

Image Segmentation

MINI PROJECT

(2020-2021)

Internal Conflict Management System

REPORT



Institute of Engineering & Technology

Akanksha Gupta (181500050)

Supervised By: -

Mr.Piyush Vashisth

Department of Computer Engineering & Applications

Image Segmentation

Contents

1.Certificate	2
2.Abstract	3
2.Introduction	4
3.Hardware and Software Requirements	5
4. Objectives	6
5. Database page	7
6.Functional specification	8
6.Existing System	9
7.Use of Project	10
8. Implementation Details with Screenshots	11-20
22.Use case diagram for Image Segmentation	21
24.Future Scope	22
25.Progress till Date & The Remaining work	23
26.References	24

Image Segmentation

GLA UNIVERSITY INSTITUTE OF ENGINEERING AND TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled "Image segmentation" has been submitted to the Department of computer Science and engineering. GLA University Institute of Engineering and Technology for the fulfilment of the requirement for the award of the degree of Bachelor of Technology in "Computer Science and Engineering" by the following student of third year B.Tech (Computer Science and Engineering).

Student Name(with Roll no.)

Akanksha Gupta (181500050)

Project Guide-

Mr. Piyush Vashisth

Head of the Department -

Anand Singh Jalal

Image Segmentation

Abstract

Image segmentation plays an important role in a pre-processing phase of images having as objective a partition of the **image** into components or regions of interest for a more detailed analysis of one or more of these regions.

Introduction

Image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics. The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image (see edge detection). Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color , intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic(s).When applied to a stack of images, typical in medical imaging, the resulting contours after image segmentation can be used create 3D reconstructions with the help of interpolation algorithms like marching cubes.

Hardware and Software specification

Software Specification:

- Language Used : Python
- Database : SQL
- User Interface Design : Virtual studio code, Jupiter Notebook

Hardware Requirements:

- Processor : 64-bit, four-core, 2.5 GHz minimum per core
- Operating System : Windows 10,
- RAM : 8GB
- Hard disk : 1024 GB
- Display : 1280 x 768 screen resolution

Image Segmentation

Database

SQL



SQL is a standard language for storing, manipulating and retrieving data in databases.

Our SQL tutorial will teach you how to use SQL in: MySQL, SQL Server, MS Access, Oracle, Sybase, Informix, Postgres, and other database systems. SQL stands for Structured Query Language. SQL lets you access and manipulate database. SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987.

Objectives

The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries(lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image (see edge detection). Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic(s). When applied to a stack of images, typical in medical imaging, the resulting contours after image segmentation can be used to create 3D reconstructions with the help of interpolation algorithms like marching.

Image Segmentation

FUNCTIONAL SPECIFICATION

You would have probably heard about object detection and image localization. When there is a single object present in an image, we use image localization technique to draw a bounding box around that object. In the case of object detection, it provides labels along with the bounding boxes; hence we can predict the location as well as the class to which each object belongs.

Image segmentation results in more granular information about the shape of an image and thus an extension of the concept of Object Detection.

We segment i.e. divide the images into regions of different colors which helps in distinguishing an object from the other at a finer level.

Image Segmentation

Existing System

We use image segmentation so that we can group certain pixels together based on certain criteria. How the result of this grouping is used depends on the application.

I work in the field of digital pathology wherein the images are of tissue, captured using microscopes/digital slide scanners. The goal is to analyse these images and calculate certain measurements (like count, area, intensity etc) of certain biological entities (like nuclei) recorded in the image.

- (1) Locating and delineating objects of interest.
- (2) Making measurements from the delineated objects

Image Segmentation

USE OF THE PROJECT

The objective of the Image Segmentation to separate the image into two part, pull out the building outlines Given a picture of a magazine page, distinguish photos from text Given a picture of a check, find the part where they wrote the amount as a number. The most common use is probably feature detection and processing. If you know what the building outline is you can use that to recreate a 3-D model of the building and/or texture that model. If you know where the check amount is, you can pretty reliably use Optical Character Recognition to convert that image into a number.s based on some criteria. Typically we're interested in some sort of either edge or region detection.

Image Segmentation

Implementation details with Screenshots

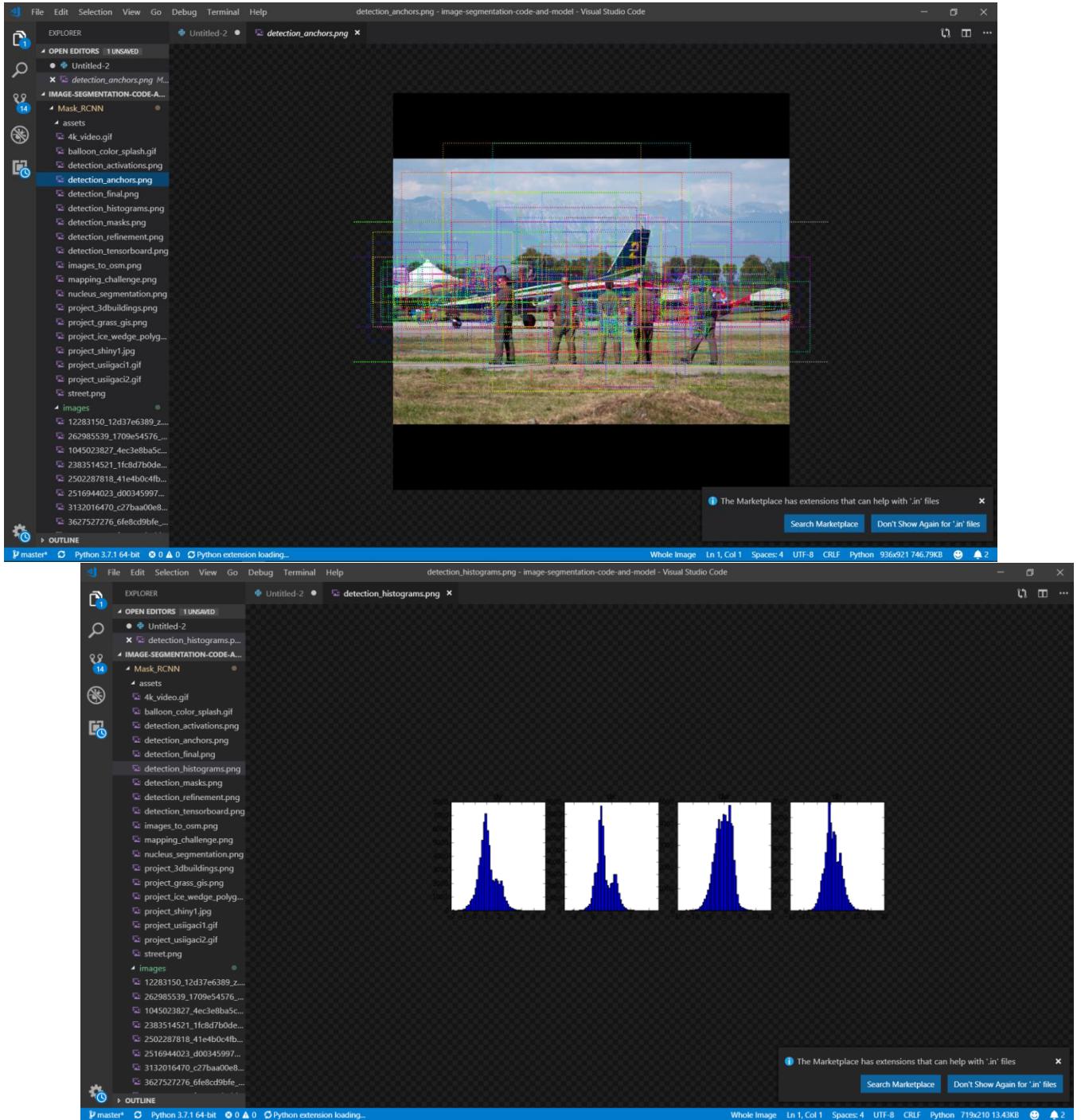


Image Segmentation

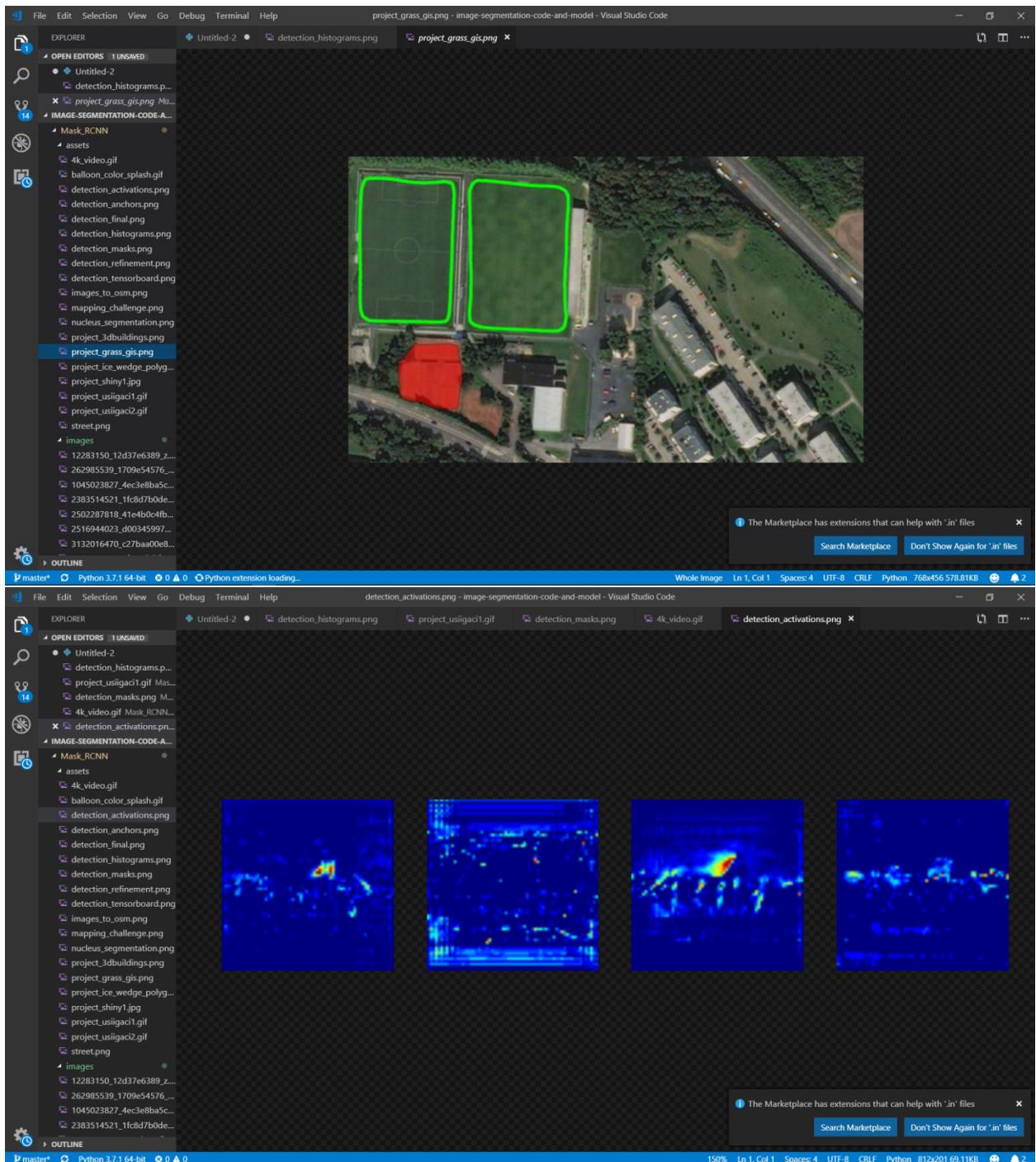


Image Segmentation

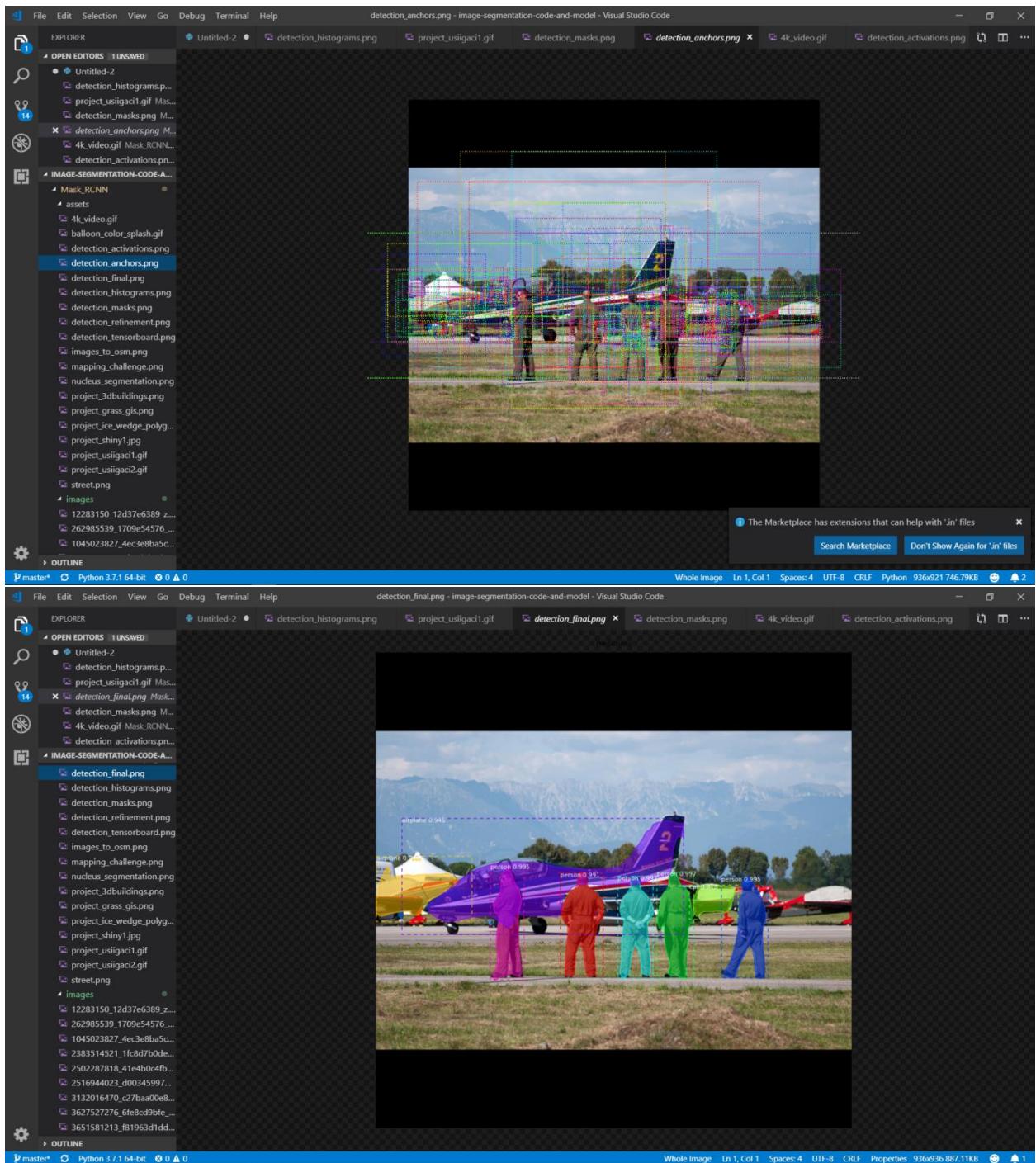


Image Segmentation

The image shows two side-by-side instances of Visual Studio Code running on a Windows operating system. Both instances have the title bar "Image segmentation.ipynb - image-segmentation-code-and-model - Visual Studio Code".

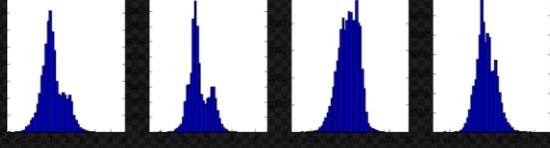
Left Instance (Top):

- Explorer:** Shows a file tree for the project. It includes files like `detection_histograms.png`, `project_usiigaci1.gif`, `demo.ipynb`, `Image segmentation.ipynb`, `detection_masks.png`, `4k_video.gif`, `Mask_RCNN.ipynb`, `detection_activations.png`, `sample.jpg`, and `samples.ipynb`. There are also several numbered files (e.g., `8734543718_3716bb0d45.ipynb`, `8829708882_4bf263491e.ipynb`, etc.).
- Editor:** Displays the content of `Image segmentation.ipynb`. The code includes imports for TensorFlow, numpy, skimage, random, math, and matplotlib, along with logic to handle project paths and visualization.

Right Instance (Bottom):

- Explorer:** Shows a file tree for the project. It includes files like `detection_histograms.png`, `project_usiigaci1.gif`, `demo.ipynb`, `Untitled-3`, `detection_masks.png`, `4k_video.gif`, `Mask_RCNN.ipynb`, `detection_activations.png`, and `detection_final.png`.
- Editor:** Displays the content of `Untitled-3`. The code includes imports for os, sys, random, math, numpy, skimage, matplotlib, and matplotlib.pyplot. It sets the working directory to the project root and imports Mask RCNN and COCO libraries.

Image Segmentation



The screenshot shows two instances of Visual Studio Code. The top instance displays four histograms side-by-side, representing the distribution of detected objects. The bottom instance shows the Python code for image segmentation, specifically the Mask R-CNN project. The code imports necessary libraries like os, sys, random, math, numpy, skimage.io, and matplotlib.pyplot. It sets the working directory to the project root and imports the Mask R-CNN library. The code then defines a function to visualize detections on images. The code editor shows syntax highlighting for Python and includes a Jupyter notebook interface.

```
%>% Change working directory from the workspace root to the ipynb file location. Turn this addition off with the DataScience.changeDirOnImportExport setting in your settings.json.
1 import os
2 try:
3     os.chdir(os.path.join(os.getcwd(), 'Mask_RCNN\samples'))
4     print(os.getcwd())
5 except:
6     pass
7
8 Run Cell | Run Above | Run Below
9 %>%
10 import os
11 import sys
12 import random
13 import math
14 import numpy as np
15 import skimage.io
16 import matplotlib
17 import matplotlib.pyplot as plt
18
19 # Root directory of the project
20 ROOT_DIR = os.path.abspath("../")
21
22 import warnings
23 warnings.filterwarnings("ignore")
24
25 # Import Mask RCNN
26 sys.path.append(ROOT_DIR) # To find local version of the library
27 from mrcnn import utils
28 import mrcnn.model as modellib
29 from mrcnn import visualize
30
31 # Import COCO config
32 sys.path.append(os.path.join(ROOT_DIR, "samples/coco/")) # To find local version
33 import coco
34
35 get_ipython().run_line_magic('matplotlib', 'inline')
36
37 %>%
38 print(ROOT_DIR)
```

Image Segmentation

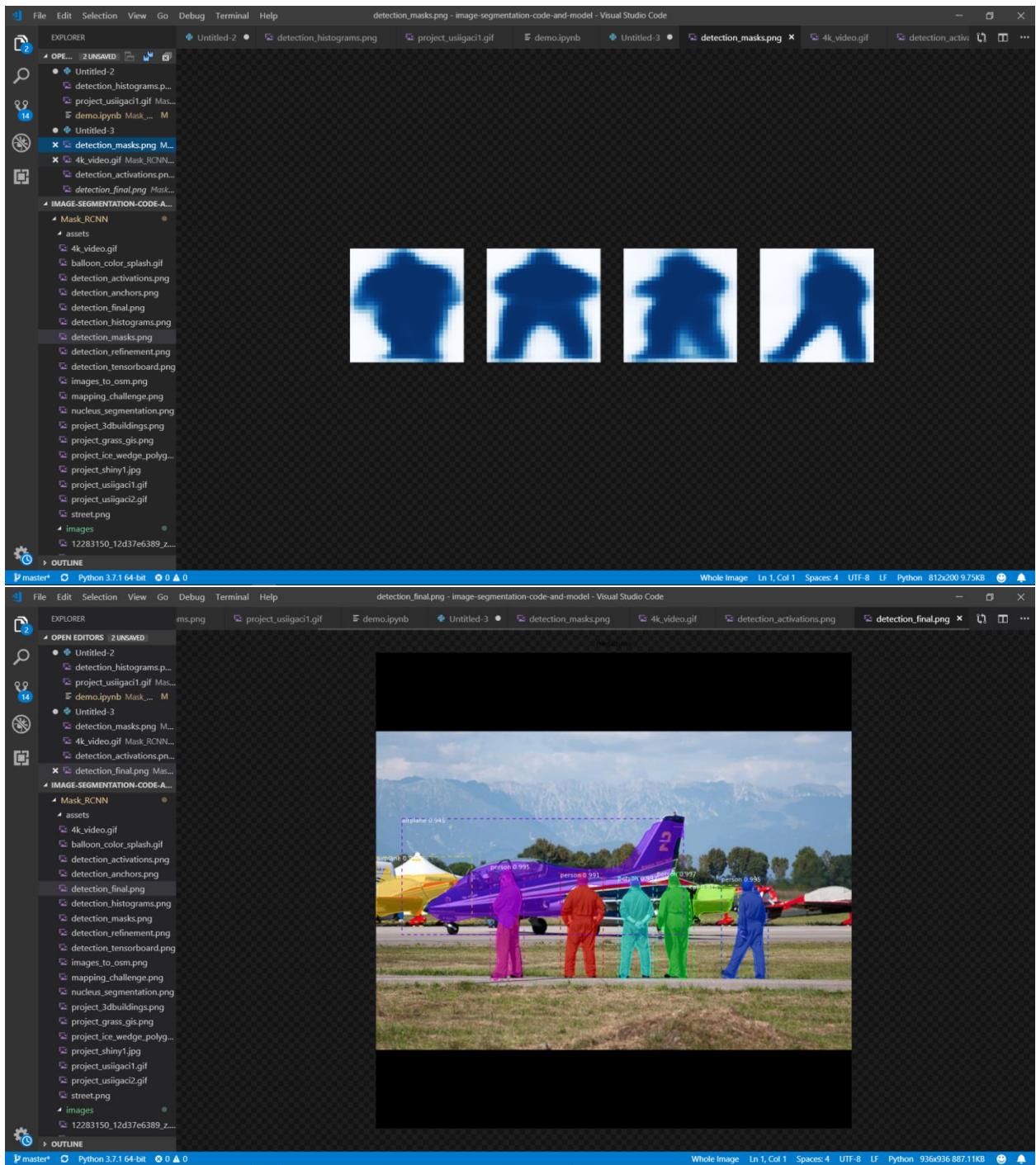


Image Segmentation

The screenshot shows a dual-monitor setup. The left monitor displays a Jupyter Notebook interface in Visual Studio Code. The notebook contains Python code for image segmentation using Mask R-CNN. The code includes imports for tensorflow, numpy, and skimage, as well as a function named `detect_and_segment`. The right monitor shows the terminal output, which includes the execution of the notebook and the generation of files such as `detection_masks.png` and `4k_video.gif`.

```
File Edi Selection View Go Debug Terminal Help demo-checkpoint.ipynb - image-segmentation-code-and-model - Visual Studio Code
EXPLORER OPEN EDITORS 2 UNSAVED
demo.ipynb Untitled-3 detection_masks.png 4k_video.gif detection_activations.png detection_final.png demo-checkpoint.ipynb x
detection_histograms.p... project_usigac1.gif Mas... demo.ipynb Mask_... M Untitled-3 detection_masks.png M 4k_video.gif Mask_RCNN... detection_activations.pn... detection_final.png Mas... demo-checkpoint.ipynb ...
IMAGE-SEGMENTATION-CODE-A...
Mask_RCNN assets images samples .ipynb_checkpoints demo-checkpoint.ipynb
Image segmentation-ch... balloon nucleus inspect_nucleus_data.ip... inspect_nucleus_modeli... nucleus.py README.md shapes demo.ipynb M Image segmentation.... U mask_rcnn_coco.h5 gignore LICENSE MANIFEST.in README.md requirements.txt setup.cfg
OUTLINE master* Python 3.7.1 64-bit 0 0 0 0
File Edi Selection View Go Debug Terminal Help Image segmentation-checkpoint.ipynb - image-segmentation-code-and-model - Visual Studio Code
EXPLORER OPEN EDITORS 2 UNSAVED
demo.ipynb Untitled-3 detection_masks.png 4k_video.gif detection_activations.png detection_final.png Image segmentation-checkpoint.ipynb x
detection_histograms.p... project_usigac1.gif Mas... demo.ipynb Mask_... M Untitled-3 detection_masks.png M 4k_video.gif Mask_RCNN... detection_activations.pn... detection_final.png Mas... Image segmentation-ch... ...
IMAGE-SEGMENTATION-CODE-A...
Mask_RCNN assets images samples .ipynb_checkpoints demo-checkpoint.ipynb
Image segmentation-ch... balloon nucleus inspect_nucleus_data.ip... inspect_nucleus_modeli... nucleus.py README.md shapes demo.ipynb M Image segmentation.... U mask_rcnn_coco.h5 gignore LICENSE MANIFEST.in README.md requirements.txt setup.cfg
OUTLINE master* Python 3.7.1 64-bit 0 0 0 0
```

Image Segmentation

The screenshot shows two instances of Visual Studio Code side-by-side. Both instances have the title bar "nucleus.py - image-segmentation-code-and-model - Visual Studio Code".

Left Instance (nucleus.py):

```
31 # rather than being imported.
32 if __name__ == "__main__":
33     import matplotlib
34     # Agg backend runs without a display
35     matplotlib.use('Agg')
36     import matplotlib.pyplot as plt
37
38     import os
39     import sys
40     import json
41     import datetime
42     import numpy as np
43     import skimage.io
44     from imgaug import augmenters as iaa
45
46     # Root directory of the project
47     ROOT_DIR = os.path.abspath("../..")
48
49     # Import Mask RCNN
50     sys.path.append(ROOT_DIR) # To find local version of the library
51     from mrcnn.config import Config
52     from mrcnn import utils
53     from mrcnn import model as modellib
54     from mrcnn import visualize
55
56     # Path to trained weights file
57     COCO_WEIGHTS_PATH = os.path.join(ROOT_DIR, "mask_rcnn_coco.h5")
58
59     # Directory to save logs and model checkpoints, if not provided
60     # through the command line argument --logs
61     DEFAULT_LOGS_DIR = os.path.join(ROOT_DIR, "logs")
62
63     # Results directory
64     # Save submission files here
65     RESULTS_DIR = os.path.join(ROOT_DIR, "results/nucleus/")
66
67     # The dataset doesn't have a standard train/val split, so I picked
68     # a variety of images to serve as a validation set.
69     VAL_IMAGE_IDS = [
70         "0c255ba23ba0af29a7575de8c61690d3c31bc897dd5ba66caeac201d201a278c2",
71         "92f31f591929a30e4309ab75185c96ff4314ce0a7ead2ed2c2171897ad1da0c7",
```

Right Instance (README.md):

```
1 # Nuclei Counting and Segmentation
2
3 This sample implements the [2018 Data Science Bowl challenge](https://www.kaggle.com/c/data-science-bowl-2018).
4 The goal is to segment individual nuclei in microscopy images.
5 The nucleus.py file contains the main parts of the code, and the two Jupyter notebooks
6
7
8 ## Command line Usage
9 Train a new model starting from ImageNet weights using 'train' dataset (which is 'stage1_train' minus validation set)
10 ...
11 python3 nucleus.py train --dataset=/path/to/dataset --subset=train --weights=imagenet
12
13
14 Train a new model starting from specific weights file using the full 'stage1_train' dataset
15 ...
16 python3 nucleus.py train --dataset=/path/to/dataset --subset=stage1_train --weights=/path/to/weights.h5
17
18
19 Resume training a model that you had trained earlier
20 ...
21 python3 nucleus.py train --dataset=/path/to/dataset --subset=train --weights=last
22
23
24 Generate submission file from 'stage1_test' images
25 ...
26 python3 nucleus.py detect --dataset=/path/to/dataset --subset=stage1_test --weights=<last or /path/to/weights.h5>
27
28
29
30 ## Jupyter notebooks
31 Two Jupyter notebooks are provided as well: 'inspect_nucleus_data.ipynb' and 'inspect_nucleus_model.ipynb'.
32 They explore the dataset, run stats on it, and go through the detection process step by step.
```

Image Segmentation

The screenshot displays two instances of Visual Studio Code side-by-side, both showing the same project structure and files.

Top Window (requirements.txt - image-segmentation-code-and-model - Visual Studio Code):

- OPEN EDITORS:** 2 UNSAVED
- File Explorer:** Shows various files and folders including `detection_masks.png`, `4k_video.gif`, `detection_activations.png`, `detection_final.png`, `train_shapes.ipynb`, and `requirements.txt`.
- Code Editor:** Displays the contents of `requirements.txt`:

```
1 numpy
2 scipy
3 Pillow
4 cython
5 matplotlib
6 scikit-image
7 tensorflow>1.3.0
8 keras>>2.0.8
9 opencv-python
10 h5py
11 imgaug
12 IPython[all]
```

- Status Bar:** Shows "master" and "Python 3.7.1 64-bit".

Bottom Window (setup.cfg - image-segmentation-code-and-model - Visual Studio Code):

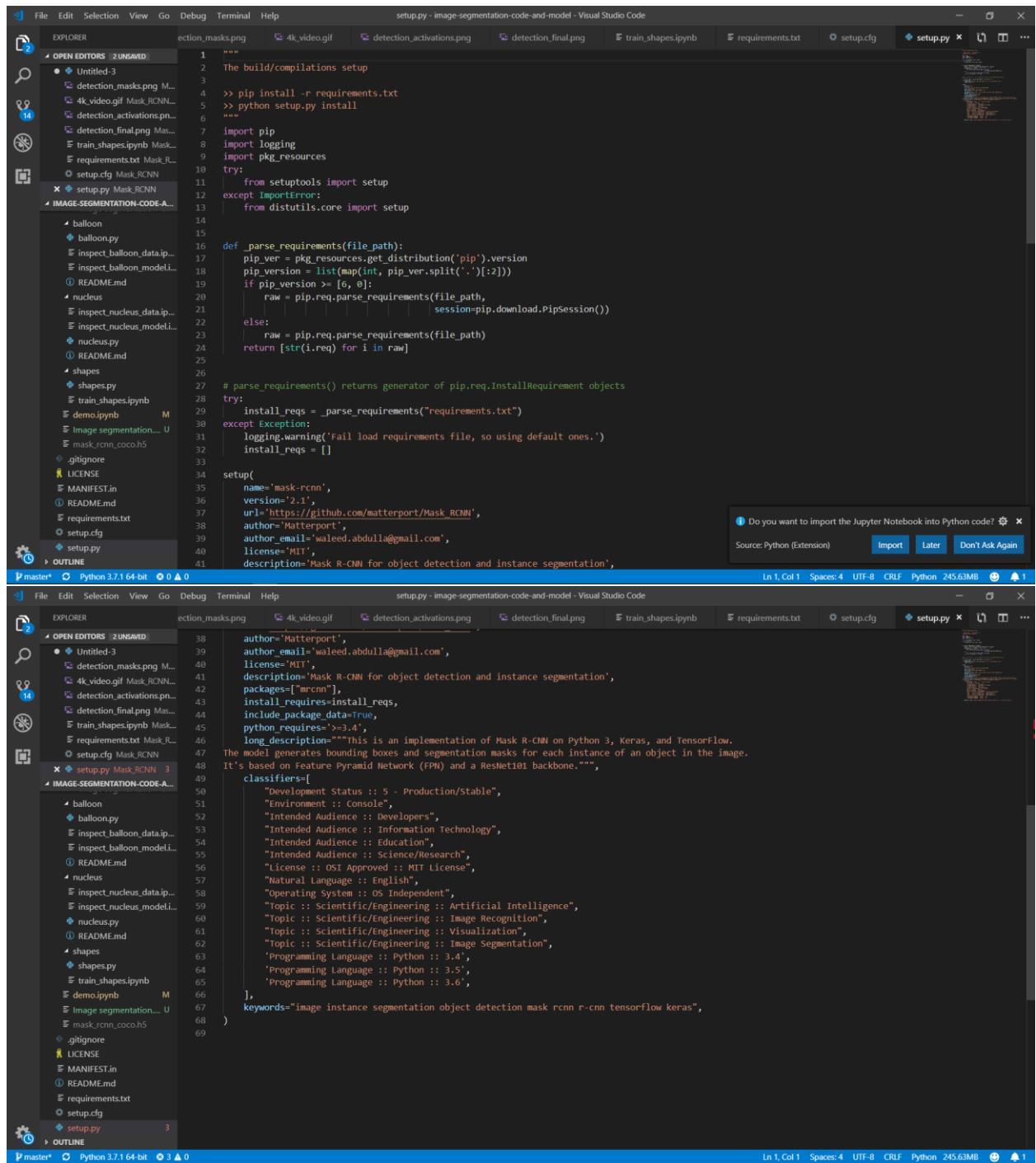
- OPEN EDITORS:** 2 UNSAVED
- File Explorer:** Shows the same files and folders as the top window.
- Code Editor:** Displays the contents of `setup.cfg`:

```
[metadata]
description-file = README.md
license-file = LICENSE
requirements-file = requirements.txt
```

- Status Bar:** Shows "master" and "Python 3.7.1 64-bit".

In both windows, there is a message bar at the bottom right asking if you want to import Jupyter Notebooks into Python code, with options "Import", "Later", and "Don't Ask Again".

Image Segmentation



The image shows two side-by-side instances of the Visual Studio Code editor. Both instances have the same title bar: "setup.py - image-segmentation-code-and-model - Visual Studio Code". The left instance shows lines 1 through 41 of the setup.py script. The right instance shows lines 38 through 69 of the same script. The code is written in Python and defines a setup function for a Mask R-CNN project, including details like author, version, URL, and long description.

```
1 """
2 The build/compilations setup
3
4 >> pip install -r requirements.txt
5 >> python setup.py install
6 """
7 import pip
8 import logging
9 import pkg_resources
10 try:
11     from setuptools import setup
12 except ImportError:
13     from distutils.core import setup
14
15
16 def _parse_requirements(file_path):
17     pip_ver = pkg_resources.get_distribution('pip').version
18     pip_version = list(map(int, pip_ver.split('.')[2]))
19     if pip_version >= [6, 0]:
20         raw = pip.req.parse_requirements(file_path,
21                                         session=pip.download.PipSession())
22     else:
23         raw = pip.req.parse_requirements(file_path)
24     return [str(i.req) for i in raw]
25
26
27 # parse_requirements() returns generator of pip.req.InstallRequirement objects
28 try:
29     install_reqs = _parse_requirements("requirements.txt")
30 except Exception:
31     logging.warning('Fail load requirements file, so using default ones.')
32     install_reqs = []
33
34 setup(
35     name='mask-r-cnn',
36     version='2.3',
37     url='https://github.com/matterport/Mask_RCNN',
38     author='Matterport',
39     author_email='waleed.abdulla@gmail.com',
40     license='MIT',
41     description='Mask R-CNN for object detection and instance segmentation',
42     classifiers=[
43         "Development Status :: 5 - Production/Stable",
44         "Environment :: Console",
45         "Intended Audience :: Developers",
46         "Intended Audience :: Information Technology",
47         "Intended Audience :: Education",
48         "Intended Audience :: Science/Research",
49         "License :: OSI Approved :: MIT License",
50         "Natural Language :: English",
51         "Operating System :: OS Independent",
52         "Topic :: Scientific/Engineering :: Artificial Intelligence",
53         "Topic :: Scientific/Engineering :: Image Recognition",
54         "Topic :: Scientific/Engineering :: Visualization",
55         "Topic :: Scientific/Engineering :: Image Segmentation",
56         "Programming Language :: Python :: 3.4",
57         "Programming Language :: Python :: 3.5",
58         "Programming Language :: Python :: 3.6",
59     ],
60     keywords="image instance segmentation object detection mask r-cnn tensorflow keras",
61 )
62
63
64
65
66
67
68
69 )
```

Image Segmentation

Use Case Diagram of Image Segmentation :

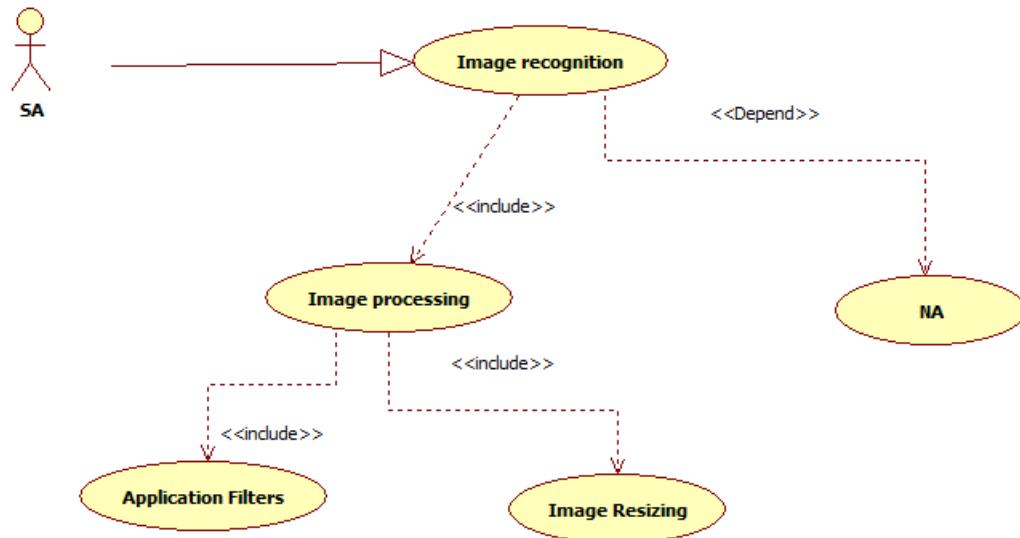
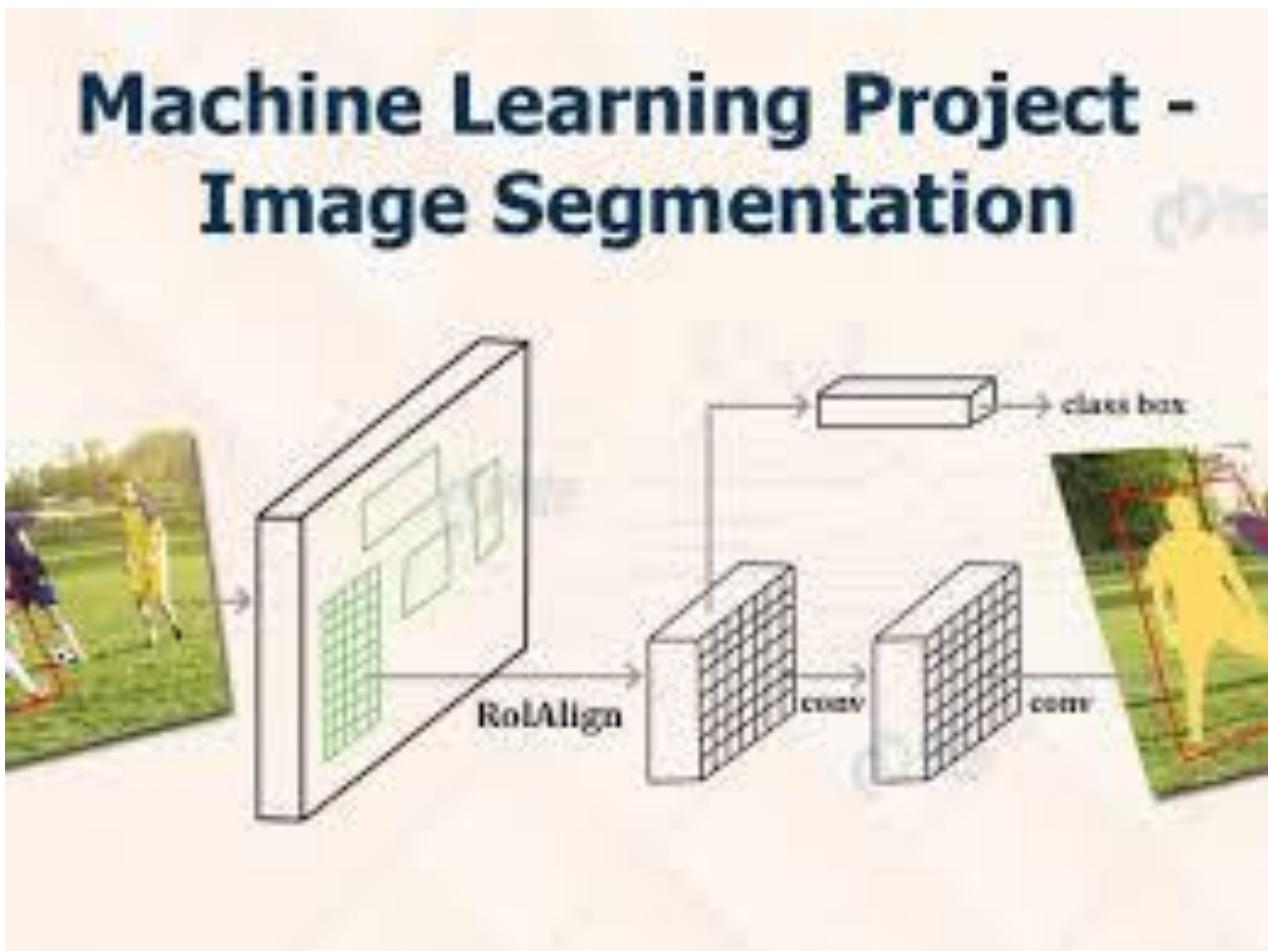


Image Segmentation



FUTURE SCOPE

The future of image processing will involve scanning the heavens for other intelligent life out in space. Also new intelligent, digital species created entirely by research scientists in various nations of the world will include advances in image processing applications. Due to advances in image processing and related technologies there will be millions and millions of robots in the world in a few decades time, transforming the way the world is managed. The future image processing applications of satellite based imaging ranges from planetary exploration to surveillance applications.

Image Segmentation

Progress till Date & The Remaining work

100% of the work is completed up to date and for the further work history the above screenshots are shown.

References

- 3D Entropy Based Image Segmentation
- *Frucci, Maria; Sanniti di Baja, Gabriella (2008). "From Segmentation to Binarization of Gray-level Images". Journal of Pattern Recognition Research.*

Department of Computer Engineering & Applications