Desarrollo de una aplicación para la detección y seguimiento de objetos mediante la transformada de Hough

Patricio Andrés Jaramillo Torres
Sistemas Informáticos y Computación, UTPL
Universidad Técnica Particular de Loja
Loja, Ecuador
pajaramillo2@utpl.edu.ec

Manuel Fabricio Coronel Camacho
Geología y Minas, UTPL
Universidad Técnica Particular de Loja
Loja, Ecuador
mfcoronel2@utpl.edu.ec

***Resumen*—** El seguimiento de objetos es un campo que se ubica dentro de la visión artificial, es base en diversas aplicaciones tanto en el área de video vigilancia, robótica, entretenimiento, entre otras. El seguimiento se maneja por ciertos puntos que son clave como es la detección de objetos en movimiento, el seguimiento del objeto en cada momento y el análisis respectivo. De acuerdo a lo mencionado en este artículo se describirán algunas características fundamentales de la librería de visión artificial y código abierto OpenCV, esta librería brinda el marco de trabajo de alto nivel para el desarrollo de aplicaciones de visión por computador en tiempo real. Con el objetivo de crear un juego entretenido para niños, se tomara como base los algoritmos como CAMShift, transformada de Hough y Haarcascade. Para el seguimiento se aplican técnicas de filtrado lo que da como resultado una segmentación del objeto de interés, y presenta un menor nivel de ruido que facilita obtener mejores resultados en la detección y seguimiento.

***Palabras clave*—** OpenCV, seguimiento de objetos, CAMShift, procesamiento de imágenes, visión asistida por computador.

I. INTRODUCIÓN

El seguimiento de objetos es una tarea relevante dentro del campo de la visión por computador con la proliferación de computadoras de alta performance, la disponibilidad de cámaras de vídeo de alta calidad y de bajo costo, y la creciente necesidad de análisis de video de manera automatizado ha generado un gran interés en los algoritmos de seguimiento de objetos [*1*]. En su forma más simple, el seguimiento puede ser definido como el problema de la estimación de la posición y trayectoria de un objeto en el plano de la imagen mientras se mueve alrededor de una escena [*2*]. Nuestro trabajo de investigación se ha enfocado en desarrollar una aplicación que emplea algoritmos para la detección de movimiento en video. Esto ha dado origen a dos trabajos, uno enfocado en la optimización con respecto al tiempo de respuesta y otro enfocado en la optimización con respecto a la robustez de seguimiento. Este trabajo propone un algoritmo de seguimiento que se enfoca en optimizar tiempos de procesamiento para seguimiento en tiempo real de objetos que se encuentran sobre un fondo con baja frecuencia espacial de color. Dado que la característica a seguir del objeto es el color, el algoritmo se basa en la reconstrucción iterativa de su histograma por medio de puntos exploratorios que toman muestras de color [*3*]. Dichos puntos exploratorios se distribuyen aleatoriamente dentro de la región de interés donde se halla el objeto. Se realiza un ajuste continuo al histograma patrón que caracteriza al objeto incorporando a este en forma iterativa a cada frame las variaciones de brillo y color producidos a causa de los cambios de iluminación, tamaño y perspectiva que sufre el mismo propio del movimiento con respecto al dispositivo de captura de imagen. Se considera solo una cantidad de puntos exploratorios menor al 20% del tamaño de la región de interés que encierra al objeto a seguir. De esta manera en comparación con la cantidad de puntos analizados que consideran los algoritmos estándar de filtro de partículas se permite optimizar el tiempo de procesamiento. Con estos puntos exploratorios se reconstruye el histograma patrón que caracteriza al objeto de manera iterativa a fines de maximizar su correspondencia al objeto y garantizar su seguimiento [*4*].

II. TRABAJOS RELACIONADOS

La librería OpenCV proporciona un marco de trabajo apropiado de acuerdo a las necesidades de seguimiento y reconocimiento de objetos, por lo que permitió realizar el desarrollo de una aplicación basada en la visión por computador. OpenCV posee numerosas librerías de apoyo que son de fácil y rápida implementación, y permite obviar aspectos como son características de la cámara, formatos de imagen o video.

La librería OpenCV proporciona varios paquetes de alto nivel para el desarrollo de aplicaciones de visión por computador. Todos ellos se pueden agrupar en librerías de C/C++ dirigidas a usuarios avanzados y en herramientas de scripting dirigidas, en este caso, a usuarios de nivel medio (ideal para practicar con las distintas técnicas de procesamiento de imágenes y visión [*2*].

A continuación se describen tres algoritmos de la librería OpenCV que permiten la detección y seguimiento de objetos, entre estos algoritmos tenemos los siguientes:

A. Transformada de Hough

La transformación de Hough es una técnica de extracción de características utilizada en análisis de imágenes, visión por computadora y procesamiento de imágenes digitales [*5*]. El

propósito de la técnica es encontrar instancias imperfectas de objetos dentro de una cierta clase de formas mediante un procedimiento de votación. Este procedimiento de votación se lleva a cabo en un espacio de parámetros, del que se obtienen los candidatos de objeto como máximos locales en un espacio denominado acumulador que se construye explícitamente por el algoritmo para calcular la transformada de Hough.

La transformación clásica de Hough se refería a la identificación de líneas en la imagen, pero más tarde la transformación de Hough se ha extendido a identificar posiciones de formas arbitrarias, generalmente círculos o elipses [*6*].

El objetivo de la transformada de Hough es encontrar puntos alineados que puedan existir en la imagen, es decir, puntos en la imagen que satisfagan la ecuación de la recta, para distintos valores de ρ y θ [*7*].

$$\rho = x \cdot cos\ \theta + \cdot sen\ \theta \quad (1)$$

Por tanto hay que realizar una transformación entre el plano imagen (coordenadas x-y) y el plano o espacio de parámetros (ρ,θ). Para aplicar la transformada de Hough es necesario discretizar el espacio de parámetros en una serie de celdas denominadas celdas de acumulación. Esta discretización se realiza sobre los intervalos (ρ_{min}, ρ_{max}) y (θ_{min}, θ_{max}).

Aunque la versión de la transformación descrita anteriormente se aplica sólo a la búsqueda de líneas rectas, se puede utilizar una transformación similar para encontrar cualquier forma que pueda ser representada por un conjunto de parámetros. Un círculo, por ejemplo, puede ser transformado en un conjunto de tres parámetros, representando su centro y radio, de modo que el espacio de Hough se convierta en tridimensional. También se pueden encontrar elipses y curvas arbitrarias de esta manera, como cualquier forma fácilmente expresada como un conjunto de parámetros.

Alterar el algoritmo para detectar formas circulares en lugar de líneas es relativamente sencillo. En un espacio bidimensional, un círculo puede ser descrito por:

$$(x + a)^2 + (y + b)^2 = r^2 \quad (2)$$

Donde (a, b) es el centro del círculo, y *r* es el radio. Si un punto 2D (x, y) es fijo, entonces los parámetros se pueden encontrar según. El espacio de parámetros sería tridimensional, (a, b, r). Y todos los parámetros que satisfacen (x, y) estarían sobre la superficie de un cono invertido de ángulo recto cuyo vértice está en (x, y, 0). En el espacio 3D, los parámetros del círculo pueden ser identificados por la intersección de muchas superficies cónicas que están definidas por puntos en el círculo 2D. Este proceso se puede dividir en dos etapas. La primera etapa es el radio de fijación y luego encontrar el centro óptimo de los círculos en un espacio de parámetros 2D. La segunda etapa es encontrar el radio óptimo en un espacio de parámetros unidimensional.

Este método también puede detectar círculos que están parcialmente fuera del espacio del acumulador, siempre que haya suficiente área del círculo dentro de ella.

B. CAMShift

El Camshift (continuamente adaptativo Mean Shift) es un algoritmo de segmentación de imágenes de color introducida por Gary Bradski en 1998, el Camshift explota hábilmente el algoritmo de media por desplazamiento cambiando el tamaño de la ventana cuando se trataba de la convergencia . El Camshift acoplado es una adaptación a las secuencias de imágenes en color, y es operado en la búsqueda de objetos en tiempo real. Una implementación libre de este algoritmo se encuentra en la biblioteca de software de visión por ordenador OpenCV.

Para el diseño del algoritmo de seguimiento de objetos propuesto se realiza una implementación basada en la funcionalidad CAMShift de OpenCV [*8*], en el cual se agregan características para mejorar la detección del objeto y el seguimiento. Estas mejoras se basan en el filtrado de la imagen inicial dentro de la ROI inicial y en la detección del color del objeto a seguir.

En las siguientes secciones se detallan las características de cada una de las partes del algoritmo [*9*].

1. Inicialización: Se adquiere la imagen y se realiza la transformación de espacio de colores de RGB a HSV. Esto permite calcular el histograma inicial en base a el matiz del color una vez que es seleccionada la zona donde se encuentra el objeto a seguir (sección A) .

2. Cálculo de histograma: Durante esta etapa se realizan los cálculos de comparación y promedio ponderado del histograma de color.

Para observar discrepancias entre histogramas es necesario obtener un valor numérico que indique el grado de relación entre ellos. Entonces, dadas las distribuciones normales p y q, la distancia de Bhattacharyya está definida:

$$\mathrm{d(p,q)} = \frac{1}{4} ln\left(\frac{1}{4}\left(\frac{\sigma_p^2}{\sigma_q^2} + \frac{\sigma_p^2}{\sigma_p^2} + 2\right)\right) + \frac{1}{4}\left(\frac{\mu_p^2 + \mu_q^2}{\sigma_p^2 + \sigma_q^2}\right) \quad (3)$$

Donde μ_p, σ_p, μ_q y σ_p son la media y el desvío estándar de las distribuciones p y q respectivamente.

3. Seguimiento: Recibe como parámetro el histograma inicial perteneciente a la etapa de inicialización, y en esta fase realiza los cálculos correspondiente a la posición del objeto dando como resultado una nueva zona o región de interés (region of interest, ROI). Esta información se realimenta al

bloque cálculo de histograma para obtener el valor del histograma promedio y así, nuevamente, determinar la posición actualizada del objeto en seguimiento.

C. *Haarcascade*

Para la detección de Objetos utilizando Haar basados en características clasificadores en cascada es un método de detección de objetos efectivo propuesto por Pablo Viola y Michael Jones en su artículo, "Detección de Objetos rápido usando una cascada de Impulsado Simple Features" [*10*]. Se trata de un enfoque basado en el aprendizaje de las máquinas donde una función de cascada es entrenado de una gran cantidad de imágenes positivas y negativas. Entonces se utiliza para detectar objetos en otras imágenes.

Las características tipo Haar se definen sobre regiones rectangulares de una imagen en escala de grises. Una característica está formado por un número finito de rectángulos y su valor escalar consistirá la suma de los pixeles de cada rectángulo sumados aplicando un cierto factor de peso [*11*].

$$característica_j = \sum_{1 \le i \le N} wi * suma\ rectangulo\ (r_i) \quad (4)$$

Donde $\{r_1 \ldots\ldots\ldots r_N\}$ son los rectángulos que forman la característica y w_i el peso de cada uno. Por ejemplo El número de características diferentes que se pueden generar para una imagen de 21x21 pixeles es de más de 90000.

Como hemos visto, se pueden definir más de 90000 características diferentes en una ventana de 21x21 pixeles modificando el tamaño de las características tipo Haar y su polaridad. Sumar uno a uno los pixeles de cada uno de los diferentes rectángulos tiene un coste computacional proporcional al ´área del rectángulo pero los autores propusieron un método para calcular los valores en tiempo constante a partir de lo que ellos llamaron la imagen integral.

.La imagen integral es una representación alternativa para la imagen que se puede calcular de manera muy rápida [*12*]. Un valor de la imagen integral ii(x,y) será igual al valor del pixel i(x,y) de la imagen sumado a todos los pixeles de la imagen que estén a la izquierda y arriba de la posición (x,y).

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y') \quad (5)$$

La imagen integral se puede calcular iterativamente comenzando por la posición (0,0). El pixel de abajo a la derecha contiene la suma de todas las intensidades de los pixeles de la imagen.

$$s(x,y) = s(x,y-) + i(x,y) \quad (6)$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \quad (7)$$

Gracias a esta representación podemos calcular la suma de los pixeles de un rectángulo de cualquier tamaño utilizando únicamente 4 accesos a memoria. Para calcular el valor de un rectángulo A cuya esquina superior izquierda está en (x,y) y cuyo tamaño es (sx,sy) únicamente necesitamos acceder al valor de la imagen integral de 4 esquinas.

III. DESARROLLO DE LA APLICACIÓN

A. *Requerimientos*

Para esta aplicación se establecen dos requerimientos principales:

1. Desarrollar una aplicación capaz de diferenciar y reconocer objetos en movimiento.

2. Dar seguimiento en tiempo real al objeto detectado.

Para ello se utiliza varias técnicas de procesamiento de imágenes, algoritmos básicos de visión por ordenador y la biblioteca OpenCV, como su nombre lo indica, es una biblioteca de visión por computadora de código abierto. OpenCV es bastante fácil de usar si usted tiene conocimientos básicos de procesamiento de imágenes.

B. *Desarrollo*

El desarrollo de esta aplicación se basa en dos aspectos fundamentales: la detección de las circunferencias de colores y el seguimiento de las mismas en tiempo real. Para ello se estableció los siguientes pasos.

El proceso inicia con el filtrado del color de interés del objeto a detectar en la imagen (ver figura 1).

Figura 1 Imagen original

A continuación se convierte la imagen en el espacio de color HSV (ver figura 2), seguidamente se usa el filtro cvInRange dos veces para filtrar los colores de interés en este caso azul (ver figura 3).

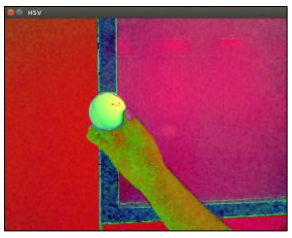


Figura 2 Espacio de color HSV

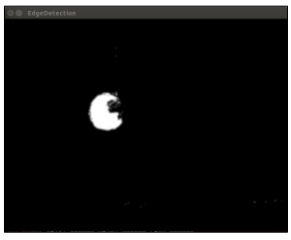


Figura 3 Filtrado del color azul

Para la el seguimiento del objeto se utilizó la transformada de Hough, la cual determina la forma del objeto en este caso

un círculo (Ver figura 4). Antes de aplicar la transformada de Hough se debe suavizar la imagen, ya que parece mejorar los resultados.

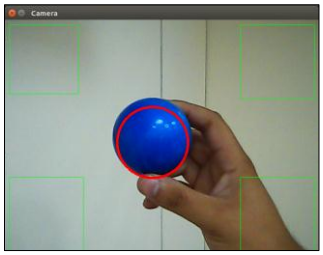


Figura 4 Detección y seguimiento del objeto

Seguidamente se procedió a establecer cuatro cuadros en los vértices de la imagen resultante mediante la función de openCV cvRectangle en la cual especificamos la coordenada inicial y final (ver figura 4). A continuación se determinó el centro del objeto detectado, con el fin de que cuando el objeto ingrese al cuadro establecido con las coordenadas inicial y final mediante la función cvRectangle este rellene el cuadrado de color (ver figura 5).

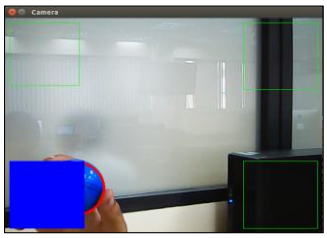


Figura 5 Resultado del ingreso del objeto al cuadrado

C. Resultados y pruebas

El resultado es bastante impresionante. El código funciona muy bien y detecta el objeto en distintas circunstancias y condiciones como movimiento lento, movimiento rápido, distancia de la cámara (lejos de la cámara, cerca de la cámara, etc.) iluminación.

TABLA 1 TABLA DE VALIDACIÓN A 1M

COLOR	Iluminación		Distancia (cámara-esfera)	Movimiento (esfera)	
	Alta	*Baja*	*1m*	*Lento*	*Rápido*
Amarillo	✓	✓	✓	✓	✓
Verde	✓	✓	✓	✓	✓
Azul	✓	✓	✓	✓	✓

TABLA 2 TABLA DE VALIDACIÓN A 2M

COLOR	Iluminación		Distancia (cámara-esfera)	Movimiento (esfera)	
	Alta	*Baja*	*2m*	*Lento*	*Rápido*
Amarillo	✓	✓	✓	✓	✓
Verde	✓	✓	✓	✓	✓
Azul	✓	✓	✓	✓	✓

TABLA 3 TABLA DE VALIDACIÓN A 3M

COLOR	Iluminación		Distancia (cámara-esfera)	Movimiento (esfera)	
	Alta	*Baja*	*3m*	*Lento*	*Rápido*
Amarillo	✓	✓	✓	✓	✓
Verde	✓	✓	✓	✓	✓
Azul	✓	✓	✓	✓	✓

TABLA 4 TABLA DE VALIDACIÓN A 4M

COLOR	Iluminación		Distancia (cámara-esfera)	Movimiento (esfera)	
	Alta	*Baja*	*4m*	*Lento*	*Rápido*
Amarillo	X	X	X	X	X
Verde	X	X	X	X	X
Azul	X	X	X	X	X

Dentro de las pruebas que se realizaron se logró determinar la distancia máxima en la cual la aplicación es capaz de detectar las esferas de color amarillo, verde y azul con el respectivo seguimiento en tiempo real de las mismas. Como se observa en las Tablas 1, 2 y 3 la aplicación funciona correctamente entre las distancias 1m a 3m. Pasada esta distancia la aplicación ya no reconoce el objeto.

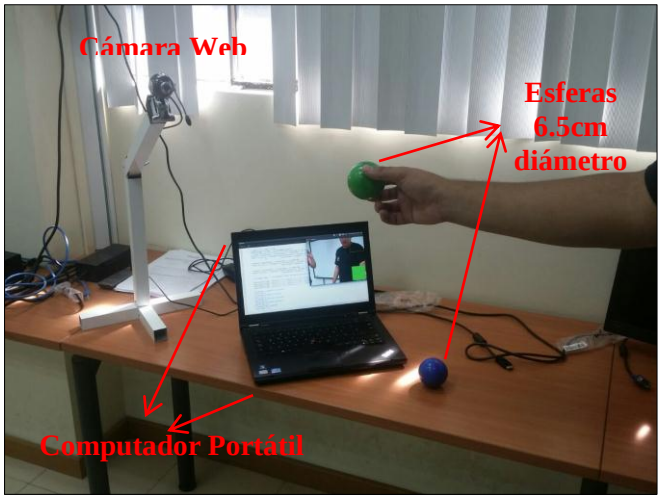
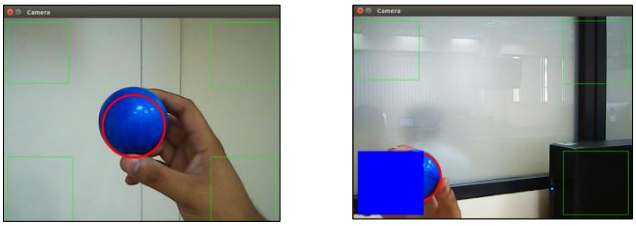
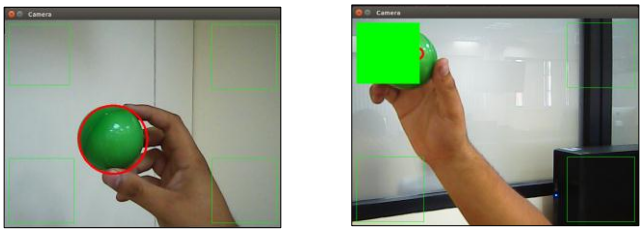


Figura 6 Componentes de la Aplicación



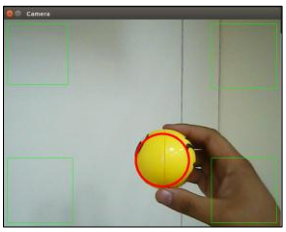
(a) (b)

Figura 7 Resultado objeto color azul



(a) (b)

Figura 8 Resultado objeto color verde



(a) (b)

Figura 9 Resultado objeto color amarillo

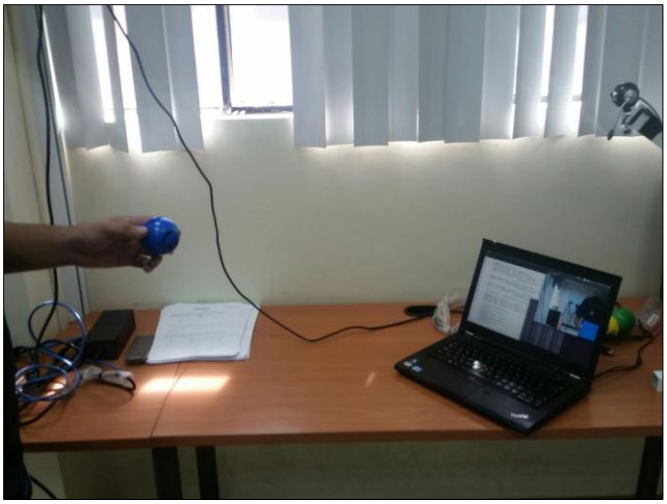


Figura 10 Prueba de distancia 1m (cámara-esfera)

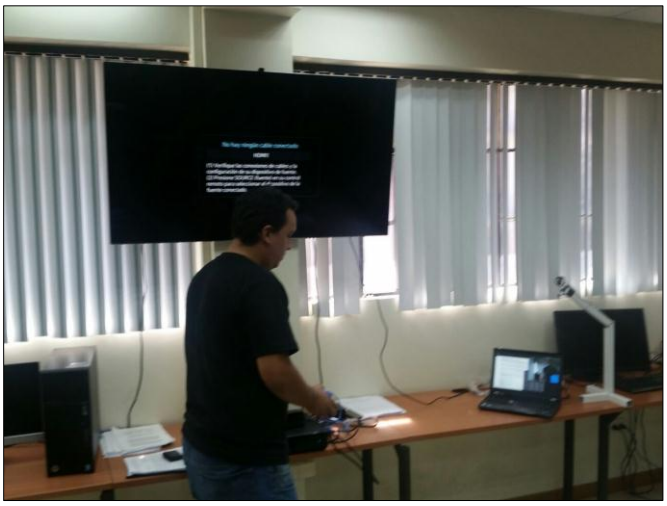


Figura 11 Prueba de distancia 2m (cámara-esfera)

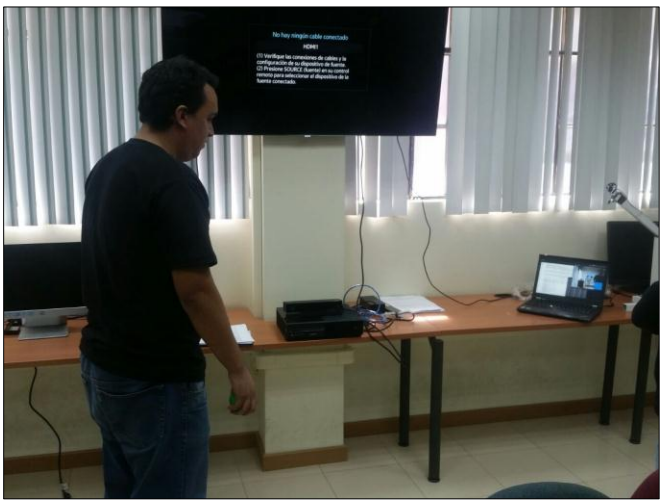


Figura 12 Prueba de distancia 3m (cámara-esfera)

IV. Conclusiones

- La aplicación está dirigida a niños, con el propósito de mejorar su motricidad a través de un juego.
- La aplicación se desarrolló para detectar y seguir los objetos en tiempo real para los colores azul, verde y amarillo obteniendo muy buenos resultados.
- La aplicación tiene un 100% de confiabilidad en cuanto a sus resultados obtenidos para las distancias de 1m, 2m y 3m.
- Pasado la distancia de los 3m en relación cámara web y esferas, la aplicación presenta dificultades para detectar las esferas para los colores amarillo, verde y azul.

Referencias

[1] Christian Sánchez Montero, "Sistema de Seguimiento de Objetos Mediante Agrupación de Trayectorias en Terminales Móviles," *Universidad de Alcalá*, Junio 2015.

[2] J. González, G.Ambrosio V. M. Arevalo, "La Librería Artificial OpenCV Aplicación a la Docencia e Investigación," *Dpto. De Ingeniería de Sistemas y Automática, Universidad de Málaga.*, 2002. [Online]. http://mapir.uma.es/varevalo/drafts/arevalo2004lva1.pdf

[3] Cesar A. Romero Cesar A. Díaz. (2011) Congreso Internacional de Mecatrónica. [Online]. http://revistas.unab.edu.co/index.php?journal=mecatronica&page=article&op=viewArticle&path%5B%5D=1500

[4] M. Marufo, L. Di Matteo, R. Verrastro, A. Hernandéz, J. Gomez, C. Verrastro M. Prieto, "Algoritmo de seguimiento de objetos en imágenes mediante reconstrucción iterativa de hisyograma en tiempo real," *Grupo de Ingeniería Artificial y Robótica. Univ. Técnologica Nacional Fac. Bs.*, p. 6.

[5] George Stockman Linda Shapiro, "Computer Vision," *The University of Washington*, Marzo 2000. [Online]. ftp://91.193.237.1/pub/docs/linux-support/computer%20science/computer%20vision/Computer%20Vision%20-%20Linda%20Shapiro.pdf

[6] Peter E. Hart Richard O. Duda, "Use of the Hough Transformation to Detect Lines and Curves in Pictures ," *Stanford Research Institute, Menlo Park, California*, vol. 15, no. 11-15, January 1972.

[7] Peter E. Hart Richard O. Duda, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Artificial Intelligence Center. SRI International*, Abril 1971. [Online]. http://www.dtic.mil/dtic/tr/fulltext/u2/a457992.pdf

[8] Gary R. Bradski, "Computer Vision Face Tracking For Use in a Perceptual User Interface," *Intel Technology Journal*, vol. No Q2, 1998. [Online]. http://opencv.jp/opencv-1.0.0_org/docs/papers/camshift.pdf

[9] R. Verrastro, L. Di Matteo, M. Prieto, M. Marufo, J. Gomez, C. Verrastro A. Hernandéz, "Algoritmo de seguimiento de objetos basado en visión asistida por computador en tiempo real utilizando CAMShift e histogramas ponderados," *Grupo de Inteligencia Artifical y Robótica. Univ. Técnologica Nacional Fac. Regional Bs.*, November 2014.

[10] Michael Jones Paul Viola, "Rapid object detection using a boosted cascade of simple features," *Computer Vision and Pattern Recognition*, 2001.

[11] Michael Oren, Tomaso Poggio Constantine P. Papageorgiou, "A general framework forobject detection," *Sixth International Conferen-ce on*, pp. 555-562, 1998. [Online]. http://turing.iimas.unam.mx/~elena/CompVis/Papageogiou98.pdf

[12] Valveny Ernest, "Imagen Integral," *Departamento de ciencias de la computacion. Universidad Autónoma de Barcelona*. [Online]. https://es.coursera.org/learn/deteccion-objetos/lecture/CROQ4/l5-3-imagen-integral