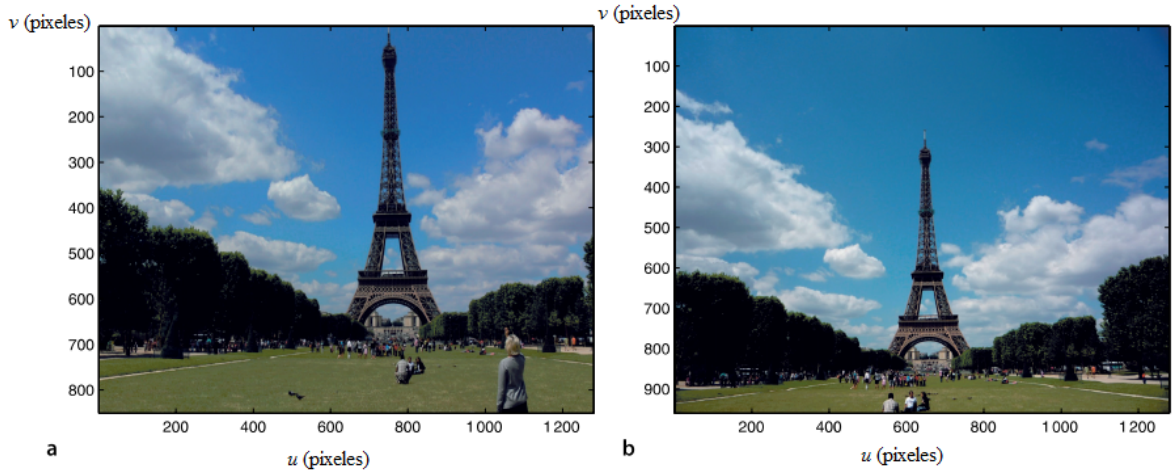

Figura 3.8: Dos vistas de la torre Eiffel [1].

3.3. Usando múltiples imágenes

La coordenada tridimensional del punto del mundo se pierde en el proceso de proyección de perspectiva. Todo lo que se sabe es que el punto del mundo se encuentra a lo largo de algún rayo en el espacio correspondiente a la coordenada del pixel, como se muestra en la Figura 3.1. Para recuperar la tercera dimensión, se necesita información adicional. En la sección anterior la información adicional eran parámetros de calibración de la cámara más un modelo de objeto geométrico, y esto nos permite estimar la *pose* tridimensional del objeto a partir de los datos de imagen bidimensionales [1].

Las coordenadas de pixeles desde una sola vista limitan el punto del mundo a lo largo de algún rayo. Si se localiza el mismo punto del mundo en otra imagen, tomado de una pose diferente pero conocida, se puede determinar otro rayo a lo largo del cual ese punto del mundo debe estar. El punto del mundo está en la intersección de estos dos rayos (un proceso conocido como triangulación o reconstrucción 3D). Si se observan suficientes puntos, se puede estimar el movimiento 3D de la cámara entre las vistas, así como la estructura 3D del mundo. El reto subyacente es encontrar el mismo punto del mundo en múltiples imágenes. Éste es el problema de correspondencia, un problema importante pero no trivial que se discute a continuación [1].

3.3.1. Correspondencia de características

Correspondencia es el problema de encontrar las coordenadas de pixeles en dos imágenes diferentes que corresponden al mismo punto en el mundo. Considere el par de imágenes reales mostradas en la Figura 3.8.

Se muestra la misma escena vista desde dos posiciones diferentes usando dos cámaras

diferentes, el tamaño del pixel, la distancia focal y el número de pixeles para cada imagen son diferentes. Las escenas son complejas y la determinación de la correspondencia no es trivial. Más de la mitad de los pixeles de cada escena corresponden al cielo azul y es imposible hacer coincidir un pixel azul en una imagen con el pixel azul correspondiente en el otro, estos pixeles no son suficientemente distintos. Esta situación es común y puede ocurrir con regiones de imagen homogéneas, tales como sombras oscuras, láminas lisas de agua, nieve u objetos artificiales lisos tales como paredes o los lados de los coches [1].

La solución es escoger solamente aquellos puntos que son distintivos. Se pueden usar los detectores de puntos de interés para encontrar las características de esquina Harris o SURF. Se ha simplificado el problema; en lugar de millones de píxeles se tienen sólo cientos de puntos distintivos [1].

3.3.2. Geometría de múltiples vistas

En la Figura 3.9 se muestran las relaciones geométricas entre imágenes de un solo punto P observado desde dos puntos de vista diferentes. Esta geometría puede representar el caso de dos cámaras que visualizan simultáneamente la misma escena, o una cámara que toma una imagen desde dos puntos de vista diferentes. El centro de cada cámara, los orígenes de 1 y 2, más el punto del mundo P definen un plano en el espacio, el plano epipolar. El punto P se proyecta sobre los planos de imagen de las dos cámaras y las coordenadas de pixeles ${}^1p$ y ${}^2p$ respectivamente. Estos puntos se conocen como puntos conjugados [1].

Considerando la imagen uno. El punto de la imagen ${}^1e$ es una función de la posición de la cámara dos. El punto de imagen ${}^1p$ es una función del punto mundial P. El centro de cámara, ${}^1e$ y ${}^1p$ definen el plano epipolar y por lo tanto la línea epipolar ${}^2l$ en la imagen dos. Por definición, el punto conjugado ${}^2p$ debe estar en esa línea. Por el contrario, ${}^1p$ debe situarse a lo largo de la línea epipolar en la imagen uno ${}^1l$ que está definida por ${}^2p$ en la imagen dos [1].

Esta es una relación geométrica fundamental e importante. Dado un punto en una imagen sabemos que su conjugado está a lo largo de una línea en la otra imagen [1].

3.3.3. La Matriz fundamental

La relación epipolar mostrada en la Figura 3.9 puede expresarse concisa y elegantemente como

$$ {}^2\tilde{p}^T F {}^1\tilde{p} = 0 \tag{3.37}$$

donde ${}^1\tilde{p}$ y ${}^2\tilde{p}$ son los puntos de la imagen ${}^1p$ y ${}^2p$ expresados en forma homogénea y F es una matriz 3×3 conocida como matriz fundamental [1].

Se pueden agrupar los últimos dos términos como

$$ {}^2\tilde{l} \backsimeq F {}^1\tilde{p} \tag{3.38}$$

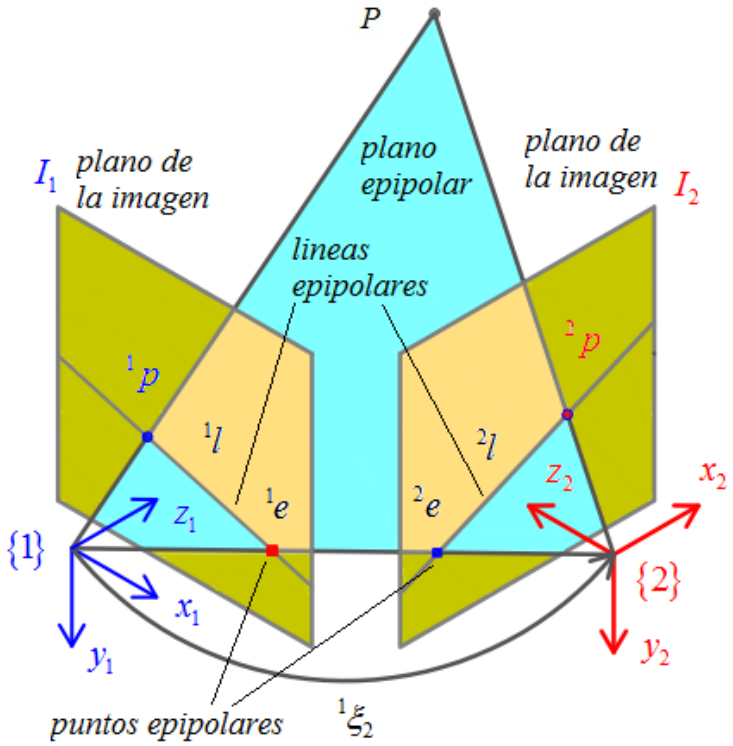

Figura 3.9: Geometría Epipolar.

el cual es la ecuación de una línea, la línea epipolar, a lo largo de la cual el punto conjugado en la imagen dos debe estar.

$$ {}^2\tilde{p}^T \, {}^2\tilde{l} = 0 \tag{3.39}$$

Esta línea es una función, ecuación (3.38), del punto ${}^1\tilde{p}$ en la imagen uno y es una prueba poderosa en cuanto a si un punto en la imagen dos es o no un posible conjugado. Tomando la transposición de ambos lados de la ecuación (3.37) se tiene

$$ {}^1\tilde{p}^T F^T \, {}^2\tilde{p} = 0 \tag{3.40}$$

donde se puede escribir la línea epipolar para la cámara uno

$$ {}^1\tilde{l} \backsimeq F^T \, {}^2\tilde{p} \tag{3.41}$$

en términos de un punto visto por la cámara dos [1].

La matriz fundamental es una función de los parámetros de la cámara y la posición relativa de la cámara entre las vistas.

$$ F \backsimeq K^{-1} S(T) R K \tag{3.42}$$

donde K es la matriz intrínseca de la cámara, $S(\cdot)$ es una matriz anti-simétrica que codifica la traslación de la cámara y ${}^2\xi_1 \sim (R, T)$ es la pose relativa de la cámara uno con respecto a la dos [1].

$$ {}^1\xi_2 = \begin{bmatrix} R_{12} & T_2 \\ 0 & 1 \end{bmatrix} \tag{3.43} $$

La matriz fundamental es singular de rango dos y tiene siete grados de libertad. Los epipolos están codificados en el espacio nulo de la matriz. El epipolo para la cámara uno es el espacio nulo derecho de F. El epipolo para la cámara dos es el espacio nulo izquierdo de la transposición de la matriz fundamental [1].

3.3.4. La Matriz Esencial

La restricción geométrica epipolar también puede expresarse en términos de coordenadas de imagen normalizadas

$$ {}^2\tilde{x}^T E^1\tilde{x} = 0 \tag{3.44} $$

donde E es la matriz esencial y ${}^1\tilde{x}$ y ${}^2\tilde{x}$ son puntos conjugados en coordenadas de la imagen normalizadas homogéneas. Esta matriz es función de la pose relativa de la cámara.

$$ E \backsimeq K^{-1} S(T) R K \tag{3.45} $$

donde ${}^2\xi_1 \sim (R, T)$ es la pose relativa de la cámara uno con respecto a la dos. La matriz esencial es singular, tiene rango dos y tiene dos valores iguales singulares y uno cero. La matriz esencial tiene sólo 5 grados de libertad y está completamente definida por 3 parámetros de rotación y 2 de traslación. Para la rotación pura, cuando $T = 0$, la matriz esencial no está definida [1].

Teniendo en cuenta que $\tilde{p} \backsimeq K\tilde{x}$ y sustituyéndolo en (3.44) se puede escribir:

$$ {}^2\tilde{p}^T K_2^{-T} E K_1^{-11} \tilde{P} = 0 \tag{3.46} $$

de donde

$$ F \backsimeq K_2^{-T} E K_1^{-1} \tag{3.47} $$

y de forma similar se tiene

$$ E \backsimeq K_2^T F K_1 \tag{3.48} $$

en términos de los parámetros intrínsecos de las dos cámara implicadas. Dado que $E \backsimeq \lambda E$, la parte traslacional de la matriz de transformación homogénea tiene un factor de escala desconocido [1].

En resumen, estas matrices, la fundamental y la esencial, codifican la geometría de las dos cámaras. La matriz fundamental y un punto en una imagen definen una línea epipolar en la otra imagen a lo largo de la cual debe estar el punto conjugado. La matriz esencial

codifica la posición relativa de los centros de las dos cámaras y la pose puede extraerse, con dos valores posibles, y con traslación escalada por un factor desconocido. En el mundo real el movimiento de la cámara es difícil de medir y la cámara puede no estar calibrada. En su lugar, se puede estimar la matriz fundamental directamente a partir de puntos de la imagen correspondientes [1].

3.3.5. Estimación de la matriz fundamental

Suponiendo que se tienen N pares de puntos correspondientes en dos vistas de la misma escena $({}^1p_i, {}^2p_2), i = 1 \cdots N$. Si $N \geq 8$ la matriz fundamental puede estimarse a partir de estos dos conjuntos de puntos correspondientes. La matriz estimada tiene la propiedad de rango requerida [1]. La matriz fundamental puede estimarse con el algoritmo llamado de Muestreo y Consenso Aleatorio o RANSAC [1].

La estimación de una matriz fundamental requiere ocho puntos por lo que se eligen al azar ocho candidatos correspondientes (la muestra) y se estima F para crear un modelo. Este modelo se prueba en contra de todos los pares de candidatos y los que encajan votan por este modelo. El proceso se repite varias veces y devuelve el modelo que tuvo más apoyo (el consenso). Dado que la muestra es pequeña, la probabilidad de que contenga todos los pares de candidatos válidos es alta. Los pares de puntos que soportan el modelo se llaman *inliers* y los que no son *outliers*. RANSAC es eficaz y eficiente en la búsqueda del conjunto *inlier*, incluso en presencia de un gran número de outliers (más del 50 %) [1].

3.3.6. Homografía planar

En esta sección se considera una cámara que ve un grupo de puntos del mundo P_i que se encuentran en un plano. Son vistos por dos cámaras diferentes y la proyección en las cámaras son ${}^1p_i$ y ${}^2p_i$ respectivamente que están relacionados por

$$ {}^2\tilde{p}_i \backsimeq H {}^1\tilde{p}_i \tag{3.49} $$

donde H es una matriz no-singular de 3×3 conocida como homografía, una homografía planar o la homografía *inducida* por el plano [1].

Considerando el par de cámaras de la Figura 3.9 y poniendo un conjunto de puntos. Los puntos son proyectados en ambas cámaras. Y las imágenes se muestran en la Figura 3.10 a y b, respectivamente.

Así como se hizo para la matriz fundamental, si $N \geq 8$ se puede estimar la matriz H a partir de dos conjuntos de puntos correspondientes [1].

De acuerdo con la ecuación (3.49) se puede obtener la posición de los puntos de la rejilla en la imagen dos desde las coordenadas correspondientes de la imagen uno, se superponen en la imagen dos como símbolos +. Esto se muestra en la Figura 3.10b y se ve que los

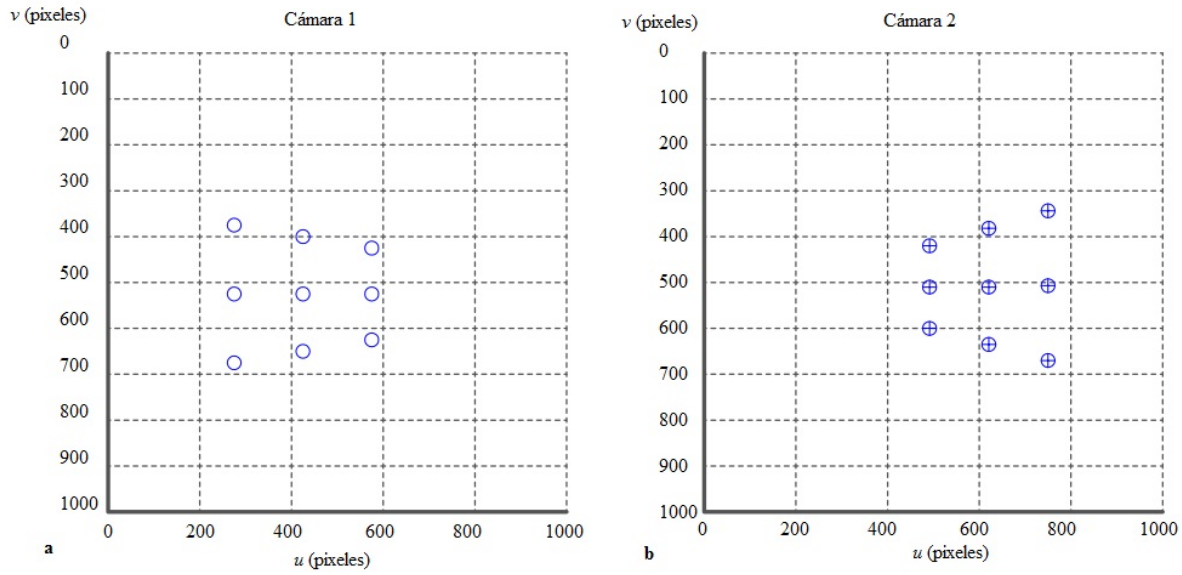

Figura 3.10: Vistas de la rejilla plana oblicua de puntos desde dos puntos de vista diferentes.

puntos predichos están perfectamente alineados con la proyección real de los puntos del mundo. La inversa de la matriz homográfica

$$^1\tilde{p}_i \backsimeq H^{-1}{}^2\tilde{p}_i \tag{3.50}$$

realiza el mapeo inverso desde las coordenadas de la imagen dos a la uno [1].

La matriz fundamental obliga al punto conjugado a situarse a lo largo de una línea, pero la homografía nos dice exactamente dónde está el punto conjugado en la otra imagen, siempre que los puntos se encuentren en un plano. Se puede usar esta condición como una prueba para determinar si los puntos están o no en un plano [1].

En la práctica no se sabe de antemano qué puntos pertenecen al plano por lo que se puede usar de nuevo RANSAC, que encuentra la homografía que mejor explica la relación entre los conjuntos de puntos de la imagen [1].

La geometría relacionada con la homografía se muestra en la Figura 3.11. Se puede expresar la homografía en coordenadas normalizadas (cuando $f = 1$) de la imagen [1].

$$^2\tilde{x} \backsimeq H_E\, ^1\tilde{x} \tag{3.51}$$

donde H_E es la homografía Euclidiana que se puede escribir como

$$H_E \backsimeq R + \frac{t}{d} n^T \tag{3.52}$$

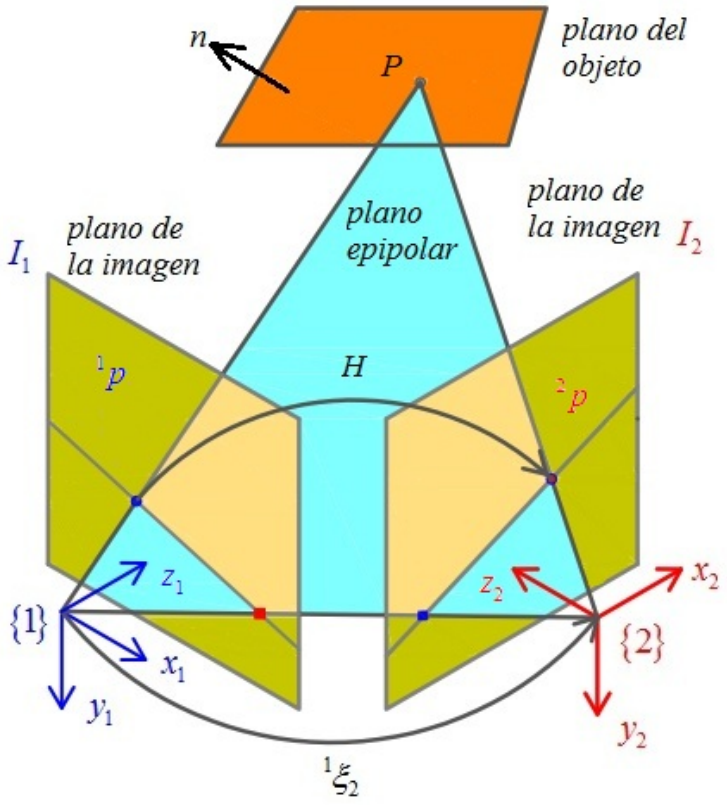

Figura 3.11: Geometría de la homografía.

En términos de movimiento $(R, t) \sim^2 \xi_1$ y el plano $n^T P + d = 0$ con respecto al marco $\{1\}$. Las homografías Euclidiana y proyectiva están relacionadas por

$$H_E \backsimeq K^{-1} H K \tag{3.53}$$

donde K es la matriz paramétrica de la cámara [1].

En cuanto a la matriz esencial, la homografía proyectiva puede descomponerse para producir la *pose* relativa $^1\xi_2$ en forma de transformación homogénea así como la normal al plano [1].

3.3.7. Estructura y movimiento

Suponiendo dos imágenes observadas por una cámara. Los puntos correspondientes entre la imagen actual y la previa se utilizan para estimar la matriz fundamental y luego actualizar a una esencial. Después se descomponen en una estimación de movimiento de la última a la actual posición de la cámara. La parte rotacional de la *pose* estimada es exacta, pero la parte traslacional tiene el problema del factor de escala desconocido [1].

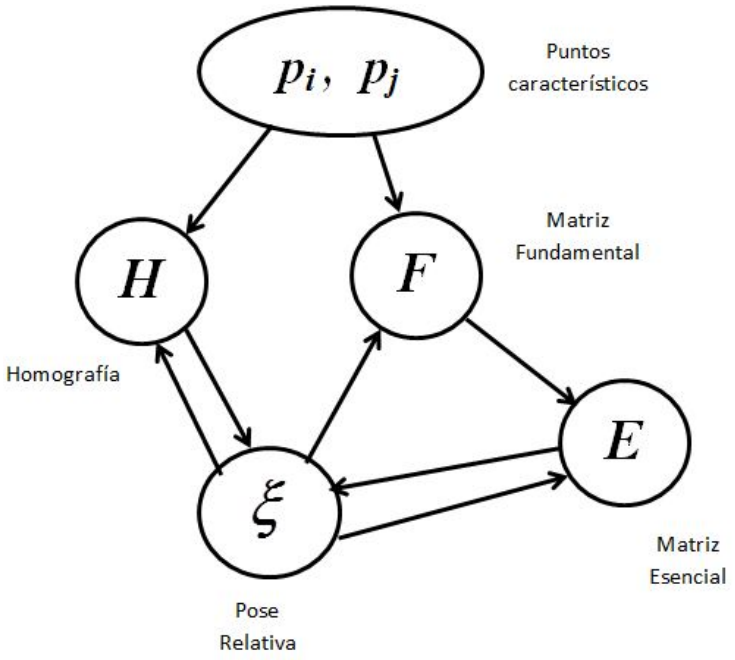

Figura 3.12: Relaciones entre puntos característicos, matrices y *pose* de la cámara.

Sin conocimiento del factor de escala no es posible estimar el movimiento incremental entre las vistas. La solución implica incorporar otras fuentes de información. La odometría o el GPS pueden proporcionar una estimación de la magnitud del movimiento de traslación. Alternativamente se puede usar una cámara estéreo para proporcionar la información de profundidad directamente. Otra opción es considerar que se conoce la altura sobre el suelo de un solo punto observado en el mundo [1].

3.4. Visual SLAM

Una vez que se ha descrito la forma en que se puede obtener la *pose* (la cual contiene posición) de una cámara a través de imágenes, se procede a describir un sistema de SLAM visual. El cual implementa herramientas de extracción de características. La base para entender los extractores de características se ha abordado en la sección 3.2.2, y aunque se utiliza un extractor diferente a los clásicos, los conceptos de sus funcionamientos son equivalentes (por ejemplo la invarianza a la escala o el núcleo de la convolución). También se implementa el calculo de la matriz fundamental, esencial, homografía planar y parámetros de la cámara. Las relaciones entre estas matrices y el movimiento de la cámara se resumen en la Figura 3.12. En la actualidad se han desarrollado una gran cantidad de algoritmos referentes a estos problemas. En esta sección se da una breve descripción de algunos de ellos.

A continuación se describen conceptos útiles del sistema de SLAM visual. Se le da el nombre de ORB SLAM dado que se implementa el detector de características ORB.

3.4.1. Algunos conceptos

- ***Bundle Adjustment* (BA).** Es el problema de refinar una reconstrucción visual para producir conjuntamente de manera óptima una estructura 3D y estimaciones de parámetros de visualización (*pose* de la cámara y/o calibración). *Óptimo* significa que las estimaciones de parámetros se encuentran minimizando una función de costo que cuantifica el error de ajuste del modelo, y conjuntamente que la solución es simultáneamente óptima con respecto a las variaciones tanto de estructura como de cámara. El nombre se refiere a los "paquetes" de rayos de luz que dejan cada característica 3D y convergen en cada centro de la cámara, los cuales se ajustan de manera óptima con respecto a las características y posiciones de la cámara. Todos los parámetros de la estructura y de la cámara se ajustan de manera conjunta "en un paquete" [17].

  BA es un problema geométrico de estimación de parámetros geométricos, siendo los parámetros las coordenadas combinadas de características 3D, las *poses* de la cámara y las calibraciones [17].

- ***Oriented FAST and Rotated BRIEF* (ORB).** Técnica de detección y descripción de puntos clave. Tiene un funcionamiento extremadamente rápido, mientras que sacrifica muy poco en exactitud de funcionamiento. ORB es invariante a la escala y rotación, es robusto al ruido y transformaciones afines, y tiene una velocidad de 25 fps [18].

  El algoritmo es una combinación del detector de puntos clave FAST, y el algoritmo descriptor de puntos clave BRIEF modificado para manejar los puntos clave orientados [18].

- ***Oriented FAST Keypoints* (FAST).** El detector original de puntos clave FAST prueba 16 pixeles en un círculo alrededor de un pixel. Si el pixel central es más oscuro o más brillante que un número de umbral de pixeles de 16, se determina que es una esquina. Para hacer este procedimiento más rápido, se utiliza un enfoque de aprendizaje de la máquina para decidir un orden eficiente de la comprobación de los 16 pixeles. La adaptación de FAST en ORB detecta esquinas a múltiples escalas haciendo una pirámide de escala de la imagen, y añade orientación a estas esquinas encontrando el centroide de intensidad. El centroide de intensidad de un grupo de pixeles está dado por [18]

$$Centroide = \left(\frac{m_{10}}{m_{00}} - \frac{m_{01}}{m_{00}} \right) \tag{3.54}$$

donde

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{3.55}$$

La orientación del conjunto es la orientación del vector que conecta el centro del conjunto con el centroide de intensidad. Específicamente

$$\theta_{orientacion} = \arctan 2(m_{01}, m_{10}) \tag{3.56}$$

- ***Binary Robust Independent Elementary Features*(BRIEF).** El descriptor se basa en la filosofía de que un punto clave en una imagen puede ser descrito por una serie de pruebas de intensidad de pixeles binarios alrededor del punto clave. Lo hace seleccionando pares de pixeles alrededor del punto clave, de acuerdo con un patrón de muestreo aleatorio o no aleatorio, para luego comparar las intensidades. La prueba devuelve uno si la intensidad del primer pixel es mayor que la del segundo y cero en caso contrario. Dado que todas estas salidas son binarias, se pueden conjuntar en bytes para un almacenamiento de memoria eficiente. Una gran ventaja es también que, puesto que los descriptores son binarios, la medida de distancia entre dos descriptores es Hamming y no Euclidiana. La distancia Hamming entre dos cadenas binarias de la misma longitud es el número de bits que difieren entre ellos. La distancia Hamming puede implementarse muy eficientemente haciendo una operación XOR bit a bit entre los dos descriptores y luego contando el número de unos [18].

  La implementación BRIEF en ORB utiliza un algoritmo de aprendizaje de máquina para obtener un patrón de selección de pares para escoger 256 pares que capturan la mayor cantidad de información [18].

  Para compensar la orientación del punto clave, las coordenadas del conjunto alrededor de este se rotan antes de seleccionar pares y realizar las 256 pruebas binarias [18].

  En la Figura 3.13 se muestra un ejemplo de la implementación del detector ORB.

- **RANdom Sample Consensus (RANSAC).** Potente método para encontrar puntos de datos que se ajusten a un modelo en particular. RANSAC es bastante resistente a los "*outlieres*" (puntos de datos que no se ajustan al modelo dado). Para explicar el algoritmo, se considera el problema de encontrar una línea en un conjunto de datos ruidoso de puntos 2D. La estrategia a seguir es; muestrear de manera aleatoria puntos de un conjunto de datos. Encontrar la ecuación de una línea que se ajuste a esos puntos. Encontrar "*inlieres*" (puntos que se ajustan a este modelo de la línea). Para determinar si un punto es un *inlier* o un *outlier* con respecto a un modelo, se necesita una medida de la distancia. Esa medida puede ser la distancia Euclidiana de la línea desde el punto. Iterar hasta encontrar una línea que tiene más *inlieres* que un determinado número pre-decidido.

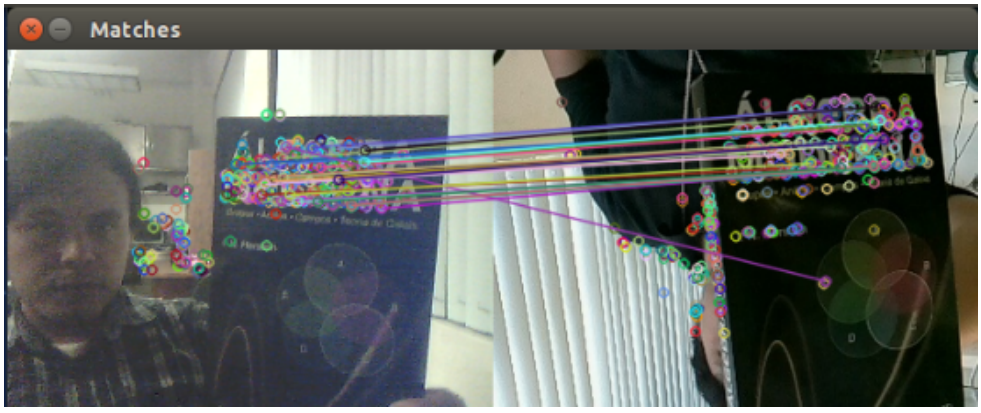

Figura 3.13: Detector de características ORB. La imagen de la derecha esta guardada en la memoria. La imagen de la izquierda corresponde a un cuadro tomado en linea. Se muestra los acoplamientos con líneas.

- **Gráfica de covisibilidad.** Un mapa de covisibilidad M_t, es una gráfica no dirigida con nodos que representan puntos de referencia (características visuales distintivas) y bordes que representan la covisibilidad entre cada punto de referencia. A medida que se procesa cada imagen, un conjunto de características visuales ℓ_i, detectan y se representan por un descriptor vectorial como SIFT o SURF. El punto de referencia está además asociado con una palabra visual cuantificada, que se toma por la coincidencia más cercana en un diccionario visual precargado V [19]. Por tanto, cada imagen proporciona un conjunto de palabras, que representan una observación Z_t, capaz de mantener cierta invariancia a los cambios de punto de vista y de iluminación. En cuadros siguientes, algunas características se siguen y se representan con la misma referencia ℓ_i. Un ejemplo simple de un mapa de covisibilidad y observación de consulta se puede ver en la Figura 3.14 [20]

3.4.2. Descripción del Sistema

El sistema ORB SLAM implementado se basa en [2]. Esta sección describe las herramientas que utiliza.

Selección de características

Se utilizan características que requieran para su extracción mucho menos que 33ms por imagen, lo cual excluye a los populares SIFT ($\sim 300ms$) [21], SURF ($\sim 300ms$) [22] o al reciente A-KAZE($\sim 100ms$) [23]. Se requiere también rotación invariante para obtener capacidades de reconocimiento de zona general. Por lo tanto, el sistema utiliza ORB [24], el cual esta orientado a esquinas FAST multiescala con un descriptor de 256 bits asociado.

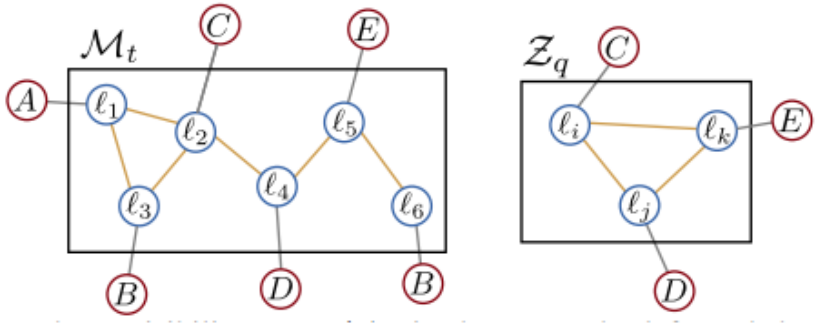

Figura 3.14: Mapa de covisibilidad. En la figura se representa que palabras (A, B, C, D o E) están asociadas con cada referencia ℓ_i.

Además ORB tiene buen rendimiento en el reconocimiento de zonas.

Hilos de Seguimiento y mapeo local

Una vista general se muestra en la Figura 3.15, donde se representan los hilos que se ejecutan en paralelo: seguimiento y mapeo local. El seguimiento está a cargo de la localización de la cámara con cada cuadro y decide cuando insertar un nuevo cuadro clave (*keyframe*). Se lleva a cabo primero un acoplamiento inicial con el cuadro previo y se optimiza la *pose* utilizando el *BA de sólo movimiento*. Si el seguimiento se pierde (por ejemplo debido a oclusiones o movimientos abruptos), el módulo de reconocimiento de zona se utiliza para llevar a cabo una relocalización global. Una vez que hay una estimación inicial de la *pose* de la cámara y de acoplamientos de características, se recupera un mapa visible local usando la gráfica de covisibilidad de cuadros clave que se obtienen por el sistema, Figura 3.16(a) y 3.16(b). Entonces se buscan las correspondencias con los puntos del mapa local mediante reproyección y se optimiza la pose de la cámara nuevamente con todos los acoplamientos. Finalmente el hilo de seguimiento decide si se inserta un nuevo *keyframe*.

El mapeo local procesa nuevos cuadros clave y realiza un *BA local* para alcanzar una reconstrucción óptima en las cercanías de la pose de la cámara. Se buscan nuevas correspondencias para características ORB desacopladas en el nuevo cuadro clave en relación con cuadros clave de la gráfica de covisibilidad para triángular nuevos puntos.

Después de la creación, basándose en la información recopilada durante el seguimiento, se aplica una política exigente de selección de puntos con el fin de conservar únicamente los de alta calidad. El mapeo local también está a cargo de identificar cuadros clave redundantes.

Finalmente se lleva a cabo una optimización de la *pose* sobre las restricciones de similitud para alcanzar la consistencia global. Además se realiza la optimización sobre una

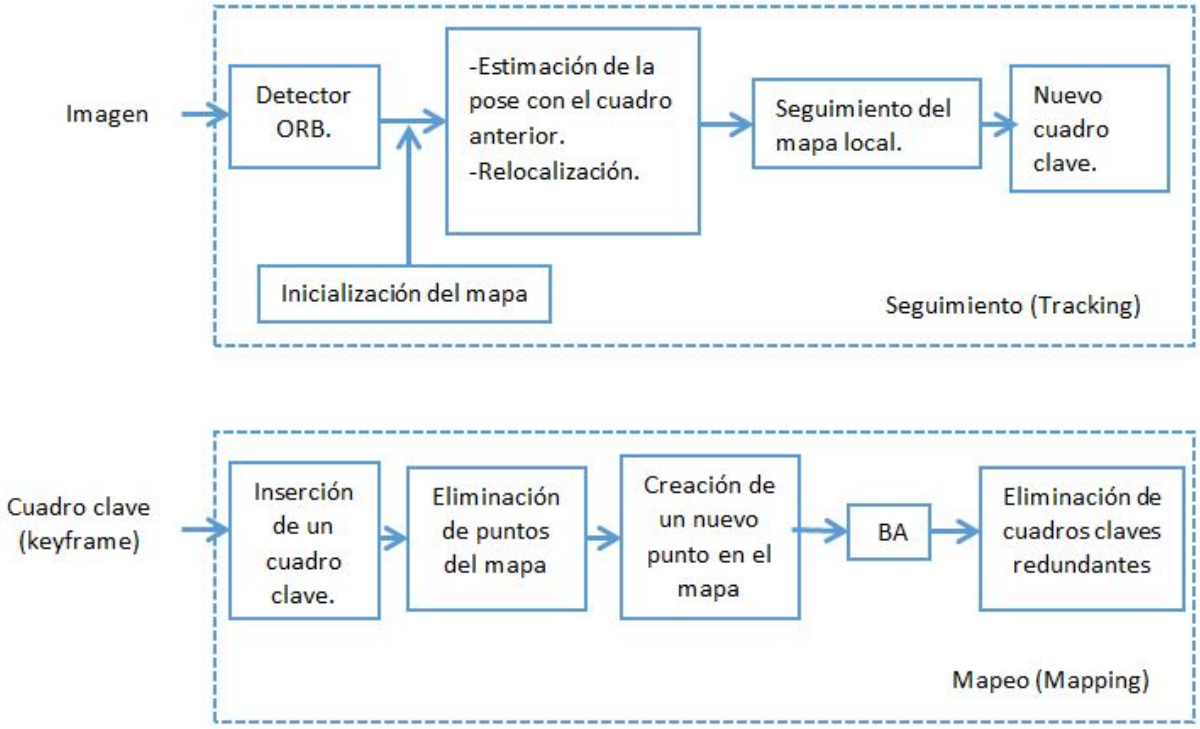

Figura 3.15: Descripción general del sistema ORB SLAM.

Gráfica esencial, una subgráfica más dispersa que la gráfica de covisibilidad.

Se utiliza el algoritmo de Levenberg-Marquardt para llevar a cabo todas las optimizaciones.

Puntos del mapa, cuadros clave y sus selecciones

Cada punto de mapa p_i almacena

- Su posición 3D $X_{w,i}$ en el sistema de coordenas del mundo.
- La dirección visible n_i, el cual es el vector principal unitario de todas sus direcciones visibles (los rayos que unen el punto con el centro óptico de los cuadros clave que lo observan).
- Un descriptor representativo ORB D_i, cuya distancia Hamming es mínima con respecto a todos los descriptores asociados en los cuadros clave que observan el punto.
- Las distancias máxima d_{max} y mínima d_{min} a las cuales el punto puede observarse, de acuerdo a los límites de la escala de invariancia de las características ORB.

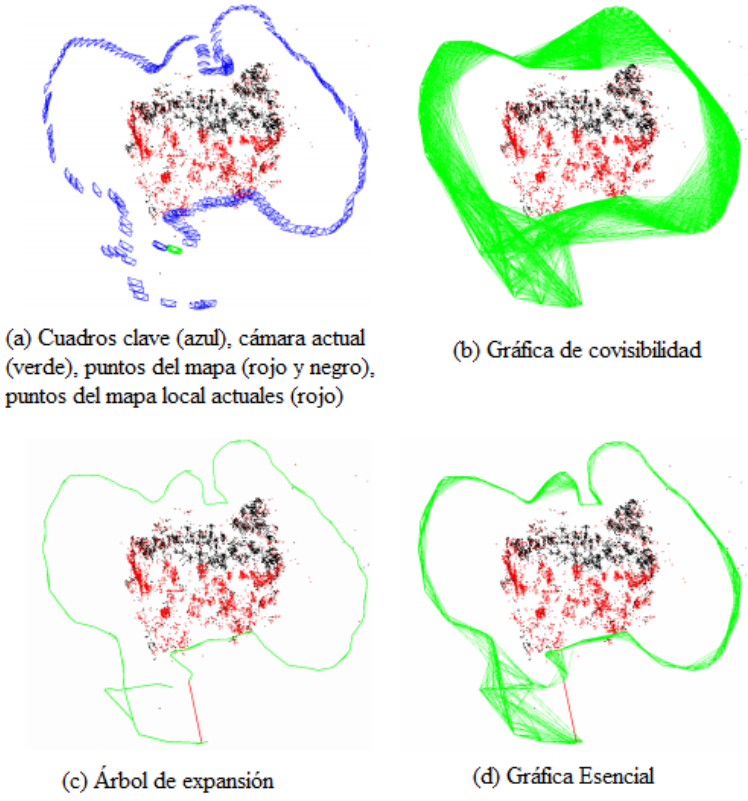

Figura 3.16: Reconstrucción y gráficas [2].

Cada cuadro clave K_i almacena:

- La pose de la cámara T_{iw}, la cual transforma puntos con respecto al mundo al sistema de coordenadas de la cámara.
- Los parámetros intrínsecos de la cámara, incluyendo la distancia focal y el punto principal.
- Todas las características ORB extraídas en el cuadro, asociadas o no a un punto del mapa cuyas coordenadas no están distorsionadas si se provee un modelo de distorsión.

Se crean puntos del mapa y cuadros clave con una política, mientras un mecanismo posterior de recolección exigente se encarga de la detección de cuadros clave redundantes y puntos del mapa erróneamente acoplados o sin seguimiento. Esto permite una expansión flexible del mapa durante la exploración, lo cual aumenta la robustez del seguimiento bajo condiciones difíciles (por ejemplo, rotaciones y movimientos rápidos). Mientras su tamaño

es delimitado por revisiones continuas en el mismo entorno, es decir, una operación que puede continuar por siempre. Adicionalmente los mapas contienen muy pocos *outliers*, a expensas de contener menos puntos.

Gráfica de covisibilidad y gráfica esencial

La información de covisibilidad entre cuadros clave es muy útil en diversas tareas del sistema y está representada como una gráfica ponderada no dirigida. Cada nodo es un cuadro clave y un borde entre dos cuadros clave existe si estas comparten observaciones de los mismos puntos del mapa (al menos 15), siendo el peso θ_{cov} del borde el número de puntos del mapa comunes.

Con la intención de no incluir todos los bordes proporcionados por la gráfica de covisibilidad, las cuales pueden ser muy densas, se construye una gráfica esencial que retenga todos los nodos (cuadros clave), excepto los bordes, preservando una conexión fuerte. El sistema construye de manera incremental, un árbol de expansión desde el cuadro clave inicial, lo cual proporciona una subgráfica conectada de la gráfica de covisibilidad con un mínimo de bordes. Cuando se inserta un nuevo cuadro clave, se incluye un árbol ligado al cuadro clave que comparte la mayoría de las observaciones del punto y cuando el cuadro clave se borra por la política de recolección, el sistema actualiza las conexiones afectadas por ese cuadro clave. La gráfica esencial contiene el árbol de expansión, el subconjunto de bordes de la gráfica de covisibilidad con alta covisibilidad ($\theta_{covMin} = 100$), resultando en una fuerte red de cámaras. En la Figura 3.16 se muestra un ejemplo de una gráfica de covisibilidad, un árbol de expansión y una gráfica esencial asociada.

Paquetes de palabras de reconocimiento de zonas

El sistema incorpora un módulo de paquetes de palabras de reconocimiento de zonas, para llevar a cabo la relocalización. Las palabras visuales son una discretización del espacio descriptor, lo cual es conocido como el vocabulario visual. El vocabulario se crea fuera de línea con los descriptores ORB extraídos de un juego de imágenes. Si las imágenes son lo bastante generales, se puede utilizar el mismo vocabulario para diferentes entornos. El sistema construye de una forma incremental una base de datos, el cual se almacena para cada palabra visual del vocabulario, en los cuales los cuadros clave han sido vistos, por lo que la consulta de la base de datos puede ser muy eficiente. La base de datos también se actualiza cuando un cuadro clave se borra por el procedimiento de selección.

Debido a que existe superposición visual entre cuadros clave, al consultar la base de datos no existe un único cuadro clave con alta puntuación. Entonces se agrupan las palabras clave que están conectadas en la gráfica de covisibilidad. Además, la base de datos devuelve todas las coincidencias de cuadros clave cuyas puntuaciones son mayores al 75 por ciento de la mejor puntuación.

Cuando se desea calcular las correspondencias entre dos conjuntos de características

ORB, se puede restringir la fuerza bruta coincidente sólo para esas características que pertenecen al mismo nodo en el árbol de vocabulario a cierto nivel (se selecciona el segundo de un total de seis). Se usa este truco cuando se buscan coincidencias para triangular nuevos puntos y en relocalización.También se refinan las correspondencias con una prueba de consistencia de orientación, que descarta *outliers* asegurando una rotación coherente para todas las correspondencias.

3.4.3. Inicialización automática del mapa

El objetivo de la inicialización del mapeo, implementado en el hilo de seguimiento de la Figura 3.15, es calcular la *pose* relativa entre dos cuadros para triangular un conjunto inicial de puntos del mapa. Este método debe ser independiente de la escena (planar o general) y no debe requerir intervención humana para seleccionar una buena configuración de dos vistas, es decir, con paralaje significativo. Se propone para el calculo en paralelo dos modelos geométricos; una homografía suponiendo una escena planar y una matriz fundamental suponiendo una no planar. Se selecciona de manera heurística el modelo y se trata de recuperar la *pose* con el método específico de acuerdo al modelo seleccionado.

El método sólo inicializa cuando la configuración de dos vistas es segura, detectando casos de bajo paralaje y evitando inicializar un mapa erróneo. Los pasos del algoritmo son:

1. Encontrar correspondencias iniciales. Extraer características ORB en el cuadro actual F_c y buscar acoplamientos ${}^c\tilde{p} \leftrightarrow {}^r\tilde{p}$ en el cuadro de referencia F_r. Si no se encuentran suficientes acoplamientos se reinicia el cuadro de referencia.

2. Cálculo en paralelo de los dos modelos. Se calcula en forma paralela en dos hilos, una homografía H_{cr} y la matriz fundamental F_{cr} (ecuaciones (3.37) y (3.49))

$$ {}^c\tilde{p} = H_{cr}\,{}^r\tilde{p}, \qquad {}^c\tilde{p}^T F_{cr} = {}^r\tilde{p} \tag{3.57} $$

   con un esquema RANSAC. Para hacer homogéneo el procedimiento de ambos modelos, el número de iteraciones esta prefijado y es el mismo junto con los puntos que se utilizaran en cada iteración, ocho para la matriz fundamental y cuatro para la homografía. En cada iteración se calcula un "marcador" S_M para cada modelo M, H para la homografía y F para la matriz fundamental

$$ \begin{aligned} S_M &= \sum_i (\rho_M(d^2_{cr}({}^c\tilde{p}^i, {}^r\tilde{p}^i, M)) + \rho_M(d^2_{rc}({}^c\tilde{p}^i, {}^c\tilde{p}^i, M) \\ \rho_M(d^2) &= \begin{cases} \Gamma - d^2 & \text{si } d^2 < T_M \\ 0 & \text{si } d^2 \geq T_M \end{cases} \end{aligned} \tag{3.58} $$

   donde d^2_{cr} y d^2_{rc} son los errores de transferencia simétrica de un cuadro a otro. T_M es el umbral de rechazo de *outliers*, $T_H = 5.99$, $T_F = 3.84$ suponiendo una desviación

estándar de un pixel en el error de medición. Γ está definido igual a T_H tal que ambos marcadores de los modelos equivalen para el mismo d en sus regiones de *inliers*, de nuevo para hacer al proceso homogéneo. Se guarda la homografía y la matriz fundamental con mayor marcador.

3. Selección del modelo. Si la escena es planar, cercanamente planar o existe un bajo paralaje, la representación es una homografía. Sin embargo, una matriz fundamental puede también encontrarse aún si el problema no está bien restringido y cualquier intento de recuperar el movimiento a partir de la matriz fundamental puede llevar a resultados erróneos. Se debe seleccionar la homografía como el método de reconstrucción que inicializara correctamente desde un plano o detectará el caso de bajo paralaje y rechazara la inicialización. Por otro lado, una escena no planar con suficiente paralaje sólo puede explicarse con la matriz fundamental, pero también se puede encontrar una homografía describiendo un subconjunto de acoplamientos si se encuentran en un plano o tienen bajo paralaje, están muy lejos. En este caso se debe seleccionar una matriz fundamental. Una forma heurística robusta se calcula como

$$R_H = \frac{S_H}{S_H + S_F} \tag{3.59}$$

y se selecciona la homografía si $R_H > 0.45$, donde se captura de manera adecuada los casos planar y de bajo paralaje. De otra forma se selecciona la matriz fundamental.

4. Recuperación del movimiento y de su estructura. Una vez seleccionado un modelo, se recuperan las hipótesis de movimiento asociadas. En el caso de la homografía se recuperan ocho hipótesis de movimiento utilizando el método de Faugeras [25]. Se propone triangular directamente las 8 soluciones, verificando si existe una solución con la mayoría de los puntos vistos con paralaje, en frente de ambas cámaras y con error de reproyección bajo. Si no existe claramente una solución ganadora no hay inicialización y se continua desde el primer paso. Ésta técnica hace de la inicialización robusta a bajo paralaje.

   En el caso de la matriz fundamental, se convierte en una esencial usando la matriz de calibración K ecuación (3.48):

$$E_{rc} = K^T F_{rc} K \tag{3.60}$$

   luego se recuperan cuatro hipótesis de movimiento con el método de descomposición del valor singular desarrollado en [26]. Se triangulan las cuatro soluciones y se selecciona la reconstrucción por homografía.

5. *Bundle Adjustment*. Finalmente se realiza un completo BA para refinar la reconstrucción inicial.

3.4.4. Seguimiento

El seguimiento se realiza mediante los pasos de extracción de características, estimación de la *pose* con el cuadro previo o vía relocalización, seguimiento del mapa local e inserción de un nuevo cuadro clave. En ésta sección se describen los pasos que el hilo de seguimiento (Figura 3.15) realiza en cada cuadro de la cámara.

Extracción ORB

Se extraen esquinas en ocho niveles de escala con un factor de escala de 1.2. Para asegurar una distribución homogénea, se divide cada nivel de escala en una cuadrícula, tratando de extraer al menos cinco esquinas por celda. Después se detectan las esquinas en cada célula, adaptando el umbral del detector si no se encuentran suficientes esquinas. La cantidad de esquinas retenidas por célula también se adapta si algunas celdas no contienen esquinas, textura o bajo contraste. La orientación y el descriptor ORB se calculan entonces en las esquinas FAST retenidas. El descriptor ORB se utiliza en todas las características acopladas.

Estimación de la pose inicial a partir del cuadro previo

Si el seguimiento fue satisfactorio para el último cuadro, se utiliza un modelo de velocidad de movimiento constante para predecir la *pose* de la cámara y realizar una búsqueda guiada de los puntos del mapa observados en el último cuadro. Si no se encontraron suficientes coincidencias, es decir, el modelo de movimiento es claramente violado, se utiliza una búsqueda más amplia de los puntos del mapa alrededor de su posición en el último cuadro. La *pose* se optimiza con las correspondencias encontradas.

Estimación de la pose inicial vía relocalización global

Si se pierde el seguimiento, se convierte el cuadro en un paquete de palabras y se consulta la base de datos de reconocimiento para los candidatos a cuadro clave para la relocalización global. Se calculan correspondencias con ORB asociadas a puntos del mapa en cada cuadro clave. A continuación se realizan alternativamente iteraciones RANSAC para cada cuadro clave y se trata de encontrar una *pose* de la cámara utilizando el algoritmo PnP [27]. Si se encuentra una *pose* de la cámara con suficientes *inliers*, se optimiza y se realiza una búsqueda guiada de más acoplamientos con los puntos del mapa del cuadro clave candidato. Finalmente, la *pose* de la cámara se optimiza de nuevo y si se admite con suficientes *inliers*, el procedimiento de seguimiento continúa.

Seguimiento del mapa local

Una vez que se tiene una estimación de la *pose* de la cámara y un conjunto inicial de acoplamiento de características, se puede proyectar el mapa en el cuadro y buscar más correspondencias de puntos del mapa. Para limitar la complejidad en grandes mapas, sólo se proyecta un mapa local. El mapa local contiene el conjunto de cuadros clave K_1, que comparten puntos del mapa con el cuadro actual y un conjunto K_2 cercanos a los cuadros clave K_1 de la gráfica de covisibilidad. El mapa local también tiene un cuadro clave de referencia $K_{ref} \in K_1$ que comparte la mayoría de los puntos del mapa con el cuadro actual. Ahora cada punto del mapa visto en K_1 y K_2 se busca en el cuadro actual de la siguiente manera:

1. Se calcula la proyección del punto del mapa $\mathbf{x}$ en el cuadro actual. Se descarta si se coloca fuera de los límites de la imagen.

2. Se calcula el ángulo entre el rayo de visión actual $\mathbf{v}$ y la dirección de visión media del punto del mapa $\mathbf{n}$. Se descartan si $\mathbf{v} \cdot \mathbf{n} < \cos(60)$.

3. Se calcula la distancia d desde el punto del mapa hasta el centro de la cámara. Se descarta si está fuera de la región de invariancia de escala del punto de mapa $d \notin [d_{min}, d_{max}]$.

4. Se calcula la escala en el cuadro con la relación d/d_{min}.

5. Se compara el descriptor representativo D del punto del mapa con las características ORB aún no acopladas en el cuadro, en la escala predicha y cerca de $\mathbf{x}$, y se asocia el punto del mapa con la mejor coincidencia.

La *pose* de la cámara finalmente se optimiza con todos los puntos del mapa encontrados en el cuadro.

Decisión para un nuevo cuadro clave

El último paso es decidir si el cuadro actual se genera como un nuevo cuadro clave. Dado que existe un mecanismo en el mapeo local para eliminar cuadros clave redundantes, se intenta insertar cuadros clave lo más rápido posible, ya que esto hace que el seguimiento sea más robusto a los movimientos de cámara desafiantes como las rotaciones. Para insertar un nuevo cuadro clave se deben cumplir las condiciones siguientes:

1. Más de 20 cuadros clave deben haber pasado desde la ultima relocalización global.

2. El mapeo local está inactivo o más de 20 cuadros han pasado desde la última inserción de un cuadro clave.

3. El cuadro actual siguió al menos 50 puntos.

4. El cuadro actual siguió menos del 90 % de puntos que el K_{ref}.

La condición 1 asegura una buena relocalización y la condición 3 un buen seguimiento. Si se inserta un cuadro clave cuando la asignación local está ocupada (segunda parte de la condición 2), se envía una señal para detener el *BA local* para poder procesar lo antes posible el nuevo cuadro clave. Además se impone un criterio de un cambio visual mínimo (condición 4).

3.4.5. Mapeo local

El mapeo se realiza mediante los pasos de inserción de un nuevo cuadro clave, eliminación de puntos, creación de nuevos puntos, BA y eliminación de cuadros clave redundantes. En esta sección se describen los pasos realizados por el mapeo local (Figura 3.15) con cada nuevo cuadro clave K_i.

Inserción de un cuadro clave

Primero se actualiza la gráfica de covisibilidad, añadiendo un nuevo nodo para K_i y actualizando los bordes resultantes de los puntos del mapa compartidos con otros cuadros clave. A continuación se actualiza el árbol de expansión que vincula K_i con la mayoría de puntos en común. Después se calcula la representación de los paquetes de palabras del cuadro clave, que ayudará en la asociación de datos para la triangulación de nuevos puntos.

Puntos recientes del mapa descartados

Los puntos del mapa deben pasar una prueba restrictiva durante los primeros tres cuadros clave después de la creación, lo que garantiza que puedan seguirse y no triangularse erróneamente, es decir, el punto debe cumplir estas dos condiciones:

1. El seguimiento debe encontrar el punto en más del 25 % de los cuadros en los que se prevé que sea visibles.

2. Si más de un cuadro clave ha pasado de la creación de puntos del mapa, debe observarse desde al menos tres cuadros clave.

Una vez que un punto del mapa ha pasado esta prueba, sólo se puede eliminar si en cualquier momento se observa desde menos de tres cuadros clave. Esto puede ocurrir cuando los cuadros clave son eliminados y cuando el *BA local* descarta outliers. Esta política hace que el mapa contenga muy pocos outliers.

Creación de un nuevo punto en el mapa

Los nuevos puntos del mapa se crean triangulando características ORB a partir de cuadros clave conectados K_c en la gráfica de covisibilidad. Para cada característica ORB desacoplada en K_i, se busca una coincidencia con otro punto sin acoplar en otro cuadro clave. Se descartan aquellos acoplamientos que no cumplen con la restricción epipolar. Los pares de características ORB están triangulados y para aceptar nuevos puntos, se comprueba la profundidad positiva en ambas cámaras, el paralaje, el error de reproyección y la consistencia de la escala. Inicialmente se observa un punto del mapa a partir de dos cuadros clave, pero puede ser acoplado en otros, por lo que se proyecta en el resto de los cuadros clave conectados y se buscan las correspondencias.

Bundle Adjustment local

El *BA local* optimiza el cuadro clave K_i, todos los cuadros clave conectados a él en la gráfica de covisibilidad K_c y todos los puntos del mapa vistos por esos cuadros clave. Todos los demás cuadros clave que ven esos puntos pero no están conectados al procesado actualmente se incluyen en la optimización pero permanecen fijos. Las observaciones marcadas como outliers se descartan en el centro y al final de la optimización.

Eliminación de un cuadro clave local

Con el fin de mantener una reconstrucción compacta, el mapeo local intenta detectar cuadros clave redundantes y eliminarlos. Esto es beneficioso a medida que la complejidad de *BA* crece con el número de cuadros clave, pero también porque permite la operación de forma continua en el mismo entorno, ya que el número de cuadros clave no crecerá sin límites, a menos que cambie el contenido visual en la escena. Se descartan todos los cuadros clave de K_c en donde el 90 % de los puntos del mapa se han visto en al menos otros tres cuadros clave en la misma escala o en una más fina.

Capítulo 4

Resultados y Conclusiones

4.1. Hardware y software implementado

El software utilizado para las simulaciones y experimentos se describe a continuación.

- **Sistema Operativo (SO).** El SO utilizado es Ubuntu 16.02. Se implementó este trabajo en este SO debido a que muchos algoritmos (ORB, RANSAC, etc) y librerías (*OpenCV*,) están desarrolladas en el.

- **Programas y Librerias.** El leguaje de programación utilizado es C, debido a su versatilidad para implementar diversas plataformas y equipos (Windows, Linux, Computadora de escritorio, computadoras de placa como Raspberry u Odroid). A continuación se enumeran las librerías utilizadas:
  - *OpenCV.* Es una librería libre de visión artificial multiplataforma que contiene diversos algorítmos utilizados en este trabajo.
  - *Pangolin.* Herramienta de visualización e interfaz de usuario.
  - *Eigen3.* Librería para álgebra lineal, contiene algoritmos para operaciones entre matrices, vectores y métodos numéricos.
  - *DBoW2.* Biblioteca para indexar y convertir imágenes en una representación de paquetes de palabras. Implementa un árbol jerárquico para aproximar a los vecinos más cercanos en el espacio de la característica de la imagen y crear un vocabulario visual. DBoW2 también implementa una base de datos de imágenes con archivos invertidos y directos para indexar imágenes y permitir consultas rápidas y comparaciones de características [28]. - *ORB-SLAM2* Librería en tiempo real de SLAM para cámaras monoculares, estéreo y RGB-D [29].

El hardware utilizado para los experimentos se describe a continuación.

- **Procesador**. Intel(R) core i7 920 @2.67Hz.

- **Sistema Optitrak**. Con este sistema de cámaras fijas se obtienen la posición del cuatrirotor.
- **Cámara** Cámara Estenopeica de 8 Megapixeles de 30 cuadros por segundo implementada para la obtención de características del entorno. Los parámetros calculados en OpenCV son: $f_x = 517.30$, $f_y = 318.64$, $c_x = 318.64$, $c_y = 255.31$, $k_1 = 0.2624$, $k_2 = -0.953$, $k_3 = 1.1633$.
- **Cuatrirotor** Los parámetros del cuatrirotor utilizado son los siguientes:

Parámetro	Valor
$m(Kg)$	1.120
$g(m/s^2)$	9.81
$l(m))$	0.214
$\rho(Kg/m^3)$	1
$I_{xx}(Kgm^2)$	0.002
$I_{yy}(Kgm^2)$	0.002
$I_{zz}(Kgm^2)$	0.004

Cuadro 4.1: Parámetros del cuatrirotor.

- **Módulos de comunicación** Se implementaron módulos WiFly para la comunicación inalámbrica por medio del protocolo TCP/IP entre las computadoras y el cuatrirotor.

4.2. Implementación del control predictivo generalizado (GPC)

El controlador propuesto se evalúo utilizando simulaciones numéricas en el software Matlab. Para el desempeño mostrado se utilizaron parámetros de horizonte de predicción $N_p = 60$, horizonte de control $N_u = 8$, $R = diag(19, 19, 1)$ y $Q = diag(1.2, 1.2, 1)$. la trayectoria de referencia está dada como $x_{ref} = \cos(\omega t)\ m$, $y_{ref} = \sin(\omega t)\ m$ y $z = -0.5\ m$.

La elección de los parámetros del GPC se realizó de manera empírica tomando en cuenta criterios importantes. El horizonte de predicción suele ayudar en la rapidez con que se llega a la referencia sin embargo implica un incremento en gasto computacional ya que debe ejecutarse la multiplicación de una matriz $N_p \times N_p$. El horizonte de control entonces es pequeño y ayuda a reducir gasto computacional ya que su implementación reduce el horizonte de predicción. Las matrices Q y R le dan más peso ya sea al error o al

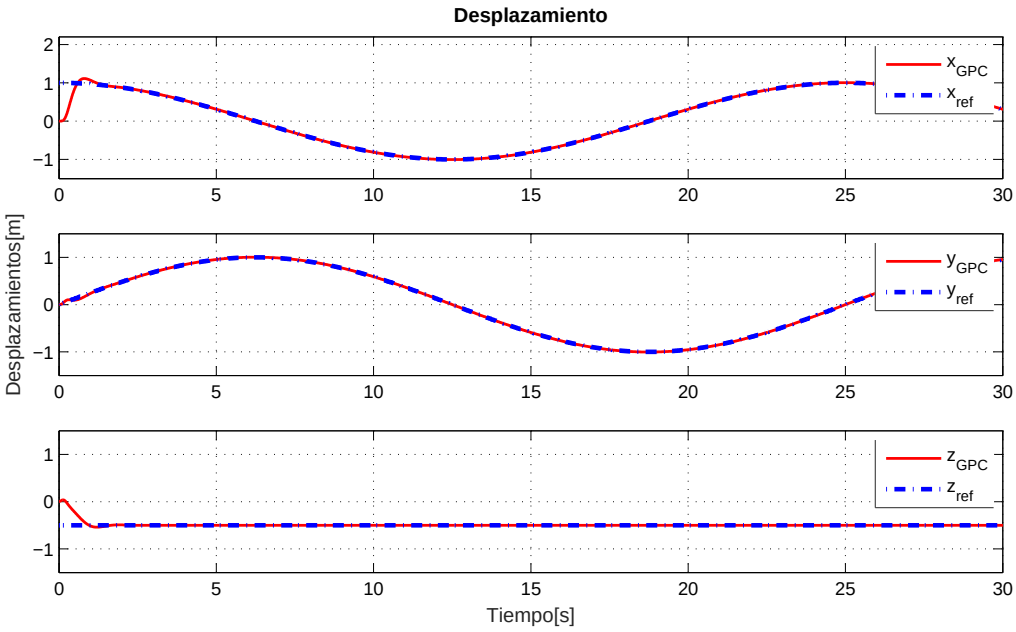

Figura 4.1: Desplazamiento. En las gráficas se puede observar que el seguimiento de la trayectoria satisfactorio para los tres ejes se consigue en menos de tres segundos y con un error imperceptible.

control. En este caso se decidió darle más ponderación al error, lo cual implica no exigirle tanto a las señales de control.

La Figura 4.1 muestra el seguimiento de la trayectoria del cuatrirotor a lo largo de los ejes x, y y z utilizando el controlador predictivo. En línea punteada se muestran las señales de referencia.

En la Figura 4.2 se presenta el error de seguimiento de la trayectoria en los ejes x, y y z donde se observa la convergencia del error a cero.

La Figura 4.3 muestran el error de seguimiento de los ángulos de orientación del cuatrirotor alrededor de los ejes x_b, y_b y z_b

En la Figura 4.4 se presenta un esquema general utilizado en el laboratorio para realizar el experimento. A continuación se da una explicación:

- El sistema de cámaras optitrack obtiene las posiciones del cuatrirotor y las envía a una computadora.
- La computadora ejecuta el algorítmo del GPC y como salida envía los controles virtuales u_x, u_y y u_z, definidos a partir de la ecuación (2.17), mediante un modem.
- El DSP, abordo del cuatrirotor, recibe las señales mediante un dispositivo inalámbrico $WiFly$. También recibe por protocolo RS-232 la orientación y velocidad angular

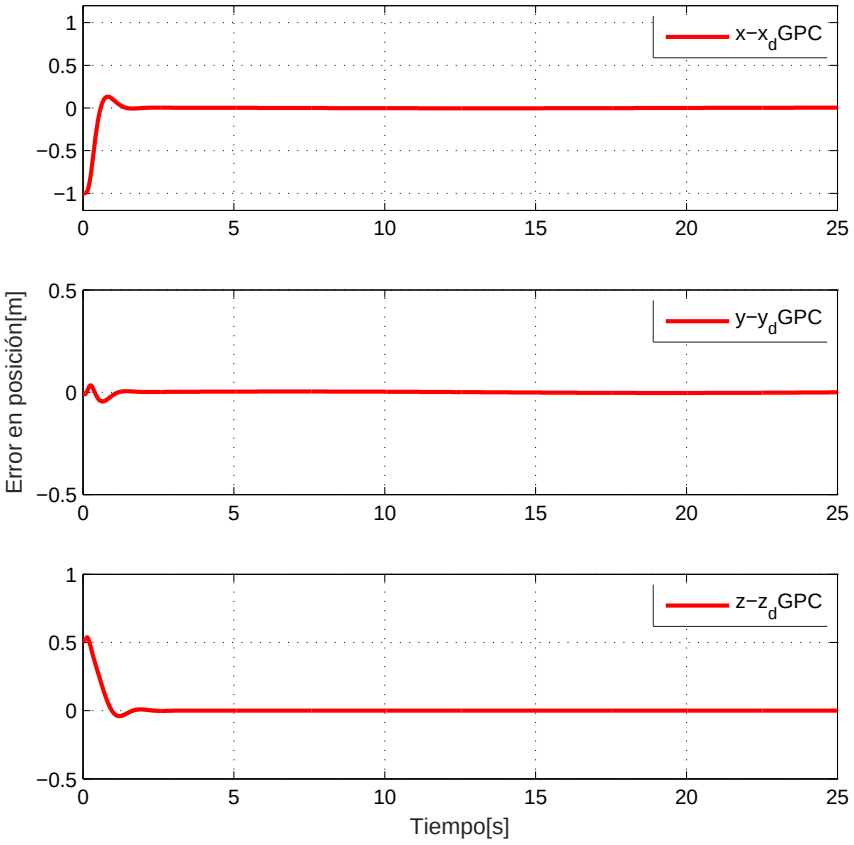

Figura 4.2: Error de desplazamiento. Éste error es prácticamente cero para los tres ejes después de los cinco segundos.

del cuatrirotor de un sistema de referencia de orientación y rumbo (SROR).

- El DSP utiliza los controles virtuales y señales de la IMU para realizar el calculo del control de orientación.
- Las señales de control se traducen en PWMs que son enviados a los motores.

Finalmente, se muestran en la Figuras 4.5 y 4.6 gráficas experimentales obtenidas en el laboratorio. Cabe mencionar que en éste experimento se agregaron dos *waypoint* como referencia a la trayectoria circular, en las gráficas de referencia se pueden notar.

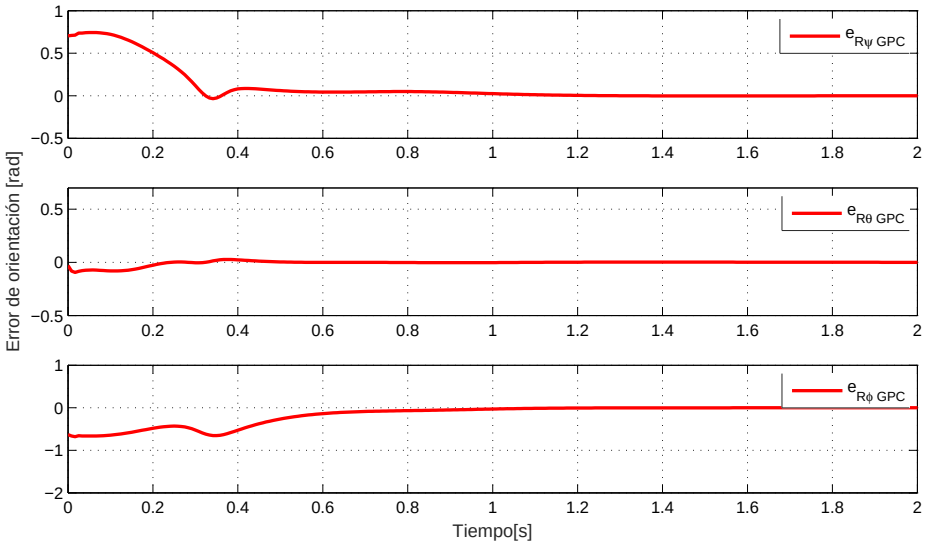

Figura 4.3: Error de orientación. Los errores en orientación se hacen prácticamente cero en $t = 1.2s$. Se verifica que se controló mas rápido la dinámica rotacional que la traslacional.

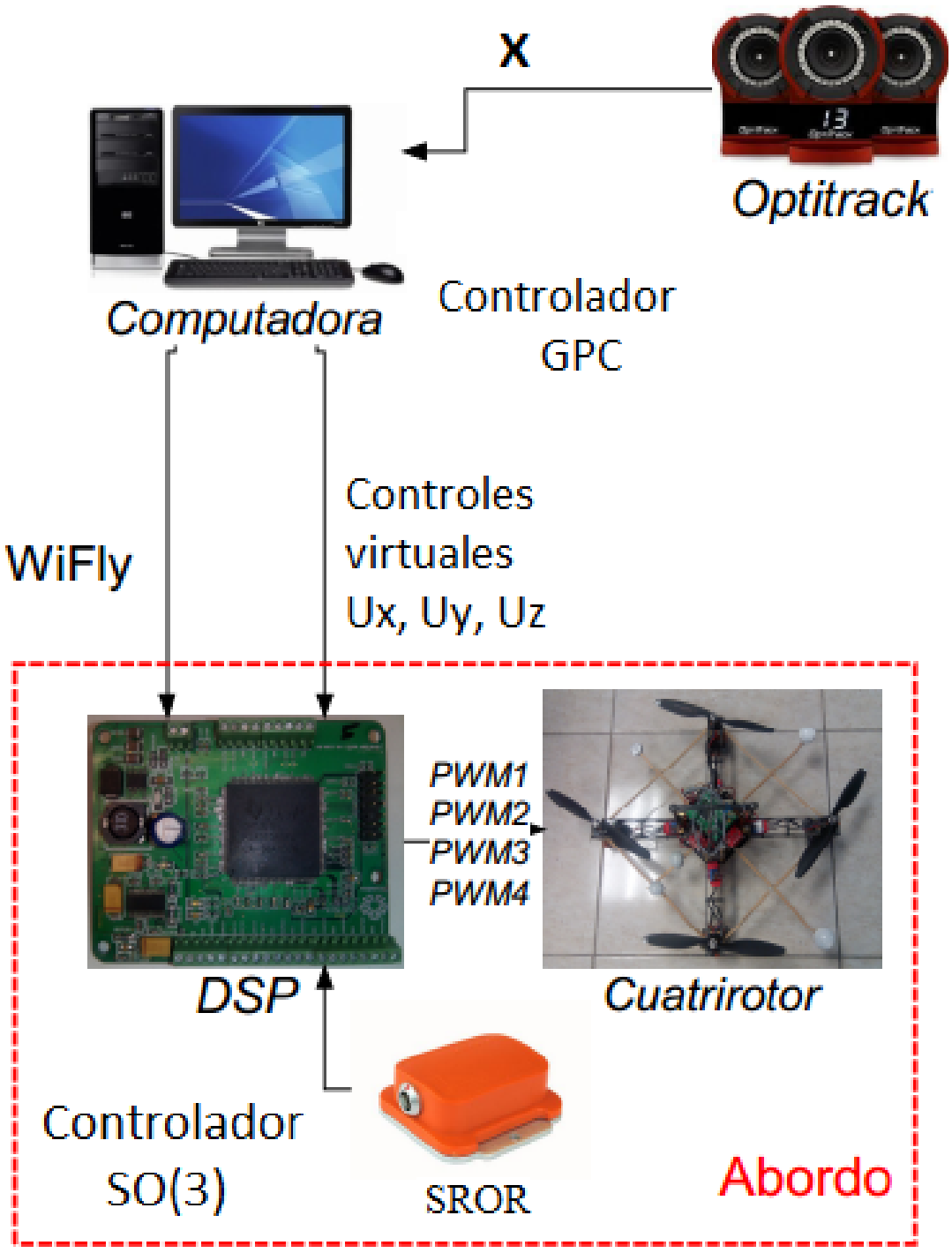

Figura 4.4: Esquema general de conexión para el experimento.

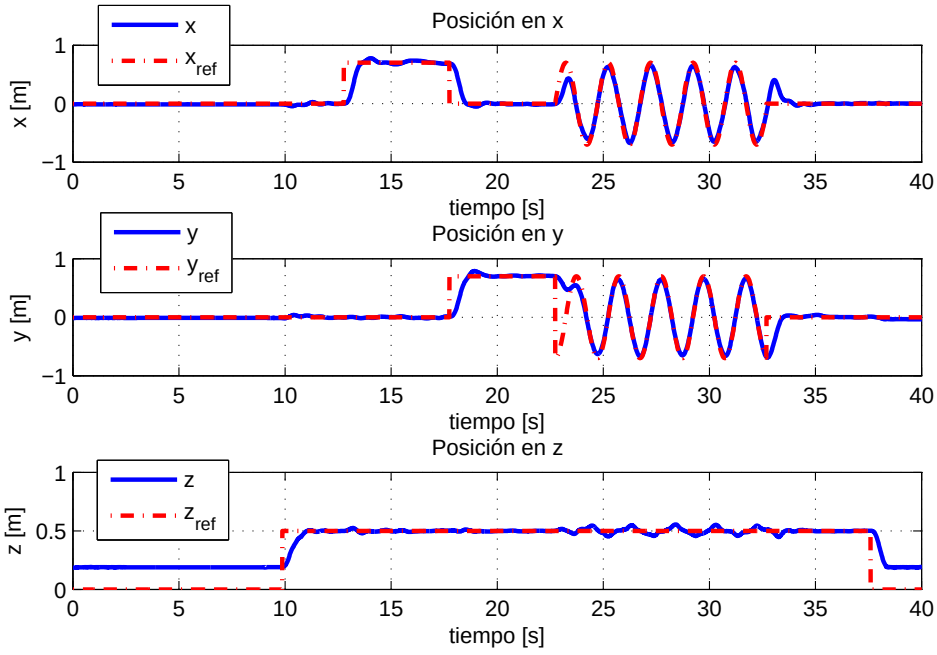

Figura 4.5: Posición en x, y y x del cuatrirrotor.

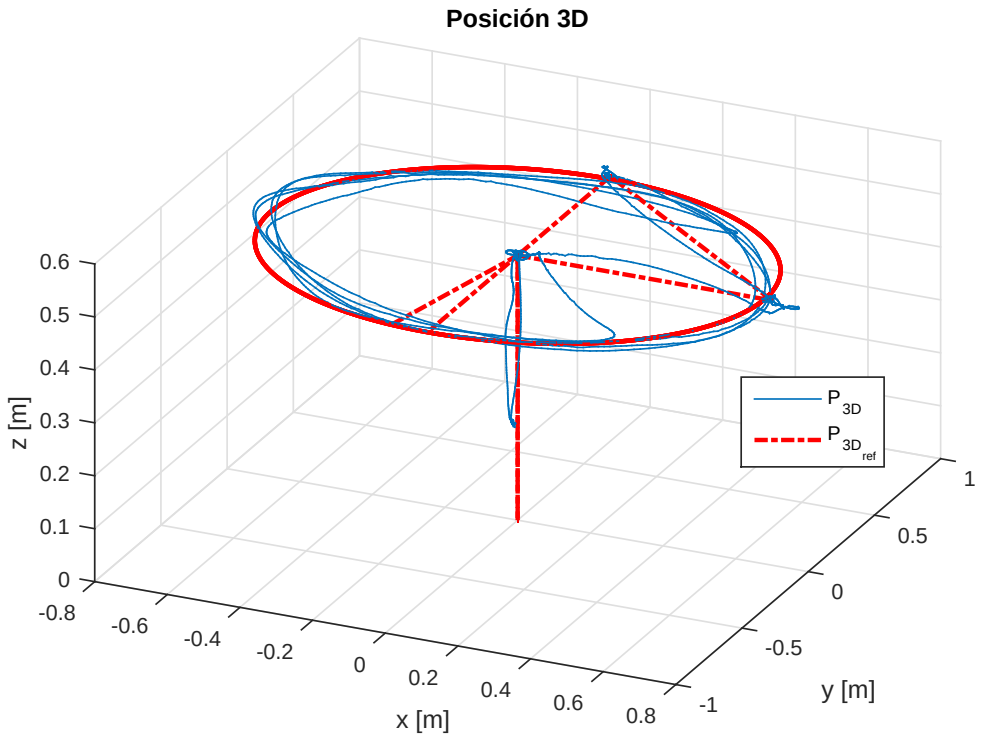

Figura 4.6: Posición en 3D.

4.3. Implementación del sistema ORB-SLAM

El sistema ORB-SLAM se implementó en el Sistema Operativo Ubuntu 14.2. con una frecuencia de muestreo aproximada de 10 cuadros por segundo. La Fig. 4.7 muestra un ejemplo de la implementación del SLAM con una cámara móvil.

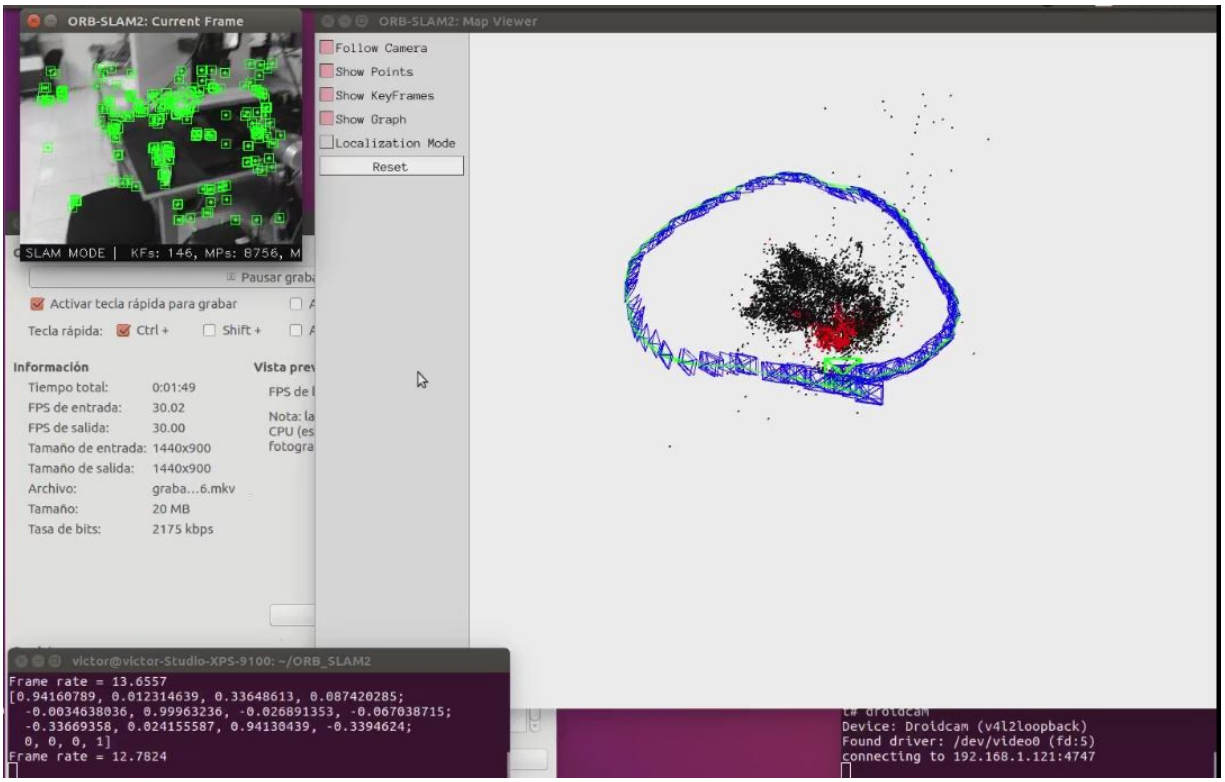

Figura 4.7: ORB-SLAM ejecutándose en Ubuntu. Este ejemplo se hizo en el laboratorio de la sección de Mecatrónica.

En la Fig. 4.8 se muestra el diagrama de conexiones del hardware utilizado para la implementación del sistema ORB-SLAM.

El ordenador 1 recibe las imágenes de la cámara montada en el vehículo y ejecuta el sistema ORB-SLAM. Como salidas se obtiene la matriz homogénea de posición y orientación. A partir de esta matriz se calcula el algoritmo GPC para la posición en el eje y y se envía un control virtual u_y.

El ordenador 2 recibe la posición del vehículo desde el sistema Optitrack y ejecuta el algoritmo GPC para la posición en los ejes x y y. Se envían como salida los controles virtuales u_x y u_z. La razón de usar dos computadoras es que el software del sistema Optitrack sólo existe para el sistema operativo de *Windows* y el sistema de visión está desarrollado en *Linux*. El DSP recibe los tres controles virtuales u_x, u_y y u_z que actuaran

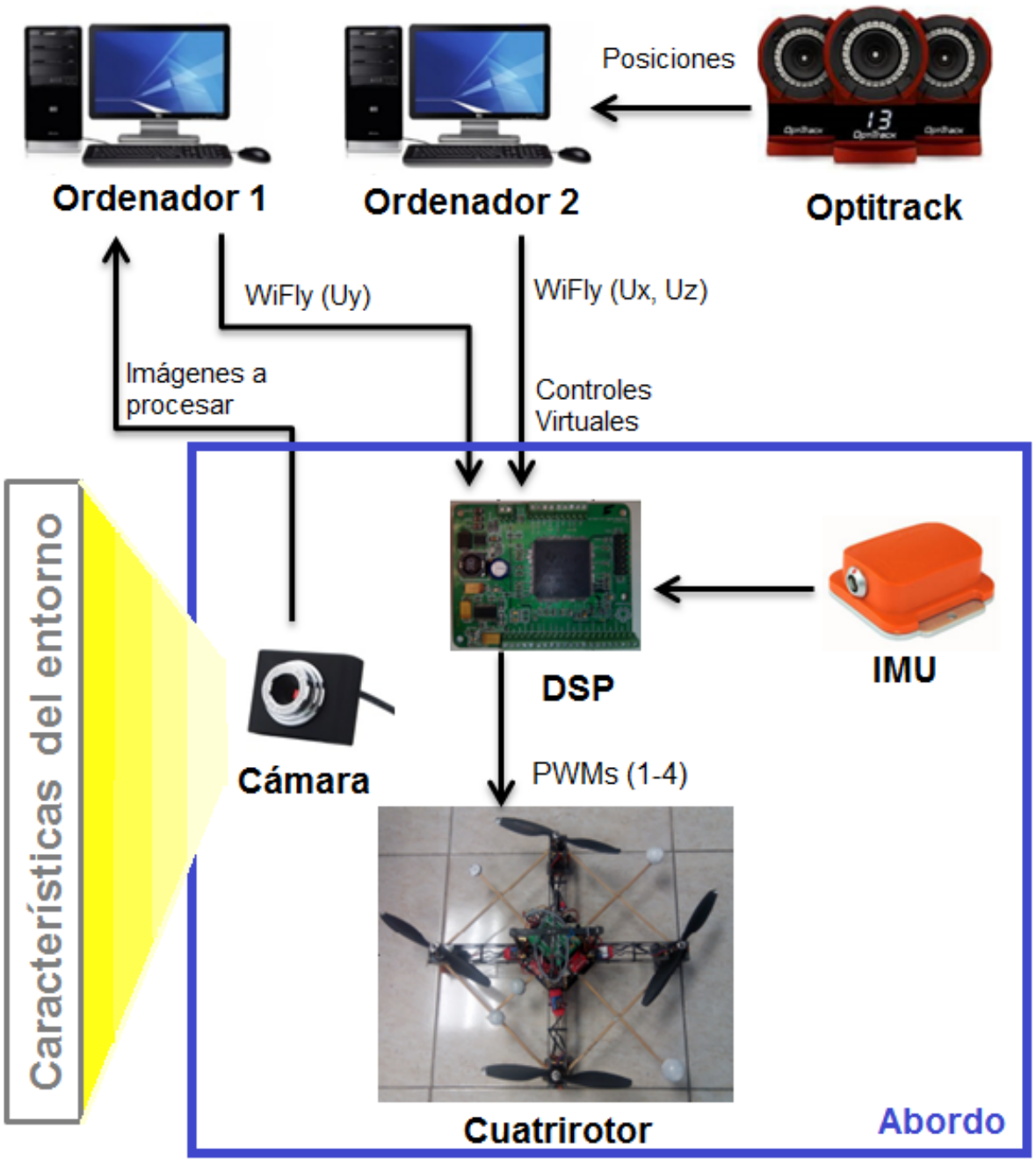

Figura 4.8: Diagrama de conexiones de Hardware.

sobre la dinámica traslacional de vehículo. Finalmente, con ayuda del SROR y los controles virtuales ya mencionados, se implementa un control de orientación en el espacio $SO(3)$ como se describe en [13].

En la Figura 4.9, se muestra un diagrama de flujo del programa ejecutado en la computadora 1. Se ejecuta el sistema de SLAM seguido del GPC y se envían los datos de forma serial a un transmisor inalámbrico. Este proceso se ejecuta para cada cuadro dado por la cámara.

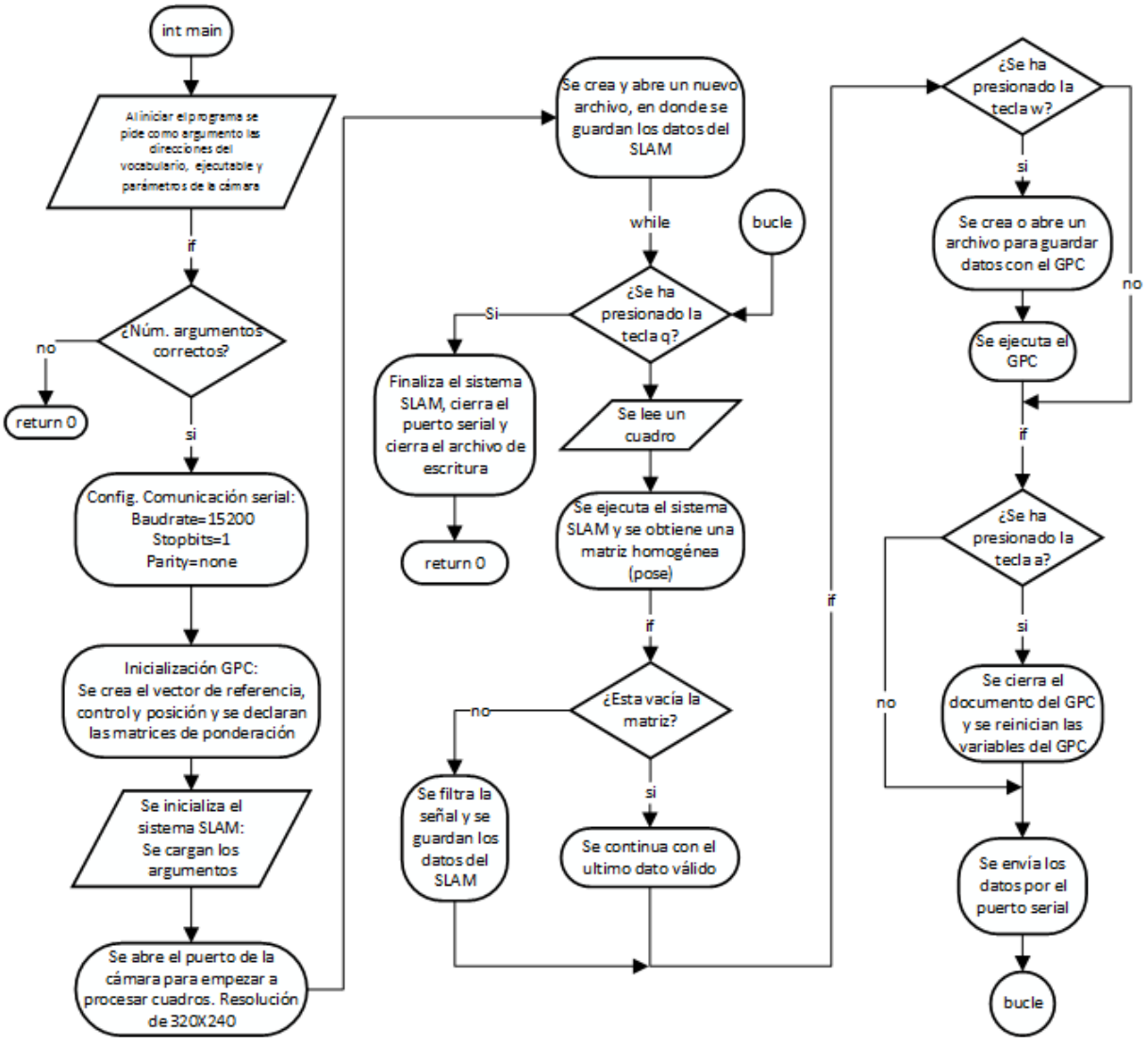

Figura 4.9: Diagrama de flujo ejecutado en el ordenador 1.

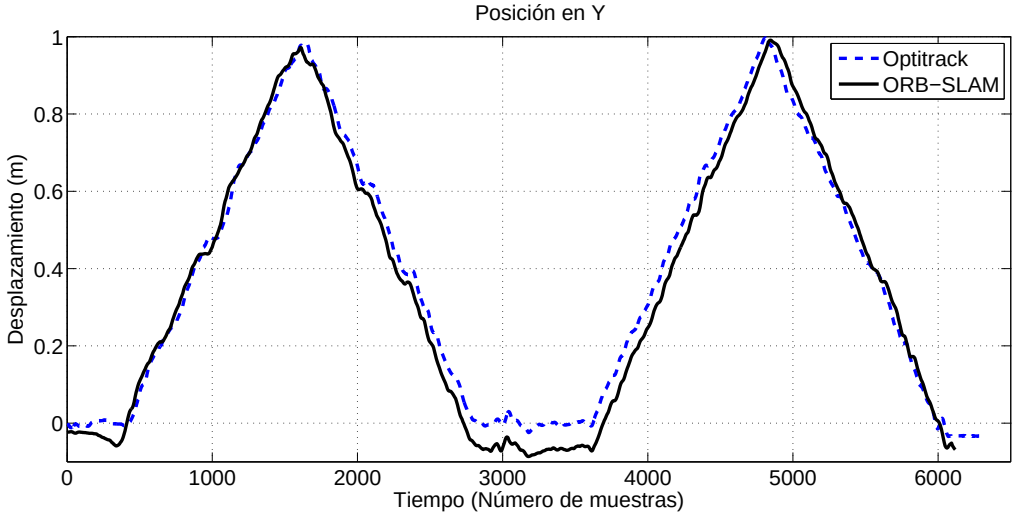

Figura 4.10: En ésta gráfica se observa que además de sensar la posición con el algoritmo de visión de manera similar al del Optitrack, la trayectoria medida no se pierde y se repite.

Se realizaron dos experimentos utilizando el sistema ORB-SLAM. Para probar el sistema de visión, se realizó un vuelo controlado por el algoritmo GPC cuyas medidas de posición fueron realizadas mediante el Optitrack. El sistema ORB-SLAM se implementó y simplemente se compararon sus mediciones en el eje y con las del Optitrack. En la Fig. 4.10 se muestra una gráfica comparativa entre los dos sistemas.

El segundo experimento consistió en controlar únicamente la posición en el eje y implementando el sistema ORB-SLAM. El resto de las variables a controlar se realizaron mediante el Optitrack. A continuación se muestra en la figura 4.11 una gráfica del desplazamiento en y y en la figura 4.12 una de la señal de control u_y generada a partir del sistema ORB-SLAM.

Este experimento consistió en un vuelo estacionario (*hover*), por lo tanto la referencia de y es cero. Si bien el error llega a ser de hasta 30 cm, este no se dispara. Este error se puede reducir mejorando el tipo de cámara utilizada (con mayor número de cuadros por segundo o estéreo), implementando un estabilizador en la cámara, para reducir las vibraciones, implementar una tarjeta abordo que ejecute el SLAM visual, esto evita la comunicación inalámbrica.

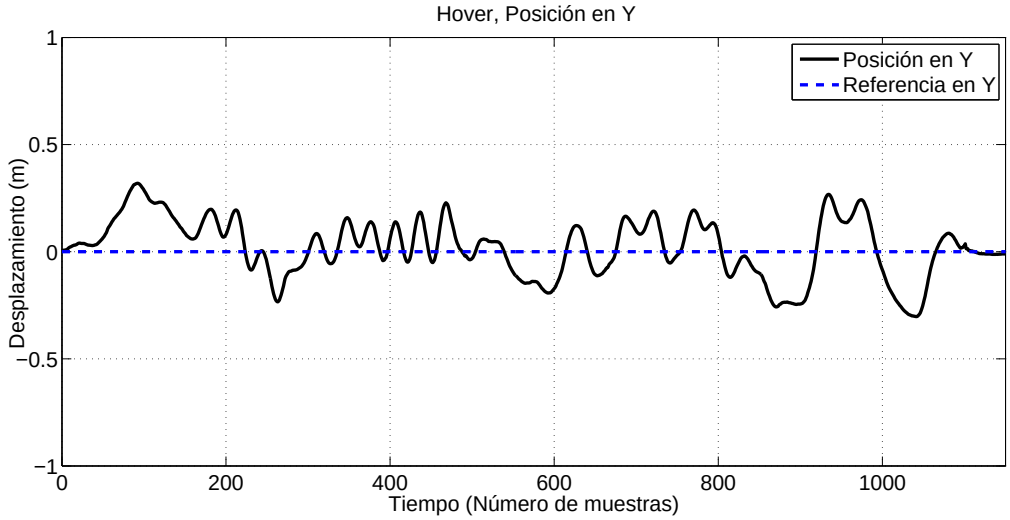

Figura 4.11: Solo se controló la variable y utilizando el sistema ORB-SLAM, dado que pueden ser peligrosos los experimentos para el vehículo si se implementan mas variables.

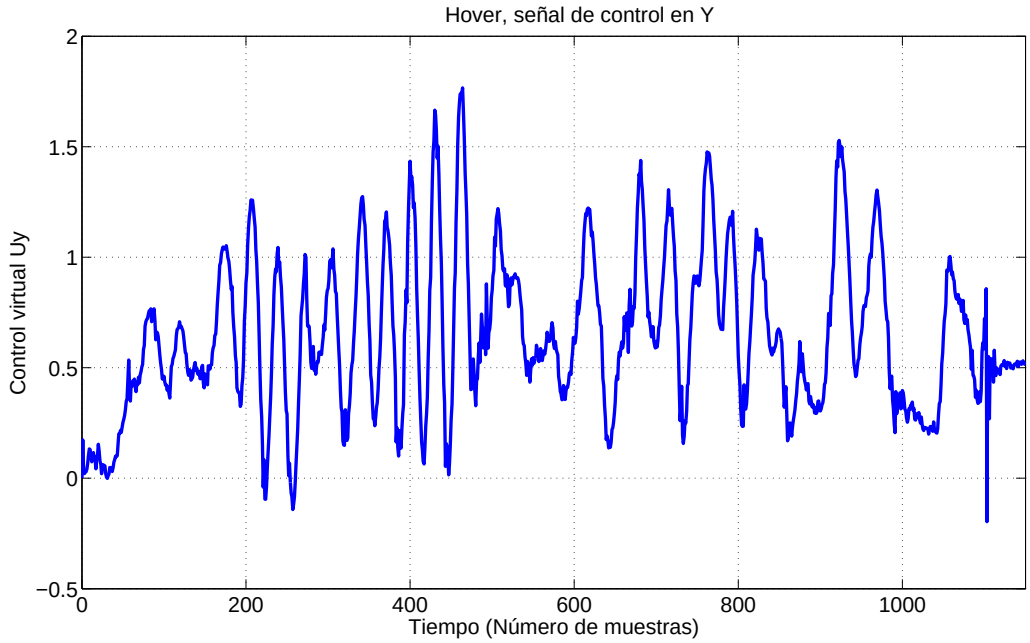

Figura 4.12: La señal de control virtual en y se envía al DSP abordo.

4.4. Conclusiones y trabajo futuro

En este trabajo de tesis se presentó la implementación de un método para la localización de un cuatrirotor y la construcción del mapa de su entorno a partir de imágenes obtenidas por una cámara (SLAM visual). Además, como control de posición en el cuatrirotor, se implementó un controlador predictivo generalizado (GPC).

La estrategia de control con el (GPC) se logró implementar de manera conjunta con el sistema SLAM visual (con el que se recupero la posición del cuatrirotor) para el control del cuatrirotor en un eje. Aunque el sistema de SLAM visual puede implementarse en los tres ejes cartesianos, puede ser peligroso para el cuatrirotor navegar unicamente con visión, ya que el SLAM visual en cualquier momento puede no encontrar suficientes acoplamientos de características y no enviar posiciones al controlador.

Una ventaja de la implentación del SLAM visual es que muchos algoritmos necesarios (ORB, RANSAC, FASTA, BRIEF, etc.) estan desarrollados y disponibles de manera libre en la red. SLAM visual es un método que dota de una mayor autonomía al cuatrirotor, sin embargo, la implementación puede mejorarse con el uso de una cámara estéreo, dadas las características de la geometría epipolar. En el sistema actual una sola cámara procesa dos imágenes consecutivas para formar el plano epipolar, en cambio con dos cámaras se capturan las imágenes de forma paralela, por lo que se obtiene un mejor rendimiento.

En esta tesis ademas se implementa de manera exitosa el controlador GPC tanto en simulación como en pruebas experimentales. Sin embargo dada la cantidad de memoria que necesita el controlador, se tuvo que programar en una computadora de escritorio en lugar de la tarjeta abordo. Esto se debe a los horizontes de predicción y control, ya que al aumentarlos demasiado, se aumentan las matrices de ponderación y por lo tanto se requieren muchas localidades de memoria para guardar los elementos de estas.

Como trabajo futuro se propone la implementación del sistema con una cámara estéreo para mejorar la estimación. Agregar sensores de profundidad al sistema, por ejemplo un láser, el cual se puede unir a las estimaciones mediante filtros de *Kalman*. Cabe mencionar que puede hacerse una fusión del SLAM con el modelo dinámico mediante filtros de *kalman* para obtener mejor información. Implementar el sistema en una tarjeta abordo. Implementar la detección de bucle, el cual ayuda a mejorar la construcción del mapa, dado que se toman en cuenta las características observadas en tiempos anteriores en la operación del sistema. Y finalmente la implementación del GPC considerando restricciones geométricas.

Bibliografía

[1] Peter Corke. *Robotics, vision and control: fundamental algorithms in MATLAB*, volume 73. Springer, 2011.

[2] Raul Mur-Artal, JMM Montiel, and Juan D Tardós. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[3] Penelope Probert Smith and Penelope Probert Smith. *Active sensors for local planning in mobile robotics*, volume 26. World Scientific, 2001.

[4] Chris Harris. Geometry from visual motion. In *Active vision*, pages 263–284. MIT press, 1993.

[5] HF Durrant-Whyte, MWMG Dissanayake, and PW Gibbens. Toward deployment of large scale simultaneous localisation and map building (slam) systems. In *ROBOTICS RESEARCH-INTERNATIONAL SYMPOSIUM-*, volume 9, pages 161–168, 2000.

[6] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[7] David W Clarke. Advances in model-based predictive control. 1994.

[8] David W Clarke. Application of generalized predictive control to industrial processes. *IEEE Control systems magazine*, 8(2):49–55, 1988.

[9] J Richalet. Industrial applications of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.

[10] Michael Neunert, Cédric de Crousaz, Fadri Furrer, Mina Kamel, Farbod Farshidian, Roland Siegwart, and Jonas Buchli. Fast nonlinear model predictive control for unified trajectory optimization and tracking. In *IEEE/RSJ International Conference on Robotics and Automation (ICRA)*, pages 1–7, Stockholm, Sweden, May 2016.

[11] Manas Mejari, Ankit Guptaand N.M.Singh, and Faruk Kazi. *10TH International Conference on Soft Computing Models in Industrial and Enviromental Applications*. Springer International Publishing, Switzerland, 2015.

[12] David W Clarke, C Mohtadi, and PS Tuffs. Generalized predictive control—part i. the basic algorithm. *Automatica*, 23(2):137–148, 1987.

[13] Vásquez Beltrán Marco Augusto. *Tesis: Seguimiento de una referencia visual en un plano con un Cuatrirotor*. PhD thesis, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, México, 2015.

[14] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer-Verlag, Great Britain, 2004.

[15] Lee Taeyoung, Leok Melvin, and Harris McClamroch N. Geometric tracking control of a quadrotor uav on se(3). In *49th IEEE Conference on Decision and Control*, pages 5420–5425, Atlanta, GA USA, December 2010.

[16] Jean-Yves Bouguet. Camera calibration toolbox for matlab. 2004.

[17] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999.

[18] Samarth Brahmbhatt. *Practical OpenCV*. Apress, 2013.

[19] Josef Sivic, Andrew Zisserman, et al. Video google: A text retrieval approach to object matching in videos. In *iccv*, volume 2, pages 1470–1477, 2003.

[20] Elena Stumm, Christopher Mei, and Simon Lacroix. Probabilistic place recognition with covisibility maps. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4158–4163. IEEE, 2013.

[21] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[22] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[23] Pablo F Alcantarilla and T Solutions. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *IEEE Trans. Patt. Anal. Mach. Intell*, 34(7):1281–1298, 2011.

[24] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

[25] Wei Tan, Haomin Liu, Zilong Dong, Guofeng Zhang, and Hujun Bao. Robust monocular slam in dynamic environments. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 209–218. IEEE, 2013.

[26] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[27] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155, 2009.

[28] Dorian Gálvez-López and J. D. Tardós. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, October 2012.

[29] Montiel J. M. M. Mur-Artal, Raúl and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.