

# Localización de robots móviles utilizando información métrica

Cristian David Cavenio  
cristian.cavenio@gmail.com

21 de octubre de 2015

## 1. Introducción

En la actualidad existen, en el área de investigación, múltiples proyectos que involucran la obtención de datos mediante el uso de cámaras de video. Estos proyectos van desde reconocimiento y seguimiento de objetos hasta generación de planos y mapas a partir de secuencias de imágenes.

La localización y mapeo simultáneos (SLAM por sus siglas en inglés) es una problemática muy común en robótica, dado que admite la utilización de robots para generar planos de una zona de trabajo y ubicarse a si mismo dentro del mismo. Esta operación puede realizarse en diferentes configuraciones, utilizando una o más cámaras y fusionando sensores. Este proyecto se enfoca en el SLAM monocular, el cual requiere de una sola cámara para realizar el algoritmo y obtener un resultado.

Existen múltiples trabajos referidos a SLAM que utilizan diferentes métodos y configuraciones, en particular, luego de una búsqueda de algoritmos libres, se ha seleccionado el trabajo SceneLib2, realizado por Hanme Kim<sup>1</sup> que implementa un SLAM monocular con un filtro extendido de kalman. Esta implementación se corresponde con una actualización de un trabajo presentado por Andrew Davison en el año 2001.

Se procede entonces a realizar pruebas sobre el software elegido y modificarlo para hacerlo útil a la hora de cumplir el objetivo principal a largo plazo, que es el de adicionar una cámara frontal sobre la plataforma QA3-Mini y capturar imágenes que serán procesadas en una estación terrena.

## 2. Instalación y prueba de SceneLib2

El algoritmo SceneLib2 se encuentra disponible en un repositorio git<sup>2</sup>.

Luego de clonar el repositorio, instalar las librerías de dependencias y compilar la aplicación se procede a probarla de dos formas diferentes:

- Probar el software utilizando como entrada al programa un dataset que viene dentro del repositorio justamente para probar el correcto funcionamiento del mismo.
- Probar el software conectando una cámara USB previamente calibrada a la pc y filmando el entorno de trabajo.

### Funcionamiento con dataset de pruebas

Al establecer la entrada del software con el dataset propuesto, el sistema funciona correctamente, generando un mapeo de puntos característicos de las imágenes que se observan.

Una muestra del algoritmo en ejecución puede verse en la figura 1.

### Funcionamiento con cámara USB

Para realizar estas pruebas se utilizó una cámara industrial ImagingSource DFK21BU04. La calibración de esta cámara se realizó mediante el software Camera Calibration 2.0 del proyecto MRPT.

---

<sup>1</sup><https://wp.doc.ic.ac.uk/robotvision/person/hanme-kim/> - Grupo de Robótica y Visión-Departamento de Computación-Imperial College London

<sup>2</sup><https://github.com/hanmekim/SceneLib2>

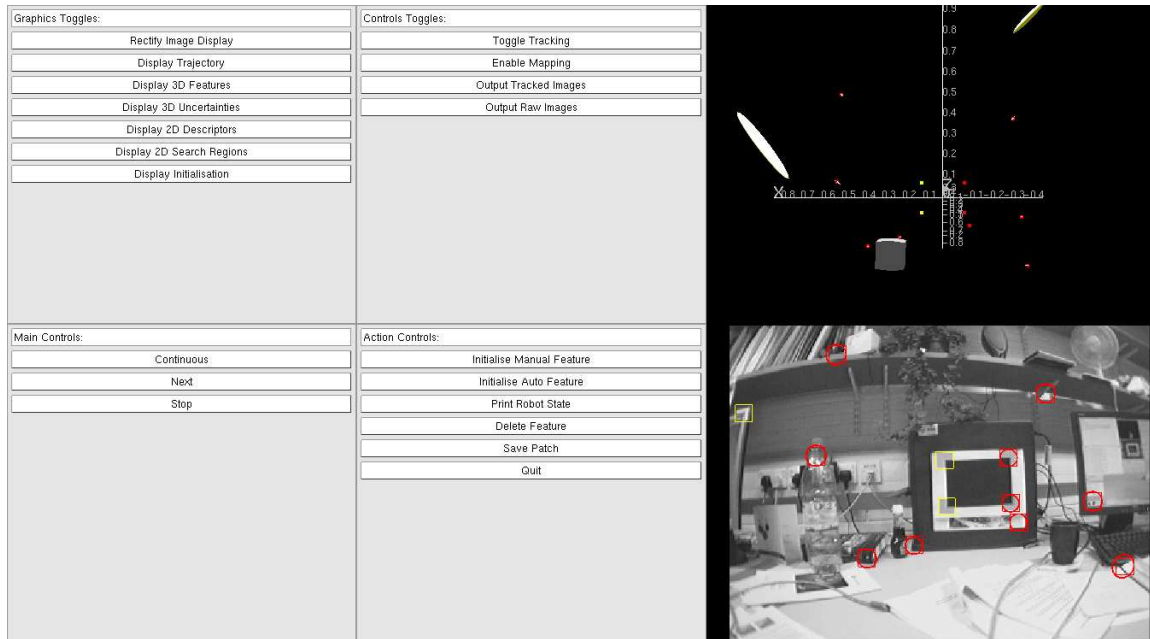


Figura 1: Funcionamiento con dataset propio de SceneLib2.

Una vez realizada la calibración se configuró la entrada de SceneLib2 mediante la cámara USB y el software funcionó correctamente pero sólo si la resolución de la cámara era 320x240 píxeles o menor.

Naturalmente el sistema tiene una inicialización que debió realizarse en la imagen inicial captada por la cámara.

### 3. Estudio de la estructura del software SceneLib2

Una vez corroborado el correcto funcionamiento de la aplicación se procedió a desmenuzar el código fuente, de modo de entender por qué partes está compuesto y cómo funcionan las mismas.

El esquema completo del software se puede observar en la figura 2.

De esta estructura presentada, lo que se desea modificar ampliamente es lo correspondiente a *Motion model* ya que el mismo es un modelo de velocidad constante que no es útil para un cuadrotor dado que realiza movimientos estocásticos con velocidades y aceleraciones variables.

### 4. Cálculo del modelo de movimiento estocástico

El modelo de movimiento busca obtener el estado siguiente del robot, conforme al estado actual  $\mathbf{x}_k$ , la acción de control  $\mathbf{u}_k$  y el ruido  $\mathbf{w}_k$ .

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \quad (1)$$

donde el vector de estado es

$$\mathbf{x} = [\mathbf{p} \quad \mathbf{q} \quad \mathbf{v} \quad \mathbf{b}_a \quad \mathbf{b}_\omega]^T, \quad (2)$$

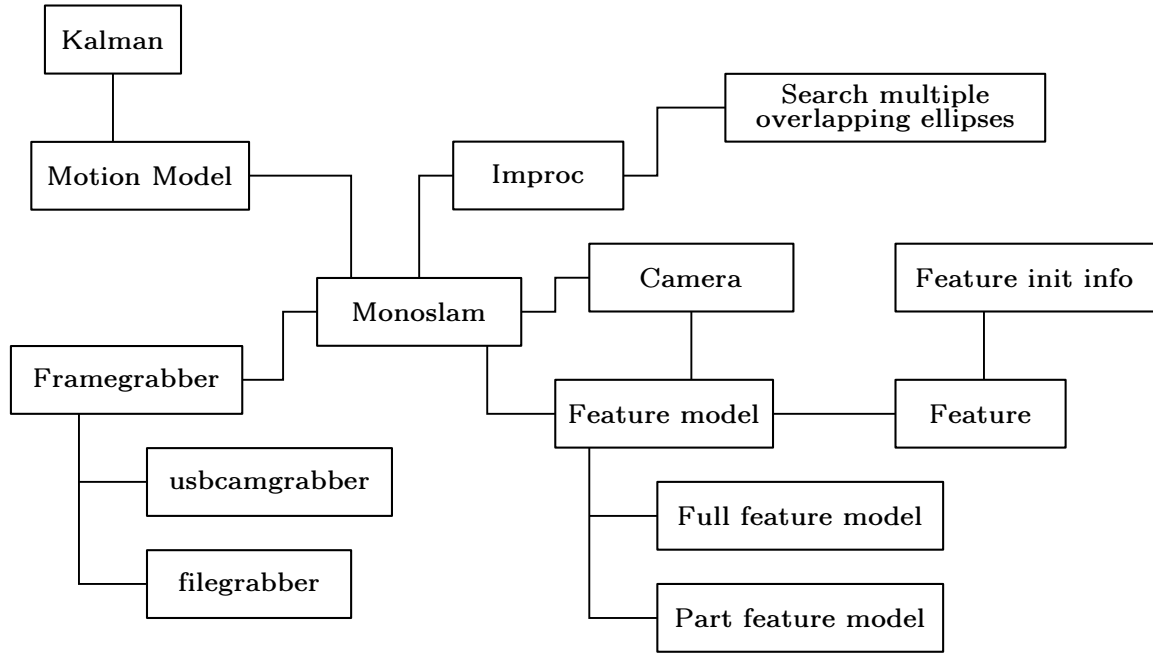


Figura 2: Esquema de software de SceneLib2.

cuyas componentes son

- **p**: vector de posición en el sistema de referencia global
- **q**: cuaternion unitario que representa la orientación respecto al sistema de referencia global
- **v**: velocidad lineal en el sistema de referencia global
- **b<sub>a</sub>**: bias del acelerómetro
- **b<sub>ω</sub>**: bias del giróscopo

Las acciones de control son las mediciones de los sensores inerciales  $\mathbf{u} = [\mathbf{a} \ \boldsymbol{\omega}]^T$ , y el ruido  $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$  es el ruido de dichas mediciones, con

$$\mathbf{Q} = \begin{bmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_\omega^2 \end{bmatrix}. \quad (3)$$

Por lo tanto, el modelo estocástico  $\mathbf{x}_{k+1}$  queda definido por el siguiente juego de ecuaciones:

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{v}_k \Delta t \quad (4)$$

$$\mathbf{q}_{k+1} = \mathbf{q}_k \otimes \mathbf{q}((\boldsymbol{\omega}_k - \mathbf{b}_{\omega,k} + \mathbf{w}_{\omega,k}) \Delta t) \quad (5)$$

$$\mathbf{v}_{k+1} = \mathbf{v}_k + \mathbf{R}(\mathbf{q}_k) (\mathbf{a}_k - \mathbf{b}_{a,k} + \mathbf{w}_{a,k}) \Delta t - \mathbf{g} \Delta t \quad (6)$$

$$\mathbf{b}_{a,k+1} = \mathbf{b}_{a,k} \quad (7)$$

$$\mathbf{b}_{\omega,k+1} = \mathbf{b}_{\omega,k} \quad (8)$$

donde

$$\mathbf{R}(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 + q_3q_0) & 2(q_1q_3 - q_2q_0) \\ 2(q_1q_2 - q_3q_0) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 + q_1q_0) \\ 2(q_1q_3 + q_2q_0) & 2(q_2q_3 - q_1q_0) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad (9)$$

Luego, la propagación de incertidumbre del modelo se realiza mediante la linealización de primer orden de Taylor y resulta

$$\mathbf{P}_{k+1} = \mathbf{F}_k \mathbf{P}_k \mathbf{F}_k^T + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T, \quad (10)$$

donde  $\mathbf{P}_k$  es la matriz de covarianzas del estado y

$$\mathbf{F}_k = \left. \frac{\partial f(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}_k}, \quad \mathbf{G}_k = \left. \frac{\partial f(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{u}} \right|_{\mathbf{u}=\hat{\mathbf{u}}_k}.$$

### Calculo del jacobiano $\mathbf{F}_k$

El jacobiano del modelo de movimiento dado en por (4) a (8) respecto al estado esta dado por

$$\mathbf{F}_k = \frac{\partial f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)}{\partial \mathbf{x}_k} = \begin{bmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 4} & \mathbf{I}_{3 \times 3} \Delta t & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{4 \times 3} & \frac{\partial \mathbf{q}_{k+1}}{\partial \mathbf{q}_k} & \mathbf{0}_{4 \times 3} & \mathbf{0}_{4 \times 3} & \frac{\partial \mathbf{q}_{k+1}}{\partial \mathbf{b}_{\omega, k}} \\ \mathbf{0}_{3 \times 3} & \frac{\partial \mathbf{v}_{k+1}}{\partial \mathbf{q}_k} & \mathbf{I}_{3 \times 3} & \frac{\partial \mathbf{v}_{k+1}}{\partial \mathbf{b}_{a, k}} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 4} & \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 4} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} \end{bmatrix}, \quad (11)$$

donde la submatrices de (11) tienen las siguientes dimensiones

- la submatriz (2, 2) es de  $4 \times 4$ ,
- la submatriz (2, 5) es de  $4 \times 3$ ,
- la submatriz (3, 2) es de  $3 \times 4$ ,
- la submatriz (3, 4) es de  $3 \times 3$ .

### Submatriz (2, 2)

Para obtener  $\partial \mathbf{q}_{k+1} / \partial \mathbf{q}_k$  se expresa (5) mediante producto de matriz como

$$\mathbf{q}_{k+1} = \mathbf{q}_k \otimes \mathbf{q}(\cdot) = \mathbf{q}_k \otimes \tilde{\mathbf{q}} = \tilde{\mathbf{Q}} \mathbf{q}_k \quad (12)$$

donde

$$\tilde{\mathbf{Q}} = \begin{bmatrix} \tilde{q}_0 & -\mathbf{v}_{\tilde{\mathbf{q}}}^T \\ \mathbf{v}_{\tilde{\mathbf{q}}} & \tilde{q}_0 \mathbf{I}_{3 \times 3} + \mathbf{S}_{\mathbf{v}_{\tilde{\mathbf{q}}}}^T \end{bmatrix} = \begin{bmatrix} \tilde{q}_0 & -\tilde{q}_1 & -\tilde{q}_2 & -\tilde{q}_3 \\ \tilde{q}_1 & \tilde{q}_0 & \tilde{q}_3 & -\tilde{q}_2 \\ \tilde{q}_2 & -\tilde{q}_3 & \tilde{q}_0 & \tilde{q}_1 \\ \tilde{q}_3 & \tilde{q}_2 & -\tilde{q}_1 & \tilde{q}_0 \end{bmatrix}. \quad (13)$$

Entonces

$$\frac{\partial \mathbf{q}_{k+1}}{\partial \mathbf{q}_k} = \frac{\partial}{\partial \mathbf{q}_k} \tilde{\mathbf{Q}} \mathbf{q}_k = \tilde{\mathbf{Q}}. \quad (14)$$

**Submatriz (2, 5)**

La derivada

$$\frac{\partial \mathbf{q}_{k+1}}{\partial \mathbf{b}_{\omega,k}} = \frac{\partial}{\partial \mathbf{b}_{\omega,k}} \left( \mathbf{q}_k \otimes \mathbf{q}((\boldsymbol{\omega}_k - \mathbf{b}_{\omega,k})\Delta t) \right)$$

se obtiene aplicando la regla de la cadena

$$\frac{\partial \mathbf{q}_{k+1}}{\partial \mathbf{b}_{\omega,k}} = \frac{\partial \mathbf{q}_{k+1}}{\partial \mathbf{q}((\boldsymbol{\omega}_k - \mathbf{b}_{\omega,k})\Delta t)} \frac{\partial \mathbf{q}((\boldsymbol{\omega}_k - \mathbf{b}_{\omega,k})\Delta t)}{\partial \mathbf{b}_{\omega,k}}, \quad (15)$$

donde

$$\frac{\partial \mathbf{q}_{k+1}}{\partial \mathbf{q}((\boldsymbol{\omega}_k - \mathbf{b}_{\omega,k})\Delta t)} = \begin{bmatrix} q_0 & -\mathbf{v}_q^T \\ \mathbf{v}_q & q_0 \mathbf{I}_{3 \times 3} + \mathbf{S}_{\mathbf{v}_q} \end{bmatrix} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_1 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix},$$

y el segundo factor se calcula a partir de la conversión de vector a cuaternion. El cuaternion obtenido a partir del vector es

$$\mathbf{q}((\boldsymbol{\omega} - \mathbf{b}_\omega)\Delta t) = \left[ \cos\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right) \quad \frac{(\boldsymbol{\omega} - \mathbf{b}_\omega)^T}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|} \sin\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right) \right]^T, \quad (16)$$

o sea

$$q_0((\boldsymbol{\omega} - \mathbf{b}_\omega)\Delta t) = \cos\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right) \quad (17)$$

$$q_i((\boldsymbol{\omega} - \mathbf{b}_\omega)\Delta t) = \frac{\omega_i - b_{\omega,i}}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|} \sin\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right), \quad i = 1, 2, 3 \quad (18)$$

con  $\mathbf{b}_\omega = [b_{\omega,x} \quad b_{\omega,y} \quad b_{\omega,z}]^T = [b_{\omega,1} \quad b_{\omega,2} \quad b_{\omega,3}]^T$ .

Separando las derivadas por cada componente se tiene

$$\frac{\partial \mathbf{q}((\boldsymbol{\omega}_k - \mathbf{b}_{\omega,k})\Delta t)}{\partial \mathbf{b}_{\omega,k}} = \frac{\partial \mathbf{q}(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial \mathbf{b}_{\omega,k}} = \begin{bmatrix} \frac{\partial q_0(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,1}} & \frac{\partial q_0(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,2}} & \frac{\partial q_0(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,3}} \\ \frac{\partial q_1(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,1}} & \frac{\partial q_1(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,2}} & \frac{\partial q_1(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,3}} \\ \frac{\partial q_2(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,1}} & \frac{\partial q_2(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,2}} & \frac{\partial q_2(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,3}} \\ \frac{\partial q_3(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,1}} & \frac{\partial q_3(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,2}} & \frac{\partial q_3(\bar{\boldsymbol{\omega}}_k \Delta t)}{\partial b_{\omega,3}} \end{bmatrix}, \quad (19)$$

donde

$$\frac{\partial q_0(\boldsymbol{\omega} - \mathbf{b}_\omega)}{\partial b_{\omega,i}} = \frac{\Delta t}{2} \frac{\omega_i - b_{\omega,i}}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|} \sin\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right), \quad i = 1, 2, 3 \quad (20)$$

$$\begin{aligned} \frac{\partial q_i(\boldsymbol{\omega} - \mathbf{b}_\omega)}{\partial b_{\omega,i}} &= \left( \frac{(\omega_i - b_{\omega,i})^2}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|^3} - \frac{1}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|} \right) \sin\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right) \\ &\quad - \frac{\Delta t}{2} \frac{(\omega_i - b_{\omega,i})^2}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|^2} \cos\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right), \quad i = 1, 2, 3 \end{aligned} \quad (21)$$

$$\begin{aligned} \frac{\partial q_i(\boldsymbol{\omega} - \mathbf{b}_\omega)}{\partial b_{\omega,j}} &= \frac{(\omega_i - b_{\omega,i})(\omega_j - b_{\omega,j})}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|^3} \sin\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right) \\ &\quad - \frac{\Delta t}{2} \frac{(\omega_i - b_{\omega,i})(\omega_j - b_{\omega,j})}{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|^2} \cos\left(\frac{\|\boldsymbol{\omega} - \mathbf{b}_\omega\|\Delta t}{2}\right), \quad i \neq 0, i \neq j \end{aligned} \quad (22)$$

O sea que (20) es la primer fila de (19), (21) es la diagonal de la submatriz inferior de  $3 \times 3$  de (19), y (22) son los elementos fuera de la diagonal de dicha submatriz.

**Submatriz (3, 2)**

$$\begin{aligned}\frac{\partial \mathbf{v}_{k+1}}{\partial \mathbf{q}_k} &= \frac{\partial}{\partial \mathbf{q}_k} \left( \mathbf{R}(\mathbf{q}_k) \bar{\mathbf{a}}_k \Delta t \right) = \Delta t \frac{\partial \bar{\mathbf{a}}_k^n}{\partial \mathbf{q}_k}, \\ &= \Delta t \begin{bmatrix} \frac{\partial \bar{\mathbf{a}}_k^n}{\partial q_0} & \frac{\partial \bar{\mathbf{a}}_k^n}{\partial q_1} & \frac{\partial \bar{\mathbf{a}}_k^n}{\partial q_2} & \frac{\partial \bar{\mathbf{a}}_k^n}{\partial q_3} \end{bmatrix}\end{aligned}\quad (23)$$

con  $\bar{\mathbf{a}}_k = \mathbf{a}_k - \mathbf{b}_{a,k} + \mathbf{w}_{a,k}$  el valor real de aceleración lineal, y  $\bar{\mathbf{a}}_k^n = \mathbf{R}(\mathbf{q}_k) \bar{\mathbf{a}}_k$  dicho valor expresado en el marco de referencia de navegación.

Luego, dado que

$$\frac{\partial \bar{\mathbf{a}}_k^n}{\partial q_i} = \frac{\partial \mathbf{R}(\mathbf{q}_k) \bar{\mathbf{a}}_k}{\partial q_i} = \frac{\partial \mathbf{R}(\mathbf{q}_k)}{\partial q_i} \bar{\mathbf{a}}_k, \quad i = 0, \dots, 3,$$

la derivada de la matriz de rotación respecto a cada componente del cuaternion queda

$$\frac{\partial \mathbf{R}(\mathbf{q}_k)}{\partial q_0} = \begin{bmatrix} 2q_0 & -2q_3 & 2q_2 \\ 2q_3 & 2q_0 & -2q_1 \\ -2q_2 & 2q_1 & 2q_0 \end{bmatrix} \quad (24)$$

$$\frac{\partial \mathbf{R}(\mathbf{q}_k)}{\partial q_1} = \begin{bmatrix} 2q_1 & 2q_2 & 2q_3 \\ 2q_2 & -2q_1 & -2q_0 \\ 2q_3 & 2q_0 & -2q_1 \end{bmatrix} \quad (25)$$

$$\frac{\partial \mathbf{R}(\mathbf{q}_k)}{\partial q_2} = \begin{bmatrix} -2q_2 & 2q_1 & 2q_0 \\ 2q_1 & 2q_2 & 2q_3 \\ -2q_0 & 2q_3 & -2q_2 \end{bmatrix} \quad (26)$$

$$\frac{\partial \mathbf{R}(\mathbf{q}_k)}{\partial q_3} = \begin{bmatrix} -2q_3 & -2q_0 & 2q_1 \\ 2q_0 & -2q_3 & 2q_2 \\ 2q_1 & 2q_2 & 2q_3 \end{bmatrix} \quad (27)$$

**Submatriz (3, 4)**

$$\frac{\partial \mathbf{v}_{k+1}}{\partial \mathbf{b}_{a,k}} = -\frac{\partial \mathbf{R}(\mathbf{q}_k) \mathbf{b}_{a,k} \Delta t}{\partial \mathbf{b}_{a,k}} = -\mathbf{R}(\mathbf{q}_k) \Delta t \quad (28)$$

**4.1. Calculo del jacobiano  $\mathbf{G}_k$** 

El jacobiano del modelo de movimiento dado por (4) a (8) respecto a la entrada o mediciones inerciales  $\mathbf{u} = [\mathbf{a} \ \boldsymbol{\omega}]^T$  esta dado por

$$\mathbf{G}_k = \frac{\partial f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)}{\partial \mathbf{u}_k} = \begin{bmatrix} \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{4 \times 3} & \frac{\partial \mathbf{q}_{k+1}}{\partial \boldsymbol{\omega}_k} \\ \frac{\partial \mathbf{v}_{k+1}}{\partial \mathbf{a}_k} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \end{bmatrix}, \quad (29)$$

donde la submatrices de (29) tienen las siguientes dimensiones

- la submatriz (2, 2) es de  $4 \times 3$ ,
- la submatriz (3, 1) es de  $3 \times 3$ .

### Submatriz (2, 2)

Derivada respecto a  $\omega$

$$\frac{\partial q_0(\omega - \mathbf{b}_\omega)}{\partial \omega_i} = -\frac{\Delta t}{2} \frac{\omega_i - b_{\omega_i}}{\|\omega - \mathbf{b}_\omega\|} \sin\left(\frac{\|\omega - \mathbf{b}_\omega\| \Delta t}{2}\right), \quad i = 1, 2, 3 \quad (30)$$

$$\frac{\partial q_i(\omega - \mathbf{b}_\omega)}{\partial \omega_i} = \left( \frac{1}{\|\omega - \mathbf{b}_\omega\|} - \frac{(\omega_i - b_{\omega_i})^2}{\|\omega - \mathbf{b}_\omega\|^3} \right) \sin\left(\frac{\|\omega - \mathbf{b}_\omega\| \Delta t}{2}\right) \quad (31)$$

$$+ \frac{\Delta t}{2} \frac{(\omega_i - b_{\omega_i})^2}{\|\omega - \mathbf{b}_\omega\|^2} \cos\left(\frac{\|\omega - \mathbf{b}_\omega\| \Delta t}{2}\right), \quad i = 1, 2, 3 \quad (32)$$

$$\frac{\partial q_i(\omega - \mathbf{b}_\omega)}{\partial \omega_j} = -\frac{(\omega_i - b_{\omega_i})(\omega_j - b_{\omega_j})}{\|\omega - \mathbf{b}_\omega\|^3} \sin\left(\frac{\|\omega - \mathbf{b}_\omega\| \Delta t}{2}\right) \quad (33)$$

$$+ \frac{\Delta t}{2} \frac{(\omega_i - b_{\omega_i})(\omega_j - b_{\omega_j})}{\|\omega - \mathbf{b}_\omega\|^2} \cos\left(\frac{\|\omega - \mathbf{b}_\omega\| \Delta t}{2}\right), \quad i \neq 0, i \neq j \quad (34)$$

### Submatriz (3, 1)

$$\frac{\partial \mathbf{v}_{k+1}}{\partial \mathbf{a}_k} = \frac{\partial}{\partial \mathbf{a}_k} \mathbf{R}(\mathbf{q}_k)(\mathbf{a}_k - \mathbf{b}_{a,k}) \Delta t = \Delta t \mathbf{R}(\mathbf{q}_k) \quad (35)$$

## 5. Implementación del modelo en software

La implementación en software se realizó para PC bajo el lenguaje C++. Se utilizaron las siguientes librerías para simplificar los cálculos matriciales y diferentes problemas que surgen a la hora de optimizar los algoritmos.

- Eigen3 para cálculos matriciales.
- Boost para transformación de tipos de datos.

Con todo esto en conjunto se generó una librería llamada *Imu motion model* la cual reemplaza a *Motion model* en la estructura de software presentada en la figura 2.



## Referencias

- Rainer Voigt, J. Nikolic, C. Hürzeler, S. Weiss, L. Kneip, and R. Siegwart. Robust embedded egomotion estimation. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2694–2699, Sept 2011. doi: 10.1109/IROS.2011.6095122.
- Rudolph van der Merwe, Eric Wan, and Simon Julier. Sigma-point kalman filters for non-linear estimation and sensor-fusion: Applications to integrated navigation. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, August 2004. doi: 10.2514/6.2004-5120.
- Nikolas Trawny and Stergios I. Roumeliotis. Indirect Kalman filter for 3D attitude estimation. Technical Report 2005-002, University of Minnesota, Dept. of Comp. Sci. & Eng., March 2005.
- Martín A. Pucheta, Claudio J. Paz, and Maira E. Pereyra. Representaciones cinemáticas de orientación y ecuaciones de estimación. In *XXI Congreso sobre Métodos Numéricos y sus Aplicaciones ENIEF*, volume XXXIII, pages 2303–2324, 2014.
- A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052 – 1067, june 2007. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1049.
- Javier Civera, Oscar G. Grasa, Andrew J. Davison, and J. M. M. Montiel. 1-point ransac for extended kalman filtering: Application to real-time structure from motion and visual odometry. *J. Field Robotics*, 27(5):609–631, 2010.