
CAPÍTULO 1

LSD: “Line Segment Detection”

1.1. Introducción

LSD es un algoritmo de detección de segmentos publicado por Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel y Gregory Randall en abril de 2010. Es temporalmente lineal, tiene precisión inferior a un píxel y no requiere de un tuneo previo de parámetros, como casi todos los demás algoritmos de idéntica función; puede ser considerado el estado del arte en cuanto a detección de segmentos en imágenes digitales. Como cualquier otro algoritmo de detección de segmentos, LSD basa su estudio en la búsqueda de contornos angostos dentro de la imagen. Estos son regiones en donde el nivel de brillo de la imagen cambia notoriamente entre píxeles vecinos, por lo que el gradiente de la misma resulta de vital importancia. Se genera previo al análisis de la imagen, un campo de orientaciones asociadas a cada uno de los píxeles denominado por los autores *level-line orientation field*. Dicho campo se obtiene de calcular las orientaciones ortogonales a los ángulos asociados al gradiente de la imagen. Luego, LSD puede verse como una composición de tres pasos:

- (1) División de la imagen en las llamadas *line-support regions*, que son grupos conexos de píxeles con idéntica orientación, hasta cierta tolerancia.
- (2) Búsqueda del segmento que mejor aproxime cada *line-support region*: aproximación de las regiones por rectángulos.
- (3) Validación o no de cada segmento detectado en el punto anterior.

Los puntos (1) y (2) están basados en el algoritmo de detección de segmentos de Burns, Hanson y Riseman y el punto (3) es una adaptación del método *a contrario* de Desolneux, Moisan y Morel.

1.2. *Line-support regions*

El primer paso de LSD es el dividir la imagen en regiones conexas de píxeles con igual orientación, a menos de cierta tolerancia τ , llamadas *line-support regions*. El método para realizar tal división es del tipo “región creciente”; cada región comienza por un píxel y cierto ángulo asociado al mismo. Luego, se testean sus ocho vecinos y los que cuenten con un ángulo similar al del píxel original son incluidos en la región. En cada iteración el ángulo asociado a la región es calculado como el promedio de las orientaciones de cada píxel dentro de la *line-support region*; la iteración

termina cuando ya no se pueden agregar más píxeles a la misma.

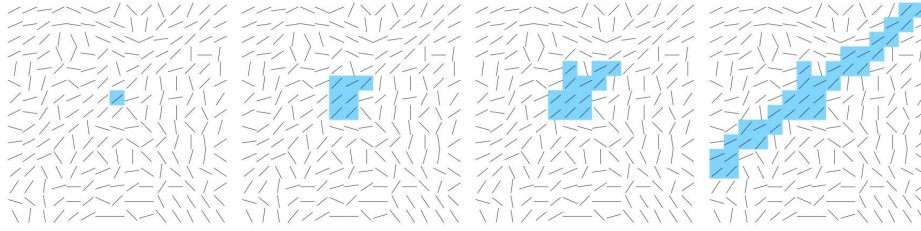


Figura 1.1: Proceso de crecimiento de una región. El ángulo asociado cada píxel de la imagen está representado por los pequeños segmentos y los píxeles coloreados representan la formación de la región. Fuente [?].

Los píxeles agregados a una región son marcados de manera que no se vuelvan a testear. Para mejorar el desempeño del algoritmo, las regiones comienzan a evaluarse por los píxeles con gradientes de mayor amplitud ya que estos representan mejor los bordes.

Existen algunos casos puntuales en los que el proceso de búsqueda de *line-support regions* puede arrojar errores. Por ejemplo, cuando se tienen dos segmentos que se juntan y que son colineales a no ser por la tolerancia τ descrita anteriormente, se detectarán ambos segmentos como uno solo; ver figura 1.2. Este potencial problema es heredado del algoritmo de Burns, Hanson y Riseman.



Figura 1.2: Potencial problema heredado del algoritmo de Burns, Hanson y Riseman. Izq.: Imagen original. Ctro.: Segmento detectado. Der.: Segmentos que deberían haberse detectado. Fuente [?].

Sin embargo, LSD plantea un método para ahorrarse este tipo de problemas. Durante el proceso de crecimiento de las regiones, también se realiza la aproximación rectangular a dicha región (paso (2) de los tres definidos anteriormente); y si menos del 50% de los píxeles dentro del rectángulo corresponden a la *line-support region*, entonces lo que se tiene no es un segmento. Se detiene entonces el crecimiento de la región.

1.3. Aproximación de las regiones por rectángulos

Cada *line-support region* debe ser asociada a un segmento. Cada segmento será determinado por su centro, su dirección, su anchura y su longitud. A diferencia de lo que pudiérase dictar la intuición, la dirección asociada al segmento no se corresponde con la asociada a la región (el promedio de las direcciones de cada uno de los píxeles). Sin embargo, se elige el centro del segmento como el centro de masa de la región y su dirección como el eje de inercia principal de la misma; la magnitud del gradiente asociado a cada píxel hace las veces de masa. La idea detrás de este método es que los píxeles con un gradiente mayor en módulo, se corresponden mejor con la percepción de un borde. La anchura y la longitud del segmento son elegidos de manera de cubrir el 99% de la masa de la región.

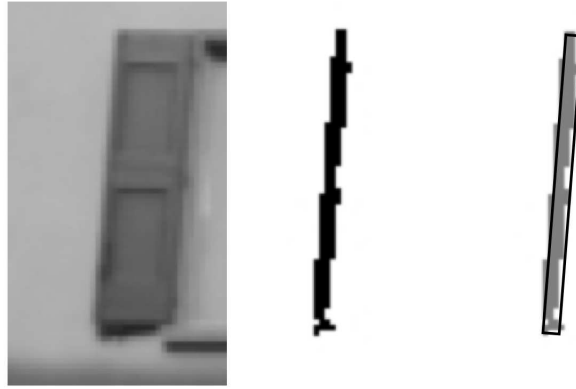


Figura 1.3: Búsqueda del segmento que mejor aproxime cada *line-support region*: aproximación de una región por un rectángulo. Izq.: Imagen original. Ctro.: Una de las regiones computadas. Der.: Aproximación rectangular que cubre el 99 % de la masa de la región. Fuente [?].

1.4. Validación de segmentos

La validación de los segmentos previamente detectados se plantea como un método de test de hipótesis. Se utiliza un modelo *a-contrario*: dada una imagen, sin ningún tipo de estructura, de ruido blanco y Gaussiano; se sabe que cualquier detección sobre la misma será casual y por lo tanto irrelevante. En rigor, se sabe que para cualquier imagen de este tipo, su *level-line orientation field* toma, para cada píxel, valores independientes y uniformemente distribuidos entre $[0, 2\pi]$. Dado entonces un segmento en la imagen analizada, se estudia la probabilidad de que dicha detección se dé en la imagen de ruido, y si ésta es lo suficientemente baja, se la considerará una detección válida, de lo contrario se considerará que se está bajo la hipótesis H_0 : un conjunto aleatorio de píxeles que casualmente se alinearon de manera de detectar un segmento.

Para estudiar la probabilidad de una cierta detección en la imagen de ruido, se deben tomar en cuenta todos los posibles rectángulos en la imagen. Dada una imagen $N \times N$, habrán N^4 orientaciones potenciales para los segmentos, N^2 puntos de inicio y N^2 puntos de fin. Si se consideran N posibles valores para la anchura de los rectángulos, se obtienen N^5 posibles segmentos. Por su parte, dado cierto rectángulo r , detectado en la imagen x , se denota $k(r, x)$ a la cantidad de píxeles alineados dentro del mismo. Se define además un valor llamado *Number of False Alarms* (NFA) que está fuertemente relacionado con la probabilidad de detectar al rectángulo en cuestión en la imagen de ruido X :

$$NFA(r, x) = N^5 \cdot P_{H_0}[k(r, X) \geq k(r, x)]$$

véase que el valor se logra multiplicando la probabilidad de que un segmento de la imagen de ruido, de tamaño igual a r , tenga un número mayor o igual de píxeles alineados que éste, por la cantidad potencial de segmentos N^5 . Cuanto menor sea el número NFA, más significativo será el segmento detectado r ; pues tendrá una probabilidad de aparición menor en una imagen sin estructuras. De esta manera, se descartará H_0 ; o lo que es lo mismo, se aceptará el segmento detectado como válido, si y sólo si:

$$NFA(r) \leq \varepsilon$$

donde empíricamente $\varepsilon = 1$ para todos los casos.

Si se toma en cuenta que cada píxel del gradiente de la imagen ruidosa toma un valor independiente de los demás, se concluye que también lo harán el gradiente y el *level-line orientation field* de la misma. De esta manera, definido sobre ésta un segmento con cierta orientación, la probabilidad de que uno de sus píxeles cuente con la misma orientación, a menos de la ya mencionada tolerancia τ ;

será:

$$p = \frac{\tau}{\pi}$$

además, se puede modelar la probabilidad de que cierto rectángulo en la imagen ruidosa, con cualquier orientación, formado por $n(r)$ píxeles, cuente con al menos $k(r)$ de ellos alineados, como una distribución binomial:

$$P_{H_0}[k(r, X) \geq k(r, x)] = B(n(r), k(r), p).$$

Finalmente, el valor *Number of False Alarms* será calculado para cada segmento detectado en la imagen analizada de la siguiente manera:

$$NFA(r, x) = B(n(r), k(r), p);$$

si dicho valor es menor o igual a $\varepsilon = 1$, el segmento se tomará como válido; de lo contrario se descartará.

1.5. Optimización del algoritmo para tiempo real

[?].