

Informe

Detección de Estantes en Imágenes de Góndolas de Supermercados

Profesor: Claudio Pérez
Auxiliar: Alonso Astroza
Estudiante: Diego Campanini
Fecha: 26 de diciembre de 2015

Índice

1. Introducción	3
2. Metodología	4
2.1. Detección de Estantes	4
2.2. Medida de Desempeño	4
3. Resultados y Análisis	6
3.1. Imagen RGB y en escala de Grises	6
3.2. Imágenes filtradas y candidatos a líneas	6
3.2.1. Filtrado y Aplicación de Transformada de Hough	6
3.2.2. Filtrado e imagen para detección final de líneas	8
3.3. Máscara resultante y medición del rendimiento	10
4. Conclusión	12
5. Anexos	13

Índice de figuras

1.	Imagen <i>db45.jpg</i> para la cual se mostraran los resultados	6
2.	Resultado de aplicar filtro Sobel Horizontal y dilatar las líneas	7
3.	Candidatos a Bordes del Estante	8
4.	Imagen resultante al aplicar canny sobre la imagen en escala de grises	9
5.	Resultado de aplicar filtro Sobel vertical y dilatar las líneas	9
6.	Imagen resultante de aplicar Canny menos las líneas verticales dilatadas	10
7.	Máscara groundtruth y máscarta detectada	11

Índice de cuadros

1. Introducción

El objetivo del proyecto es implementar un detector de estantes en imágenes de góndolas de supermercados. El método utilizado fue aplicar dos etapas en paralelo, ambas considerando distintos filtros (Sobel y Canny) y además de la Transformada de Hough para detección de candidatos a líneas de los estantes.

Para probar los algoritmos desarrollados se utilizó el conjunto número 1 de la base de datos WebMarket, las imágenes que contiene esta base de datos son de góndolas de supermercados vistas de frente, las imágenes son RGB de dimensiones de 2272 por 1704.

2. Metodología

2.1. Detección de Estantes

Para la detección de estantes en las imágenes de góndolas de supermercados se utilizaron distintos filtros (horizontales, verticales y combinados) y la transformada de Hough, para detección de las líneas horizontales de los estantes.

El proceso para la detección de los estantes se realizó de forma paralela en dos etapas, por un lado se generó una imagen para aplicar la transformada de Hough y por otro lado se generó una imagen solamente para realizar la detección final de las líneas, basándose en las candidatas a líneas. A continuación se describen las dos etapas.

- 1. Filtrado y aplicación de transformada de Hough:** Lo primero que se efectuó en esta etapa fue pasar leer la imagen RGB y pasarl a escala de grises, luego sobre esta imagen se aplicó un filtro Sobel horizontal, para eliminar de la imagen las líneas verticales que no serán analizadas con esta método, ya que, las líneas más largas son las horizontales.

Una vez obtenida la imagen con solo líneas horizontales, se procedió a agrandar estas líneas, de este modo se logró una mejor detección de líneas horizontales con Hough. Al aplicar Hough a la imagen filtrada de forma horizontal y con sus líneas ensanchadas, se formó un conjunto candidato a línea horizontal del estante, el cual se usó en conjunto con la imagen resultante de la etapa 2.

- 2. Filtrado e imagen para detección final de líneas:** El otro procesamiento de la imagen que se realizó de forma paralela consistió en aplicar filtro Canny sobre la imagen en escala de grises para detectar líneas verticales horizontales en esta, además aplicó un filtro Sobel vertical a la imagen en escala de grises almacenando el resultado en una nueva imagen, posteriormente a esta nueva imagen se le dilataron las líneas verticales.

Según lo explicado en el párrafo anterior se tienen dos imágenes, una la que resulta de aplicar Canny a la imagen original en escala de grises y la segunda es la resultante de dilatar las líneas verticales encontradas con el filtro Sobel, en seguida, se usa las líneas verticales para eliminarlas en la imagen resultante de aplicar Canny, obteniendo la imagen que finalmente se utilizó para la detección de líneas de los estantes, en base a las líneas candidatas obtenidas con Hough en la etapa paralela a esta (punto 1).

Una vez efectuadas las etapas explicadas en los puntos 1 y 2, los candidatos a líneas horizontales obtenidos con la Transformada de Hough y la imagen resultante de la etapa 2, ingresan a una función de nominada *drawEdge*, la cual se encarga de detectar las líneas horizontales que forma un estante, según un umbral *u*, el cual indica que si las líneas candidatas están separadas una distancia inferior a un umbral *u* y mayor a cero, para evitar líneas superpuestas, entonces estas pertenecen a los bordes horizontales de un estante y se procede a marcar el estante como el área comprendida entre las coordenadas de las dos líneas comparadas.

2.2. Medida de Desempeño

Para medir el desempeño de los algoritmos desarrollados para la tarea de detección de los estantes, se programó una función denominada *rendimiento* la cual recibe como parámetros dos máscaras binarias, una

correspondiente al *groundtruth* y la otra a los estantes detectados por el sistema de detección programado, con estas dos máscaras la función calcula el rendimiento como se indica en la ecuación (1)

$$M = \frac{B_1 \cap B_2}{B_1 \cup B_2} \quad (1)$$

El *groundtruth* corresponde a una medida de correspondencia a los estantes marcados de forma manual, en este proyecto para esta tarea se utilizó principalmente el programa Paint. Mencionar que la construcción de la citada máscara manual no es perfecta y existen errores asociados a esta debido principalmente a que no todas las líneas de los estantes eran rectas perfectamente horizontales o verticales, algunas eran diagonales, lo cual conllevaba a un error al momento de trazar las líneas con las herramientas que posee Paint.

Con respecto a la ecuación 1, el numerador corresponde a identificar los casos de correcta detección, lo cual ocurre cuando en ambas máscaras hay un cero (pixel negro), por su parte, el denominador corresponde a la suma de los casos de correcta detección y los de incorrecta detección, estos últimos se identifican cuando los píxeles de las máscaras no coinciden entre sí, es decir, cuando en una hay un 1 en la otra hay un 0 y viceversa.

3. Resultados y Análisis

Se proceden a exponer las imágenes obtenidas en las distintas etapas de filtrado de estas, también se muestra en esta sección las líneas candidatas a pertenecer a los estantes y el resultado final obtenido como máscara utilizando la metodología antes expuesta.

3.1. Imagen RGB y en escala de Grises

La imagen para la cual se muestran los resultados es la *db45.jpg*, del primer conjunto de la base de datos de WebMarket. Las imagen original en RBG y en escala de grises se aprecia en la Figura 1. El procesamiento de las imágenes en este trabajo se realizó sobre las imágenes en escala de grises.



Figura 1: Imagen *db45.jpg* para la cual se mostraran los resultados

3.2. Imágenes filtradas y candidatos a líneas

Como se expuso en la sección 2 se realizaron sucesivos filtrados sobre la imagen en escala de grises, los cuales se separaron en dos etapas, los resultados de estos filtrados se muestran a continuación.

3.2.1. Filtrado y Aplicación de Transformada de Hough

Se aplicó filtro Sobel horizontal sobre la imagen en escala de grises y luego se dilataron estas líneas, el resultado se puede apreciar en la figura 2. Claramente se aprecia el efecto de la dilatación de las líneas horizontales en la Figura 2 (b), esto favorece la detección de líneas candidatas a pertenecer a los estantes.

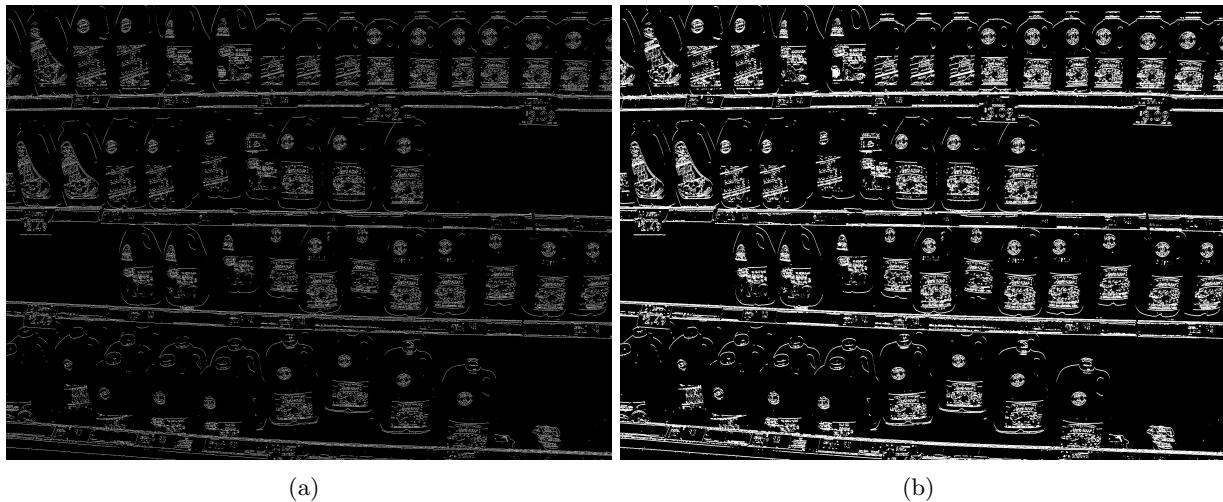


Figura 2: Resultado de aplicar filtro Sobel Horizontal y dilatar las líneas

Lo que sigue en el sistema de detección es aplicar Transformada de Hough sobre la imagen con líneas horizontales dilatadas, obteniéndose lo que se aprecia en la Figura 3, en este caso se tiene una acertada detección de líneas candidatas a pertenecer a los bordes de los estantes, el número de líneas a detectar variará dependiendo de los parámetros que se escogen para la función que implementa Hough en Matlab.

La Transformada de Hough en Matlab se utilizó como se muestra en el siguiente cuadro. Los parámetros que se muestran se mantuvieron invariantes para todas las pruebas y se escogieron los valores que se aprecian mediante sucesivas pruebas empíricas, hasta obtener los mejores resultados posibles de forma global.

```

1 rotI=im_horDilate ;
2 [H,Theta ,Rho] = hough( rotI );
3 P = houghpeaks(H,20 , 'threshold' , ceil (0.2*max(H(: ))));
4 x = Theta(P(:,2));
5 y = Rho(P(:,1));
6 lines = houghlines(rotI ,Theta ,Rho,P , 'FillGap' ,20 , 'MinLength' ,200);
7 figure , imshow(rotI) , hold on

```

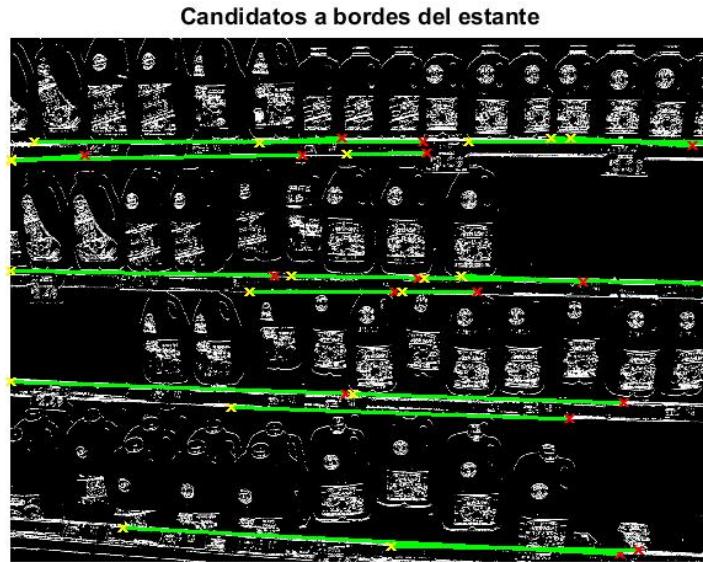


Figura 3: Candidatos a Bordes del Estante

3.2.2. Filtrado e imagen para detección final de líneas

El proceso que se realizó paralelo al anteriormente descrito comenzó por aplicar Canny a las imágenes en escala de grises, el resultado de esto en la imagen *db45* se aprecia en la Figura 4, se aprecia que con este método se obtiene una detección mucho más limpia de los bordes de los objetos y de los estantes.

Para obtener una mejor detección de las líneas predominantes en los estantes, que son las horizontales, se plantea la necesidad de eliminar tanto como se posible, las líneas verticales en la Figura 4, para esto se aplica un filtro Sobel Vertical a la imagen en escala de grises, posteriormente dilatar estas líneas, los dos efectos antes mencionados se muestran en la Figura 5



Figura 4: Imagen resultante al aplicar canny sobre la imagen en escala de grises

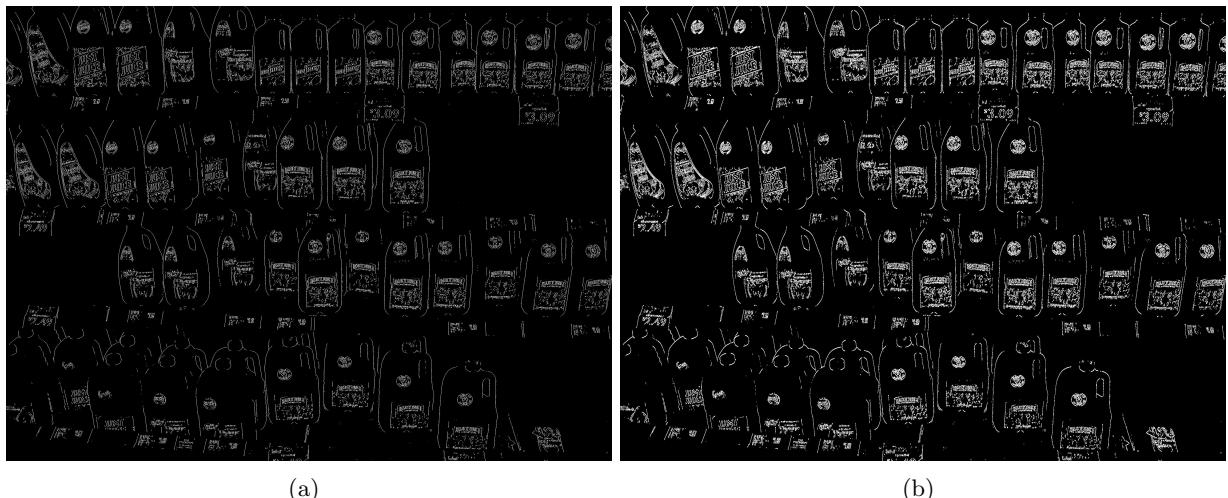


Figura 5: Resultado de aplicar filtro Sobel vertical y dilatar las líneas

Finalmente se restan las líneas verticales obtenidas con Sobel y dilatación, a la imagen resultante de aplicar Canny a la imagen gris, lo que resulta en la Figura 6, esta imagen fue la que se utilizó finalmente para formar la imagen binaria con los estantes detectados.



Figura 6: Imagen resultante de aplicar Canny menos las líneas verticales dilatadas

3.3. Máscara resultante y medición del rendimiento

Al utilizar las líneas candidatas a pertenecer a los bordes de un estante obtenidas con la transformada de Hough (Figura 3) y la imagen resultante de la etapa dos (Figura 6), se obtiene como resultado desde la función *drawEdge* la figura 7 (b), el rendimiento para esta detección fue de $M = 12.88\%$, para un umbral $u = 70$, que corresponde al máximo ancho que puede tener una línea.

El resultado obtenido se considera satisfactorio, ya que, se logra detectar el número correcto de estantes, 4 en este caso, además las dimensiones de estos son similares a las de ground truth (Figura 7), algunos problemas que tiene la detección y que son propias del método utilizado, es la incapacidad para detectar líneas verticales, ya que, detectan sólo líneas horizontales y que sean largas, por lo que, detectar un detector de líneas verticales cortas, para los bordes de los estantes implicaría una mejora del sistema detección.

Otro aspecto a considerar para mejorar el sistema es trazar líneas por segmentos y con una cierta pendiente, ya que, actualmente se está dibujando una línea recta, obteniéndose una diferencia considerable con aquellas imágenes que poseen bordes con cierto ángulo distinto de cero con respecto a la horizontal.

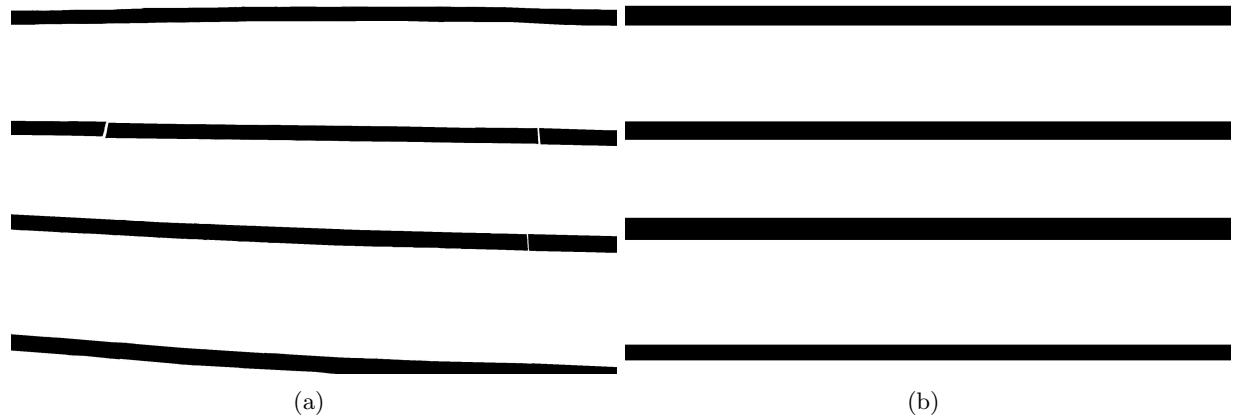


Figura 7: Máscara groundtruth y máscara detectada

4. Conclusión

La detección de estantes en góndolas de supermercados resulta, factible realizando sucesivas operaciones de filtrado y de detección de línea utilizando la transformada de Hough. Los desafíos de la detección radican en realizar un método robusto, que pueda trabajar con imágenes con distinta iluminación y con distintos grados de rotaciones.

Como trabajo futuro se propone considerar distintos criterios de decisión para detectar los bordes de un estante, se considera el presente trabajo como un aceptable punto de partida, al cual se le puede agregar análisis de líneas verticales, un criterio de decisión comparando los colores en el eje horizontal y las intensidades de brillo, ya que, éstas son distintas en los objetos de las góndolas y en los estantes.

5. Anexos

```

1 %% Imagen a color
2 clear
3 addpath('db1')
4 [a map]=imread('db45.jpg');
5 figure(1)
6 imshow(a,map);
7 title('Imagen original db45 ','fontsize',14)
8 imwrite(a,'Informe1/Imagenes/rgb.jpg')
9
10 %% Pasar a grises
11 ImagenGris=rgb2gray(a);
12 figure(2)
13 imshow(ImagenGris)
14 title('Imagen en escala de grises ','fontsize',14)
15 imwrite(ImagenGris,'Informe1/Imagenes/gris.jpg')
16 %% Filtro Sobel Horizontal
17 im_hor=edge(ImagenGris,'Sobel',0.05,'horizontal');%La dirección se
18 % usa solo para el Prewitt
19 % o el Sobel
20 figure(66)
21 imshow(im_hor)
22 title('Imagen resultante de aplicar filtro Sobel horizontal',...
23 'fontsize',14)
24 imwrite(im_hor,'Informe1/Imagenes/SobelHor.jpg')
25
26 %% Filtro Sobel vertical
27 im_vert=edge(ImagenGris,'Sobel',0.05,'vertical');%La dirección se
28 % usa solo para el Prewitt
29 % el Sobel
30 figure(4)
31 imshow(im_vert)
32 title('Imagen resultante de aplicar filtro Sobel vertical',...
33 'fontsize',14)
34 imwrite(im_vert,'Informe1/Imagenes/SobelVer.jpg')
35
36 %% Filtro de Canny
37
38 im_canny=edge(ImagenGris,'Canny',0.7);
39 figure(3)
40 imshow(im_canny)
41 imwrite(im_canny,'Informe1/Imagenes/canny.jpg')
42
43 %% Agrandar verticales
44 se90 = strel('line', 2, 90);%ancho=1, grados=90
45 im_vertDilate = imdilate(im_vert, [se90]);
46 figure
47 imshow(im_vertDilate)
48 title('Imagen resultante de agrandar las líneas verticales',...
49 'fontsize',14)
50 imwrite(im_vertDilate,'Informe1/Imagenes/SobelVerDilate.jpg')
51

```

```

52 %% Agrandar horizontales
53 se90 = strel('line', 2, 90); % ancho=1, grados=90
54 im_horDilate = imdilate(im_hor, [se90]);
55 figure
56 imshow(im_horDilate)
57 title('Imagen resultante de agrandar las líneas horizontales',...
58      'fontsize',14)
59 imwrite(im_horDilate, 'Informe1/Imagenes/SobelHorDilate.jpg')
60
61 %% Imagen resultante
62 [ im_out] = deleteVert(im_canny,im_vertDilate);
63 figure(5)
64 imshow(im_out)
65 title('Imagen resultante de aplicar Canny menos las líneas verticales dilatadas',...
66      'fontsize',14)
67 imwrite(im_out, 'Informe1/Imagenes/CannyMenosVert.jpg')

```

```

1 close all
2 addpath('db1')
3 rotI=im_horDilate;
4 [H,Theta,Rho] = hough(rotI);
5 P = houghpeaks(H,20, 'threshold', ceil(0.2*max(H(:)) ));
6 x = Theta(P(:,2));
7 y = Rho(P(:,1));
8 lines = houghlines(rotI,Theta,Rho,P, 'FillGap',20, 'MinLength',200);
9 figure, imshow(rotI), hold on
10
11 %% Código para generar imágenes
12 max_len = 0;
13 for k = 1:length(lines)
14     xy = [lines(k).point1; lines(k).point2];
15     plot(xy(:,1),xy(:,2), 'LineWidth',2, 'Color','green');
16
17     % Plot beginnings and ends of lines
18     plot(xy(1,1),xy(1,2), 'x', 'LineWidth',2, 'Color','yellow');
19     plot(xy(2,1),xy(2,2), 'x', 'LineWidth',2, 'Color','red');
20
21 end
22 title('Candidatos a bordes del estante',...
23      'fontsize',14)

```

```

1 %% Genera el resultado final
2 [ imout2] = drawEdge( im_out, lines ,70);
3 %% Rendimiento
4 addpath('groundT')
5 addpath('ground')
6 ground=imread('db45_ground.jpg');
7 [M] = rendimiento( im_out, ground )
8 %%
9 figure(1)
10 imshow(ground)
11 title('Máscara Groundtruth', 'fontsize',14)

```

```
12 imwrite(ground , 'Informe1/Imagenes/MGround.jpg')
13
14 figure(2)
15 imshow(imout2)
16 title('Máscara Detectada','fontsize',14)
17 imwrite(imout2 , 'Informe1/Imagenes/MDetectada.jpg')
```

```
1 function [ imout2 ] = drawEdge( im_out , lines ,u )
2
3 [f,c]=size(im_out);
4 npoint=length(lines);
5 for i=1:npoint
6     for m=1:npoint
7         xy1=lines(m).point1;
8         xy2=lines(i).point1;
9         if abs(xy1(1,2)-xy2(1,2))<u && abs(xy1(1,2)-xy2(1,2))>5
10            for p=1:abs(xy1(1,2)-xy2(1,2))
11                for k=1:c
12                    min_xy=min(xy1(1,2),xy2(1,2));
13                    im_out(min_xy+p,k)=1;
14                end
15            end
16        end
17    end
18 end
19 %
20 px0=find(im_out==0);
21 px1=find(im_out==1);
22 im_out(px0)=1;
23 im_out(px1)=0;
24 for q=1:f
25     n0=find(im_out(q,:)==0); % Detectar los pixels negros
26     if length(n0)<300 % Eliminar si hay pocos pixels negros en una fila
27         im_out(q,:)=1;
28     end
29 end
30
31 imout2=im_out;
32 end
```