Transformada de Hough Generalizada

Tomás Cerdá, Marcelo Lynch

Análisis y Tratamiento de Imágenes
Instituto Tecnológico de Buenos Aires

September 9, 2020

Contenidos

Transformada clásica

- Nos permite encontrar curvas solo si están dadas por ecuaciones paramétricas $f(\mathbf{x}, \mathbf{a}) = 0$
- Costo computacional elevado: ante m parámetros con M valores posibles y con p píxeles de borde en la imagen
  - Algoritmo general / naïve: $O(pM^m)$
  - Un parámetro puede determinarse de la ecuación $f(\mathbf{x}, \mathbf{a}) = 0$, volviéndolo $O(pM^{m-1})$
  - Puede reducirse usando información direccional a $O(pM^{m-2})$

Algoritmo general

1. Inicializar un arreglo de votación $A(\mathbf{a})$
2. Para cada píxel de borde $\mathbf{x}$ y para cada $\mathbf{a}$ en el espacio de parametros discretizado, si $f(\mathbf{x}, \mathbf{a}) < \epsilon$, hacer $A(\mathbf{a}) = A(\mathbf{a}) + 1$
3. Buscar máximos locales en A, que determinan los parámetros de las curvas presentes en la imagen

Recorro todo el espacio de parámetros para cada pixel de borde: $O(pM^m)$.

Información extra

Si se están buscando curvas dadas por una ecuación específica, podemos utilizarla para reducir la exploración del espacio de parámetros.
Ejemplo: supongamos que queremos detectar elipses. Tenemos:

$$\frac{x - x_0}{a^2} + \frac{y - y_0}{b^2} = 1$$

Los parámetros son x_0, y_0, a, b.

Utilizando información direccional

Si llamamos $X = x - x_0$, $Y = y - y_0$

$$\frac{X^2}{a^2} + \frac{Y^2}{b^2} = 1$$

Derivando respecto de X vemos que también se debe cumplir:

$$\frac{2X}{a^2} + \frac{2Y}{b^2}\frac{dY}{dX} = 0$$

La clave: ¡podemos obtener $\frac{dY}{dX}$ aplicando operadores direccionales a la imagen!

Manipulando la ecuación de la elipse

A partir de las expresiones anteriores podemos despejar

$$x_0 = x \pm \frac{a^2}{\sqrt{1 + \frac{b^2}{a^2\xi^2}}}$$
$$y_0 = y \pm \frac{b^2}{\sqrt{1 + \frac{a^2}{b^2\xi^2}}}$$

donde $\xi = \frac{dX}{dY}$.
Con esto podemos iterar únicamente por los parámetros a, b, mientras que x_0, y_0 quedan determinados según a, b y el pixel de borde (x, y) que se está considerando. Esto es, el orden queda $O(pM^{m-2})$.

Algoritmo mejorado (caso elipse)

1. Inicializar un arreglo de votación $A(a, b, x_0, y_0)$
2. Para cada píxel de borde (x, y): recorrer cada par (a, b) en el espacio de parametros discretizado, determinar los x_0 e y_0 que corresponden a ese píxel de borde, y hacer $A(a, b, x_0, y_0) = A(a, b, x_0, y_0) + 1$
3. Buscar máximos locales en A, que determinan los parámetros de las curvas presentes en la imagen

Recorro solamente el subespacio de parámetros a, b por cada pixel de borde: $O(pM^{m-2})$

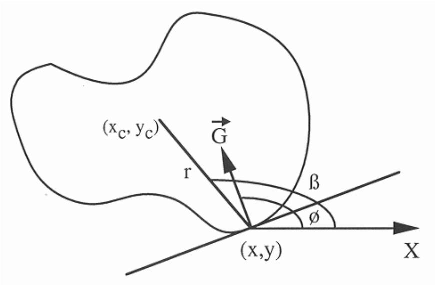
La transformada generalizada

1981: Ballard publica *Generalizing the Hough transform to detect arbitrary shapes*.

- La transformada generalizada nos permite encontrar curvas no analíticas, es decir, que no tienen una ecuación paramétrica
- También puede ser útil para curvas analíticas que se describen con demasiados parámetros (tiene una complejidad "fija")
- Utiliza información direccional
- Altamente paralelizable (igual que la transformada clásica)

Transformada generalizada: idea

Necesitamos una descripción de la forma que vamos a detectar



- Usamos un punto de referencia (x_c, y_c), por ejemplo el centroide
- Cada punto de borde tiene un vector **r** asociado a su posición respecto de (x_c, y_c). Podemos caracterizarlo con su módulo r y ángulo β
- Sumamos la información direccional, esto es, el ángulo ϕ del vector gradiente

La tabla R

Con esta información podemos describir a la figura en la llamada *tabla-R*

$\phi_1 = 0$	$(r,\beta)_{1_1}$	$(r,\beta)_{1_2}$	$\cdots$	$(r,\beta)_{1_{n_1}}$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
ϕ_j	$(r,\beta)_{j_1}$	$(r,\beta)_{j_2}$	$\cdots$	$(r,\beta)_{j_{n_1}}$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\phi_k = \pi$	$(r,\beta)_{k_1}$	$(r,\beta)_{k_2}$	$\cdots$	$(r,\beta)_{k_{n_1}}$

- La información de cada pixel de borde (esto es, los vectores **r**) se agrupa según la direccion del gradiente (se utiliza una discretización)
- La tabla-R entonces es una descripción de la forma con información direccional
- Transformaciones básicas de la tabla permiten expresar transformaciones de la forma

Expresando transformaciones

Notemos al conjunto de vectores de la tabla R de una forma Γ bajo la orientación ϕ por $R_\Gamma(\phi)$. ¿Cómo se transforma la tabla si se transforma la forma?

- **Traslaciones**: la tabla no cambia, pues la información que tiene es relativa al centroide
- **Transformaciones de escala**: si T_s es una transformación de escala donde la forma se ve escalada por s, entonces $R_{T_s(\Gamma)}(\phi) = sR_\Gamma(\phi)$, es decir, cada vector $\mathbf{r}$ se ve escalado por s
- **Rotaciones**: si la forma se ve rotada por θ y llamamos a esa transformacion T_θ, tenemos

$$R_{T_\theta(\Gamma)}(\phi) = Rot\{R_\Gamma[(\phi - \theta)\ mod\ 2\pi],\ \theta\}$$

Esto es, tanto las orientaciones del gradiente como de los vectores $\mathbf{r}$ se ven modificadas.

Construcción de la tabla-R

1. Obtener la informacion de gradiente y los pixeles de borde para la imagen de referencia (el objeto o forma)
2. Inicializar vacías las entradas de la tabla R para los ángulos $\phi_1, \phi_2, \cdots, \phi_n$ (siguiendo cierta discretización para ϕ)
3. Calcular un punto de referencia (x_0, y_0) para la forma (por ejemplo el centroide)
4. Para cada pixel de borde (x, y), calcular:

$$\begin{cases} r = \sqrt{(x - x_0)^2 + (y - y_0)^2} \\ \beta = arctg(\frac{y_0 - y}{x_0 - x}) \end{cases}$$

5. Buscar $\phi_{x,y}$, la dirección del gradiente en ese pixel
6. Guardar la entrada (r, β) en $R(\phi_i)$, donde ϕ_i es el ángulo más cercano de la discretización a $\phi_{x,y}$

Algoritmo de detección

- Construida la tabla-R, podemos detectar la forma en una imagen con transformaciones (como las descritas) arbitrarias
- El espacio de parámetros esta dado por (x_0, y_0, s, θ)
- La información direccional nos permite reducir la búsqueda a entradas específicas de la tabla-R, y agrega precisión a la detección
- Los elementos de la tabla nos permiten determinar un centroide dados s, θ, x, y (similar a como se determinaba el centro de la elipse)

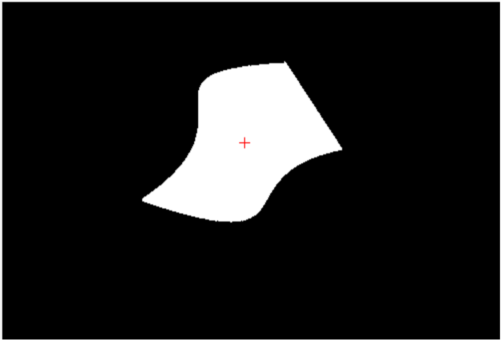
Algoritmo de detección

1. Obtener la informacion de gradiente y los pixeles de borde para la imagen
2. Inicializar en 0 el vector de acumulación $A[x_0, y_0, s, \theta]$
3. Para cada punto de borde (x,y), iterar por el subespacio de parámetros dado por (s, θ):
   - Con la direción del gradiente $\phi_{x,y}$ obtener las entradas de la tabla-R en $R(\phi_{x,y} - \theta \mod 2\pi)$
   - Para cada (r, β) en esa entrada de la tabla determinar x_0, y_0 según:

$$\begin{cases} x_0 = x - s(r\cos(\beta + \theta)) \\ y_0 = y - s(r\sin(\beta + \theta)) \end{cases}$$

   - Incrementar $A[x_0, y_0, s, \theta]$
4. Buscar máximos locales en A. Estos determinan los parámetros de la forma transformada en la imagen (posición del centroide, escala y rotación).

Ejemplo



Detección con imágen sintética (se detecta la forma de la derecha en la imagen de la izquierda).

Otros comentarios

- Podemos agregar más parámetros para expresar transformaciones
  - en lugar de una escala uniforme s, dos parámetros de escala ortogonales (s_x, s_y)
  - reflexiones
- estas transformaciones afectan la determinación del centroide a la hora del voto (paso 3.2 del algoritmo anterior): en efecto son transformaciones de la tabla-R
- El algoritmo es completamente paralelizable