TECNOLOGICO NACIONAL DE MEXICO

INSTITUTO TECNOLOGICO DE MORELIA

“José María Morelos y Pavón”

Graficación

Jesús Eduardo Alcaraz Chávez

Funciones Primitivas de Dibujo OpenCv

Carlos Jahir Castro Cázares

Ingeniería en Sistemas Computacionales

Grupo: A

Semestre Ene-Jun 2019

00 de Mes del 2019

circle

La función círculo dibuja un círculo simple o lleno con un centro y un radio dados.

Función:

```
void circle(Mat& img, Point center, int radius,
const Scalar& color, int thickness=1, int
lineType=8, int shift=0)
```

Parámetros:

- **img -** Imagen donde se dibuja el círculo.
- **center -** centro del círculo.
- **radius -** Radio del círculo.
- **color -** círculo de color.
- **thickness -** grosor del contorno del círculo, si es positivo. El grosor negativo significa que se debe dibujar un círculo relleno.
- **lineType -** Tipo del límite del círculo. Ver la descripción de la línea ().
- **shift -** Número de bits fraccionarios en las coordenadas del centro y en el valor del radio.

line

La función dibuja el segmento de línea entre los puntos pt1 y pt2 en la imagen. La línea está recortada por los límites de la imagen. Para líneas no suavizadas con coordenadas enteras, se utiliza el algoritmo de Bresenham conectado a 8 o conectado a 4. Las líneas gruesas se dibujan con terminaciones redondeadas. Las líneas antialias se dibujan usando filtrado gaussiano. Para especificar el color de la línea, puede usar la macro CV_RGB (r, g, b).

Función:

```
void line(Mat& img, Point pt1, Point pt2, const
Scalar& color, int thickness=1, int lineType=8, int
shift=0)
```

Parámetros:

- **img -** Imagen.
- **pt1 -** Primer punto del segmento de línea.
- **pt2 -** Segundo punto del segmento de línea.
- **color -** color de línea.
- **thickness -** espesor de la línea.
- **lineType –**Tipo de la línea:8 (u omitido) - 8 líneas conectadas, 4 - 4 líneas conectadas y CV_AA - línea suavizada.
- **shift -** Número de bits fraccionarios en las coordenadas del punto.

arrowedLine

La función arrowedLine dibuja una flecha entre los puntos pt1 y pt2 en la imagen. Véase también la línea ().

Función:

```
void arrowedLine(Mat& img, Point pt1, Point pt2,
const Scalar& color, int thickness=1, int
line_type=8, int shift=0, double tipLength=0.1)
```

Parámetros:

- **img -** Imagen.
- **pt1 -** El punto desde el que comienza la flecha.
- **pt2 -** El punto al que apunta la flecha.
- **color -** color de línea.
- **thickness -** espesor de la línea.
- **line_type -** Tipo de línea: 8 (u omitido) - 8 líneas conectadas, 4 – 4 líneas conectadas y CV_AA - línea suavizada.
- **shift -** Número de bits fraccionarios en las coordenadas del punto.
- **tipLength -** la longitud de la punta de la flecha en relación con la longitud de la flecha

ellipse

Las funciones elipse con menos parámetros dibujan un contorno de elipse, una elipse rellena, un arco elíptico o un sector de elipse rellena. Se utiliza una curva lineal por partes para aproximar el límite del arco elíptico. Si necesita más control de la representación de la elipse, puede recuperar la curva utilizando ellipse2Poly () y luego renderizarla con polilíneas () o rellenarla con fillPoly (). Si usa la primera variante de la función y desea dibujar la elipse completa, no un arco, pase startAngle = 0 y endAngle = 360. La siguiente figura explica el significado de los parámetros.

Función:

```
void ellipse(Mat& img, Point center, Size axes,
double angle, double startAngle, double endAngle,
const Scalar& color, int thickness=1, int
lineType=8, int shift=0)
```

```
void ellipse(Mat& img, const RotatedRect& box, const
Scalar& color, int thickness=1, int lineType=8)
```

Parámetros:

- **img -** Imagen.
- **center -** centro de la elipse.
- **axes -** La mitad del tamaño de los ejes principales de la elipse.
- **angle -** ángulo de rotación elipse en grados.
- **startAngle -** ángulo de inicio del arco elíptico en grados.
- **endAngle -** ángulo de finalización del arco elíptico en grados.
- **box -** Representación de elipse alternativa a través de RotatedRect o CvBox2D. Esto significa que la función dibuja una elipse inscrita en el rectángulo girado.
- **color -** color elipse.

- **thickness -** grosor del contorno del arco de elipse, si es positivo. De lo contrario, esto indica que se debe dibujar un sector de elipse relleno.
- **lineType -** Tipo de límite de elipse. Ver la descripción de la línea (). shift - Número de bits fraccionarios en las coordenadas del centro y valores de los ejes.

fillConvexPoly

La función fillConvexPoly dibuja un polígono convexo relleno. Esta función es mucho más rápida que la función fillPoly. Puede rellenar no solo polígonos convexos sino también cualquier polígono monótono sin auto-intersecciones, es decir, un polígono cuyo contorno intersecta cada línea horizontal (línea de escaneo) dos veces como máximo (sin embargo, su extremo superior y / o el borde inferior podrían ser horizontal).

Función:

void fillConvexPoly(Mat& img, const Point* pts, int npts, const Scalar& color, int lineType=8, int shift=0)

Parámetros:

- **img -** Imagen.
- **pts -** vértices de polígonos.
- **npts -** Número de vértices de polígonos.
- **color -** color del polígono.
- **lineType -** Tipo de los límites del polígono.
- **shift -** Número de bits fraccionarios en las coordenadas del vértice.

fillPoly

La función fillPoly rellena un área delimitada por varios contornos poligonales. La función puede rellenar áreas complejas, por ejemplo, áreas con orificios, contornos con auto-intersecciones (algunas de sus partes), etc.

Función:

void fillPoly(Mat& img, const Point** pts, const int* npts, int ncontours, const Scalar& color, int lineType=8, int shift=0, Point offset=Point())

Parámetros:

- **img -** Imagen.
- **Pts -** matriz de polígonos donde cada polígono se representa como una matriz de puntos.
- **npts -** Arreglo de contadores de vértices de polígonos.
- **Ncontours -** número de contornos que se unen a la región rellena.
- **color -** color del polígono.
- **lineType -** Tipo de los límites del polígono.
- **shift -** Número de bits fraccionarios en las coordenadas del vértice.
- **offset -** Offset opcional de todos los puntos de los contornos.

rectangle

La función dibuja un contorno de rectángulo o un rectángulo relleno cuyas dos esquinas opuestas son pt1 y pt2.

Función:

void rectangle(Mat& img, Point pt1, Point pt2, const Scalar& color, int thickness=1, int lineType=8, int shift=0)

void rectangle(Mat& img, Rect rec, const Scalar& color, int thickness=1, int lineType=8, int shift=0)

Parámetros:

- **img -** Imagen.
- **pt1 -** Vértice del rectángulo.
- **pt2 -** Vértice del rectángulo opuesto a pt1.
- **rec -** especificación alternativa del rectángulo dibujado.
- **color -** color del rectángulo o brillo (imagen en escala de grises).
- **thickness -** grosor de las líneas que forman el rectángulo. Los valores negativos, como CV_FILLED, significan que la función tiene que dibujar un rectángulo relleno.
- **lineType -** Tipo de línea. Ver la descripción de la línea ().
- **shift -**Número de bits fraccionarios en las coordenadas del punto.

polylines

La función dibuja una o más curvas poligonales.

Función:

void polylines(Mat& img, const Point** pts, const int* npts, int ncontours, bool isClosed, const Scalar& color, int thickness=1, int lineType=8, int shift=0)

void polylines(InputOutputArray img, InputArrayOfArrays pts, bool isClosed, const Scalar& color, int thickness=1, int lineType=8, int shift=0)

Parámetros:

- **img -** Imagen.
- **Ptos -** Arreglo de curvas poligonales.
- **npts -** Arreglo de contadores de vértices de polígonos.
- **ncontours -** Número de curvas.
- **isClosed:** indicador que indica si las polilíneas dibujadas están cerradas o no. Si están cerradas, la función dibuja una línea desde el último vértice de cada curva hasta su primer vértice.
- **color -** color polilínea.
- **thickness -** Grosor de los bordes de polilínea.
- **lineType -** Tipo de los segmentos de línea.
- **shift -** Número de bits fraccionarios en las coordenadas del vértice.

putText

La función putText representa la cadena de texto especificada en la imagen. Los símbolos que no se pueden representar con la fuente especificada se reemplazan por signos de interrogación. Consulte getTextSize () para ver un ejemplo de código de representación de texto.

Función:

void putText(Mat& img, const string& text, Point org, int fontFace, double fontScale, Scalar color, int thickness=1, int lineType=8, bool bottomLeftOrigin=false)

Parámetros:

- **img -** Imagen.
- **text –**Cadena de texto a dibujar.
- **org -** esquina inferior izquierda de la cadena de texto en la imagen.
- **font –** la estructura CvFont se inicializó utilizando InitFont ().
- **FontFace -** Tipo de fuente.
- **fontScale -** factor de escala de la fuente que se multiplica por el tamaño base específico de la fuente.
- **color -** color del texto.
- **thickness –** grosor de las líneas utilizadas para dibujar un texto.
- **lineType -** Tipo de línea.
- **bottomLeftOrigin -** cuando es verdadero, el origen de los datos de la imagen se encuentra en la esquina inferior izquierda. De lo contrario, está en la esquina superior izquierda.