HOUGH TRANSFORM : CIRCLE

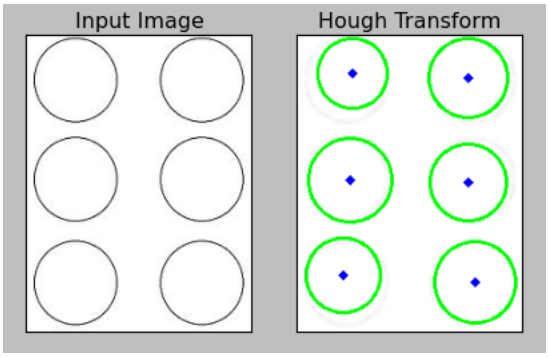
Hough transform

The classical Hough transform was developed to identify lines in the image, but later the Hough transform has been extended to identify the positions of arbitrary shapes, most commonly circles or ellipses.

"In many cases an edge detector can be used as a pre-processing stage to obtain image points or image pixels that are on the desired curve in the image space. Due to imperfections in either the image data or the edge detector, however, there may be missing points or pixels on the desired curves as well as spatial deviations between the ideal line/circle/ellipse and the noisy edge points as they are obtained from the edge detector. For these reasons, it is often non-trivial to group the extracted edge features to an appropriate set of lines, circles or ellipses. The purpose of the Hough transform is to address this problem by making it possible to perform groupings of edge points into object candidates by performing an explicit voting procedure over a set of **parameterized** image objects." - wiki - Hough transform (http://en.wikipedia.org/wiki/Hough_transform).

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

bgr_img = cv2.imread('b.jpg') # read as it is

if bgr_img.shape[-1] == 3:           # color image
    b,g,r = cv2.split(bgr_img)       # get b,g,r
    rgb_img = cv2.merge([r,g,b])     # switch it to rgb
    gray_img = cv2.cvtColor(bgr_img, cv2.COLOR_BGR2GRAY)
else:
    gray_img = bgr_img

img = cv2.medianBlur(gray_img, 5)
cimg = cv2.cvtColor(img,cv2.COLOR_GRAY2BGR)

circles = cv2.HoughCircles(img,cv2.HOUGH_GRADIENT,1,20,
                           param1=50,param2=30,minRadius=0,maxRadius=0)

circles = np.uint16(np.around(circles))

for i in circles[0,:]:
    # draw the outer circle
    cv2.circle(cimg,(i[0],i[1]),i[2],(0,255,0),2)
    # draw the center of the circle
    cv2.circle(cimg,(i[0],i[1]),2,(0,0,255),3)

plt.subplot(121),plt.imshow(rgb_img)
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(cimg)
plt.title('Hough Transform'), plt.xticks([]), plt.yticks([])
plt.show()
```

The function we use here is **cv2.HoughCircles()**. It has plenty of arguments which are well explained in the documentation. So we directly go to the code.



Edge Detection - Sobel and Laplacian Kernels (/python/OpenCV_Python/pythor

Canny Edge Detection (/python/OpenCV_Python/pythor

Hough Transform - Circles (/python/OpenCV_Python/pythor

Watershed Algorithm : Marker-based Segmentation I (/python/OpenCV_Python/pythor

Watershed Algorithm : Marker-based Segmentation II (/python/OpenCV_Python/pythor

Image noise reduction : Non-local Means denoising algorithm (/python/OpenCV_Python/pythor local_Means_Denoising_Algorithr

Image object detection : Face detection using Haar Cascade Classifiers (/python/OpenCV_Python/pythor

Image segmentation - Foreground extraction Grabcut algorithm based on graph cuts (/python/OpenCV_Python/pythor

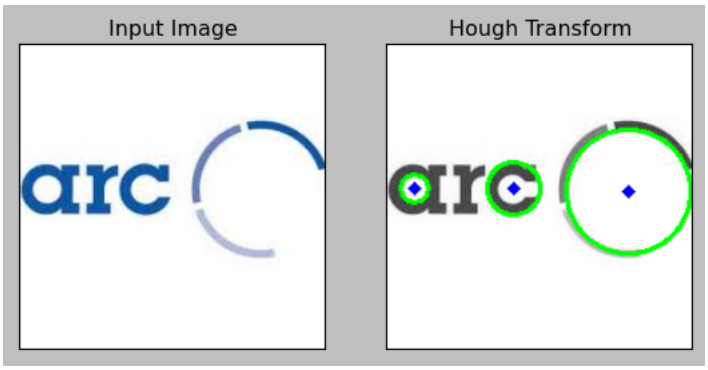
Image Reconstruction - Inpainting (Interpolation) - Fast Marching Methods (/python/OpenCV_Python/pythor

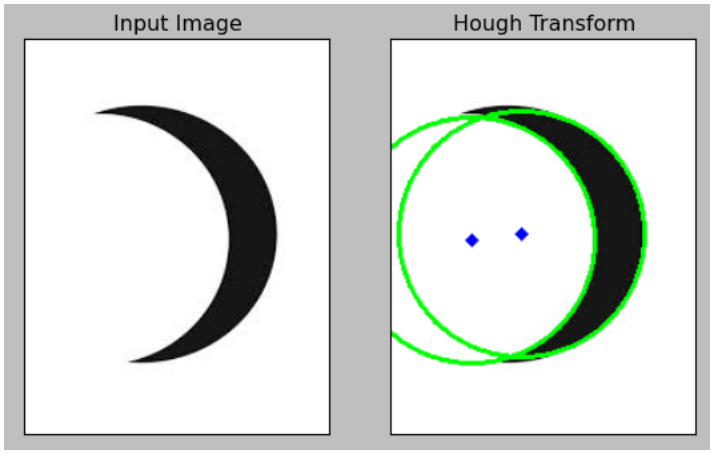
Video : Mean shift object tracking (/python/OpenCV_Python/pythor

Machine Learning : Clustering - K-Means clustering I (/python/OpenCV_Python/pythor Means_Clustering_Vector_Quanti

Machine Learning : Clustering - K-Means clustering II (/python/OpenCV_Python/pythor Means_Clustering_Vector_Quanti

Machine Learning :

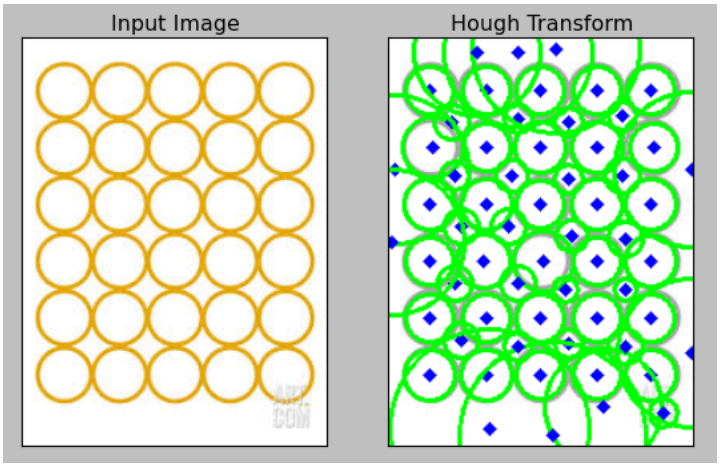




Spurious Circles

So far so good.

In the following cases, however, the Hough transform finds hidden circles which are not meant to be perceived as circles:

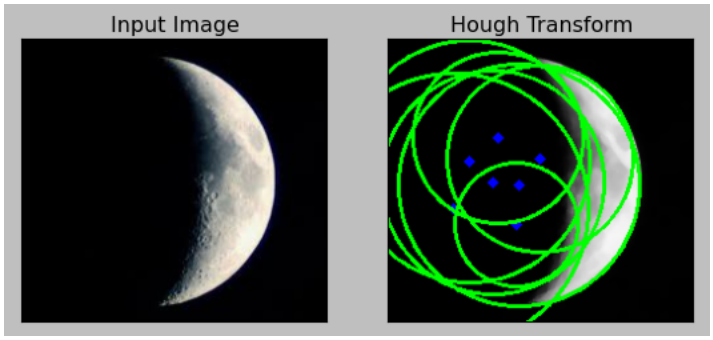


Classification - k-nearest neighbors (k-NN) algorithm (/python/OpenCV_Python/pythor nearest_neighbors_k-NN.php)

Python tutorial

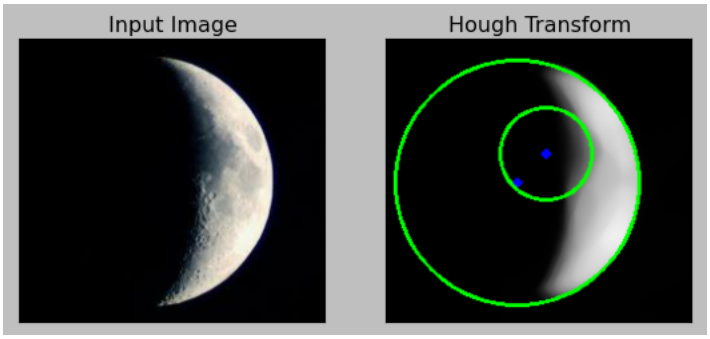
Python Home (/python/pytut.php)

Introduction

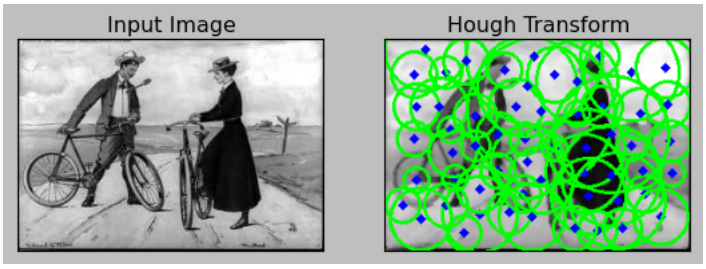


In this Crescent case, we can make it better by blurring more:

```
img = cv2.medianBlur(gray_img,5) => img = cv2.medianBlur(gray_img,25)
```



In general, without blurring, the algorithms tends to extract too many circular features. So, to be more successful, the prepocessing seems to be crucial.



With blurring value 51:

```
img = cv2.medianBlur(gray_img,51)
```

Parameterization

A circle can be fully defined with three parameters: the center coordinates (a, b) and the radius (R):

$$x = a + R\cos\theta$$

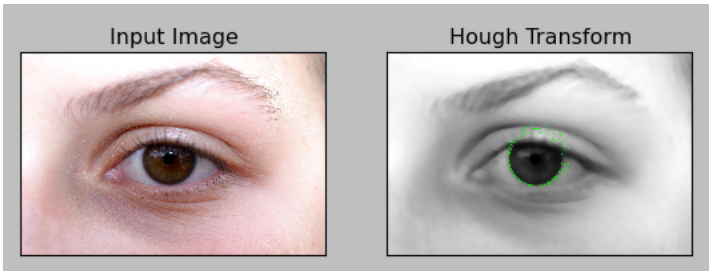
$$y = b + R\sin\theta$$

As the θ varies from 0 to 360, a complete circle of radius R is created.

So with the Circle Hough Transform, we expect to find triplets of (x, y, R) from the image. In other words, our purpose is to find those three parameters.

Therefore, we need to construct a 3D accumulator for Hough transform, which would be highly ineffective. So, OpenCV uses more trickier method, Hough Gradient Method which uses the gradient information of edges.

Hough transform Code



OpenCV 3 Tutorial
image & video processing

Installing on Ubuntu 13 (/OpenCV/opencv_3_tutorial_ubuntu14_install_cmake.php)

Mat(rix) object (Image Container) (/OpenCV/opencv_3_tutorial_mat_object_image_matrix_image_container.php)

Creating Mat objects (/OpenCV/opencv_3_tutorial_creating_mat_objects.php)

The core : Image - load, convert, and save (/OpenCV/opencv_3_tutorial_load_convert_save_image.php)

Smoothing Filters A - Average, Gaussian (/OpenCV/opencv_3_tutorial_imgproc_gausian_median_blur_bilateral_filter_image_smoothing.php)

Smoothing Filters B - Median, Bilateral (/OpenCV/opencv_3_tutorial_imgproc_gausian_median_blur_bilateral_filter_image_smoothing_B.php)

map, filter, and reduce (/python/python_fncs_map_filter

Decorators (/python/python_decorators.php

List Comprehension (/python/python_list_comprehen

Sets (union/intersection) and itertools - Jaccard coefficient and shingling to check plagiarism (/python/python_sets_union_inte

Hashing (Hash tables and hashlib) (/python/python_hash_tables_ha

Dictionary Comprehension with zip (/python/python_dictionary_com

The yield keyword (/python/python_function_with_y

Generator Functions and Expressions (/python/python_generators.php

generator.send() method (/python/python_function_with_g

Iterators (/python/python_iterators.php)

Classes and Instances (__init__, __call__, etc.) (/python/python_classes_instanc

if__name__ == '__main__' (/python/python_if__name__equa

argparse (/python/python_argparse.php)

Exceptions (/python/python_try_except_final

@static method vs class method (/python/python_differences_bet

Watershed Algorithm : Marker-based Segmentation I (/python/OpenCV_Python/python_opencv3_Image_Watershed_Algorithm_Marker_Based_Segmentation.php)

Watershed Algorithm : Marker-based Segmentation II (/python/OpenCV_Python/python_opencv3_Image_Watershed_Algorithm_Marker_Based_Segmentation_2.php)

Image noise reduction : Non-local Means denoising algorithm (/python/OpenCV_Python/python_opencv3_Image_Non-local_Means_Denoising_Algorithm_Noise_Reduction.php)

Image object detection : Face detection using Haar Cascade Classifiers (/python/OpenCV_Python/python_opencv3_Image_Object_Detection_Face_Detection_Haar_Cascade_Classifiers.php)

Image segmentation - Foreground extraction Grabcut algorithm based on graph cuts (/python/OpenCV_Python/python_opencv3_Image_Segmentation_by_Foreground_Extraction_Grabcut_Algorithm_based_on_Graph_Cuts.php)

Image Reconstruction - Inpainting (Interpolation) - Fast Marching Methods (/python/OpenCV_Python/python_opencv3_Image_reconstruction_Inpainting_Interpolation.php)

Video : Mean shift object tracking (/python/OpenCV_Python/python_opencv3_mean_shift_tracking_segmentation.php)

Machine Learning : Clustering - K-Means clustering I (/python/OpenCV_Python/python_opencv3_Machine_Learning_Clustering_K-Means_Clustering_Vector_Quantization.php)

Machine Learning : Clustering - K-Means clustering II (/python/OpenCV_Python/python_opencv3_Machine_Learning_Clustering_K-Means_Clustering_Vector_Quantization_2.php)

Machine Learning : Classification - k-nearest neighbors (k-NN) algorithm (/python/OpenCV_Python/python_opencv3_Machine_Learning_Classification_K-nearest_neighbors_k-NN.php)

MongoDB with PyMongo I - Installing MongoDB ... (/python/MongoDB_PyMongo/py

Python HTTP Web Services - urllib, httplib2 (/python/python_http_web_servi

Web scraping with Selenium for checking domain availability (/python/python_Web_scraping_v

REST API : Http Requests for Humans with Flask (/python/python_REST_APIs_Http-Requests-for-Humans-with-Flask.php)

Blog app with Tornado (/python/Tornado/Python_Torna

Multithreading ... (/python/Multithread/python_mu

Python Network Programming I - Basic Server / Client : A Basics (/python/python_network_progra

Python Network Programming I - Basic Server / Client : B File Transfer (/python/python_network_progra

Python Network Programming II - Chat Server / Client (/python/python_network_progra

Python Network Programming III - Echo Server using socketserver network framework (/python/python_network_progra

Python Network Programming IV - Asynchronous Request Handling : ThreadingMixIn and ForkingMixIn (/python/python_network_progra

Python Interview Questions I (/python/python_interview_quest

Python Interview Questions II (/python/python_interview_quest

OpenCV 3 image and video processing with Python

OpenCV 3 with Python (/python/OpenCV_Python/python_opencv3.php)

Image - OpenCV BGR : Matplotlib RGB (/python/OpenCV_Python/python_opencv3_matplotlib_rgb_brg_image_load_display_save.php)

Basic image operations - pixel access (/python/OpenCV_Python/python_opencv3_basic_image_operations_pixel_access_image_load.php)

iPython - Signal Processing with NumPy (/python/OpenCV_Python/python_opencv3_NumPy_Arrays_Signal_Processing_iPython.php)

Signal Processing with NumPy I - FFT and DFT for sine, square waves, unitpulse, and random signal (/python/OpenCV_Python/python_opencv3_Signal_Processing_with_NumPy_Fourier_Transform_FFT_DFT.php)

Signal Processing with NumPy II - Image Fourier Transform : FFT & DFT (/python/OpenCV_Python/python_opencv3_Signal_Processing_with_NumPy_Fourier_Transform_FFT_DFT_2.php)

Inverse Fourier Transform of an Image with low pass filter: cv2.idft() (/python/OpenCV_Python/python_opencv3_Image_Fourier_Transform_FFT_DFT.php)

Image Histogram (/python/OpenCV_Python/python_opencv3_image_histogram_calcHist.php)

Video Capture and Switching colorspaces - RGB / HSV (/python/OpenCV_Python/python_opencv3_Changing_ColorSpaces_RGB_HSV_HLS.php)

Adaptive Thresholding - Otsu's clustering-based image thresholding (/python/OpenCV_Python/python_opencv3_Image_Global_Thresholding_Adaptive_Thresholding_Otsus_Binarization_Segmentations.php)

Edge Detection - Sobel and Laplacian Kernels (/python/OpenCV_Python/python_opencv3_Image_Gradient_Sobel_Laplacian_Derivatives_Edge_Detection.php)

Canny Edge Detection (/python/OpenCV_Python/python_opencv3_Image_Canny_Edge_Detection.php)

Hough Transform - Circles (/python/OpenCV_Python/python_opencv3_Image_Hough%20Circle_Transform.php)

Private attributes and private methods (/python/python_private_attribut

bits, bytes, bitstring, and constBitStream (/python/python_bits_bytes_bitst

json.dump(s) and json.load(s) (/python/python-json-dumps-loads-file-read-write.php)

Python Object Serialization - pickle and json (/python/python_serialization_pi

Python Object Serialization - yaml and json (/python/python_yaml_json_conv

Priority queue and heap queue data structure (/python/python_PriorityQueue_I

Graph data structure (/python/python_graph_data_stru

Dijkstra's shortest path algorithm (/python/python_Dijkstras_Short

Prim's spanning tree algorithm (/python/python_Prims_Spanning

Closure (/python/python_closure.php)

Functional programming in Python (/python/python_functional_prog

Remote running a local file using ssh (/python/python_ssh_remote_ru

SQLite 3 - A. Connecting to DB, create/drop table, and insert data into a table (/python/python_sqlite_connect_

SQLite 3 - B. Selecting, updating and deleting data (/python/python_sqlite_select_up

Python Interview Questions III
(/python/python_interview_quest

Python Interview Questions IV
(/python/python_interview_quest

Python Interview Questions V
(/python/python_interview_quest

Python Interview Questions VI
(/python/python_interview_quest

Python Interview Questions VII
(/python/python_interview_quest

Image processing with Python
image library Pillow
(/python/python_image_processi

Python and C++ with SIP
(/python/python_cpp_sip.php)

PyDev with Eclipse
(/python/pydev_eclipse_plugin_ir

Matplotlib
(/python/python_matplotlib.php)

Redis with Python
(/python/python_redis_with_pyth

NumPy array basics A
(/python/python_numpy_array_tu

NumPy Matrix and Linear
Algebra
(/python/python_numpy_matrix_

Pandas with NumPy and
Matplotlib
(/python/python_Pandas_NumPy

Celluar Automata
(/python/python_cellular_automa

Batch gradient descent
algorithm
(/python/python_numpy_batch_g

Longest Common Substring
Algorithm
(/python/python_longest_commo

Python Unit Test - TDD using unittest.TestCase class (/python/python_unit_testing.php

Simple tool - Google page ranking by keywords (/python/python_site_page_ranki

Google App Hello World (/python/GoogleApp/python_Goo

Google App webapp2 and WSGI (/python/GoogleApp/python_Goo

Uploading Google App Hello World (/python/GoogleApp/python_Goo

Python 2 vs Python 3 (/python/python_differences_Pyt

virtualenv and virtualenvwrapper (/python/python_virtualenv_virtu

Uploading a big file to AWS S3 using boto module (/DevOps/AWS/aws_S3_uploadin

Scheduled stopping and starting an AWS instance (/DevOps/AWS/aws_stopping_sta

Cloudera CDH5 - Scheduled stopping and starting services (/Hadoop/BigData_hadoop_CDH5

Removing Cloud Files - Rackspace API with curl and subprocess (/python/python_Rackspace_API_

Checking if a process is running/hanging and stop/run a scheduled task on Windows (/python/python-Windows-Check-if-a-Process-is-Running-Hanging-Schtasks-Run-Stop.php)

Apache Spark 1.3 with PySpark (Spark Python API) Shell (/Hadoop/BigData_hadoop_Apac

Apache Spark 1.2 Streaming (/Hadoop/BigData_hadoop_Apac

bottle 0.12.7 - Fast and simple WSGI-micro framework for small web-applications ... (/python/Bottle/Python_Bottle_Fr

Flask app with Apache WSGI on Ubuntu14/CentOS7 ... (/python/Flask/Python_Flask_Blo

Selenium WebDriver (/python/python_Selenium_WebI

Fabric - streamlining the use of SSH for application deployment (/python/Fabric/python_Fabric.pl

Ansible Quick Preview - Setting up web servers with Nginx, configure enviroments, and deploy an App (/DevOps/Ansible/Ansible_Setting

Neural Networks with backpropagation for XOR using one hidden layer (/python/python_Neural_Networ

NLP - NLTK (Natural Language Toolkit) ... (/python/NLTK/NLTK_install.php)

RabbitMQ(Message broker server) and Celery(Task queue) ... (/python/RabbitMQ_Celery/pythc

OpenCV3 and Matplotlib ... (/python/OpenCV_Python/pythor

Simple tool - Concatenating slides using FFmpeg ... (/FFMpeg/ffmpeg_fade_in_fade_c

iPython - Signal Processing with NumPy (/python/OpenCV_Python/pythor

iPython and Jupyter - Install Jupyter, iPython Notebook, drawing with Matplotlib, and publishing it to Github

(/python/IPython/IPython_Jupyte

iPython and Jupyter Notebook with Embedded D3.js (/python/IPython/iPython_Jupyte

Downloading YouTube videos using youtube-dl embedded with Python (/VideoStreaming/YouTube/youtu dl-embedding.php)

Machine Learning : scikit-learn ... (/python/scikit-learn/scikit_machine_learning_Su

Django 1.6/1.8 Web Framework ... (/python/Django/Python_Django_

CONTACT

BogoToBogo
contactus@bogotobogo.com (mailto:contactus@bogotobogo.com)

FOLLOW BOGOTOBOGO

(https://www.facebook.com/KHongSanFrancisco) (https://twitter.com/KHongTwit)

ABOUT US (/ABOUT_US.PHP)

contactus@bogotobogo.com (mailto:contactus@bogotobogo.com)

Golden Gate Ave, San Francisco, CA 94115

Golden Gate Ave, San Francisco, CA 94115