Implementación de un filtro de Kalman de dos etapas para el seguimiento de palillos de batería en tiempo real

Joaquín Manuel Ferraro, Francisco Costanza, Joaquín Huarita y Sebastián Verrastro

Universidad Tecnológica Nacional, Facultad Regional Buenos Aires,
Medrano 951, (C1179AAQ), Ciudad Autónoma de Buenos Aires, Argentina

jferraro@frba.utn.edu.ar

Recibido el 17 de abril de 2025, aprobado el 30 de mayo de 2025

Resumen

En el presente trabajo se desarrolla un sistema capaz de calcular la posición a tiempo real de palillos de batería mediante la utilización de sensores y reconocimiento de imágenes. Para su ejecución, se plantea la utilización de sensores inerciales, conocidos como IMUs—tales como un acelerómetro, giroscopio y magnetómetro—y las imágenes en colores con mapa de profundidad (RGB-D) de una cámara Microsoft Kinect, analizadas mediante una única red neuronal convolucional (YOLO). Asimismo, para calcular la odometría de los palillos, se utilizarán técnicas de fusión de sensores a través distintos filtros de Kalman, dependiendo de lo que se requiera calcular. El objetivo final del mismo es poder emular el funcionamiento de una batería real.

PALABRAS CLAVE: IMU - RGB-D - YOLO - FILTRO DE KALMAN – FUSIÓN

Abstract

In the present work, a system capable of calculating the real-time position of drumsticks is developed using sensors and image recognition. For its implementation, the use of inertial sensors, known as IMUs—such as an accelerometer, gyroscope, and magnetometer—is proposed, along with color images with depth maps (RGB-D) from a Microsoft Kinect camera, analyzed using a single convolutional neural network (YOLO). Likewise, to calculate the odometry of the drumsticks, sensor fusion techniques using various Kalman filters will be employed, depending on the required calculation. The goal is to emulate the functioning of a real drum set.

KEYWORDS: IMU - RGB-D - YOLO - KALMAN'S FILTER - FUSION

Introducción

En la actualidad, la necesidad de conocer la posición de un objeto es vital en múltiples aplicaciones. En dispositivos como aviones o automóviles, estas dos características no poseen una gran incidencia, ya que con conocer la ubicación con una precisión de metros y con una periodicidad de segundos, es suficiente. Sin embargo, en ciertas aplicaciones, como el de un palillo de batería, donde el seguimiento debe ser preciso, y al moverse rápidamente, el tiempo de cálculo debe ser veloz, el enfoque cambia radicalmente al tomado en los casos mencionados.

Existen muchas metodologías para calcular la posición de un objeto en concreto. Entre ellas, la más ampliamente conocida es el Global Positioning System, o GPS abreviado, el cual es un sistema de navegación satelital que permite determinar la latitud, longitud y altitud de un receptor mediante el uso de al menos cuatro satélites y un proceso llamado trilateración (Kaplan *et al.* 2005). Asimismo, existen otros métodos conocidos como la utilización de pulsos de luz o radiofrecuencia (LIDAR y radares respectivamente).

Sin embargo, estos sistemas cuentan con ciertas limitaciones en lo que respecta a rapidez de respuesta y precisión (Misra *et al.* 2006). Dentro del campo de la odometría, debido a las limitaciones con las que cuenta cada metodología, se comenzó a utilizar en gran medida lo conocido como Filtros de Kalman (Kalman, 1960). Este algoritmo permite la utilización de distintas fuentes de medición, ponderar sus incertidumbres y combinarlas para obtener un resultado capaz de compensar sus defectos.

Tal es el caso del Inertial Navigation System (INS), donde se emplean acelerómetros, magnetómetros y giroscopios para calcular la variación de la posición mediante mediciones inerciales (Groves, 2013). Por un lado, el giroscopio aportará su precisión inicial, mientras que el acelerómetro y magnetómetro contribuirán su estabilidad a largo plazo (Mahony *et al.* 2008), para el cálculo de la orientación. Sin embargo, si bien este sistema cuenta con la virtud de funcionar sin depender de señales externas, no posee una gran precisión debido a que su error es acumulativo (Kok *et al.* 2017). Por ello, este es utilizado simultáneamente con otros sistemas tales como GPS, de manera de compensar sus debilidades.

Contribuciones

Este artículo está orientado a la utilización de sensores inerciales combinados con métodos de odometría y reconocimiento de imágenes a través del uso de redes neuronales convolucionales.

Mediante estas estrategias es posible cumplir con la rapidez y precisión requeridas. No obstante, cabe señalar que ninguna de las dos metodologías cumple con estas características individualmente, sino que cada una cubre los aspectos faltantes de la otra. Dicho de otro modo, la rapidez de los sensores y la precisión del reconocimiento de imágenes combinados, permitirán cumplir con lo requerido. Para lograrlo, será necesario utilizar alguna de las metodologías existentes para fusionar respuestas similares.

Es relevante mencionar que metodologías como la ponderación o el promediado resultarán insuficientes para cumplir con la tarea encomendada. Esto es porque es necesario compensar constantemente el error generado por los sensores, ya que estos serán los que aportarán información periódicamente al sistema. Por esta razón, se utilizará un Filtro de Kalman (KF), y su versión alineal el Filtro Extendido de Kalman (EKF).

Este enfoque permite cubrir los puntos débiles de los métodos utilizados actualmente. Este es el caso para sistemas de seguimiento combinando GPS e INS (Lu *et al.* 2013), donde suelen alcanzar precisiones en el orden de metros o decímetros en condiciones óptimas, lo cual resulta insuficiente cuando se requiere precisión a nivel de centímetros o

metodologías como SLAM (Cadena *et al.* 2016), las cuales no son compatibles con entornos con condiciones de iluminación variables, fondos complejos o movimientos rápidos.

Finalmente, debido a que el presente método fue inicialmente pensado para lograr el seguimiento de palillos de batería, una vez obtenida la posición, se realizó el seguimiento del desplazamiento de estos y el reconocimiento de los cuerpos de la batería al momento de entrar en contacto con los mismos, ubicados manualmente en algún punto del espacio. Para poder comunicar los sonidos correspondientes se utilizó el estándar tecnológico conocido como *Musical Instrument Digital Interface* (Chattah, 2014), o MIDI abreviado.

Marco teórico

You Only Look Once: YOLO

You Only Look Once (YOLO) es un algoritmo de código abierto para detección de objetos y segmentación de imágenes en tiempo real desarrollado por Joseph Redmon y Ali Farhadi en la Universidad de Washington.

Su funcionamiento se basa en el uso de una única red neuronal convolucional, la cual divide la imagen a analizar en regiones, prediciendo e identificando los recuadros delimitadores y la probabilidad de una clase en una sola pasada (Redmon *et al.* 2016).

El proceso de entrenamiento comienza con la recopilación y anotación de un conjunto de imágenes del o los objetos que se desean detectar y la separación en subconjuntos de entrenamiento, test y validación. Existen numerosas plataformas que facilitan la tarea de construir un *dataset* apropiado y que también proveen facilidades para realizar técnicas de *data augmentation*.

Es posible la utilización de un modelo pre-entrenado en un *dataset* amplio lo cual reduce el tiempo de entrenamiento considerablemente respecto a un modelo sin entrenar al ya haber aprendido características generales (detección de bordes, texturas y patrones). Para ello se debe de configurar un mínimo de hiperparámetros necesarios para el entrenamiento, tales como el *Epoch number, Batch size, Learning rate* e *Image size*.

Durante el entrenamiento, cada imagen (o *batch*) se pasa por la red, se calculan las predicciones y se evalúa la pérdida total a partir de los distintas componentes (localización, clasificación y confianza). Luego al terminar cada época se calculan los gradientes de la función de pérdida para actualizar los pesos del modelo y se registran métricas como la precisión media (mAP).

Finalizado el proceso de entrenamiento, se evalúa su desempeño con el conjunto de imágenes de validación y se obtienen métricas como la precisión media (mAP) a distintos umbrales de *Intersection over Union* (IoU), la matriz de confusión, la curva F1 vs Confianza y la curva de precisión vs *recall*.

Microsoft Kinect

La Microsoft Kinect es una cámara inteligente de captura de movimiento desarrollada por Microsoft. Su principal innovación radica en la capacidad de detectar y rastrear el movimiento humano en tres dimensiones, lo que la ha hecho muy popular en el desarrollo de interfaces naturales, videojuegos, robótica, y aplicaciones en visión por computadora.

Con lo que respecta al presenta artículo, cuenta con cámara de color (RGB) y un proyector y cámara infrarroja (IR). La cámara IR posee una resolución de 320x240 píxeles a 30 fps y la RGB una resolución de 640x480 pixeles 30 fps.

A través de estas características, es capaz de elaborar un mapa de profundidad (Khoshelham *et al.* 2012), mediante la técnica de luz estructurada. Esta consiste en proyectar un patrón IR previamente conocido sobre el entorno, y capturar cómo ese patrón se ve en las superficies de los objetos. Ya que existe una distancia (*baseline*) entre el proyector y la cámara, lo emitido y lo captado, cuenta con cierta deformación. Por ello, mediante la técnica conocida como triangulación, es posible calcular la distancia real deseada. Asimismo, es necesario tener en cuenta las propiedades geométricas de la cámara, tal como la distancia focal.

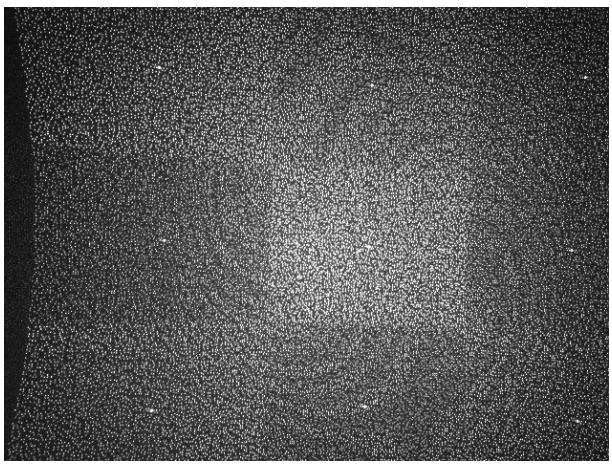


Fig. 1. Patrón empleado para utilizar el mapa de profundidad de la Kinect

El proyector IR de la cámara Kinect, emite un patrón fijo de puntos claros y oscuros (Fig. 1). Dicho patrón se genera a partir de un conjunto de rejillas de difracción, con especial cuidado para minimizar el efecto de la propagación de orden cero de un punto brillante central. El mencionado patrón es memorizado a una profundidad conocida y se utiliza una pequeña ventana de correlación para cada pixel de la cámara IR para comparar con el patrón local y 64 píxeles vecinos en una ventana horizontal. La mejor coincidencia otorga un desplazamiento de la profundidad conocida, llamada disparidad. El dispositivo realiza una interpolación adicional de la mejor coincidencia para obtener una precisión de subpíxel de 1/8 de píxel.

$$Profunidad \propto \frac{Baseline \times Focal}{Disparidad}$$

Filtro de Kalman

En el año 1960, Rudolf Emil Kalman publicó un artículo describiendo una solución recursiva para el filtrado de datos discretos en sistemas lineales. Desde ese momento, debido al gran avance tecnológico que ha sufrido el mundo, el Filtro de Kalman comenzó a utilizarse en un gran número de dispositivos, sistemas, máquinas y artefactos, tales como aviones, automóviles y barcos, robótica o incluso seguimiento de indicadores macroeconómicos.

El Filtro de Kalman es un algoritmo recursivo utilizado para estimar el estado de un sistema dinámico a partir de mediciones ruidosas. La manera en que este lo logra es mediante los procesos de predicción y corrección (Welch *et al.* 2006).

Para expresarlo de manera más clara y concisa, es un sistema capaz de estimar dinámicamente el ruido de las mediciones de entrada a través de la comparación de estas con una medición de un origen distinto. Su virtud nace en la capacidad de ajustar a

necesidad la confianza que se tiene a cada una de las mediciones del sistema. Cuanto más cercana a la realidad sean las covarianzas Q y R empleadas en el sistema, la respuesta será más acertada. Mourikis *et al.* (2007) es buen ejemplo de ello.

Filtro de Kalman lineal

Predicción

$$\hat{x}_k^- = F\hat{x}_{k-1}^- + Bu_k$$

$$P_k^- = FP_{k-1}^- F^T + Q$$

Corrección

$$K_k = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - H\hat{x}_k^-\right) = \hat{x}_k^- + K_k y_k$$

$$P_k = \left(I - K_k H\right)P_k^-$$

Filtro de Kalman alineal

Predicción

$$\hat{x}_k^- = f(\hat{x}_{k-1}^-, u_k)$$

$$P_k^- = FP_{k-1}^- F^T + Q$$

Corrección

$$K_k = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - h(\hat{x}_k^-)\right) = \hat{x}_k^- + K_k y_k$$

$$P_k = (I - K_k H)P_k^-$$

Sintetizando el procedimiento del algoritmo, en una primera instancia se realiza una predicción de la salida utilizando la entrada del sistema y la salida anterior. Luego para acercar esta predicción lo más posible a la realidad, se compara lo predicho con una referencia conocida. Con dicha metodología es posible actualizar la salida mediante las matrices K e y, al conocer una estimación de las incertidumbres de ambas partes, las cuales evidenciarán que tan cercano es el valor real a la predicción y a la referencia conocida.

Parte experimental

Hardware empleado

En la etapa de medición, se emplearon los módulos MPU6050 (giroscopio y acelerómetro) y LSM303DHL (magnetómetro), y el protocolo I2C para la comunicación. Luego, en la etapa de control de sensores y comunicación con la aplicación, se utilizó el microcontrolador ESP32 WROOM 32 de la familia de Espressif, el cual cumple con el rol de adecuar la información obtenida y enviarla mediante WiFi a la aplicación de manejo

de datos. Asimismo, se utilizaron diodos LED en los puntos a detectar de los palillos de batería, de manera de mejorar el reconocimiento de estos mediante YOLO.

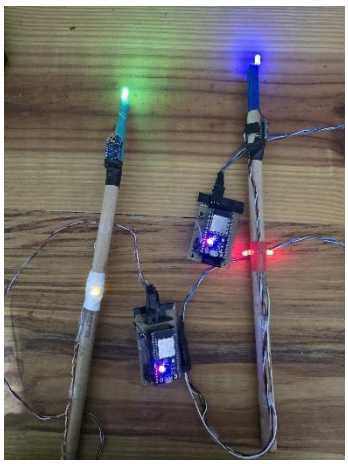


Fig. 2. Palillos de batería empleados

YOLO

El entrenamiento del modelo de YOLO se compuso mediante un *dataset* de 7806 imágenes, obtenidas mediante la filmación de los palillos durante una sesión de práctica normal. Se realizaron dos filmaciones, en una habitación con espacio reducido y luz artificial, y en una habitación más amplia con luz natural.

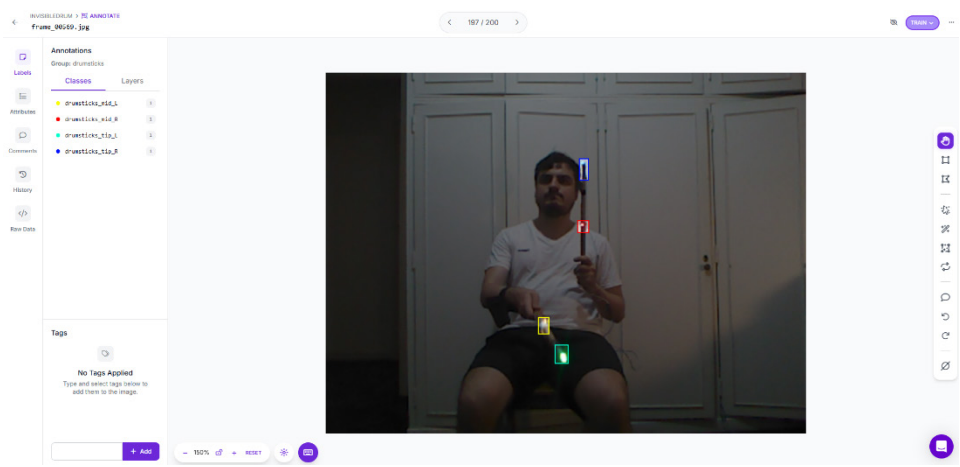


Fig. 3. Proceso de clasificación de imágenes

Se clasificó cada imagen a mano usando cuatro clases, *drumsticks_mid_L* y *drumsticks_mid_R* para los puntos medios de ambos palillos, *drumsticks_tip_L* y *drumsticks_tip_R* para las puntas de estas. Finalizada la clasificación manual, se constituyó la versión del *dataset* para entrenar, dividiendo de forma automática las imágenes en 80% para entrenamiento y 20% en partes iguales entre validación y *testing*. Por último, se realizó un preprocesamiento del *dataset* ajustando la resolución de las imágenes a 640x480 y *data augmentation* para generar imágenes artificiales variando el brillo entre un +/-15%. Como resultado, se obtuvo un *dataset* final de 13271 imágenes.

El entrenamiento fue realizado mediante la versión *small* del último modelo ofrecido por Ultralitics YOLO v11 con los hiperparámetros epochs=100, imgsz=640, batch=30 y warmup_epochs=5. La cantidad de épocas (epochs) fue determinada luego de observar que las métricas de precisión, *recall* y las medias de precisión promedio en los

dos umbrales, mAP50 y mAP50-95, se estabilizaran. El tamaño del lote (batch) fue seleccionado empleando un compromiso entre el consumo de recursos de la máquina virtual con la plataforma que se entrenó el modelo y la calidad del aprendizaje. Para el parámetro de warmup_epochs se buscó estabilizar el entrenamiento inicial y mitigar actualizaciones abruptas del modelo en las primeras épocas, lo cual podría llevar a una mala convergencia. Por último, para las tasas de aprendizaje (*learning rate* inicial y final) fueron utilizados los valores por defecto de 0,01.

Cálculo de orientación del palillo

Para calcular la orientación, se contó con las lecturas del acelerómetro, giroscopio y magnetómetro, cuyas unidades son $\frac{m}{s^2}$, $\frac{rad}{s}$ respectivamente y otorgan las mediciones de cada uno de sus tres ejes (*Degrees Of Freedom*, 3 DOF) *[x,y,z]*.

Mediante estos sensores, será posible calcular la rotación del objeto de dos orígenes distintos, maximizando el Kalman en la etapa de innovación. Asimismo, el giroscopio reforzó el sistema ante movimientos bruscos sin rotación y el acelerómetro/magnetómetro apaciguaron el error a largo plazo con el que cuenta el giroscopio.

Por otro lado, es importante tener en cuenta que previamente a trabajar con las lecturas, se trasladaron al marco de referencia correspondiente (en este caso, NED) (Groves, 2013). Para lograrlo, hay que conocer la dirección y sentido de los ejes de los sensores.

Inicialmente, es necesario comprender cómo es posible calcular la orientación mediante estas lecturas. Para lograrlo usando un acelerómetro y un magnetómetro, se aprovechan las propiedades de ambos sensores para obtener información sobre la inclinación y la dirección en el espacio.

Para el caso del acelerómetro, se utiliza la gravedad, la cual permitirá calcular únicamente el *roll* y el *pitch*, ya que no es posible obtener la variación en el eje z, ya que la gravedad no se verá afectada por movimientos sobre dicho eje, y por consiguiente no es posible calcular el *yaw*.

$$roll = arctan\frac{a_y}{a_z}$$

$$pitch = arctan\ -\frac{a_x}{\sqrt{a_y^{\ 2}+a_z^{\ 2}}}$$

Luego, para el caso del magnetómetro, se emplea el campo magnético terrestre medido por éste.

$$yaw = arctan\ -\frac{m_y}{m_x}$$

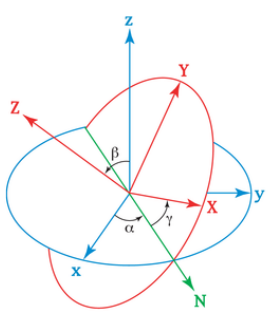


Fig. 4. Ángulos de Euler

Para determinar la orientación mediante el giroscopio, se detectan los cambios en la orientación midiendo la velocidad angular en cada uno de sus ejes.

$$\Delta\theta = \omega \times \Delta t$$

Sin embargo, los ángulos de Euler sufren singularidades (Diebel, 2006). Por esta razón, se necesita calcular la orientación mediante cuaterniones (Trawny *et al*. 2005).

Para ello, la metodología de cálculo para cada sensor se ve modificada. Para el caso del magnetómetro y acelerómetro, se utiliza la ecuación de rotación de un vector mediante la matriz de rotación correspondiente, obtenida a través del cuaternión de referencia (Madgwick, 2010).

$$v_{rotado} = R \times v$$

Siendo R la Direction Cosine Matrix (DCM) correspondiente a la rotación efectuada.

$$R = \begin{bmatrix} cos\theta\, cos\psi & cos\psi\, sin\theta\, sin\phi - sin\psi\, cos\phi & cos\psi\, sin\theta\, cos\phi - sin\psi\, sin\phi \\ sin\psi\, cos\theta & sin\psi\, sin\theta\, sin\phi + cos\psi\, cos\phi & sin\psi\, sin\theta\, cos\phi - cos\psi\, sin\phi \\ -sin\theta & cos\theta\, sin\phi & cos\theta\, cos\phi \end{bmatrix}$$

$$= \begin{bmatrix} 1 - 2(q_y{}^2 + q_z{}^2) & 2(q_x q_y - q_w q_z), & 2(q_x q_z + q_w q_y) \\ 2(q_x q_y + q_w q_z) & 1 - 2(q_x{}^2 + q_z{}^2) & 2(q_x q_z - q_w q_x) \\ 2(q_x q_z - q_w q_y) & 2(q_x q_z + q_w q_x) & 1 - 2(q_x{}^2 + q_y{}^2) \end{bmatrix}$$

Por lo tanto, al tener definido un vector de referencia, es posible obtener la medición de la aceleración y del magnetómetro como los vectores rotados, y por consecuencia, obtener la DCM.

En el caso del giroscopio, se utilizará la ecuación $q(t+\Delta t)=q(t)+q(t)dt$ de la derivada de un cuaternión. Luego, con la ecuación y un Δt definido es posible calcular la nueva orientación.

$$\dot{q}(t) = \frac{1}{2} q(t) \otimes \Omega(t)$$

$$\Omega = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & -\omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_z & 0 \end{bmatrix}$$

Una vez definida la metodología de cálculo, solo resta definir las matrices de Kalman, para luego aplicar las ecuaciones. Es importante destacar que al ser un sistema alineal, será inevitable implementar el EKF. Para definir la matriz $f(\hat{x}_{k-1}^{-}, u_k)$ se utilizó el cálculo de la derivada mediante la velocidad angular medida por el giroscopio, utilizada para calcular Ω.

$$f(\hat{x}_{k-1}^{-}, u_k) = \begin{bmatrix} 1 + \frac{1}{2}\Delta t \Omega_w \\ 1 + \frac{1}{2}\Delta t \Omega_x \\ 1 + \frac{1}{2}\Delta t \Omega_y \\ 1 + \frac{1}{2}\Delta t \Omega_z \end{bmatrix} \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix}$$

Luego, la matriz F será la misma expresión, pero derivada en función de q.

$$F = \begin{bmatrix} 1 + \frac{1}{2}\Delta t\Omega_w \\ 1 + \frac{1}{2}\Delta t\Omega_x \\ 1 + \frac{1}{2}\Delta t\Omega_y \\ 1 + \frac{1}{2}\Delta t\Omega_z \end{bmatrix}$$

Continuando con Q, se utiliza el ruido propio del giroscopio y el jacobiano de la matriz $f(\hat{x}_{k-1}^-, u_k)$ en función de la velocidad angular.

$$Q = N_\Omega \frac{1}{2}\Delta t \begin{pmatrix} -q_1 & -q_2 & -q_3 \\ q_0 I_3 + skew \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} \end{pmatrix}, \qquad skew \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{pmatrix}$$

Donde el *skew (q)* es la matriz asimétrica del cuaternión y N_Ω el ruido de Ω. Luego, la matriz $h(\hat{x}_k^-)$ se encuentra definida como la rotación de las referencias del acelerómetro y magnetometro (marco NED), rotadas por el cuaternión calculado en la predicción.

$$h(\hat{x}_k^-) = \begin{bmatrix} \begin{bmatrix} 1-2({q_y}^2+{q_z}^2) & 2(q_xq_y - q_wq_z), & 2(q_xq_z + q_wq_y) \\ 2(q_xq_y + q_wq_z) & 1-2({q_x}^2+{q_z}^2) & 2(q_xq_z - q_wq_x) \\ 2(q_xq_z - q_wq_y) & 2(q_xq_z + q_wq_x) & 1-2({q_x}^2+{q_y}^2) \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\ \begin{bmatrix} 1-2({q_y}^2+{q_z}^2) & 2(q_xq_y - q_wq_z), & 2(q_xq_z + q_wq_y) \\ 2(q_xq_y + q_wq_z) & 1-2({q_x}^2+{q_z}^2) & 2(q_xq_z - q_wq_x) \\ 2(q_xq_z - q_wq_y) & 2(q_xq_z + q_wq_x) & 1-2({q_x}^2+{q_y}^2) \end{bmatrix} \begin{bmatrix} 0 \\ \cos(r\_mag) \\ -sen(r\_mag) \end{bmatrix} \end{bmatrix}$$

Donde la referencia magnética se encuentra asociada con la posición global en la que fue efectuada la medición del sensor.

Para el cálculo de H, es necesario derivar la matriz $h(\hat{x}_k^-)$ en función del cuaternión. A continuación, se realiza el desarrollo de la obtención de dicha matriz.

Definiendo a v como el vector a rotar y q como la rotación efectuada sobre el vector.

$$v = [0, x_v, y_v, z_v], \qquad q = [w_q, x_q, y_q, z_q], \qquad q^* = [w_q, -x_q, -y_q, -z_q]$$

Es posible decir que:

$$q \times v = \begin{bmatrix} -x_q x_v - y_q y_v - z_q z_v \\ w_q x_v + y_q z_v - z_q y_v \\ w_q y_v - x_q z_v + z_q x_v \\ w_q z_v + x_q y_v - y_q x_v \end{bmatrix}$$

$$A = -x_q x_v - y_q y_v - z_q z_v$$

$$B = w_q x_v + y_q z_v - z_q y_v$$

$$C = w_q y_v - x_q z_v + z_q x_v$$

$$D = w_q z_v + x_q y_v - y_q x_v$$

$$q \times v \times q^* = \begin{bmatrix} Aw_q + Bx_q + Cy_q + Dz_q \\ -Ax_q + Bw_q - Cz_q + Dy_q \\ -Ay_q + Bz_q + Cw_q - Dx_q \\ -Az_q - By_q + Cx_q + Dw_q \end{bmatrix} \Longrightarrow V_{rot} = \begin{bmatrix} -Ax_q + Bw_q - Cz_q + Dy_q \\ -Ay_q + Bz_q + Cw_q - Dx_q \\ -Az_q - By_q + Cx_q + Dw_q \end{bmatrix}$$

Luego, para efectuar la derivada de una función por un cuaternión, se debe derivar la misma por cada componente que conforma el cuaternión ([wq,xq,yq,zq]).

$$\frac{\partial V_{rot}}{\partial w_q} = 2 \times \begin{bmatrix} x_v w_q + y_v z_q - z_v y_q \\ -x_v z_q + y_v w_q + z_v x_q \\ x_v y_q - y_v x_q + z_v w_q \end{bmatrix}, \quad \frac{\partial V_{rot}}{\partial x_q} = 2 \times \begin{bmatrix} x_v x_q + y_v y_q - z_v z_q \\ x_v y_q - y_v x_q + z_v w_q \\ x_v z_q - y_v w_q - z_v x_q \end{bmatrix}$$

$$\frac{\partial V_{rot}}{\partial y_q} = 2 \times \begin{bmatrix} -x_v y_q + y_v x_q - z_v w_q \\ x_v x_q + y_v y_q + z_v z_q \\ x_v x_q + y_v z_q - z_v y_q \end{bmatrix}, \quad \frac{\partial V_{rot}}{\partial z_q} = 2 \times \begin{bmatrix} -x_v z_q + y_v w_q - z_v x_q \\ -x_v w_q - y_v z_q + z_v y_q \\ x_v x_q + y_v y_q + z_v z_q \end{bmatrix}$$

Finalmente, reemplazando el vector por las referencias de aceleración y campo magnético, podemos obtener la matriz *H*.

$$H = 2 \times \begin{bmatrix} \begin{bmatrix} x_a w_q + y_a z_q - z_a y_q & x_a x_q + y_a y_q - z_a z_q & -x_a y_q + y_a x_q - z_a w_q & -x_a z_q + y_a w_q - z_a x_q \\ -x_a z_q + y_a w_q + z_a x_q & x_a y_q - y_a x_q + z_a w_q & x_a x_q + y_a y_q + z_a z_q & -x_a w_q - y_a z_q + z_a y_q \\ x_a y_q - y_a x_q + z_a w_q & x_a z_q - y_a w_q - z_a x_q & x_a x_q + y_a z_q - z_a y_q & x_a x_q + y_a y_q + z_a z_q \end{bmatrix} \\ \begin{bmatrix} x_m w_q + y_m z_q - z_m y_q & x_m x_q + y_m y_q - z_m z_q & -x_m y_q + y_m x_q - z_m w_q & -x_m z_q + y_m w_q - z_m x_q \\ -x_m z_q + y_m w_q + z_m x_q & x_m y_q - y_m x_q + z_m w_q & x_m x_q + y_m y_q + z_m z_q & -x_m w_q - y_m z_q + z_m y_q \\ x_m y_q - y_m x_q + z_m w_q & x_m z_q - y_m w_q - z_m x_q & x_m x_q + y_m z_q - z_m y_q & x_m x_q + y_m y_q + z_m z_q \end{bmatrix} \end{bmatrix}$$

Por último, la matriz *R* será igual a:

$$R = \begin{bmatrix} N_a & 0 & 0 & 0 & 0 & 0 \\ 0 & N_a & 0 & 0 & 0 & 0 \\ 0 & 0 & N_a & 0 & 0 & 0 \\ 0 & 0 & 0 & N_m & 0 & 0 \\ 0 & 0 & 0 & 0 & N_m & 0 \\ 0 & 0 & 0 & 0 & 0 & N_m \end{bmatrix}$$

Donde N_a representa la incertidumbre del acelerómetro y N_m la del magnetómetro.

Sincronización de datos

Antes de efectuar el cálculo de posición, fusionando los sensores con YOLO, es de suma importancia sincronizar los datos empleados tanto espacialmente, como temporalmente. Asimismo, es necesario que el marco de referencia utilizado, para este caso el de la aplicación, sean idénticos (World frame a App frame).

Para la sincronización espacial, se empleó un punto de referencia definido manualmente. A partir de dicho punto conocido, se calcularon los desplazamientos de forma que tanto sensores como YOLO tengan el mismo origen. A partir de dichas coordenadas, se trazaron las ubicaciones de los cuerpos de batería utilizados posteriormente para la obtención de resultados.

Por otro lado, la sincronización temporal utilizó el mismo principio. Al contar con una referencia, configurada en el mismo instante que se definió el origen de posición, cada muestra procesada tendrá como marca temporal el tiempo transcurrido desde la misma. Es importante destacar, que ambas referencias temporales (sensores y YOLO) deben ser tomadas en simultáneo.

Finalmente, debido a las diferentes tasas de respuesta con las que cuenta cada sistema, previo a realizar una nueva iteración del filtro de Kalman, se realiza una comparación temporal. Si bien ambas muestras cuentan con el mismo origen, la existencia de demoras es inminente en un sistema a tiempo real, por lo que es necesario establecer un umbral de aceptación entre muestras. Para el caso del presente artículo, se empleó una diferencia temporal de 50 ms para determinar que curso de acción tomar para cada iteración. En caso de desincronización temporal y la existencia de una nueva lectura de los sensores,

se realizó una iteración del Kalman empleando la última medición de la cámara. Para la situación de desincronización temporal y solo nueva lectura de YOLO, se utilizó únicamente dichas muestras, contemplando a la aceleración de los sensores como nula.

Cálculo de la posición del palillo

El cálculo de la posición es sencillo en lo que concierne a ecuaciones, ya que simplemente consta de aplicar las fórmulas conocidas para un MRUV, partiendo de la aceleración y de la velocidad y posición del instante anterior. La única particularidad es la eliminación del efecto de la gravedad de la medición de la aceleración.

Para lograrlo, es necesario rotar el vector de gravedad global y restarlo a la aceleración.

$$a_{s/g} = a - g_{rot}$$

Sin embargo, en el Kalman no participará únicamente la posición calculada por los sensores. Al igual que se hizo con la orientación, en la etapa de corrección se utilizará una medición de referencia de un origen distinto. Para el caso del presente artículo, la referencia será la posición calculada por el procesamiento de imagen.

IMU + YOLO

Como ya fue mencionado, el Filtro de Kalman utilizará las ecuaciones conocidas para un sistema MRUV para definir las matrices de predicción y la posición calculada por la cámara como referencia para la etapa de corrección. En este caso, será posible emplear el modelo discreto.

Predicción

$$x = \begin{bmatrix} x_x \\ x_y \\ x_z \\ v_x \\ v_y \\ v_z \end{bmatrix}, \quad F = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 & 0 \\ 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & 0 & \frac{1}{2}\Delta t^2 \\ \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{bmatrix}, \quad Q = \begin{bmatrix} Q_i & 0 & 0 & 0 & 0 & 0 \\ 0 & Q_i & 0 & 0 & 0 & 0 \\ 0 & 0 & Q_i & 0 & 0 & 0 \\ 0 & 0 & 0 & Q_i & 0 & 0 \\ 0 & 0 & 0 & 0 & Q_i & 0 \\ 0 & 0 & 0 & 0 & 0 & Q_i \end{bmatrix}$$

Corrección

$$z = \begin{bmatrix} x_{x\,cam} \\ x_{y\,cam} \\ x_{z\,cam} \end{bmatrix}, \quad H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad R = \begin{bmatrix} R_{cam} & 0 & 0 \\ 0 & R_{cam} & 0 \\ 0 & 0 & R_{cam} \end{bmatrix}$$

Donde para el caso de la matriz de covarianza de predicción, en un primer instante se cuenta con una confianza inicial según que tanto se confíe en las primeras estimaciones del Kalman.

Asimismo, los valores de las matrices Q y R son calculados en base a la precisión conocida de los sensores y el reconocimiento de imágenes. En el presente documento, debido a que los sensores empleados son de bajo costo y el cálculo de posición a partir de ellos cuenta con error acumulativo, en la incertidumbre considerada se ve reflejado este efecto (Kong, 2000). Mientras tanto, para el caso del reconocimiento de imágenes, se toma en cuenta como la única referencia absoluta del sistema. Por otro lado, se utilizó como criterio que no existen incertidumbres cruzadas (Q_{ij} donde $i \neq j$).

Resultados

Entrenamiento YOLO

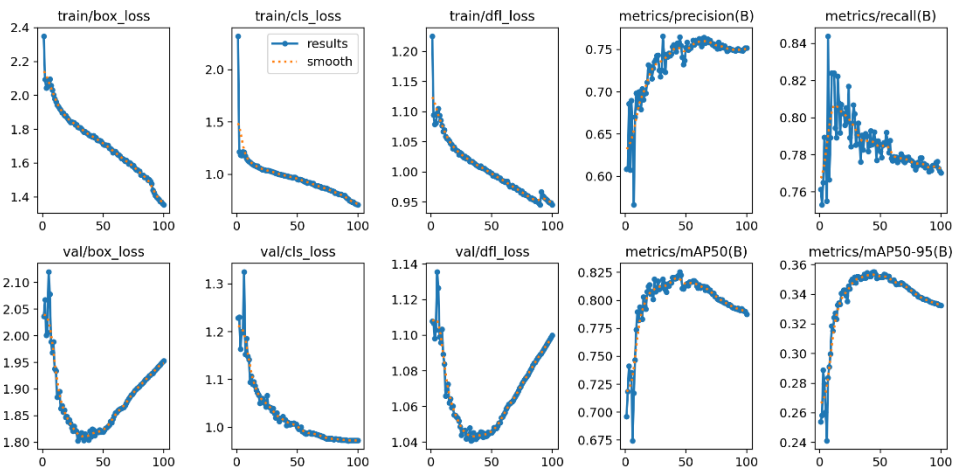


Fig. 5. Evolución de las métricas en cada Época de entrenamiento

Tabla 1. Resultados del entrenamiento del modelo YOLO

Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
all	1562	5375	0,764	0,793	0,827	0,355
drumsticks_mid_L	1125	1137	0,757	0,847	0,826	0,306
drumsticks_mid_L	1549	1561	0,787	0,842	0,873	0,408
drumsticks_mid_L	1128	1138	0,724	0,661	0,748	0,285
drumsticks_mid_L	1535	1539	0,788	0,823	0,861	0,422

En la Tabla 1 se resumen las métricas obtenidas durante el entrenamiento del modelo YOLO. En ella se presentan, para cada clase, el número de imágenes e instancias utilizadas, junto con las métricas de precisión de las cajas (Box(P)), *recall* (R) y la media de precisión promedio (mAP) en dos umbrales: mAP50 y mAP50-95.

Los resultados indican que el modelo muestra un rendimiento sólido. La métrica Box(P) oscila entre 0,724 y 0,788 lo que sugiere que las detecciones son bastante precisas. Asimismo, el *recall* varía entre 0,661 y 0,847, mostrando que el modelo es capaz de recuperar la mayoría de las instancias relevantes. Se observa que las clases *drumsticks_mid_R* y *drumsticks_tip_R* alcanzan los valores más altos de mAP50, lo que indica una detección consistente en esas categorías y se debe a la inclusión de imágenes de un entrenamiento previo únicamente con dicho palillo, por lo que al tener más muestras es natural que presenten mejores métricas respecto a *drumsticks_mid_L* y *drumsticks_tip_L*.

La diferencia entre mAP50 y mAP50-95 es notable y esperada, ya que el segundo valor evalúa el rendimiento del modelo en un rango más amplio y estricto de umbrales de intersección sobre la unión (IoU). Esto resalta la complejidad adicional que implica mantener una alta precisión en condiciones más rigurosas. En conjunto, estos resultados confirman que el modelo presenta un equilibrio adecuado entre precisión y exhaustividad, lo cual es fundamental para aplicaciones en tiempo real.

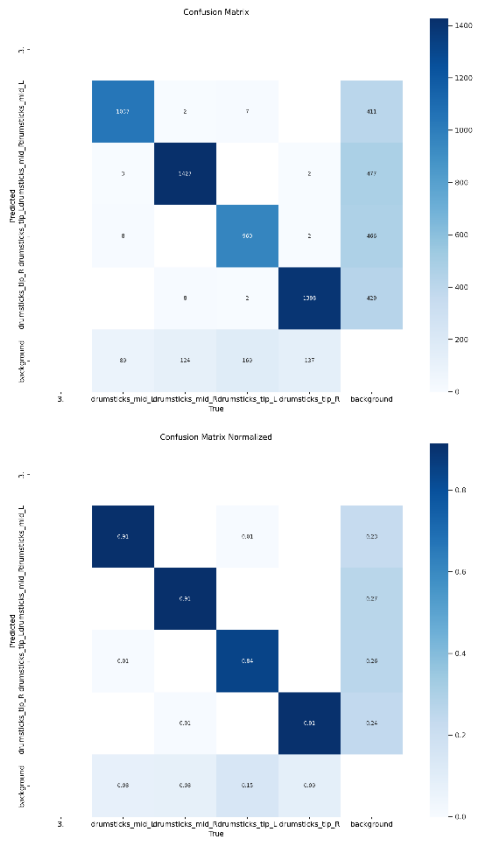
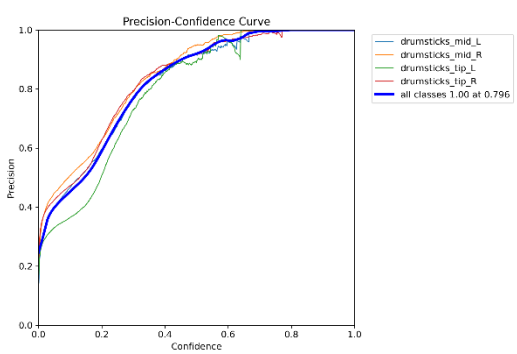


Fig. 6. Matriz de confusión

La Figura 6 muestra la matriz de confusión. La normalizada profundiza la relación entre las clases reales y las predichas por el modelo, mostrando en forma de proporciones cuántas instancias de cada clase se han clasificado correctamente (valores en la diagonal) y cuáles han sido erróneamente asignadas a otras categorías.

Podemos observar que los resultados obtenidos para todas las categorías exceptuando a *drumsticks_tip_L* fueron de 0,91 lo que quiere decir que el 91 % de las imágenes darán verdaderos positivos. En el caso de *drumsticks_tip_L* se observa que los verdaderos positivos son del 85 %, por lo que podemos inferir que reforzar el *dataset* con más imágenes de esa clase en particular puede ayudar a mejor a la detección. Sin embargo, las pruebas de funcionamiento resultaron conformes en cuanto a la detección de dicha clase en particular, porque lo que no se vio necesidad de reforzar el *dataset* y entrenar nuevamente.



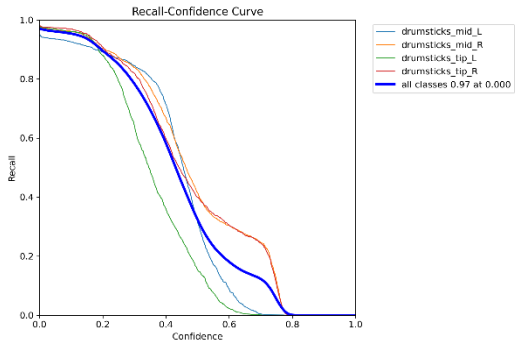


Fig. 7. Curvas de Precisión y Recall vs Confianza

La Figura 7 muestra las curvas de Precisión (P) y de *Recall* (R). La curva de Precisión (P) muestra cómo varía la precisión del modelo a medida que se modifican los umbrales de decisión. A partir de un umbral de decisión de 0,4 ya es notorio que la precisión es óptima para nuestra aplicación.

La curva de *Recall* (R) representa la capacidad del modelo para recuperar todas las instancias relevantes, evidenciando cómo se comporta al detectar verdaderos positivos. Un *recall* alto es fundamental en escenarios en los que es crucial no omitir instancias de una determinada clase. Sin embargo, este decrece a medida que el umbral de decisión aumenta, por lo que se debe realizar una relación de compromiso entre la precisión y *recall* y decidir en qué umbral funcionará mejor el modelo.

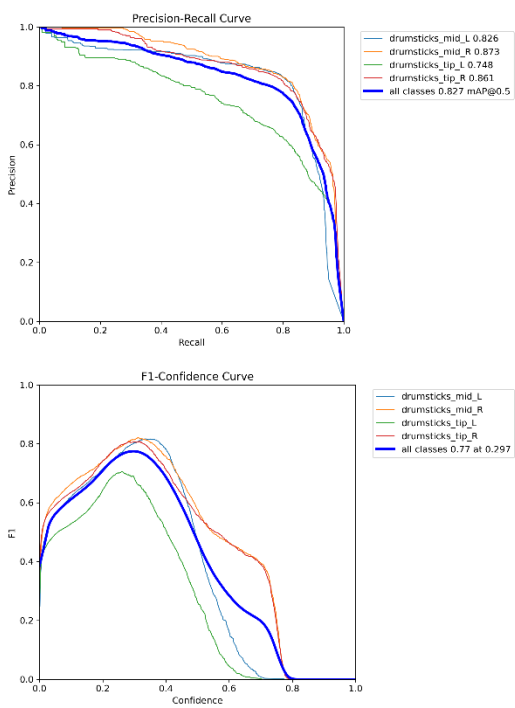


Fig. 8. Curvas de Precisión vs Recall y F1 Score

En la Figura 8 se aprecia esta relación de compromiso con la curva P vs R, donde entre los valores de 0,2 y 0,4 se encuentra el mejor equilibrio de detección para todas las clases, teniendo en cuenta que *drumsticks_tip_L* es la clase con menor *recall*.

Finalmente, la curva F1_Score presenta la evolución de la métrica F1. Una puntuación F1 elevada sugiere un buen balance entre ambas métricas. De esta forma, al confirmar que el mejor umbral para utilizar el modelo se encuentra entre los valores de 0,2 y 0,4 será posible reducir los falsos positivos.

Cálculo de posición utilizando únicamente la cámara

Como fue mencionado anteriormente, la cámara aporta lecturas absolutas de posición, pero carece de la velocidad necesaria para lograr un seguimiento preciso de los palillos. Por esta razón, en los resultados obtenidos puede verse como el seguimiento es coherente, pero falla fundamentalmente a la hora de modificar la orientación.

Para evaluar el desempeño del sistema se optaron por tres caminos diferentes para poder analizar los resultados obtenidos.

El primer camino fue el de definir un circuito señalizado fijo y conocido, el cual entrara dentro de la visión de la cámara.

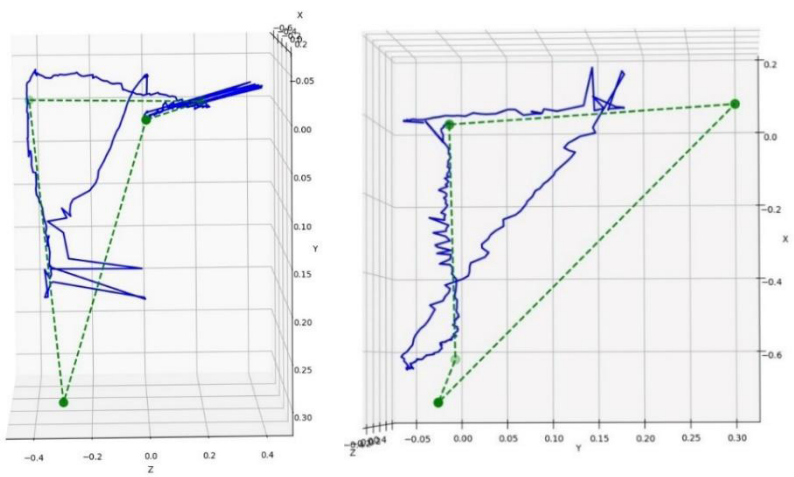
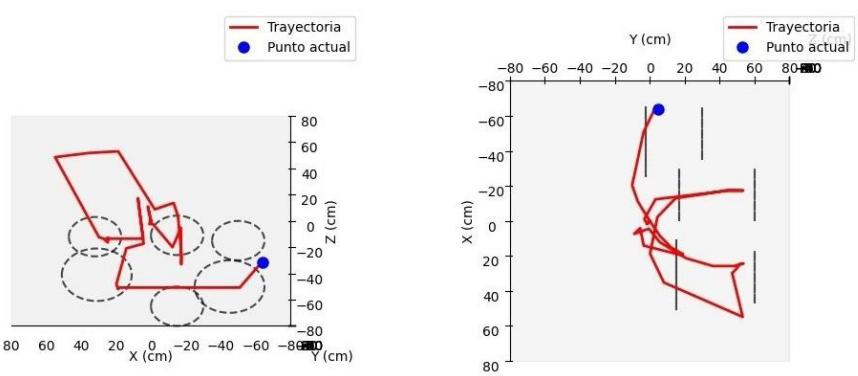


Fig. 9. Lecturas vs Circuito fijo

Como se puede ver en la Figura 9, si bien la forma es consistente, no es precisa en los puntos donde debe modificar su orientación. Asimismo, dichos puntos se encuentran en los extremos de la imagen, lo cual también generan problemas. Es relevante destacar que los movimientos efectuados fueron lentos para favorecer la precisión.

El segundo camino fue el de definir cuerpos de la batería a impactar y visualizar que tan preciso era el recorrido detectado.



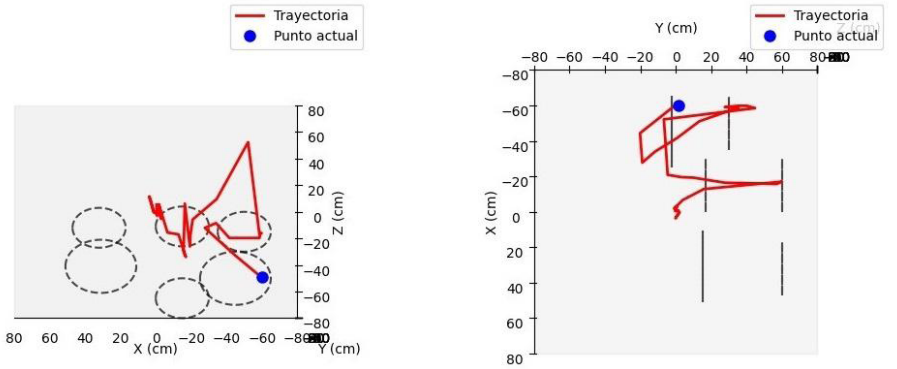


Fig. 10. Lecturas vs Circuito fijo

Como se puede ver en la Figura 10, el seguimiento parecería ser correcto, llegando efectivamente al área donde se encuentran ubicados los cuerpos de la batería. Sin embargo, mediante el tercer camino propuesto, se concluyó que esto no era así.

En este último camino, se decidió contabilizar la cantidad de golpes exitosamente detectados, dentro de una muestra de 50 golpes para cada cuerpo. De esta manera se podría definir efectivamente si el cuerpo fue detectado o no.

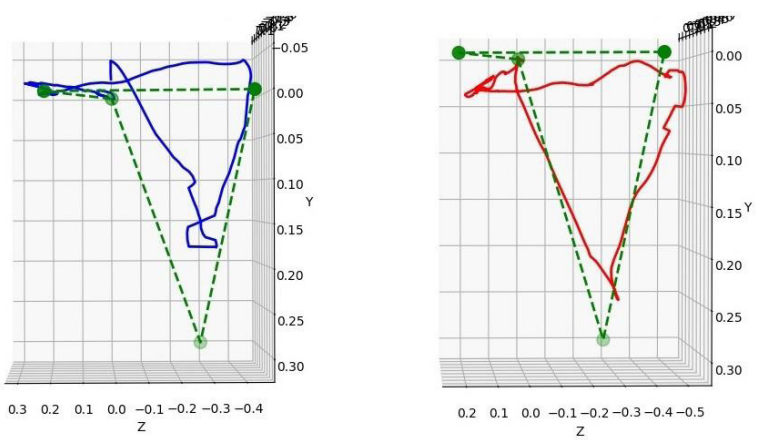
Tabla 2. Efectividad utilizando solo la cámara

Cuerpos	*Snare*	*Hi Hat*	*Crash*	*High Tom*	*Low Tom*	*Ride*
Aciertos	2	29	6	12	10	37
Equivocaciones	5	0	3	5	18	0
Total	50	50	50	50	50	50
Efectividad	4 %	58 %	12 %	24 %	20 %	74 %

Como se ve en la Tabla 2, la efectividad es muy pequeña e insuficiente.

Cálculo de posición: YOLO + IMU's

Al igual que en el apartado anterior, se tomaron los mismos tres caminos. De esta manera es posible comparar ambas metodologías y visualizar claramente los cambios.



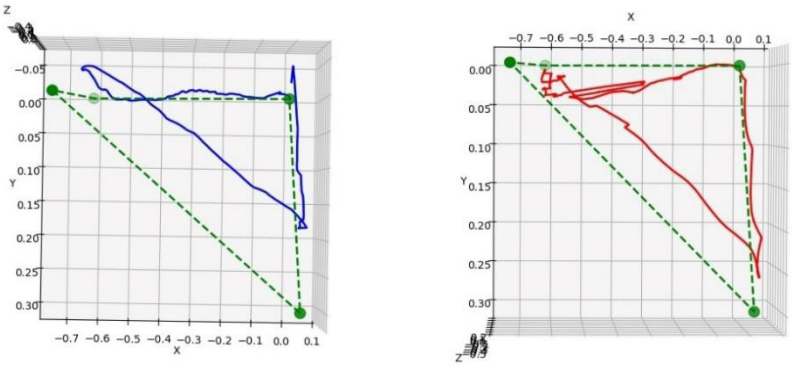


Fig. 11. Lecturas vs Circuito fijo – con filtro de media móvil

Como se ve en Figura 11, utilizando la cámara y los sensores, se logró una mayor precisión, sobre todo en los puntos donde se modifica la orientación bruscamente. No obstante, debido a que el sistema es más veloz, este cuenta con mayor ruido. En el segundo camino, no se ve una gran diferencia, tal y como se detalló anteriormente.

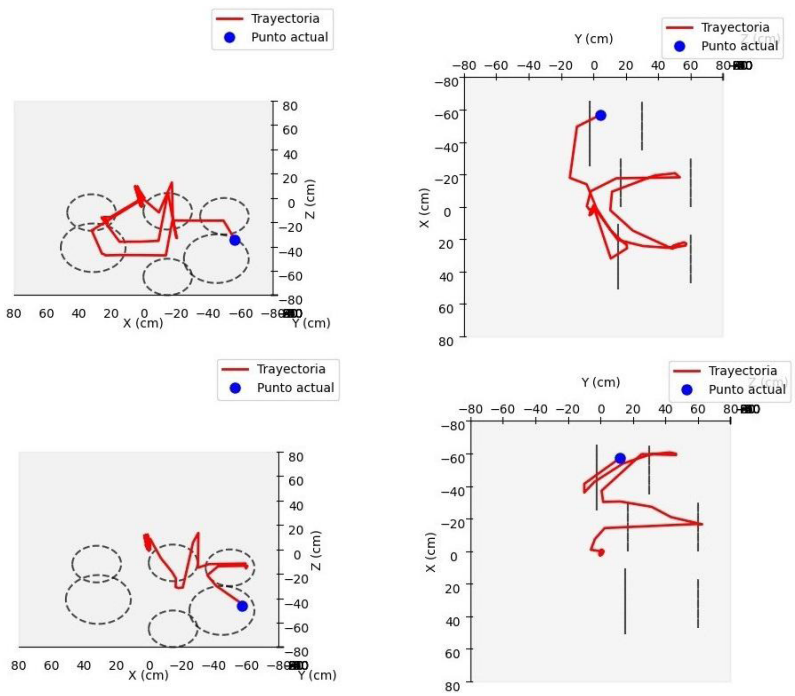


Fig. 12. Lecturas vs Circuito fijo

Sin embargo, al observar los resultados del tercer camino, se puede concluir que la tasa de aciertos crece favorablemente.

Tabla 3. Efectividad utilizando la cámara y los sensores

Cuerpos	*Snare*	*Hi Hat*	*Crash*	*High Tom*	*Low Tom*	*Ride*
Aciertos	43	40	40	47	45	41
Equivocaciones	0	0	3	5	18	0
Total	50	50	50	50	50	50
Efectividad	86 %	80 %	86 %	94 %	90 %	82 %

Conclusiones

En el presente trabajo, se desarrolló un algoritmo conformado por un filtro de Kalman de dos etapas, el cual mediante los datos obtenidos a través de sensores inerciales (IMUs) y el reconocimiento de imágenes (YOLO) tomadas con la cámara Microsoft Kinect, es capaz de realizar el seguimiento de palillos de batería.

Dicho seguimiento se logró realizar con una precisión suficiente para obtener una tasa de aciertos aproximada del 86 %, incluso al utilizar dos palillos al mismo tiempo, como es de costumbre en la práctica.

Asimismo, se demostró la mejoría al fusionar la posición calculada por la cámara y los sensores, obteniendo un resultado más preciso que al utilizar estos individualmente. Dicho avance permite abrir un campo poco explorado dentro del
seguimiento de objetos, ya que este se encuentra especializado en objetos grandes sin necesidad de mucha exactitud.

Con respecto a posibles mejorías, la utilización de mejores cámaras y sensores llevarían a resultados más precisos, al igual que un desarrollo más exhaustivo de las matrices de covarianza Q y R.

Referencias

CADENA, C.; CARLONE, L.; CARRILLO, H.; LATIF, Y.; SCARAMUZZA, D.; NEIRA, J.; REID, I. y LEONARD, J. J., (2016). "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age". En: IEEE Transactions on Robotics 32.6, págs. 1309-1332.

CHATTAH, J., (2014). "Music Instrument Digital Interface (MIDI)". En: Music in the Social and Behavioral Sciences: An Encyclopedia. SAGE Publications, págs. 749-751. URL: https://www.researchgate.net/publication/273059997_ Music_Instrument_Digital_Interface_MIDI_-_Music_in_the_Social_and_ Behavioral_Sciences.

DIEBEL, J., (2006). Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors. Inf. téc.. Stanford University.

GROVES, P. D., (2013). Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems. Waltham, MA: Artech House.

KALMAN, R. E., (1960). "A New Approach to Linear Filtering and Prediction Problems". En: Journal of Basic Engineering 82. Series D, págs. 35-45.

KAPLAN, E. D. y HEGARTY, CH. J., (2005). Understanding GPS: Principles and Applications. 2nd. Artech House.

KHOSHELHAM, K. Y ELBERINK, S. O., (2012). "Accuracy and resolution of Kinect depth data for indoor mapping applications". En: Sensors 12.2., págs. 1437-1454. DOI: 10.3390/s120201437.

KOK, M.; HOL, J. D. y SCHÖN, T. B., (2017). "Using Inertial Sensors for Position and Orientation Estimation". En: arXiv preprint arXiv:1704.06053. URL: https://arxiv.org/abs/1704.06053.

KONG, K., (2000). Inertial Navigation System Algorithms for Low Cost IMU. Inf. téc. AU: Department of Mechanical and Mechatronic Engineering, University of Sydney.

LU, F. y WU, S., (2013). "Research on Navigation System Based on GPS/INS Integrated Positioning and Kalman Filtering". En: Journal of Navigation 66.1, págs. 79-88.

MADGWICK, S. O. H., (2010). An Efficient Orientation Filter for Inertial and Inertial/Magnetic Sensor Arrays. Technical Report. Available online at https:// x-io.co.uk/res/doc/madgwick_internal_report.pdf. University of Bristol. URL: https://x-io.co.uk/res/doc/madgwick_internal_report.pdf.

MAHONY, R.; HAMEL, T. y PFLIMLIN, J.-M., (2008). "Nonlinear Complementary Filters on the Special Orthogonal Group". En: IEEE Transactions on Automatic Control 53.5.

PRATAP, M. y ENGE, P. K., (2006). Global Positioning System: Signals, Measurements, and Performance. Cambridge, MA: GangaJamuna Press.

MOURIKIS, A. I. y ROUMELIOTIS, S., (2007). "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation". En: IEEE International Conference on Robotics and Automation. US.

REDMON, J.; DIVVALA, S.; GIRSHICK, R. y FARHADI, A., (2016). "You Only Look Once: Unified, Real-Time Object Detection". En: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), págs. 779-788.

TRAWNY, N. y ROUMELIOTIS, S. I., (2005). Indirect Kalman Filter for 3D Attitude Estimation: A Tutorial for Quaternion Algebra. Inf. téc. 2005-002, Rev. 57. US: Department of Computer Science & Engineering, University of Minnesota.

WELCH, G. y BISHOP. G., (2006). An Introduction to the Kalman Filter. Inf. téc. NC 27599-3175, US: Department of Computer Science, University of North Carolina at Chapel Hill.