Capítulo 8

DETECCION DE LINEAS Y CURVAS

En el capítulo 6 fue mostrado como con la ayuda de los filtros apropiados puede encontrase la magnitud del gradiente y su orientación en un determinado píxel de la imagen. A partir de la aplicación de estos filtros puede decidirse (quizás mediante la aplicación de un determinado umbral) si un píxel pertenece o no al borde de un objeto. Considerando este mapa binario obtenido, aparecerán líneas, círculos o alguna otra curva que describirá geométricamente el tipo de objeto contenido en la imagen. El tema de este capitulo es a partir de una imagen binaria producida, encontrar posibles formas contenidas en ella, las cuales podrían atribuirse a tipos de objetos previamente especificados.

1.1 ESTRUCTURAS EN UNA IMAGEN

Un enfoque intuitivo para encontrar estructuras en una imagen podría consistir en empezando de un determinado punto perteneciente a un borde, paso a paso añadir los píxeles que pertenecen al borde completo y con ellos determinar la estructura. La aproximación anterior puede ser intentada tanto en imágenes provenientes de una umbralización del gradiente así como de imágenes provenientes de algún tipo de segmentación. Esta aproximación sin embargo fallara al no considerar las fracturas y ramificaciones en los bordes producto del ruido y la incertidumbre propia de los algoritmos de cálculo del gradiente o la segmentación que no incorporan ninguna clase de criterio sobre las formas que se buscan en la imagen.

Un enfoque totalmente diferente es la búsqueda global de estructuras presentes en la imagen que de alguna manera se aproximan o relacionan a un tipo de forma previamente especificada. Como puede verse en la figura 8.1 a los ojos de un humano son claramente diferenciables los tipos de estructuras presentes en las imágenes a pesar de la otra gran cantidad de puntos añadidos. Es hasta ahora desconocido el mecanismo biológico responsable que permite relacionar y reconocer estructuras en las imágenes percibidas por humanos o animales. Una técnica que permite al menos desde el punto de vista computacional resolver este problema es la llamada Trasformada de Hough, la cual será tratada en detalle en este capitulo.

1.2 LA TRASFORMADA DE HOUGH

El método de la Trasformada de Hough ideado por Paul Hough y patentado en Los Estados Unidos de América es normalmente conocido en la comunidad de visión como la Trasformada de Hough. Dicha trasformada permite localizar formas paramétricas a partir de una distribución de puntos presentes en una imagen. Por formas paramétricas se refiere a líneas, círculos o elipses las cuales pueden ser descritas mediante la utilización de pocos parámetros. Debido a que este tipo de objetos (líneas, círculos y elipses) como muestra la figura 8.2 se presentan con frecuencia en imágenes, es de especial importancia encontrarlos en forma automática.

Se analizara primeramente el uso de la Trasformada de Hough para la detección de líneas sobre imágenes binarias producidas por la umbralización del gradiente, esta es una frecuente aplicación en muchos sistemas de visión o de procesamiento de imagen. Una línea en un espacio bidimensional es descrita mediante la utilización de dos parámetros reales tal y como se expresa de la siguiente manera:

$$y = kx + d \tag{8.1}$$

donde k representa la pendiente y d el punto del eje-y donde la línea lo intercepta (véase Figura 8.3). Una línea la cual pasa por dos diferentes puntos $p_1 = (x_1, y_1)$ y $p_2 = (x_2, y_2)$ debe de satisfacer las siguientes condiciones:

$$y_1 = kx_1 + d \quad \text{y} \quad y_2 = kx_2 + d \tag{8.2}$$

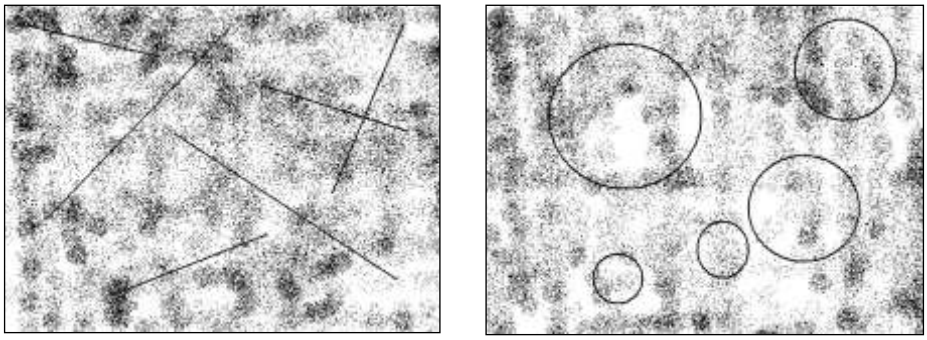


Figura. 8.1 Los mecanismos biológicos existentes en los humanos permiten identificar claramente las estructuras paramétricas de líneas y círculos sin incertidumbre a pesar del conjunto de puntos añadidos como ruido a la imagen.

Figura 8.2 Figuras paramétricas tales como líneas, círculos o elipses se presentan con frecuencia en imágenes.

donde $k,d \in \Re$. El objetivo por lo tanto es estimar los parámetros k y d de una línea la cual pasa por diferentes puntos pertenecientes a los bordes de un objeto. Ante esta situación surge la pregunta: ¿Cómo puede determinarse cuantos posibles puntos engloba una línea? Un enfoque posible seria dibujar todas las posibles líneas en una imagen y contar exactamente los puntos que pasan sobre cada una de ellas para acto seguido borrar todas aquellas líneas las cuales no contienen más de un determinado número de puntos. Lo anterior es posible realizar, sin embargo debido a que el número de líneas posibles que se trazarían sobre la imagen es muy grande lo harían especialmente ineficiente.

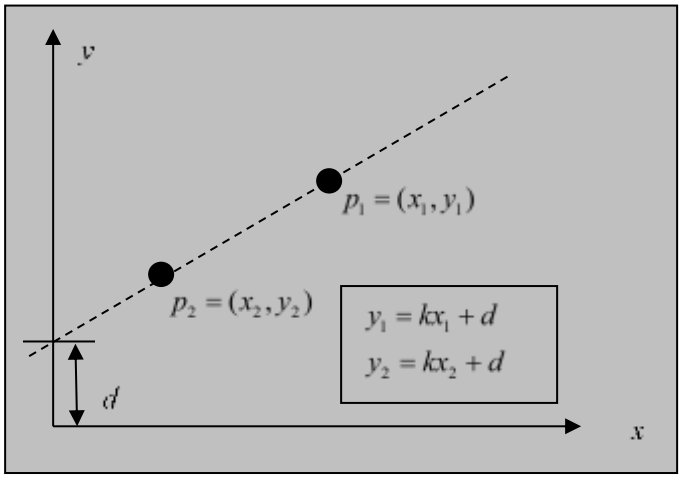


Figura 8.3 Dos puntos p_1 y p_2 que pertenece a la misma línea, por lo que para determinar la ecuación de la recta habrá que estimar los valores k y d .

1.2.1 El espacio de parámetros

La Trasformada de Hough resuelve el problema de la detección de líneas de una manera ligeramente diferente, al producir todas las posibles líneas que pasan a través de un píxel el cual corresponde a un borde de la imagen. Cada línea L_p la cual pasa a través de un punto $p_0 = (x_0, y_0)$ posee la siguiente ecuación:

$$L_P : y_0 = kx_0 + d \qquad (8.3)$$

donde los valores de k y d son variados para trazar todas las posibles líneas que tienen en común x_0 y y_0. El conjunto de soluciones para k y d de la ecuación 8.3 corresponde a una cantidad infinita de rectas las cuales pasan por el punto x_0 y y_0 (véase figura 8.4). Para un determinado valor de k se produce la correspondiente solución de 8.3 en función de d la cual implicaría:

$$d = y_0 - kx_0 \qquad (8.4)$$

Esta representa una función lineal, donde k y d son las variables y x_0 y y_0 las constantes de los parámetros considerados en la función. El conjunto de soluciones $\{(k,d)\}$ de la ecuación 8.4 describe los parámetros de todas las posibles rectas L_p las cuales pasan a través del punto $p_0 = (x_0, y_0)$.

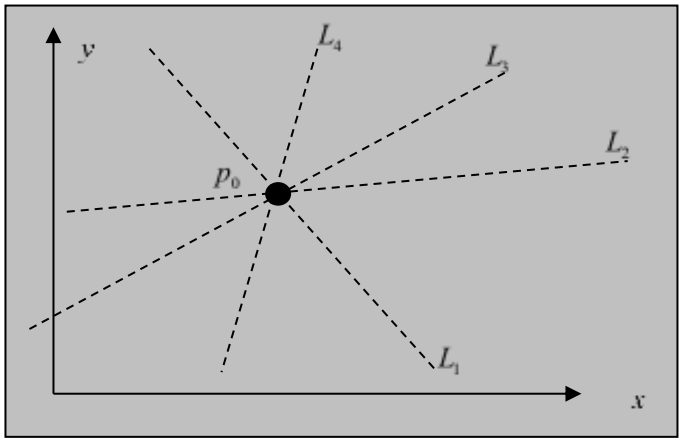


Figura 8.4 Conjunto de rectas las cuales pasan a través de un punto p_0 en común. Todas las posibles líneas L_p que pasan por p_0 tienen la expresión $y_0 = kx_0 + d$ donde los valores k y d son variables y determinan la diferencia entre cada una de las rectas definidas.

Para un determinado píxel de la imagen $p_i = (x_i, y_i)$ según la ecuación 8.4 le corresponde un conjunto de rectas definidas por:

$$R_i : d = y_i - kx_i \qquad (8.5)$$

Teniendo como parámetros variables k y d los cuales definen **el espacio de parámetros** también llamado el espacio de Hough, mientras que x_i y y_i son los parámetros fijos llamados también el espacio de parámetros de la imagen. La forma en como se relacionan ambos espacios es resumida en la tabla 8.1.

Cada punto p_i en el espacio de parámetros de la imagen corresponde a una recta en el espacio de parámetros de Hough. Considerando lo anterior se esta interesado en aquellos puntos donde las rectas establecidas en el espacio de parámetros de Hough se interceptan, que corresponderán a los valores de k y d que representan a la recta del espacio de la imagen que pasa por los puntos que formularon las rectas en el espacio de parámetros de Hough. Como se muestra en la figura 8.5, las rectas R_1 y R_2 se cortan en el punto $q = (k_{12}, d_{12})$ en el espacio de parámetros de Hough, las cuales representan a los puntos p_1 y p_2 en el espacio de parámetros de la imagen. Por lo que la recta en el espacio de la imagen tendrá una pendiente de k_{12} y un punto de intercepción en el eje y de d_{12}. Entre mas rectas en el espacio de parámetros de Hough se cortan significara que la línea en el espacio de la imagen estará formada por ese numero de puntos. Por lo que podría establecerse que:

> Si NR es el numero de rectas que se interceptan en $(\bar{k}, \bar{d})$ del espacio de parámetros de Hough, entonces habrá NR puntos que se encuentran sobre la línea definida por $y = \bar{k}x + \bar{d}$ en el espacio de la imagen.

Espacio de parámetros de la imagen (x, y)	**Espacio de parámetros de Hough** (k, d)
Punto: $p_i = (x_i, y_i)$	Línea: $R : d = y_i - kx_i$
Línea: $L : y = k_j x + d_j$	Punto: $q_j = (k_j, d_j)$

Tabla 8.1 Relación existente de puntos y líneas entre los diferentes espacios de parámetros.

1.2.2 Matriz de registros de acumulación

El método en la localización de rectas en una imagen se fundamenta en encontrar las coordenadas en el espacio de parámetros de Hough en donde varias líneas son cortadas. Para el cálculo de la trasformada de Hough se necesita primeramente discretizar de manera escalonada el intervalo de valores que corresponden a k y d. Al momento de realizar el conteo de cortes producidos por la intercepción de varias rectas en el espacio de parámetros de Hough, se utiliza una matriz de registros acumuladores en donde cada celda es incrementada en función de la cantidad de rectas que pasan por esa celda. De tal menara que el número final N_p almacenado en el registro significaría que esa recta en específico, se encuentra constituida de N_p píxeles del espacio de parámetros de la imagen. La figura 8.6 ilustra este proceso considerando como ejemplo el tomado en la figura 8.5.

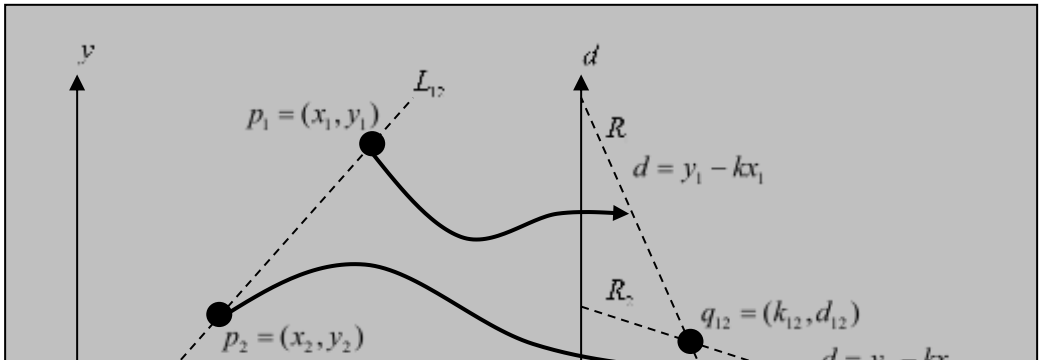


Figura 8.5 Relación entre el espacio de parámetros de la imagen y el espacio de parámetros de Hough. (a) Espacio de parámetros de la imagen y (b) espacio de parámetros de Hough. Puntos en el espacio de parámetros corresponden a rectas en el espacio de parámetros de Hough, mientras que lo contrario también es correcto, un punto en el espacio de parámetros de Hough corresponde a una línea en el espacio de parámetros de la imagen. En la imagen los puntos p_1 *y* p_2 *del espacio de parámetros de la imagen corresponden a las rectas* R_1 *y* R_2 *en el espacio de parámetros de Hough, así mismo el punto* $q_{12} = (k_{12}, d_{12})$ *del espacio de parámetros de Hough corresponde a la línea* L_{12} *del espacio de parámetros de la imagen.*

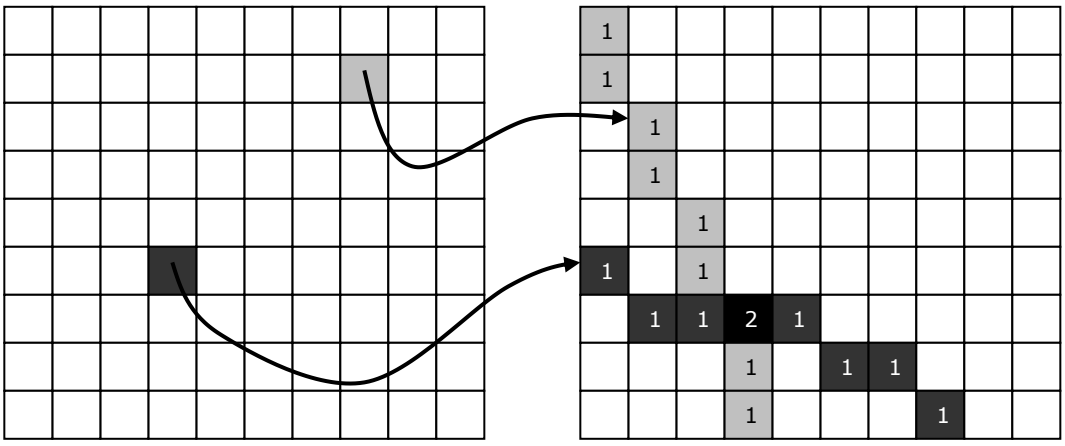
1.2.3 El cambio de modelo paramétrico

Hasta ahora ha sido mostrada la idea fundamental de la Trasformada de Hough. Sin embargo la representación de la recta definida en la ecuación 8.1 no puede ser utilizada debido al error computacional ocasionado en las líneas verticales donde $k = \infty$. Una mejor opción representa la utilización de la ecuación:

$$x \cdot \cos(\theta) + y \cdot \sin(\theta) = r \qquad (8.6)$$

La cual no presenta ninguna singularidad y además permite una cuantificación lineal de sus parámetros r y θ. La figura 8.7 ilustra el problema de singularidad de la ecuación 8.1 y muestra la manera de relacionar la ecuación 8.6 con el de la recta. Con la utilización de la ecuación 8.6 para la descripción de rectas, el espacio de parámetros de Hough varía. Por lo que el espacio de parámetros de las coordenadas r y θ y el punto $p_i = (x_i, y_i)$ del espacio de parámetros de la imagen se relacionan de acuerdo a la ecuación:

$$r_{x_i, y_i} = x_i \cdot \cos(\theta) + y_i \cdot \cos(\theta) \qquad (8.7)$$



(a) *(b)*

Figura 8.6 Idea fundamental de la Trasformada de Hough. (a) Espacio de parámetros de la imagen y (b) acumulador en el espacio de parámetros de Hough. Como puede la matriz de registros de acumulación es la versión discretizada del espacio de parámetros de Hough. Para cada punto en el espacio de parámetros de la imagen (a) se corresponde una línea en el espacio de parámetros de Hough. La operación que se desarrolla es aditiva, lo que significa que cada celda de la matriz es incrementada en un valor de uno conforme una recta pasa por el. De esta manera los puntos que mantienen un máximo local en el espacio de parámetros de Hough representan los valores k y d que representan a las rectas en el espacio de parámetros de la imagen.

Donde el intervalo de valores para θ es $0 \le \theta < \pi$ (véase la figura 8.8). Si se utiliza el centro de la imagen (x_c, y_c) como punto de referencia para definir coordenadas de los píxeles de la imagen (de tal forma que pudieran existir índices positivos y negativos tanto para x como para y), entonces el intervalo de valores de r se restringe a la mitad quedando definido por:

$$-r_{\max} \le r_{x,y}(\theta) \le r_{\max} \tag{8.8}$$

Donde:

$$r_{\max} = \sqrt{\left(\frac{M}{2}\right)^2 + \left(\frac{N}{2}\right)^2} \tag{8.9}$$

donde M y N representar el ancho y alto de la imagen.

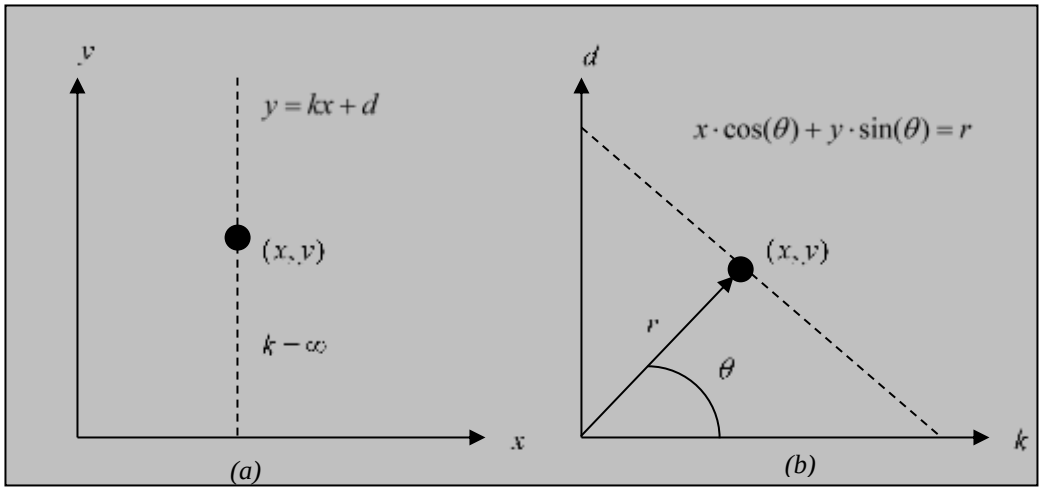
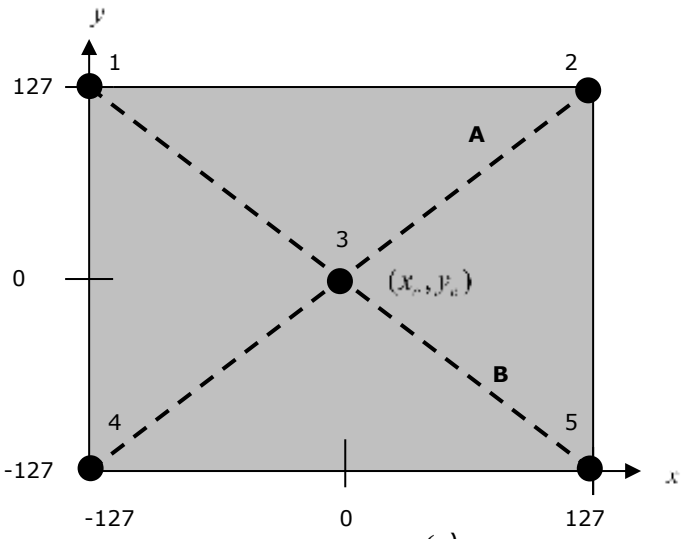


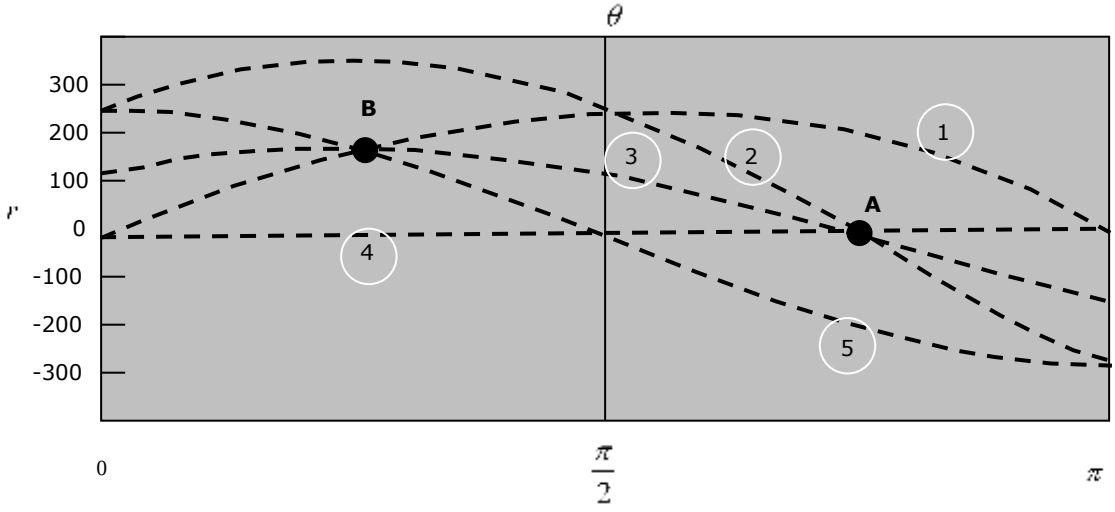
Figura 8.7 Diferentes tipos de parámetros que pueden utilizarse para caracterizar líneas, sin embargo la opción (a) muestra la singularidad para el caso de cuando se intenta caracterizar una recta vertical, ya que en ese caso se tiene que $k = \infty$, por lo que al elegir la caracterización de rectas por este tipo de parámetros se pudiera representar cualquier tipo de rectas exceptuando aquellas cercanas a la posición vertical, lo que limitaría la utilización del método. Un enfoque mejor seria la utilización de la ecuación 8.6 que además de no mostrar la singularidad permite una cuantización lineal de sus parámetros, donde la relación de la recta y los parámetros se encuentra caracterizada en (b), siendo la recta referida, la normal al vector definido por r y θ.

1.3 IMPLEMENTACIÓN DE LA TRANSFORMADA DE HOUGH

En el algoritmo 8.1 es mostrado la forma en como mediante la Transformada de Hough son encontrados los parámetros de las líneas presentes en una imagen, usando como modelo de recta el mostrado en la ecuación 8.6. Se utiliza como entrada al algoritmo una imagen binaria $I(x, y)$, la cual contiene información de los bordes (extraídos utilizando métodos como los explicados en el capítulo 6). En lo sucesivo se utilizará como convención que un uno en la imagen determina que el píxel es parte del borde y cero que es parte del fondo.



(a)



(b)

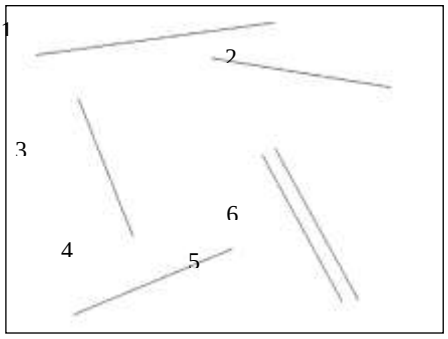
Figura 8.8 (a) Espacio de parámetros de la imagen y (b) espacio de parámetros de Hough usando como modelo de recta el establecido por la ecuación 8.6.

Detector de líneas por Trasformada de Hough ($I(x,y)$)
MRAcc(θ , r) $\rightarrow$ Matriz de registros acumuladores
(x_c, y_c) $\rightarrow$ Coordenadas del centro de $I(x,y)$

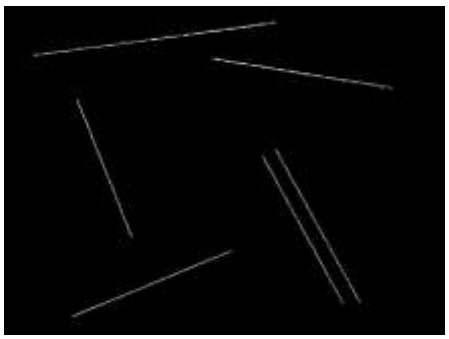
1: $0 \rightarrow$ **MRAcc(θ , r)**
2: **for** todas las coordenadas de la imagen $I(x,y)$ **do**
3: **if** ($I(x,y)$ es un borde) **then**
4: $(x - x_c, y - y_c) \rightarrow (u, v)$
5: **for** $\theta_i = 0 \ldots \pi$ **do**
6: $r = u \cdot \cos(\theta_i) + v \cdot \text{sen}(\theta_i)$
7: se incrementa **MRAcc(θ , r)**
8: Por ultimo se encuentran los registros **MRAcc(θ , r)** cuyos valores sean máximos.

Algoritmo 8.1 Algoritmo para detectar líneas utilizando la Trasformada de Hough.

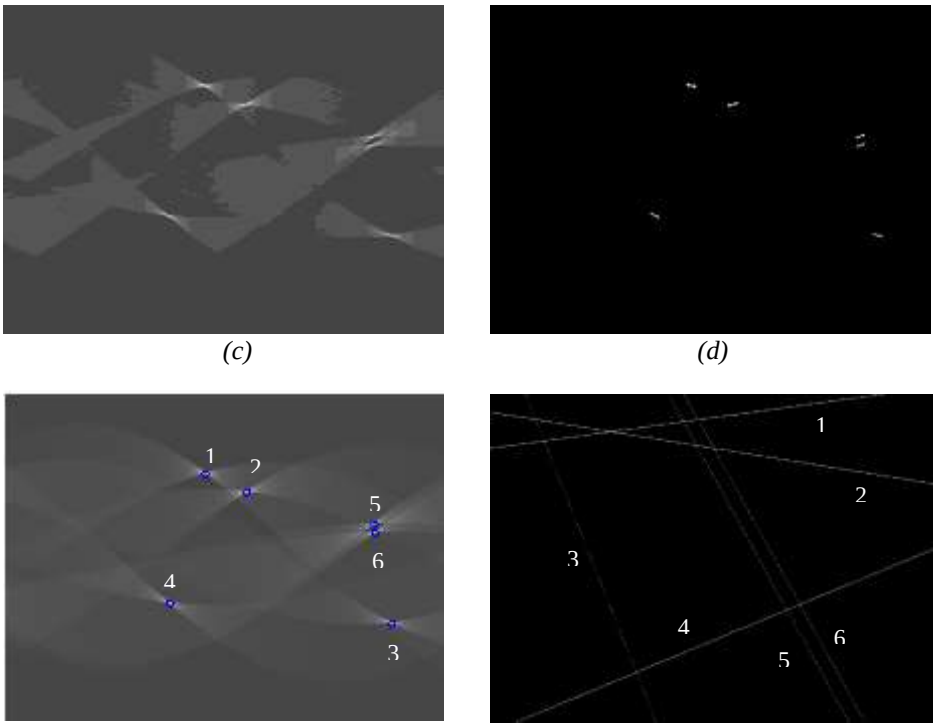
Como primer paso se colocan todas las celdas de la matriz de registros de acumulación a cero (sentencia 1). Después se recorre la imagen binaria (sentencia 3) $I(x,y)$, de tal forma de que cada vez que se encuentra un píxel borde (es decir que su valor sea uno), se obtiene los valores de r a partir de la ecuación 8.6, realizando un barrido del parámetro θ de 0 a π (sentencia 5 y 6). Sin embargo para encontrar los valores de r se emplean como coordenadas de referencia el centro (x_c, y_c) de la imagen $I(x,y)$. Para cada par (r,θ) obtenido del barrido, se incrementa el registro de la matriz de acumulación (sentencia 7) en los índices correspondientes a r y θ. De esta forma, una vez recorrido todos los píxeles de la imagen, los registros que se hayan incrementado, obteniendo los máximos locales (sentencia 8) que corresponderán a los valores de (r,θ) que definirán, a las líneas identificadas en la imagen $I(x,y)$. Para poder encontrar los máximos locales de la matriz de registros acumuladores, primero se aplica un umbral, de tal manera que solo los puntos mayores a ese umbral permanezcan, después se realiza una búsqueda en la imagen encontrando los máximos locales que se encuentren en la imagen considerando una determinada vecindad. El hecho de que los valores de varios registros vecinos al registro de valor máximo, cuyos índices r y θ representan a los parámetros que modelan a la recta real contenida en $I(x,y)$, tengan valores grandes, es debido al ruido e imprecisiones producidas por una defectuosa cuantizacion del espacio de parámetros de Hough.



(a)



(b)



(c) *(d)*

(e) *(f)*

Figura 8.9 Resultados parciales obtenidos de la aplicación del algoritmo de Hough para la detección de líneas. (a) Imagen original, (b) los bordes de la imagen original, (c) la matriz de los registros acumuladores del espacio de parámetros de Hough, (d) imagen obtenida de la aplicación del umbral a (c), (e) localización de los puntos máximos y (f) líneas obtenidas por la aplicación del algoritmo de Hough.

La figura 8.9 muestra una serie de imágenes que representan los resultados parciales del algoritmo 8.1. La figura 8.9(a) representa la imagen original producida artificialmente para ejemplificar el uso del algoritmo de Hough para la detección de líneas, la figura 8.9(b) muestra los bordes obtenidos por la aplicación del algoritmo de Canny (véase capitulo 6). En 8.9(c) Se muestra el contenido de la Matriz de registros acumuladores de la imagen. Puede observarse que los puntos brillantes representan aquellas localidades cuyos valores han sido incrementados significativamente, convirtiéndose en puntos altamente potenciales de los parámetros reales r y θ que representan a las líneas reales de la imagen original $I(x,y)$**.** La figura 8.9(d) muestra la imagen producida mediante la aplicación de un umbral a la matriz de registros acumuladores. En la figura 8.9(e) se localiza los registros que tienen un valor máximo considerando una determinada vecindad. Por ultimo en la Figura 8.9(f) se muestra las rectas encontradas mediante el algoritmo de Hough de parámetros r y θ.

De la figura 8.9(e) puede verse de los puntos máximos obtenidos, que los puntos mas cercanos (5 y 6), prácticamente contiguos, representan a las líneas paralelas de la figura 8.9(a) que poseen la misma pendiente, es decir, mismo valor de θ.

1.4 LA TRASFORMADA DE HOUGH IMPLEMENTADA EN MATLAB.

En esta sección se implementa la trasformada de Hough en la detección de líneas utilizando MatLAB como lenguaje de programación. En esta sección no se hace uso de las herramientas ya implementadas de MatLAB y Simulink que permiten detectar directamente las líneas en la imagen mediante la utilización de funciones y bloques monolíticos.

Para la implementación de la trasformada de Hough es necesario considerar 3 diferentes partes del código. El programa 8.1 muestra el código comentado que implementa la trasformada de Hough usada para la detección de líneas.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Programa que implementa la transformada de Hough
% usada para la detección de líneas
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Erik Cuevas, Daniel Zaldivar, Marco Pérez
% Versión: 02/02/08
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Se obtienen las dimensiones de la imagen binaria BW
%donde los píxeles que tienen el valor de 1 es que son
%parte del borde, mientras que cero, significaría
%que son parte del fondo
[m n]=size(BW);
%PARTE UNO (se obtiene MRAcc)
%Se inicializan algunas matrices que participan
%en el procesamiento.
%Se inicializa la Matriz de registros de acumulación
MRAcc=zeros(m,n);
%Se inicializa la matriz donde se almacenan los
% máximos locales obtenidos.
Fin=zeros(m,n);

%Se define como referencia de coordenadas
%el centro de la imagen
m2=m/2;
n2=n/2;
%Se calcula el valor máximo de r dependiendo
%de las dimensiones de la imagen (véase ecuación 8.9).
rmax=round(sqrt(m2*m2+n2*n2));
%Se obtiene el escalamiento lineal de los
%parámetros de Hough, Tetha y r.
iA=pi/n;
ir=(2*rmax)/m;
%Se recorre la imagen BW poniendo atención en
%los puntos bordes donde BW es uno.

for re=1:m
    for co=1:n
        if(BW(re,co))
            for an=1:n
                %Se considera como referencia
                %el centro de la imagen.
                x=co-n2;
                y=re-m2;
                theta=an*iA;
                %se obtiene el valor de r a partir de 8.6
                r=round(((x*cos(theta)+y*sin(theta))/ir)+m2);
                if((r>=0)&&(r<=m))
                   %Se incrementa la celda en uno la celda
                   %correspondiente a los parámetros r y theta
```

```
                    MRAcc(r,an)= MRAcc(r,an)+1;
                end
            end
        end
    end
end
%PARTE 2 (Se selecciona el registro máximo localmente)
%Se le segmentan los píxeles de MRAcc aplicando
%como umbral (th) 100. De esta manera los píxeles de
%Bandera que sean uno representaran a aquellos
%registros que constituyen líneas formadas de
%al menos 100 puntos.
Bandera=MRAcc>100;
%Se establece para la búsqueda del máximo
%una vecindad de 10 píxeles.
pixel=10;
%Se barre la imagen en busca de los puntos
%potenciales
for re=1:m
    for co=1:n
        if(Bandera(re,co))
            %Se establece el la región de vecindad
            %de búsqueda
            I1=[re-pixel 1];
            I2=[re+pixel m];
            I3=[co-pixel 1];
            I4=[co+pixel n];
            datxi=max(I1);
            datxs=min(I2);
            datyi=max(I3);
            datys=min(I4);
            Bloc=MRAcc(datxi:1:datxs,datyi:1:datys);
            MaxB=max(max(Bloc));
            %Se selecciona el píxel de valor máximo contenido
            %en esa vecindad.
            if(MRAcc(re,co)>=MaxB)

                %El píxel de valor máximo es
                %marcado en la matriz Fin.
                Fin(re,co)=255;

            end
        end
      end
end
%Se obtienen las coordenadas de los píxeles
%cuyo valor fue el máximo, que representara a
% los registros, los cuales sus índices representan
%a los parámetros de las líneas detectadas.
[dx dy]=find(Fin);
%PARTE 3 (Las líneas encontradas se despliegan).
%Se obtiene en indx el numero de líneas detectadas
%que implica el número de elementos de Fin.
[indx nada]=size(dx);
rm=m;
cn=n;
apunta=1;
%Se inicializa a cero la matriz M donde se desplegaran
%las líneas encontradas
M=zeros(rm,cn);
%Se despliegan todas las líneas encontradas
for dat=1:indx
%Se recuperan los valores de los parámetros de
```

```
%las líneas encontradas
pr=dx(dat);
pa=dy(dat);
%Se considera que los valores de los parámetros
%se definen considerando el centro de la imagen
re2=rm/2;
co2=cn/2;
%Se escalan los valores de r y theta
pvr=(pr-re2)*ir;
pva=pa*(pi/cn);
%Se obtienen las proyecciones verticales y
%horizontales de r, ya que r es el vector definido
%en el origen y perpendicular a la línea detectada
x=round(pvr*cos(pva));
y=round(pvr*sin(pva));
%Se elimina el offset considerado por utilizar como
%referencia el centro de la imagen
Ptx(apunta)=x+co2;
Pty(apunta)=y+re2;
%Se incrementa el índice considerado para
%apuntar a los parámetros encontrados que definen
%el numero de líneas
apunta=apunta+1;
%Se barre el modelo de recta con los parámetros
%detectados y almacenados en la matriz de registros
%acumuladores.
%Primero en un sentido.
for c=x:1:co2
    r=round((-1*(x/y)*c)+y+(x*x/y))+re2;
    if((r>0)&&(r<rm))
        M(r,c+co2)=1;
    end
end


% después en el sentido no considerado.
for c=x:-1:1-co2
     r=round((-1*(x/y)*c)+y+(x*x/y))+re2;
    if((r>0)&&(r<rm))
        M(r,c+co2)=1;
    end
end
end
```

Programa 8.1 Implementación en MatLAB de la transformada de Hough.

En la primera parte además de inicializar algunas variables se implementa la transformada de Hough indicada por los pasos 1 a 7, del algoritmo 8.1.

La segunda parte del programa 8.1 lo esta integrado por las instrucciones necesarias para la elección de los píxeles cuyo valor de **`MRAcc`** sea máximo dentro de un bloque considerado como la región de vecindad. Para programar tal hecho se recorre secuencialmente píxel a píxel la matriz (a la cual se le aplico el umbral t_h) encontrando el valor máximo de un bloque o región de vecindad $R(x, y)$ establecida alrededor del punto en cuestión. Para ello se fijan los límites del bloque relativos al píxel en cuestión, en el caso del programa 8.1 se fija el intervalo igual a 10, que considerándolo en ambos sentidos sería 20. Una vez establecido los limites se extrae el bloque y se encuentra el máximo, si el máximo del bloque corresponde al píxel central del bloque, relativo al del proceso de barrido, se coloca un uno en esa posición en la matriz **Fin**, de tal forma

que los índices donde se encuentren unos en **Fin** corresponderán a los parámetros que definen las líneas de la imagen original. La figura 8.10 muestra el proceso de encontrar el máximo dentro de la vecindad considerada alrededor del punto en cuestión.

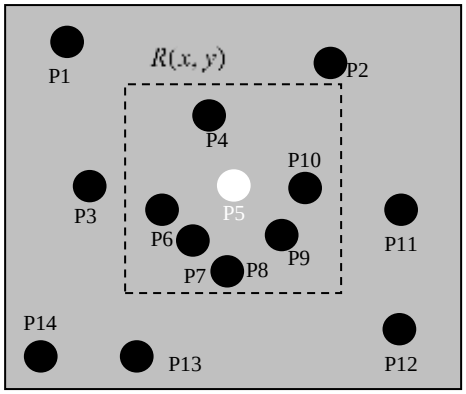


Figura 8.10 Proceso de obtención del registro de los acumuladores significantes. De todos los puntos encontrados mediante la aplicación del umbral t_h, se selecciona aquel cuyo valor **`MRAcc`** *es el máximo en una vecindad definida $R(x, y)$. En este caso el valor de punto P5 pose el valor máximo de* **`MRAcc`** *en comparación a los demás valores P4, P6, P7, P8, P9 y P10 que se encuentran dentro de la región de vecindad definida por $R(x, y)$.*

En la tercera parte se despliega las rectas encontradas mediante la aplicación de la trasformada de Hough. Para ello se considera que el valor de los parámetros r y θ representan al vector el cual es perpendicular a la recta que representa en realidad a la detectada en el espacio de parámetros de la imagen. Para formular el modelo de la recta a desplegar lo único que hay que considerar es que si la pendiente del vector se encuentra definida por $m = p_y / p_x$, considerando que p_y y p_x es su proyección vertical y horizontal respectivamente, la pendiente de la recta detectada la cual es la normal a este vector, será la inversa y opuesta a la del vector $(-p_x / p_y)$. La figura 8.11 muestra una ilustración de este proceso.

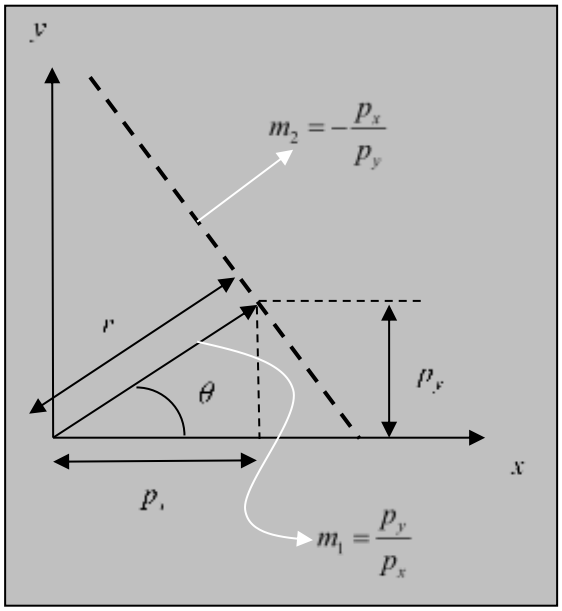


Figura 8.11 Pendiente de la recta detectada, la cual es la inversa y opuesta a la del vector definido por los parámetros de la trasformada de Hough.

Las imágenes contenidas en la figura 8.9 fueron obtenidas mediante la ejecución del programa 8.1.

1.5 FUNCIONES DE MATLAB PARA LA DETECCIÓN DE LÍNEAS

MatLAB contiene 3 funciones, `hough`, `houghpeaks` y `houghlines` las cuales al aplicarlas secuencialmente permiten la detección de líneas en una imagen. El núcleo de estas funciones lo representa la trasformada de Hough implementada en la función hough, mientras que las otras dos `houghpeaks` y `houghlines` son utilizadas como funciones auxiliares de detección y despliegue.

Como fue explicado este conjunto de funciones deben ser utilizadas secuencialemente para una aplicación de detección de líneas. Las operaciones desarrolladas por cada una de estas funciones corresponden con las partes en las que fue dividido el programa 8.1 codificando en MatLAB para la misma funcionalidad.

La función `hough` implementa la trasformada de Hough, generando la matriz de registros de acumulación, la función `houghpeaks` determina los registros cuyo valor máximo definen los parámetros de las líneas detectadas, por último la función `houghlines` permite desplegar las líneas encontradas por la combinación de las funciones `hough` y `houghpeaks`.

La función `hough` permite calcular la transformada de Hough sobre una imagen binaria que como prerrequisito contiene los bordes de una imagen. La sintaxis de esta función es:

```
[H, theta, rho] = hough(BW)
```

Donde `BW` es una imagen binaria que contiene los bordes. La función regresa 3 diferentes valores que son `H`, la cual contiene la matriz de registros acumuladores, y dos vectores `theta` y `rho` que contienen información de cómo linealmente fue dividido los parámetros r y θ en la matriz de registros acumuladores de `H`.

Cuando la función hough se utiliza con la sintaxis mostrada, divide linealmente los parámetros r y θ, en 1997 valores al parámetro r y en 180 al valor de θ. Por lo que el tamaño de la matriz de registros acumuladores será de 1997x180. La resolución de `rho` depende, como es mostrado en la ecuación 8.9 del tamaño de la imagen, mientras que el de `rho` es de 1 grado, ya que son 180 diferentes índices para representar 180 grados.

La función `houghpeaks` permite localiza los valores máximos de la matriz de registros de acumulación. La sintaxis de esta función es:

```
peaks = houghpeaks(H, numpeaks, 'Threshold', val)
```

donde `H` es la matriz de registros acumuladores calculada por la función `hough`. El valor `numpeaks` es un escalar que especifica el número de valores máximos a identificar en `H`. Si el valor de `numpeaks` es omitido en la sintaxis el valor por defecto es 1. `'Threshold'` es un escalar que especifica el

umbral a partir del cual los valores de `H` son considerados máximos. Una forma sencilla de determinar un umbral que sea correcto y adecuado para la matriz de registros de acumulación es el utilizar una parte proporcional del valor máximo encontrado. Un ejemplo lo representa la expresión `0.5*max(H(:))`, la cual implica que el umbral definido, es el 50% del registro de valor máximo.

La función `houghpeaks` regresa como resultado una matriz de dimensiones Qx2, donde Q es un valor de 0 a `numpeaks` (valor de máximos a identificar) y 2 implica a los índices correspondientes a los parámetros r y θ que definen al registro acumulador encontrado como máximo.

La función `houghlines` extrae las líneas asociadas a la imagen `BW` calculadas por la transformada de Hough, implementada a partir de la combinación de las funciones `hough` y `houghpeaks`. La sintaxis de esta función es definida por:

```
lines = houghlines(BW, theta, rho, peaks)
```

donde `BW` es la imagen binaria a la cual se le detectaron las líneas a través de la trasformada de Hough, `theta` y `rho` son los vectores que definen la forma en como linealmente fue dividida la matriz de registros acumuladores y a la vez ambos calculados por la función `hough`. Por otra parte `peaks` es una matriz que fue calculada por la función `houghpeaks` de dimensión Qx2, donde Q es el numero de líneas detectadas y 2 los índices que corresponden a los parámetros r y θ.

La función `houghlines` regresa como resultado un arreglo de estructuras llamado `lines`. Una estructura es una agrupación de datos de tipo diferente bajo un mismo nombre. Estos datos se llaman ***campos*** (*fields*) y son accedidos mediante el formato:

```
Nombre_de_la _estructura.campo1=3;
```

Por lo que en el ejemplo anterior al `campo1`, parte de la estructura `Nombre_de_la_estructura` se le asigno el valor de 3. Un arreglo de estructuras es un conjunto de estructuras controladas por un índice que pueden accedidas mediante el formato

```
Nombre_de_la _estructura(1).campo1=3;
```

donde el valor entre paréntesis se refiere a una estructura identificada por el índice 1.

Los campos que posee la estructura `lines` son `point1`, `point2`, `theta` y `rho`. Sus significados se encuentran resumidos en la tabla 8.1.

Campo	Descripción
point1	Es un vector de 2 elementos (x,y) que especifica el inicio del segmento.
point2	Es un vector de 2 elementos (x,y) que especifica el final del segmento.
theta	Vector que contienen información de cómo linealmente fue dividido el parámetro θ.
rho	Vector que contienen información de cómo linealmente fue dividido el parámetro r.

Tabla 8.1 Campos de la estructura devuelta por la función `houghlines`.

1.5.1 Ejemplo de detección de líneas usando las funciones de MatLAB

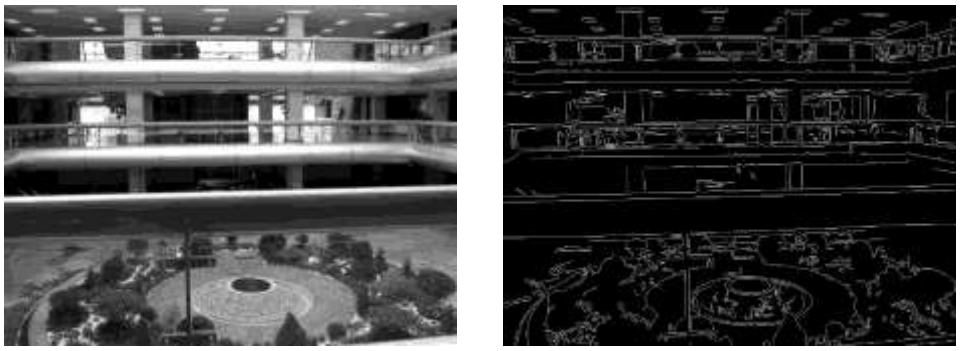
Como final de esta sección se establece un ejemplo que integre la forma en como las funciones del toolbox de procesamiento de imágenes de MatLAB, descritas anteriormente, son utilizadas para la detección de líneas en una imagen.

Considerando como imagen `Im` a la cual se le desea extraer sus líneas como la mostrada en la figura 8.12(a), se le extraen sus bordes mediante la aplicación de la función:

```
BW = edge(Im,'canny',0.7,0.1);
```

Dando como resultado la imagen 8.12(b). A partir de esta imagen se aplica la trasformada de hough mediante la ejecución de la función:

```
[H, theta, rho] = hough(BW);
```



(a) *(b)*

Figura 8.12 (a) Imagen original `Im` *y (b) imagen a ala cual se le calcularon sus bordes* `BW`.

Para poder graficar la matriz de registros de acumulación `H`, habrá primeramente convertir esta matriz del tipo `double` al tipo `uint8`, escribiendo:

```
Hu=uint8(H);
```

Para después realizar la graficación mediante la ejecución de la siguiente secuencia:

```
imshow(Hu,[],'XData',theta,'YData',rho,'InitialMagnification','fit')
```

Donde los valores de los parámetros `'XData'` y `'YData'` sustituyen a los índices normales de la matriz `Hu` y la combinación de los parámetros `'InitialMagnification','fit'` permite desplegar la imagen a escala completa de la ventana que contiene a la grafica. El resultado se pone

de manifiesto en la figura 8.13(a). Puede de igual manera mejorar el despliegue mediante la utilización del comando:

```
axis on, axis normal
```

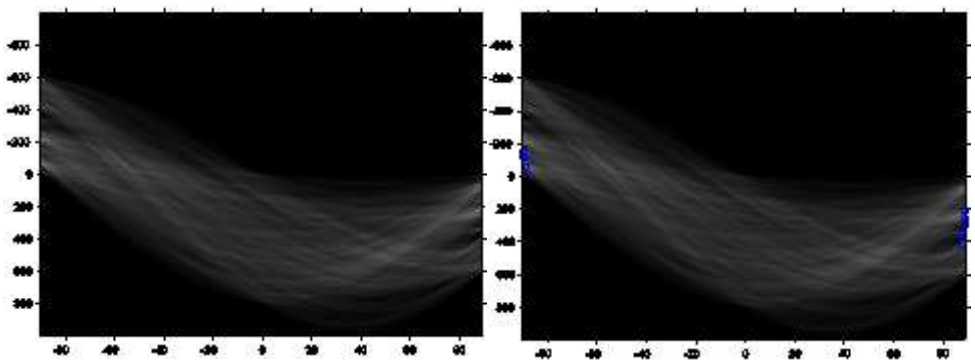
Mediante la aplicación de la función:

```
P = houghpeaks(H,8,'threshold',ceil(0.3*max(H(:))));
```

Se obtienen como máximo los 8 puntos de mayor valor de la matriz de registros de acumulación `H`, considerando como umbral el 30% del valor encontrado como máximo. Para poder desplegar los puntos máximos encontrados de la matriz de acumulación se ejecuta la siguiente secuencia de comandos:

```
hold on
x = theta(P(:,2));
y = rho(P(:,1));
plot(x,y,'s');
```

Los cuales permiten retener la imagen para colocar objetos sobre ella y obtener los vectores de las coordenadas x e y de los puntos máximos. La imagen obtenida por la anterior secuencia se muestra en la figura 8.13(b).



(a) *(b)*

Figura 8.13 (a) Matriz de registros de acumulación y (b) los puntos máximos marcados con cuadros azules, tras la acción del comando `plot(x,y,'s')`.

Por ultimo se obtienen las líneas correspondientes a las de los parámetros encontrados mediante la función `houghlines`. Sin embargo debido a que el colocar todos los segmentos de líneas encontrados presupone hacerlo mediante un método de repetición se programa un archivo.m que permite encontrar los segmentos de línea y desplegar las líneas una a una sobre la imagen de bordes `BW`. El archivo.m se muestra codificado en el programa 8.2.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Programa que despliega las líneas calculadas por las
% funciones hough, houghpeaks sobre BW
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Erik Cuevas, Daniel Zaldivar, Marco Pérez
% Versión: 03/02/08
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
lines = houghlines(BW,theta,rho,P);
Imshow(BW);
hold on
max_len = 0;
%se barren los arreglos de estructuras encontrados lines
%que contienen los valores de las líneas
for k = 1:length(lines)
xy = [lines(k).point1; lines(k).point2];
plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
% Se grafica el inicio y final de las líneas
plot(xy(1,1),xy(1,2),'x','LineWidth',2,'Color','yellow');
plot(xy(2,1),xy(2,2),'x','LineWidth',2,'Color','red');
%Se determina el final del segmento mas largo
len = norm(lines(k).point1 - lines(k).point2);
if ( len > max_len)
    max_len = len;
xy_long = xy;
end
end
% Se resalta los segmentos largos
plot(xy_long(:,1),xy_long(:,2),'LineWidth',2,'Color','cyan');
```

Programa 8.2 Este programa permite encontrar los segmentos de línea y desplegar las líneas una a una sobre la imagen de bordes `BW`.

La figura 8.14 muestra el resultado de aplicar el programa 8.2.

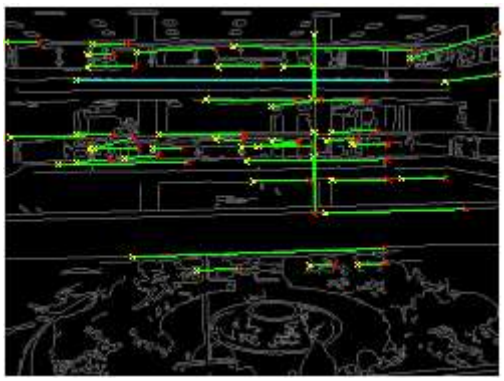


Figura 8.14 Imagen BW sobre la cual se colocan los segmentos de líneas encontradas por la combinación de las funciones `hough`, `houghpeaks` *y* `houghlines`. *Para el despliegue de las líneas se uso el código mostrado en el programa 8.2.*

1.6 BLOQUES DE SIMULINK PARA LA DETECCIÓN DE LÍNEAS

La librería para el procesamiento de imágenes y video de Simulink posee una serie de bloques que permiten en su conjunto detectar líneas en imágenes. Al igual que en la sección pasada, que fue necesario aplicar en secuencia una serie de funciones para detectar las líneas de una imagen, también en el caso de los bloques de Simulink se debe colocar un conjunto de bloques con funcionalidades bien definidas para la detección completa de líneas en imágenes.

Esta sección comienza con una descripción de los bloques que permiten en su conjunto implementar la transformada de Hough y termina con la descripción de bloques que de manera auxiliar ayudan a la detección de las líneas de una imagen.

Hough Transform (Transformada de Hough)

Este bloque de la librería de procesamiento de imágenes y video de Simulink, ubicado en la categoría "Transforms" (transformadas), permite encontrar líneas en una imagen mediante el cálculo de la matriz de registros acumuladores. Como ya ha sido visto el bloque realiza un mapeo de espacio de los puntos cartesianos de la imagen al espacio de parámetros de Hough, mediante la relación:

$$r = x \cdot \cos(\theta) + y \cdot \sin(\theta) \qquad (8.10)$$

La figura 8.15 muestra la representación de este bloque. Como puede verse el bloque recibe como entrada una imagen binaria BW correspondiente a los bordes de la imagen, mientras que como salida devuelve tres diferentes salidas, la matriz de registros de acumulación Hough y dos vectores Theta y Rho que contienen información de cómo linealmente fue dividido los parámetros r y θ en la matriz de registros acumuladores de Hough.

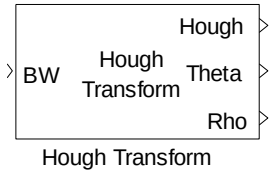


Figura 8.15 Bloque que implementa la transformada de Hough, calculando la matriz de registros acumuladores Hough.

El bloque en su funcionamiento por defecto divide linealmente los parámetros r y θ, en 1997 valores al parámetro r y en 180 al valor de θ. Por lo que el tamaño de la matriz de registros acumuladores será de 1997x180. La resolución de Rho depende, como es mostrado en la ecuación 8.9 del tamaño de la imagen, mientras que el de Theta es de 1 grado, ya que son 180 diferentes índices para representar 180 grados. Sin embargo el bloque puede ser configurado para desempeñar otro tipo de división lineal. La figura 8.16 muestra la ventana de configuración de este bloque, del cual puede ser observado que mediante la manipulación de los campos "Theta resolution" (resolución del ángulo, 1) y "Rho Resolution" (resolución del radio, 2), puede ser cambiado la resolución de los parámetros r y θ.

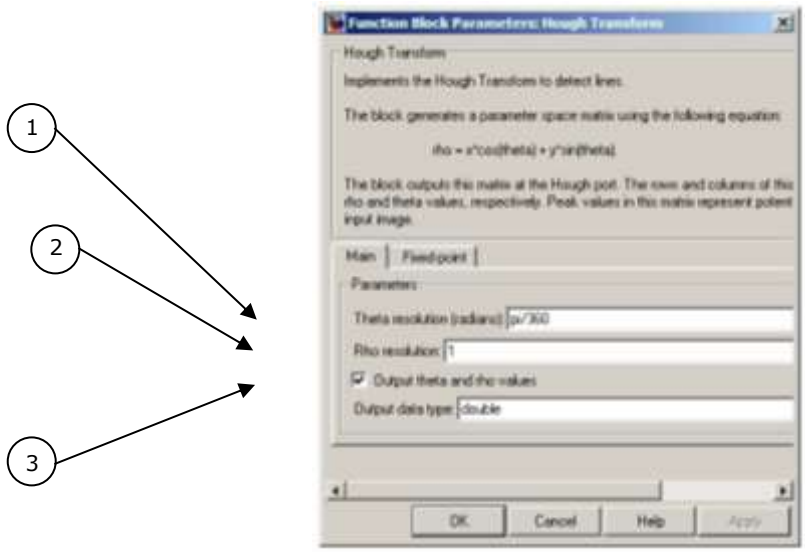


Figura 8.16 Ventanilla de configuración del bloque de Hough Transform.

Algo importante de la configuración de este bloque es la necesidad de seleccionar que el bloque devuelva como salidas (3) los valores de theta y rho. Lo anterior se debe al hecho de que la resolución de estos parámetros puede cambiar, por lo que para recuperar los parámetros de la matriz de los registros acumuladores es necesario conocer la correspondencia de los índices con los de la división lineal ejecutada.

Hough Lines (Obtención de líneas de Hough)

Este bloque de la librería de procesamiento de imágenes y video de Simulink, ubicado en la categoría "Transforms" (transformadas), localiza las coordenadas de las líneas descritas a partir de los parámetros r y θ. La figura 8.17 muestra la representación de este bloque. El bloque recibe como entradas los valores de rho y theta, que definen el vector al cual se mapearon los puntos de la imagen mediante la ecuación 8.10. A partir de estos datos y la imagen binaria BW acoplada a la entrada Ref I, de la cual solo son importantes las dimensiones, se calculan las coordenadas de las líneas definidas.

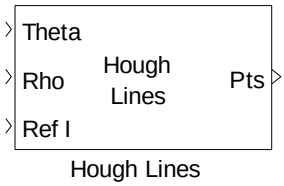
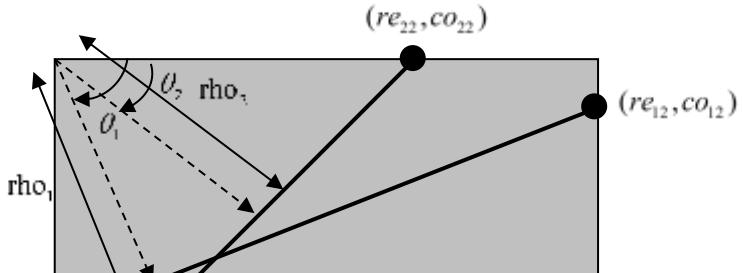


Figura 8.17 Bloque que implementa la localización de líneas a partir de los valores parametritos de r y θ, calculando la matriz de registros acumuladores Hough.

La información calculada por este bloque son las coordenadas de los puntos que definen el inicio y final del segmento de línea detectado y correspondiente a los valores de Rho y Theta. La figura 8.18 muestra los conceptos y nomenclatura envueltos en el proceso de localización.



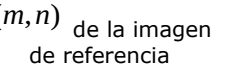


Figura 8.18 Nomenclatura y parámetros involucrados en la localización de los segmentos de líneas a partir de los parámetros de Hough r y θ.

La salida de este bloque es una matriz cuyos índices empiezan en (0,0) con las coordenadas de las intersecciones de las líneas con las fronteras de la imagen de referencia. Las dimensiones de esta matriz es de 4xNL, donde NL es el número de segmentos de líneas. Para el caso ejemplificado de la figura 8.18 se tendría que la salida Pts de del bloque Hough Lines seria:

$$\begin{bmatrix} re_{11} & re_{21} \\ co_{11} & co_{21} \\ re_{12} & re_{22} \\ co_{12} & co_{22} \end{bmatrix} \qquad (8.1)$$

Una parte importante para el funcionamiento del bloque es la configuración para que trabaje, en lugar de utilizar una tabla para realizar los cálculos trigonométricos de conversión se utiliza la función trigonometríca (1) como tal. Para ello solo habrá que seleccionar esta opción en su ventanilla de configuración. La figura 8.19 muestra la ventanilla de configuración y el proceso de elección.

A continuación serán descritos bloques de Simulink que de manera auxiliar permiten la detección de líneas.

Find Local Maxima (Encontrar máximos locales)

Este bloque encuentra los máximos locales presentes en una matriz, considerando para ello una determinada vecindad. La figura 8.20 muestra una representación de este bloque.

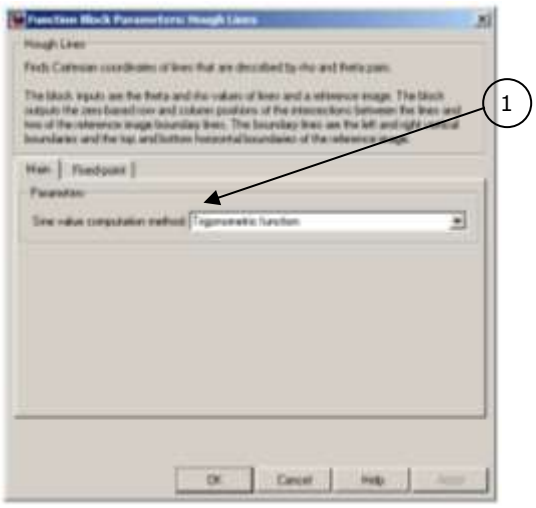


Figura 8.19 Ventanilla de configuración del bloque Hough Lines.

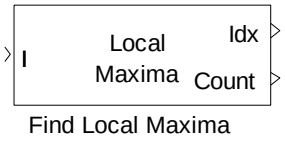


Figura 8.20 Bloque "Find Local Maxima" de Simulink.

El bloque recibe como entrada una matriz I, y devuelve en Idx el valor de los índices de I donde se encontraron los puntos máximos, mientras que Count señaliza el número de máximos validos. Varios parámetros de este bloque pueden ser configurados los cuales tienen incidencia directa en los resultados obtenidos. La figura 8.21 muestra la ventanilla de configuración de este bloque.

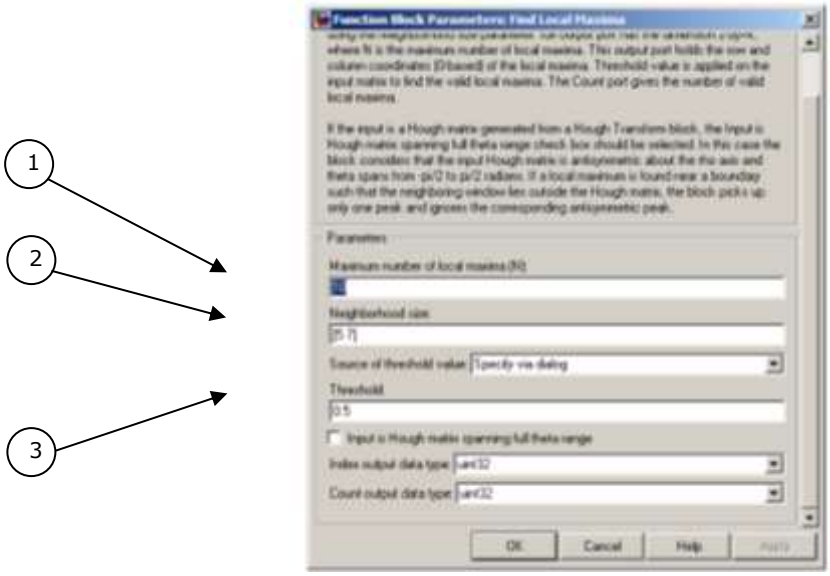


Figura 8.21 Ventana de configuración del bloque "Find Local Maxima".

Dentro de los parámetros importantes se tiene (1) el numero total de máximos locales ("Maximum number of local maxima (N)"), (2) la región de vecindad ("Neighborhood size") y (3) el umbral. De tal manera que el funcionamiento del bloque es el siguiente: considerando como entrada la matriz I se le aplica un umbral definido en (3) a la imagen a partir del cual se tienen un menor numero de puntos potenciales de búsqueda en relación a toda la imagen. Con estos puntos se hace una búsqueda en la matriz en cada uno de los puntos encontrando el máximo dentro de una vecindad definida en (2). El número total de máximos se define en (1).

La salida de este bloque es Idx que es un vector de 2xNp, donde Np es el número de máximos obtenidos y el índice 2 las coordenadas de los puntos máximos. Si por alguna razón el número total de máximos locales definidos en (2) excede el número de máximos encontrados la salida Count arrojara un escalar que indicara el número de máximos verdaderos que definen solamente los que deben de ser tomados en cuenta.

Variable Selector (selector de variables)

Este bloque de la librería de procesamiento de imágenes y video de Simulink, ubicado en la categoría "Utilities" (utilerías), permite extraer un renglón o una columna (o un conjunto de renglones o columnas) de una matriz. La figura 8.22 muestra la representación de este bloque en sus dos configuraciones posibles, (a) selección de renglones y (b) selección de columnas.

(a) *(b)*

Figura 8.22 Representación del bloque "Variable Selector". (a) Configuración para seleccionar columnas y (b) configuración para seleccionar renglones.

Como resulta evidente el bloque soporta varios parámetros de configuración los cuales definen su funcionalidad. La figura 8.23 muestra su ventana de configuración. En el bloque puede configurarse el numero de entradas con las que se operara al modificar el campo (1) "Number of input signals" (numero de señales de entrada). Mediante la manipulación del parámetro (2) "Select" (selección) se selecciona si se opera sobre renglones (row) o sobre columnas (columns). Al operar este campo, el bloque cambia su funcionalidad y para hacerlo evidente modifica su etiqueta principal, tal y como se muestra en la figura 8.22.

Mediante la modificación del campo (3) "Selector mode" (selección de modalidad) permite configurar si la selección del renglón o columna será fijo (Fixed) o variable (variable). Si "Selector Mode" es configurado a "Variable", aparecerá una entrada de bloque Idx, la cual permitirá seleccionar de forma variable la columna o renglón que será seleccionada. Sin embargo si "Selector Mode" es seleccionado a "Fixed" la entrada Idx desaparece y la forma de seleccionar el renglón o columna será mediante la colocación de el renglón o columna a seleccionar, en el campo (4) "Elements". Ya sea para el caso "Variable" o "Fixed" la selección es equivalente a usar en MatLAB los siguientes comandos: Para el caso del renglón,

```
y=u(idx,:)
```

y para el caso de columna,

```
y=u(idx,:)
```

considerando a u como una matriz MxN. El valor de idx puede también ser un vector, cuya longitud define el numero de renglones o columnas que serán seleccionados. De tal manera que si idx es un vector de L elementos y el bloque se configuro para seleccionar renglones, la salida del bloque será una matriz de LxN, mientras que si fue configurado para seleccionar columnas la salida será una matriz de dimensión MxL.

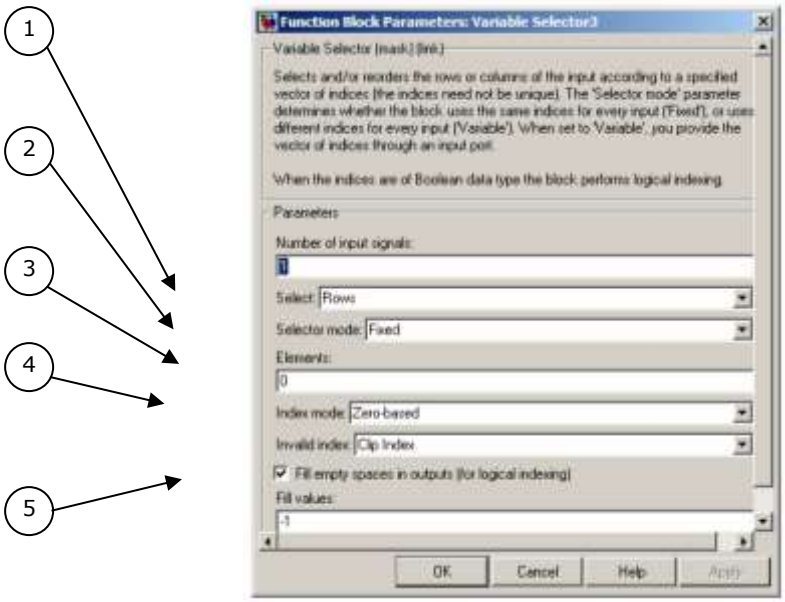
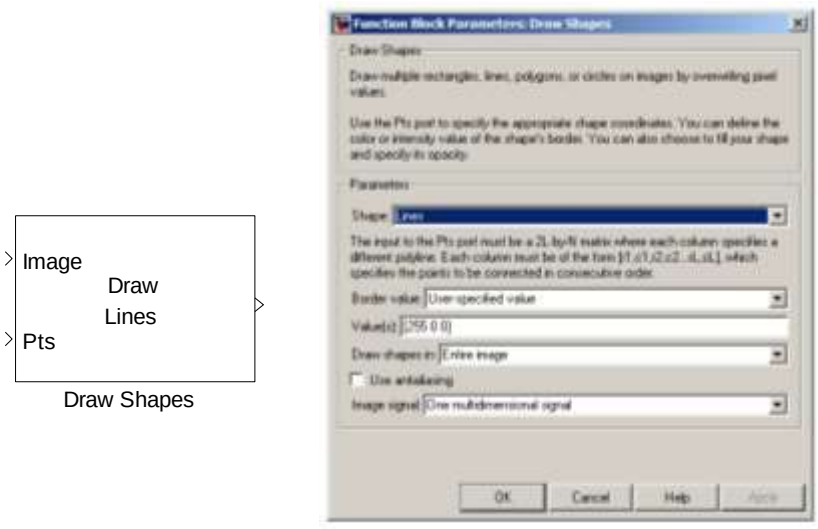


Figura. 8.23 Ventana de configuración del bloque "Variable selector"

Muchas funciones de MatLAB permiten producir resultados ya sea de vectores o matrices considerando como índices iniciales 0 o (0,0). Cuando esto sucede se dice que el vector o matriz es basado en cero (Zero-based), por lo que para seleccionar índices adecuadamente será necesario configurar el campo "Index mode" a "Zero-based".

Draw Shapes (Dibujar Figuras)

Este bloque de la librería de procesamiento de imágenes y video de Simulink, ubicado en la categoría "Texts & Graphics" (Texto y graficas), dibuja primitivas geométricas tales como, rectángulos, círculos, polígonos y líneas. La figura 8.24(a) muestra una representación el bloque.



(a) *(b)*

Figura. 8.24 (a) Representación del bloque "Draw Shapes" y (b) su ventana de configuración.

Dependiendo del tipo de figura a dibujar será los datos y el formato en que se requerirán por el bloque. En el caso de la aplicación que aquí en este capitulo se presenta se dará especial atención al despliegue de líneas, para mayor información sobre otras opciones puede consultarse la ayuda en línea de MatLAB o bien otros capítulos del libro donde se hace extensivo el uso del bloque para otro tipo de figuras.

Para poder usar el bloque en la funcionalidad de marcar líneas, será necesario indicar en la ventana de configuración, en el campo "Shape" (figura) la opción "Lines" (líneas). La figura 8.24(b) ilustra el proceso anterior.

Si la opción de dibujar líneas es elegida el bloque recibirá en la entrada Pts un vector de cuatro elementos [r1 c1 r2 c2] donde r1 y c1 son las coordenadas del renglón y columna del inicio del segmento de línea, mientras que r2 y c2 son las coordenadas del final del segmento.

1.6.1 Ejemplo de detección de líneas usando los bloques de Simulink.

Esta sección termina con una aplicación integradora que permita visualizar en conjunto el empleo de los bloques que implementan la trasformada de Hough así como aquellos que de manera auxiliar posibiliten generar los resultados y su visualización.

Como ejemplo se implementara usando los bloques de Simulink de la librería de procesamiento de imagen y video, la transformada de Hough en tiempo real sobre imágenes captadas de la Webcam. Para lograr esto se genera el programa en Simulink, realizando la conexión de bloques tal y como es mostrado en la figura 8.25.

El programa comienza con la captación de la imagen desde la Webcam mediante la utilización del bloque "From Video Device", el bloque "Color Space Conversión" realiza la conversión de la imagen RGB captada por la Webcam a escala de grises, mientras que por su parte el bloque "Edge Detection" permite extraer los bordes de la imagen mediante la aplicación del filtro sobel a la imagen de intensidad. Todos estos bloques ya han sido tratados con profundidad en varios capítulos anteriores, particularmente en el capítulo 6.

Considerando como entrada la imagen de bordes se utiliza el bloque “Hough Transform”, el cual como fue visto implementa la transformada de Hough, calculando la matriz de registros de acumulación. El bloque se utiliza con las opciones de defecto, con la excepción de que se configura la opción “Output theta and Rho values”.

Una vez calculada la matriz de registros acumuladores Hough, se obtienen los máximos locales mediante el empleo del bloque “Find Local Maxima”. La configuración de este bloque es mostrada en la figura 8.26. Una observación importante es que el bloque, como fue ya visto en párrafos anteriores, entrega como resultado una matriz de 2xNp, con la particularidad de que su índice es basado a cero. La Terminal Count del bloque es conectada a un “Terminador”, con ello el valor de esta Terminal estará conectado y no dará ningún tipo de advertencia mientras se ejecuta[1].

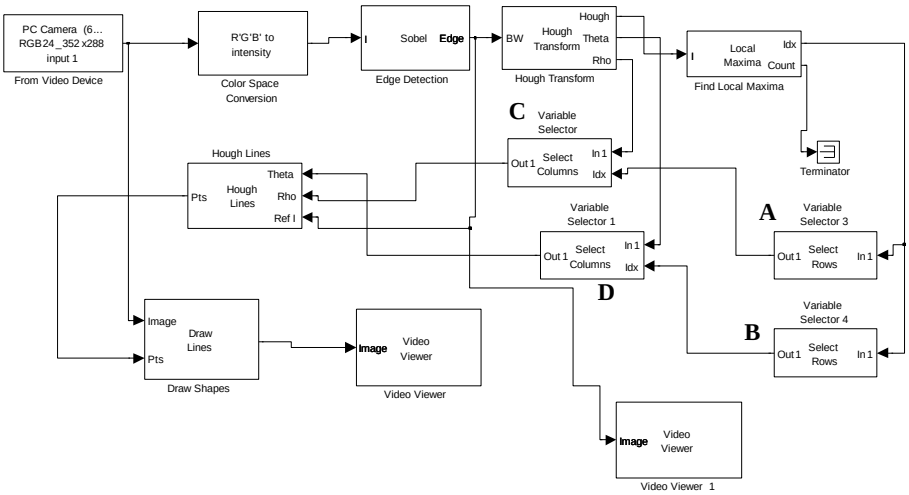


Figura 8.25 Programa en Simulink que implementa la transformada de Hough en tiempo real sobre imágenes captadas de la Webcam.

La salida Hough del bloque “Hough Transform” es de dimensión 2xNp. Donde Np es el número de máximos encontrados, el primer renglón representa al parámetro Rho, mientras que el segundo renglón al parámetro Theta. Es necesario mencionar que esta matriz es basada a índice cero, por lo que el primer renglón estará indexado por cero y el segundo renglón por 1. En consecuencia a lo anterior, se utilizan dos bloques (A y B) del tipo “Variable Selector”, los cuales separan los renglones.

Tal y como fue visto anteriormente, el bloque “Hough Transform” entrega los vectores “Theta” y “Rho”, con la información de cómo estos parámetros fueron divididos linealmente. De tal forma que los índices de la matriz de registros acumuladores corresponderían a los valores de los parámetros con el incremento apropiado. De esta manera si se acopla la salida de los bloques **A** y **B**, los cuales contienen la información de los índices de los registros de valor máximo, a los bloques **C** y **D** que reciben como entrada a los vectores Rho y Theta entregados por el bloque “Hough Transform”, obtendríamos a su salida los valores de Rho y Theta que corresponden a los valores máximos de los parámetros que representan a las rectas identificadas en la imagen. Estos valores son enviados al bloque “Hough Lines”. Este bloque como fue explicado ya, identificara las coordenadas

[1] Siempre que no se conecta una terminal de un bloque Simulink arroja una advertencia, que simplemente indica que no se ha realizado esta operación.

de los segmentos de rectas normales a los vectores definidos por los parámetros Rho y Theta recibidas como entradas.

Figura. 8.26 Ventana de configuración del bloque "Find Local Maxima" utilizado para el ejemplo de implementar la transformada de Hough en tiempo real.

Finalmente el bloque "Hough Lines" calcula las coordenadas del los segmentos correspondientes a los valores paramétricos de Rho y Theta. La matriz de salida de este bloque al ser 4xNL posee las características necesarias para el acoplamiento directo entre este bloque y el bloque "Draw Shapes" donde las líneas son dibujadas sobre la imagen de entrada la cual al desplegarla a través del "Video Viewer" permite observar el resultado del algoritmo completo. La figura 8.27 muestra el resultado de haber ejecutado el programa en Simulink mostrado en 8.25.

1.7 TRANSFORMADA DE HOUGH PARA LA DETECCIÓN DE CÍRCULOS

Las líneas son primitivas geométricas de 2 dimensiones que son completamente descritas mediante la definición de 2 parámetros. Un círculo que también es representado en el espacio bidimensional necesita 3 diferentes parámetros para ser completamente definido, los cuales son el centro especificado por el par ordenado (u,v) y el radio ρ. La figura 8.28 muestra los parámetros y su relación con la primitiva geométrica.

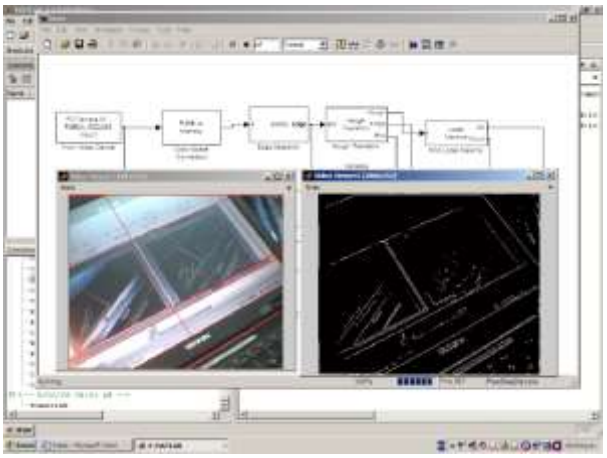


Figura 8.27 Resultado de haber ejecutado el programa en Simulink mostrado en 8.25.

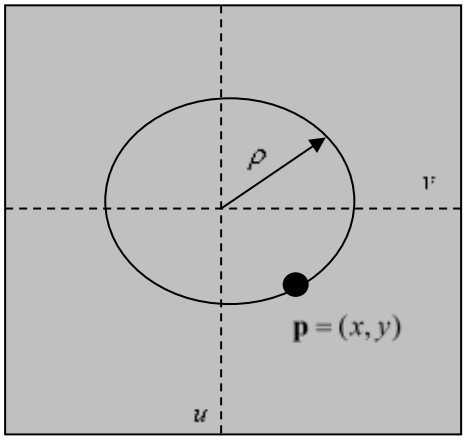


Figura 8.28 Parámetros que definen a un círculo.

Atendiendo ala figura 8.25, un punto $\mathbf{p} = (x, y)$ forma parte del circulo si la condición

$$\rho^2 = (x-u)^2 + (y-v)^2 \qquad (8.12)$$

se cumple.

Considerando lo anterior para implementar la transformada de Hough en la detección de círculos, se necesita un arreglo de registros acumuladores de 3 dimensiones ($\mathbf{MRAcc(}\, u, v, \rho\, \mathbf{)}$).

El algoritmo para detectar los círculos usando la transformada de Hough se basa enteramente en el utilizado para las rectas, con las debidas adaptaciones debidas a la condición impuesta por la ecuación 8.12. El algoritmo 8.2 muestra en detalle los pasos que deberán ser desarrollados.

Detector de círculos por Trasformada de Hough ($I(x, y)$ **)**

1: $0 \rightarrow$ **MRAcc(** u, v, ρ **)**
2: **for** todas las coordenadas de la imagen $I(x, y)$ **do**
3: **if** ($I(x, y)$ es un borde) **then**
4: **for** todos los valores (u, v) **do**

Detector de círculos por Trasformada de Hough ($I(x,y)$)

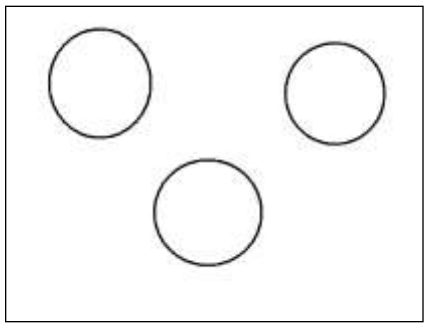
5: $\rho^2 = (x-u)^2 + (y-v)^2$

6: se incrementa **MRAcc(u,v,ρ)**

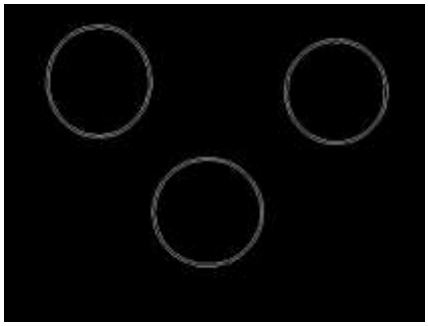
7: Por ultimo se encuentran los registros **MRAcc(u,v,ρ)** cuyos valores sean máximos.

Algoritmo 8.2 Algoritmo para detectar círculos utilizando la Trasformada de Hough.

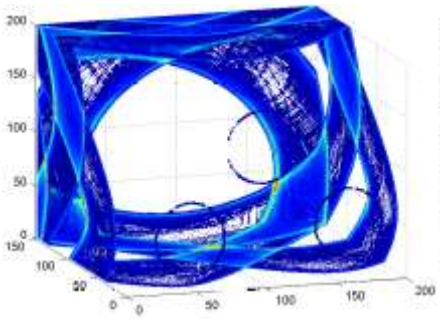
Como puede verse del algoritmo 8.2 los pasos a seguir son similares a los explicados para el caso de líneas. Como primer paso se colocan todas las celdas de la matriz de registros de acumulación a cero (sentencia 1). Después se recorre la imagen binaria (sentencia 3) $I(x,y)$, de tal forma de que cada vez que se encuentra un píxel borde (es decir que su valor sea uno), se obtiene los valores de ρ a partir de la ecuación 8.12, realizando un barrido de los parámetros (u,v) de (1,1) a (m,n) (sentencia 4, donde (m,n) son las dimensiones de la imagen sobre la cual se quieren detectar los círculos). Para cada vector (u,v,ρ) obtenido del barrido, se incrementa el registro de la matriz de acumulación (sentencia 6) en los índices correspondientes a u,v,ρ. De esta forma, una vez recorrido todos los píxeles de la imagen, los registros tridimensionales que se hayan incrementado, obteniendo los máximos locales (sentencia 7) que corresponderán a los valores de u,v,ρ que definirán, a los círculos identificados en la imagen $I(x,y)$. Para poder encontrar los máximos locales del arreglo de registros acumuladores, primero se aplica un umbral, de tal manera que solo los puntos mayores a ese umbral permanezcan, después se realiza una búsqueda en el arreglo encontrando los máximos en el volumen que se encuentren considerando una determinada vecindad volumétrica.



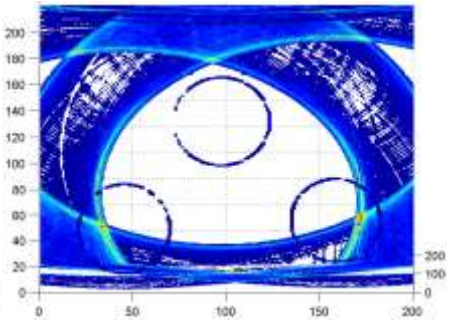
(a)



(b)



(c)



(d)

Figura 8.29 Resultados parciales del algoritmo 8.2. (a) Imagen original, (b) bordes de (a) detectados por el algoritmo de Canny, (c) representación tridimensional del arreglo de registros acumuladores y (d) detección de máximos de los registros.

La figura 8.29 muestra una serie de imágenes que representan los resultados parciales del algoritmo 8.2. La figura 8.29(a) representa la imagen original producida artificialmente para ejemplificar el uso del algoritmo de Hough para la detección de círculos, la figura 8.29(b) muestra los bordes obtenidos por la aplicación del algoritmo de Canny (véase capitulo 6). En 8.29(c) Se muestra el contenido del arreglo de registros acumuladores. En la figura 8.29(d) puede observarse que los puntos mas rojos representan aquellas localidades cuyos valores han sido incrementados significativamente, convirtiéndose en puntos altamente potenciales de los parámetros reales u, v, ρ que representan a los círculos reales de la imagen original $I(x, y)$. Es de notar que las imágenes 8.29(c) y 8.29(d) por rotaciones impuestas al volumen a parecen invertidas en relación a 8.29(a) y 8.29(b).

Una forma mas eficiente de desarrollar el algoritmo 8.2 sin realizar una búsqueda en el espacio tridimensional, es la de localizar en el plano (u,v) el valor máximo para un valor determinado de ρ. Realizando esta adecuación es posible reducir el tiempo de cálculo considerablemente. La figura 8.30 muestra el plano (u,v) del arreglo de registros acumuladores considerando que ρ=30. De la figura puede observarse como los puntos máximos (de color rojo) corresponden a los centros de los círculos detectados en la imagen original representada en la figura 8.29(a).

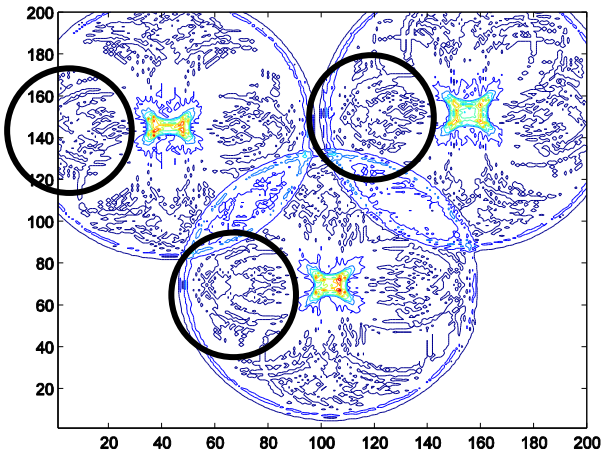


Figura 8.30 Plano (u,v) del arreglo de registros acumuladores considerando que ρ =30.

1.8 LA TRASFORMADA DE HOUGH IMPLEMENTADA EN MATLAB PARA LA DETECCIÓN DE CIRCULOS.

En esta sección se implementa la trasformada de Hough en la detección de círculos utilizando MatLAB como lenguaje de programación. MatLAB no dispone ni de ninguna función, así como tampoco ningún bloque de Simulink que permita mediante la transformada de Hough encontrar figuras circulares en una imagen. Por lo que habrá de programarse utilizando los recursos programáticos de los que dispone MatLAB. El programa 8.3 muestra el código comentado que implementa la trasformada de Hough usada para la detección de círculos.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Programa que implementa la transformada de Hough
% para la detección de círculos
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Erik Cuevas, Daniel Zaldivar, Marco Pérez
% Versión: 08/02/08
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Se obtienen las dimensiones de la imagen binaria BW
%donde los píxeles que tienen el valor de 1 es que son
%parte del borde, mientras que cero, significaría
%que son parte del fondo
[m n]=size(BW);
%Se inicializa el arreglo de registros de acumulación
Mp=zeros(m,n,150);

for r=1:m
    for c=1:n
        if(BW(r,c))
            for re=1:m
                for co=1:n
                     %se obtiene el valor de ph a partir de
8.11
                    ph=round(sqrt((r-re)*(r-re)+(c-co)*(c-
co)));
                    if(ph==0)
                        ph=1;
                    end
                    %Si el valor de ph es menor a 150
                    if(ph<=150)
                      %Se incrementa la celda en uno la celda
                    %correspondiente a los parámetros Ph,u y v
                        Mp(re,co,ph)=Mp(re,co,ph)+1;
                    end
                end
            end
        end
    end
end
```

Programa 8.3 Implementación en MatLAB de la transformada de Hough para la detección de círculos.

El programa 8.3 implementa el algoritmo 8.2 calculando el arreglo de registros acumuladores. De todo lo que se ha visto en este capitulo puede imaginarse que mediante la transformada de Hough se es capaz de detectar no solo líneas y círculos, sino también, elipses o segmentos de círculos, con la única dificultad de poder definir a cada una de estar primitivas geométricas a través de un conjunto de valores paramétricos.