Imágenes Biomédicas
Tarea I

Joel Chacón Castillo

Guanajuato, México

---

1. Punto I

Analizar y reproducir por partes el método propuesto por Uguz et al. [1].

1.1. Desarrollo

En este trabajo se propone un método especial para la detección de bordes. Específicamente, se emplean los siguiente tres conceptos clave. Se utiliza un autómata celular lineal (CA) cuyas reglas son definidas de forma especial para tratar el problema en cuestión. Además, se define una función lógica difusa *fuzzy logic* la cual permite capturar la incertidumbre de los datos. Finalmente, los parámetros "óptimos" que pertenecen a función de lógica difusa son obtenidos por medio de un algoritmo estocástico el cual es conocido como PSO *Particle Swarm Optimization*. Cada uno de estos componentes es analizado en detalle en las siguientes secciones.

1.1.1. Estructura de un autómata celular

Un autómata celular es un sistema discreto dinámico que consiste en una red de celulas finitas las cuales cambian sus estados en función al estado de los vecinos, esto se realizan de acuerdo a una regla de actualización. Todas las células cambian su estado de forma simultánea, por medio de la misma regla de actualización. Este proceso es repetido en pasos de tiempo discreto. Así, por medio de reglas de actualización simples se podrían producir sistemas dinámicos extremadamente complejos [2].

1.1.2. Lógica Difusa

Es una forma lógica en que las variables que pertenecen a las variables "Verdaderas" pueden ser cualquier número real entre $[0, 1]$. Es decir se utiliza

para manejar el concepto de la verdad parcial, donde un valor verdadero puede cambiar entre completamente verdadero y completamente falso [3]. Esto se realiza por medio de una función que define el grado de pertenencia a los **conjuntos difusos**, de esta forma la lógica difusa permite capturar incertidumbre en los datos [4].

Es este trabajo considera un esquema similar a [5], en este último trabajo se emplean las imágenes como conjuntos difusos donde se denota a X como una imágen y $g_{mn} \in [0, L]$ como la escala de grises de cada pixel en la posición (m, n). Además, X puede ser considerado como un array con entradas que pertenecen al conjunto difuso ${}_{mn} \in [0, 1]$, el cual indica el grado de intensidad en cada pixel. Este array es defindo a continuación:

$$X = \cup_{m=1}^{M} \cup_{n=1}^{N} \frac{\mu_{mn}}{g_{mn}} \tag{1}$$

Además, un ejemplo de la función de pertenencia μ puede ser determinada normalizando la matriz X:

$$\mu_{mn} = \frac{g_{mn}}{max_{i \in [1,M], j \in [1,N]} g_i j} \tag{2}$$

Finalmente, la imagen X' que contiene todos los bordes es:

$$X' = \cup_{m=1}^{M} \cup_{n=1}^{N} \frac{\hat{\mu}_{mn}}{g_{mn}} \tag{3}$$

donde $\hat{\mu}_{mn}$ indica el grado en que un pixel puede ser un borde. De esta forma definiendo una función de pertenencia adecuada el proceso de detección de bordes puede ser fácilmente realizado. Sin embargo, este prodecimiento puede ser considerado como una heurística ya que no existe una función de pertenencia única. La función de perteneceia considerada en este trabajo es como se indica a continuación.

$$\mu_{mn} = \frac{\sum_i \sum_j |g_{ij} - g_{mn}|}{\Delta + \sum_i \sum_j |g_{ij} - g_{mn}|} \tag{4}$$

donde $\Delta \in [0, L)$ es un parámetro propio. Además, se detectarán una menor cantidad de bordes con un meno parámetro Δ.

1.1.3. Reglas de transición celular difusas por medio de un umbral

Dado el pixel central $C_{i,j}$ un vecindario es considerado como los pixeles que pertenecen a la misma ventana $k, l \in \{-1, 0, 1\}$. Además, un vecindario

popular donde consideran a todos los vecinos es conocido como el vecindario de "Moore". De esta forma, se puede definir un grado de pertenencia el cual mide la contribución a los bordes por medio de una función de pertenencia adecuada. Esta función de pertenencia debe asignar un valor no nulo al pixel central en la ventana considerada y eventualmente todos los pixeles que pertenecen a la regla previamente definida serán utilizados en base al parámetro Δ como se indica a continuación.

$$\mu(C_{i,j}) = \frac{\Phi(C_{i,j})}{\Delta + \Phi(C_{i,j})} \tag{5}$$

donde $\Phi(C_{i,j}) = \sum_k \sum_l |C_{i,j} - C_{i+k,j+l}|$

De esta forma se puede calcular la contribución de los pixeles vecinos al grado de "edginess" del pixel central. Finalmente, al juntar las operaciones previemente definidas y en base a una función de umbral parcial se pueden definir los pixeles que pertenecen al borde y a los que no de la siguiente forma.

$$F(C_{i,j}) = \begin{cases} 1, & \mu(C_{i,j}) > \tau \\ 0, & \mu(C_{i,j}) \leq \tau \end{cases} \tag{6}$$

1.1.4. Particle Swarm Optimization

En base a lo mencionado previamente, se utiliza PSO para optimizar los parámetros Δ y τ, donde Δ es indica el grado de "edginess" y τ es el parámetro o umbral de pertenencia. La función objetivo está compuesta por la función de medida conocida como "Baddleys Delta Metric" (BDM), la cual es una medición inicialmente diseñada para la comparación de conjuntos binarios, sin embargo en este caso se utiliza para comparar imágenes binarias dado que los mapas de los bordes consisten de las mismas estructuras. El BDM trata de medir la disimilaridad del los subconjuntos de puntos, el cual usualmente es representado por 1. Dadas dos imágenes binarias (B_1 y B_2) con las mismas dimensiones $M \times N$, y dado el conjunto de posiciones $\Omega = \{1, M\} \times \{1, N\}$. Dado un valor $1 < k < \infty$ el k-indice entre las imágenes B_1 y B_2 (denotado por $(\prod)^k(B_1, B_2)$) es definido de la siguiente forma.

$$(\prod)^k(B_1, B_2) = \left[\frac{1}{|\Omega|} \sum_{t \in \Omega} |w(d(t, B_1)) - w(d(t, B_2))| \right]^{\frac{1}{k}} \tag{7}$$

donde $d(t, B_i)$ representa la distancia de la posición t al punto característico mas cercano a la imagen B_i y $w : [0, \infty] \to [0, \infty]$, es una función concava

incremental. En este trabajo se utiliza la distancia Euclideana para el cálculo de d. Por lo tanto $d(t, B_i)$ es la distancia Euclideana mínima de la posición t a algún punto de B_i, para la función w, se utiliza $w(x) = x$ y un valor $k = 2$.

1.2. Discusión

Inicialmente, los autómatas celulares que describen en el artículo tienen un enfoque muy distinito al que se suele aplicar a los autómatas celulares usuales, principalmente porque no se define una forma para actualizar las reglas, es decir ellos indican reglas estáticas. La parte en la que se une el principio de lógica difusa con PSO me parece bastante interesante. Sin embargo, parece ser un problema escoger una función objetivo para evaluar la calidad de las soluciones, esta no es la excepción y al parecer se utiliza la métrica BDM con el *ground truth*. Además, al parecer revisaron todas las combinaciones de reglas (512) para ver cual es la que proporciona mejores resultados. El algorithmo evolutivo que utilizan también debería ser probado con distintos parámetros, además sugeriría que antes de probar un método de optimización estocástico se tratara de probar un método de Newton, esto en base a que se dispone de las fórmulas de la función objetivo, y al parecer la función objetivo es convexa (BDM - prueba con la desigualdad de Jensen).

1.3. Conclusión

En primer instante me pareció un trabajo muy extenso y no reproducible fácilmente, y al parecer con resultados no significativamente mejores que los métodos clásicos. Si cambiaría algo de este trabajo sería quitar el enfoque de las reglas que le pusieron y en su lugar utilizar una combinación lineal de operadores, en donde se desean optimizar los pesos, así se podría incorporar a la función de lógica difusa sin problemas. En realidad una idea interesante sería realizar un suma ponderada de operadores, donde sea asignado el peso que mejor se adecúa al problema, ya sea Sobel, Laplaciano, entre otros operadores.

2. Punto II

De un método de detección de bordes, analizar y concluir sobre la definición de su operador de gradiente de acuerdo al autor.

2.1. Desarrollo

Una taxonomía bastante acertada[1] en la propuesta en [6]. Principalmente, se pueden encontrar los métodos de *Sobel*, *Prewitt*, *Laplacian of Gaussian* y *Canny Edge Detector* [7]. Es importante mencionar que no existe el "mejor método", es decir, en algunos escenarios algunos métodos son más ideales que otros. Por lo regular, en los métodos de detección de bordes existe un *trade-off*. Así, mientras que algunos métodos pueden ser más rápidos, otros métodos podrían ser más robustos. Por lo tanto, lo ideal es aplicar el método más "adecuado"[2] dado un conjunto de imágenes.

Particularmente, en este trabajo se considera el método de *Sobel*. Una de las razones de esto, es que a pesar de su simplicidad y que es muy antiguo, en la actualidad es considerado como uno de los mejores métodos, y por lo tanto se suele utilizar como estado del arte. Aunque el método de Sobel fue inicialmente propuesto en [8] por Irwin Sobel, se sugiere revisar [9], este último está ampliamente descrito. El operador de Sobel realiza el cálculo del gradiente espacial 2D en una imagen y enfatiza las regiones con mayor gradiente espacial que corresponde a los bordes. Por lo regular, es utilizado para encontrar la magnitud del gradiente en cada pixel de una imagen en escala de grises. La principal ventaja del operador de Sobel es que tiene un efecto de suavizar el ruido ya que se basa en un promedio.

El método de *Sobel* calcula el gradiente por medio de diferencias finitas entre filas y columans en un vecindario 3×3. El operador de Sobel está basado en convolucionar la imágen con un filtro pequeño, separable y cuyos valores son enteros.

Este operador fue sugerido en una plática por Irwin Sobel en el 1968 [8], posteriormente este trabajo tuvo divulgación por Pingle (1969) y además se agregó en el libro realizado por Duda y Hart [10]. La motivación de este operador es desarrollar una estimación del gradiente de forma eficiente, el cual es más isotrópico que el operador de "Robert Cross"[11].

El principio de este operador es la estimación del gradiente de una imagen en un punto que involucre la suma vectorial de los cuatro posibles estimaciones simples y centrales del gradiente que se puede obtener en un vecindario de 3×3. La operación de sumar los vectores proporciona un promedio sobre

[1] Se revisaron varios trabajos que se anexan a este trabajo

[2] En este contexto adecuado se refiere a que proporcione soluciones de calidad de forma rápida

a	b	c
d	e	f
g	h	i

Cuadro 1: Punto en el grid cartesiano

las mediciones de direcciones del gradiente. Los cuatro gradientes tendrían el mismo valor si la función de densidad fuera realmente plana sobre su vecindario. Así, cualquier diferencia son desviaciones de la planaridad local de la función sobre el vecindario. La intención es extraer la dirección del mejor plano, sin embargo no se había intentado realizar esto de forma más robusta. Para un vecindario 3×3 cada estimación simple del gradiente central es una suma de pares de vectores ortogonales. Cada vector ortogonal es una estimación de la derivada direccional multiplicada por un vector unitario especificando la dirección derivativa. La suma vectorial de estas cuatro estimaciones de vectores simples equivalen a la suma vectorial de los ocho vectores de derivada direccional. Específicamente, la estimación del vector direccional derivativo G es definido como (diferencia de densidad)/(distancia al vecino). Este vector es determinado tal que la dirección de G será dada por el vector unitario para aproximar al vecino. Es importante notar que los vecinos son agrupados en pares antipodales[3]. Así, si se considera el punto central y sus ocho vecinos como se indica en la tabla 1, los pares antipodales son : (a,i), (b,h), (c,g), (f,d). La suma vectorial para esta estimación del gradiente es como se indica a continuación (se nota que el **e** no está en la ecuación pues es eliminado en la resta).

$$G = \frac{(c-g)}{4} * [1,1] + \frac{(a-i)}{4} * [-1,1] + \frac{(b-h)}{2} * [0,1] + \frac{(f-d)}{2} * [1,0] \quad (8)$$

por lo tanto este vector resultante es como se indica a continuación.

$$G = \frac{(c-g-a-i)}{4} + \frac{(f-d)}{2}, \frac{(c-g+a-i)}{4} + \frac{(b-h)}{2} \quad (9)$$

Es importante notar que no se considera la raíz cuadrada de esta representación, además se debería dividir por cuatro para obtener el promedio. Sin

[3]antipodal:diametrically opposed to

Cuadro 2: Componente x

-1	0	1
-2	0	2
-1	0	1

Cuadro 3: Componente y

1	2	1
0	0	0
-1	-2	-1

embargo, dado que se está trabajando con enteros y por cuestiones numéricas se opta por multiplicar por cuatro dejando la siguiente fórmula (el error numérico está relacionado con el corrimiento de bits).

$$G' = 4G = [c - g - a - i + 2(f - d), c - g + a - i + 2(b - h)] \tag{10}$$

Finalmente, se expresa la ecuación 10 como una sumatoria de densidades ponderadas utilizando las funciones de ponderación que se utilizan en las tablas 2, 3.

2.2. Propuesta

Para realizar una propuesta sólida, principalmente se realizaron los siguientes trabajos.

- El-Sayed and Hafeez [12]: La propuesta consiste en realizar el cálculo del threshold de forma iterativa (utiliza la media entre las dos clases, creo que este método está relacionado al visto en clase). El procedimiento consiste primero en convertir en binario la imagen dado un threshold arbitrario, posteriormente se reliza un barrido de la imagen por medio de una ventana, en cada pixel central se mide la incertidumbre (se utiliza la entropía) de que un pixel pertenezca al borde. Además dividen la imagen en cuatro regiones. Una desventaja es que ellos primero hicieron un análisis del un rango para el threshold. La parte interesante es que introducen el criterio para medir la incertidumbre

- Bhandarkar et al. [13]: En este trabajo se plantea el problema de *Edge Detection* como uno de optimización, este último es resuelto por medio de un genético "personalizado". Específicamente, se definen vecindarios

válidos (ventanas) de dimensión 3×3, la función objetivo a optimizar está comprendida por el sumatorio poderado de distintos tipos de costos (dissimilarily, edge thickness, edge curvature, edge fragmentation, and number of edge pixels detected), estos costos son calculados en base a un árbol de decision. Principalmente, para calcular el costo *dissimilarity* se utiliza la imágen en escala de grises la cual es procesada (*enhancement*) y posteriormente comparada con la de cada individuo. En la validación experimental se utiliza la métrica de *The Pratt Figure of Merit* (PFM).

- Jin-Yu et al. [14]: En este trabajo se realizan dos propuestas, en la primera se presenta una versión mejorada del método de Sobel en la que utilizan cuatro direcciones en plantillas de 5×5 en lugar de 3×3, esto en base a que el método de Sobel clásico considera únicamente dos direcciones (horizontal y vertical) y por lo tanto se presentan problemas en la detección de texturas complicadas, además para obtener el threshold se utiliza el método de Otsu optimizado por un algoritmo evolutivo.

La propuesta es una extensión del método de Sobel, la idea es consiste primero en identificar distintas clases de la imagen (en escala de grises), esto se puede realizar por medio del algoritmo $k - means$, posteriormente por cada clase determinar el threshold ideal, hasta esta parte el enfoque es similar al utilizado en [15]. Por otra parte para encontrar el threshold de cada grupo se sugiere aplicar un enfoque similar al de [14], el cual se enfoca en encontrar el threshold de cada clase que maximice la varianza, la función objetivo sería la sumatoria de las varianzas de las clases. Adicionalmente, se le puede pre-procesar la imagen con *wavelets-denoising* [16].

Plantillas utilizadas en la forma mejorada de Sobel:

$$T_x = \begin{bmatrix} 2 & 3 & 0 & -3 & -2 \\ 3 & 4 & 0 & -4 & -3 \\ 6 & 6 & 0 & -6 & -6 \\ 3 & 4 & 0 & -4 & -3 \\ 2 & 3 & 0 & -3 & -2 \end{bmatrix} \; T_y = \begin{bmatrix} 2 & 3 & 6 & 3 & 2 \\ 3 & 4 & 6 & 4 & 3 \\ 0 & 0 & 0 & 0 & 0 \\ -3 & -4 & -6 & -4 & -3 \\ -2 & -3 & -6 & -3 & -2 \end{bmatrix} \tag{11}$$

$$T_{45} = \begin{bmatrix} 0 & -2 & -3 & -2 & -6 \\ 2 & 0 & -4 & -6 & -2 \\ 3 & 4 & 0 & -4 & -3 \\ 2 & 6 & 4 & 0 & -2 \\ 6 & 2 & 3 & 2 & 0 \end{bmatrix} \; T_{135} = \begin{bmatrix} -6 & -2 & -3 & -2 & 0 \\ -2 & -6 & -4 & 0 & 2 \\ -3 & -4 & 0 & 4 & 2 \\ -2 & 0 & 4 & 6 & 2 \\ 0 & 2 & 3 & 2 & 6 \end{bmatrix} \tag{12}$$

2.3. Análisis experimental

En esta sección se realizan dos análisis en el primero se prueba el método de Sobel proporcionado por matlab, el método de Sobel programado y el método de Sobel mejorado. En el segundo análisis se verifica el cambio la varianza del método de Sobel con distintos thresholds y con distintas imágenes.

En las imagenes 1, 2 y 3 se encuentran los resultados de aplicar distintas variantes del método de Sobel. Se puede observar que al considerar más direcciones (método de Sobel mejorado) los bordes con distintas formas se capturan más adecuadamante.

Por otra parte en la figura 4 se ilustra la varianza de las dos clases (borde y fondo), principalmente se puede observar que la función es "suave" y en estos tres casos tienen un punto de máxima varianza, que posiblemente esté relacionado con el threshold más adecuado, en esto se basa el método de Otsu.

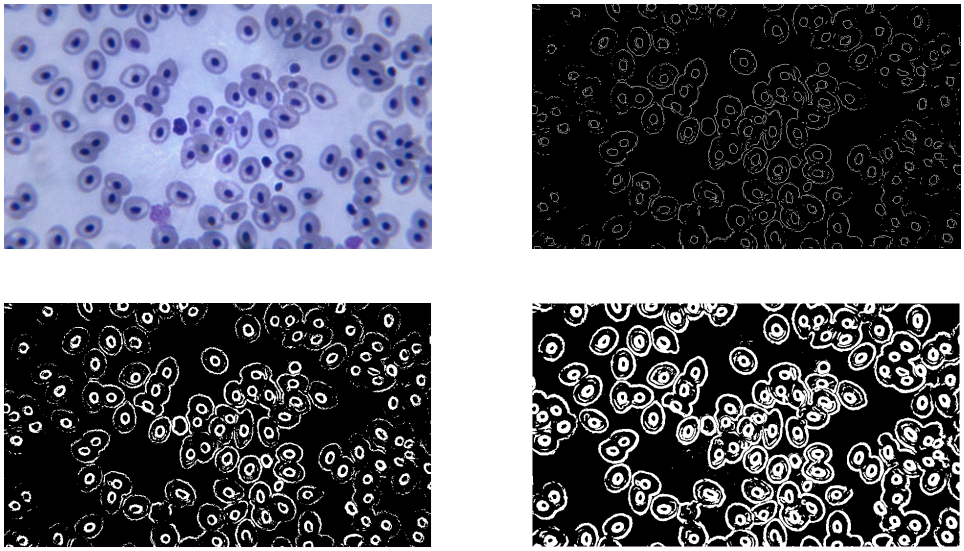

Figura 1: En la parte superior izquierda está la imagen original, en la superior derecha corresponde al método de Sobel de la librería libreria de MATLAB, en la parte inferior izquierda corresponde al método de Sobel como se indica en el artículo clásico y en la parte inferior derecha corresponde al método de Sobel mejorado.

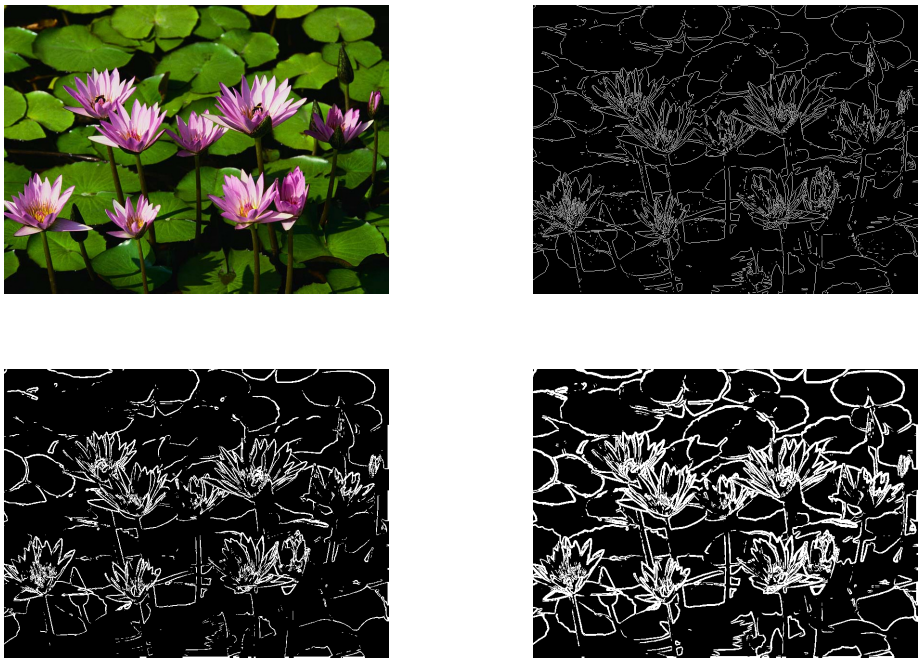

Figura 2: En la parte superior izquierda está la imagen original, en la superior derecha corresponde al método de Sobel de la librería libreria de MATLAB, en la parte inferior izquierda corresponde al método de Sobel como se indica en el artículo clásico y en la parte inferior derecha corresponde al método de Sobel mejorado.

Figura 3: En la parte superior izquierda está la imagen original, en la superior derecha corresponde al método de Sobel de la librería libreria de MATLAB, en la parte inferior izquierda corresponde al método de Sobel como se indica en el artículo clásico y en la parte inferior derecha corresponde al método de Sobel mejorado.

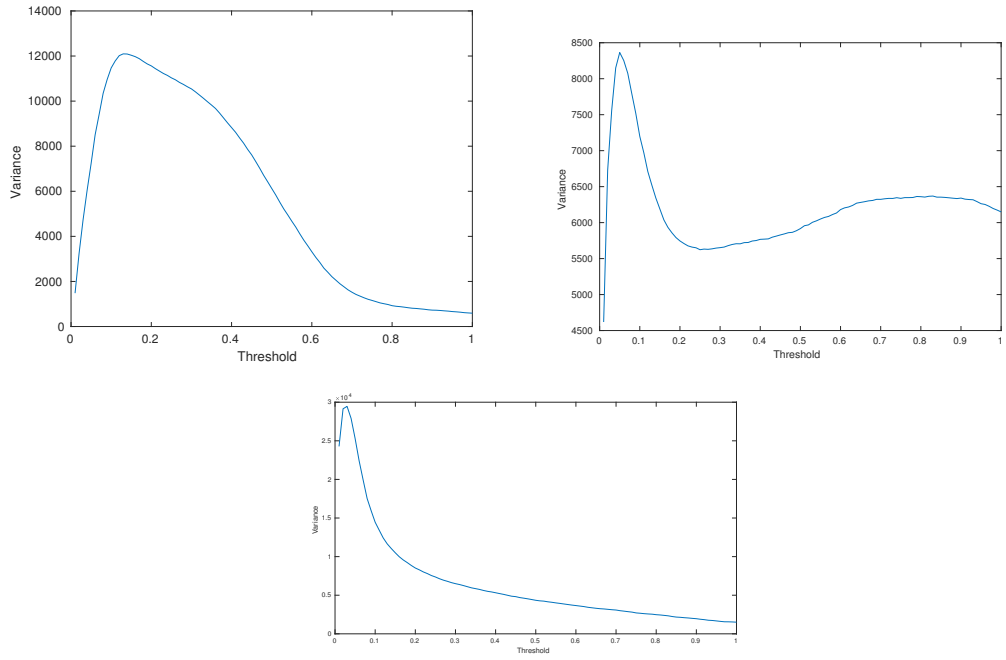

Figura 4: Varianza de con distintos thresholds que corresponden a la implementación de la librería de MATLAB. La parte superior izquierda corresponde a la varianza de las celular, en la parte superior derecha corresponde a las flores y en la parte inferior corresponde a la foto de una persona.

2.4. Conclusión

Se revisaron varios trabajos relacionados con detección de bordes, entre ellos uno de los métodos más populares es el de Sobel. El método de Sobel consiste en el cálculo del promedio de los gradientes en cada pixel central. Visualmente la forma de Sobel aumentada proporciona buenos resultados, por lo tanto sería interesante analizar ventanas más grandes $(2i + 1)$ como son de 7×7, 9×9. Además sería interesante combinar ventanas de distintos tamaños basados en el operador de Sobel. También sería útil implementar clasificación en cascada para identificar los bordes, es decir, remover por capas las regiones que no estan relacionadas con los bordes. Además, sería de utilidad contar con un método que no utilice threshold y que se adaptativo (esto ya suena a un método de Canny).

1 [1] S. Uguz, U. Sahin, F. Sahin, Edge detection with fuzzy cellular au-

tomata transition function optimized by pso, Computers & Electrical Engineering 43 (2015) 180–192.

[2] J. Kari, Theory of cellular automata: A survey, Theoretical computer science 334 (2005) 3–33.

[3] V. Novák, I. Perfilieva, J. Mockor, Mathematical principles of fuzzy logic, volume 517, Springer Science & Business Media, 2012.

[4] C.-C. Kang, W.-J. Wang, A novel edge detection method based on the maximizing objective function, Pattern Recognition 40 (2007) 609–618.

[5] H. R. Tizhoosh, Fast fuzzy edge detection, in: Fuzzy Information Processing Society, 2002. Proceedings. NAFIPS. 2002 Annual Meeting of the North American, IEEE, pp. 239–242.

[6] M. Heath, S. Sarkar, T. Sanocki, K. Bowyer, Comparison of edge detectors: a methodology and initial study, Computer vision and image understanding 69 (1998) 38–54.

[7] P. P. Acharjya, R. Das, D. Ghoshal, Study and comparison of different edge detectors for image segmentation, Global Journal of Computer Science and Technology (2012).

[8] I. Sobel, History and definition of the so-called"sobel operator", more appropriately named the sobel-feldman operator, Sobel, I., Feldman, G.,.A 3x3 Isotropic Gradient Operator for Image Processing", presented at the Stanford Artificial Intelligence Project (SAIL) in 2015 (1968).

[9] O. R. Vincent, O. Folorunso, A descriptive algorithm for sobel image edge detection, in: Proceedings of Informing Science & IT Education Conference (InSITE), volume 40, Informing Science Institute California, pp. 97–107.

[10] R. O. Duda, P. E. Hart, Pattern classification and scene analysis, A Wiley-Interscience Publication, New York: Wiley, 1973 (1973).

[11] L. Roberts, Machine perception of three-dimensional solids, in optical and electro-optical information processing (j. tippett, ed.), 159-197, 1965.

[12] M. A. El-Sayed, T. A.-E. Hafeez, New edge detection technique based on the shannon entropy in gray level images, arXiv preprint arXiv:1211.2502 (2012).

[13] S. M. Bhandarkar, Y. Zhang, W. D. Potter, An edge detection technique using genetic algorithm-based optimization, Pattern Recognition 27 (1994) 1159–1180.

[14] Z. Jin-Yu, C. Yan, H. Xian-Xiang, Edge detection of images based on improved sobel operator and genetic algorithms, in: Image Analysis and Signal Processing, 2009. IASP 2009. International Conference on, IEEE, pp. 31–35.

[15] N. Mathur, S. Mathur, D. Mathur, A novel approach to improve sobel edge detector, Procedia Computer Science 93 (2016) 431–438.

[16] W. Gao, X. Zhang, L. Yang, H. Liu, An improved sobel edge detection, in: Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, volume 5, IEEE, pp. 67–71.