

**A PROJECT PRELIMINARY REPORT**  
**ON**

**“Augmented Reality”**

**B.TECH- IV (ELECTRONICS & COMMUNICATION)**

**SUBMITTED BY:**

**AAKASH SHANBHAG**

**(Roll No.: U12EC149)**

**ANUSHA RELWANI**

**(Roll No.: U12EC016)**

**KESHA BODAWALA**

**(Roll No.: U12EC004)**

**GUIDED BY:**

**Prof. KISHOR P. UPLA**

**ECED, SVNIT**



**DEPARTMENT OF ELECTRONICS ENGINEERING**

**Year: 2015-16**

**SARDAR VALLABHBHAI NATIONAL INSTITUTE OF  
TECHNOLOGY (SVNIT)**

**SURAT-395007**

**Sardar Vallabhbhai National Institute of Technology,**

**Surat-07**

**Electronics Engineering Department**



## **CERTIFICATE**

This is to certify that candidate **Mr. Aakash Shanbhag (u12ec149), Ms. Anusha Relwani (u12ec016) and Ms. Keshu Bodawala (u12ec004)** of **B.TECH IV, 7<sup>TH</sup> Semester** have successfully and satisfactorily presented **UG Project** & submitted the Report on the topic entitled “**Augmented Reality**” for the partial fulfillment of the degree of Bachelor of Technology (B.Tech) in **May 2016**.

**Guide:** \_\_\_\_\_

**Examiner 1** Sign: \_\_\_\_\_ Name: \_\_\_\_\_

**Examiner 2** Sign: \_\_\_\_\_ Name: \_\_\_\_\_

**Examiner 3** Sign: \_\_\_\_\_ Name: \_\_\_\_\_

**Head,**

**ECED, SVNIT.**

**(Seal of the Department)**

## **Acknowledgement**

We would like to express our appreciation to all those who provided us with the opportunity to complete this report. We would like to thank Dr. (Mrs.) U. D. Dalal, for giving us the chance to work on the project. We also would like appreciate the efforts put in by our final year project guide, Mr. Kishor P. Upla, whose guidance and encouragement, helped us to organize our project work.

Furthermore we would also like to acknowledge the role of the whole of Electronics and Communication Department and all those who gave the permission to use all required equipment and the necessary material to complete the task. A special thanks goes to our internship guide, Mr. Tariq Merchant, who help us to get the required knowledge base for future endeavours. We would like to appreciate the guidance given by the other supervisor as well as the panel members especially, in our project presentation. This helped us to enhance our presentation skills to a great extent.

## **Abstract**

*Augmented Reality has a great potential and scope of development. The new developments in this field have proved to be extremely valuable and have ushered a revolution. In this project, we have made an attempt to develop a foldable display design, which employs the concept of Augmented Reality to make low cost and flexible marker(s) the source of a variety of images. Various orientations and the expansion of marker(s) can be used to construct images, which can prove useful for a multitude of applications. Using this interface, users can cut costs on paper and other materials by re-using the same marker according to the application.*

*We made use of Open CV to develop this program. An interface formed through a device with Open CV, a camera and the physical environment is exploited. As a part of this project report an attempt is made to cover algorithms from the basic, like those followed by OpenCV functions, to the algorithm required in their integration to form the project. These algorithms are followed by the results and limitations of the model.*

## Table of Contents

<b>Acknowledgement .....</b>	<b>3</b>
<b>Abstract .....</b>	<b>4</b>
<b>List of Figures:.....</b>	<b>7</b>
<b>List of Tables: .....</b>	<b>9</b>
<b>CHAPTER 1: INTRODUCTION .....</b>	<b>10</b>
<b>1.1 Augmented Reality .....</b>	<b>10</b>
1.1.1 Future of Augmented Reality.....	11
1.1.2 Implementation of AR .....	12
<b>1.2 Computer Vision .....</b>	<b>12</b>
<b>1.3 Outline of Report.....</b>	<b>13</b>
<b>CHAPTER 2: LITERATURE REVIEW .....</b>	<b>14</b>
<b>2.1 Platforms of AR: [3].....</b>	<b>14</b>
<b>2.2 Augmented Reality Methods [3] .....</b>	<b>15</b>
2.2.1 Outline.....	15
2.2.2 Location .....	16
2.2.3 Surface .....	17
2.2.4 Pattern .....	19
<b>2.3 Challenges in Marker Based AR: [4].....</b>	<b>19</b>
2.3.1 Sensor Accuracy:[4].....	19
2.3.2 Recognition and Tracking Challenges: [4] .....	20
<b>2.4 Applications of AR [4].....</b>	<b>20</b>
2.4.1 Education .....	21
2.4.2 Task Support .....	21
2.4.3 Navigation.....	22
2.4.4 Home and Industrial.....	23
2.4.5 Advertising.....	23
2.4.6 Translation .....	24
2.4.7 Foldable Displays [5].....	25
<b>BASICS OF IMAGE PROCESSING .....</b>	<b>27</b>
<b>3.1 Edge Detection Techniques: .....</b>	<b>27</b>
3.1.1 Canny Edge Detection Algorithm: [7] .....	28
<b>3.2 Thresholding [12] .....</b>	<b>32</b>

3.2.1	Adaptive threshold .....	33
3.2.2	Otsu's Thresholding [14] .....	34
<b>3.3</b>	<b>Homography [15].....</b>	<b>39</b>
3.3.1	Affine and Perspective Transformations .....	41
MARKER BASED AR.....		43
<b>4.1</b>	<b>Schematics of proposed algorithm: .....</b>	<b>43</b>
4.1.1	Specifications:.....	43
4.1.2	Main program: .....	43
4.1.3	Search marker module: .....	43
4.1.4	Marker detection module: .....	45
4.1.5	Display module: .....	46
<b>4.1</b>	<b>Results: .....</b>	<b>48</b>
SHORTCOMINGS .....		51
<b>5.1</b>	<b>Main issues in AR application development [16] .....</b>	<b>51</b>
<b>5.2</b>	<b>Limitations of proposed algorithm: .....</b>	<b>53</b>
CONCLUSION.....		55
References: .....		56

## List of Figures:

Fig.2. 1 : Augmented Sunglasses [3] .....	16
Fig.2. 2 : An AR Browser Showing Various Locations to the Viewer [3] .....	17
Fig.2. 3 : A User Interacts with Virtual Documents on a Physical Desk Using LightSpace [3] .....	18
Fig.2. 4: Au Augmented Reality Floor [3].....	18
Fig.2. 5 : Marker-Based Augmented Reality [3] .....	19
Fig.2. 6 : Augmented Reality Used in a Traditional Book [4].....	21
Fig.2. 7: An AR Manual Giving Step-by-Step Instruction on How to Change the Ink Cartridge in a Printer [4] .....	22
Fig.2. 8: The TapNav Display [4] .....	22
Fig.2. 9: Augmented Reality Being Used to Show the Placement of a New Television [4] ..	23
Fig.2. 10 : AR ad for a Mini Cooper [4] .....	24
Fig.2. 11 : The WorldLens Augmented Reality Translator [4].....	24
Fig.2. 12 : An illustration of the proposed Foldable Display model.....	26
Fig.3. 1 : Prewitt, Sobel and Roberts operators [1] .....	27
Fig.3. 2 : Block diagram for Canny edge detection [1] .....	28
Fig.3. 3 : Masks used by Sobel operator [2] .....	29
Fig.3. 4 : A block of 5 X 5 pixels [2] .....	30
Fig.3. 5 : Edge direction detection [2] .....	30
Fig.3. 6 : Image used for edge detection analysis [2] .....	31
Fig.3. 7 : Results of edge detection on Fig.3.6 Canny had the best results [2].....	32
Fig.3. 8 : Comparison of different thresholding methods .....	38
Fig.3. 9 : Histogram of different thresholding method .....	39
Fig.3. 10 : View of a planar object as described by homography: a mapping—from the object plane to the image plane—that simultaneously comprehends the relative locations of those two planes as well as the camera projection matrix [6] .....	41
Fig.3. 11 : Affine and perspective transformations [6] .....	41
Fig.3. 12 : Predict-Match –Update cycle [7].....	42
Fig.4. 1 : Schematic of main program.....	43
Fig.4. 2 : Schematic of ‘search marker’ module .....	44
Fig.4. 3 : Edge detection in absence of marker .....	44

Fig.4. 4 : Edge detection in presence of marker .....	45
Fig.4. 5 : Schematic of ‘marker detection’ module .....	45
Fig.4. 6 : Contour with 4 corners; rejected as a marker during pattern recognition .....	46
Fig.4. 7 : Marker captured for pattern recognition.....	46
Fig.4. 8 : Conversion to black and white image for Otsu adaptive thresholding.....	46
Fig.4. 9 : Schematic of ‘display’ module .....	47
Fig.4. 10 : Placement of image on marker using warp perspective transform.....	47
Fig.4. 11 : Ouptut for marker 1 .....	48
Fig.4. 12 : Output for marker2 .....	48
Fig.4. 13 : Output for marker3 .....	49
Fig.4. 14 : Output for marker4 .....	49
Fig.4. 15 : Output for combination of marker1 and marker2.....	49
Fig.4. 16 : Output for combination of marker3 and marker4.....	50



## List of Tables:

Table 1: Properties of different transformation spaces [6] .....	40
--	----

# CHAPTER 1: INTRODUCTION

Throughout history humans have had the need to expand our perceived reality by any means available. Up until now these means have been mostly limited to different kind of artistic and non-real-time ways of expressing ourselves or using pure imagination. The advent of widespread mobile computing combined with ever advancing imaging and display technologies has for the first time enabled us to bring this augmentation of our reality to a more of a concrete, physical and most crucially real-time and lifelike experience.

## 1.1 Augmented Reality

Augmented reality (AR) is a technique for augmenting –adding and supplementing digital content over real world using computers and typically mobile AV devices like data glasses. It is a subdivision of mixed reality (MR) which stands for mixing computer generated content – a virtual world – in to real world. *Augmented reality* (AR) combines real world and digital data. At present, most AR research uses live video images, which the system processes digitally to add computer-generated graphics. In other words, the system *augments* the image with digital data. Encyclopaedia Britannica gives the following definition for AR:

*“Augmented reality, in computer programming, a process of combining or ‘augmenting’ video or photographic displays by overlaying the images with useful computer-generated data.”*

Augmented reality research combines the fields of computer vision and computer graphics. The research on computer vision as it applies to AR includes among others marker and feature detection and tracking, motion detection and tracking, image analysis, gesture recognition and the construction of controlled environments containing a number of different sensors.

Computer graphics as it relates to AR includes for example photo realistic rendering and interactive animations.

Researchers commonly define augmented reality as a real-time system. However, we also consider augmented still images to be augmented reality as long as the system does the augmentation in 3D and there is some kind of interaction involved.

Augmented reality as a system is identified by three characteristics:

- It combines the real and the virtual
- It is interactive in real time
- It is registered in 3D.

A simple augmented reality system consists of a camera, a computational unit and a display. The camera captures an image, and then the system augments virtual objects on top of the image and displays the result.

The variety of possible devices for an augmented reality system is huge. These systems can run on a PC, laptop, mini-PC, tablet PC, mobile phone or other computational unit. Depending on the application, they can use a digital camera, USB camera, Fire Wire Camera or the built-in camera of the computational unit. They can use a head-mounted display, see-through display, external display or the built-in display of the computational unit, or the system may project the augmentation on to the real world or use a stereo display. The appropriate setup depends on the application and environment.

Modern Augmented Reality technologies are built upon mobile devices such as smart phones and tablets that can be carried around with ease. A common practice is to use the cameras on these devices to capture video, draw additional content over it and display the result on the screen of the same device. All done in real-time and possibly with very complex modification to the original video feed involving virtual 3D models and imaging algorithms. By combining the data from other sensors on the device, such as GPS, compass, gyroscope and touch screen, this augmentation can be made even more immersive and allow the user to interact with the virtual content. [1]

### **1.1.1 Future of Augmented Reality**

As the everyday usage of augmented reality technologies is currently only starting to take shape, it is still very hard to predict how casual users will adapt to it and what will be the key points of attraction towards using it. Naturally, it will even more enhance our possibility to stay constantly connected to not just our physically surrounding world but also to the virtual reality we so much have become accustomed to in the form of the internet. Sharing our

experiences and thoughts will become even more instant and less intrusive from both the sharer's and receiver's point of view.

### **1.1.2 Implementation of AR**

AR tools often use third party libraries for lower level tasks (external tools) and wrap them into the level needed for AR. They use OpenCV for computer vision and image processing, for example, and Eigen or LAPACK for linear algebra. In addition, they may provide an interface for existing tools for image acquisition (e.g. Highgui) and camera calibration (e.g. OpenCV), or provide their own utilities for these tasks. An AR application developer may naturally use any other software for image acquisition and calibration as well. Respectively, AR applications normally use existing graphics libraries and 3D engines for graphics and rendering (e.g. OpenGL, OpenScene Graph, OGRE, Papervision3D, etc.).

Augmented reality tools are difficult to compare, as some of them are specialized to one purpose (e.g. marker-based tracking or mobile environments), some support only certain platforms (e.g. Windows or iOS) and others support several platforms and are used for several purposes

## **1.2 Computer Vision**

Computer vision is the transformation of data from a still or video camera into either a decision or a new representation. All such transformations are done for achieving some particular goal. Vision can mean many things in the world of computers. In some cases we are analyzing still frames loaded from elsewhere. In other cases we are analyzing video that is being read from disk. In still other cases, we want to work with real-time data streaming in from some kind of camera device. [2]

In a machine vision system, however, a computer receives a grid of numbers from the camera or from disk, and that's it. For the most part, there's no built-in pattern recognition, no automatic control of focus and aperture, no cross-associations with years of experience. For the most part, vision systems are still fairly naïve

OpenCV is aimed at providing the basic tools needed to solve computer vision problems. In some cases, high-level functionalities in the library will be sufficient to solve the more complex problems in computer vision. Even when this is not the case, the basic components

in the library are complete enough to enable creation of a complete solution of your own to almost any computer vision problem. In the latter case, there are several tried-and-true methods of using the library; all of them start with solving the problem using as many available library components as possible.

### **1.3 Outline of Report**

In this project report, Chapter 2 deals with a literature survey on Augmented Reality which provides an overall view of the subject. It includes the platforms and methods in which AR projects utilize and also the various applications which can be developed using this technology. The project we are developing has been described in detail. Chapter 3 deals with the basic algorithms which are used by OpenCV functions. The working of the major functions, which include Canny edge detector, Thresholding and Perspective Transform, used in the code are mentioned in depth. Chapter 4 mentions the step-by-step algorithm behind the project presented. Furthermore Chapter 5 deals with the shortcomings and drawbacks of Augmented Reality in general followed by the limitations of our project.

## CHAPTER 2: LITERATURE REVIEW

### 2.1 Platforms of AR: [3]

Augmented Reality requires various hardware and software equipment to function. The application interface can be executed on other devices which are capable of supporting computer vision. The various interfaces or platforms that Augmented Reality requires to function are:

1. **Personal Computers with Webcams:** Since most PCs contain some, if not all, the needed components for viewing Augmented Reality on this platform are an obvious choice. Because of the fixed nature of the device (compared to mobile phones and tablets), a marker is placed within view of a Webcam, which shows a live feed. Once it identifies the marker, it creates the augmentation on the screen for the user to interact with. This method is often used to augment magazine advertisements, business cards, baseball cards, and almost anything else that could be made into a portable marker and placed in front of the Webcam. Gaming systems such as the Xbox are also starting to be used more and more for Augmented Reality.
2. **Kiosks, Digital Signage, and Window Displays:** Kiosks are simply stations where customers can bring items to find out more about them with Augmented Reality information. One example is the Lego Store kiosk, which displays the completed Lego set inside the box. Kiosks are also used at trade shows and conventions to give attendees a richer experience. Digital signs and window display are also used and are basically large static markers that users interact with via their mobile devices.
3. **Smartphones and Tablets:** The use of smartphones to access Augmented Reality content is arguably the most common method today. Smartphones can not only use their cameras and screens to identify markers they are pointed at but can also use the compass and GPS functions to augment the locations or points of interest based on relative location. Tablet computers also fall under this general platform category as many of the higher-end models on the market today have HD cameras and GPS capability.

4. **AR Glasses and Head-Mounted Displays:** While not yet common, AR-enabled glasses such as those made by Vuzix do exist and are available for purchase. In time, as the technology improves and prices come down, AR-enabled glasses will likely become as common as iPads and smartphones giving the wearer the option for a continuous Augmented Reality feed based on individual needs and preferences.

## **2.2 Augmented Reality Methods [3]**

Most augmented reality application requires an identifier to define the location of the overlay object. There are various identifiers or markers that augmented reality softwares are capable of detecting. These identifiers can fall under any one of the following categories:

1. Outline.
2. Location.
3. Surface.
4. Marker

### **2.2.1 Outline**

The outline method of augmented reality is the process by which a part of the body is recognized, such as hands, face, or body and is then blended seamlessly with any digital element. With the outline method a person is able to interact with a 3D object using natural movements, such as picking up a virtual object with a real hand. The camera tracks the outline of a person's hand and adjusts the virtual object accordingly.

The method is similar when tracking a face. When the AR software detects a face, it determines the position of various facial features, eyes, nose, mouth, and so on then uses those positions as reference points for overlaying digital objects on the face. Once the software has recognized the face, it can also adjust for movement, redrawing the virtual objects in real time. (Fig. 1)



Fig.2. 1 : Augmented Sunglasses [3]

### 2.2.2 Location

It is based on detailed GPS or triangulation location information. Using this information and the position view of the camera the AR system can precisely overlay icons and virtual objects over buildings or people as you move around in the real world. Modern mobile phones have all the necessary components to enable location-based augmented reality packed into one device: a camera, a screen, GPS capabilities, accelerometers, and a digital compass.

Much like Internet browsers that let you find information on the Internet, AR browsers let you find information in the real world (Fig. 2). AR browsers are designed to allow you to see information about almost anything you point your mobile's camera at. A common example is the location of a coffee shop that may not be within your direct line of sight but is only a couple of minutes walk away or a review for a restaurant you are standing in front of. This information is "attached" to the specific GPS coordinates of interest points around you and displays the information on your mobile, in real time.

Due to the portability and high-resolution camera of the mobile device, the AR browser is also capable of using Pattern and Outline methods such as identifying QR codes. For example, an advertisement with a QR code, once it is recognized by the AR browser, can provide you information about the product or point you in the direction of the nearest store where you can find it.





Fig.2. 2 : An AR Browser Showing Various Locations to the Viewer [3]

## 2.2.3 Surface

Surface augmented reality is accomplished using screens, floors, or walls that respond to the touch of people or objects and provide them with virtual real-time information. In 2007 Microsoft released a coffee-table-sized computer called “Surface” which sees and responds to touch and real-world objects. In time this Surface computer was combined with augmented reality components and a project called LightSpace was created at Microsoft. LightSpace, combining surface computing and AR, creates an environment where any surface, and even the space between surfaces, is fully interactive (Fig. 3). This combination of surface computing and augmented reality is called spatial computing.

One recent development by a Kinect enthusiast was to develop an AR-type platform that allowed any surface to be used as a computing interface.

Another example of the surface method of augmented reality is the AR floor (Fig. 4). The AR floor uses a special tile, which through the use of precisely calibrated vibrations can simulate pebbles, sand, snow, grass, and a variety of other surfaces. Sensors in the floor detect the force from a person’s foot and then calibrate a response in the plate which when vibrated at the right frequency provides the simulated feel of different materials. Speakers inside the platform add the appropriate sounds, completing the illusion. In essence the floor becomes a

large touch-sensitive screen. There is certainly potential for this type of technology for gaming, training, and entertainment. In time homes could be outfitted with this type of flooring to create the feel of any environment a person could want indoors.

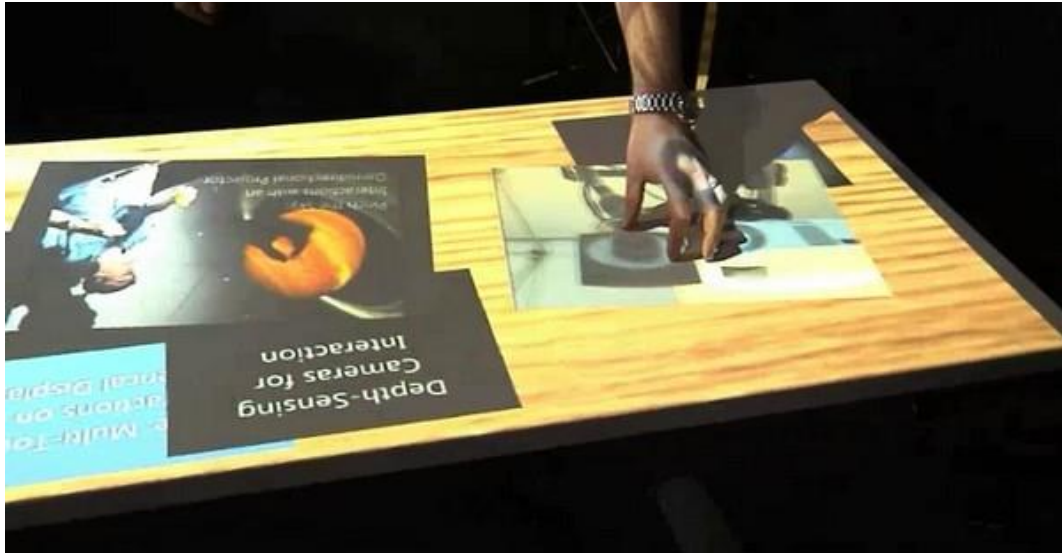


Fig.2. 3 : A User Interacts with Virtual Documents on a Physical Desk Using LightSpace [3]



Fig.2. 4: Au Augmented Reality Floor [3]

### 2.2.4 Pattern

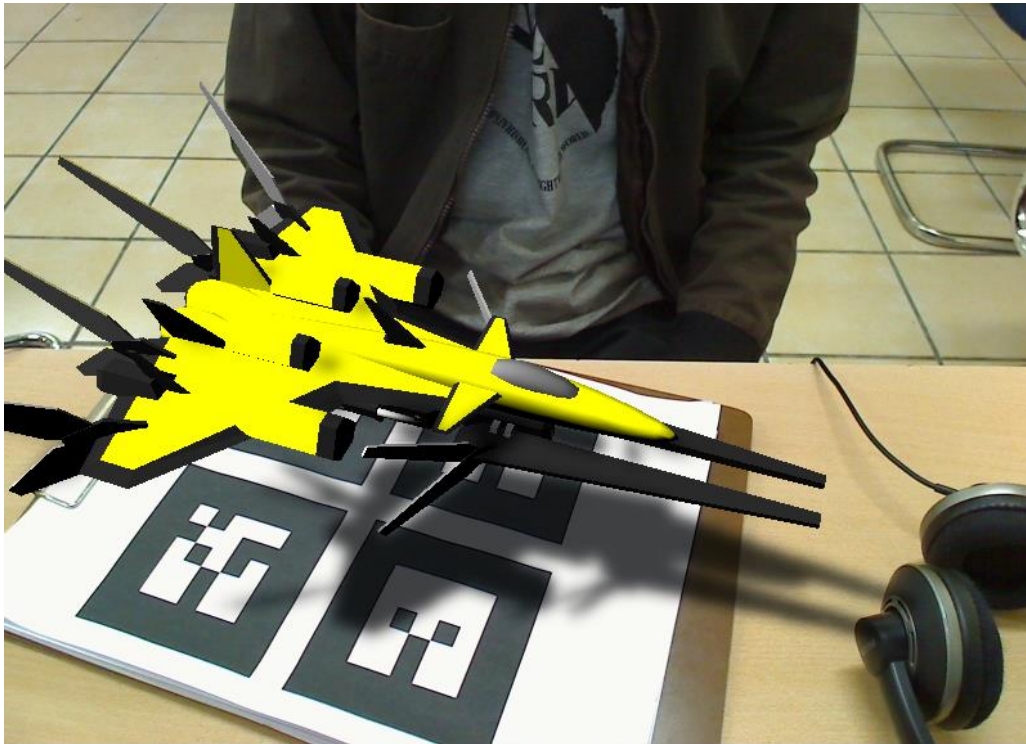


Fig.2. 5 : Marker-Based Augmented Reality [3]

This type of augmented reality system performs simple pattern recognition on a basic shape or marker. When recognized the system replaces that area with a static or moving digital element which can be anything from a 3D model, an audio or video clip, or some other piece of information as shown (Fig. 5). This method is most commonly used when interacting with AR using a PC and a Webcam and often the person is part of the augmented video feed.

## 2.3 Challenges in Marker Based AR: [4]

The biggest technical challenges affecting AR are object recognition and sensor accuracy. Many companies such as Sony and Apple are actively working to overcome these problems and work towards a more accurate functioning.

### 2.3.1 Sensor Accuracy:[4]

Object recognition or the “registration” problem is one of the most basic problems currently limiting Augmented Reality applications. Objects in the real and virtual worlds must be properly aligned with respect to each other, or the illusion that the two worlds coexist will be affected, sometimes severely.

Sensor accuracy applies to mobile AR and the systems that support it. Modern mobile Augmented Reality systems use one or more of the following tracking technologies: digital cameras and/or other optical sensors, accelerometers, GPS, gyroscopes, solid-state compasses, RFID, and wireless sensors. These technologies offer varying levels of accuracy and precision. Indoor positioning and line-of-sight also create challenges when dealing with location-based AR.

### **2.3.2 Recognition and Tracking Challenges: [4]**

Computers still have threshold limits when it comes to distinguishing background and foreground images in less than ideal conditions. There are a number of tracking challenges, listed and defined below, which lead to recognition problems:

- *Occlusion*: is the obstruction or blocking of sight.
- *Unfocused camera*: an unfocused camera lens will cause the marker details to be interpreted with lower precision, which leads to errors in positioning of the virtual object or complete non-recognition of the marker.
- *Motion blur*: is the apparent streaking of rapidly moving objects. In AR the effect of motion blur usually does not originate with the object but with the camera, usually on a mobile device.
- *Uneven lighting*: can obscure a marker by darkening portions in shadow making it unrecognizable, or recognized as a different marker to the AR application.

## **2.4 Applications of AR [4]**

Augmented Reality is the thing of today. The advent of widespread mobile computing combined with ever advancing imaging and display technologies has for the first time enabled us to bring this augmentation of our reality to a more of a concrete, physical and most crucially real-time and lifelike experience. It is a very big advancement in the field of computer vision and is already being employed in countless applications. In principle, the augmentations can be derived from any information that is available and can be associated with the real world objects around the user. Various examples are as follows.



### 2.4.1 Education

#### Augmented Reality Books:

Augmented Reality can breathe new life into old books, and even new e-books, by augmenting specific content with 3D graphics or animations, audio or visual information as depicted in Fig. 6. AR could be used in a traditional, hardcopy book where the fundamental information may not change a great deal but updates and advances could be viewed as AR in the appropriate section of the book and allowing an interaction with the content in a more engaging way.



Fig.2. 6 : Augmented Reality Used in a Traditional Book [4]

### 2.4.2 Task Support

AR has been, and will continue to be used to aid people to more easily carrying out complex tasks such as assembly and maintenance.

#### Augmented Manuals

Today user guides and repair manuals are digitally available online. In time, guides and manuals may be converted to become interactive instruction sets in Augmented Reality. Augmented manuals would be easier to understand if they moved beyond text and pictures to 3D drawings superimposed upon the actual equipment and providing step-by-step instructions.

An example of this that exists today is a laser printer maintenance application (Fig. 7) also built by Steven Feiner's group at Columbia University.

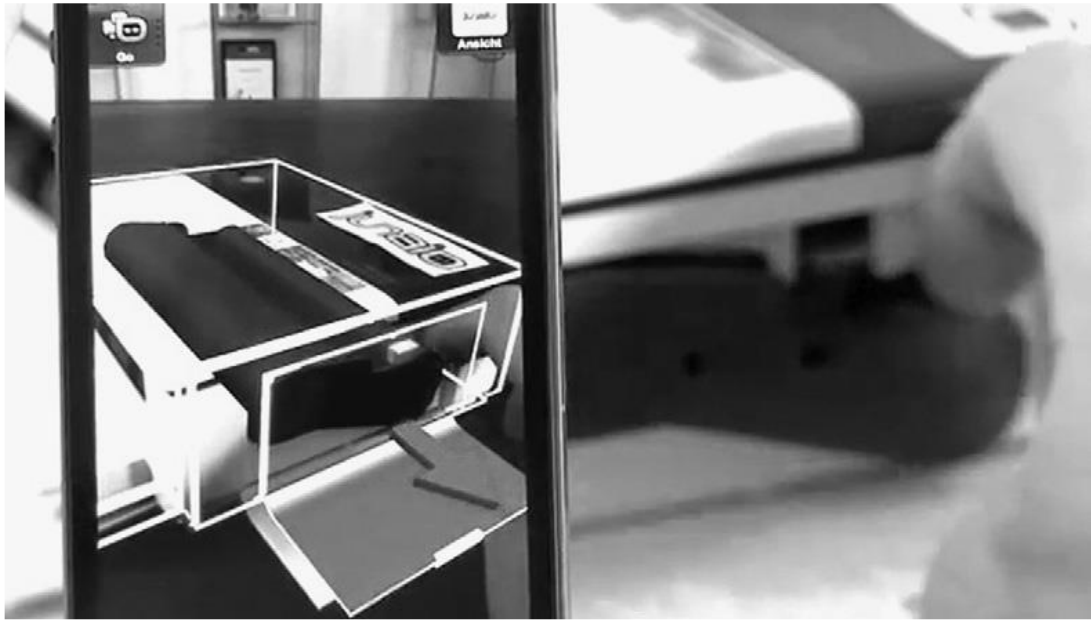


Fig.2. 7: An AR Manual Giving Step-by-Step Instruction on How to Change the Ink Cartridge in a Printer [4]

### 2.4.3 Navigation

City guides such as Yelp and NRU (pronounced “near you”) which help people find places to eat, drink, and shop have Augmented Reality capabilities that give user real-time visual directions to the places their looking for.

Another application called TapNav uses AR to overlay your route on the road ahead. (Fig.2.8)



Fig.2. 8: The TapNav Display [4]

#### 2.4.4 Home and Industrial

For home use, companies like Iyka and Total Immersion's Magic Mirror use Augmented Reality to place and scale representations of furniture or appliances letting the viewer gets a sense for how it looks (Fig. 9). The same application could also be used for larger projects such as comparing digital mock-ups with physical mock-ups to find any discrepancies between them.



Fig.2. 9: Augmented Reality Being Used to Show the Placement of a New Television  
[4]

#### 2.4.5 Advertising

Companies such as Nissan, Toyota, BMW, and Mini are using magazine advertisements and AR to give the viewer a full 3D view of the car being advertised.(Fig. 10)

The motion picture industry has also taken advantage of Augmented Reality to promote movies such as Transformers, Iron Man, and Star Trek.

One of the bigger examples is the “N Building” in the Tokyo shopping district, which is outlined in the Quick Response (QR) code section that allows shoppers and passers-by to use AR to get real-time information about what’s inside the building, show Tweets that are being posted by people inside the building, and have different augmented decorations depending on the season.



Fig.2. 10 : AR ad for a Mini Cooper [4]

### 2.4.6 Translation

Optical character recognition has improved steadily over the years and an outgrowth of this has led to the development of an Augmented Reality translator. The user simply points a smartphone at the text he or she wishes to translate and the answer appears on the screen. Word Lens is one such AR translation application that reads the text visible in the camera window, translates it, and then overlays the original text with the translation. (Fig. 11)



Fig.2. 11 : The WorldLens Augmented Reality Translator [4]



### 2.4.7 Foldable Displays [5]

Modern computer displays tend to be in fixed size, rigid, and rectilinear rendering them insensitive to the visual area demands of an application or the desires of the user. Foldable displays offer the ability to reshape and resize the interactive surface at our convenience and even permit us to carry a very large display surface in a small volume. Foldable display are designed using image projection with low-cost tracking and orientation sensitivity.

Many of the displays we see in hand-held devices today are small LCD displays of fixed shape and size. In this respect, they are insensitive to the desires of a user or the needs of an application. Ideally, we would like displays that we can dynamically reshape or resize to suit our desired usage, similar to the way we might handle a newspaper, or simply so that we are able to fit a large display into our pocket. In this paper, we explore this concept of inactive foldable displays and create a number of working prototypes.

Emerging technologies such as electronic paper and organic light emitting diode (OLED) displays are expected to provide some degree of flexibility. However, current prototypes remain quite rigid and are typically rectilinear. This prevents them from becoming truly foldable in the sense that we think of paper as being foldable. Additionally, performing input on such flexible displays is an entirely separate technological hurdle. We use flexible displays by augmenting the appearance of passives surfaces with image projection. This allows us to combine the flexibility and minimal weight of plain paper or fabric with the dynamic content capabilities of a computer display creating a coherent and fully functional user experience.

Foldable Displays are re-shapeable. Semi-rigid hand-held displays using an affordable, low-cost location tracking technology. The user can gracefully increase or decrease the viewing area simply by unfolding or folding the display as shown in Figure 2.12.

The displays react to the user's movement, expansion state, orientation, and input styling. Foldable displays offer the ability store large display surfaces in a small volume allowing users to quickly expand and collapse them when convenient. The current state of projection technology is not yet practical for supporting spatially augmented reality in truly mobile scenarios. As tracking technologies improve, surface modeling assumptions can be reduced improving robustness. With the emerging flexible display and portable projection technologies, foldable designs may become practical in portable consumer electronics in the

near future.

The foldable displays implemented in our project use fixed markers. Two different binary scale markers are chosen at random and the code is designed to identify each pattern using flags. Using two different marker types we can display 4 or more different images on these markers by re-orientation and expansion of markers. The expansion of the markers results in both markers being detected at the same time in different orientations-horizontal and vertical. The desired code detects the orientation and displays the required image. Moreover the program can include the possibility of detecting each marker in different orientations of itself provided the pattern to be detected allows. As a result, by a mere movement or rotation of a marker we can a lot of images.

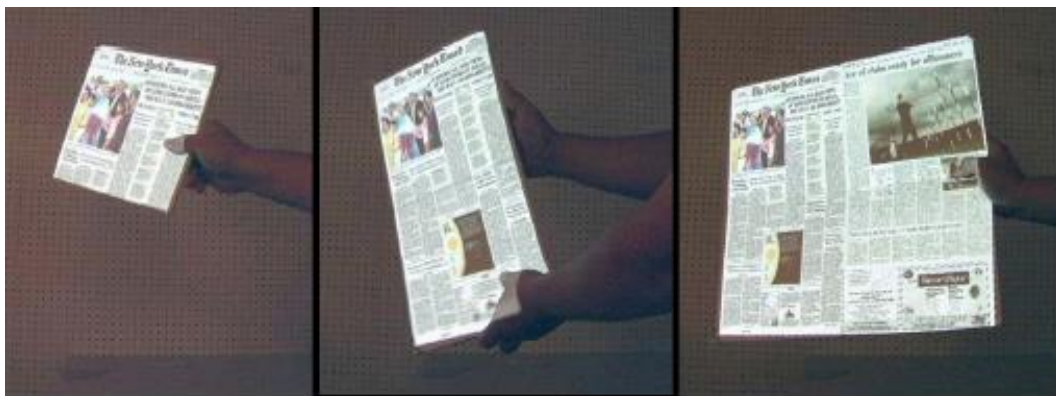


Fig.2. 12 : An illustration of the proposed Foldable Display model [5]

# BASICS OF IMAGE PROCESSING

In this chapter some technical aspects necessary to extract marker from environment and to place object on marker are discussed. We will first start with Edge detection as it is the first step towards extraction of marker.

## 3.1 Edge Detection Techniques:

Different Edge Detection Techniques:

1. Sobel's operator
2. Robert's cross operator
3. Prewitt's operator
4. Laplacian of Gaussian
5. Canny Edge detection

The operators used in some of the techniques are specified below in fig.3.1: [6]

-1	-1	-1	-1	0	1
0	0	0	-1	0	1
1	1	1	-1	0	1

Prewitt

-1	-2	-1	-1	0	1
0	0	0	-2	0	2
1	2	1	-1	0	1

Sobel

-1	0	0	-1
0	1	1	0

Roberts

Fig.3. 1 : Prewitt, Sobel and Roberts operators [6]

### 3.1.1 Canny Edge Detection Algorithm: [7]

The Canny edge detection algorithm is known to many as the optimal edge detector. Canny's intentions were to enhance the many edge detectors already out at the time he started his work. He was very successful in achieving his goal. He followed a list of criteria to improve current methods of edge detection. The first and most obvious is low error rate. It is important that edges occurring in images should not be missed and that there be no responses to non-edges. The second criterion is that the edge points be well localized. In other words, the distance between the edge pixels as found by the detector and the actual edge is to be at a minimum. A third criterion is to have only one response to a single edge. The block diagram explaining the working of canny edge detector is as shown below in fig. 3.2

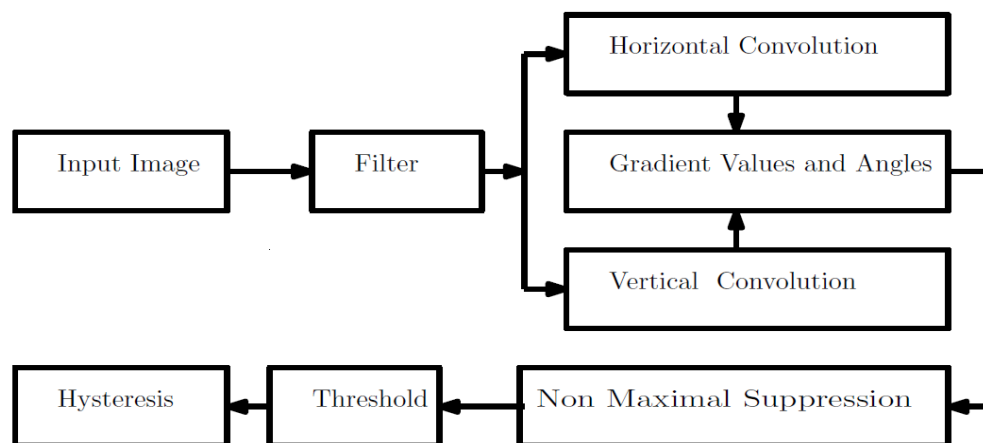


Fig.3. 2 : Block diagram for Canny edge detection [8]

This was implemented because the first two were not substantial enough to completely eliminate the possibility of multiple responses to an edge.

#### Step 1:-

In order to implement the canny edge detector algorithm, a series of steps must be followed. The first step is to filter out any noise in the original image before trying to locate and detect any edges. And because the Gaussian filter can be computed using a simple mask, it is used exclusively in the Canny algorithm. Once a suitable mask has been calculated, the Gaussian smoothing can be performed using standard convolution methods. A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The larger the width of the Gaussian mask, the

lower is the detector's sensitivity to noise. The localization error in the detected edges also increases slightly as the Gaussian width is increased.

Step 2:-

After smoothing the image and eliminating the noise, the next step is to find the edge strength by taking the gradient of the image. The Sobel operator performs a 2-D spatial gradient measurement on an image. Then, the approximate absolute gradient magnitude (edge strength) at each point can be found. The Sobel operator [9] showed in fig.3.3 uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). They are shown below:

-1	0	+1
-2	0	+2
-1	0	+1

$G_x$

+1	+2	+1
0	0	0
-1	-2	-1

$G_y$

Fig.3. 3 : Masks used by Sobel operator [7]

The magnitude, or edge strength, of the gradient is then approximated using the formula:

$$|G| = |G_x| + |G_y| \quad (1)$$

Step 3:-

The direction of the edge is computed using the gradient in the x and y directions. However, an error will be generated when sumX is equal to zero. So in the code there has to be a restriction set whenever this takes place. Whenever the gradient in the x direction is equal to zero, the edge direction has to be equal to 90 degrees or 0 degrees, depending on what the value of the gradient in the y-direction is equal to. If GY has a value of zero, the edge direction will equal 0 degrees. Otherwise the edge direction will equal 90 degrees. The formula for finding the edge direction is just:

$$\text{Theta} = \text{invtan} (G_y / G_x) \quad (2)$$

Step 4:-

Once the edge direction is known, the next step is to relate the edge direction to a direction that can be traced in an image. So if the pixels of a 5x5 image are aligned as follows in fig. 3.4:

X	X	X	X	X
X	X	X	X	X
X	X	a	X	X
X	X	X	X	X
X	X	X	X	X

Fig.3. 4 : A block of 5 X 5 pixels [7]

Then, it can be seen by looking at pixel "a", there are only four possible directions when describing the surrounding pixels - 0 degrees (in the horizontal direction), 45 degrees (along the positive diagonal), 90 degrees (in the vertical direction), or 135 degrees (along the negative diagonal). So now the edge orientation has to be resolved into one of these four directions depending on which direction it is closest to (e.g. if the orientation angle is found to be 3 degrees, make it zero degrees). Think of this as taking a semicircle and dividing it into 5 regions.

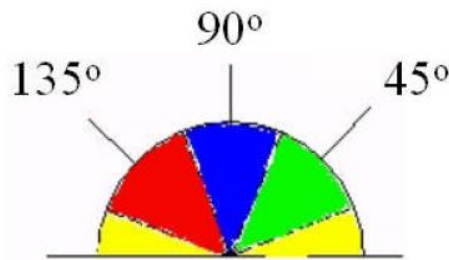


Fig.3. 5 : Edge direction detection [7]

Therefore, any edge direction falling within the yellow range (0 to 22.5 & 157.5 to 180 degrees) is set to 0 degrees. Any edge direction falling in the green range (22.5 to 67.5 degrees) is set to 45 degrees. Any edge direction falling in the blue range (67.5 to 112.5 degrees) is set to 90 degrees. And finally, any edge direction falling within the red range (112.5 to 157.5 degrees) is set to 135 degrees as seen in fig. 3.5.

#### Step 5:-

After the edge directions are known, non-maximum suppression now has to be applied. Non-maximum suppression is used to trace along the edge in the edge direction and suppress any pixel value (sets it equal to 0) that is not considered to be an edge. This will give a thin line in the output image.

Step 6:-

Finally, hysteresis [10] is used as a means of eliminating streaking. Streaking is the breaking up of an edge contour caused by the operator output fluctuating above and below the threshold. If a single threshold,  $T_1$  is applied to an image, and an edge has an average strength equal to  $T_1$ , then due to noise, there will be instances where the edge dips below the threshold. Equally it will also extend above the threshold making an edge look like a dashed line. To avoid this, hysteresis uses 2 thresholds, a high and a low.

- a. If a pixel gradient is higher than the *upper* threshold, the pixel is accepted as an edge
- b. If a pixel gradient value is below the *lower* threshold, then it is rejected.
- c. If the pixel gradient is between the two thresholds, then it will be accepted only if it is connected to a pixel that is above the *upper* threshold.

Visual Comparison of various edge detection Algorithms:



Fig.3. 6 : Image used for edge detection analysis [7]

Edge detection of all four types was performed on Fig.3.6 [11]. Canny yielded the best results. This was expected as Canny edge detection accounts for regions in an image. Canny yields thin lines for its edges by using non-maximal suppression. Canny also utilizes hysteresis with thresholding.

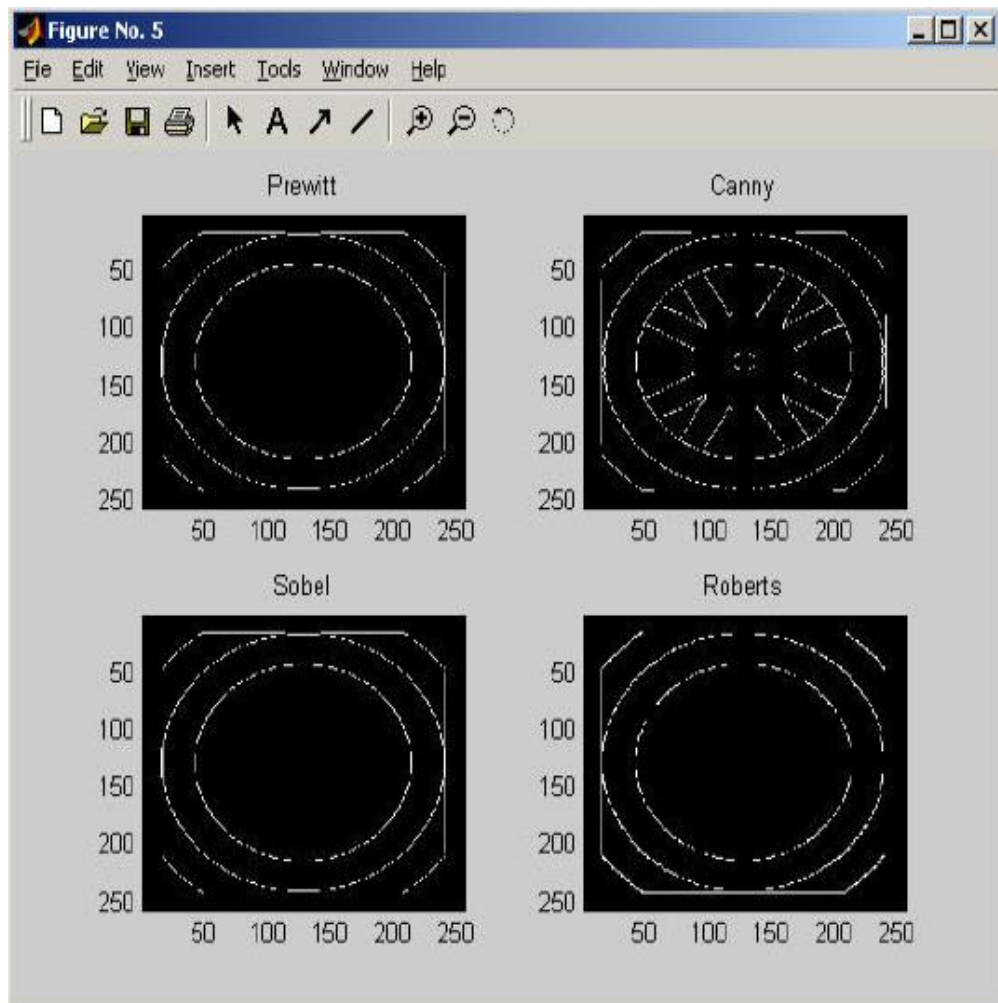


Fig.3. 7 : Results of edge detection on Fig.3.6 Canny had the best results [7]

### 3.2 Thresholding [12]

With the growth of image processing applications, image segmentation has become an important part of image processing. The simplest method to segment an image is thresholding. Using the thresholding method, segmentation of an image is done by fixing all pixels whose intensity values are more than the threshold to a foreground value. The remaining pixels are set to a background value.

Thresholding is used to create a binary image from a grayscale image. The binary image is then partitioned into multiple regions (sets of pixels, each of them representing a single skeletal element). The thresholding operation is applied to a single-channel (grayscale) image. The threshold level (the gray value which separates the pixels classified as object pixels from background pixels) is designated using two algorithms. Otsu's (1979) method for



image thresholding is recommended. This algorithm is one of the better solutions for threshold selection that are available nowadays. Otsu's method is used by the described program by default. Another method utilizes the OpenCV function `cvAdaptiveThreshold`, which applies an adaptive threshold to an image. Two parameters of the function (`int block_size`, `double param1`) are set manually by the user. This method is therefore highly subjective and it is recommended to use it with caution. The adaptive threshold proved satisfactory, however, when analyzing skeletons containing voids, fractures and borings filled with calcite cement or sediment.

### 3.2.1 Adaptive threshold

In image processing, thresholding is applied to obtain binary images from grayscale images. Adaptive thresholding is found to be better as compared to conventional thresholding technique. In an image, some parts remain under more shadow and often illuminations also affect the image. In conventional thresholding method, a global or standard threshold value is taken as mean value. In an image, if darker part or pixels contain a value larger than the threshold value, then that part of the image appears in the foreground. Similarly if the value is less than the threshold value, then that pixel or part appears in the background. [12]

Binary image is the resultant image of adaptive thresholding as it depicts the differences between different threshold values. The white region describes values less than threshold and black region describes values greater than threshold value. [13]

The function `cvAdaptiveThreshold` calculates the weighted mean in a rectangular neighbourhood of a pixel and takes this value, subtracted by a selected value, is the local threshold. The weighting is given by a Gaussian mask of the size of the neighbourhood, which is given by the parameter Window Size, where the actual size of the neighbourhood, in pixels, is neighbourhood [13]

$$\text{Size} = (2\text{windowSize} + 1)^2 - (2\text{windowSize} + 1) \quad (3)$$

The constant subtracted from the weighted mean to calculate the local threshold is given by the parameter 'Threshold'.

One problem with the approach described above is that it depends on the characteristics of

the neighbourhood, e.g. near the borders of a text block where most of the area in the neighbourhood is pure background, the threshold tends to be higher than in the middle of a text block. As we know that we always have to distinguish two classes of pixels (namely dark characters and light pixels from the background) and pixels from each class can be assumed to have a locally similar intensity. [13]

The `cvAdaptiveThreshold()` function:

```
Void cvAdaptiveThreshold(  
    CvArr* src,  
    CvArr* dst,  
    double max_val,  
    int adaptive_method = CV_ADAPTIVE_THRESH_MEAN_C  
    int threshold_type = CV_THRESH_BINARY,  
    int block_size = 3,  
    double param1 = 5  
);
```

`cvAdaptiveThreshold()` allows for two different adaptive threshold types depending on the settings of `adaptive_method`. In both cases the adaptive threshold  $T(x, y)$  is set on a pixel-by-pixel basis by computing a weighted average of the  $b$ -by- $b$  region around each pixel location minus a constant, where  $b$  is given by `block_size` and the constant is given by `param1`. If the method is set to `CV_ADAPTIVE_THRESH_MEAN_C`, then all pixels in the area are weighted equally. If it is set to `CV_ADAPTIVE_THRESH_GAUSSIAN_C`, then the pixels in the region around  $(x, y)$  are weighted according to a Gaussian function of their distance from that center point. [2]

The adaptive threshold technique is useful when there are strong illumination or reflectance gradients that you need to threshold relative to the general intensity gradient. This function handles only single-channel 8-bit or floating-point images, and it requires that the source and destination images be distinct.

### **3.2.2 Otsu's Thresholding [14]**

In Otsu's method, the algorithm assumed that the image was classified into two classes of pixels: foreground and background. Afterwards, these two classes are separated in order to

minimize their respective inter-class variance, which led to computation of the optimum threshold. This method is always independent of the probability density function.

However, bimodal distribution of gray level values was assumed by this method. This assumption was the main drawback of this method. It also failed to determine the threshold value when the classes are very unequal.

Let the pixels of a given picture be represented in  $L$  gray levels  $[1, 2, \dots, L]$ .

The number of pixels at level  $i$  is denoted by  $n_i$  and the total number of pixels by

$$N = n_1 + n_2 + \dots + n_L \quad (4)$$

In order to simplify the discussion, the gray-level histogram is normalized and regarded as a probability distribution:

$$p_i = n_i/N, \quad p_i \geq 0, \quad \sum_{i=1}^L p_i = 1. \quad (5)$$

Now suppose that we dichotomize the pixels into two classes  $C_0$  and  $C_1$  (background and objects, or vice versa) by a threshold at level  $k$ ;  $C_0$  denotes pixels with levels  $[1, \dots, k]$ , and  $C_1$  denotes pixels with levels  $[k+1, \dots, L]$ . Then the probabilities of class occurrence and the class mean levels, respectively, are given by

$$\omega_0 = \Pr(C_0) = \sum_{i=1}^k p_i = \omega(k) \quad (6)$$

$$\omega_1 = \Pr(C_1) = \sum_{i=k+1}^L p_i = 1 - \omega(k) \quad (7)$$

and

$$\mu_0 = \sum_{i=1}^k i \Pr(i|C_0) = \sum_{i=1}^k ip_i/\omega_0 = \mu(k)/\omega(k) \quad (8)$$

$$\mu_1 = \sum_{i=k+1}^L i \Pr(i|C_1) = \sum_{i=k+1}^L ip_i/\omega_1 = \frac{\mu_T - \mu(k)}{1 - \omega(k)}, \quad (9)$$

where

$$\mu(k) = \sum_{i=1}^k ip_i \quad (10)$$

are the zeroth- and the first-order cumulative moments of the histogram up to the  $k^{\text{th}}$  level, respectively, and

$$\mu_T = \mu(L) = \sum_{i=1}^L i p_i \quad (11)$$

is the total mean level of the original picture. We can easily verify the following relation for any choice of  $k$ :

$$\omega_0 \mu_0 + \omega_1 \mu_1 = \mu_T, \quad \omega_0 + \omega_1 = 1. \quad (12)$$

The class variances are given by

$$\sigma_0^2 = \sum_{i=1}^k (i - \mu_0)^2 \Pr(i|C_0) = \sum_{i=1}^k (i - \mu_0)^2 p_i / \omega_0 \quad (13)$$

$$\sigma_1^2 = \sum_{i=k+1}^L (i - \mu_1)^2 \Pr(i|C_1) = \sum_{i=k+1}^L (i - \mu_1)^2 p_i / \omega_1. \quad (14)$$

These require second-order cumulative moments (statistics).

In order to evaluate the "goodness" of the threshold (at level  $k$ ), we shall introduce the following discriminant criterion measures (or measures of class separability) used in the discriminant analysis

$$\lambda = \sigma_B^2 / \sigma_W^2, \quad \kappa = \sigma_T^2 / \sigma_W^2, \quad \eta = \sigma_B^2 / \sigma_T^2, \quad (15)$$

where

$$\sigma_W^2 = \omega_0 \sigma_0^2 + \omega_1 \sigma_1^2 \quad (16)$$

$$\begin{aligned} \sigma_B^2 &= \omega_0 (\mu_0 - \mu_T)^2 + \omega_1 (\mu_1 - \mu_T)^2 \\ &= \omega_0 \omega_1 (\mu_1 - \mu_0)^2 \end{aligned} \quad (17)$$

and

$$\sigma_T^2 = \sum_{i=1}^L (i - \mu_T)^2 p_i \quad (18)$$

are the within-class variance, the between-class variance, and the total variance of levels, respectively. Then our problem is reduced to an optimization problem to search for a threshold  $k$  that maximizes one of the object functions (the criterion measures).

This standpoint is motivated by a conjecture that well thresholded classes would be separated in gray levels, and conversely, a threshold giving the best separation of classes in gray levels would be the best threshold. The discriminant criteria maximizing A, K, and q, respectively, for k are, however, equivalent to one another;

$$\sigma_W^2 + \sigma_B^2 = \sigma_T^2. \quad (19)$$

The optimal threshold  $k^*$  that maximizes t, or equivalently maximizes  $a_2$  is selected in the following sequential search by using the simple cumulative quantities

$$\eta(k) = \sigma_B^2(k)/\sigma_T^2$$

$$\sigma_B^2(k) = \frac{[\mu_T \omega(k) - \mu(k)]^2}{\omega(k)[1 - \omega(k)]}$$

and the optimal threshold is

$$\sigma_B^2(k^*) = \max_{1 \leq k < L} \sigma_B^2(k). \quad (20)$$

A method to select a threshold automatically from a gray level histogram has been derived from the viewpoint of discriminant analysis. This directly deals with the problem of evaluating the goodness of thresholds. An optimal threshold (or set of thresholds) is selected by the discriminant criterion; namely, by maximizing the discriminant measure q (or the measure of separability of the resultant classes in gray levels). The proposed method is characterized by its nonparametric and unsupervised nature of threshold selection and has the following desirable advantages.

- 1) The procedure is very simple; only the zeroth and the first order cumulative moments of the gray-level histogram are utilized.
- 2) A straightforward extension to multithresholding problems is feasible by virtue of the criterion on which the method is based.
- 3) An optimal threshold (or set of thresholds) is selected automatically and stably, not based on the differentiation (i.e., a local property such as valley), but on the integration (i.e., a global property) of the histogram.
- 4) Further important aspects can also be analyzed (e.g., estimation of class mean levels,

evaluation of class separability, etc.).

5) The method is quite general: it covers a wide scope of unsupervised decision procedure.

The range of its applications is not restricted only to the thresholding of the gray-level picture, such as specifically described in the foregoing, but it may also cover other cases of unsupervised classification in which a histogram of some characteristic (or feature) discriminative for classifying the objects is available. Taking into account these points, the method suggested in this correspondence may be recommended as the most simple and standard one for automatic threshold selection that can be applied to various practical problems.

A comparison is drawn out between the outputs of adaptive thresholding and Otsu thresholding which is shown in Fig 3.8 and Fig 3.9. It is clear that the usage of these functions depends on the environment and context it is required to be used in.

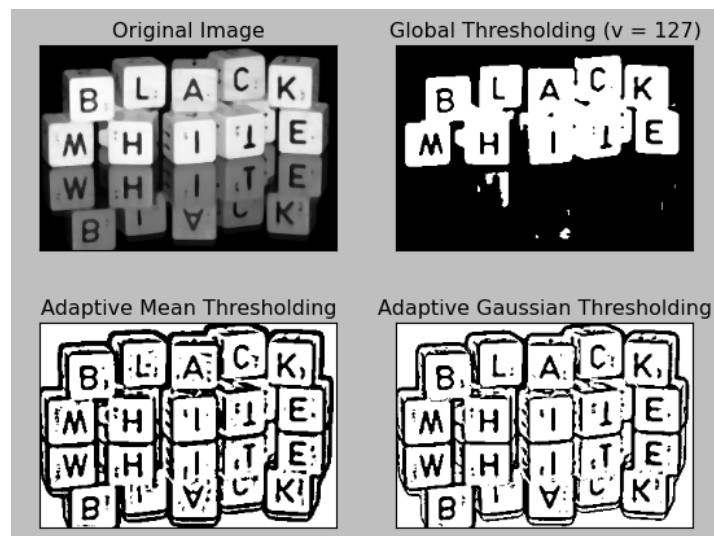


Fig.3. 8 : Comparison of different thresholding methods

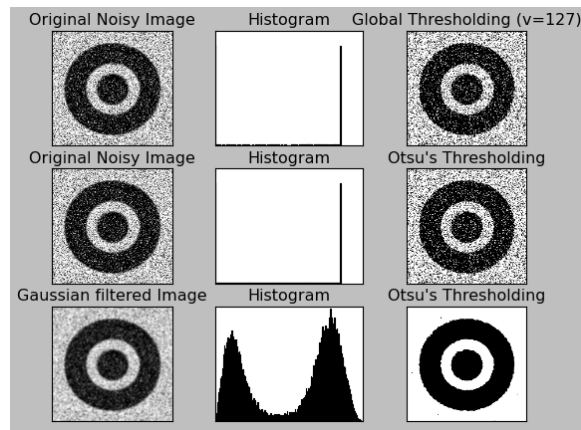


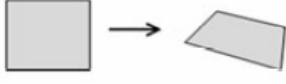
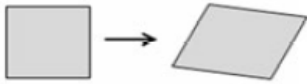
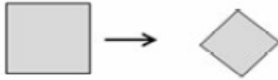
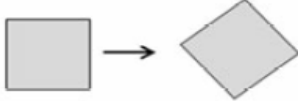
Fig.3. 9 : Histogram of different thresholding method

### 3.3 Homography [15]

In computer vision, planar homography is the projective mapping from one plane to another. Thus, the mapping of points on a two-dimensional planar surface to the image of the camera is an example of planar homography.

In AR as in 3D computer vision in general, perspective projections play an essential role. Geometric transformations form a hierarchy of subsets. Homogeneous coordinates provide a framework for geometric operations in projective space. Euclidean geometry is a special case of projective geometry with more restrictions. Thus, it is possible to use homogeneous presentation in Euclidean geometry as well, if the operations are restricted to Euclidean ones. Accordingly, homogeneous presentation can also be used in affine and similarity transformations. Thus, all geometric transformations and their combinations can be presented with matrix multiplications using homogeneous coordinates. In a projective transform, only collinearity, cross-ratios and incidences remain invariant. Affine transformations also preserve parallelism and the ratios of areas. Similarity transforms preserve angles and length ratios. Euclidean transformations preserve angles, lengths and areas. Table 1 identifies the degrees of freedom and the corresponding transforms required are specified.

Table 1: Properties of different transformation spaces [16]

Transform	DOF	Invariants	
Perspective (Projective)	8DOF	Collinearity, Cross-ratios, Incidences	
Affine	6DOF	Parallelism, Ratios of areas	
Similarity	4DOF	Angles, Length ratios	
Linear (Euclidean)	3DOF	Angles, Lengths, Areas	

It is possible to express this mapping in terms of matrix multiplication if homogeneous coordinates to express both the viewed point  $Q$  and the point  $q$  on the imager to which  $Q$  is mapped.

$$\tilde{Q} = [X \ Y \ Z \ 1]^T$$

$$\tilde{q} = [x \ y \ 1]^T$$

$$\tilde{q} = sH\tilde{Q} \quad (21)$$

The parameter  $s$ , which is an arbitrary scale factor (intended to make explicit that the homography is defined only up to that factor). It is conventionally factored out of  $H$ . With a little geometry and some matrix algebra, the transformation matrix can be solved. The most important observation is that  $H$  has two parts: the physical transformation, which essentially locates the object plane we are viewing; and the projection, which introduces the camera intrinsics matrix.



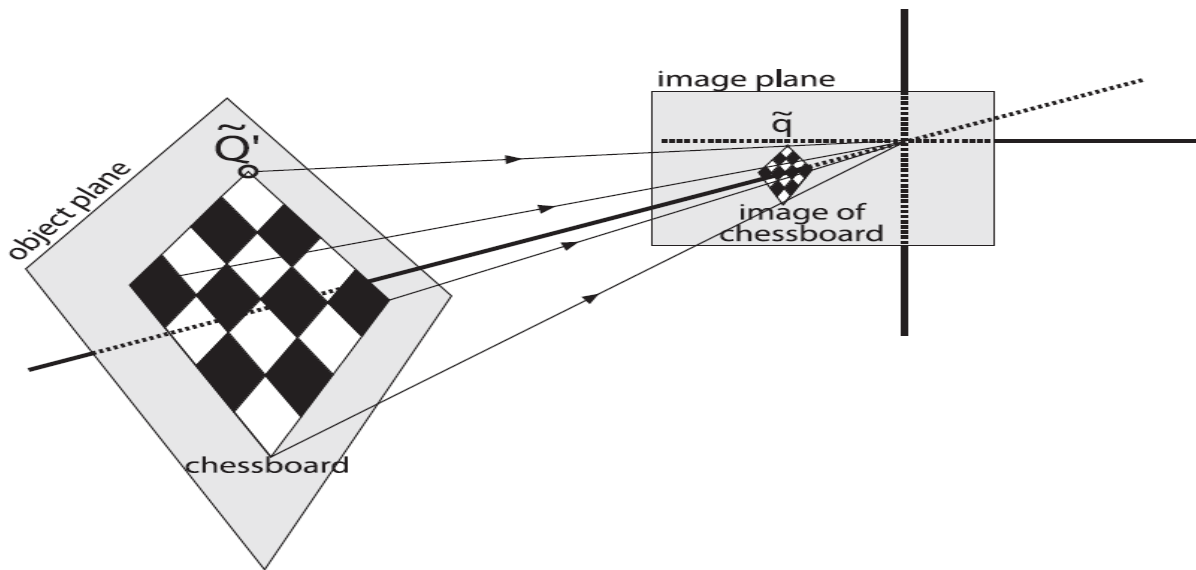


Fig.3. 10 : View of a planar object as described by homography: a mapping—from the object plane to the image plane—that simultaneously comprehends the relative locations of those two planes as well as the camera projection matrix [15]

### 3.3.1 Affine and Perspective Transformations

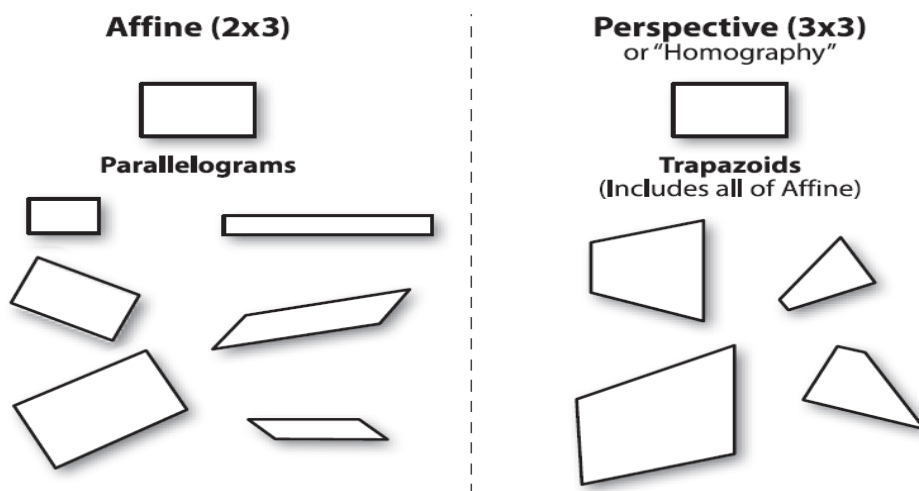


Fig.3. 11 : Affine and perspective transformations [15]

The *perspective transformation* is closely related to the *perspective projection*. Fig 3.11 explains the difference in the usage of affine and perspective transforms respectively. The perspective projection maps points in the three-dimensional physical world onto points on the two-dimensional image plane along a set of projection lines that all meet at a single point called *the center of projection*. The perspective transformation, which is a specific kind of

*homography*, relates two different images that are alternative projections of the same three-dimensional object onto two different *projective planes* (and thus, for non degenerate configurations such as the plane physically intersecting the 3D object, typically to two different centers of projection). `cvPerspectiveTransform()` performs the perspective transform a list of points whereas `cvWarpPerspective()` carries out the perspective transform on the whole image. In case of AR the process which is undertaken is known as the Predict-Match – Update cycle shown below in Fig 3.12:

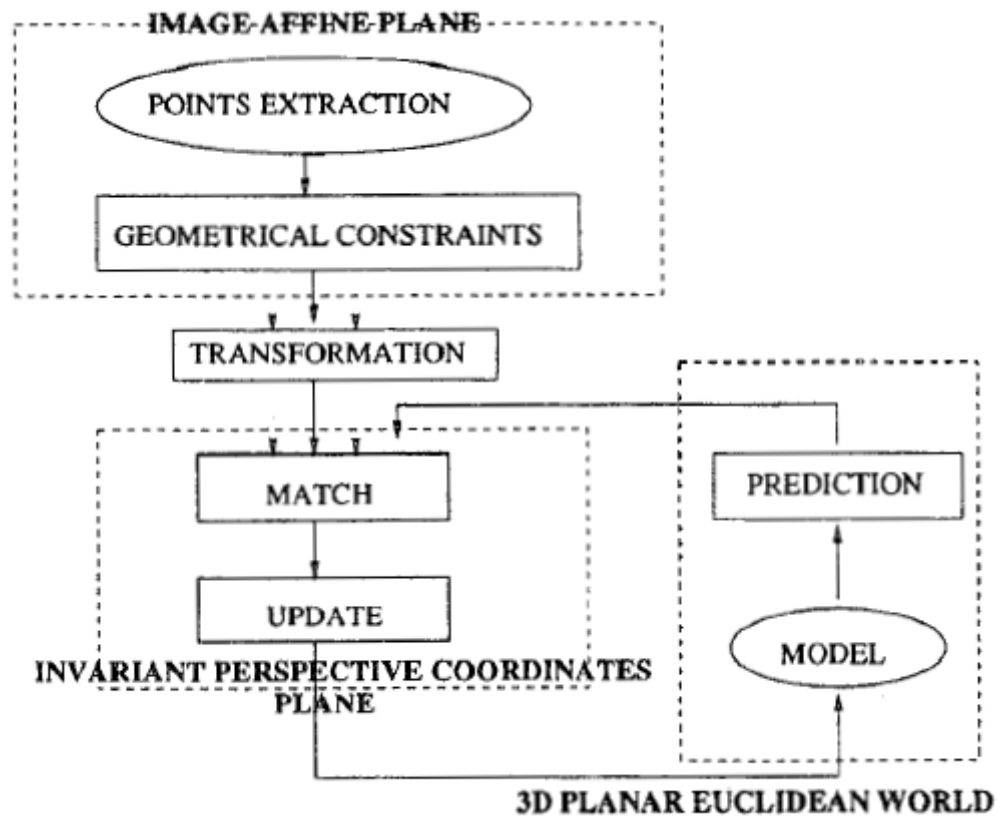


Fig.3. 12 : Predict-Match –Update cycle [17]

# MARKER BASED AR

## 4.1 Schematics of proposed algorithm:

The proposed algorithm required for the overlay of marker over the foldable display is explained in the block diagram below in fig 4.1.3. The efficient overlay of the images depends on the light conditions in the surrounding environment and could affect the performance.

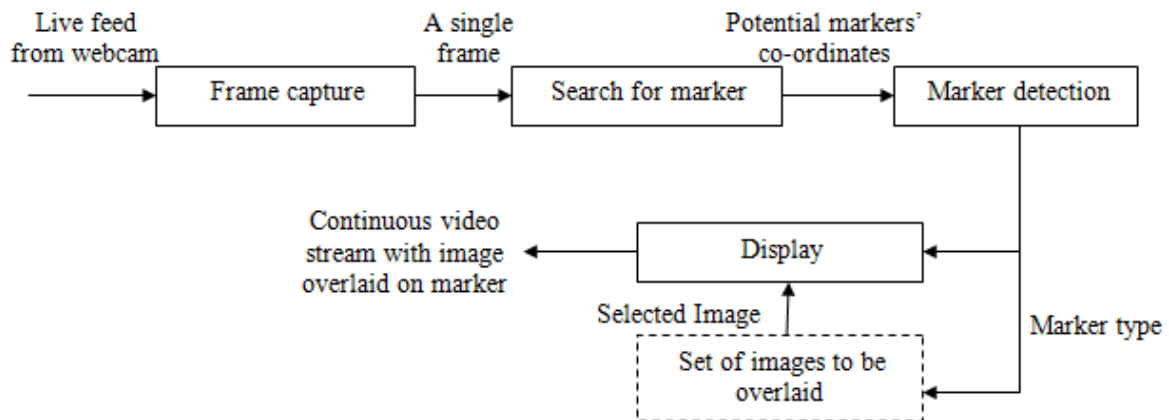


Fig.4. 1 : Schematic of main program

### 4.1.1 Specifications:

Frame capture size: Height : 480 pixels

Width : 640 pixels

Depth: 8

### 4.1.2 Main program:

**Step 1.**Initialise flags for all markers and counter for marker with 0.

**Step 2.**Obtain a frame from video capture.

**Step 3.**Send this frame to subroutine to search marker.

**Step 4.**Display the frame returned by Display subroutine.

### 4.1.3 Search marker module:

**Step 1.**Perform edge detection as the first step to extract marker from environment.(Fig. 4.3 , Fig. 4.4)

**Step 2.** Perform contouring to extract rectangular objects from environment.

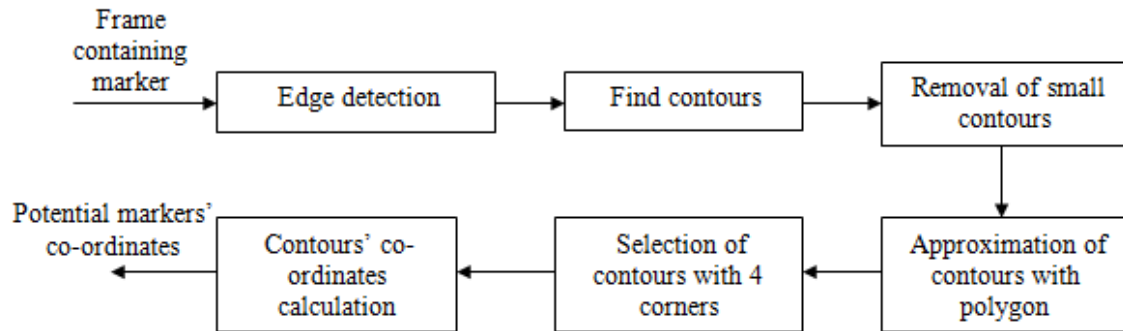


Fig.4. 2 : Schematic of 'search marker' module

**Step 3.** Approximate each contour with a polygon.

**Step 4.** Chose every polygon that has 4 corners and obtain its coordinates to check whether that contour is a marker or not.

**Step 5.** Repeat this process for each rectangular contour.

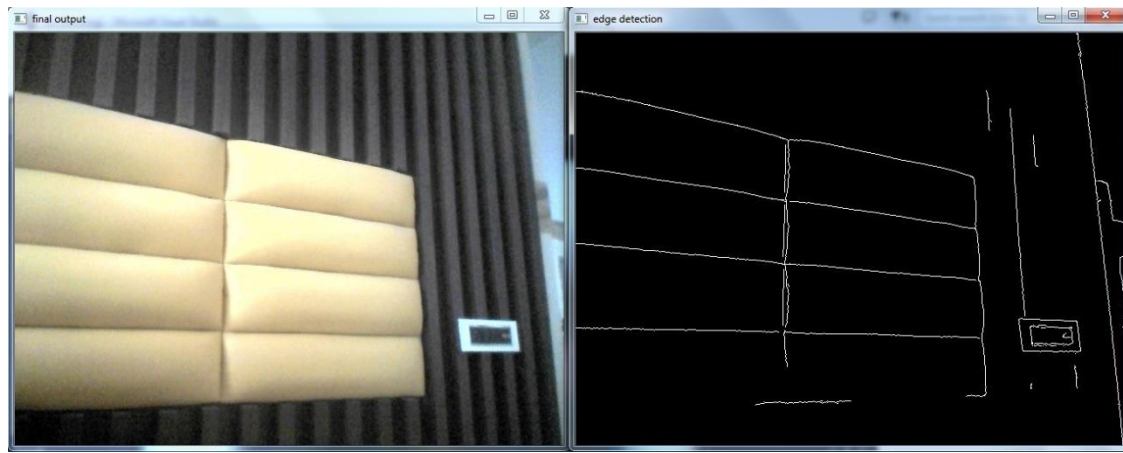


Fig.4. 3 : Edge detection in absence of marker

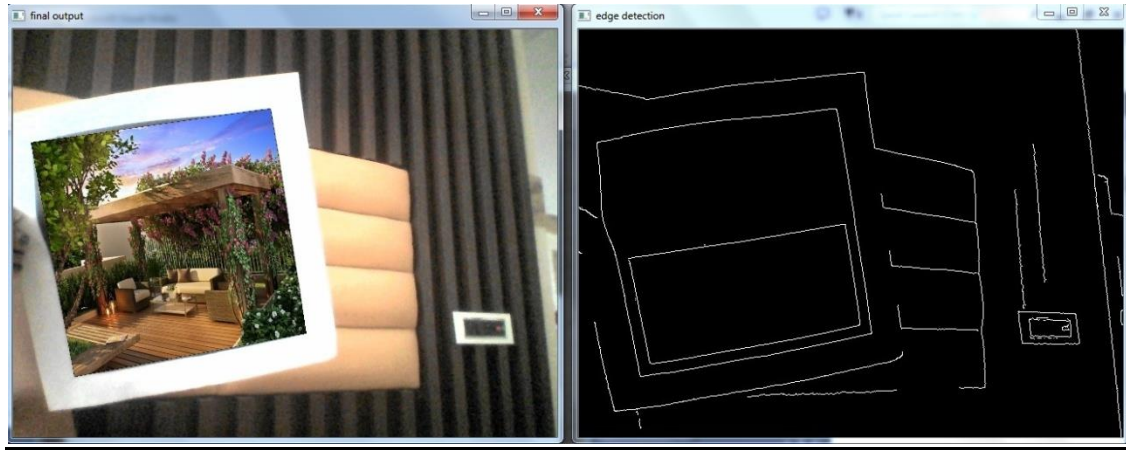


Fig.4. 4 : Edge detection in presence of marker

#### 4.1.4 Marker detection module:

The marker detection module explained in fig 4.5 includes the transformation between different planes through warp perspective transform. According to the pattern recognised the overlap is specified.

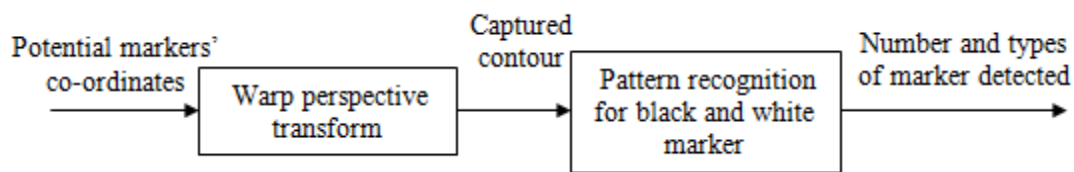


Fig.4. 5 : Schematic of 'marker detection' module

**Step 1.** Apply warp perspective transform to fit the contour to the screen. (Fig.4.6, Fig. 4.7)

**Step 2.** Convert this image to black and white image using Otsu adaptive thresholding for pattern recognition.(Fig. 4.8)

**Step 3.** Access each pixel value of the contour and decide whether that contour is marker or not.

**Step 4.** If it is a marker then set flag for that marker and increment the counter for keeping track of number of markers in the frame.

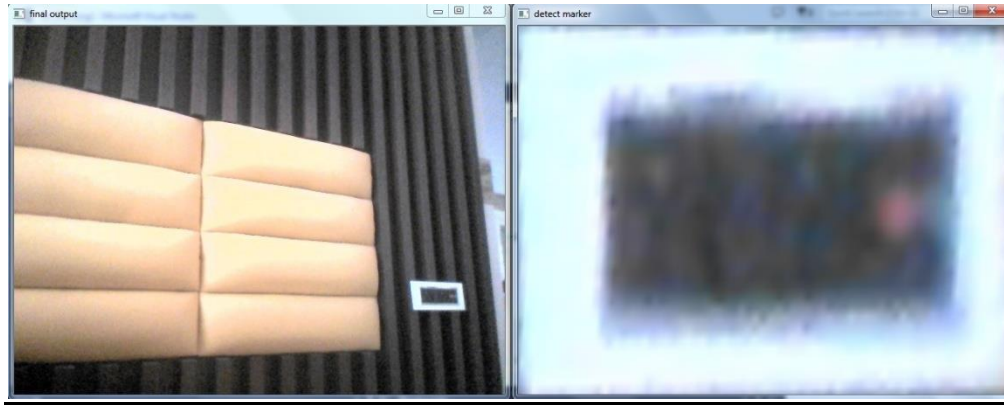


Fig.4. 6 : Contour with 4 corners; rejected as a marker during pattern recognition

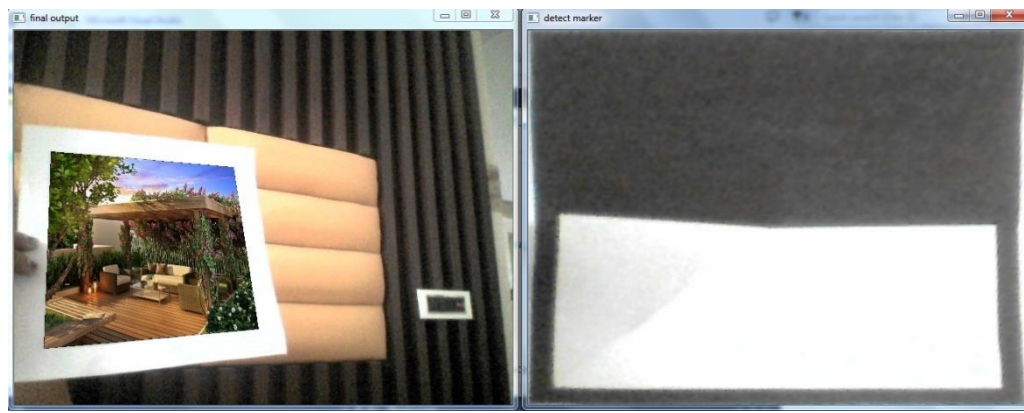


Fig.4. 7 : Marker captured for pattern recognition

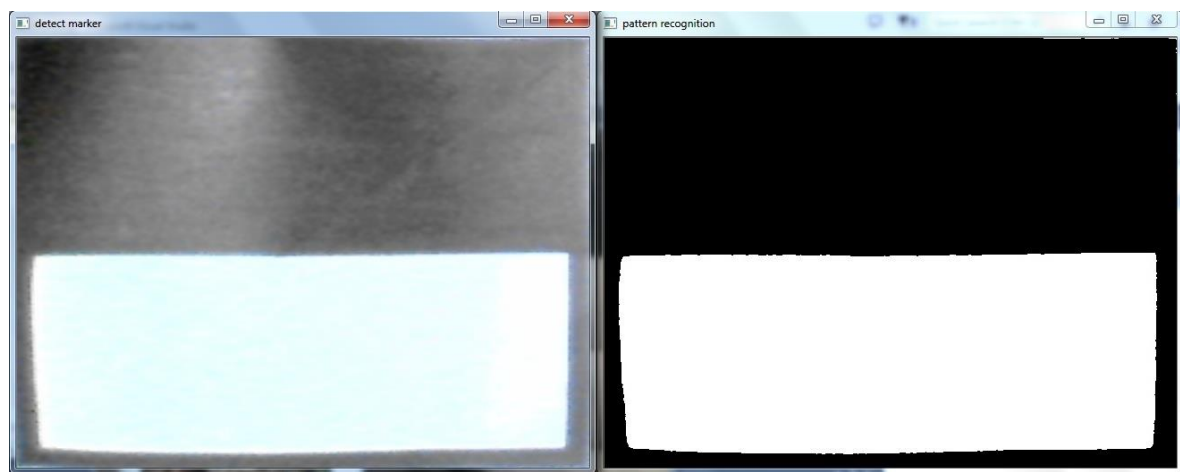


Fig.4. 8 : Conversion to black and white image for Otsu adaptive thresholding

#### 4.1.5 Display module:

The display module which is generally taken to be the foldable display in incorporated below through fig 4.9.

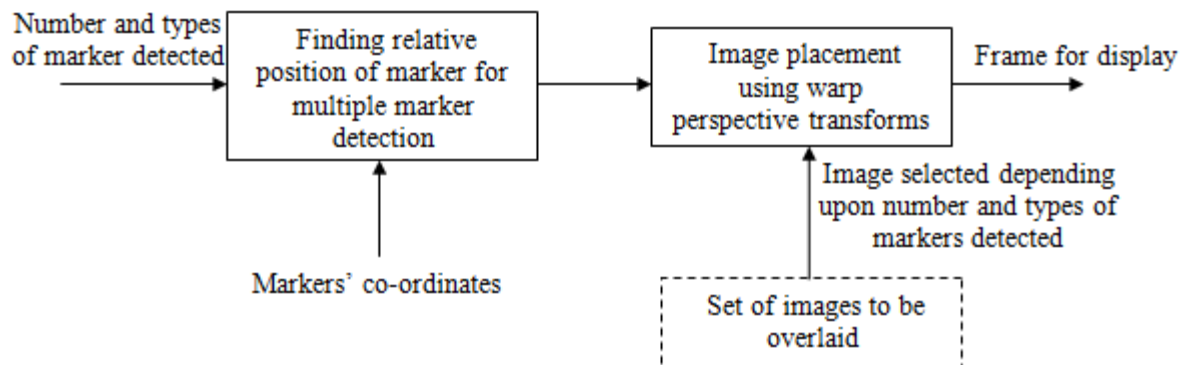


Fig.4. 9 : Schematic of 'display' module

**Step 1.** From the coordinates of the markers present, find their relative position and the coordinates of points to which image must be overlaid.

**Step 2.** Select image to be overlaid depending upon number of markers and types of markers present in the frame.

**Step 3.** Apply warp perspective transform to this image and fit it at the position of marker. At the end of step2, we get an image shown in Fig.4.10

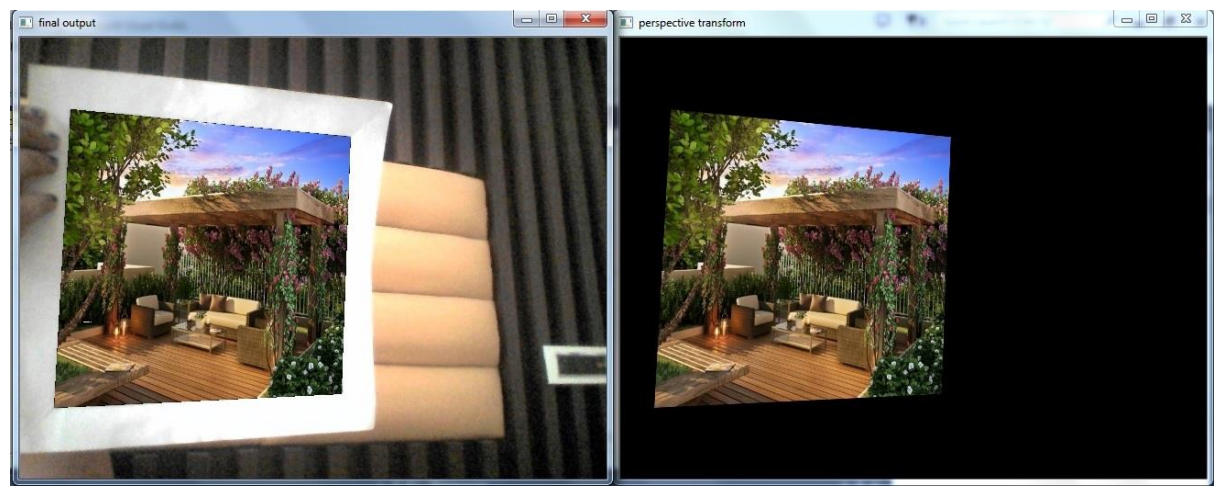


Fig.4. 10 : Placement of image on marker using warp perspective transform

**Step 4.** To mix this image with the environment, replace pixels of this image with corresponding pixels in original frame capture if the pixel is complete black.

**Step 5.** Return this image to main program for display.



## 4.1 Results:

Depending upon number and types of markers in original frame, different images are overlaid on the markers. Output for different combination of markers is shown below:

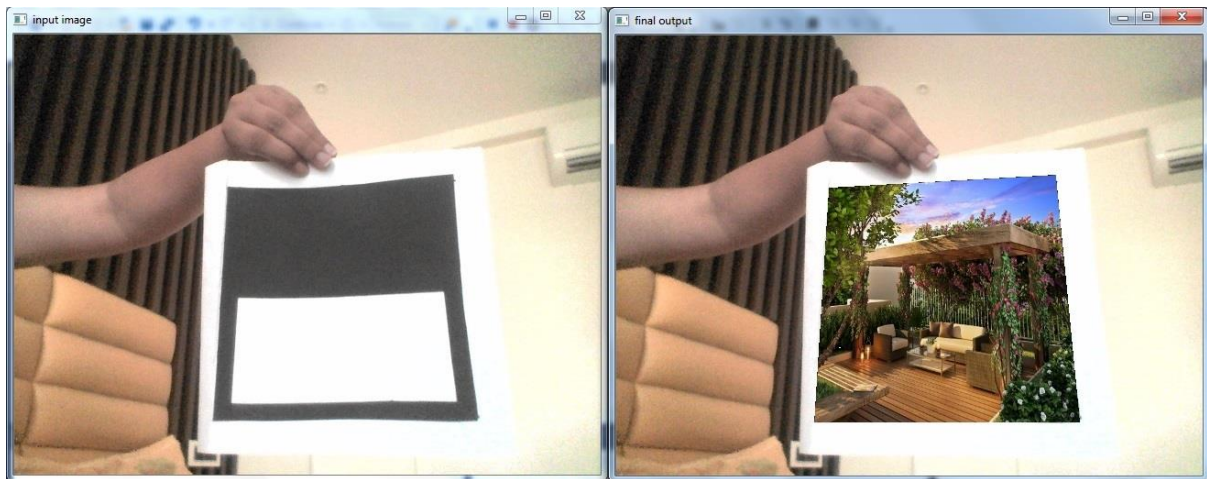


Fig.4. 11 : Ouptut for marker 1

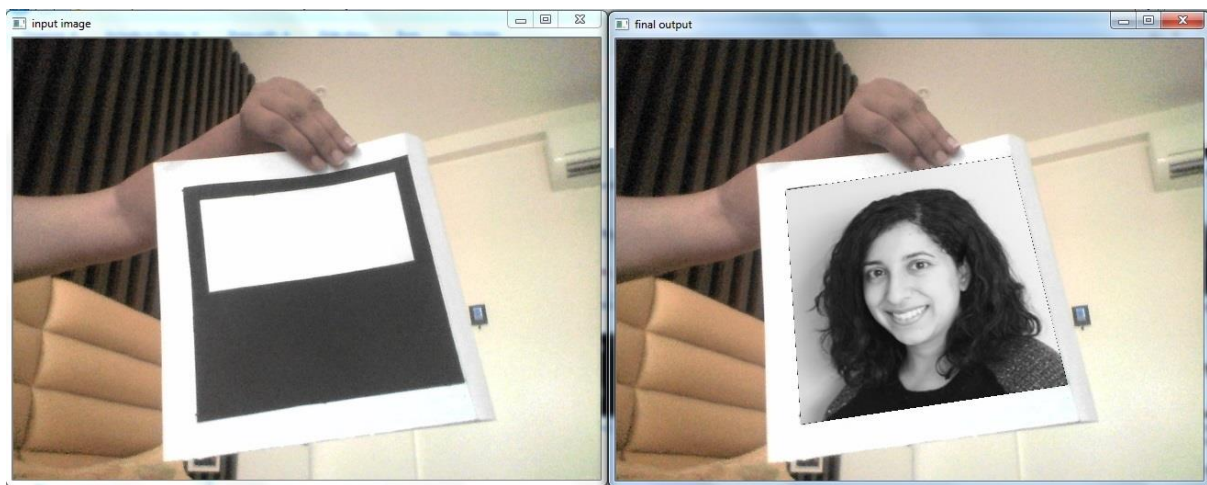


Fig.4. 12 : Output for ma88rker2



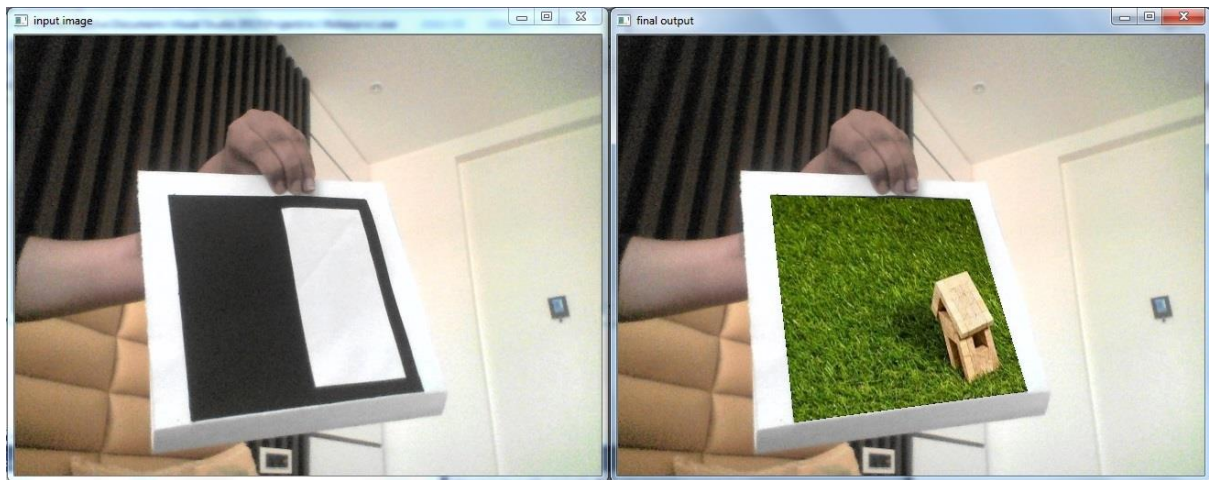


Fig.4. 13 : Output for marker3

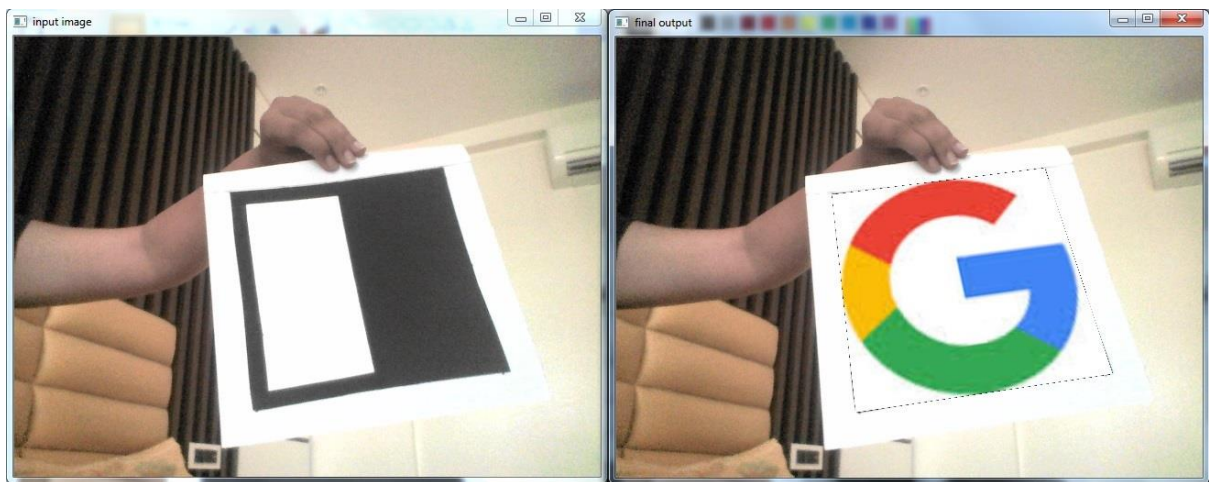


Fig.4. 14 : Output for marker4

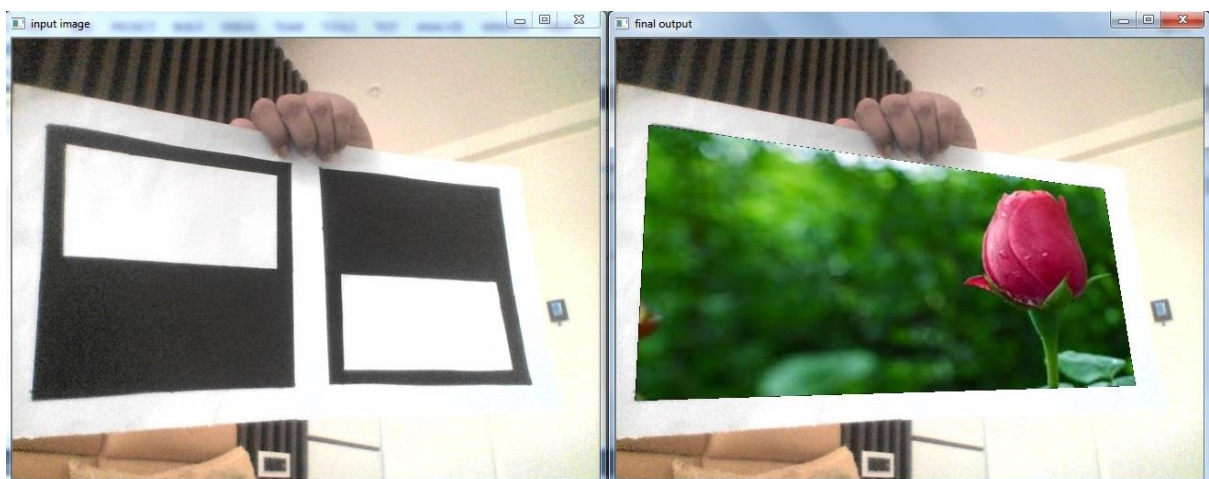


Fig.4. 15 : Output for combination of marker1 and marker2

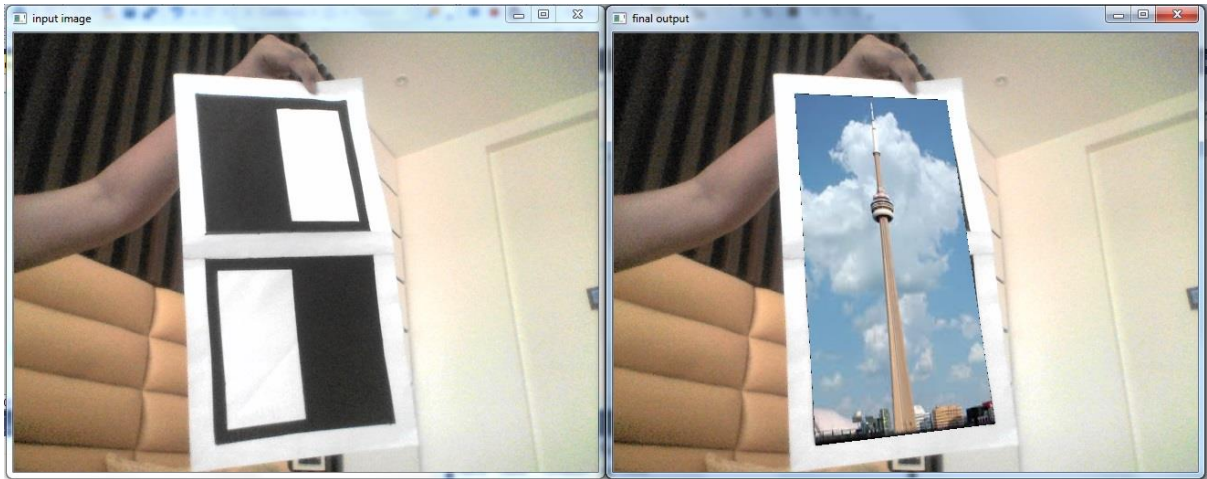


Fig.4. 16 : Output for combination of marker3 and marker4

# SHORTCOMINGS

Being a newly developed domain, augmented reality hasn't overcome all the hurdles that it faces. There are various issues faced by developers of AR apart from disparity of access which have been described in the sections below.

## **5.1 Main issues in AR application development [16]**

The augmented reality application developer needs to take into consideration several different issues: technical, application and other issues affecting the user experience. The main technological issues relate directly to the definition of augmented reality (real-time, interactive, 3D, combining real and virtual). Application issues arise from the ease of creating AR applications. Other important issues relate to user experience.

The main technological issues in augmented reality are

1. Performance
2. Interaction
3. Alignment.

The main application issues are

1. Content creation
2. Authoring

Other important issues affecting the user experience are

- 1) Visual perception
- 2) User interface
- 3) Devices
- 4) Power consumption.

An augmented reality system needs to be able to perform in real-time. Otherwise, the system may augment old or flawed information, or the augmentation may not correspond to the current state of the environment. Performance issues are characteristic to all AR algorithm and application development. Research results from other fields (e.g. image processing) are not directly applicable to AR. For instance, traditional image in painting methods do not fulfil

the real-time requirement and therefore they cannot be used for diminished reality as such. Performance is an issue especially in mobile environment where the processing power and memory are limited.

The user should be able to interact with the system naturally. The usability and the user experience are disturbed if the interaction is unnatural. The interaction needs to be natural in the user interface level.

The same holds true at the application level; the interaction between the real world objects and virtual objects needs to be smooth as well. Application needs to adapt virtual elements according to real scene, as for example in our interior design application where the user was able to adjust virtual lights easily according to real ones. At times, the application needs to remove existing objects virtually to be able to augment virtual objects on the same place.

The camera calibration needs to be correct and the tracking needs to be accurate. Otherwise, the augmented data is shifted in the real environment: the virtual overlay is in the wrong place or it flutters. People find this alignment error annoying.

The content creation is also an important aspect of application development. An application can visualize information from a database (e.g. in augmented assembly) or provide textual information (e.g. in AR browsers). Sometimes the information in database is in unsuitable format and format conversion is needed. In addition, when no database is available someone needs to create the content. Furthermore, if nice graphics are required, they need to be created to the appropriate degree of accuracy and in the right format. The same content does not suit both mobile environments and high quality visualization.

Besides content creation, authoring is a big application issue. Creation of AR applications should be brought to a non-expert non-programming level, where users can combine objects, interactions and events at a conceptual level.

Visual perception should support the purpose of the application. Some applications require realistic rendering, other applications benefit from focus and content -type highlighting of augmented objects. The user should be able to concentrate on the task, and the visual perception should sustain the task, without distracting the user. The user interface should be,

as always, easy to use and intuitive. It should support the task at hand and make the user experience smooth.

The AR application should run on the appropriate device; mobile applications on lightweight devices, high-end visualizations on larger good-quality monitors. Furthermore, the terminal device should be taken into account already at the application design stage. There is no point in implementing computationally intensive methods on mobile phones if the application would then run on a slow frame rate. Devices often play very important role in the development process. The mobile platforms is perhaps the main obstacle for wider use of mobile AR applications. Applications need to be ported mostly to each platform separately, which deprives resources from application development. Furthermore, mobile devices are an ideal platform for consumer applications; they are equipped with cameras and new models with various additional sensors; people carry them with them all the time. Likewise, in special applications where an expert operates the system, it is feasible to invest in special devices such as HMDs, 3D displays, additional sensors, etc. if they support the task.

One more aspect that significantly affects user experience is power consumption. Many applications require the user to be able to move freely, and thus wireless devices are optimal and then battery life plays a big role. A mobile application that discharges the battery in 15 minutes is unrealistic

In conclusion, the most important issue of augmented reality application development is the user experience, which is affected by all technological, application and other issues.

## **5.2 Limitations of proposed algorithm:**

The biggest limitation of any AR based project is lightening conditions. As different lightening conditions affect reflection of light from marker, this can make marker detection process more complex and erroneous. By using adaptive thresholding and Otsu thresholding, we have tried to make marker detection more robust but above certain range of lightening conditions, proposed algorithm may lead to malfunctioning.

Further, image to be superimposed should not have pure black colour. As black pixels replaced with original frame capture in last step, black pixels of markers will also be replaced by original frame capture. As shown in Fig. 4.17,

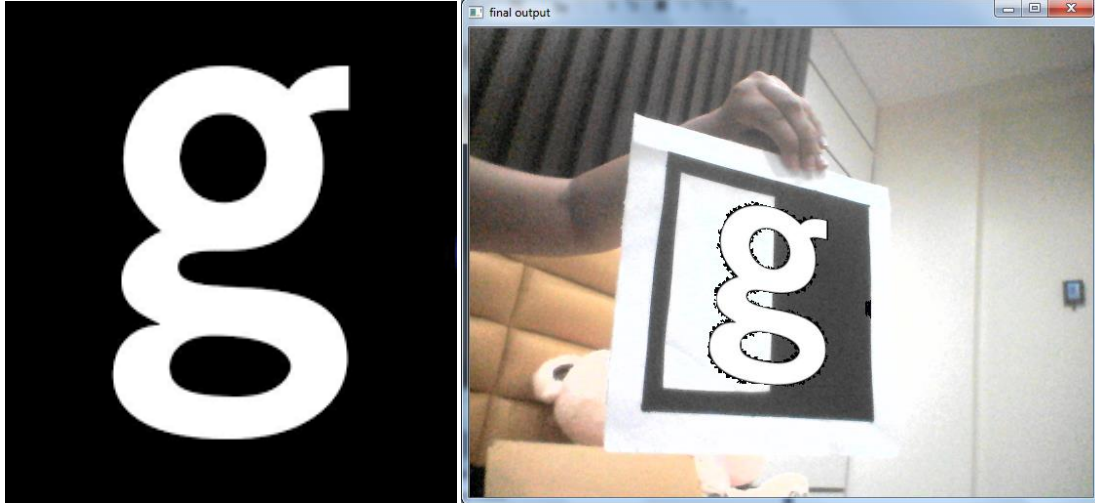


Fig.4. 17 : Black image superimposed on marker

## CONCLUSION

Augmented Reality is just starting to break out of its infancy, because of this the possible applications in the future are tremendous. It is an efficient visualization technique for on-site 3D visualization and location-based services. It is beneficial in situations where the perception skills of a human need to be enhanced. AR is applied in the fields of maintenance, assembly, building and construction, interior design, etc. Due to its “magical” nature, it is also suited to marketing and advertising purposes.

The universal development of the processing capacity and battery life of mobile devices, and the development of display devices together with cloud computing, will enable use of computationally demanding methods on mobile devices. Furthermore, customized solutions with special devices will provide high-end visualizations and instructions on specific tasks. The full potential of the technology is still on the way; it largely relies on general technological development and the development of devices supporting AR.

## References:

- [1] Heimo, Olli, Kai K. Kimppa, Seppo Helle, Timo Korkalainen, and Tapio Lehtonen. "Augmented reality-Towards an ethical fantasy?." In *Ethics in Science, Technology and Engineering, 2014 IEEE International Symposium on*, pp. 1-7. IEEE, 2014.
- [2] Gary Bradski and Adrian Kaehler, "Overview", in *Learning OpenCV*, 1<sup>st</sup> ed. Sebastopol, California, USA, O'REILLY, 2008, pp. 1-15.
- [3] Gregory Kripper, Joseph Rampolla, "The Types of Augmented Reality", in *Augmented Reality*, 2<sup>nd</sup> ed. Waltham, MA 02451, USA, SYNGRESS, 2013, pp. 29-50.
- [4] Gregory Kripper, Joseph Rampolla, "What is Augmented Reality?", in *Augmented Reality*, 2<sup>nd</sup> ed. Waltham, MA 02451, USA, SYNGRESS, 2013, pp. 1-28.
- [5] Lee, Johnny Chung, Scott E. Hudson, and Edward Tse. "Foldable Interactive Displays." (2008).
- [6] Gonzalez, Rafael C., and Richard E. Woods. "Digital image processing." (2002).
- [7] Maini, Raman, and Himanshu Aggarwal. "Study and comparison of various image edge detection techniques." *International journal of image processing (IJIP)* 3.1 (2009): 1-11.
- [8] Kong, Yinan, and M. N. Hasan. "Performance analysis of Canny's edge detection method for modified threshold algorithms." *2015 International Conference on Electrical & Electronic Engineering (ICEEE)*. IEEE, 2015.
- [9] J. Matthews. "An introduction to edge detection: The sobel edge detector," Available at <http://www.generation5.org/content/2002/im01.asp>, 2002.
- [10] J. Canny. "Finding edges and lines in image". Master's thesis, MIT, 1983.
- [11] R. A. Kirsch. "Computer determination of the constituent structure of biomedical images". *Comput. Electron. Res.*, vol. 4, pp. 315-328, 1971.
- [12] Roy, Pranab, et al. "Adaptive Thresholding: A comparative study." *Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014 International Conference on*. IEEE, 2014.
- [13] Frieder, G., and A. Lüdtk A. Miene. "Report Technical 43."
- [14] Otsu, Nobuyuki. "A threshold selection method from gray-level histograms." *Automatica* 11.285-296 (1975): 23-27.
- [15] Bradski, Gary, and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc.", 2008.
- [16] Siltanen, Sanni. *Theory and applications of marker-based augmented reality*. 2012.



- [17] Gurdjos, Pierre, and Patrice Dalle. "Tracking 3D coplanar points in the invariant perspective coordinates plane." *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*. Vol. 1. IEEE, 1996.