

**TRIBHUVAN UNIVERSITY**

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

**Proposal for 3rd semester Project on Object oriented Program Using c++:**

Facial feature tracking video Using Webcam for Input

and its application:tic tac toe

**Submitted to:**

Department Of Electronics and Computer Engineering

**Submitted By:**

-Aavaas Gajurel

068/BCT/501

-Aman Kr. Karn

068/BCT/503

On:January 7, 2013

1.Acknowledgement

We would like to thank the Department of Electronics and Computer Engineering for providing the opportunity to do this project. Wa are also grateful to our subject teacher Mr. Daya Sagar Baral who is an outstanding teacher for making us capable and competent enough to endeavor this project. Any constructive criticism and suggestions for the betterment of this project will be highly appreciated and gratefully acknowledged.

**2.Abstract**

Image processing is an very interesting topic to endeavor, with many applications in real life occasions. One of the fields in which Image processing have been utilized for is in video surveillance and tracking. The real-time image stream which is a video can be used for tracking different features in time. This can be used for tracking of facial features to make it applicable for human computer interaction. Thus in this project is for demonstration of image processing principles and using it for a proof of concept application. Here, we take a video stream from the webcam attached to the computer and try to predict the cursor by analyzing facial features.

**Table of Contents**

1. Abstract
2. Table of Contents
3. Introduction
4. Objectives
5. Application
6. Literature survey
7. Existing System
8. Methodology
9. Implementation
   1. Block diagram
10. Results
11. Problems Faced and solutions
12. Limitations and future enhancements
13. Conclusion and recommendations
14. References

**Introduction:**

**C++**

This is the project report for the 3rd semester project on Object oriented Program and Design. C++ is a statically typed, free-form, multi-paradigm, compiled, general-purpose programming language. It is regarded as an intermediate-level language, as it comprises a combination of both high-level and low-level language features. Developed by Bjarne Stroustrup starting in 1979 at Bell Labs, it adds object oriented features, such as classes, and other enhancements to the C programming language. Originally named C with Classes, the language was renamed C++ in 1983, increment operator signifying the improvement over C.  
  
It implements "data abstraction" using a concept called "classes", along with other features to allow object-oriented programming. Parts of the C++ program are easily reusable and extensible; existing code is easily modifiable without actually having to change the code. C++ adds a concept called "operator overloading" not seen in the earlier OOP languages and it makes the creation of libraries much cleaner.

C++ maintains aspects of the C programming language, yet has features which simplify memory management. Additionally, some of the features of C++ allow low-level access to memory but also contain high level features.

C++ could be considered a superset of C. C programs will run in C++ compilers. C uses structured programming concepts and techniques while C++ uses object oriented programming and classes which focus on data. Read about the History of C and also about the History of C++.

**Image processing**

Image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it. It is a type of signal dispensation in which input is image, like video frame or photograph and output may be image or characteristics associated with that image. Usually Image Processing system includes treating images as two dimensional signals while applying already set signal processing methods to them.

It is among rapidly growing technologies today, with its applications in various aspects of a business. Image Processing forms core research area within engineering and computer science disciplines too.

*Image processing basically includes the following three steps.*

· Importing the image with optical scanner or by digital photography.

· Analyzing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not to human eyes like satellite photographs.

· Output is the last stage in which result can be altered image or report that is based on image analysis.

Purpose of Image processing

The purpose of image processing is divided into 5 groups. They are:

1. Visualization - Observe the objects that are not visible.

2. Image sharpening and restoration - To create a better image.

3. Image retrieval - Seek for the image of interest.

4. Measurement of pattern – Measures various objects in an image.

5. Image Recognition – Distinguish the objects in an image.

**OpenCV**

(Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision, developed by Intel, and now supported by Willow Garage and Itseez. It is free for use under the open source BSD license. The library is cross-platform. It focuses mainly on real-time image processing .The library may use various resources and proprietary optimized routines to accelerate itself.

**Objectives:**

**Design**

-To know about various features of c++.

-To use various Object oriented features.

-To be able to design and build an application.

-To be able to use application development utilities while developing the application.

-To know about process of object oriented program design.

-To make some useful application that can be used further than the project itself.

-To be more proficient in programming overall.

**Application**

**-**To laearn about Image processing and its basic methodologis.

-To be able to use OpenCV and build the application on it.

-Gain understanding of low level image processing structures.

-Use high level image processing functions to process data.

-To be able to apply image processing in real life scenario.

-To learn efficient application designing principles.

**Application:**

The project is an accessibility system that utilizes the normal webcam to track user’s eye for various inputs to the computer. The digital video signal will be taken form the webcam and will be processed in real time to generate a coordinate that will be the best possible prediction of the user’s gaze on the screen.

So, the general movement of user can be modeled and converted in some way to predict the gaze which will be the objective of this system at its bare minimum. The momentous blinking if detected, can be used for click response or double blinking if required.

There are various high end eye tracking systems that are head mounted or/and use infrared and other high end technologies to detect the user’s gaze. Altough they are presise, they are non realistic for the normal dayto day use. So, if the normal mounted webcam can be utilized to do the same job, it would be far more practical.

**Specifically**

-This can be used to implement the cursor movement system for mouse free user interface and blinking or double blinking can be detected for the click detection.

-can be used for user gaze and interest detection for various advertisements ; still or moving.

-this will be a great boon for disabled people who cannot use pointing devices.

-This can be place in public interfaces where pointing devices are impractical.

-This can increase productivity for advance users.

**Existing System:**

Various High end hardware are being used currently for various works. They use head mount cameras or infrared eye pupil detection. But are impractical for normal use and are costly too. But general algorithmic and procedural ideas can be inferred form them for this project.  
  
Currently existing Webcam eye tracking software exist but they are proprietary. Nevertheless, It gives the proof of concept of this project and its feasibility. Gazehawak which is an online application and is used for ad tracking for crowd sourcing of the information on websites. It cannot be used as an accessibility tool as proposed before.  
  
One open source project was also found but is probably an abandon ware; Opengazer is a small app ~500 kb in size and is GNU V 2 licensed. Because the last update to it was in 2005, there have been no further improvements. And because the technology has come a long way since then, various fast algorithms have been developed and the hardware has tremendously increased in power. So this project is a very feasible project to endeavor.

**Methodology:**

-Web cam interfacing

-eye detection using openCV/ sift

-GAZE PREDICTION ALGORITHM

-interface for pointing device

-interface programming

-The image feature detection library (openCV ).

-Web cam input is provided by openCV,  
  
-gnu g++ as a compiler.

-codeblocks as the IDE.

-development environment Operating system independent (programming in windows 8)

**Implementation:**

OpenCV has a modular structure, which means that the package includes several shared or static libraries. The following modules are available:

• **core** - a compact module defining basic data structures, including the dense multi-dimensional array Mat and basic functions used by all other modules.

• **imgproc** - an image processing module that includes linear and non-linear image filtering, geometrical image transformations (resize, affine and perspective warping, generic table-based remapping), color space conversion, histograms, and so on.

• **video** - a video analysis module that includes motion estimation, background subtraction, and object tracking algorithms.

• **calib3d** - basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo correspondence algorithms, and elements of 3D reconstruction.

• **features2d** - salient feature detectors, descriptors, and descriptor matchers.

**• objdetect** - detection of objects and instances of the predefined classes (for example, faces, eyes, mugs, people, cars, and so on).

**• highgui** - an easy-to-use interface to video capturing, image and video codecs, as well as simple UI capabilities.

**• gpu** - GPU-accelerated algorithms from different OpenCV modules.

We will be requiring the Core module, imgproc module, highgui module and the objdetect module for this project.

The core module defines all the basic data structures:(Key OpenCV Classes)

Point Template 2D point class

Point3-Template 3D point class

Size-Template size (width, height) class

Vec-Template short vector class

Matx-Template small matrix class

Scalar-4-element vector

Rect-Rectangle

Range-Integer value range

Mat-2D or multi-dimensional dense array

(can be used to store matrices, images,

histograms, feature descriptors, voxel

volumes etc.)

SparseMat-Multi-dimensional sparse array

Ptr-Template smart pointer class

OpenCV also provides various techniques to manipulate them too.

**Interfacing** is doen by the help of simple functions in openCV only, to take the input from webcam and display it we use following functions form the highgui module.

void **imshow**(const string& winname, InputArray mat)

The function imshow displays an image in the specified window.

void **namedWindow**(const string& winname, int flags=WINDOW\_AUTOSIZE )

The function namedWindow creates a window that can be used as a placeholder for images and trackbars

int **waitKey(**int delay=0)

Waits for a pressed key.

class **VideoCapture**

Class for video capturing from video files or cameras. The class provides C++ API for capturing video from cameras or for reading video files. Its different members are.

VideoCapture::open

VideoCapture::isOpened

VideoCapture::release

VideoCapture::grab

VideoCapture::retrieve

VideoCapture::read

Then , the imgproc module can be used for blurring, eroding, and filtering tasks.

**OBJECT DETECTION**

The main module of our concern is the objdetect module . Specially the cascade feature detectors. We use the HaarCascade detectors as the main backbone for our system.

Haar-like features are digital image features used in object recognition. They owe their name to their intuitive similarity with Haar wavelets and were used in the first real-time face detector. Viola and Jones[2] adapted the idea of using Haar wavelets and developed the so called Haar-like features. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image. For example, let us say we have an image database with human faces. It is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore a common haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is defined relative to a detection window that acts like a bounding box to the target object (the face in this case).

In the detection phase of the Viola–Jones object detection framework, a window of the target size is moved over the input image, and for each subsection of the image the Haar-like feature is calculated. This difference is then compared to a learned threshold that separates non-objects from objects. Because such a Haar-like feature is only a weak learner or classifier (its detection quality is slightly better than random guessing) a large number of Haar-like features are necessary to describe an object with sufficient accuracy. In the Viola–Jones object detection framework, the Haar-like features are therefore organized in something called a classifier cascade to form a strong learner or classifier.The key advantage of a Haar-like feature over most other features is its calculation speed. Due to the use of integral images, a Haar-like feature of any size can be calculated in constant time

We import the pre trained eye and nose cascades and apply the Viola Jones method to detect the face, then the eye and then the nose…Then after, we use simple arithmetic and algebraic methods to extract the coordinates.

For smooting the noise, we implement the deque for 10 iterations and average their values.

**Block Diagram:**

Interface

Video stream

Coordinate Generation

Image Processing and Feature Extraction

Smooting Algorithm

Previous State

Difference detection

Gaze Predection Algorithm

Detection

Current State

Results**:**

We were thus able to satisfactorily track the features and extract the crusor although the data was very noisy. The Haar features did a good job of tracking the features and was the key point in this application.

We have implemented a shared memory model such that the eyetracker application can provide data to any application requiring it. So, the application can get the cursor data from the shared memory.

The blink and wink detection also work considerably well but fail when the face is in extreme poses. The absence of the eye , both or any one could be detected and thus the blink data gathered.

Problems Faced and solutions

There were many problems faced due to the complexity of the problem,

-Due to the low resolution of the capture device, tremendous noise were introduced, thus we had to employ smoothing measures.

-The gaze of eye even though could be predicted, was very crude due to the less traverse of the eye as seen from the remote source.

-The features couldn’t be detected when the face was at extreme poses.

-The complexity of the gaze tracking algorithm made the realtime processing unfeasible, thus we had to resort to using facial features to predict the gaze

-The lighting condition also effects the robustness of feature trackers as they cannot work in very low light conditions.

-Extreme difference in lighting conditions also take a toll of feature detectors.

Limitations and future enhancements

-The accuracy of cursor is lacking.

-The inherent hardness of the problem means the program couldn’t run in real time.

-The eye degree couldn’t be measured precisely.

-The project can be tremendously enhanced if given enough time, the problems can be approached in newer ways and more research could be done.

-Can be made more robust and near to real-time for eye gaze.

-same algorithms can be run on high end cameras, and peoples gaze can be detected from far .

Conclusion and recommendations

The application was a satisfactory success and we are positive about the end result. We learnt a great deal during the process that would be helpful to us in future. The designing and application was enjoyable. Although the expected result was not achieved, it is because of the inherent hardness of the problem. The problem of gaze tracking using monocular camera has been tackled for a long time giving rise to numerous research and methodologies without any exact recommendations. Thus, this is a good field for research and the advancements in technology and processing power will lead to its realization in near future.

References

-Wikipedia

-Stackoverflow

-Secrets of OOP ic C++ by Mr Daya Sagar Baral and Diwakar Baral

-openCV refrence manual and turorials