Procesamiento Global empleando la Transformada de Hough

En este apunte vamos a conocer a la transformada de Hough. A diferencia de los métodos de análisis local que ya se han visto, la transformada de Hough considera las relaciones globales entre píxeles de borde permitiendo encontrar ciertos patrones en la imagen como líneas y círculos.

Supongamos que para n puntos de la imagen se desean encontrar aquellos subconjuntos de puntos que caen en líneas rectas. Una posible solución podría ser en primer lugar encontrar todas las líneas determinadas por cada par de puntos y entonces encontrar todos los subconjuntos de puntos que están cerca de cada recta en particular. Este problema así planteado requiere encontrar $n(n-1)/2 \sim n^2$ rectas y realizar $n(n(n-1))/2 \sim n^3$ comparaciones de cada punto a línea. Este método no será viable salvo en casos triviales.

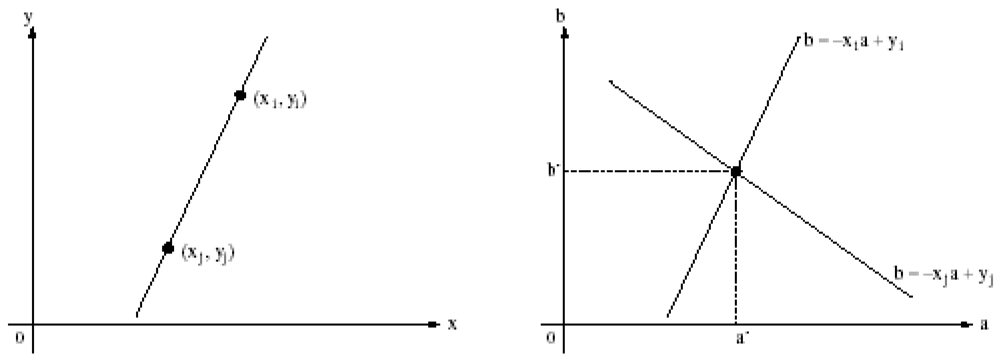


Figura 1: (a) Plano xy. (b) Espacio de parámetros.

Una alternativa al método de búsqueda exhaustiva es la transformada de Hough. Consideramos un punto (x_i,y_i) y la ecuación de la recta, de la forma pendiente y ordenada al origen,

$$y_i = ax_i + b \tag{1}$$

Por el punto (x_i,y_i) pasan infinitas rectas, pero todas satisfacen la ecuación anterior para diferentes valores de ***a*** y ***b***. Sin embargo, escribiendo esta ecuación en la forma

$$b = -x_i a + y_i \tag{2}$$

y considerando el plano **ab** (también denominado espacio de parámetros) da lugar a una única recta para el par (x_i,y_i) constante. Si ahora consideramos un segundo punto (x_j,y_j), también va a tener su recta asociada en el espacio de parámetros. Estas dos rectas se cortarán en el espacio de parámetros en un punto ***(a',b')***, donde ***a'*** es la pendiente y ***b'*** la ordenada al origen de la recta que contiene a los puntos (x_i,y_i) y (x_j,y_j) en el plano **xy**, como se puede ver en la figura 1. De hecho, todos los puntos de esa recta en el plano **xy** darán lugar a rectas diferentes en el espacio de parámetros que se cortan en un único punto ***(a',b')***.

El atractivo de la transformada de Hough proviene de subdividir el espacio de párametros en celdas acumuladoras, como se puede ver en la figura 2, donde (a_{min},a_{max}) y (b_{min},b_{max}) son los rangos esperados para la pendiente y la ordenada al origen. La celda de coordenadas ***(i,j)*** con un valor de acumulador ***A(i,j)*** corresponde al cuadrado asociado con las coordenadas ***(a_i,b_j)*** del espacio de parámetros. Inicialmente se ponen todos los a-cumuladores a cero. Entonces para cada punto ***(x_k,y_k)*** de la imagen, permitimos que el parámetro a pueda tomar cualquier valor de entre los ***a_i*** permitidos y calculamos ***b*** usando la ecuación 2. Los valores resultantes para el parámetro ***b*** se redondean hasta los ***b_j*** permitidos. Si para un valor ***a_p*** resultó un valor ***b_q*** se tiene que

$$A(p,q) = A(p,q) + 1 \tag{3}$$

Al final, un valor de ***M*** en el acumulador ***A(i,j)*** significa que ***M*** puntos del plano **xy** caen sobre la recta ***$y=a_ix+b_j$***. La precisión en la colinealidad de estos puntos depende del número de celdas del espacio de parámetros.

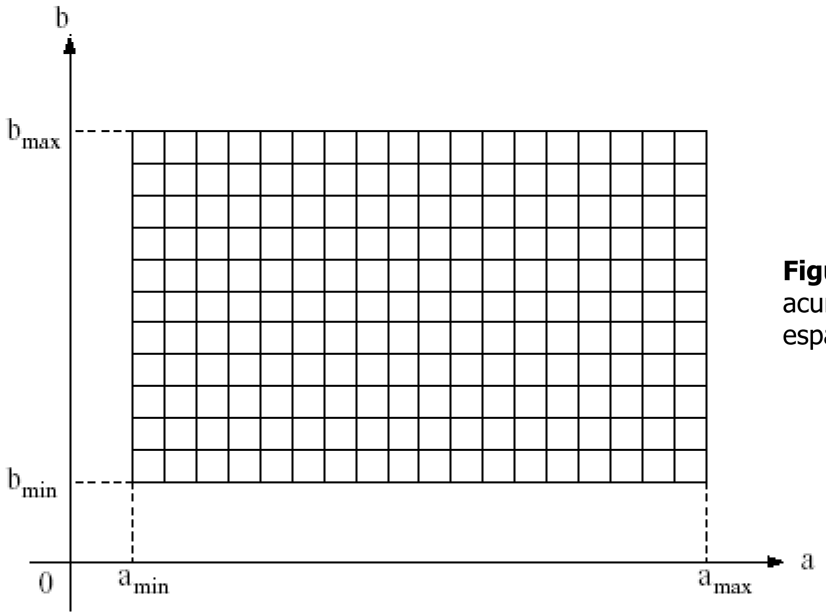


Figura 2: Celdas acumuladoras en el espacio de parámetros.

Si subdividimos el eje a en ***K*** celdas, para cada punto ***(x_k,y_k)***, obtenemos ***K*** valores de ***b*** correspondientes a los ***K*** posibles valores de ***A***. Si la imagen tiene ***n*** puntos, la carga computacional es del orden de ***nK***. La transformada de Hough es lineal en ***n***, y el producto ***nK*** es mucho menor que si hubiéramos empleado una búsqueda exhaustiva, a menos que ***K*** sea del orden o exceda a ***n***.

Un problema que surge al emplear la ecuación de la recta ***y=ax+b*** para representar una línea es que tanto la pendiente como la ordenada al origen pueden llegar a valer infinito, según la línea se hace vertical. Una forma de solventar este problema consiste en utilizar la representación normal de la recta

$$x * \cos\theta + y * sen\,\theta = \rho \tag{4}$$

En la figura 3 se puede ver el significado de los nuevos parámetros ***(ρ,θ)***. El uso de esta representación para construir la tabla de acumuladores es similar al método explicado para las rectas en la forma pendiente y ordenada al origen. A cada punto del plano **xy** corresponde ahora una sinusoide en el plano $\rho\theta$ en lugar de una recta. Al igual que antes, ***M*** puntos colineales a la recta ***x cos θ_j + y sen θ_j = ρ_i*** darán lugar a ***M*** sinusoides que se cortan en el punto ***(ρ_i,θ_j)*** en el espacio de parámetros. Incrementando θ y calculando ρ, obtendremos ***M*** entradas en el acumulador ***A(i,j)*** correspondiente al par ***(ρ_i,θ_j)***.

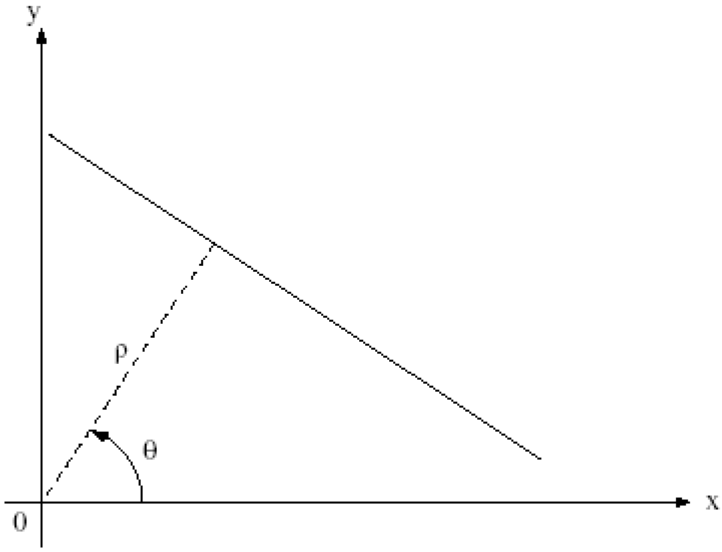


Figura 3: Representación normal de una recta

En la figura 4 podemos ver la tabla de acumuladores del espacio de parámetros en este caso. El rango para el ángulo θ es ±90º, medido con respecto al eje de abscisas. Se permiten valores negativos de ρ para rectas por detrás del origen de coordenadas del plano **xy**. Por ejemplo, una recta horizontal corresponde a un valor de ***θ=0º*** y un valor de ρ igual a la ordenada al origen, mientras que una recta vertical corresponde a un valor de ***θ=90º*** y un valor de ρ igual a la abscisa en el origen.

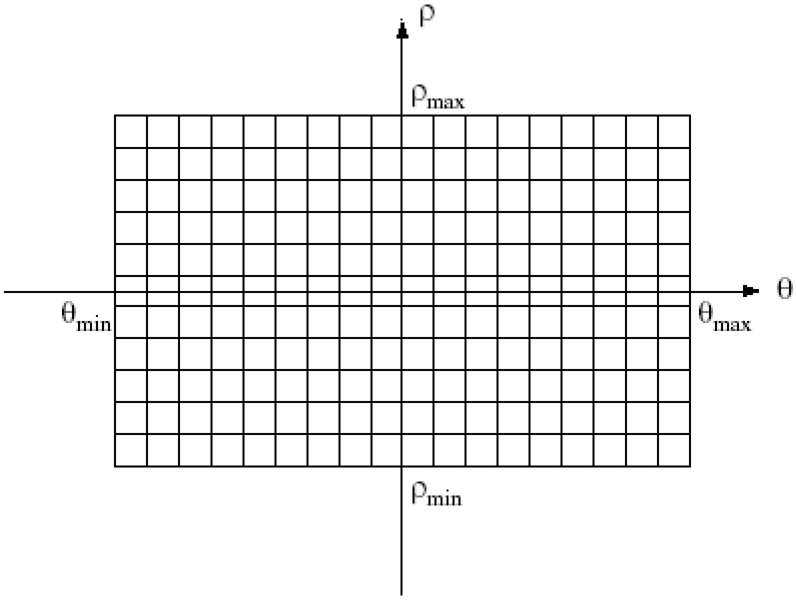
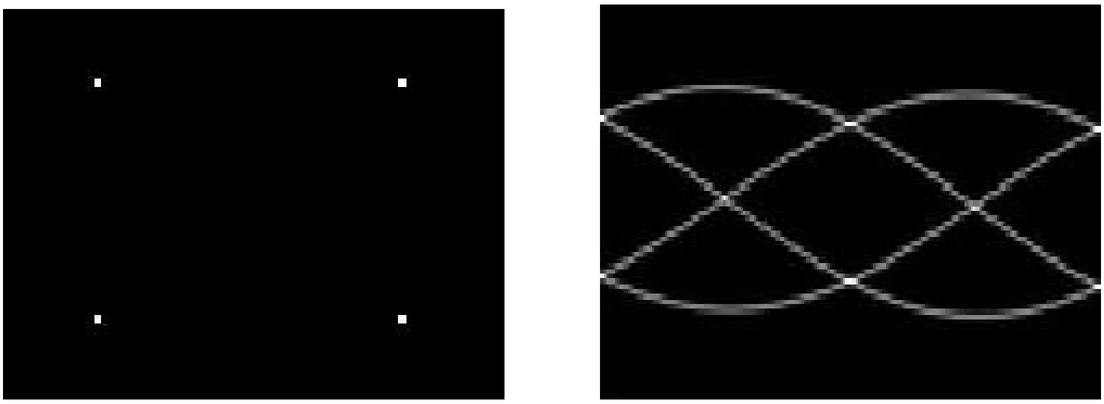


Figura 4: Celdas acumuladoras en el espacio ρθ.

En la figura 5 se ilustra con un ejemplo cómo funciona la transformada de Hough. La imagen está deformada por cuatro puntos correspondientes a las esquinas de un cuadrado. Estos cuatro puntos dan lugar a cuatro sinusoides en el espacio ρθ. Las cuatro sinusoides se cortan en seis puntos (en la figura 2 (b) aparecen ocho puntos, pero hay que recordar que los dos puntos para θ=90º son los mismos que los puntos para θ=-90º, por lo que de ocho son seis distintos), correspondientes a las seis rectas posibles que pasan por los cuatro puntos del plano **xy**, que son a saber, los cuatro lados del cuadrado y las dos diagonales.



(a) (b)

Figura 5: (a) Imagen con cuatro puntos de borde. (b) Transformada de Hough mostrando seis puntos de corte correspondientes a las seis rectas que pueden pasar por los cuatro puntos.

Aunque hemos hecho un análisis para el caso de rectas, la transformada de Hough también es aplicable a cualquier función de la forma

$$g(v,c)=0 \tag{5}$$

donde **v** es un vector de coordenadas y **c** es un vector de coeficientes. Por ejemplo, puntos que caen en el círculo

$$(x-c_1)^2+(y+c_2)^2=c_3^2 \tag{6}$$

se pueden detectar empleando también la transformada de Hough. En este caso tenemos tres parámetros ***(c₁, c₂, c₃)***, lo que dará lugar a un espacio de parámetros de tres dimensiones, con celdas con forma de cubo y acumuladores de la forma ***A(i,j,k)***. El procedimiento en este caso es para cada punto del plano **xy**, para cada c_1 y para cada c_2, calcular el valor de c_3 y actualizar el acumulador correspondiente a (c_1, c_2, c_3). La complejidad de la transformada de Hough es claramente dependiente del tamaño del espacio de parámetros.

$$(x-c_1)^2+(y+c_2)^2=c_3^2 \tag{6}$$

En el apéndice se puede encontrar un algoritmo hecho en un script de Matlab que realiza la transformación de Hough.

Apéndice

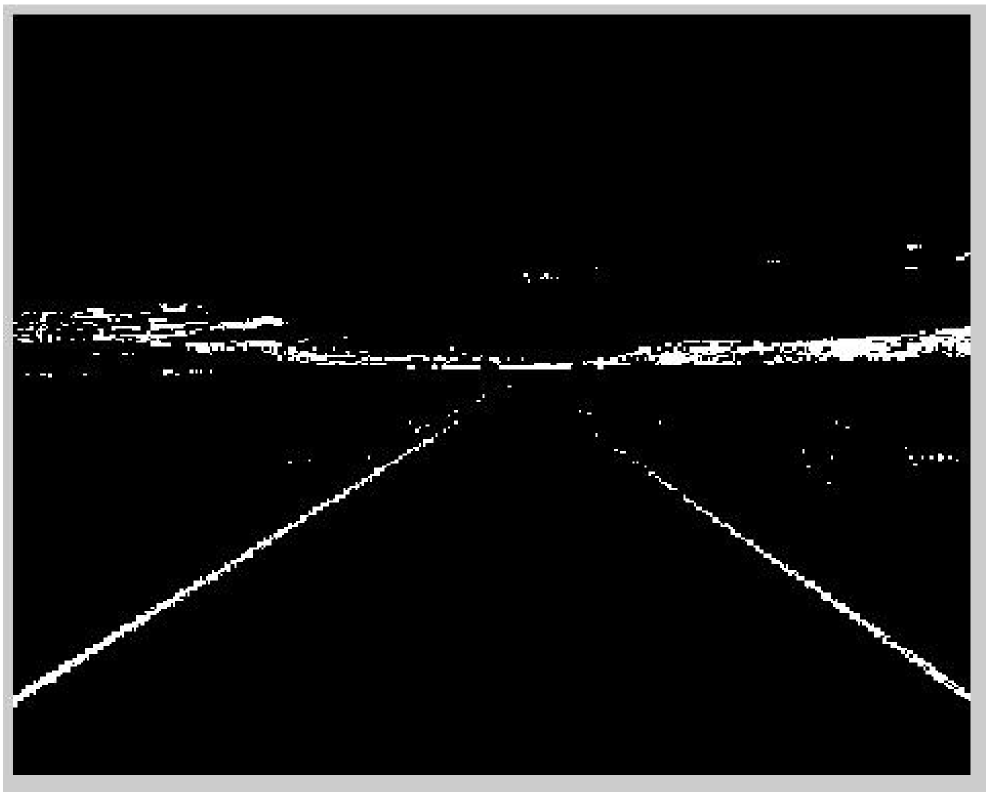
```
% TRANSFORMADA DE HOUGH PARA DETECCION DE RECTAS

% Universidad Nacional de Quilmes
% Ingeniería en Automatización y Control Industrial
% Cátedra: Visión Artificial - Agosto de 2005


A=double(imread('ruta2.bmp','bmp'));
B=A>240;                                %Binarizacion de la imagen

IMAGEN=B;
[m,n]=size(IMAGEN);
figure(1);imshow(IMAGEN);title('IMAGEN BINARIZADA');

%%%%%%%%%%%%%%%%%%%%%%%%%%%PARÁMETROS%%%%%%%%%%%%%%%%%%%%%%%%%

NTITA=500;                                 %cantidad de divisiones del  eje tita
NRO=500;                                   %cantidad de divisiones del eje ro

TITAmax=pi/2;                              %variacion de  la variable tita
TITAmin=-pi/2;
TITAdelta=(TITAmax-TITAmin)/NTITA;         %variacion minima de tita

ROmax=sqrt(m^2+ n^2);                      %variacion de ro
ROmin=0;
ROdelta=(ROmax-ROmin)/NRO;                 %minima variacion de ro

%%%%%%%%%%%%%%%%%%%%%%%%%CAMBIO DE COORDENADAS DE LAS RECTAS%%%%%%%%%%%%%%%%

ET=zeros(NTITA,NRO);                      %ET: Espacio Transformado de (ro,tita)
[X,Y]=find(IMAGEN);       %Retorna las coordenas (x,y) de los ptos de la imagen

if length(Y)>1
   for i=1:length(Y)                            %Recorre la eje Y
      for j=1:NTITA                             %Recorre el eje TITA
                                                %%%CONVERSION DE COORDENADAS%%%
         TITAc=TITAmin + TITAdelta*(j-0.5);     %Desplazamiento por casilleros
         ROc=X(i)*cos(TITAc)+ Y(i)*sin(TITAc);  %Armado de senos a partir de
                                                %(X,Y)
         ROg=floor((ROc/ROmax)*(NRO-1)+0.5)+1;  %Redondeado de RO desplazando
                                                %cada casillero
         if (ROg>=1)&(ROg<=NRO)                 %Si esta dentro de la grilla
                                                %escribo el valor en ET
         ET(j,ROg)=ET(j,ROg)+1;                 %LLeno las grila con los
                                                %puntos, Obtencion de Senos
         end                                    %en funcion A TRANSFORMACION%%%
      end
   end
end

%%%%%%%%%%%%%%%%%%%%%%%%BUSQUEDA DE LA RECTA MAS POPULAR%%%%%%%%%%%%%%%%%%%

[TITA,RO]=find(ET==max(ET(:)));             %Busqueda de la recta mas popular
TITAr=TITAmin + TITAdelta*(TITA-0.5);   %nuevos parametros para la reconstrucion
ROr=ROmin + ROdelta*(RO-0.5);

IMAGEN2=zeros(m,n,3);
IMAGEN2(:,:,1)=IMAGEN*255;
IMAGEN2(:,:,2)=IMAGEN*255;
IMAGEN2(:,:,3)=IMAGEN*255;

for xi=1:m
   yi=floor((ROr-xi*cos(TITAr))/sin(TITAr)+0.5); %Reconstruccion de de la recta
   if (yi>=1) &(yi<=n)
```

```
      IMAGEN2(xi,yi,1)=255;                        %Imagen reconstruida
   end
end

figure(2);
imshow(uint8(IMAGEN2));title('Original y Reconstruida')

%%%%%%%%%%%%%%%%%%%%%ELIMINACION DE LA PRIMERA RECTA ENCONTRADA%%%%%%%%%%%%%

 for i=1:length(Y)                                 %Recorre la eje Y
    for j=1:NTITA                                  %Recorre el eje TITA
        ROc=0;
        ROg=floor((ROc/ROmax)*(NRO-1)+0.5)+1;      %Redondeado de RO desplazando
                                                    %cada casillero
        if (ROg>=1)&(ROg<=NRO)                     %Si esta dentro de la grilla
                                                    %escribo el valor en ET
        ET(j,ROg)=ET(j,ROg)+1;                    %LLeno las grila con los puntos
        end
    end
 end


%%%%%%%%%%%%%%%%%%%%%%BUSQUEDA DE LA SEGUNDA RECTA MAS POPULAR%%%%%%%%%%%%%
[TITA,RO]=find(ET==max(ET(:)));                %Busqueda de la recta mas popular
TITAr=TITAmin + TITAdelta*(TITA-0.5);  %nuevos parametros para la reconstrucion
ROr=ROmin + ROdelta*(RO-0.5);

for xi=1:m
   yi=floor((ROr-xi*cos(TITAr))/sin(TITAr)+0.5); %Reconstruccion de de la recta
   if (yi>=1) &(yi<=n)                             %Imagen reconstruida
      IMAGEN2(xi,yi,2)=255;
   end

end
figure(3);
imshow(uint8(IMAGEN2));title('Original y Reconstruida')
```

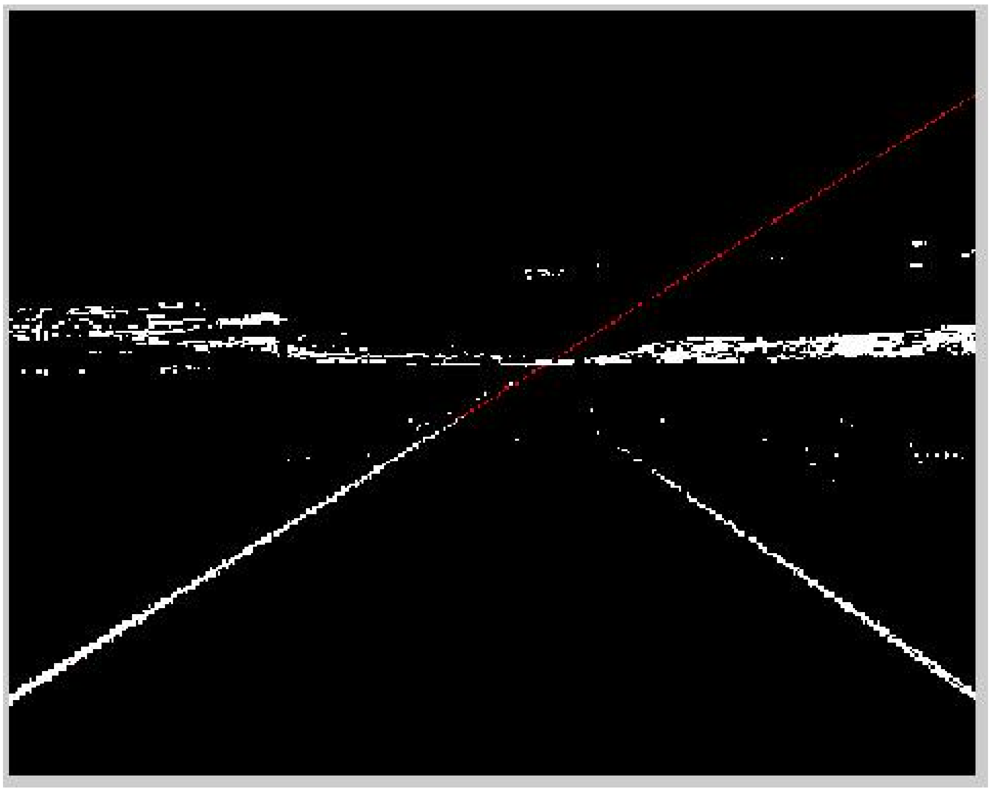
Luego, las imágenes obtenidas con este algoritmo fueron las siguientes:

La imagen original sobre la que se aplica el algoritmo es la que sigue:

El resultado de binarizar la imagen original con un umbral de 240 es el siguiente:



La recta más popular se puede ver en la siguiente figura:



Por último, las dos rectas más populares se muestran en la figura que sigue: