OpenCV--绘图函数

openCV也提供了很多绘图的方法库。

- 直线

cvLine是绘图函数中最简单的方法，只需要用bresenham算法画一条直线即可。

```
/** @brief Draws 4-connected, 8-connected or antialiased line segment
connecting two points
@see cv::line
*/
CVAPI(void)  cvLine( CvArr* img, CvPoint pt1, CvPoint pt2,
                     CvScalar color, int thickness CV_DEFAULT(1),
                     int line_type CV_DEFAULT(8), int shift CV_DEFAULT(0)
);
```

cvPoint是一个简单的参数类，包括了x,y两个参数。我们可以通过CVPoint（int x,int y）来快速构造一个cvPoint类。剩下的参数都可以通过名称来了解一下功能。

- 圆形和椭圆

cvCircle是提供的一个画圆函数方法。

```
/** @brief Draws a circle with specified center and radius.
   Thickness works in the same way as with cvRectangle
@see cv::circle
*/

CVAPI(void)  cvCircle( CvArr* img, CvPoint center, int radius,
                       CvScalar color, int thickness CV_DEFAULT(1),
                       int line_type CV_DEFAULT(8), int shift
CV_DEFAULT(0));
```

画一个椭圆相对于画一个圆形来说参数较为复杂一些。

```
/** @brief Draws ellipse outline, filled ellipse, elliptic arc or filled
elliptic sector
   depending on _thickness_, _start_angle_ and _end_angle_ parameters. The
resultant figure
   is rotated by _angle_. All the angles are in degrees
@see cv::ellipse
*/
CVAPI(void)  cvEllipse( CvArr* img, CvPoint center, CvSize axes,
                        double angle, double start_angle, double end_angle,
                        CvScalar color, int thickness CV_DEFAULT(1),
                        int line_type CV_DEFAULT(8), int shift
CV_DEFAULT(0));
```

一个完整的椭圆必须要将参数start_angle和end_angle两个参数设置为0度到360度。

其次，还有一种外接矩形的椭圆绘制。

```
CV_INLINE  void  cvEllipseBox( CvArr* img, CvBox2D box, CvScalar color,
                               int thickness CV_DEFAULT(1),
                               int line_type CV_DEFAULT(8), int shift
CV_DEFAULT(0) )
{
    CvSize axes;
    axes.width = cvRound(box.size.width*0.5);
    axes.height = cvRound(box.size.height*0.5);
    cvEllipse( img, cvPointFrom32f( box.center ), axes, box.angle,
               0, 360, color, thickness, line_type, shift );
}
```

- 多边形

绘制多边形函数方法。cvFillPoly,cvFillConvexPoly，cvPolyLine.

```
/** @brief Fills convex or monotonous polygon.
@see cv::fillConvexPoly
*/
CVAPI(void)  cvFillConvexPoly( CvArr* img, const CvPoint* pts, int npts,
CvScalar color,
                               int line_type CV_DEFAULT(8), int shift
CV_DEFAULT(0));
/** @brief Fills an area bounded by one or more arbitrary polygons
@see cv::fillPoly
*/
CVAPI(void)  cvFillPoly( CvArr* img, CvPoint** pts, const int* npts,
                         int contours, CvScalar color,
                         int line_type CV_DEFAULT(8), int shift
CV_DEFAULT(0) );
/** @brief Draws one or more polygonal curves
@see cv::polylines
*/
CVAPI(void)  cvPolyLine( CvArr* img, CvPoint** pts, const int* npts, int
contours,
                         int is_closed, CvScalar color, int thickness
CV_DEFAULT(1),
                         int line_type CV_DEFAULT(8), int shift
CV_DEFAULT(0) );
```

- 文字

最后一种形式是绘制文字，cvPutText方法。并且提供几种字体可以用。

```
/** Font structure */
typedef struct CvFont
{
  const char* nameFont;   //Qt:nameFont
```

```
  CvScalar color;       //Qt:ColorFont -> cvScalar(blue_component,
green_component, red_component[, alpha_component])
    int        font_face;    //Qt: bool italic         /** =CV_FONT_* */
    const int*  ascii;      //!< font data and metrics
    const int*  greek;
    const int*  cyrillic;
    float      hscale, vscale;
    float      shear;      //!< slope coefficient: 0 - normal, >0 - italic
    int        thickness;    //!< Qt: weight               /** letters
thickness */
    float      dx;       //!< horizontal interval between letters
    int        line_type;    //!< Qt: PointSize
}
CvFont;
/** @brief Initializes font structure (OpenCV 1.x API).
The function initializes the font structure that can be passed to text
rendering functions.
@param font Pointer to the font structure initialized by the function
@param font_face Font name identifier. See cv::HersheyFonts and
corresponding old CV_* identifiers.
@param hscale Horizontal scale. If equal to 1.0f , the characters have the
original width
depending on the font type. If equal to 0.5f , the characters are of half
the original width.
@param vscale Vertical scale. If equal to 1.0f , the characters have the
original height depending
on the font type. If equal to 0.5f , the characters are of half the
original height.
@param shear Approximate tangent of the character slope relative to the
vertical line. A zero
value means a non-italic font, 1.0f means about a 45 degree slope, etc.
@param thickness Thickness of the text strokes
@param line_type Type of the strokes, see line description
@sa cvPutText
*/
CVAPI(void)  cvInitFont( CvFont* font, int font_face,
                         double hscale, double vscale,
                         double shear CV_DEFAULT(0),
                         int thickness CV_DEFAULT(1),
                         int line_type CV_DEFAULT(8));
CV_INLINE CvFont cvFont( double scale, int thickness CV_DEFAULT(1) )
{
    CvFont font;
    cvInitFont( &font, CV_FONT_HERSHEY_PLAIN, scale, scale, 0, thickness,
CV_AA );
    return font;
}
/** @brief Renders text stroke with specified font and color at specified
location.
   CvFont should be initialized with cvInitFont
@see cvInitFont, cvGetTextSize, cvFont, cv::putText
```

```
*/
CVAPI(void)  cvPutText( CvArr* img, const char* text, CvPoint org,
                        const CvFont* font, CvScalar color );
/** @brief Calculates bounding box of text stroke (useful for alignment)
@see cv::getTextSize
*/
CVAPI(void)  cvGetTextSize( const char* text_string, const CvFont* font,
                            CvSize* text_size, int* baseline );
/** @brief Unpacks color value
if arrtype is CV_8UC?, _color_ is treated as packed color value, otherwise
the first channels
(depending on arrtype) of destination scalar are set to the same value =
_color_
*/
CVAPI(CvScalar)  cvColorToScalar( double packed_color, int arrtype );
```

CV可以支持字体库导入。

```
#define CV_FONT_HERSHEY_SIMPLEX         0
#define CV_FONT_HERSHEY_PLAIN           1
#define CV_FONT_HERSHEY_DUPLEX          2
#define CV_FONT_HERSHEY_COMPLEX         3
#define CV_FONT_HERSHEY_TRIPLEX         4
#define CV_FONT_HERSHEY_COMPLEX_SMALL   5
#define CV_FONT_HERSHEY_SCRIPT_SIMPLEX  6
#define CV_FONT_HERSHEY_SCRIPT_COMPLEX  7
#define CV_FONT_ITALIC                 16
#define CV_FONT_VECTOR0    CV_FONT_HERSHEY_SIMPLEX
```

并且可以选择配置字体的输出形式。