Estimation of Error Constant of an Electromechanical Energy Meter with Machine Vision in a Mobile Application

K. Palomino, J. Flórez, and E. Muñoz

***Abstract*—This article describes an algorithm in a smartphone designed to support the accuracy test, periodically done by network operators in energy meters. The algorithm is implemented in a mobile application with the IDE Android Studio. The applied methodology implies in the first stage video stabilization, in the second stage a novel method for the rejection of noise by illumination changes, and in the third stage the estimation of the percentual error of an electromechanical meter. The results of the application show a standard deviation of 0.1354 s, lower than the standard deviation of the manual method = 0.1966 s, which indicates a good performance of the mobile application. The experiment performed show 95% of the accuracy tests performed match the result with those performed manually**

***Index Terms*—Lighting conditions, Mobile application, Shaking hands, OpenCV, Machine vision.**

I. Introducción

COLOMBIA se halla entre los 100 países que más energía consumen en el mundo, ocupando el puesto 96 en el 2016, año que consumió 69.031 GWh y se proyectó un crecimiento en consumo de 3.5% para finales del 2017 [1]. El departamento del Cauca consumió 2.087,92 GWh durante el año 2016 [2], siendo el principal operador de red la Compañía Energética de Occidente S.A.S. E.S.P. (CEO). Esta empresa al igual que cualquier otra que preste el servicio energético debe garantizar que la medida del consumo por parte del usuario sea la correcta. Para ello se realizan una serie de verificaciones en sitio a los medidores de energía eléctrica regidos por la NTC 5900, entre estas se encuentra la prueba de exactitud que es la que indica que porcentaje de error o constante de error tiene el medidor a la hora de realizar el conteo de energía. Existen múltiples problemas para llevar a cabo esta verificación, sobre todo en los medidores electromecánicos, ya que se realiza de una forma manual empleando varios equipos (Analizador verificador de medidores, Panel Hand TGO, Cronómetro y botón emisor de pulsos) conectados entre sí y manejados por el operario para contar un numero de revoluciones (siete) y estimar la constante del error del medidor, aumentando la probabilidad de fallos humanos y también de problemas por parte de los equipos, es así que surge la necesidad de la empresa de realizar un escalamiento tecnológico hacia una aplicación de visión de máquina en un terminal tipo teléfono inteligente que en campo permita al operario realizar esas actividades. En la literatura revisada se encuentran varios trabajos académicos que proponen usar visión artificial para realizar mediciones en el área industrial y calibración de instrumentos, [3]-[11], pero estás se encuentran bajo condiciones controladas en el proceso de experimentación.

En el desafío planteado se detectan una serie de problemáticas como: posibles movimientos del operario al manipular el terminal y cambios de iluminación, ya que estas verificaciones se realizan en ambientes no controlados, por lo que el algoritmo de visión de máquina debe rechazar cambios de iluminación y suavizar los movimientos del operario para realizar el cálculo del error del medidor electromecánico. En la literatura, en [12] y [13], se plantean buenas soluciones para estabilizar video mediante hardware, dentro de las soluciones software se cuenta con dos importantes alternativas: estimación de movimiento y seguimiento de características. Para la primera, en [14], [15], [16], [17] y [18] se emplean las texturas de la imagen como vectores de movimiento para estimar una transformación afín global y luego reconstruir los marcos estables. Para la segunda, en [19] - [23] se localiza un conjunto pequeño de características confiables en marcos adyacentes para calcular el desplazamiento de la cámara y a partir de estos corregir el movimiento. El conjunto puede obtenerse a partir de "*Scale-Invariant Future Transform*" (SIFT) [24], y también con "*Speeded Up Robust Features*" (SURF) [25], [19] y [20], utilizar características locales resulta más complicado de implementar en una aplicación móvil debido a sus requerimientos computacionales. En cuanto a contrarrestar los cambios de iluminación, se cuenta con algoritmos de sustracción de fondo, en [26], [27], [28], [29] y [30] se describen técnicas de dos pasos: primero se construye un modelo que describa los objetos estáticos de la escena (*background* o modelo de fondo), y segundo se sustraen los objetos en movimiento a partir de la diferencia que se presenta en el marco de vídeo comparado con el modelo de fondo construido. En [31] se utilizan algoritmos de sustracción de fondo para el conteo y la identificación de vehículos en grandes autopistas mediante el modelo de mezcla gaussiana, hallando un modelo de fondo que se actualiza en un periodo de tiempo

Este trabajo ha sido desarrollado gracias a la Universidad del Cauca y la Compañía Energética de Occidente S.A.S E.S.P.

K. Palomino, Universidad del Cauca, Popayán, Colombia, (email: keviin_palomino@unicauca.edu.co).

J. Flórez, Universidad del Cauca, Popayán, Colombia, (email: jflorez@unicauca.edu.co).

E. Muñoz, Universidad del Cauca, Popayán, Colombia, (email: elenam@unicauca.edu.co).

corto, lo que hace que responda bien ante cambios de iluminación.

En la literatura se encuentran múltiples trabajos: [12]-[20] y [26]-[31], de cómo abordar la estabilización de video y contrarrestar los cambios de iluminación, pero ninguno es realizado en un teléfono inteligente. También se proyecta una potencial mejora en los tiempos de ejecución de las tareas que desarrollan las brigadas de la Compañía Energética de Occidente cuando en campo realizan la prueba de exactitud en medidores de energía eléctrica.

En esta investigación se diseñó un algoritmo de visión por computadora para obtener el número de revoluciones del rotor de medidores electromecánicos para determinar la constante de error porcentual, que sea robusto ante condiciones ambientales no controladas (iluminación, ángulo de lectura o vibraciones); implementando el algoritmo en una aplicación móvil y realizando pruebas de desempeño. Este documento está dividido en cuatro secciones, en primer lugar, en la introducción se plantea el problema, después se presenta la metodología con la que se abordó la solución del mismo y los materiales usados, en la tercera sección se describen las pruebas, resultados y análisis, y finalmente la conclusión a la que se llegó con el desarrollo del trabajo.

II. Metodología y Materiales

Se plantea un trabajo dividido en tres actividades: diseño, desarrollo y evaluación del algoritmo. El diseño tiene tres etapas, desarrolladas con el IDE Android Studio y funciones de la librería OpenCV de uso libre [32], se evalúa el algoritmo en un Teléfono Inteligente y en condiciones ambientales no controladas. Las etapas del algoritmo propuesto son (fig. 1):

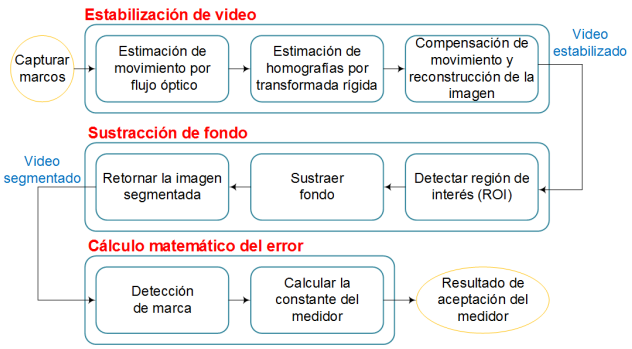

Fig. 1. Diagrama de flujo del algoritmo propuesto.

E1: se encarga de la estabilización del video, que contrarresta el problema de manos temblorosas, se implementa con un sub-algoritmo de estabilización de video basado en flujo óptico.

E2: se encarga de la sustracción de fondo que rechaza los cambios de iluminación, se implementa con un sub-algoritmo sustractor de fondo basado en el modelo de mezcla gaussiana.

E3: se encarga de realizar el cálculo matemático de la constante de error del medidor electromecánico, se cuentan un número determinado de revoluciones (7), se calcula el tiempo del evento y a partir de estos datos se calcula el error porcentual.

Los materiales que se usaron para el desarrollo de este proyecto fueron: un Smartphone Samsung Galaxy Grand Prime de 1 Gb RAM con un procesador cuatro núcleos a 1.2 GHz, librería OpenCV versión 3.1.0 para SO Android, Android Studio IDE, medidor electromecánico bifásico ISKRA E82C2, computador Toshiba Satellite C45 con 6 Gb de memoria RAM y procesador Intel Core i5 3ra Generación, tarjeta Arduino uno R3 y un motor DC de 5 Voltios.

A. Primera Etapa: Estabilización de Video

Esta etapa se basó en [33], en este se propone un algoritmo de tres sub etapas: la primera sub-etapa reconoce los puntos característicos y calcula el vector de flujo óptico; la segunda sub-etapa calcula el desplazamiento cuantitativo de estos puntos entre marcos adyacentes y la tercera sub-etapa suaviza la trayectoria, con la información obtenida del flujo óptico, obteniendo un video estable.

Para la primera, se usó el "detector de esquinas Shi-Tomasi" [34], e implementado con la función de la librería de OpenCV *goodfeaturestotrack()*, determina los puntos característicos de la imagen en los cuales se presentan grandes variaciones de intensidad en todas las direcciones, estos puntos también son conocidos como esquinas, ver Fig. 2.

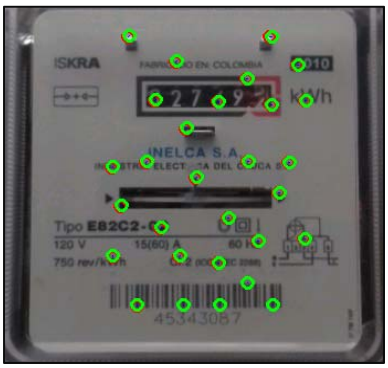

Fig. 2. Resultados de aplicar el detector de esquinas (algoritmo Shi-Tomasi) en carátula de un medidor.

En [24] plantearon matemáticamente como detectar la diferencia en intensidad para un desplazamiento (u, v) en todas las direcciones, ver (1).

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \qquad (1)$$

Donde w(x,y) es una ventana que asigna un peso a la vecindad de los píxeles, $I(x, y)$ es la intensidad en (x,y) y $I\ (x + u, y + v)$ es la intensidad con un desplazamiento determinado por $(x + u, y + v)$.

Se buscan ventanas en las cuales exista una esquina, es decir puntos característicos, para dar con las grandes variaciones de intensidad [24], proponiendo maximizar "$([(x + u, y + v) - (x, y)]^2)$" de (1) usando la expansión de Taylor expresada en forma matricial (M), ver (2):

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \qquad (2)$$

A partir de los auto valores de M (λ_1, λ_2), se determina si una

ventana puede contener una esquina o no (3):

$$R = \min(\lambda_1, \lambda_2) \quad (3)$$

Cuando R es mayor a un umbral (usualmente 3.5) la región es una esquina. Cuando R es pequeña, la ventana se califica como borde o como plana.

Se detectan un máximo de 30 esquinas, con una calidad mínima calidad de 0.01 y una distancia euclidiana de 20 píxeles.

Seguido a esto se calcula el vector de flujo óptico en marcos adyacentes, definido como el movimiento aparente de los objetos, en una secuencia de imágenes, esto se representa en un vector de desplazamiento (para cada punto) que muestra el movimiento de los puntos entre dos marcos, para realizar este procedimiento se usa el algoritmo de Lucas-Kanade [34], se implementa mediante la función *calcOpticalFlowPyrLK()*, ver Fig. 3. En una representación aplicada de esta sub etapa del algoritmo ver Fig. 4, se aprecia el punto característico en el marco anterior (círculo verde), en el marco actual (círculo rojo) y el vector de flujo óptico (línea que los conecta).

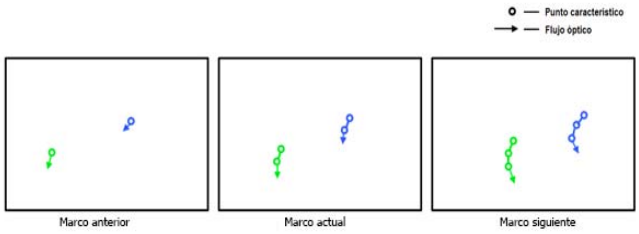

Fig. 3. Representación del vector de flujo óptico.

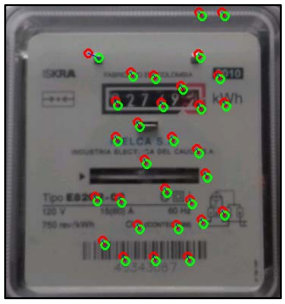

Fig. 4. Resultado del cálculo del vector de flujo óptico (Algoritmo Lucas Kanade) en carátula de un medidor.

En la segunda sub-etapa se estima el desplazamiento con ayuda de los vectores de flujo óptico calculados previamente. Una secuencia de imágenes tiene distintos tipos de desplazamiento, generado por el movimiento del objeto o el movimiento de la cámara, entre marcos consecutivos, para el desarrollo de este proyecto se contemplaron tres tipos de movimiento: Translación en los ejes X y Y, rotación, y escalamiento uniforme, ver Fig. 5.

Se propone hallar los factores de desplazamiento calculando transformaciones rígidas o afín. Con esta operación se obtienen varias matrices con los factores de desplazamiento, cada una con los datos de los posibles movimientos anteriormente descritos. Es válido notar que estas transformaciones permiten calcular los datos de desplazamiento de hasta seis grados de libertad, que se resumen en cuatro movimientos: translación, rotación, cizallamiento y escalamiento, pero para el caso de la implementación solo se calcularon transformaciones parciales, que constan de 4 grados de libertad (rotación, translación y escalamiento uniforme).

Fig. 5. Movimientos considerados para el proyecto en la careta de un medidor (a. Imagen original, b. Translación en ejes (x,y), c. Rotación y d. Escalamiento uniforme).

Se propone realizar el cálculo de transformaciones parciales y omitir el cizallamiento, ya que este fenómeno es el menos propenso a ser presentado. El cálculo de estas matrices se realiza por medio de la función *estimateRigidTransform()*.

Finalmente, la tercera sub etapa emplea el método "Suavizado de ventanas de promedio" [35], donde se utiliza un procedimiento básico para acumular parámetros de trayectoria y luego suavizarlos usando una ventana de promedio, este método trata de llevar el punto característico del marco actual a la coordenada en donde se encontraba el mismo punto en el marco anterior, se implementa mediante la función *warpAffine()*, este método deforma la escena que se esté mostrando en el marco, ver Fig. 6.

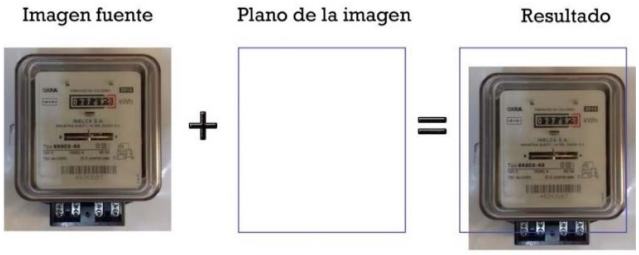

Fig. 6. Potencial resultado del algoritmo estabilizador de imagen en carátula de un medidor.

B. Segunda etapa: Sustracción de Fondo

Usa el algoritmo de Zoran-Zivkovic [36] siendo una mejora del trabajo publicado por el mismo autor [37] e implementado con la función *BackgroundSubtractorMOG2()*. Esta función

determina si cada pixel muestreado pertenece al marco del primer plano (*ForeGround*, FG) o al marco del fondo (*BackGround*, BG) [26], teniendo en cuenta que el valor de cada píxel muestreado en el tiempo *t* en RGB o algún otro espacio de colores se denota $\vec{x}^{(t)}$, la decisión de a que marco pertenece cada píxel está dada por la "decisión bayesiana", ver (4).

$$R = \frac{p(BG|\vec{x}^{(t)})}{p(FG|\vec{x}^{(t)})} = \frac{p(\vec{x}^{(t)}|BG)p(BG)}{p(\vec{x}^{(t)}|FG)p(FG)} \quad (4)$$

Los resultados de la sustracción de fondo se propagan a algunos módulos de nivel superior, por ejemplo, los objetos detectados son rastreados, mientras se rastrea un objeto se puede obtener algún conocimiento sobre la apariencia del objeto rastreado y este conocimiento puede ser usado para mejorar la sustracción de fondo [37]. Como un caso general en el que no se sabe nada acerca de los objetos de primer plano, cuándo se pueden ver, ni con qué frecuencia se van a presentar en escena se establece que $p(FG) = p(BG)$ y se asume una distribución uniforme para la apariencia del objeto en primer plano $p(\vec{x}^{(t)}|FG) = c_{FG}$, hay la certeza que el píxel pertenece al fondo si y solo si:

$$p(\vec{x}^{(t)}|BG) > C_{thr} \quad (5)$$

$p(\vec{x}^{(t)}|BG)$ hace referencia al modelo de fondo y C_{thr}es un umbral. Este modelo se estima a partir de un conjunto de entrenamiento denotado como χ. El modelo estimado se denotará por $\hat{p}(\vec{x}^{(t)}|\chi, BG)$ y se obtiene mediante el método de Mezcla de Gaussianas (MoG) [36] el cual consiste en modelar la intensidad de los píxeles con una mezcla de k distribuciones Gaussianas (donde k es un número pequeño, frecuentemente se utiliza de 3 a 5) definidas por los siguientes parámetros: media, varianza y peso. El algoritmo se aplica solamente a la región de interés, ver Fig. 7.

Fig. 7. Región de interés (enmarcada en un rectángulo verde).

Para reconocer esta única zona en una imagen más limpia se realiza pre procesamiento digital, se binariza la imagen para aplicar transformaciones morfológicas de apertura y cierre asegurando que los contornos estén completos y se hayan removido las partículas más pequeñas de la imagen, a continuación se usa la función de reconocimiento de contornos *findContours()*, que implementa el algoritmo propuesto por Satoshi Suzuki en [38], el cual facilita el reconocimiento de bordes en una imagen digital, posteriormente se condiciona una sola región acorde a la forma del rectángulo, para ello se calcula el área de cada contorno hallado con el apoyo de la función *moments()* basada en el "Teorema de Green", esta retorna un valor de área para cada contorno, después se comparan entre si y el contorno elegido será el de mayor área, a esta se le dibuja un rectángulo acotado a su alrededor para diferenciarlo del resto de área, considerando 20 marcos para estimar el fondo y un valor de umbral de 200 se aplica la función *BackgroundSubstractorMOG2()*, ver Fig. 8.

Fig. 8. Sustracción de fondo aplicada a la región de interés.

C. *Tercera etapa: Cálculo Matemático del Error*

En esta etapa final ya se cuentan con los algoritmos de procesamiento de imagen que permiten identificar y diferenciar la zona de interés del medidor, en particular la marca del disco giratorio, aquí es necesario que el operario intervenga indicándole a la aplicación el inicio, para ello usa un botón alarma. Esto permite asegurar que la región detectada sea la correcta. Para realizar la medición del tiempo se modificó la función *chronometer()* de la librería de Android Studio (ya que esta no realiza conteo de milésimas de segundo, fundamental para la prueba de exactitud) y para el conteo de las revoluciones se usó un contador que se activa cada vez que la marca sea detectada. Para identificar la marca se usa la función *moments()*, que calcula el área de los objetos dentro de la región de interés. Solamente se cuenta cuando el área de dicho objeto pase un umbral, eliminando falsos conteos que son generados por el ruido del video. Una vez el contador llegue a siete se detienen estas dos funciones y con el dato del tiempo se calcula N_e y posteriormente la constante de error porcentual. Ver (6) y (7).

$$e_{pe} = \frac{N_e - N_r}{N_r} x100 \quad (6)$$

Donde:
N_r: es el número de impulsos registrados por el dispositivo contador durante la calibración (para este tipo de contador siempre son 7 impulsos o revoluciones).
N_e: es el número de impulsos del patrón esperados durante el periodo de calibración, este valor se calcula con la ecuación (7).

$$N_e = \frac{K_d\ I\ U\ T}{2700000} \quad (7)$$

K_d : Constante del medidor en $\frac{imp}{kWh}$.
I : Corriente en amperios.
U : Tensión en voltios.
T : Tiempo de la duración de la prueba.

III. Resultados y Discusión

Se realizó un conjunto de tres pruebas: para el algoritmo de la etapa 1, de la etapa 2 y para el algoritmo global.

A. Pruebas Etapa 1

Se graban una serie de videos con el teléfono inteligente sostenido por el operario aproximadamente a diez centímetros del medidor en funcionamiento, para evaluar el desempeño de la estabilización de la secuencia de video.

Estos videos son procesados por el algoritmo de estabilización y se obtiene un video estable, (Fig. 9). Se extraen los datos de desplazamiento promediados de los puntos característicos en uno de los videos antes y después de aplicar el algoritmo estabilizador, en los ejes (x, y, θ).

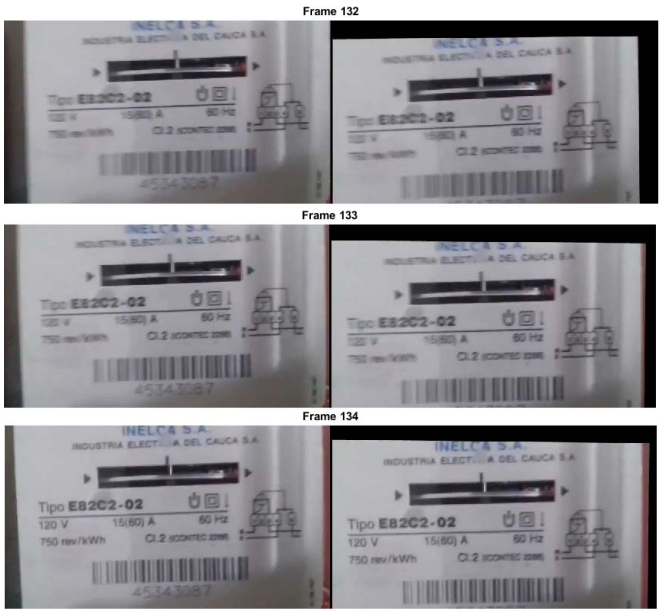

Fig. 9. Aplicación del algoritmo de estabilización de video: izq. Marcos del video original, der. Marcos del video estabilizado.

Se observa que las curvas del video estabilizado son más suaves que las curvas del video original, permitiendo una estabilización de video (Fig. 10), el algoritmo logra suavizar desplazamientos bruscos o vibraciones en los ejes (x, y) de hasta ± 20 pixeles, al igual que variaciones de cambios de ángulo alrededor de 10 grados, considerando una ventana de 20 marcos. (Ejemplos de estabilizaciones de video están disponibles en los enlaces https://goo.gl/kaJuKu, https://goo.gl/yqSgLw.).

B. Pruebas Etapa 2

Se graban una serie de videos al medidor con el teléfono inteligente a una distancia aproximada de diez centímetros y se aplican cambios de iluminación artificial, con el fin de adicionar o retirar dicha iluminación (Fig. 11).

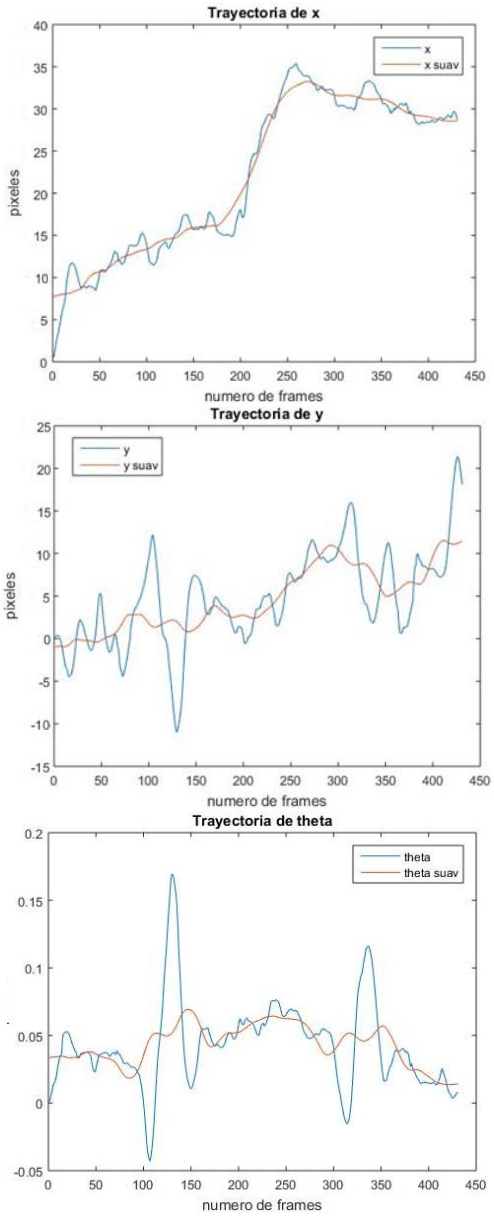

Fig. 10. Trayectorias de los ejes (x, y, θ) de un video antes (azul) y después (rojo) de ser estabilizado.

Se observa como el algoritmo de sustracción de fondo de la etapa 2, (Fig. 11) rechaza el ruido provocado por los cambios de iluminación. En el marco a) se tiene una escena con luz natural, mientras que la misma escena en el marco c) se le aplica iluminación artificial con ayuda de una linterna, lo que provoca que se genere ruido dentro de la región de interés (rectángulo verde), siendo este un resultado indeseable, pero en e) el algoritmo sustractor de fondo es capaz de contrarrestar dichos cambios de iluminación disminuyendo estos ruidos, esto pasa un marco después de que se adiciona luz artificial. En la columna derecha de imágenes se tiene otra evaluación esta vez realizada inversamente, es decir en el marco b) la escena presentada cuenta con iluminación artificial y en d) dicha iluminación es retirada, provocando ruido dentro de la región de interés, y en el marco f) (un marco después) dicho ruido es

reducido por el algoritmo. En las secuencias de prueba (Fig. 11), considerando un marco de referencia iluminado con luz ambiente con un nivel intensidad promedio de 125/256 en escala de grises, los cambios de iluminación realizados producen variaciones de la intensidad promedio hasta en ±100/256, las pruebas muestran que el algoritmo es capaz de contrarrestar dichos cambios de iluminación.

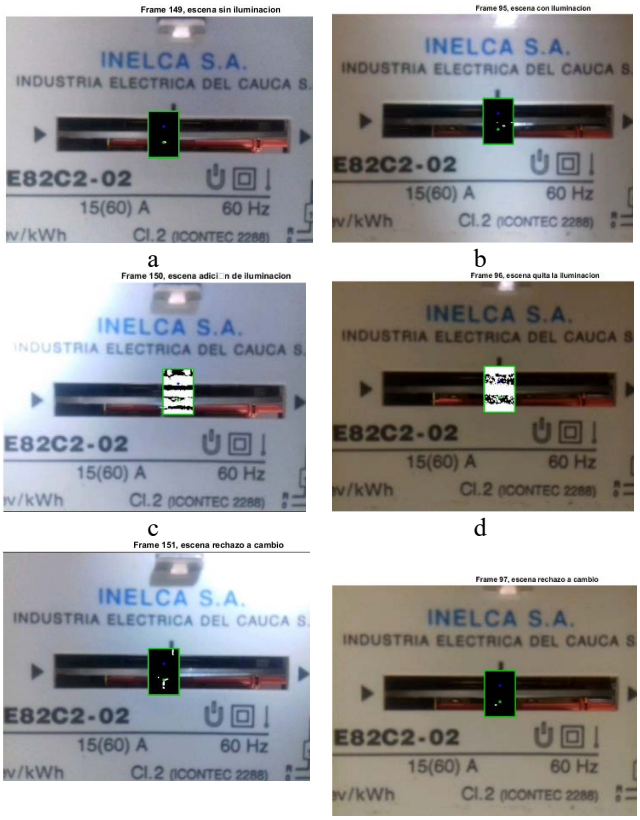

Fig. 11. Resultados de la prueba de cambios de iluminación (izq.: marcos con adición de luz artificial, der.: sustracción de luz artificial).

(Ejemplos variaciones de iluminación están disponibles en los siguientes enlaces https://goo.gl/xNkBMG, https://goo.gl/s9GdA4.)

C. *Pruebas Algoritmo Global*

Para la evaluación global se adaptó al eje de giro del disco del medidor electromecánico un moto reductor de 5 VDC, controlando su velocidad de giro por una tarjeta Arduino Uno R3. Se establecieron tres valores de PWM alrededor ($\pm 6\%\ de\ PWM$) de un tiempo ideal (PWM = 25%) para la prueba de exactitud (despejando T de (6) y (7) cuando $e_{pe} = 0$ ($T_{ideal} = 35\ seg$) y se contabilizó diez veces el tiempo para cada valor de PWM aplicado al motor que gira el disco. Se promedia dicho tiempo y se establece una "medida patrón" con el fin de comprobar la robustez del algoritmo, ver TABLA I y Fig. 12. Posteriormente se aplican 10 variaciones de PWM en el rango del 19% al 31%, midiendo los tiempos manualmente (cronómetro) y la aplicación móvil, ver TABLA II y Fig. 13.

TABLA I
VARIACIONES DE PWM Y ESTIMACIÓN DE LA MEDIDA PATRÓN

MEDICIÓN	PWM (19%) TIEMPO (S)	PWM (25%) TIEMPO (S)	PWM (31%) TIEMPO (S)
1	41.2	34.5	28.8
2	40.8	35.2	29.2
3	41.0	34.3	28.9
4	41.1	34.2	29.5
5	41.2	34.4	29.4
6	40.7	35.3	29.4
7	40.8	34.4	29.6
8	40.7	34.1	29.2
9	41.2	35.4	29.6
10	40.5	34.1	29.5
PROMEDIO	40.92	34.59	29.31

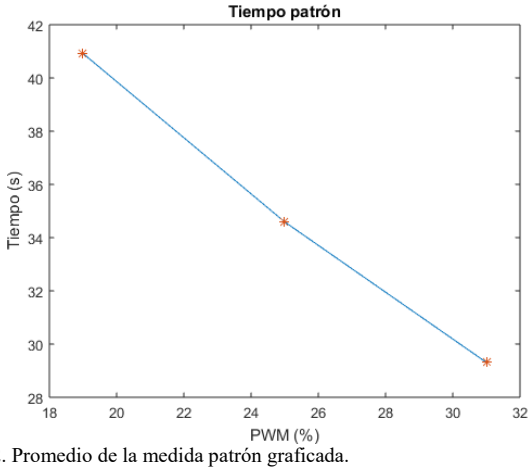

Fig. 12. Promedio de la medida patrón graficada.

TABLA II
COMPARACIÓN DE TIEMPO MANUAL VS TIEMPO DE LA APLICACIÓN MÓVIL

PWM	T. APP. (S)	T. CRON. (S)
1 (20%)	39.37	40.42
2 (21%)	38.96	38.89
3 (22%)	37.96	38.15
4 (23%)	37.09	37.55
5 (24%)	36.02	36.25
6 (26%)	34.25	34.26
7 (27%)	33.06	33.19
8 (28%)	31.56	32.05
9 (29%)	31.07	30.70
10 (30%)	30.52	30.25

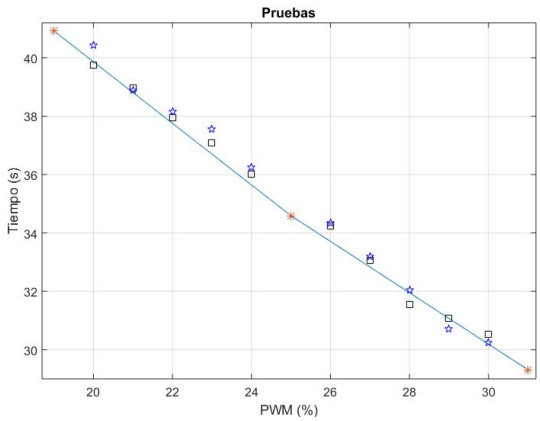

Fig. 13. Comparación de los tiempos medidos manualmente (cuadrado negro) y tiempos medidos con la aplicación (estrella azul).

Se observa que los valores obtenidos mediante la aplicación (Fig. 13), en su mayoría, se acercan a la medida patrón. Se calculó la desviación estándar de estas medidas en comparación con la medida patrón con la finalidad de estimar la dispersión de las medidas de la aplicación y cuantificar su desempeño, ver TABLA III.

TABLA III
DESVIACIONES ESTÁNDAR PARA MEDIDAS MANUALES Y MEDIDAS DE LA APLICACIÓN MÓVIL

PWM	T. APP. (S)	T. CRON. (S)	T. PATRÓN (S)	σ APP (S).	σ MANUAL (S)
1 (20%)	39.37	40.42	39.86	0.054	0.280
2 (21%)	38.96	38.89	38.81	0.074	0.039
3 (22%)	37.96	38.15	37.75	0.104	0.199
4 (23%)	37.09	37.55	36.70	0.194	0.424
5 (24%)	36.02	36.25	35.64	0.190	0.305
6 (26%)	34.25	34.26	33.71	0.269	0.274
7 (27%)	33.06	33.19	32.83	0.114	0.180
8 (28%)	31.56	32.05	31.95	0.195	0.050
9 (29%)	31.07	30.70	31.07	0	0.185
10 (30%)	30.52	30.25	30.19	0.160	0.030

Calculando los promedios de las desviaciones estándar se tiene:

- Promedio desviación estándar, de las mediciones realizadas con la aplicación móvil = 0.1354 s.
- Promedio de desviación estándar de las mediciones realizadas manualmente = 0.1966 s.

La aplicación en promedio es más exacta en la toma de medidas, en comparación a como se realizaría manualmente.

También con estos datos se calcula la correlación lineal entre estos valores (tiempos app vs tiempos manuales), ver Fig. 14.

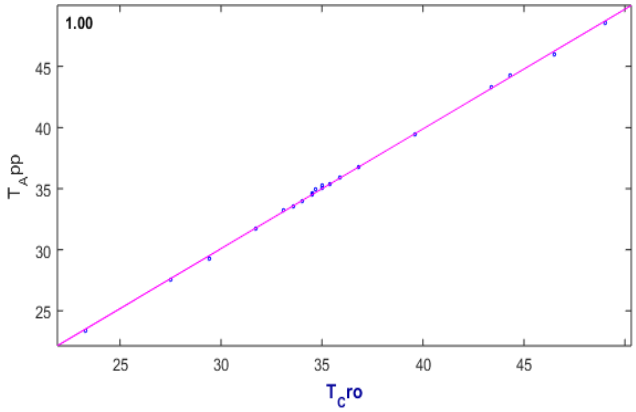

Fig. 14. Correlación entre tiempos obtenidos mediante la aplicación y tiempos obtenidos de medición manual.

Se obtuvo una correlación igual a 1, que significa que los resultados de la aplicación tienen una fuerte correlación con los resultados calculados manualmente.

Para calcular la constante de error porcentual de la prueba de exactitud se reemplaza el T medido con la aplicación o el cronometro en (7) y se determina N_e cuando $K_d = 1000$ imp/kWh, $I = 4.5$ A y $U = 120$ V. Con el valor de N_e y siendo $N_r = 7$ revoluciones, se determina e_{pe}, si este no sobrepasa el $\pm 5\%$ se puede decir que el medidor está correctamente calibrado, si este error supera el error máximo permitido la empresa procede a trasladar el medidor al laboratorio, ver TABLA IV.

TABLA IV
CALCULO DEL e_{pe} DEL MEDIDOR PARA LAS MEDIDAS MANUALES Y MEDIDAS DE LA APLICACIÓN MÓVIL

PWM	ERROR APP.	ERROR CRON.	ERROR PATRÓN
1 (20%)	12,4850	15,4850	13,8885
2 (21%)	11,3140	11,1140	10,8850
3 (22%)	8,4570	9,0000	7,8570
4 (23%)	5,9710	7,2850	4,8570
5 (24%)	2,9140	3,5710	1,8280
6 (26%)	-2,1420	-2,1140	-3,6850
7 (27%)	-5,5420	-5,1710	-6,2000
8 (28%)	-9,8280	-8,4280	-8,7140
9 (29%)	-11,2280	-12,2850	-11,2280
10 (30%)	-12,8000	-13,5710	13,7420

De los datos de la TABLA IV se observa que solo para PWM 4 la aplicación determina que se supera el error máximo permitido, mientras que el error patrón indicaría que el medidor está correctamente calibrado. Notar que en la TABLA III, la mayor desviación estándar se obtiene para PWM 6, pero la respuesta de la prueba de exactitud no se ve afectada.

Para ampliar las pruebas, se realizan 20 mediciones adicionales, se calculó la constante de error porcentual de manera manual y con la aplicación, se observa que solo la prueba 5 no coincide con los resultados, es decir mientras que manualmente el cálculo indicó que la prueba era negativa, la aplicación concluyo un resultado positivo para el medidor, se logra un 95% de resultados exitosos, ver TABLA V.

TABLA V
PRUEBAS ADICIONALES DE CÁLCULO e_{pe} DEL MEDIDOR PARA LAS MEDIDAS MANUALES Y MEDIDAS DE LA APLICACIÓN MÓVIL

PRUEBA	ERROR APP.	ERROR CRON.	PRUEBA	ERROR APP.	ERROR CRON.
1	-4%	-4.26%	11	-9.28%	-9.28%
2	32.86%	31.33%	12	-1.29%	-1.45%
3	-5.29 %	-5.20 %	13	-1.29%	-1.06%
4	-2.71	-2.93%	14	1.28%	1.07%
5	5.29%	4.91%	15	24%	23.87%
6	0%	0.67%	16	26.71%	26.4%
7	-16%	-16.40	17	-33.29%	-33.2%
8	2.57%	2.57%	18	0%	0.27%
9	13.29%	12.67	19	40%	38.8%
10	-21.29%	-21.20%	20	-1.43	-0.133

IV. CONCLUSIONES

Se diseñó e implementó un algoritmo de visión de máquina en una aplicación móvil capaz de asistir en la prueba de exactitud (NTC 5900) realizada por la Compañía Energética de Occidente S.A.S. E.S.P. Este brinda un error de medición menor a la medida manual, a su vez la aplicación móvil hace el proceso de medición repetible y reproducible independientemente del operario y brigada de turno.

Los resultados obtenidos al evaluar la aplicación móvil globalmente fueron mejores que los resultados obtenidos si la prueba de exactitud se realizara manualmente, teniendo una desviación estándar promedio igual a 0.1354 para los datos

tomados con la aplicación móvil y una desviación promedio igual a 0.1966 para los datos obtenidos de forma manual en comparación a una medida patrón.

Con el fin de evaluar los alcances de esta clase de algoritmos, se propone como investigación futura evaluar las mediciones realizadas, en distintos escenarios con variables de disturbio parametrizadas como: iluminación, cambios de ángulo y/o vibraciones durante la medición. De igual forma analizar el desempeño del algoritmo, contrastando distintos métodos de detección de esquina.

AGRADECIMIENTOS

Los autores agradecen al ingeniero Jorge Millán director de innovación y proyectos de la Compañía Energética de Occidente y a la Universidad del Cauca por el apoyo recibido para realizar este proyecto.

REFERENCIAS

[1] UPME, "Eléctrica y Potencia Máxima en Colombia Revisión Junio de 2016," *Sielgov*, p. 55, 2016.

[2] UPME, "Proyección de la demanda de energía eléctrica y potencia en Colombia," *Sielgov*, 2016.

[3] F. Alegria and A. Serra, "Automatic calibration of analog and digital measuring instruments using computer vision," *IEEE Trans. Instrum. Meas.*, vol. 49, no. 1, pp. 94–99, 2000.

[4] S. L. Pang and W. L. Chan, "Computer vision application in automatic meter calibration," *Fourtieth IAS Annu. Meet. Conf. Rec. 2005 Ind. Appl. Conf. 2005.*, vol. 3, pp. 1731–1735, 2005.

[5] Q. Li, Y. Fang, Y. He, F. Yang, and Q. Li, "Automatic reading system based on automatic alignment control for pointer meter," in *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*, 2014, pp. 3414–3418.

[6] G. Andria, G. Cavone, L. Fabbiano, N. Giaquinto, and M. Savino, "Automatic Calibration System for Digital Instruments Without Built-in Communication Interface," pp. 857–860, 2009.

[7] O. Chang, E. Pruna, M. Pilatasig, I. Escobar, and L. Mena, "Calibration of residential water meters by using computer vision," *CHILECON 2015 - 2015 IEEE Chil. Conf. Electr. Electron. Eng. Inf. Commun. Technol. Proc. IEEE Chilecon 2015*, pp. 351–356, 2016.

[8] Y. Yang *et al.*, "EAST-AIA deployment under vacuum: Calibration of laser diagnostic system using computer vision," *Fusion Eng. Des.*, vol. 112, pp. 563–568, 2016.

[9] R. Lu and M. Shao, "Sphere-based calibration method for trinocular vision sensor," *Opt. Lasers Eng.*, vol. 90, no. October 2016, pp. 119–127, 2017.

[10] P. A. Belan, S. A. Araujo, and A. F. H. Librantz, "Segmentation-free approaches of computer vision for automatic calibration of digital and analog instruments," *Measurement*, vol. 46, no. 1, pp. 177–184, 2013.

[11] C. Zheng, S. Wang, Y. Zhang, P. Zhang, and Y. Zhao, "A robust and automatic recognition system of analog instruments in power system by using computer vision," *Measurement*, vol. 92, pp. 413–420, 2016.

[12] S. Chang, Y. Zhong, Z. Quan, Y. Hong, J. Zeng, and D. Du, "A Real-time Object Tracking and Image Stabilization System for Photographing in Vibration Environment using OpenTLD Algorithm," *Adv. Robot. its Soc. Impacts (ARSO), 2016 IEEE Work.*, pp. 0–4, 2016.

[13] L. Pettazzi, E. Fedrigo, R. Muradore, P. Haguenauer, and L. Pallanca, "Improving the accuracy of interferometric measurements through adaptive vibration cancellation," in *2015 IEEE Conference on Control and Applications, CCA 2015 - Proceedings*, 2015, no. July 2014, pp. 95–100.

[14] G. Puglisi and S. Battiato, "A robust image alignment algorithm for video stabilization purposes," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 10, pp. 1390–1400, Oct. 2011.

[15] O. K. O. Kwon, J. S. J. Shin, and J. P. J. Paik, "Edge based adaptive Kalman filtering for real-time video stabilization," *2006 Dig. Tech. Pap. Int. Conf. Consum. Electron.*, pp. 75–76, 2006.

[16] B. H. Chen *et al.*, "Improved global motion estimation via motion vector clustering for video stabilization," *Eng. Appl. Artif. Intell.*, vol. 54, pp. 39–48, Sep. 2016.

[17] L. M. Fuentes, S. A. Velastin, and S. Member, "Vigilancia Avanzada : del tracking a la detección de sucesos," IEEE Am. Lat., vol. 2, no. 3, pp. 206–211, 2004.

[18] A. G. S. S. C. Felipussi, "ratoca – Mouse via Detecção de Marcadores por Câmera de Vídeo em Ambiente Não Controlado," IEEE Am. Lat., vol. 4, no. 6, pp. 443–448, 2006.

[19] S. Battiato, G. Gallo, G. Puglisi, and S. Scellato, "SIFT features tracking for video stabilization," in *Proceedings - 14th International conference on Image Analysis and Processing, ICIAP 2007*, 2007, pp. 825–830.

[20] K. Feng, H. Yonghua, and Z. Huaxiong, "Video stabilization based on multi-scale local color invariants," in *Proceedings of the International Conference on Networking and Distributed Computing, ICNDC*, 2014, pp. 65–69.

[21] J. V. C. I. R and Y. G. Lee, "Novel video stabilization for real-time optical character recognition applications q," *J. Vis. Commun. Image Represent.*, vol. 44, pp. 148–155, 2017.

[22] A. C. Bernal, D. L. A. Ojeda, and M. A. I. Manzano, "Object Detection from Range Imagen Using the Sparse Keypoint Detector Technique," IEEE Am. Lat., vol. 16, no. 5, pp. 1532–1538, 2018.

[23] R. F. V. Y. V Shkvarko, S. Member, and A. F. Problema, "Máxima Entropía y de Análisis Variacional para Reconstrucción de Imágenes de Percepción Remota," IEEE Am. Lat., vol. 3, no. 4, pp. 362–375, 2005.

[24] D. G. Lowe, "Object recognition from local scale invariant features," *Proc. Seventh IEEE Int. Conf. Comput. Vis.*, vol. 2, no. [8], pp. 1150–1157 vol.2, 1999.

[25] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[26] L. Di Stefano, F. Tombari, and S. Mattoccia, "Robust and accurate change detection under sudden illumination variations," *ACCV'07 Work. Multidimensional Multi-view Image Process. Tokyo, Nov., 2007 MM-P-02*, pp. 103–109, 2007.

[27] S. Kumar and J. Sen Yadav, "Video object extraction and its tracking using background subtraction in complex environments," *Perspect. Sci.*, 2016.

[28] Z. Zeng, J. Jia, Z. Zhu, and D. Yu, "Adaptive maintenance scheme for codebook-based dynamic background subtraction," *Comput. Vis. Image Underst.*, vol. 152, pp. 58–66, 2016.

[29] G. Gemignani and A. Rozza, "A Robust Approach for the Background Subtraction Based on Multi-Layered Self-Organizing Maps," vol. 25, no. 11, pp. 5239– 5251, 2016.

[30] H. Bhaskar, K. Dwivedi, D. P. Dogra, M. Al-Mualla, and L. Mihaylova, "Autonomous detection and tracking under illumination changes, occlusions and moving camera," *Signal Processing*, vol. 117, pp. 343–354, 2015.

[31] K. Saran and S. G, "Traffic Video Surveillance : Vehicle Detection and Classification," *2015 Int. Conf. Control. Commun. Comput. India*, no. November, pp.
516–521, 2015.

[32] Open CV. (2019, May 5). Libraries [Online]. Avalaible: https://docs.opencv.org/2.4/

[33] Y. Wang, Z. Hou, K. Leman, and R. Chang, "Real Time Video Stabilization for Unmanned Aerial Vehicles," *Conf. Mach. Vis. Appl.*, no. April, pp. 336– 339, 2011.

[34] C. T. Jianbo Shi, "Good Features to Track," *IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 593–600, 1994.

[35] Y. Kuroki, K. Takenaka, and S. Motomatsu, "Robust keypoint detection against affine transformation using moment invariants on intrinsic mode function," in *2015 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, 2015, pp. 403–407.

[36] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," *Proc. 17th Int. Conf. Pattern Recognit.*, vol. 2, no. 2, p. 28–31 Vol.2, 2004.

[37] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognit. Lett.*, vol. 27, no. 7, pp. 773–780, 2006.

[38] S. Suzuki and K. be, "Topological structural analysis of digitized binary images by border following," *Comput. Vision, Graph. Image Process.*, vol. 30, no. 1, pp. 32–46, 1985.

Kevin Palomino received the Engineering degree in Industrial Automation Engineering from Universidad del Cauca - Popayán, Cauca, Colombia, in 2018, and is an aspiring master in mechatronic systems at the University of Brasilia, has developed several works in the area of computational vision.

Elena Muñoz received the Engineering degree in Electronics and Telecommunications Engineering, with a Master's Degree in Electronics Engineering from Universidad del Cauca. Practicing teaching at the Universidad del Cauca since 1997, guiding subjects in the area of control systems, and computer vision; directing at the same time degree works related to machine vision, and medical image processing.

Juan Flórez received the Engineering degree in Electronics and Telecommunications Engineering, specializing in Telematics Networks and Services, and in Industrial Computing, with a Master's Degree in Electronics Engineering from Universidad del Cauca. Practicing teaching at the Universidad del Cauca since 1998, guiding subjects in the area of instrumentation, control, and automation; directing at the same time degree works related to ISA, ISO and IEEE standards, energy efficiency in the industry, and industrial process automation methodology.