Learn OpenCV by Examples

OpenCV simplified for beginners by the use of examples. Learn OpenCV with basic implementation of different algorithms.

Hough Circle Detection

void **HoughCircles**(InputArray image, OutputArray circles, int method, double dp, double minDist, double param1=100, double param2=100, int minRadius=0, int maxRadius=0)
Parameters:

- **image** – 8-bit, single-channel, grayscale input image.
- **circles** – Output vector of found circles. Each vector is encoded as a 3-element floating-point vector (x,y,radius) .
- **circle_storage** – Memory storage that will contain the output sequence of found circles.
- **method** – Currently, the only implemented method is `CV_HOUGH_GRADIENT.`
- **dp** – Inverse ratio of the accumulator resolution to the image resolution. For example, if `dp=1` , the accumulator has the same resolution as the input image. If `dp=2` , the accumulator has half as big width and height.
- **minDist** – Minimum distance between the centers of the detected circles. If the parameter is too small, multiple neighbor circles may be falsely detected in addition to a true one. If it is too large, some circles may be missed.
- **param1** – First method-specific parameter. In case of `CV_HOUGH_GRADIENT` , it is the higher threshold of the two passed to the `Canny()`edge detector (the lower one is twice smaller).
- **param2** – Second method-specific parameter. In case of `CV_HOUGH_GRADIENT` , it is

SEARCH CONTENTS OF THIS BLOG

Search

POPULAR POSTS

1 Find Contour

2 Basic drawing examples

3 Line Detection by Hough Line Transform

Face Detection using Haar-Cascade Classifier

Perspective Transform

Sobel Edge Detection

the accumulator threshold for the circle centers at the detection stage. The smaller it is, the more false circles may be detected. Circles, corresponding to the larger accumulator values, will be returned first.

- **minRadius** – Minimum circle radius.
- **maxRadius** – Maximum circle radius.

A good example for Hough Circle Transform is provided in OpenCV Documentation.

Steps:

1. Load image and convert to gray-scale.
2. Blur (low pass filter) the image to reduce noise.
3. Apply the Hough Transform to find the circles.(HoughCircles)
4. Draw the circles detected.(circle)
5. Show the result

Functions:
HoughCircles, circle, GaussianBlur, cvtColor, imshow, imread, namedWindow, waitKey.

Example:

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <iostream>
#include <stdio.h>

using namespace cv;
using namespace std;

int main()
{
  Mat src, gray;
  src = imread( "building.jpg", 1 );resize(src,src,Size(640,480));
  cvtColor( src, gray, CV_BGR2GRAY );

  // Reduce the noise so we avoid false circle detection
  GaussianBlur( gray, gray, Size(9, 9), 2, 2 );

  vector<Vec3f> circles;

  // Apply the Hough Transform to find the circles
  HoughCircles( gray, circles, CV_HOUGH_GRADIENT, 1, 30, 200, 50, 0

  // Draw the circles detected
  for( size_t i = 0; i < circles.size(); i++ )
  {
      Point center(cvRound(circles[i][0]), cvRound(circles[i][1]));
      int radius = cvRound(circles[i][2]);
```

Kalman Filter Implementation (Tracking mouse position)

Histogram Calculation

9 OpenCV example to convert RGB to gray / other color spaces

10 Hough Circle Detection

CATEGORIES

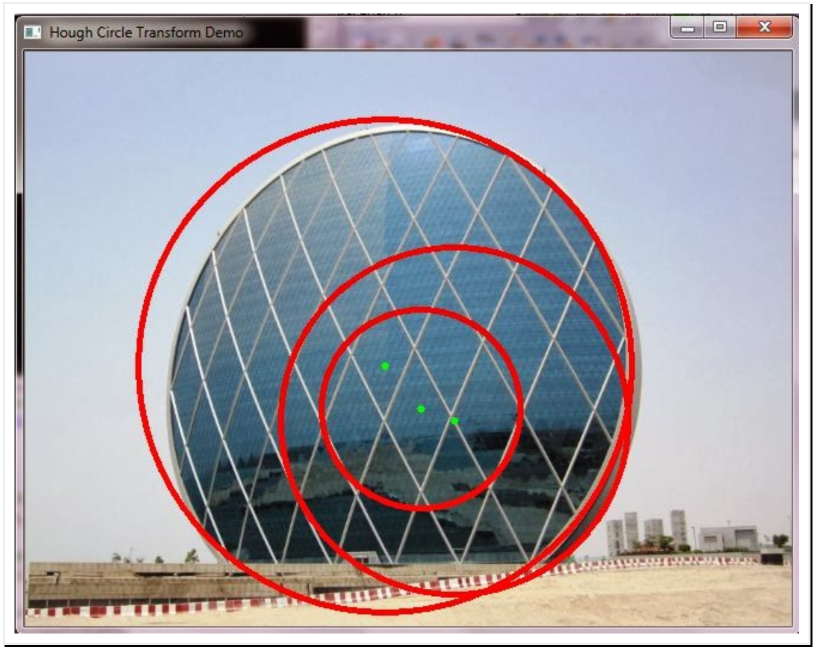
- Accessory
- Applications
- Basics
- Edge Detection
- Feature Extraction
- Filter
- Miscellaneous
- Morphological Operation

```
      circle( src, center, 3, Scalar(0,255,0), -1, 8, 0 );// circle
      circle( src, center, radius, Scalar(0,0,255), 3, 8, 0 );// ci
      cout << "center : " << center << "\nradius : " << radius << e
  }

  // Show your results
  namedWindow( "Hough Circle Transform Demo", CV_WINDOW_AUTOSIZE );
  imshow( "Hough Circle Transform Demo", src );

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

Result:



Note:

- If this code is not able to detect the circles that you want to detect, then play with the parameters (dp, minDist, param1 & param2).
- Keeping rest parameters constant, if you increase dp, then increase param2 too to avoid false detection.

g+1 Recommend this on Google