SHAPE DETECTION OF POLYGON SET

Submitted in partial fulfillment of the requirements for

Data Science Course

By

VANNADA JYOTHSNA	**16241A05N5**
GALI ANUSHA	**16241A05J4**
SHAIK GULZAR	**16241A05M9**

Under the Esteemed guidance of

V.Sri lakshmi

Asst.Professor

Department of Computer Science and Engineering

GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Approved by AICTE, Autonomous under JNTUH, Hyderabad)

Bachupally, Kukatpally, Hyderabad-500090.

2018-2019

Shape Detection of Polygon Set

GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Approved by AICTE, Autonomous under JNTUH, Hyderabad)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project entitled **"Shape Detection of polygon set"** is submitted

by

VANNADA JYOTHSNA	**16241A05N5**
G.ANUSHA	**16241A05J4**
SHAIK GULZAR	**16241A05M9**

In partial fulfillment of Data Science Course in Department of Computer Science and Engineering during academic year 2018-2019 .

INTERNAL GUIDE
Asst. Prof. P. Sri Lakshmi

COORDINATOR
Dr.G.Karuna

HEAD OF THE DEPARTMENT
Dr. Ch. Mallikarjuna Rao

ACKNOWLEDGEMENT

There are many people who helped us directly and indirectly to complete our project successfully. We would like to take this opportunity to thank one and all. First we would like to express my deep gratitude towards my internal guide **V.Sri Lakshmi, Asst. Professor**, Department of CSE for her support in the completion of our dissertation. We wish to express our sincere thanks to **Dr.G.Karuna**, Coordinator for Data Science Course, **Dr.Ch. Mallikarjuna Rao,** HOD, Department of CSE, **Xenonstack** team, Chandigarh.

We would like to extend my thanks to **Dr.Jandhyala N Murthy**, Director and **Dr. J Praveen,** principal for providing the facilities to complete the course. We would like to thank all our faculty and friends for their help and constructive criticism during the project period. Finally we are very much indebted to our parents for their moral support and encouragement to achieve our goals.

VANNADA JYOTHSNA
(16241A05N5)
GALI ANUSHA
(16241A05J4)
SHAIK GULZAR
(16241A05M9)

Shape Detection of Polygon Set

DECLARATION

I hereby declare that the project entitled **"Shape Detection of polygon set"** is the work done during the period from **23rd July 2018 to 15th October 2018** and is submitted in the partial fulfillment of the requirements for Data Science Course in the department of Computer Science and Engineering , Gokaraju Rangaraju Institute of Engineering and Technology

VANNADA JYOTHSNA
(16241A05N5)
GALI ANUSHA
(16241A05J4)
SHAIK GULZAR
(16241A05M9)

Shape Detection of Polygon Set

ABSTRACT

The project taken is **Shape detection of Polygons upto 4 Edges.**

This project is used for detecting the line ,triangle ,square ,rhombus, Rectangle .Here we use the OpenCV dataset module to load a pre trained object detection network .This will enable us to pass the input images through the network and obtain the output bounding box(x , y)- coordinates of each object in the image.

Feature Extraction and Learning algorithms are used to recognise the objects .Here an image of the above given polygons is passed to the detector where in it detects the number of pixels until the polygons are detected based on their shapes .Let us consider n to be number of edges detected .Here we use the conditions such as if n=3 that is if the number of edges is three then it is a polygon of shape "Triangle" .Similarly if n=0 then it is a "circle" and if n=2 it is a "Line" and if n=4 then it may be a "square" or a "rectangle" .Here when n=4 we identify the image to be a square or Rectangle based on the aspect ratio which is the ratio between the length and the width.

Shape Detection of Polygon Set

CONTENTS:

1. INTRODUCTION

1.1 SHAPE DETECTION:

Doing image processing and especially blob analysis it is often required to check some objects' shape and depending on it perform further processing of a particular object or not. For example, some applications may require finding only circles from all the detected objects, or quadrilaterals, rectangles, etc.

Human vision seems to make use of many sources of information to detect and recognize an object in a scene .At the lowest level of object recognition, researchers agree that edge and region information are utilized to extract a “perceptual unit” in the scene. Some of the possible invariant features are recognized and additional signal properties (texture or appearance) are sent to help in making the decision as to whether a point belongs to an object or not.

In many cases, boundary shape information, such as the rectangular shapes of vehicles in aerial imagery, seems to play a crucial role. Local features such as the eyes in a human face are sometimes useful. These features provide strong clues for recognition, and often they are invariant to many scene variables.

The study of shapes is a recurring theme in computer vision. For example, shape is one of the main sources of information that can be used for object recognition. In medical image analysis, geometrical models of anatomical structures play an important role in automatic tissue segmentation. The shape of an organ can also be used to diagnose diseases. In a completely different setting, shape plays an important role in the perception of optical illusions

(we tend to see particular shapes) and this can be used to explain how our visual system interprets the ambiguous and incomplete information available in an image.

Characterizing the shape of a specific rigid object is not a particularly hard problem, although using the shape information to solve perceptual tasks is not easy.

Content based image retrieval is one of the topics of interest in the computer vision field which nowadays is at its very peak, due to the growth in the last years of the amount of stored graphical information. For this kind of data, underlying analysis processes mainly lie on graphics recognition, allowing then classification of the images, typically in terms of available symbols. From a general viewpoint, several kinds of recognition approaches can be involved, according to data representation.

Bitmap images are usually analyzed with statistical methods, which are time-consuming and quite accurate, but can also be analyzed with structural methods, faster but requiring a pre-vectorization step. In the context of content based image retrieval, the last approach is usually preferred, as the amount of considered data implies the use of efficient processes. One of the most important visual features when classifying images is shape of the represented objects and subsequently a lot of literature deal with object recognition by shape. In the science of image processing and the science of object detection the operation is always trying to detect an object and it requires to knew the all kind of shapes and objects in order to detect them and there are some software are looking for only triangle or only the circle in the whole photo given as an input to the system. In the eye of humans also need to detect the shape which is in front them from the database which stored in the brain of humans and they need to

recognize the shape before doing any other operation or step. Researchers of detecting objects are agreeing that the information of area in a photo are important to extract the thing which called perceptual unit. Here some features could be detected and also some texture as properties of signal might be sent in order to assist the system when taking a decision to any point detected if does it related to any shape or does not related to it. In the photos which have been taken randomly contain some shapes like rectangle could play a very important role as information to the shape itself which maybe a vehicle in this case. Other features of shapes just like the shape of human eye could be useful also because this kind of feature could help in detecting the human face because it has the same shape in all faces where other shapes could be related to any other object. In computer science, many researchers are focusing on shape detection because the shape is playing important role to recognize an object. Even in medical science, detecting shape is an important step to specify the diseases especially in analyzing image for example the shape of organ maybe useful in specifying the diseases. Usually we are looking for known and normal shapes but sometimes if the shape is incomplete or the shape is looks like another shape, here we could not recognize which shape is it, so the information of the shape are important due to our vision system needs the complete information of the shape in order to recognize it and that is why we need the complete information of the shape. In computer science nowadays, retrieving photos became one of the most things of the interesting of computer vision especially because of the huge number of graphical data which have been stored last years. In these types of information and data the process of analysis depends on the operation of recognizing the graphics in order to let the photo be classified especially when the symbols are available.

Generally, there are many types of methods of detection and recognition depending on the kind of information. The approaches of statistic can be used for analyzing the photos and they are really accurate but took long time where if we try the structural approaches we will gain more time because it will be faster than the statistical ones but here we must apply them very carefully to make them accurate. In object recognition, detecting the shape of object and recognize it is very important because it will help the system to recognize the object itself in the given photo.

1.2 OPENCV:

OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

Adopted all around the world, OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 14 million. Usage ranges from interactive art, to mines inspection, stitching maps on the web or through advanced robotics.

1.2.1 HISTORY:

Officially launched in 1999, the OpenCV project was initially an Intel Research initiative to advance CPU-intensive applications,

part of a series of projects including real-time ray tracing and 3D display walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel's Performance Library Team. In the early days of OpenCV ,the goals of the project were described as:

Advance vision research by providing not only open but also optimized code for basic vision infrastructure. No more reinventing the wheel. Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable. Advance vision-based commercial applications by making portable, performance-optimized code available for free – with a license that did not require code to be open or free itself.

The first alpha version of OpenCV was released to the public at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. A version 1.1 "pre-release" was released in October 2008.

The second major release of the OpenCV was in October 2009. OpenCV 2 includes major changes to the C++ interface, aiming at easier, more type-safe patterns, new functions, and better implementations for existing ones in terms of performance (especially on multi-core systems). Official releases now occur every six months and development is now done by an independent Russian team supported by commercial corporations.

In August 2012, support for OpenCV was taken over by a non-profit foundation OpenCV.org, which maintains a developer and user site.On May 2016, Intel signed an agreement to acquire Itseez, the leading developer of OpenCV.

1.2.2 APPLICATONS:

OpenFrameworks running the OpenCV is an add-on example.

OpenCV's application areas include:

- 2D and 3D feature toolkits
- Egomotion estimation
- Facial recognition system
- Gesture recognition
- Human–computer interaction (HCI)
  - Mobile robotics
  - Motion understanding
- Object identification
- Segmentation and recognition
- Stereopsis stereo vision: depth perception from 2 cameras
- Structure from motion (SFM)
- Motion tracking
- Augmented reality

To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

- Boosting

- Decision tree learning
- Gradient boosting trees
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)
- Deep neural networks (DNN)

1.2.3 PROGRAMMING LANGUAGES:

OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation. Wrappers in other languages such as C#, Perl, Ch, Haskell and Ruby have been developed to encourage adoption by a wider audience.

All of the new developments and algorithms in OpenCV are now developed in the C++ interface.

1.2.4 HARDWARE ACCELERATION:

If the library finds Intel's Integrated Performance Primitives on the system, it will use these proprietary optimized routines to accelerate itself.

A CUDA-based GPU interface has been in progress since September 2010.

An OpenCL-based GPU interface has been in progress since October 2012, documentation for version 2.4.13.3 can be found at docs.opencv.org.

1.2.5 OS SUPPORT:

OpenCV runs on the following desktop operating systems: Windows, Linux, macOS, FreeBSD, NetBSD, OpenBSD. OpenCV runs on the following mobile operating systems: Android, iOS, Maemo, BlackBerry 10. The user can get official releases from SourceForge or take the latest sources from GitHub .OpenCV uses CMake.

1.3 GUI FEATURES IN OPENCV:

Use the function cv2.imread() to read an image. The image should be in the working directory or a full path of image should be given.

Second argument is a flag which specifies the way image should be read.

- cv2.IMREAD_COLOR : Loads a color image. Any transparency of image will be neglected. It is the default flag.
- cv2.IMREAD_GRAYSCALE : Loads image in grayscale mode.
- cv2.IMREAD_UNCHANGED : Loads image as such including alpha channel.

- cv2.waitKey(): is a keyboard binding function. Its argument is the time in milliseconds. The function waits for specified milliseconds for any keyboard event. If you press any key in that time, the program continues. If 0 is passed, it waits indefinitely for a key stroke. It can also be set to detect specific key strokes like, if key a is pressed etc which we will discuss below.

- cv2.destroyAllWindows(): simply destroys all the windows we created. If you want to destroy any specific window, use the function cv2.destroyWindow() where you pass the exact window name as the argument.

1.4 ADVANTAGES OF OPENCV:

- First and foremost, OpenCV is available free of cost
- Since OpenCV library is written in C/C++ it is quite fast
- Low RAM usage (approx 60–70 mb)
- It is portable as OpenCV can run on any device that can run C

1.5 DISADVANTAGES OF OPENCV:

- OpenCV does not provide the same ease of use when compared to MATLAB
- OpenCV has a flann library of its own. This causes conflict issues when you try to use OpenCV library with the PCL library

2. SYSTEM REQUIREMENTS

2.1 SOFTWARE REQUIREMENTS:

Supported operating systems:

•Windows 7(32 or 64 bit), Windows 8(32 or 64 bit), Windows 10(32 or 64 bit)
•Linux (Ubuntu Linux)

Supported Development environments:

. jupyter notebook
. python

2.2 HARDWARE REQUIREMENTS:

- Processor: 1.2 GHZ
- RAM: 8 GB
- ROM: 300 GB

3. TECHNOLOGY

3.1 PYTHON:

Guido van Rossum created the Python programming language in the late 1980s. In contrast to other popular languages such as C, C++, Java, and C#, Python strives to provide a simple but powerful syntax.
Python is used for software development at companies and organizations such as Google, Yahoo, CERN, Industrial Light and Magic, and NASA. Experienced programmers can accomplish great things with Python, but Python's beauty is that it is accessible to beginning programmers and allows them to tackle interesting problems more quickly than many other, more complex languages that have a steeper learning curve.

3.1.1 IDLE's INTERACTIVE SHELL:

IDLE is a simple Python integrated development environment available for Windows, Linux, and Mac OS X. In IDLE interactive shell you may type any line of Python program directly into IDLE and press enter to execute the program. Result is shown in the IDLE interactive shell.
Since it does not provide a way to save the code you enter, the interactive shell is not the best tool for writing larger programs. The IDLE interactive shell is useful for experimenting with small snippets of Python code

3.1.2 IDLE's EDITOR:

IDLE has a built in editor. From the IDLE menu, select New Window, Type the text into the editor. We can see the result in editor window with the text of the simple Python program. You can

save your program using the Save option. Save the code to a file named simple.py. The actual name of the file is irrelevant, but the name “simple” accurately describes the nature of this program. The extension .py is the extension used for Python source code. We can run the program from within the IDLE editor by pressing the F5 function key or from the editor’s Run menu: Run→Run Module. The output appears in the IDLE interactive shell window.

The editor allows us to save our programs and conveniently make changes to them later. The editor understands the syntax of the Python language and uses different colors to highlight the various components that comprise a program. Much of the work of program development occurs in the editor.

3.1.3BUILT IN FUNCTIONS IN PYTHON

abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	Slice()
ascii()	divmode()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	ininstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tupple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod(	getattr()	locals()	repr()	zip()

)				
compile()	globals()	map()	reversed()	import()
complex()	hasattr()	max()	round()	

3.1.4 BUILT IN CONSTANTS :

A small number of constants live in the built-in namespace. They are:

FALSE:

The false value of the boolean type .Assignments to false are illegal and raise a syntax error.

TRUE:

The true value of the boolean type .Assignments to true are illegal and raise a syntax error.

NONE:

The soul value of the type none type. None is frequently used to represent the absence of a value ,as when default arguments are not passed to a function .Assignments to none are legal and raise a syntax error.

NOTIMPLEMENTED:

Special value which should be returned by the binary special methods to indicate that the operation is not implemented with respect to the type; may be returned by the in-place binary special methods for the same purpose .Its truth value is true .

ELLIPSES:

The same as special value used mostly in conjunction with extended slicing syntax for user defined containing data types.

DEBUG:

This constant is true if python was not started with an -o option

CONSTANT ADDED BY THE SITE MODULE:

The site module (which is imported automatically during start up except if the -s command -line option is given) adds several constants to the built-in namespace..They are useful for the interactive interpreter shell and should not be use in programs.

quit(code=None)
exit(code=None)
objects that when printed ,print a message like "Use quit() or Ctrl+D (i.e EOF) to exit", and when called ,raise SystemExit with the specified exit code.

3.1.5 APPLICATIONS OF PYTHON:

1) Readable and Maintainable Code

While writing a software application, you must focus on the quality of its source code to simplify maintenance and updates. The syntax rules of Python allow you to express concepts without writing additional code. At the same time, Python, unlike other programming languages, emphasizes on code readability, and allows you to use English keywords instead of punctuations. Hence, you can use Python to build custom applications without writing additional code. The readable and clean code base will help you to maintain and update the software without putting extra time and effort.

2) Multiple Programming Paradigms

Like other modern programming languages, Python also supports several programming paradigm. It supports object oriented and structured programming fully. Also, its language features support various concepts in functional and aspect-oriented programming. At the same time, Python also features a dynamic

type system and automatic memory management. The programming paradigms and language features help you to use Python for developing large and complex software applications.

3) Compatible with Major Platforms and Systems

At present, Python is supports many operating systems. You can even use Python interpreters to run the code on specific platforms and tools. Also, Python is an interpreted programming language. It allows you to you to run the same code on multiple platforms without recompilation. Hence, you are not required to recompile the code after making any alteration. You can run the modified application code without recompiling and check the impact of changes made to the code immediately. The feature makes it easier for you to make changes to the code without increasing development time.

4) Robust Standard Library

Its large and robust standard library makes Python score over other programming languages. The standard library allows you to choose from a wide range of modules according to your precise needs. Each module further enables you to add functionality to the Python application without writing additional code. For instance, while writing a web application in Python, you can use specific modules to implement web services, perform string operations, manage operating system interface or work with internet protocols. You can even gather information about various modules by browsing through the Python Standard Library documentation.

5) Many Open Source Frameworks and Tools

As an open source programming language, Python helps you to

curtail software development cost significantly. You can even use several open source Python frameworks, libraries and development tools to curtail development time without increasing development cost. You even have option to choose from a wide range of open source Python frameworks and development tools according to your precise needs. For instance, you can simplify and speedup web application development by using robust Python web frameworks like Django, Flask, Pyramid, Bottle and Cherrypy. Likewise, you can accelerate desktop GUI application development using Python GUI frameworks and toolkits like PyQT, PyJs, PyGUI, Kivy, PyGTK and WxPython.

6) Simplify Complex Software Development

Python is a general purpose programming language. Hence, you can use the programming language for developing both desktop and web applications. Also, you can use Python for developing complex scientific and numeric applications. Python is designed with features to facilitate data analysis and visualization. You can take advantage of the data analysis features of Python to create custom big data solutions without putting extra time and effort. At the same time, the data visualization libraries and APIs provided by Python help you to visualize and present data in a more appealing and effective way. Many Python developers even use Python to accomplish artificial intelligence (AI) and natural language processing tasks.

7) Adopt Test Driven Development

You can use Python to create prototype of the software application rapidly. Also, you can build the software application directly from the prototype simply by refactoring the Python code. Python even makes it easier for you to perform coding and testing

simultaneously by adopting test driven development (TDD) approach. You can easily write the required tests before writing code and use the tests to assess the application code continuously. The tests can also be used for checking if the application meets predefined requirements based on its source code.

However, Python, like other programming languages, has its own shortcomings. It lacks some of the built-in features provided by other modern programming language. Hence, you have to use Python libraries, modules, and frameworks to accelerate custom software development. Also, several studies have shown that Python is slower than several widely used programming languages including Java and C++. You have to speed up the Python application by making changes to the application code or using custom runtime. But you can always use Python to speed up software development and simplify software maintenance.

BASIC DATATYPES:

• Integers (default for numbers)
z = 5 / 2 # Answer is 2, integer division.
• Floats
x = 3.456
• Strings
• Can use “” or ‘’ to specify.
“abc” ‘abc’ (Same thing.)
• Unmatched can occur within the string.
“matt’s”
• Use triple double-quotes for multi-line strings or strings than contain both ‘
and “ inside of them:
“““a‘b“c”””.

Shape Detection of Polygon Set

WHITESPACE:

Whitespace is meaningful in Python: especially indentation and placement of newlines.

• Use a newline to end a line of code.
• Use \ when must go to next line prematurely.
• No braces { } to mark blocks of code in Python…

Use consistent indentation instead.

• The first line with less indentation is outside of the block.
• The first line with more indentation starts a nested block
• Often a colon appears at the start of a new block.

(E.g. for function and class definitions.)

COMMENTS:

• Start comments with # – the rest of line is ignored.
• Can include a "documentation string" as the first line of any new function or class that you define.
• The development environment, debugger, and other tools use it: it's good style to include one.

ASSIGNMENT:

• Binding a variable in Python means setting a name to hold a reference to some object.
• Assignment creates references, not copies
• Names in Python do not have an intrinsic type. Objects have types.
• Python determines the type of the reference automatically based on the data object assigned to it.
• You create a name the first time it appears on the left side of an assignment expression: x = 3
• A reference is deleted via garbage collection after any names bound to it have passed out of scope.

Shape Detection of Polygon Set

SEQUENCE TYPES:

1. **TUPLE:**

• A simple immutable ordered sequence of items
• Items can be of mixed types, including collection types

2. **STRINGS:**

• Immutable
• Conceptually very much like a tuple

3. **LISTS:**

• Mutable ordered sequence of items of mixed types
• All three sequence types (tuples, strings, and lists) share much of the same syntax and functionality.
• Key difference:
• Tuples and strings are immutable
• Lists are mutable
• The operations shown in this section can be applied to all sequence types
• Most examples will just show the operation performed on one

DICTIONARIES: A Mapping type

• Dictionaries store a mapping between a set of keys and a set of values.
• Keys can be any immutable type.
• Values can be any type
• A single dictionary can store values of different types
• You can define, modify, view, lookup, and delete the key-value pairs in the dictionary

FUNCTIONS:

• def creates a function and assigns it a name
• return sends a result back to the caller
• Arguments are passed by assignment
• Arguments and return types are not declared

Shape Detection of Polygon Set

PASSING ARGUMENTS TO FUNCTIONS:

• Arguments are passed by assignment
• Passed arguments are assigned to local names
• Assignment to argument names don't affect the caller
• Changing a mutable argument may affect the caller

GOTCHAS:

• All functions in Python have a return value
• Even if no return line inside the code.
• Functions without a return return the special value None.
• There is no function overloading in Python.
• Two different functions can't have the same name, even if they have different arguments.
• Functions can be used as any other data type.

They can be:

• Arguments to function
• Return values of functions
• Assigned to variables
• Parts of tuples, lists, etc

MODULES:

• Modules are functions and variables defined in separate files
• Items are imported using from or import from module import function

```
import module
module.function()
```

• Modules are namespaces
• Can be used to organize variable names,

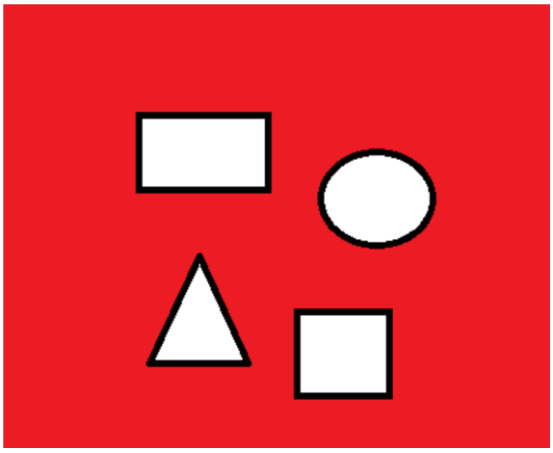
i.e.

```
atom.position = atom.position - molecule.position
```

4. IMPLEMENTATION:

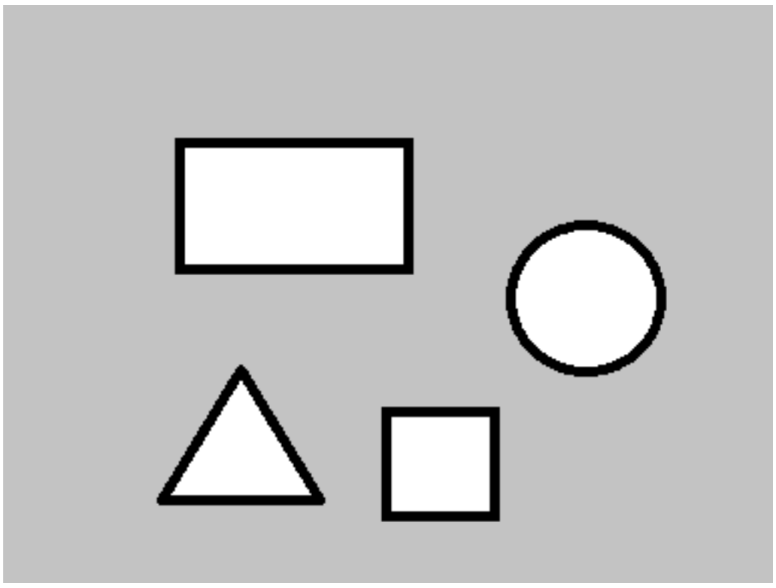
4.1 PROPOSED ALGORITHM:

We have proposed a new algorithm for recognizing and detecting shapes in the photos. The proposed algorithm has been developed to recognize and detect many different shapes given in any coloured photo and even in black white photo and the approach of the proposed algorithm is depending on the gray scale photos and even the coloured photos given as input for the proposed system will be convert to gray scale photo and for enhancing the photo, a contrast limited adaptive histogram equalization applied on small areas in the photo and it has been enhanced by the histogram equalization, so our first level is to enhance the given photo and this technique of histogram equalization is known already for analyzing the photo and enhance it. So we have applied this technique on the photo which is in gray scale level and after that we have applied the gaussian process of filtering which used and applied on the photo which resulted from the step of enhancing the photo in order to decrease the noising and this process will be applied as following:



(A) represents input coloured photo

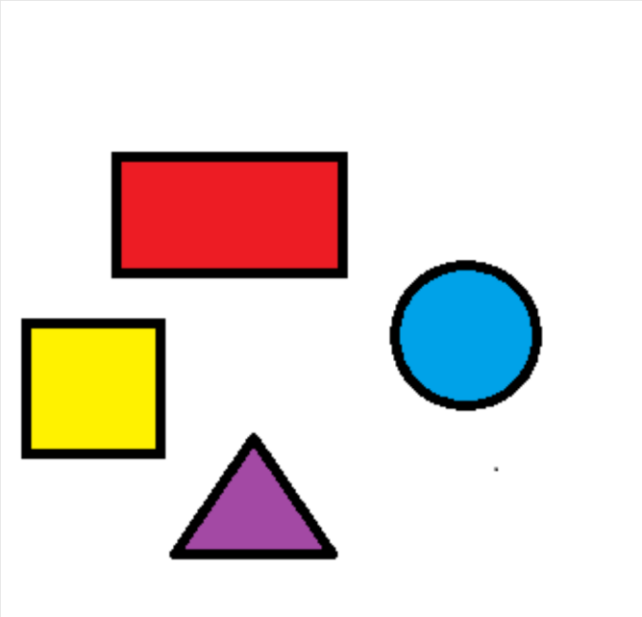
Shape Detection of Polygon Set



(B) represents the gray scale photo

(C) represents the binary photo

Shape Detection of Polygon Set



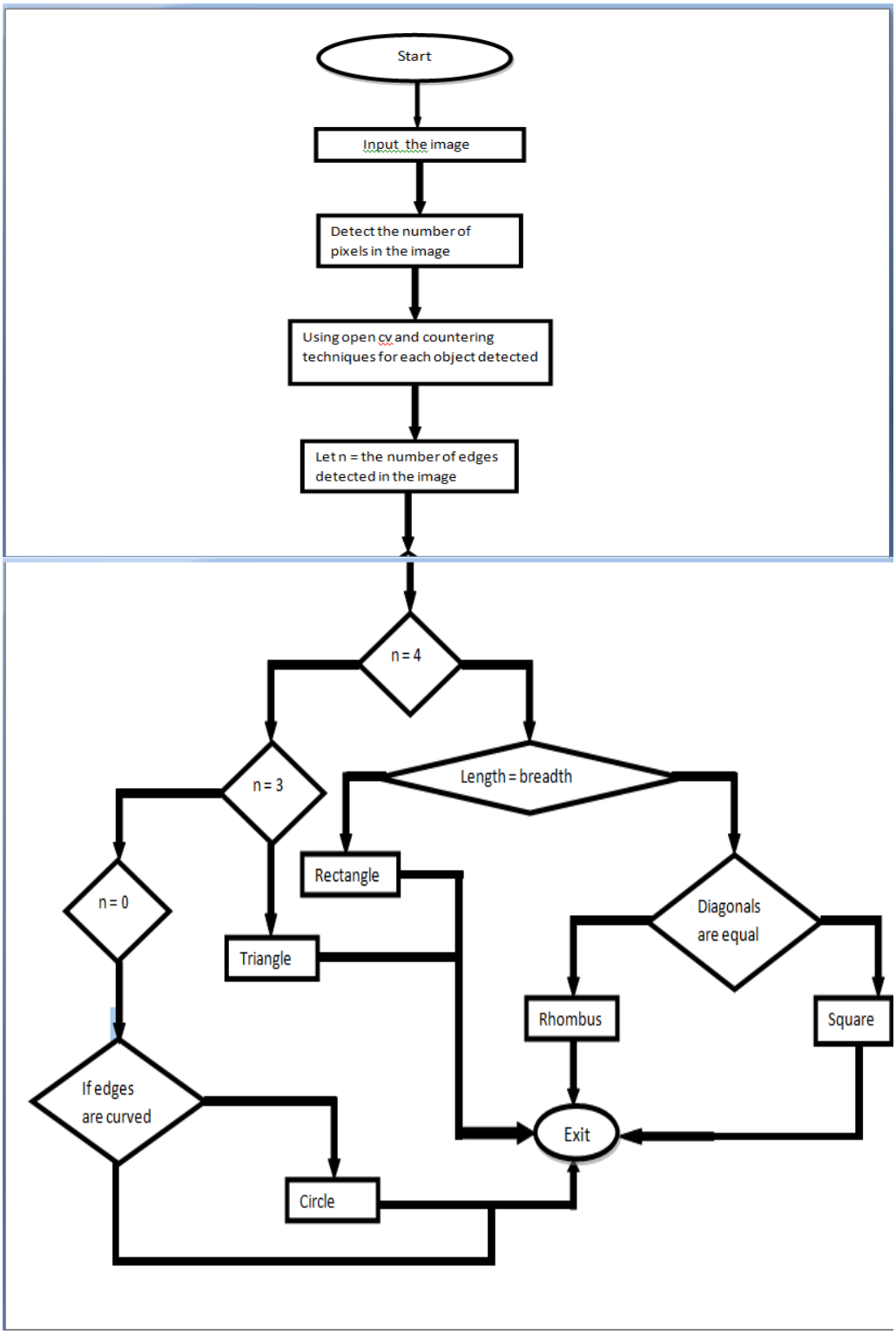
(D) Photo labelling

The following table represents the values of accept ratio:

Square : 0.484 <= SF <=0.55
Rectangle : 0.2 <= SF <=0.3
Triangle : 0.44 <= SF <=0.483
Circle : 0.7 <= SF <= 0.8

Shape Detection of Polygon Set

The proposed algorithm for the process is:



4.2 JUPYTER NOTEBOOK:

4.2.1 NOTEBOOK DOCUMENT:

Notebook documents (or "notebooks", all lower case) are documents produced by the Jupyter Notebook App, which contain both computer code (e.g. python) and rich text elements (paragraph, equations, figures, links, etc…). Notebook documents are both human-readable documents containing the analysis description and the results (figures, tables, etc..) as well as executable documents which can be run to perform data analysis.

4.2.2 JUPYTER NOTEBOOK APP:

The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet.

In addition to displaying/editing/running notebook documents, the Jupyter Notebook App has a "Dashboard" (Notebook Dashboard), a "control panel" showing local files and allowing to open notebook documents or shutting down their kernels.

4.2.3 KERNAL :

A notebook kernel is a "computational engine" that executes the code contained in a Notebook document. The ipython kernel, referenced in this guide, executes python code. Kernels for many other languages exist (official kernels).

When you open a Notebook document, the associated kernel is automatically launched. When the notebook is executed (either

cell-by-cell or with menu Cell -> Run All), the kernel performs the computation and produces the results. Depending on the type of computations, the kernel may consume significant CPU and RAM. Note that the RAM is not released until the kernel is shut-down.

4.2.4 NOTEBOOK DASHBOARD:

The Notebook Dashboard is the component which is shown first when you launch Jupyter Notebook App. The Notebook Dashboard is mainly used to open notebook documents, and to manage the running kernels (visualize and shutdown).

The Notebook Dashboard has other features similar to a file manager, namely navigating folders and renaming/deleting files.

4.2.5 INSTALLATION :

4.2.5.1 THE BROWSER:

The Browser consists in installing a modern standard-compliant browser. Either Mozilla Firefox or Google Chrome will work well. Try to avoid MS Explorer.

4.2.5.2 INSTALLING:

The easiest way to install the Jupyter Notebook App is installing a scientific python distribution which also includes scientific python packages. The most common distribution is called Anaconda:
Download Anaconda Distribution (a few 100MB), Python 3, 64 bits.
Install it using the default settings for a single user.
Official docs: Installation: If you are new to Python and Jupyter.

4.2.6 RUNNING THE JUPYTER NOTEBOOK:

4.2.6.1 LAUNCHING JUPYTER NOTEBOOK APP:

The Jupyter Notebook App can be launched by clicking on the Jupyter Notebook icon installed by Anaconda in the start menu (Windows) or by typing in a terminal (cmd on Windows):

This will launch a new browser window (or a new tab) showing the Notebook Dashboard, a sort of control panel that allows (among other things) to select which notebook to open.

When started, the Jupyter Notebook App can access only files within its start-up folder (including any sub-folder). No configuration is necessary if you place your notebooks in your home folder or subfolders. Otherwise, you need to choose a Jupyter Notebook App start-up folder which will contain all the notebooks.

See below for platform-specific instructions on how to start Jupyter Notebook App in a specific folder.

4.2.6.2 CHANGE JUPYTER NOTEBOOK STARTUP FOLDER

(WINDOWS) :

- Copy the Jupyter Notebook launcher from the menu to the desktop.
- Right click on the new launcher and change the Target field, change %USERPROFILE% to the full path of the folder which will contain all the notebooks.

- Double-click on the Jupyter Notebook desktop launcher (icon shows [IPy]) to start the Jupyter Notebook App. The notebook interface will appear in a new browser window or tab. A secondary terminal window (used only for error logging and for shut down) will be also opened.

4.2.6.3 CHANGE JUPYTER NOTEBOOK STARTUP FOLDER (MAC OS) :

- Click on spotlight, type terminal to open a terminal window.
- Enter the startup folder by typing cd /some_folder_name.
- Type jupyter notebook to launch the Jupyter Notebook App The notebook interface will appear in a new browser window or tab.

4.2.6.4 SHUT DOWN THE JUPYTER NOTEBOOK:

Closing the browser (or the tab) will not close the Jupyter Notebook App. To completely shut it down you need to close the associated terminal.

In more detail, the Jupyter Notebook App is a server that appears in your browser at a default address (http://localhost:8888). Closing the browser will not shut down the server. You can reopen the previous address and the Jupyter Notebook App will be redisplayed.

You can run many copies of the Jupyter Notebook App and they will show up at a similar address (only the number after ":", which is the port, will increment for each new copy). Since with a single Jupyter Notebook App you can already open many notebooks, we do not recommend running multiple copies of Jupyter Notebook.

4.2.6.5 CLOSE THE NOTEBOOK : KERNEL SHUT DOWN ::

When a notebook is opened, its “computational engine” (called the kernel) is automatically started. Closing the notebook browser tab, will not shut down the kernel, instead the kernel will keep running until is explicitly shut down.

To shut down a kernel, go to the associated notebook and click on menu File -> Close and Halt. Alternatively, the Notebook Dashboard has a tab named Running that shows all the running notebooks (i.e. kernels) and allows shutting them down (by clicking on a Shutdown button).

4.2.6.6 EXECUTING A NOTEBOOK :

Download the notebook you want to execute and put it in your notebook folder (or a sub-folder of it).

Then follow these steps:

- Launch the Jupyter Notebook App (see previous section).
- In the Notebook Dashboard navigate to find the notebook: clicking on its name will open it in a new browser tab.
- Click on the menu Help -> User Interface Tour for an overview of the Jupyter Notebook App user interface.
- You can run the notebook document step-by-step (one cell a time) by pressing shift + enter.
- You can run the whole notebook in a single step by clicking on the menu Cell -> Run All.
- To restart the kernel (i.e. the computational engine), click on the

menu Kernel -> Restart. This can be useful to start over a computation from scratch (e.g. variables are deleted, open files are closed, etc…).

4.3 CODE SNIPPETS :

4.3.1 PROGRAM FOR DETECTING THE POLYGONS:

```
#Packages required are:
import nbimporter
from detector.detection import PolygonDetector
import argparse
import imutils
import cv2

#change path while passing different image
Image=
cv2.imread("C:\\Users\\admin\\Desktop\\project\\pic1.png")
resized = imutils.resize(image, width=300)
ratio = image.shape[0] / float(resized.shape[0])

# converting the image to grayscale
gray = cv2.cvtColor(resized, cv2.COLOR_BGR2GRAY)
blurred = cv2.GaussianBlur(gray, (5, 5), 0)
thresh = cv2.threshold(blurred, 60, 255, cv2.THRESH_BINARY)

# contouring
cnts=cv2.findContours(thresh.copy(),cv2.RETR_EXTERNAL,cv2
.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if imutils.is_cv2() else cnts[1]
pd = PolygonDetector()
```

```
for c in cnts:
    # computing the center of the contour to detect polygon
    M = cv2.moments(c)
    cX = int((M["m10"] / M["m00"]) * ratio)
    cY = int((M["m01"] / M["m00"]) * ratio)
    polygon = pd.detect(c)

    c = c.astype("float")
    c *= ratio
    c = c.astype("int")
    cv2.drawContours(image, [c], -1, (255, 0, 0), 2)
cv2.putText(image,polygon,                                    (cX,cY),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 0, 0), 2)

    # output image
    cv2.imshow("Image", image)
    cv2.waitKey(0)
```

4.3.2 PROGRAM TO FIND WHICH POLYGON IT IS :

```
# import the necessary packages
import cv2
import nbimporter

class PolygonDetector:
    def __init__(self):
        pass

    def detect(self, c):
        # initialize the shape name and approximate the contour
        polygon = "unidentified"
        peri = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.04 * peri, True)
```

Shape Detection of Polygon Set

```
        # if the polygon is a triangle, then it has 3 vertices
        if len(approx) == 3:
            polygon = "triangle"

        # if the polygon has 4 vertices, it is either a square or a rectangle
        elif len(approx) == 4:
            (x, y, w, h) = cv2.boundingRect(approx)
            ar = w / float(h)
            polygon = "square" if ar >= 0.95 and ar <= 1.05 else "rectangle"

        # if the polygon is a pentagon, it will has 5 vertices
        elif len(approx) == 5:
            polygon = "pentagon"

        # otherwise, we assume the polygon is a circle
        else:
            polygon = "circle"
        return polygon
```

5.CONCLUSION :

In the current paper we proposed a new algorithm to detect a shape from any photo and we could even recognize the kind of the shape given in the input photo and after applying our proposed algorithm on photos we saw that the algorithm gives very good results even if they are many shapes in one photo by depending on the value of the shape factor which is proposed in the current paper and if we compare our work with other works we could see that most of other works are focusing on detecting ang recognizing some specific shapes but our work is recognizing all the kinds of shapes and we can say that our system is working very well and gives good results finally.

6. REFERENCES:

[1]. F. Pedro, "Representation and Detection of Shapes in Images," Massachusetts Institute of Tecnology, 2003.

[2].K. Karthik and ,. S. Wesley, "A Shape Recognition Algorithm Robust to Occlusion: Analysis & Performance Comparison," 2013.

[3]. K. M. S., "Shape recognition for image scene analysis," Université de Nice - Sophia-Antipolis, 2009.

[4]. M. Hankyu, C. Rama and R. Azriel, "Optimal Edge-Based Shape Detection," IEEE Transcations On Image Processsing NG, vol. 11, no. 11, pp. 1209-1226, 2002.

[5]. R. Mar, c. D. Philippe and L. Josep, "Boundary Shape Recognition Using Accumulated Length and Angle Information," Pattern Recognition and Image Analysis, vol. 4478, no. 27, p. 210–217, 2007.

[6]. S. Konrad and S. David, "Object Detection by Global Contour Shape," Article published in Pattern Recognition, pp. 1-30, 2008.

[7]. S. G., J. A., K. A., B. S., K. A, T. D.N. and N. A.K., "Object shape recognition from tactile images using regional descriptors," Nature and Biologically Inspired Computing (NaBIC), p. 53 – 58, 2012.

[8]. Z. M. Firdaus, C. H. Seng and S. S. Azmin, "Object Shape Recognition in Image for Machine Vision Application," International Journal of Computer Theory and Engineering, vol. 4, no. 1, pp. 76-80, 2012.