Distance Measurement Using OpenCV - Python

MAJOR PROJECT

Submitted in partial fulfilment for the award of Degree of

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering

to

Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, M.P.

Submitted by

Shubham Dhangar (0537CS191079)

Shubham Thakur (0537CS191080)

Vijay Verma (0537CS191088)

Under the Supervision of

Prof. Ajit Shrivastava

Department of Computer Science and Engineering

SAGAR INSTITUTE OF SCIENCE, TECHNOLOGY & RESEARCH BHOPAL, M.P.

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the work embodies in this major project entitled " **Distance Measurement Using OpenCV - Python***" being submitted* ***by Shubham Dhangar (0537CS191079),Shubham Thakur (0537CS191080), Vijay Verma (0537CS191088)*** *for partial fulfilment of the requirement for the award of "****Bachelor of Technology*** *in* ***Computer Science and Engineering"*** *discipline to* ***Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.)*** *during the academic year* ***'june - Dec 2022'*** *is a record of bona-fide piece of work, undertaken by him in the supervision of the undersigned.*

Supervised By:	**Approved By:**	**Forwarded By:**
Prof. Ajit Kumar Shrivastava	Prof. Ajit Kumar Shrivastava	Dr. Jyoti Deshmukh
A.P, CSE	Head of Dept., CSE	Principal
SISTec-R, Bhopal	SISTec-R, Bhopal	SISTec-R, Bhopal

1. Introduction

Distance measurement between a robot and the object is needed to control the action of the robot such as grabbing an object or even avoiding obstacles. There are many methods to estimate the distance such as ultrasonic ranging, laser ranging, and vision based ranging. Vision based techniques have the merit of its low cost, so in this project we will learn just a method for distance measurement between a single camera and the objects in front of it and implement it.

The challenge is to use a depth map and retrieve features from it to predict the distance of an object in a camera image. There will be discussion on other features that were discovered when investigating. Also included is the use of an object recognition mechanism, which aids in the detection and localization of objects in images.

1.1 OpenCV

OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc.

In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, a three-dimensional model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and act accordingly.

Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency.

There are a two main task which are defined below:

1. **Object Classification -** In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories.
2. **Object Identification -** In object identification, our model will identify a particular instance of an object - for example, parsing two faces in an image and tagging one as Person 1 and other one as Person 2.

1.2 How OpenCV Works

Human eyes provide lots of information based on what they see. Machines are facilitated with seeing everything, convert the vision into numbers and store in the memory. Here the question arises how computers convert images into numbers. So the answer is that the pixel value is used to convert images into numbers. A pixel is the smallest unit of a digital image or graphics that can be displayed and represented on a digital display device.

There are two common ways to identify the images:

1. Grayscale

Grayscale images are those images which contain only two colors black and white. The contrast measurement of intensity is black treated as the weakest intensity, and white as the strongest intensity. When we use the grayscale image, the computer assigns each pixel value based on its level of darkness.[3]

2. RGB

An RGB is a combination of the red, green, blue color which together makes a new color. The computer retrieves that value from each pixel and puts the results in an array to be interpreted.[3]

1.3 Why is OpenCV used for Computer Vision?

1. OpenCV is available for free of cost.
2. Since the OpenCV library is written in C/C++, it is quit fast. Now it can be used with Python.
3. It requires less RAM to use, maybe 60-70 MB.
4. Computer Vision is portable as OpenCV and can run on any device that can run on C.

2. Objectives

1. To obtain a system which can recognise objects and gives an accurate distance of that from the camera.
2. To make this project cost efficient without using some costly hardwares and some other complex machine learning algorithms.
3. To get solutions to real world machine learning problems.
4. For the purpose of learning openCV and other algorithms of machine learning.
5. For adding new functionalities, checking objects and can get distance of more than one object at a time.

3. Scope

1. Machine learning is one of the hottest topics nowadays, the demand of machine learning is increasing day by day and can be seen at its peak in the upcoming 2 to 4 years.
2. Computer Vision has its own scopes in the field of machine learning. Object detection, recognising shapes, patterns and edges around the objects are some features of computer vision.
3. OpenCV was built to provide an infrastructure for computer vision. This library has a huge range of optimized machine learning and computer vision algorithms, learning this technology provides scopes in this field.
4. Distance measurement has many uses in current technologies and will play an important role in upcoming technologies.
5. Self driving cars, Satellite images use object detection, social distance measurement applications are some popular use of distance measurement technology.

4. Description

The challenge of estimating the distance of objects from a camera remains a hot research subject in the field of computer vision, and it has several applications. In order to go on, it is necessary to understand a few words. Range calculation that is both reliable and accurate remains a difficult challenge in computer vision. Few methodologies exist for this, but they are very complex, and sensors and other heavy forms of machinery are used to measure the distance of objects. Understanding the space between two points is critical in robotics for avoiding collisions and picking up objects. Any sensor can do this, but they have a number of disadvantages. These constraints, however, can be solved by the use of image processing.
As a result, it was ultimately agreed to use image recognition to calculate the object's distance. The definition of depth picture, as well as object detection mechanisms, are heavily used to solve this issue. to build a self-contained artificial intelligence.

In this project we have used a single camera for object detection and distance measurement. First we will find the distance with the help of formula of focal length and then the distance by formula of finding distance by focal length of the camera.

By using the OpenCV library of python we will detect faces of humans. The RGB function and Grayscale function will help us to find the objects in the frames of the camera and then we will find the distance of the object by some mathematical formulas.

4.1 Mapping machine learning problem

fig.1 : Object detection

Without the help of any sensors or any heavy hardware can it be possible for a robot or any model to have a rough idea about its vicinity . Robots need to learn spatial location of objects under their vicinity. The proposed mechanism will make use of depth images, object detection mechanisms to find the spatial graph of the environment. Or by looking at the image is it possible to estimate the distance of the object from the camera.This is a Regression problem in Machine Learning, where the task is to predict a real-valued output (distance) of an object with respect to the camera present in the image or vicinity of the camera. The predicted value should be greater than zero.

4.2 Dataset Description

A dataset is required before proceeding with any machine learning or deep learning problem. Using object detection methodology and depth images, the goal is to construct a machine learning methodology that can classify the object present in the image and estimate the distance from the camera, that is, what will be the real distance from the camera to the object when those were clicked. There was no prior work and literature to address the issue of distance estimating using machine learning. As a result, the scarcity of jobs and study in this area motivates one to try to tackle the problem from the ground up. Here in this project the dataset is taken from kaggle, which will be used to detect face in the image.[1]

5. Requirements

Requirements are pretty simple you need Python, OpenCV, and Haar-cascade files for Face Detection. Having python installed on your machine, just open the terminal, and paste the following command in the terminal and you are done, Installation.

pip install opencv-python # windows 10

5.1 Capture the Reference Image

The reference image, allows us to map the real world(object plane) since we lose the depth of the object when we capture it in 2D space(image), otherwise, we need a special camera, that can capture the depth since we want to use our webcam, the reference image is required, you need to take care of few things, so accuracy won't drop, you can use mine reference image it won't affect much, but I will recommend to capture it yourself to get more accuracy. You have to take care of some things which are pointed below.

1. Resolution of the image (frame) must be the same, as in the reference image I have kept to the defaults of OpenCV which is(640,480).
2. Keep the camera straight as possible while capturing the reference images.
3. Measure distance(KNOWN_DISTANCE) from the object camera, note it down and capture the image which is set to 76.2 centimetres.

5.2 Software Used

1)	Development Environment	Visual Studio.Net 2008, Google Colab
2)	Technology Used	OpenCV, Visual Studio Code
3)	Language	Python
4)	Platform	Windows 10
5)	Documentation	Google Document

5.3 Hardware Used

1)	RAM	4GB
2)	Hard Disk	1TB
3)	Processor	Intel Core i3-7020U 2.30 GHZ

6. Software Design

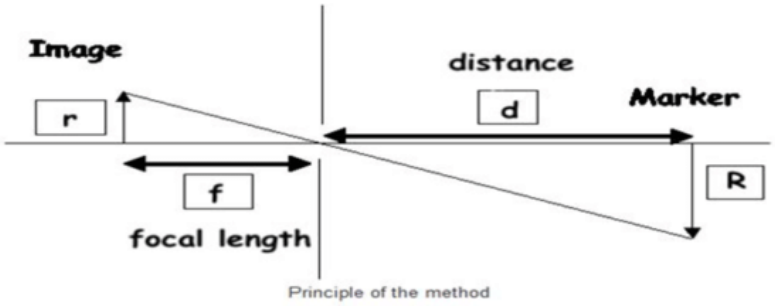
Software design is a mechanism to transform user requirements into some suitable form, which helps the programmer in software coding and implementation. It deals with representing the client's requirement.

Software design is more specific than computer programming, although many times the terms are used interchangeably. Software design focuses on the process that allows the user to define software methods and parameters. Design works with functions and the overall structure of the code.

6.1 Triangle Similarity for Object/Marker to Camera Distance

In order to determine the distance from our camera to a known object or marker, we are going to utilize triangle similarity.

The triangle similarity goes something like this: Let's say we have a marker or object with a known width *W*. We then place this marker some distance *d* from our camera. We take a picture of our object using our camera and then measure the apparent width in pixels *P*. This allows us to derive the perceived focal length *F* of our camera:



Principle of the method

Using the principle of Similar Triangles, we can obtain the formulas as follows:

$$\frac{f}{d} = \frac{r}{R} \quad (1)$$

$$f = d \times \frac{r}{R} \text{ pixels} \quad (2)$$

$$d = f \times \frac{R}{r} \text{ cm} \quad (3)$$

fig.2 : Mathematical way of measuring distance

F = (P x d) / W

For example, let's say I place a standard piece of *8.5 x 11in* piece of paper (horizontally; *W=11*) *d = 24 inches* in front of my camera and take a photo. When I measure the width of

the piece of paper in the image, I notice that the perceived width of the paper is *P = 248 pixels*. My focal length *F* is then:

F = (248px x 24in) / 11in = 543.45.

As I continue to move my camera both closer and farther away from the object/marker. I can apply the triangle similarity to determine the distance of the object to the camera:

D = (W x F) / P

Again, to make this more concrete, let's say I move my camera *3 ft* (or 36 inches) away from my marker and take a photo of the same piece of paper. Through automatic image processing I am able to determine that the perceived width of the piece of paper is now *170 pixels*. Plugging this into the equation we now get:

D = (11in x 543.45) / 170 = 35 inch

Or roughly 36 inches, which is 3 feet.

Note: When I captured the photos for this example my tape measure had a bit of slack in it and thus the results are off by roughly 1 inch. Furthermore, I also captured the photos hastily and not 100% on top of the foot markers on the tape measure, which added to the 1 inch error. That all said, the triangle similarity still holds and you can use this method to compute the distance from an object or marker to your camera quite easily.

6.2 Focal Length Finder

The Focal Length finder Function takes Three Arguments:

1. Measured_distance: It is the distance from the camera to object while capturing the Reference image, Known_distance = 72.2 #centimeter.
2. Real_width: Its measure the width of an object in real-world, here we measure the width of the face which is around Known_width =14.3 #centimeter
3. Width_in_rf_image: It is the width of the object in the image/frame it will be in pixels.
4. This function will return focal length, which is used to find distance.[2]

6.3 Distance Finder

This function has three arguments :-

1. Focal length in pixel, which is a return from the Focal length finder function
2. Real_width measures the width of an object in real-world, here we measure the width of the face which is around Known_width =14.3 #centimeter.
3. Width in reference image is the width of the object in the image/frame it will be in pixels.

The distance finder function will return the distance in the centimeters.[2]

7. Data Flow Diagram

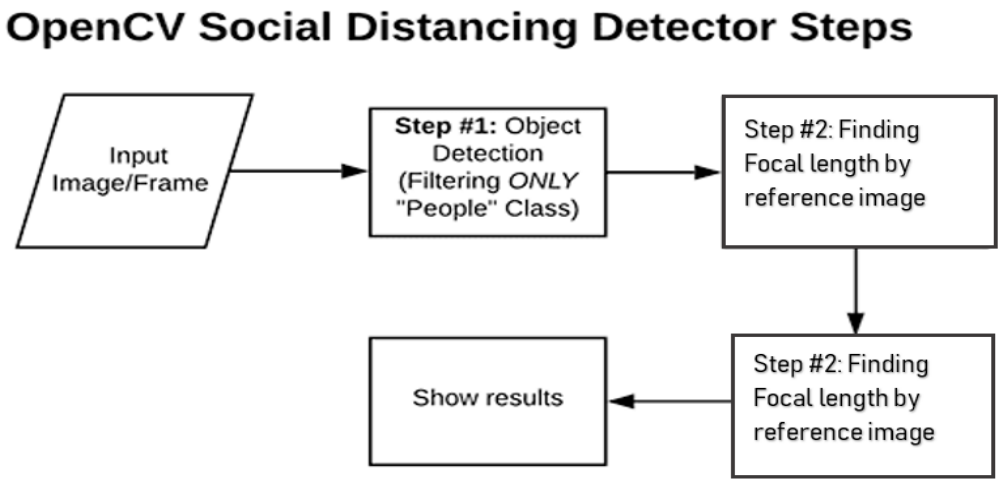


fig. 3 : Working

8. Block Diagram

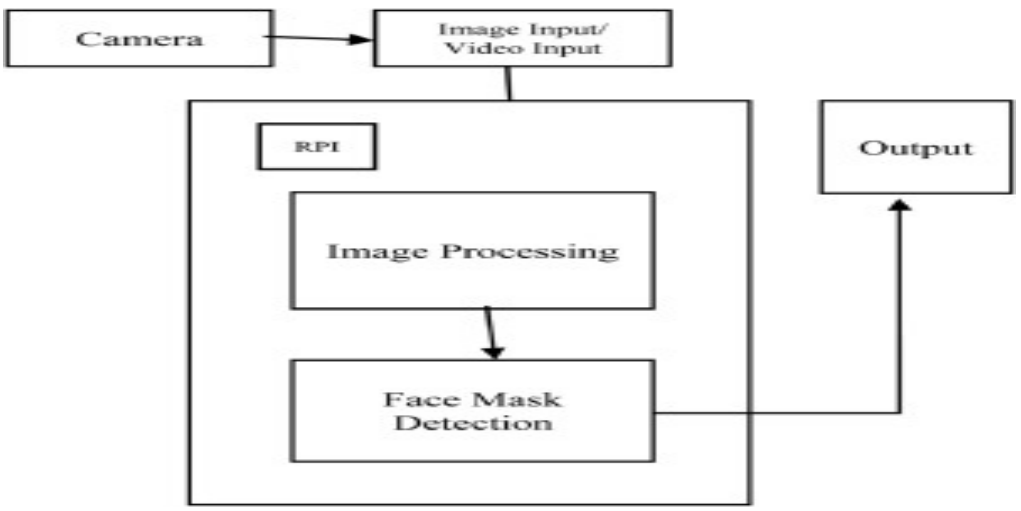


fig. 4 : Block diagram

9. Limitations

As the software is our first step towards making a robot which can detect objects and go to hold and carry an object, so it has some drawbacks also. It may have some less accuracy than an idle system.

Following are the few limitations associated with it as:

1. Using only one camera may have some less accuracy.
2. It may detect similar things as real ones and can give false output in some cases.
3. Knowledge of mathematics needed as formula for finding focal length of camera used.
4. Accurate values required at the time of training of the project, any minor error can give large error in the output.

10. Further Enhancements

1. We can add new functionalities, checking objects and can get the distance of more than one object at a time.
2. It will help in making robots which can clean window glass dust on tables and shoes.
3. We can make it more accurate by using dual cameras or stereo cameras.
4. Making it more accurate can help robots in grabbing things, holding and carrying from one place to another.

11. Completion of the project

Time duration - About 4 months in 7^{th} semester .

11.1 In contribution with

Project members – Shubham Dhangar (0537CS191079)

Vijay Verma (0536CS191088)

Shubham Thakur (0537CS191080)

In Guidance of - Prof. Ajit Kumar Shrivastava (HOD).

12. Conclusion

To accomplish this task we utilized the *triangle similarity*, which requires us to know two important parameters prior to applying our algorithm:

1. The *width (or height)* in some distance measure, such as inches or meters, of the object we are using as a marker.
2. The *distance* (in inches or meters) of the camera to the marker in step 1.
3. Computer vision and image processing algorithms can then be used to automatically determine the perceived width/height of the object in pixels and complete the triangle similarity and give us our focal length.

Then, in subsequent images we simply need to find our marker/object and utilize the computed focal length to determine the distance to the object from the camera.

13. Reference

[1] www.Kaggle.com (for getting datasets)

[2] www.stackoverflow.com (for general questions)

[3] www.Geeksforgeeks.com (for solving issues related to code)