Implementación de la transformada Hough para encontrar el punto de fuga en una imagen

Walter Alejandro Moreno Ramírez
Departamento de Estudios Multidisciplinarios
Universidad de Guanajuato
Yuriria, Guanajuato
Correo: wa.morenoramirez@ugto.mx

***Abstract*—This article describes how to implement the Hough transform, which are the most important applications and their advantages and disadvantages. Also, it describes how to find the vanishing point in an image.**

***Index Terms*—Pixel, píxeles, convolución, canny, smoothing, suavizado, gradiente, segmentación ,primera derivada, función, C++, OpenCV, ventana, máscara, vecindario.**

I. Introducción

La transformada Hough es una técnica para la detección de figuras en imágenes digitales. Las figuras que se pueden detectar son círculos, cuadrados, elipses y lineas rectas o curvas. Una característica importante que debe cumplir toda figura para poder ser detectada por esta técnica es que puedan ser expresadas matemáticamente mediante una ecuación.

La transformada de Hough fue propuesta y patentada Paul Hough en 1962, pero fue hasta 1972 que Duda R. O. y P.E. Hart crearon la transformada de Hough tal como se conoce y utiliza actualmente quienes la llamaron "Use of the Hough Transformation to Detect Lines and Curves in Pictures". Pero fue Dana H. Ballard quien la popularizó en la comunidad de visión por computadora publicando un artículo en 1981, llamado "Generalizing the Hough Transform to Detect Arbitrary Shapes".

Para poder llevar a cabo la detección de lineas y curvas utilizando la transformada de Hough, como paso anterior es necesario obtener los bordes de los objetos, que es el resultado de la implementación del algoritmo de Canny para la detección de bordes o aplicando cualquier gradiente para obtener bordes como pueden ser: Prewitt, Sobel, Roberts, etc.
Esto es necesario para tener una menor cantidad de elementos no requeridos y que pudieran afectar en la detección de lineas o curvas y, también pudieran provocar un tiempo de ejecución muy alto.

Para esta práctica, se utilizó la transformada de Hough para encontrar el punto de fuga de una imagen. Un elemento importante en una imagen digital es el punto de fuga ya que nos ayuda a dar profundidad a la imagen a través de lineas reales o imaginarias que convergen en un punto infinito. Si observamos una carretera o unas vías de tren, a medida que se alejan parece acercarse hasta converger en un mismo punto, es ese preciso momento donde observamos como las líneas parecen curzarse es lo que se llama punto de fuga en una imagen y utilizando la transformada de Hough lo obtendremos.

II. Metodología

El espacio de una imagen digital a cada pixel le corresponde una coordenada de la forma (x, y), semejante al sistema de coordenadas rectangular o cartesiano. Esto nos facilita la detección de figuras y, en este caso, el objetvio es detectar las rectas presentes en una imagen. Para ello, la ecuación de la recta, que se muestra en la Ecuación (1), servirá para cumplir este cometido.

$$y = m * x + b \tag{1}$$

Y se puede graficar para cada par ordenado (x, y) en la imagen. La transformada Hough se basa en considerar las características de una recta en término de sus parámetros (m, b), y no como sus puntos de la imagen. Basandose en lo anterior, una recta que tiene la forma (1), puede representarse en el espacio paramétrico como un punto. Sin embargo, cuando se tienen rectas verticales, los parámetros de la recta se indefinen. Para solucionar esto, se utilizan los parámetros que describen una recta en coordenadas polares, denotado por (ρ, θ).
El parámetro ρ representa la distancia del origen de coordenadas y el punto (x, y) y θ es el ángulo del vector director de la recta perpendicular a la recta original y que pasa por el origen de coordenadas.
Utilizando esta parametrización, la ecuación de la recta, mostrada en la Ecuación (1), se puede escribir como se muestra en la Ecuación (2).

$$\rho = x\cos(\theta) + y\sin(\theta) \tag{2}$$

Entonces, es posible asociar a cada rectar un par (ρ, θ) que es único si $\theta\epsilon[0, \pi)$.
Sin embargo, por un sólo punto con coordenadas (x, y) pasan un número infinito de rectas. Para poder discretizar este número de rectas en coordenadas (ρ, θ), lo mejor es considerar el intervalo $[0, \pi)$, lo que nos da como resultado una curva sinusoidal en el espacio $\rho - \theta$ para cada familia

de rectas que pasan por un punto (x, y). Entonces para cada punto en el plano cartesiano, tendremos una curva sinusoidal en el plano polar, con la posbilidad de que varias curvas sinusoidales se crucen en un mismo punto, indicando que son parte de una recta en el plano cartesiano.
Con esto logramos trasladar el problema de encontrar una secuencia de puntos o píxeles que nos indiquen una recta a buscar un números de intersecciones de varias curvas sinusoidales.

El valor máximo de intersecciones entre curvas no representará la recta dominante de la imagen y dicho valor puede servir como parámetro para encontrar las demás rectas.

Se sigue el pseudocódigo de la Figura 1. para poder realizar la conversión de espacio (x, y) a (ρ, θ).

```
para i=0 hasta rows
para j=0 hasta cols
    si pixel(i,j) == 255
        para theta=0 hasta 180
            rho = i*cos(theta) + j*sin(theta)
            si rho >= 0
                collector[p][theta]++
                si collector[p][theta] > mayor
                    mayor = collector[p][theta]
                fin si
            fin si
        fin para
    fin si
fin para
fin para
```

Figura 1: Pseudocódigo para obtener las curvas sinusoidales para cada punto (x, y) y el valor máximo correspondiente a la recta dominante.

Se recorre la imagen de entrada completamente lo que permite poder obtener cada pixel para analizarlo, si dicho pixel es un borde, se procede a calcular una ρ por cada θ que está dentro del intervalo $[0, \pi)$, con cada par ordenado se incrementa en uno el acumulador, lo que nos dará una curva sinusoidal como salida, ya que el acumulador puede mostrase como una imagen de salida pero se mostrará en la sección de resultados más adelante.

Una vez se obtuvo la recta dominante, una manera para obtener otras rectas de la imagen es umbralizar el acumulador utilizando un porcentaje del valor de la recta dominante que en el pseudocódigo se nombra como "mayor" o utilizando un valor arbitrario, como se muestra en la Figura 2.

```
umbral = 100
para i=0 hasta rowsCollector
    para j=0 hasta 180
        si collector[i][j] >= umbral
            collector[i][j] = 255;
        sino
            collector[i][j] = 0;
        fin si
    fin para
fin para
```

Figura 2: Pseudocódigo para umbralizar el acumulador y obtener las rectas dominantes de la imagen de entrada.

De esta manera, todos los valores que sean mayores o iguales al umbral serán las rectas que se mostrarán al final, los que no cumplan la condición se mandan a cero directamente.

Hasta el momento se tiene un acumulador sólo con los puntos (ρ, θ) pertenecientes a las rectas demoninanste, el siguiente procedimiento es volver del espacio polar al espacio cartesiano, para ello es necesario despejar la variable y de la Ecuación (2), este resultado se muestra en la Ecuación (3).

$$y = \frac{\rho - x\cos(\theta)}{\sin(\theta)} \tag{3}$$

Y, de acuerdo al pseudocódigo de la Figura 3, se recorre todo el acumulador buscando los valores que sean iguales a 255, lo que indica que son máximos y rectas dominantes.

```
para ro=0 hasta rowsCollector
para theta=0 grados
    si collector[ro][theta] == 255
        para x=0 hasta x<rowsImage
            si  sin(theta) > 0
                y = (ro - x*cos(theta))/sin(theta)

                si y >= 0 && y < cols
                    image[x][y] = 255
            }
    }
}
```

Figura 3: Pseudocódigo para pasar de coordenadas polares a coordenadas cartesianas con los puntos máximos.

Al momento de despejar y de (2), se obtiene la ecuación de la recta, por lo tanto tendremos que iterar con una sola variable, tal como se hizo para encontrar las curvas sinusoidales, para poder encontrar el valor de la variable dependiente.

Algo que se debe verificar es que la evaluación de $\sin(\theta)$ no sea cero, ya que se encuentra en el denominador y puede dar un resultado no definido para ρ, esto se arregla incluyendo una condición para que sólo acepte valores positivos y, de igual manera, cuidando que los valores de ρ no se salgan de las dimensiones de la imagen. Si ambas condiciones se cumplen, la imagen en el punto (x, y) recien calculado, se pone en alto o 255, lo que nos da como resultado una recta para cada punto (ρ, θ).

III. Resultados

Para la presente práctica se utilizó una fotografía de un pasillo del Departamento de Estudios Multisciplinarios, mas especificamente el pasillo que conduce al laboratorio. Dicha foto se muestra en la Figura 4.

Figura 4: Imagen de prueba para probar la implementación de la transformada de Hough.

Debido a que, para aplicar la transformada de Hough, es necesario trabajar con los bordes de una imagen, esto se obtiene de la implementación del algoritmo de Canny y se muestra en la Figura 5.

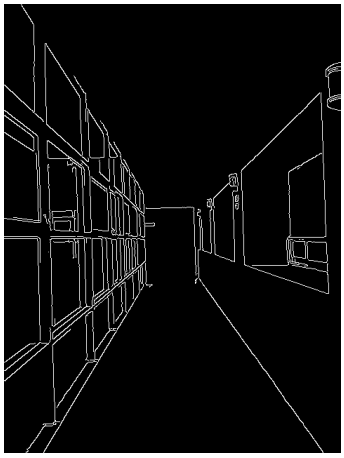

Figura 5: Bordes de la imagen de prueba.

Se puede visualizar el acumulador que contiene todas las curvas sinusoidales correspondientes a cada punto de la imagen con los bordes, esto se muestra en la Figura 6. Las imagenes resultantes de los acumuladores son sorprendetes y muy llamativas.

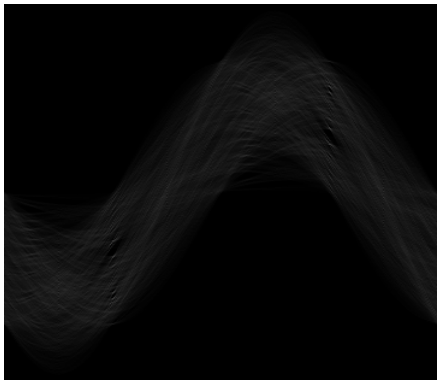

Figura 6: Acumulador correspondiente a los bordes de la imagen de prueba.

La imagen obtenida aplicando la transformada de Hough a la imagen con los bordes se mustra en la Figura 7. Se utilizó un umbral de 100 para encontrar las lineas rectas dominantes.

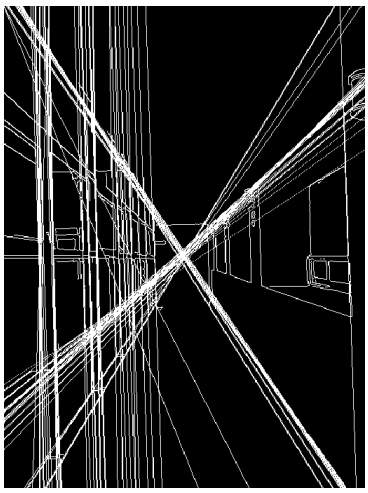

Figura 7: Lineas principales detectadas con la transformada Hough y punto de fuga.

De la Figura 7. podemos apreciar como la mayoría de lineas convergen hacia un punto central, lo que se conoce como punto de fuga de una imagen.

IV. Conclusiones

La taransformada de Hough puede aplicarse para detectar más figuras, no únicamente lineas, pero ya que se detectaron las lineas se pueden encontrar figuras que estén formadas por dichas lineas como pueden ser: cuadrados, triángulos, trapecios, etc. Sin embargo, también se pueden detectar curvas, elipses y círculos, para ello es necesario utilizar sus respectivas ecuaciones.

Una aplicación simple que se le puede dar es detectar la orientación de una pieza u objeto dentro de una barra transportadora en una linea de production, para que un brazo robótico pueda re-orientar su mano y poder tomar la pieza.