Drawing Function

Sung Soo Hwang

Drawing Function

- Rectangle
  - void rectangle(Mat& img, Point pt1, Point pt2, const Scalar& color, int thickness=1, int lineType=8, int shift=0)
    - Img – image
    - Pt1: vertex of the rectangle
    - Pt2: vertex of the rectangle opposite to pt1
    - Color: rectangle color or brightness(for grayscale)
    - Thickness: thickness of lines that make up the rectangle
      - Negative values to draw filled rectangle
    - lineType: type of the line. See line() description
    - Shift: number of fractional bits in the point coordinates
      - $Point(x, y) \rightarrow Point2f(x \times 2^{-shift}, y \times 2^{-shift})$
  - void rectangle(Mat& img, Rect rec, const Scalar& color, int thickness=1, int lineType=8, int shift=0)
    - Rect: alternative specification of the drawn rectangle
      - Rect(x_LT, y_LT, width, height)

Drawing Function

- Rectangle
  - Example code

```
int main(){
	Mat image = imread("lena.jpg");
	Rect rect = Rect(10, 10, 100, 100); // LT position, width, height
	rectangle(image, rect, Scalar(255, 0, 0), 4, 8, 0);
	imshow("image",image);
	waitKey(0);
	return 0;
}
```

Drawing Function

- Line/Circle
  - void line(Mat& img, Point pt1, Point pt2, const Scalar& color, int thickness=1, int lineType=8, int shift=0)
    - Pt1: first point of the line segment
    - Pt2: second point of the line segment
    - lineType:
      - 8(or omitted): 8-connected line.
      - 4: 4-connected line.
      - CV_AA: antialiased line
  - void circle(Mat& img, Point center, int radius, const Scalar& color, int thickness=1, int lineType=8, int shift=0)
    - Center: center of the circle
    - Radius: radius of the circle

Drawing Function

- Line/Circle
  - Example code

```
int main(){
	Mat image = imread("lena.jpg");
	Point p1(25, 25), p2(100, 50);
	line(image, p1, p2, Scalar(255, 0, 0), 3, 8, 0);
	imshow("image",image);
	waitKey(0);
	return 0;
}
```

Drawing Function

- Polygon
  - void fillPoly(Mat& img, const Point** pts, const int* npts, int ncontours, const Scalar& color, int lineType=8, int shift=0, Point offset=Point())
    - Img – image
    - Pts – Array of polygons where each polygon is represented as an array of points
    - Npts – Array of polygon vertex counters
    - Ncontours – number of contours that bind the filled region
    - Color – polygon color
    - lineType – type of the polygon boundaries
    - Shift – number of fractional bits in the vertex coordinates
    - Offset – optional offset of all points of the contours

Drawing Function

- Polygon
  - Example code

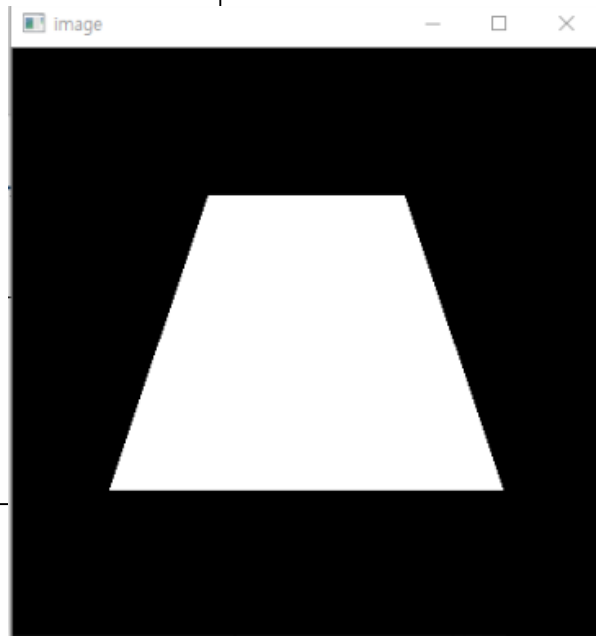
```
int main() {
    Mat image = Mat::zeros(400, 400, CV_8UC3);
    int w = 400;

    Point trapezoid[1][4];
    trapezoid[0][0] = Point(w*2 / 6, w / 4);
    trapezoid[0][1] = Point(w*4 / 6, w / 4);
    trapezoid[0][2] = Point(w*5 / 6, w*3 / 4);
    trapezoid[0][3] = Point(w / 6, w*3 / 4);

    const Point* ppt[1] = { trapezoid[0] };
    int npt[] = { 4 };

    fillPoly(image, ppt, npt, 1, Scalar(255, 255, 255), 8);
    imshow("image", image);

    waitKey(0);
}
```



Drawing Function

- Write text
  - void putText(Mat& img, const string& text, Point org, int fontFace, double fontScale, Scalar color, int thickness=1, int lineType=8, bool bottomLeftOrigin=false
    - Text: text string to be drawn
    - Org: bottom-left corner of the text string in the image
    - Font: CVFont structure using InitFont()
    - fontFace: FONT_TYPE(FONT_HERSHEY_SIMPLEX, FONT_HERSHEY_PLAIN, FONT_HERSHEY_DUPLEX, FONT_HERSHEY_COMPLEX, FONT_HERSHEY_TRIPLEX, … can be combined with FONT_HERSHEY_ITALIC)
    - fontScale: Font scale factor that is multiplied by the font-specific base size
    - bottomLeftOrgin: when true, the image data origin is at the bottom-left corner. Otherwise, it is at the top-left corner

Drawing Function

- Write text
  - String cv::format(const char *fmt, ...)
    - Returns a text string formatted using the printf-like expression.
    - Params: fmt – printf-compatible formatting specifiers.

Drawing Function

- Write text
  - Example code

```
int main() {
	// Create black empty images
	Mat image = Mat::zeros(400, 600, CV_8UC3);
	int w = image.cols;
	int h = image.rows;
	putText(image, format("width: %d, height: %d", w, h),
	Point(50, 80), FONT_HERSHEY_SIMPLEX, 1,
	Scalar(0, 200, 200), 4);
	imshow("image", image);

	waitKey(0);
	return(0);
}
```