Programación 2009

Trabajo adicional

Nos proponemos aquí implementar unas variantes del método de Hough para la detección de lineas rectas dentro de un arreglo de puntos binarios, típicamente una imagen de contornos binarizada. Este algoritmo es un algoritmo estándar para, de manera muy general, detectar formas parametrizadas (lineas, círculos, elipses...) en procesamiento de imágenes.

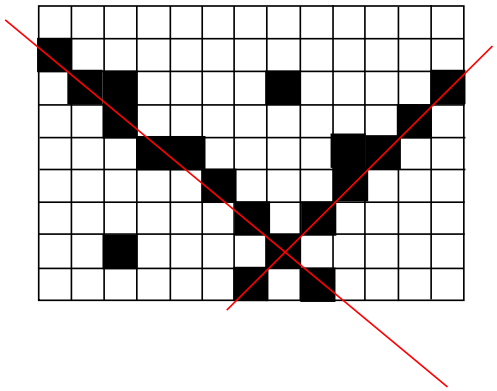

Figure 1: El método de Hough puede servir para detectar lineas rectas en una imagen binaria.

La idea es simple, y se puede explicar a través de esa figura :

1. Los puntos negros de la imagen viven en un espacio (x, y). Ahora, podemos, por ejemplo, usar para representar las lineas rectas una representación (θ, c) tal que su ecuación sea $x\cos\theta + y\sin\theta + c = 0$ (observar que es una forma única si imponemos $0 \leq \theta < \pi$).

2. Discretizar el espacio (θ, c). Disponemos entonces de un arreglo L representando como grilla los (θ_i, c_i) posibles, con cierta "resolución".

3. Hacer **votar** cada de los puntos negros (i, j) del arreglo inicial (la imagen), para todas las lineas rectas que pasan por el, en el espacio de los parámetros de lineas (θ, c). Por ejemplo, en el arreglo L, todos los "puntos" (θ_k, c_k) tales que $|i\cos\theta_k + j\sin\theta_k + c_k| < \delta$ (es decir, sobre una sinusoide) recibirían un voto del punto (i, j). El arreglo L sirve entonces de **acumulador de votos**.

4. Recoger los **maxima locales** dentro de L, con la condición que satisfacen cierto umbral de cantidades de votos.

Les pido implementar este algoritmo para imágenes de nivel de gris en un programa/una librería de clases (sin patrones, para simplificar) que contendrá lo siguiente:

- una clase Image para almacenar los datos de una imagen, es decir un arreglo doble de niveles de gris (usar enteros con signos porque consideraremos también diferencias de imagenes), con métodos simples de acceso a los valores de cada pixel en (i, j), y los operadores que le parecen adecuados;
- dentro de esta clase Image, métodos para cargar/salvar la imagen en formato pgm; el formato pgm es un formato simple de encodificación de imágenes; tiene diferentes variantes (P1, P2, P3, ...), y nosotros consideraremos el formato P2, en el cual los datos vienen escritos en modo texto, después de un encabezado sencillo. Este formato está descrito por ejemplo en

  `http://netpbm.sourceforge.net/doc/pgm.html`
- una clase Gradient implementando un operador cuyo método principal, calcula el modulo (es decir la longitud) del vector gradiente de la imagen, que guarda en otra imagen; pueden evaluar el gradiente estimando las derivadas con diferencias finitas; se podrá grabar en un archivo pgm el resultado;
- una clase HoughLine que implemente el algoritmo de detección de lineas por transformada de Hough como descrito en la pagina anterior; el resultado de la detección estará dibujado en una imagen pgm superponiendo la imagen original con las lineas detectadas. No se necesita interfaz gráfica.

Unas mejoras que pueden al algoritmo para obtener buenos resultados :

- usar el gradiente para no hacer votar todos los (θ, c) sino sólo las lineas rectas más en adecuación con el gradiente (es decir las lineas cerca de ser ortogonales al gradiente);
- en lugar de agregar un voto en (θ, c), intentar agregar votos en una vecindad de (θ, c), con la cantidad de votos atribuidos distribuidos como una Gausiana de promedio (θ, c);

Finalmente explicar brevemente (sin implementarlo) como se podría aplicar el mismo algoritmo para detectar círculos en una imagen.