Algoritmo de visión monocular para detección de obstáculos

M.C. Rogelio Ferreira Escutia
rferreir@itmorelia.edu.mx

Víctor Manuel Eslava Echagaray
canibalito@gmail.com

Cesar Ibarra Mejía
neoraseck@gmail.com

Aldo González Castro
morgot_agc_2000@yahoo.com

Instituto Tecnológico de Morelia
Av. Tecnológico #1500
Col. Lomas de Santiaguito
Tel. (443) 312 15 70 ext. 233
www.itmorelia.edu.mx

***Resumen:** El Algoritmo de Visión Monocular desarrollado para la Detección de Obstáculos, en un ambiente controlado, no requiere de una comparación entre imágenes a diferencia de otros algoritmos. Las imágenes capturadas, se transforman a través de distintos procesos de tratamiento de imagen denominados Filtros, con la finalidad de determinar que obstáculos están presentes en el entorno. La implementación del Algoritmo se desarrolló en JAVA, un lenguaje de multiplataforma que ofrece una amplia portabilidad.*

Palabras clave: algoritmo, detección de objetos, procesamiento de imágenes, vehículo autónomo, visión computacional.

I. Introducción.

En el área de la mini robótica uno de los retos a resolver por un robot móvil es, la detección de obstáculos. Se han utilizado varios tipos de sensores; algunos de ellos se activan al momento de la colisión entre el robot y el obstáculo, otros más sofisticados son costosos y de difícil implentación, además resultan escasos en los mercados de algunas regiones. Una solución ha sido el desarrollo de algoritmos de visión que se puedan implementar con dispositivos de alta disponibilidad en la mayoría de los mercados, de bajo costo y de código abierto. Este trabajo pretenden ser un aliciente para que día a día se involucren más personas en esta área.

Este algoritmo para la detección de obstáculos se diseño en forma modular, con la finalidad de permitir el desarrollo de nuevos módulos que mejoren su precisión y funcionalidad.

Para construir el módulo de procesamiento de imágenes se utilizó una arquitectura de diseño por capas, en el cual, cuando se finaliza un proceso en una de las capas la información generada pasa a la siguiente capa donde sufre otro proceso y sus datos de salida genera los datos de entrada para la siguiente, y así sucesivamente hasta llegar a la capa final.

ROC&C'2006 – CP-16 PONENCIA RECOMENDADA POR EL **COMITÉ DE COMPUTACIÓN** DEL **IEEE SECCIÓN MÉXICO** Y PRESENTADA EN LA **REUNIÓN DE OTOÑO, ROC&C'2006**, ACAPULCO, GRO., DEL 28 DE NOVIEMBRE AL 3 DE DICIEMBRE DE 2006.

la imagen realiza procesos para el tratamiento de la imagen. A dichos procesos también se les hará referencia como Filtros (Escala Grises y Filtro Binario). Una vez que la imagen pasó por dichos filtros la siguiente fase consiste en diferenciar que objeto es un obstáculo y cual no.

II. Funcionamiento del Algoritmo.

2.1 Captura de Imagen.

Como ya se ha mencionando, el Algoritmo comienza con la toma de imágenes del entorno que será analizado.

Para obtener estas imágenes se utilizó la librería Javax.media. La imagen captada por la cámara es almacenada en el formato JPG (**Joint Photographic Experts Group**), debido a que permite la visualización de imágenes en millones de colores; Este formato utiliza algoritmos de compresión de información, por lo que los archivos generados ocupan poco espacio de almacenamiento y requieren un tiempo menor de procesamiento, además es considerado como uno de los mejores formatos para la fotografía digital.

En la figura 1, se muestran los procesos efectuados y la secuencia del Algoritmo de detección de obstáculos.

Figura 1: Pasos del Algoritmo.

Figura 2 Diagrama de flujo del algoritmo

2.2 Escala de Grises

Una vez capturada la imagen, se procedió a tratarla, siendo el primer paso, la conversión de la imagen original a una imagen en escala de grises.

La razón por la cual se hace esta transformación, que una imagen en escala de grises solo consta de 256 tonos, siendo esto muy significativo si se compara con la imagen original que consta de 24 bits (16.7 Millones de colores). Esto reduce considerablemente el proceso realizado para la detección de obstáculos y optimiza la imagen para el siguiente filtro. En la figura 3 se muestra la transformación de la imagen original.

Figura 3: Imagen Original transformada a Escala de Grises.

Posteriormente se da paso al siguiente proceso, conocido como Filtro Binario.

2.3 Filtro Binario.

El filtro consiste básicamente, en tomar la imagen en escala de grises y transformarla a una imagen de dos colores (Blanco y Negro), por eso recibe el nombre de filtro Binario. Para ello fue necesario determinar el Umbral que debía utilizar para analizar las imágenes captadas por la cámara.

El Umbral es un valor que determina que parte de la imagen será Negra y que parte Blanca, contra el cual se compara cada uno de los píxeles que conforman la imagen. El píxel (del inglés *picture element*,es decir, “elemento de la imagen") es la menor unidad en la que se descompone una imagen digital, ya sea una fotografía, un fotograma de video o un gráfico. Las imágenes se conforman por una matriz rectangular de píxeles, donde cada píxel es un punto diminuto de la imagen.

Por otro lado, si el pixel analizado está por debajo del Umbral, este pixel tendrá un valor que corresponda al color blanco, pero si al contrario el valor del pixel se encuentra por encima del valor del Umbral su nuevo valor corresponderá al negro, como se observa en la figura 4.

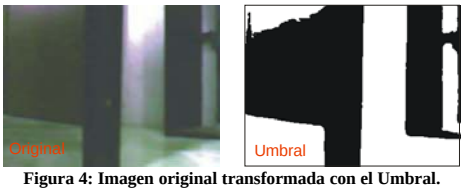

Figura 4: Imagen original transformada con el Umbral.

Hay que tener ciertas consideraciones para poder determinar el valor adecuado para fijar el Umbral.

Un factor determinante en el valor del umbral, es la luminosidad del entorno, si el entorno está muy iluminado, los obstáculos generan sombras que pueden erróneamente, interpretarse como obstáculos, si lo obstáculos son de texturas claras, se puede determinar que son espacios libres, en el caso contrario si el entorno es oscuro hay poca certeza al determinar la presencia de un obstáculo. Por esta razón para tener una mayor precisión se aconseja utilizar luz blanca; por la experiencia obtenida en las pruebas, es la más indicada para poder identificar los obstáculos en el entorno. Otro factor importante es la textura de la superficie que se considera libre de obstáculos,

algunos grabados en la superficie o cambios de textura pueden confundir al algoritmo.

Una vez terminados los procesos de filtrado, con la imagen resultante se puede determinar que obstáculos se encuentran presentes. Para esto solo se analizará el 40% de la imagen resultante (figura 5) esta área será identificada como área de barrido.

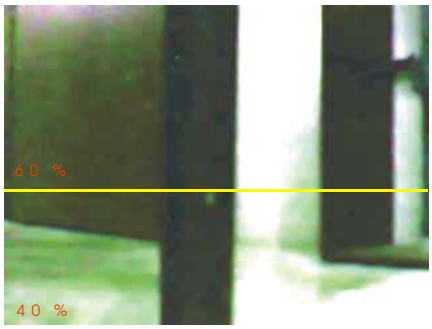

Figura 5: parte superior 60%, parte inferior 40%

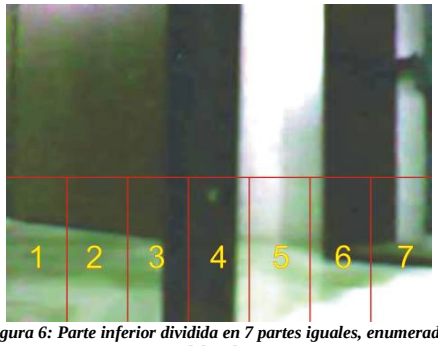

Figura 6: Parte inferior dividida en 7 partes iguales, enumeradas del 1 al 7.

En la figura 6 el área de barrido es segmentada en siete partes iguales (1 y 2 Lateral Izquierdo, 3, 4 y 5 Centrales, 6 y 7 Lateral Derecho), estas regiones se propusieron con la finalidad de detectar cual de las siete regiones representa la mejor ruta, ya que cada región corresponde a una rotación o desplazamiento.

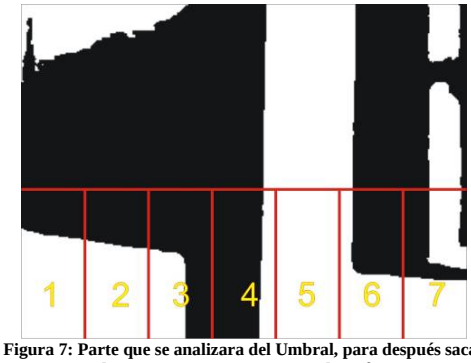

Figura 7: Parte que se analizara del Umbral, para después sacar las sumatorias y su posterior decisión.

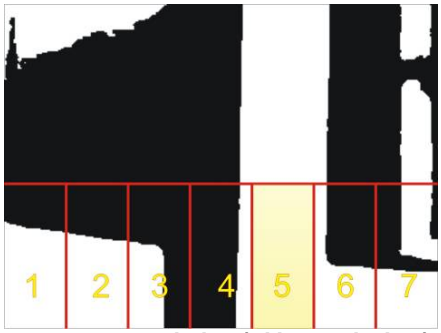

Fig. 8 Imagen seccionada, después del proceso de selección.

El proceso de selección, con el cual se determina la presencia de obstáculos, se hace mediante la sumatoria de píxeles de cada una de las secciones, después se hace una comparativa de las siete regiones donde aquellas que posean píxeles con el valor del color negro serán contabilizados y aquella región que posea la menor sumatoria será aquella que no esté obstaculizada.

En las pruebas realizadas se tiene una proporción de error del 10% al 15%, por tanto la efectividad del algoritmo es confiable. El porcentaje de error obtenido es consecuencia de la luminosidad del ambiente, ya que como se mencionó anteriormente, la intensidad de la luz afecta el Umbral y repercute en las decisiones del Algoritmo.

III. Tecnología Utilizada.

Toda la tecnología que se utilizo para la realización de este Algoritmo, es proporcionada por el Lenguaje de Programación JAVA, a traves de las siguientes herramientas:

- The JAVA 2 Standard Development Kit v.1.4.0
- JMF 2.1.1

Estos módulos se requieren para la manipulación de las imágenes y para el manejo y control del dispositivo de entrada

Pruebas

Se realizaron diversas pruebas desde el momento en que se desarrollaron los módulos de captura y de procesamiento. De manera inicial se realizaron las pruebas sin montar el dispositivo en el prototipo diseñado, con la intención de comprobar la efectividad y observar el comportamiento bajo la influencia de algunos factores tales como el color de los objetos, las condiciones de luz y las capacidades de procesamiento de los dispositivos.

Ya revisados los resultados arrojados por los procesos del algoritmo, la segunda fase de las pruebas consistió en montar el sistema de procesamiento sobre prototipo designado. Este prototipo tiene ciertas limitantes como sobre que tipo de terreno puede desplazarse.

Las pruebas fueron realizadas en una habitación con losas de piso. En la habitación se colocaron diferentes obstáculos de manera aleatoria como se muestra en la figura 9.

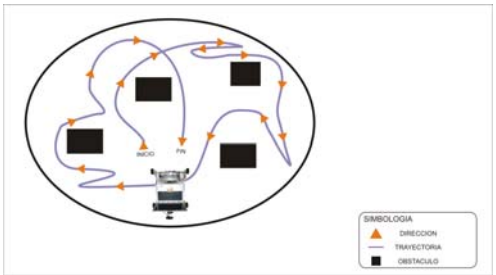

Figura 9: Trayectoria seguida por el prototipo.

IV. Referencias.

[1] Open Source Computer Vision Library: Reference Manual.

[2] Vision por Computador: Imágenes Digitales y aplicaciones. Alfa Omega, 2002.

[3] Nils J. Nilsson, (1984) Shakey the robot; Technical Note 323. Artificial Intelligence Center Computer Science and Technoloy Division, Obtenido el día 27 de Mayo de 2006 desde http://www.ai.sri.com/pubs/files/629.pdf

[4] A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone, A Versatile Computer-Controlled Assembly System, Proc. Third Int. Joint Conf. on AI, Stanford, California, pp. 298-307, 1973.

[5] Bertozzi M, Broggi A, Cellario M, Fascioli A., Lombardi P, Porta M.(Julio, 2002) Artificial Vision in Road Vehicles, Proceedings of the IEEE, vol. 99, No 7

[6] Iwan Ulrich, Illah Nourbakhsh, (2000) Appearance-Based Obstacle Detection with Monocular Color Vision, Proceedings of the AAAI National Conference on Artificial Intelligence, Austin, TX, July/August 2000

[7] Jiménez Copoya Luís E, (Febrero 2003) Algoritmo de Segmentación local dinámica aplicado al a caracterización de partículas, Obtenido el 13 de Junio de 2003 desde http://ccc.inaoep.mx/~labvision/tesis/Algoritmo%20de%20segmentacion%20global%20dinamica-%20Jimenez%20Copoya.pdf

[8] R.B. Ohlander, Analysis of natural scenes, PhD Thesis, Carnegie Institute of Technology, Dept. of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, 1975

[9] M. Cheriet, J. N. Said and C. Y. Suen, A recursive thresholding technique for image segmentation, IEEE Transactions on Image Processing, vol. 7, no. 6, pp. 918-920, 1998. Obtenido el 13 de Junio de 2006 desde http://www.livia.etsmtl.ca/publications/1998/Cheriet.OtsuR.pdf

[10] N. Otsu, A threshold selection method from grey level histograms, IEEE Transactions on Systems, Man and Cybernetics, vol. 8, pp. 62-66, 1978

[11] N. Ahuja, A. Rosenfeld and R.M. Haralick, (1980) Neighbour gray levels as features in pixel classification, Pattern Recognition,. 12:251-260, 1980.

[12] Wiesner J. Vos, (Octubre, 2004) Decision theory and discriminant analysis Obtenido el 14 de junio de 2006,pp 30-35, desde http://www.stats.ox.ac.uk/~evers/DataMining/ClassificationDAprint.pdf

CURRICULUM

Rogelio Ferreira Escutia, nació en Morelia, Michoacán en 1971. Obtuvo el grado de Ingeniero en Electrónica en el Instituto Tecnológico de Morelia en 1995 y el grado de Maestro en Ciencias de la Computación en Instituto Tecnológico y de Estudios Superiores de Monterrey Campus Morelos en 1998. Actualmente es Profesor del Departamento de Sistemas y Computación y de la Maestría en Ciencias en Ciencias de la Computación en el Instituto Tecnológico de Morelia.

Aldo González Castro, nació en México D.F. en 1980. Es candidato al grado de Ingeniero en Sistemas Computacionales en el Instituto Tecnológico de Morelia en octubre del 2006.

César Ibarra Mejía, nació en México D.F. en 1982. Es candidato al grado de Ingeniero en Sistemas Computacionales en el Instituto Tecnológico de Morelia en octubre del 2006.

Victor Manuel Eslava Echagaray, nació en México D.F. en 1980. Es candidato al grado de Ingeniero en Sistemas Computacionales en el Instituto Tecnológico de Morelia en octubre del 2006.