Declaration of Authorship

This is to certify that the work presented in this project is the outcome of the analysis and experiments carried out under the supervision of Md. Kamrul Hasan, Professor of the Department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Dhaka, Bangladesh. It is also declared that neither of this project nor any part of this project has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Authors:

Mujtahid

Syed Mujtahid Bin Tawhid

Student ID - 160041065

Istiak

Istiak Ahmad

Student ID - 160041062

Approved By:

Supervisor:

Md. Kamrul Hasan

Md. Kamrul Hasan, PhD

Professor

Department of Computer Science and Engineering (CSE)

Islamic University of Technology (IUT), OIC

Acknowledgement

We would like to express our grateful appreciation for Professor Md. Kamrul Hasan, Department of Computer Science & Engineering, IUT for being our ad-viser and mentor. His motivation, suggestions and insights for this research have been invaluable. Without his support and proper guidance this project would never have been possible. His valuable opinion, time and input provided through-out the thesis work, from rst phase of project topics introduction, subject se-lection, proposing algorithm, modi cation till the project implementation and nalization which helped us to do our project work in proper way. We are really grateful to him.

We are also grateful to Hasan Mahmud, Assistant Professor, Department of Computer Science & Engineering, IUT for his valuable inspection and suggestions on our proposal of Hand Cricket: an interactive implementaion of an indoor cricket game using augmented reality and computer vision

1 Introduction

Augmented Reality (Augmented Reality, referred to as AR for short) is one of the research hotspots of many renowned foreign universities and research institu-tions in recent years. AR technology has a wide range of applications not only in similar applications elds to VR technology, such as sophisticated weaponry, research and development on aircraft, visualization of data model, virtual train-ing, entertainment and arts and other elds, but also in medical research and anatomy training, manufacturing and maintenance of precision equipment, nav-igation of military aircraft, engineering design and remote control on robot and other areas. Besides, because of its characteristic enhancing the display output of real environment, it has more pronounced advantages than VR technology in those latter areas. Interactive games based on AR technology are games of de-velopmental direction in the future. It can unite virtual and reality, which can bring sense of deeper immersion to the players, thus greatly increasing the interest and attraction of game.Augmented reality has virtual information applied in the real world through computer technology, thus making the real environment and virtual objects overlay in real time in the same screen or space then coexist. It provides information di erent from which human can sense in general condition. It not only demonstrates the information of real world but also displays virtual information at the same time, thus having two types of information complement-ing and superimposing each other. In visualized augmented reality, the user uses the helmet display to manage the multiple synthesis of the real world and com-puter graphics, thus making us can see the real world around it.As we all know Ludo,Carrom Board,card games, etc are popular indoor games in this south Asian region.But augmented reality implementation of these games have already seeing light.We want to make something unique.That’s where our idea comes into play,it has never been implemented before. Yes it is some kind of cricket but you will play with your own hands i.e no bat,ball,wicket is required and obviously indoor.

2 Motivation

As a part of our summer semester in academic year 2019 we have to do an internship after the nal examination.We worked in battery-low interactive,a pioneer mo-bile game development company in Bangladesh.In that internship we were brie y introduced to Unity3d,a very powerful game engine which is now a days used to develop for various kinds of development works including game,animation,Ar-Vr application.Then to truly understand the power of Unity3d and collaborate with our interest of domain(augmented reality) from google scholar we came know about [9] Using Unity 3D to Facilitate Mobile Augmented Reality Game De-velopment.In this paper they clearly explain their experiences and challenges in developing Calory Battle AR, a mobile AR exergame and how their experiences can be better and challenges can be overcome with the help of unity3d develop-ment.So we got more motivated to work with unity.Another motivation came from our own University.Since the beginning of the summer semester we were privileged to choose HCI as domain of our thesis/project work under Systems and Software lab of IUT.But as we were students of CSE the we found Augmented reality more interesting to work with when we found [3].In this research they basically suggests why computational thinking skills is important and how Augmented reality can be utilized to develop computational thinking skills among students.They show it with the development of Augmented reality based game,ARQuest: A Tangible Augmented Reality Approach to Developing Computational Thinking Skills.So with that we started to motivated to make something unique with our new knowl-edge.And after our internship our motivation got stronger

3 Problem statement

In general, Video game development is a di cult and specialist activity.So generating an unique idea is not that easy task.But in today's era nothing comes by magic.Concepts are generated from concepts.Anyways let us rst talk about our idea.For clear understanding of our game idea we highly recommend watching

our video attached with this paper.But if you are a good and patient reader no worries,we have got you covered here.So the game is basically played between two person,one bowler and one batsman.More players can be in the game but at time only two will be in the game arena.How much over and how much wicket will be played that has to declared before match starts.Now at a time both bowler and batsman has to throw their hands and show some sign with their hands simulta-neously.If both the sign matches the bowler gets the wicket.Otherwise batsman gets the amount of run from the sign shown by him.Their are actually six signs in the game.One nger hand shows one run,Two nger hand two runs,three ngers hand three run,four ngers four run,six ngers six runs and thumb button showing six runs at one ball.And in case both bowler's and batsman's sign matches each other,wicket taken.Simple but pretty addicting game.So here our plan is to make an arti cial hand in your mobile with augmented reality and use hand gesture recognition to compare your sign with the arti cial hand.

4 Literature Study

4.1 Handheld Augmented Reality

Handheld Augmented Reality (HAR) is an AR system designed and embedded on a handheld device which can be hold by the user's hand. [5]A drastic improvement in handheld devices' processing power makes smartphone and tablets Personal Computer (PC)s suitable for AR system. The lower price of handheld devices has signi cantly added value to the growth of HAR. The telecommunication ability of Bluetooth, cellular networking and wireless networking in handheld also encour-ages and supports the collaboration interface for HAR across multiple devices. [6] Camera in handheld devices enable vision-based tracking such as feature points and ground plane tracking without the need of external hardware.The embedded sensors such as accelerometer, gyroscope, compass and Global Positioning System (GPS) helps the handheld devices to have a more comprehensive tracking in HAR system. These sensors also added value to 3D interaction for HAR system that

could be accomplished with single hand as discussed in [11] As discussed in [12] the hardware factors of handheld devices should be taken into consideration when developing HAR application to ensure a decent user experience. The signi cance of these factors is ordered by (1) ergonomics, (2) weight, (3) camera quality, (4) screen size, (5) screen quality and (6) performance

4.2 Collaborative Interface in Handheld Augmented Reality

Collaborative interface in HAR can be achieved by the establishment of network connection between the handheld devices and server to obtain the interest shared content . [1] The connected devices must be able to process the received data to display and manipulate the shared AR content seamlessly. The registration of AR must be accurate among the users because any misalignment will directly a ect the relative spatial positioning of the shared AR object. [8]The sharing of 3D virtual AR object must also be precisely displayed [10]. Multiuser interaction

[8] might induce display issue during synchronization due to the latency across the collaborative interface.There are two types of collaboration: (1) collocated and (2) remote collaboration. Collocated collaboration requires users to be at the same place and is normally conducted in face-to face fashion whereas remote collaboration bridges the connection between distanced users across telecommunication via display devices.A remote collaboration interface aims to increase the telepresence of multiple parties and collocated collaboration interface enable a conductive workspace where users can perceive the shared virtual data 3D space. Handheld collocated collaboration AR system is depict in [7] and [13] Mobile remote collaboration AR system can be seen in [2] and [4].

4.3 Hand gesture recognition with HAR

As mobile devices have been developed, hand gesture recognition plays a significant role in Human-Computer interface (HCI) to operate the devices in a more natural and comfortable way. Such hand gesture recognition is largely done using

one of two kinds of sensors: contact sensors or non-contact sensors. The non-contact methods do not require attaching sensors to human’s body and mostly use visual technologies such as Kinect or mobile camera sensors.Since we are us-ing smartphone as HAR device so de nitely mobile camera will be used.

5 Project challenges

As stated the workload of the overall project can be divided into 3 major parts.First is Ar development,second Hand gesture recognition and implementation of collab-orative Ar.Now we are working on the rst and second part.While Developing the Ar arti cial hand, plane detection was a major issue.Another issue was ob-ject spawning ,specially multiple objects were spawning instead of one,then hand placement was another challenge.For hand recognition part recognition of several hand gestures is also a challenge.And implementation of multiplayer version of the game is another challenge with the help of collaborative Ar technologies.Another challenge is reducing the latency of Ar object since because plane detection most of the time takes a bit time in the devices we tested

6 Proposed solutions

For the development of augmented reality work we will be using Ar foundation which is Unity’s high-level, cross platform API to support Augmented Reality. AR Foundation lets you write your app once, and build for either Android or iOS.AR Foundation includes core features from ARKit, ARCore, Magic Leap, and HoloLens, as well as unique Unity features to build robust apps.So plane de-tection other challenges gets easier.One of the major question might come that why choosing Ar foundation over Vuforia SDK although most of the reference pa-per we attached here uses Vuforia SDK.The answer is lack of support.Currently App developed with Vuforia does not support a large number of mobile devices that are available in the market.For example Vuforia does not work with xiomi and realme phones.Another reason is space(memory).Apps developed with Ar founda-

tion consumes much lesser space in mobile devices than Vuforia.AR Foundation is used to create mobile game apps, so the worst that can happen if a glitch oc-curs is that the user starts the game over.Vuforia, on the other hand, is used for industrial applications such as IIoT, Remote Service Calls, and programming industrial machines and robots.So it is much more resource heavy due to ex-tra amount of protocols.For the hand gesture recognition part we will be using OpenCv.For multiplayer interface two technologies will be studied and integrated with Unity3d.They are (1) Photon Unity Networking (PUN) and (2) Unity Net-working (UNET).They are readily supported by Unity3D game engine and can be implemented to connect handheld devices across the internet.PUN has a dedicated server that connects handheld devices but UNET provides a client hosted architec-ture that might increase the device's workload and reduce the performance. This also induces a higher latency in transmitting networked data for UNET comparing to PUN which might cause registration and interaction error in collaborative HAR system.

7 Experiments and results

7.1 Augmented Reality based development

So as stated above our rst goal is to develop the augmented reality part of the app.We are successful at developing that version and also showed the app con-taining the arti cial hand using augmented reality during the pre-project defence after seventh semester.As stated we used Unity3d for this purpose.What we did was we developed the arti cial hand with augmented reality and it can show signs from a random number generator in the background.At this stage we allowed users to give inputs using a graphical user interface in the screen.The users given input was compared against Augmented arti cial hands shown sign and thus the game is played.This was the rst demo of the game.

7.2 Computer vision based development

As planned the next step was to develop the computer vision part.We successfully developed the part where we were successfully able to detect hand gestures gesture signs showed by 1,2,3,4,5 and 6.Although at rst accuracy was low but after some changes in codebase we were also able to improve the accuracy.We were going to recognize hand gestures from a video sequence. To recognize these gestures from a live video sequence, we rst need to take out the hand region alone removing all the unwanted portions in the video sequence. After segmenting the hand region, we then count the ngers shown in the video sequence to instruct a robot based on the nger count. Thus, the entire problem is solved using 2 simple steps { I. Find and segment the hand region from the video sequence. II. Count the number of ngers from the segmented hand region in the video sequence.

7.3 Integration of Augmented reality and computer vision

So next game plan was to integrate between the two.All the time we developed the game using user centered approach.As the augmented reality part was developed with unity 3d so we at rst tried the naive way of development.That means for the support library for integrating the openCv codebase we depended on the library available at unity asset store.But the problem here is that at the time we were integrating that library was not free,although while we started at that time that was free.So we tried with some other third party libarys but unfortunately all of the apps developed were crashing.At that time we became very disappointed what to do.Actually later after a lot of research on internet what we found that those who do this type of work(integrating python or other language code with unity) they customize there support library according to their need. This type of cus-tomization usually needs system engineers who has very much deep understanding of core unity structures.But as you guessed this is beyond our scope.

7.4 Final stages of development

At this point we basically has one option to choose one between Augmented re-ality or computer vision.So we have to take a decision that we will be using the computer vision based app as by this a gamer will be able to play with his own hands.Otherwise if we have chosen the Augmented reality part then that will be very much similar to some other implementations of hand cricket which are al-ready available in the internet.So at this stage of development for the development of the entire system we are using Kivy which is an open source software library for the rapid development of applications with python.Thus the nal structure of the working game was developed.We also included sounds in our game.

8 Conclusion

Albert Einstein once said that 'Logic will get you from A to B but imagination will take you everywhere'.So we tried to give our best to decrease the gap between what we rst imagined to do.And we believe that we were successful because at least we are near to what we promised at the very early stages of development i.e 'The user will play with their own hands(hand gestures)'.But we think there are lots of possibility's with this game.So we decided that this game will be available as an open-source software in the Github.So the two separate versions will be there separately as Augmented reality based development and computer vision based development so that if anyone wants to do our un nished work so that they can easily start working.So major improvements that can be done here is the integration part and also making remote play possible(like two players from distance can play)

References

[1] Mark Billinghurst and Bruce H Thomas. Mobile collaborative augmented reality. In Recent trends of mobile collaborative augmented reality systems,

pages 1{19. Springer, 2011.

[2] Henry Chen, Austin S Lee, Mark Swift, and John C Tang. 3d collaboration method over hololenstm and skypetmend points. In Proceedings of the 3rd International Workshop on Immersive Media Experiences (ImmersiveME), pages 27{30.

[3] Anna Gardeli and Spyros Vosinakis. Arquest: A tangible augmented reality approach to developing computational thinking skills. In 2019 11th Inter-national Conference on Virtual Worlds and Games for Serious Applications (VS-Games), pages 1{8. IEEE, 2019.

[4] Ste en Gauglitz, Benjamin Nuernberger, Matthew Turk, and Tobias Hollerer. In touch with the remote world: Remote collaboration with augmented reality drawings and virtual navigation. In Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology, pages 197{205, 2014.

[5] Anders Henrysson. Bringing augmented reality to mobile phones. PhD thesis, ACM, 2007.

[6] Anders Henrysson, Mark Billinghurst, and Mark Ollila. Face to face collaborative ar on mobile phones. In Fourth ieee and acm international symposium on mixed and augmented reality (ismar'05), pages 80{89. IEEE, 2005.

[7] Anders Henrysson, Mark Billinghurst, and Mark Ollila. Ar tennis. In ACM SIGGRAPH 2006 Emerging technologies, pages 1{es. 2006.

[8] Ajune Wanis Ismail and Mohd Shahrizal Sunar. Collaborative augmented re-ality approach for multi-user interaction in urban simulation. In 2009 Inter-national Conference on Information and Multimedia Technology, pages 19{23. IEEE, 2009.

[9] Sung Lae Kim, Hae Jung Suk, Jeong Hwa Kang, Jun Mo Jung, Teemu H Laine, and Joonas Westlin. Using unity 3d to facilitate mobile augmented reality game development. In 2014 IEEE World Forum on Internet of Things (WF-IoT), pages 21{26. IEEE, 2014.

[10] Stephan Lukosch, Mark Billinghurst, Leila Alem, and Kiyoshi Kiyokawa. Collaboration in augmented reality. Computer Supported Cooperative Work (CSCW), 24(6):515{525, 2015.

[11] Annette Mossel, Benjamin Venditti, and Hannes Kaufmann. 3dtouch and homer-s: intuitive manipulation techniques for one-handed handheld augmented reality. In Proceedings of the Virtual Reality International Conference: Laval Virtual, pages 1{10, 2013.

[12] Lawrence Sambrooks and Brett Wilkinson. Handheld augmented reality: Does size matter? In Proceedings of the 16th Australasian User Interface Conference (AUIC 2015), volume 27, page 30, 2015.

[13] Daniel Wagner, Thomas Pintaric, and Dieter Schmalstieg. The invisible train: a collaborative handheld augmented reality demonstrator. In ACM SIG-GRAPH 2004 emerging technologies, page 12. 2004.

HOW TO USE/ OR Reproduce the project:

1. GET THE CODE FROM BELOW or contact syedmujtahid@iut-dhaka.edu

2. Get your code editor and copy the code

3. Now download and install Unity and other dependencies setup

4. Now just Build the project in your favorite environment android, windows etc using unity developer tools

5. And enjoy

Source Code

Augmented reality Based implementation:

Hand gesture recognition:

```
#use python for this

from cv2 import cv2
import imutils
import math

# global variables
bg = None
```

```
#--------------------------------------------------
# To find the running average over the background
#--------------------------------------------------
def run_avg(image, accumWeight):
    global bg
    # initialize the background
    if bg is None:
        bg = image.copy().astype("float")
        return

    # compute weighted average, accumulate it and update the background
    cv2.accumulateWeighted(image, bg, accumWeight)

#-------------------------------------------
# To segment the region of hand in the captured image
#-------------------------------------------
def segment(image, threshold=50):
    global bg
    # find the absolute difference between background and current frame
    diff = cv2.absdiff(bg.astype("uint8"), image)

    # threshold the diff image so that we get the foreground
    ret, thresholded = cv2.threshold(diff, threshold, 255, cv2.THRESH_BINARY)

    # get the contours in the thresholded image
    cnts, _ = cv2.findContours(thresholded.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    # return None, if no contours detected
    if len(cnts) == 0:
        return
    else:
        # based on contour area, get the maximum contour which is the hand
        segmented = max(cnts, key=cv2.contourArea)
        return (thresholded, segmented)

#-----------------------------------------------------------
# To count the number of fingers in the segmented hand region
#-----------------------------------------------------------
def count(thresholded, segmented):
    # find the convex hull of the segmented hand region
    hull = cv2.convexHull(segmented,  returnPoints=False)
    defects = cv2.convexityDefects(segmented, hull)

    # Bascially indicates how much finger is visible in screen
    countDefects = 0

    for i in range(defects.shape[0]):
        # Returns start point, end point, farthest point, approximate distance to farthest point
```

```
            s, e, f, d = defects[i, 0]
            start = tuple(segmented[s][0])
            end = tuple(segmented[e][0])
            far = tuple(segmented[f][0])

            a = math.sqrt((end[0] - start[0]) ** 2 + (end[1] - start[1]) ** 2)
            b = math.sqrt((far[0] - start[0]) ** 2 + (far[1] - start[1]) ** 2)
            c = math.sqrt((end[0] - far[0]) ** 2 + (end[1] - far[1]) ** 2)

            # This angle is used while hand is moving around
            angle = (math.acos((b ** 2 + c ** 2 - a ** 2) / (2 * b * c)) * 180) / 3.14

            # If angle < 90 degree then treat as a finger
            if angle <= 90:
                countDefects += 1

    return (countDefects + 1)

#-----------------
# MAIN FUNCTION
#-----------------
if __name__ == "__main__":
    # initialize accumulated weight
    accumWeight = 0.5

    # get the reference to the webcam
    camera = cv2.VideoCapture(0)

    # region of interest (ROI) coordinates
    top, right, bottom, left = 10, 150, 225, 490

    # initialize num of frames
    num_frames = 0

    # calibration indicator
    calibrated = False

    # keep looping, until interrupted
    while(True):
        # get the current frame
        # grabbed is bool, if the frame is read correctly then grabbed will be true
        (grabbed, frame) = camera.read()

        # resize the frame
        frame = imutils.resize(frame, width=700)

        # flip the frame so that it is not the mirror view
        frame = cv2.flip(frame, 1)
```

```
# clone the frame
clone = frame.copy()

# get the height and width of the frame
(height, width) = frame.shape[:2]

# get the ROI
roi = frame[top:bottom, right:left]

# convert the roi to grayscale and blur it
gray = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
gray = cv2.GaussianBlur(gray, (7, 7), 0)

# to get the background, keep looking till a threshold is reached
# so that our weighted average model gets calibrated
if num_frames < 30:
    run_avg(gray, accumWeight)
    if num_frames == 1:
        print("[STATUS] please wait! calibrating...")
    elif num_frames == 29:
        print("[STATUS] calibration successfull...")
else:
    # segment the hand region
    hand = segment(gray)

    # check whether hand region is segmented
    if hand is not None:
        # if yes, unpack the thresholded image and
        # segmented region
        (thresholded, segmented) = hand

        # draw the segmented region and display the frameq
        cv2.drawContours(clone, [segmented + (right, top)], -1, (0, 0, 255))

        # count the number of fingers
        fingers = count(thresholded, segmented)

        cv2.putText(clone, str(fingers), (70, 45), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)

        # show the thresholded image
        cv2.imshow("Thesholded", thresholded)

# draw the segmented hand
cv2.rectangle(clone, (left, top), (right, bottom), (0,255,0), 2)

# increment the number of frames
num_frames += 1

# display the frame with segmented hand
```

```
        cv2.imshow("Video Feed", clone)

        # observe the keypress by the user
        keypress = cv2.waitKey(1) & 0xFF

        # if the user pressed "q", then stop looping
        if keypress == ord("q"):
            break

# free up memory
camera.release()
cv2.destroyAllWindows()
```

NavigateController:

From this module rest of the code is C#,use the unity specific editor for this

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class NavigateController : MonoBehaviour
{
    public void GoToNextScene()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex +
1);
    }

    public void QuitGame()
    {
        Application.Quit();
    }

    public void BackButtonPress()
    {
```

```
        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex -
1);
    }
}
```

PlaneIndicator:

using System.Collections.Generic;

```
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;

public class PlaneIndicator : MonoBehaviour
{
    ARRaycastManager rayCastManager;
    GameObject placeDetector;
    bool poseIsValid;

    public GameObject buttonBearObj;

    void Start()
    {
        rayCastManager = FindObjectOfType<ARRaycastManager>();
        placeDetector = transform.GetChild(0).gameObject;

        placeDetector.SetActive(false);
        buttonBearObj.SetActive(false);
    }

    void Update()
    {
```

```
        placeDetection();
        spawnIndicator();
    }

    private void placeDetection()
    {
        List<ARRaycastHit> rayHits = new List<ARRaycastHit>();
        rayCastManager.Raycast(new Vector2(Screen.width / 2, Screen.height / 2), rayHits,
TrackableType.Planes);

        poseIsValid = rayHits.Count > 0;

        if (poseIsValid)
        {
            transform.position = rayHits[0].pose.position;
            transform.rotation = rayHits[0].pose.rotation;
        }
    }

    private void spawnIndicator()
    {
        if (poseIsValid && !placeDetector.activeInHierarchy)
        {
            placeDetector.SetActive(true);
            buttonBearObj.SetActive(true);

            var cameraForward = Camera.current.transform.forward;
            var cameraBearing = new Vector3(cameraForward.x, 0f, cameraForward.z).normalized;
            transform.rotation = Quaternion.LookRotation(cameraBearing);
        }
```

```
    }

}
```

ToggleController:

```
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;

public class PlaneIndicator : MonoBehaviour
{
    ARRaycastManager rayCastManager;
    GameObject placeDetector;
    bool poseIsValid;

    public GameObject buttonBearObj;

    void Start()
    {
        rayCastManager = FindObjectOfType<ARRaycastManager>();
        placeDetector = transform.GetChild(0).gameObject;

        placeDetector.SetActive(false);
        buttonBearObj.SetActive(false);
    }

    void Update()
    {
        placeDetection();
        spawnIndicator();
    }

    private void placeDetection()
    {
        List<ARRaycastHit> rayHits = new List<ARRaycastHit>();
        rayCastManager.Raycast(new Vector2(Screen.width / 2, Screen.height / 2), rayHits,
TrackableType.Planes);

        poseIsValid = rayHits.Count > 0;

        if (poseIsValid)
        {
            transform.position = rayHits[0].pose.position;
            transform.rotation = rayHits[0].pose.rotation;
        }
    }
```

```
    private void spawnIndicator()
    {
        if (poseIsValid && !placeDetector.activeInHierarchy)
        {
            placeDetector.SetActive(true);
            buttonBearObj.SetActive(true);

            var cameraForward = Camera.current.transform.forward;
            var cameraBearing = new Vector3(cameraForward.x, 0f, cameraForward.z).normalized;
            transform.rotation = Quaternion.LookRotation(cameraBearing);
        }
    }
}
```

UIController:

```
using UnityEngine;
using UnityEngine.UI;
using Random = UnityEngine.Random;

public class UIController : MonoBehaviour
{
    public Text welcomeText, playerScoreText, deviceScoreText;
    public Image opponentImage, playerImage;
    public GameObject uiObject;
    public Material[] mats;
    public GameObject[] handObjects;

    string[] optionToggles = { "Batting", "Bowling" };
    float timeStart, timeLen = 3f, fingerTimeStart;
    int idxToss = ToggleController.indexOfToss, idxOption = ToggleController.indexOfOption, randInt,
count = 0, battingInt = 0, numWickets = 0;
    int[] runArray = {1, 2, 3, 4, 5};
    float over = 0f;

    // Start is called before the first frame update
    void Start()
    {
        // Disable Hand Objects at first
        for(int i = 0; i < handObjects.Length; i++)
        {
            handObjects[i].SetActive(false);
        }

        int idx = Random.Range(0, optionToggles.Length);
```

```
        timeStart = Time.time;

        if (idx == idxToss)
        {
            welcomeText.text = "You Win the toss! Now you can play " + optionToggles[idxOption];
            playerImage.gameObject.GetComponent<Image>().material = mats[idxOption];
            opponentImage.gameObject.GetComponent<Image>().material = mats[Mathf.Abs(1 -
idxOption)];
        }
        else
        {
            welcomeText.text = "You Loss the toss! Now you have to play " + optionToggles[Mathf.Abs(1 -
idxOption)];
            opponentImage.gameObject.GetComponent<Image>().material = mats[idxOption];
            playerImage.gameObject.GetComponent<Image>().material = mats[Mathf.Abs(1 - idxOption)];
        }
    }

    // Update is called once per frame
    void Update()
    {
        if (Time.time > timeStart + timeLen)
        {
            uiObject.SetActive(false);
        }

        if(Time.time > fingerTimeStart + timeLen)
        {
            for(int i = 0; i < handObjects.Length; i++)
            {
                if (handObjects[i].activeInHierarchy)
                {
                    handObjects[i].SetActive(false);
                }
            }
        }
    }

    public void ScoreButtonPress(int idx)
    {
        randInt = Random.Range(0, 5);
        handObjects[randInt].SetActive(true);
        fingerTimeStart = Time.time;
        count += 1;

        if (count % 6 == 0)
        {
            over = 1f + (over - .1f);
            over = Mathf.Round(over);
```

```
        }
        else
        {
            over += 0.1f;
        }

        if(idxOption == 0)
        {
            // if player is batsman
            DecisionTextMethod(playerScoreText, deviceScoreText, idx, randInt);
        }
        else
        {
            // if player is bowler
            DecisionTextMethod(deviceScoreText, playerScoreText , idx, randInt);
        }
    }

    public void IntervalButton()
    {
        if (handObjects[randInt].activeInHierarchy)
        {
            handObjects[randInt].SetActive(false);
        }
    }

    void DecisionTextMethod(Text batText, Text bowlText, int idx, int rand)
    {
        if (idx != rand)
        {
            battingInt += runArray[idx];
            batText.text = battingInt.ToString();
            bowlText.text = over.ToString()+ "-" + numWickets.ToString();
        }
        else
        {
            numWickets += 1;
            batText.text = "Wicket";
            bowlText.text = over.ToString() + "-" + numWickets.ToString();
        }
    }
}
```

Computer vision based implementation:

```
from cv2 import cv2
import imutils
```

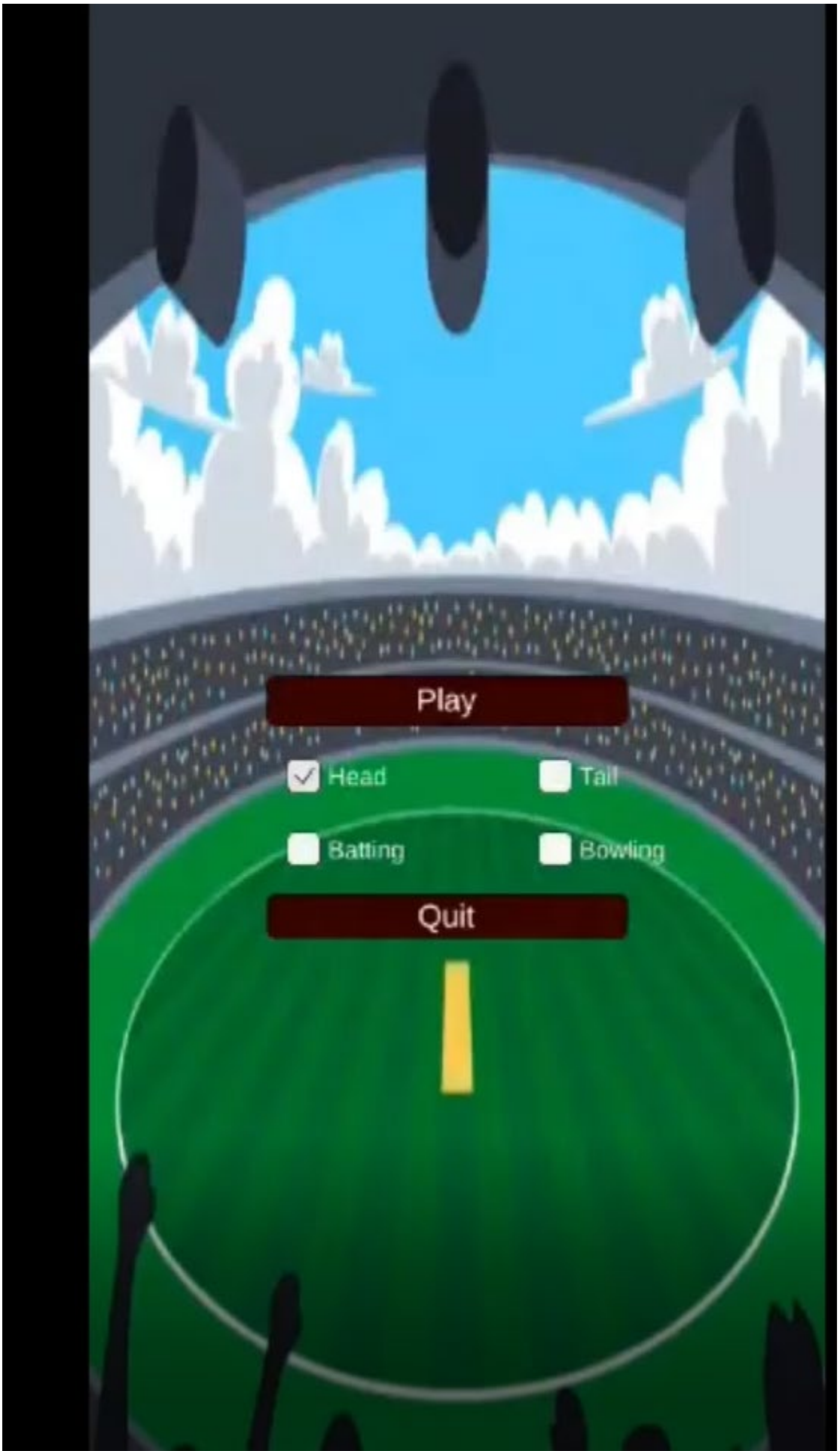
```
import math

# global variables
bg = None

#--------------------------------------------------
# To find the running average over the background
#--------------------------------------------------
def run_avg(image, accumWeight):
    global bg
    # initialize the background
    if bg is None:
        bg = image.copy().astype("float")
        return

    # compute weighted average, accumulate it and update the background
    cv2.accumulateWeighted(image, bg, accumWeight)

#-------------------------------------------
# To segment the region of hand in the captured image
#-------------------------------------------
def segment(image, threshold=50):
    global bg
    # find the absolute difference between background and current frame
    diff = cv2.absdiff(bg.astype("uint8"), image)

    # threshold the diff image so that we get the foreground
    ret, thresholded = cv2.threshold(diff, threshold, 255, cv2.THRESH_BINARY)

    # get the contours in the thresholded image
    cnts, _ = cv2.findContours(thresholded.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    # return None, if no contours detected
    if len(cnts) == 0:
        return
    else:
        # based on contour area, get the maximum contour which is the hand
        segmented = max(cnts, key=cv2.contourArea)
        return (thresholded, segmented)

#-----------------------------------------------------------
# To count the number of fingers in the segmented hand region
#-----------------------------------------------------------
def count(thresholded, segmented):
    # find the convex hull of the segmented hand region
    hull = cv2.convexHull(segmented,  returnPoints=False)
    defects = cv2.convexityDefects(segmented, hull)
```

```
    # Bascially indicates how much finger is visible in screen
    countDefects = 0

    for i in range(defects.shape[0]):
        # Returns start point, end point, farthest point, approximate distance to farthest point
        s, e, f, d = defects[i, 0]
        start = tuple(segmented[s][0])
        end = tuple(segmented[e][0])
        far = tuple(segmented[f][0])

        a = math.sqrt((end[0] - start[0]) ** 2 + (end[1] - start[1]) ** 2)
        b = math.sqrt((far[0] - start[0]) ** 2 + (far[1] - start[1]) ** 2)
        c = math.sqrt((end[0] - far[0]) ** 2 + (end[1] - far[1]) ** 2)

        # This angle is used while hand is moving around
        angle = (math.acos((b ** 2 + c ** 2 - a ** 2) / (2 * b * c)) * 180) / 3.14

        # If angle < 90 degree then treat as a finger
        if angle <= 90:
            countDefects += 1

    return (countDefects + 1)

#-----------------
# MAIN FUNCTION
#-----------------
if __name__ == "__main__":
    # initialize accumulated weight
    accumWeight = 0.5

    # get the reference to the webcam
    camera = cv2.VideoCapture(0)

    # region of interest (ROI) coordinates
    top, right, bottom, left = 10, 150, 225, 490

    # initialize num of frames
    num_frames = 0

    # calibration indicator
    calibrated = False

    # keep looping, until interrupted
    while(True):
        # get the current frame
        # grabbed is bool, if the frame is read correctly then grabbed will be true
        (grabbed, frame) = camera.read()

        # resize the frame
```

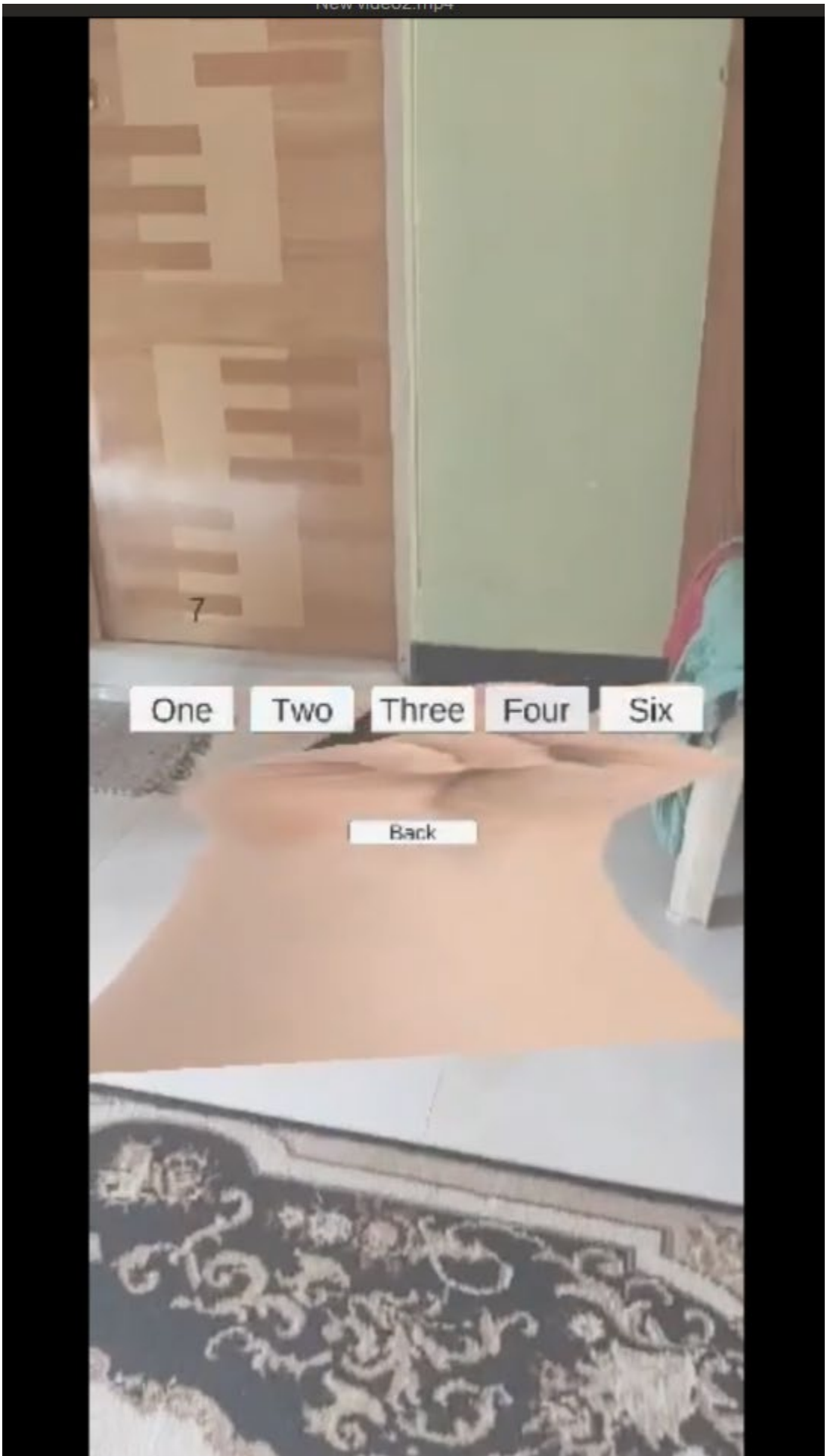
```
frame = imutils.resize(frame, width=700)

# flip the frame so that it is not the mirror view
frame = cv2.flip(frame, 1)

# clone the frame
clone = frame.copy()

# get the height and width of the frame
(height, width) = frame.shape[:2]

# get the ROI
roi = frame[top:bottom, right:left]

# convert the roi to grayscale and blur it
gray = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
gray = cv2.GaussianBlur(gray, (7, 7), 0)

# to get the background, keep looking till a threshold is reached
# so that our weighted average model gets calibrated
if num_frames < 30:
    run_avg(gray, accumWeight)
    if num_frames == 1:
        print("[STATUS] please wait! calibrating...")
    elif num_frames == 29:
        print("[STATUS] calibration successfull...")
else:
    # segment the hand region
    hand = segment(gray)

    # check whether hand region is segmented
    if hand is not None:
        # if yes, unpack the thresholded image and
        # segmented region
        (thresholded, segmented) = hand

        # draw the segmented region and display the frameq
        cv2.drawContours(clone, [segmented + (right, top)], -1, (0, 0, 255))

        # count the number of fingers
        fingers = count(thresholded, segmented)

        cv2.putText(clone, str(fingers), (70, 45), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)

        # show the thresholded image
        cv2.imshow("Thesholded", thresholded)

# draw the segmented hand
cv2.rectangle(clone, (left, top), (right, bottom), (0,255,0), 2)
```

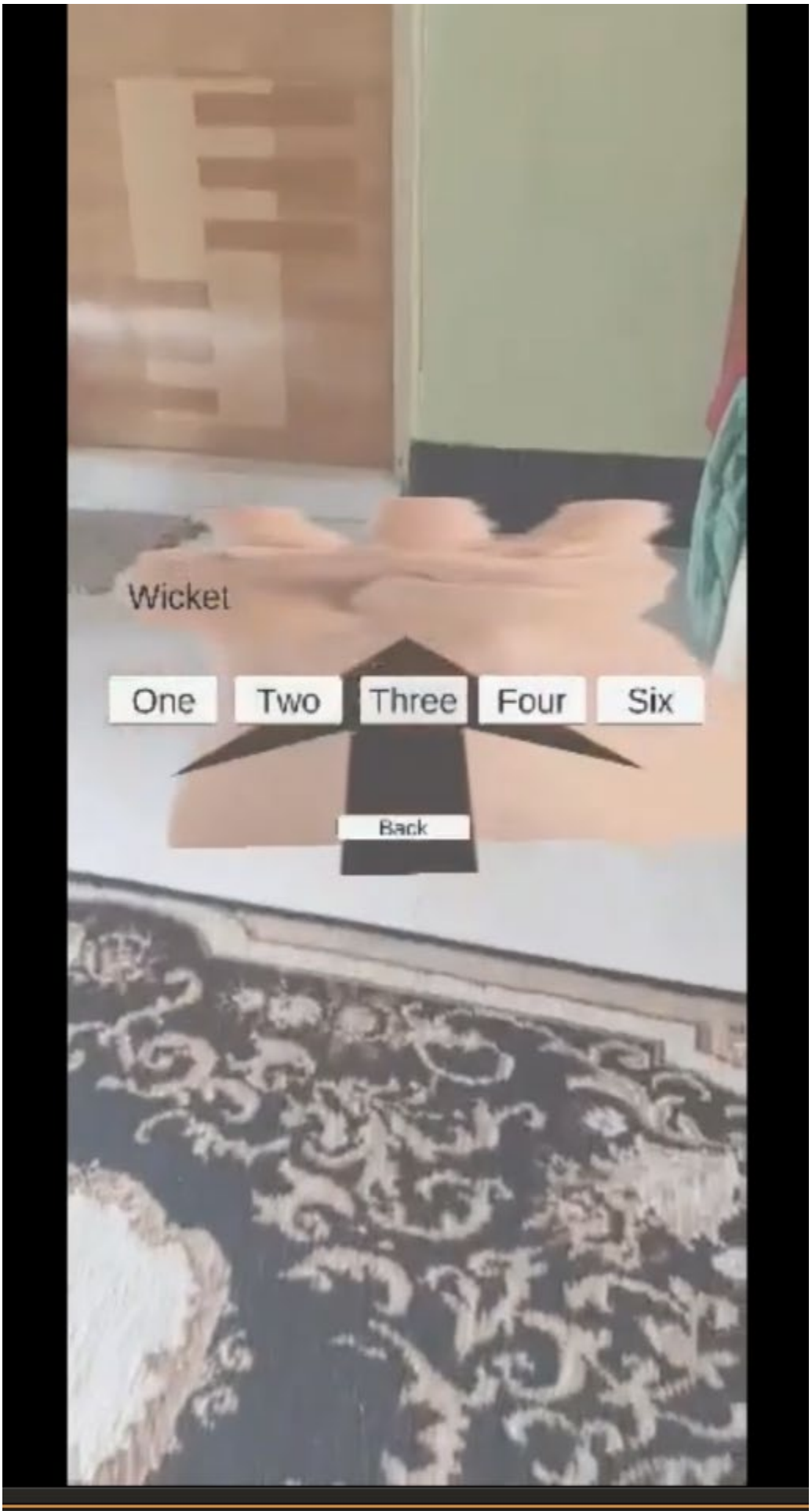
```
        # increment the number of frames
        num_frames += 1

        # display the frame with segmented hand
        cv2.imshow("Video Feed", clone)

        # observe the keypress by the user
        keypress = cv2.waitKey(1) & 0xFF

        # if the user pressed "q", then stop looping
        if keypress == ord("q"):
            break

# free up memory
camera.release()
cv2.destroyAllWindows()
```

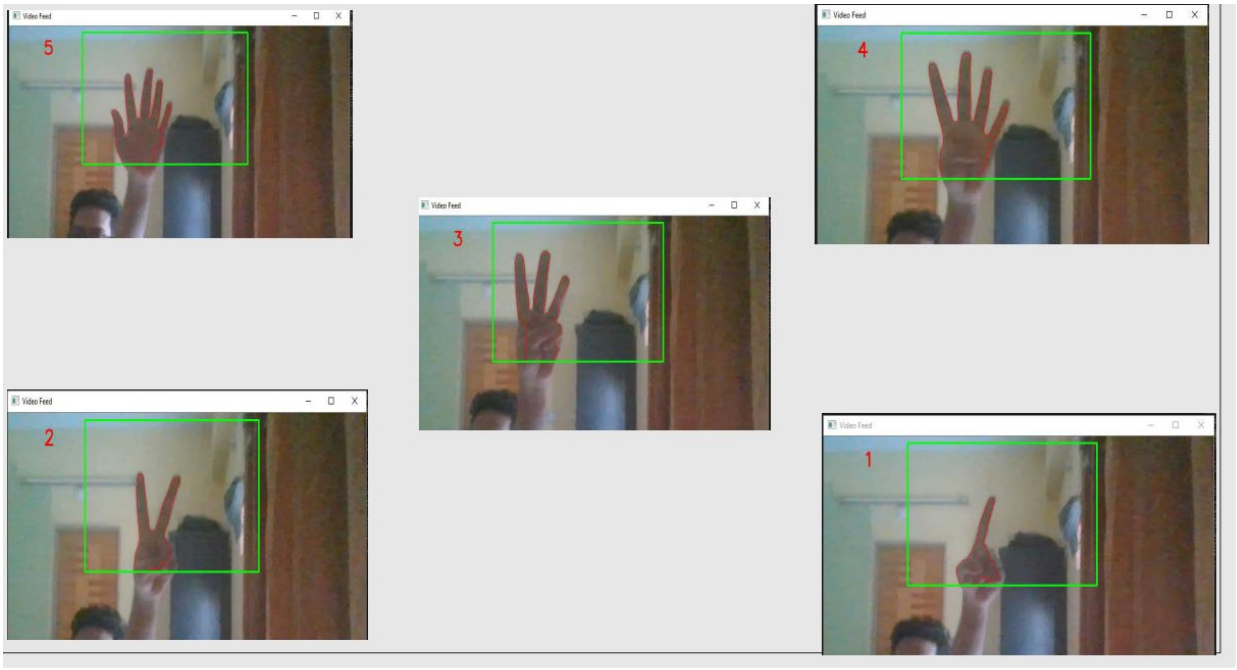
Project implementation screenshots


Play
Head
Tail
Batting
Bowling
Quit


3
One
Two
Three
Four
Six
Back

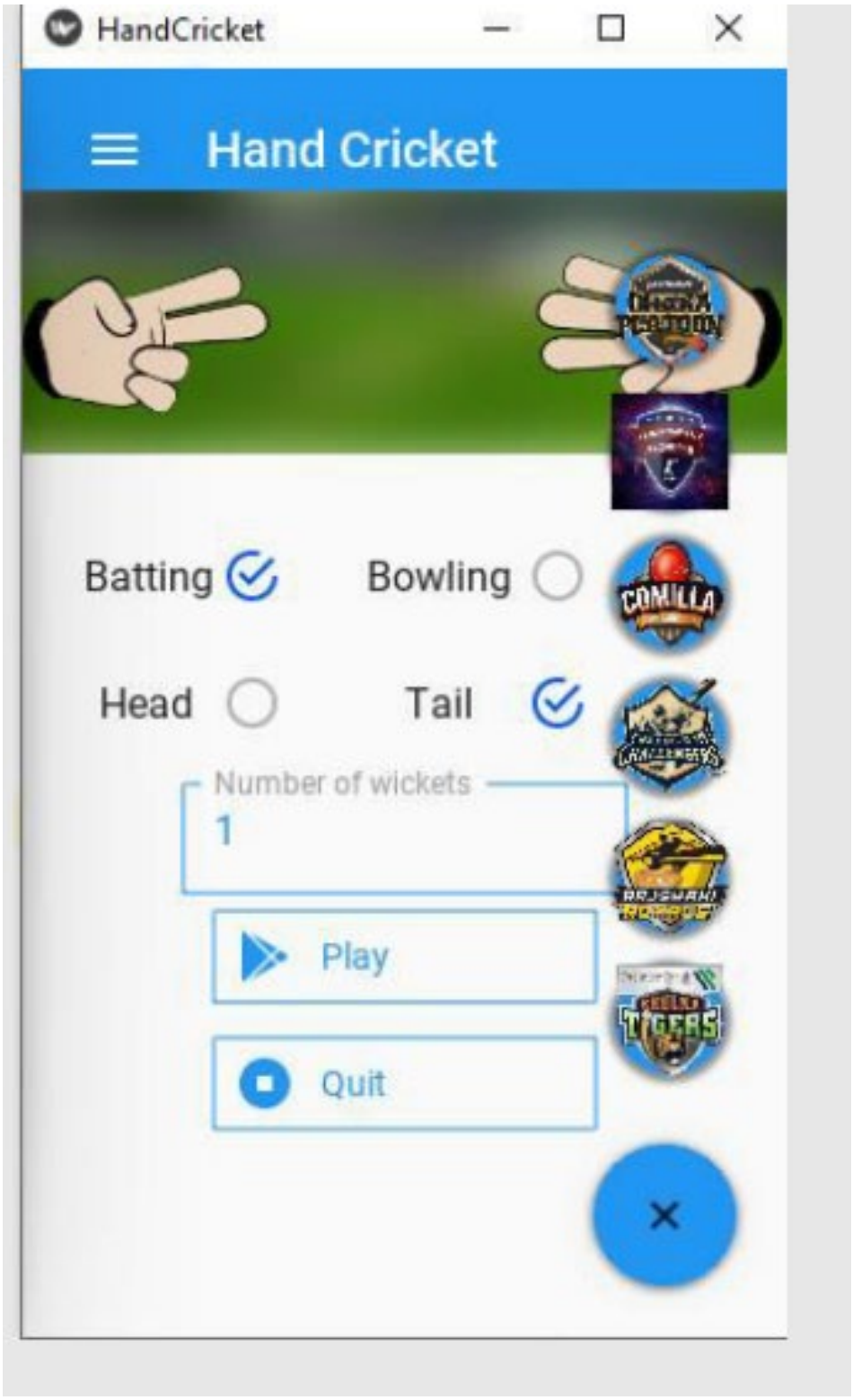

7
One
Two
Three
Four
Six
Back

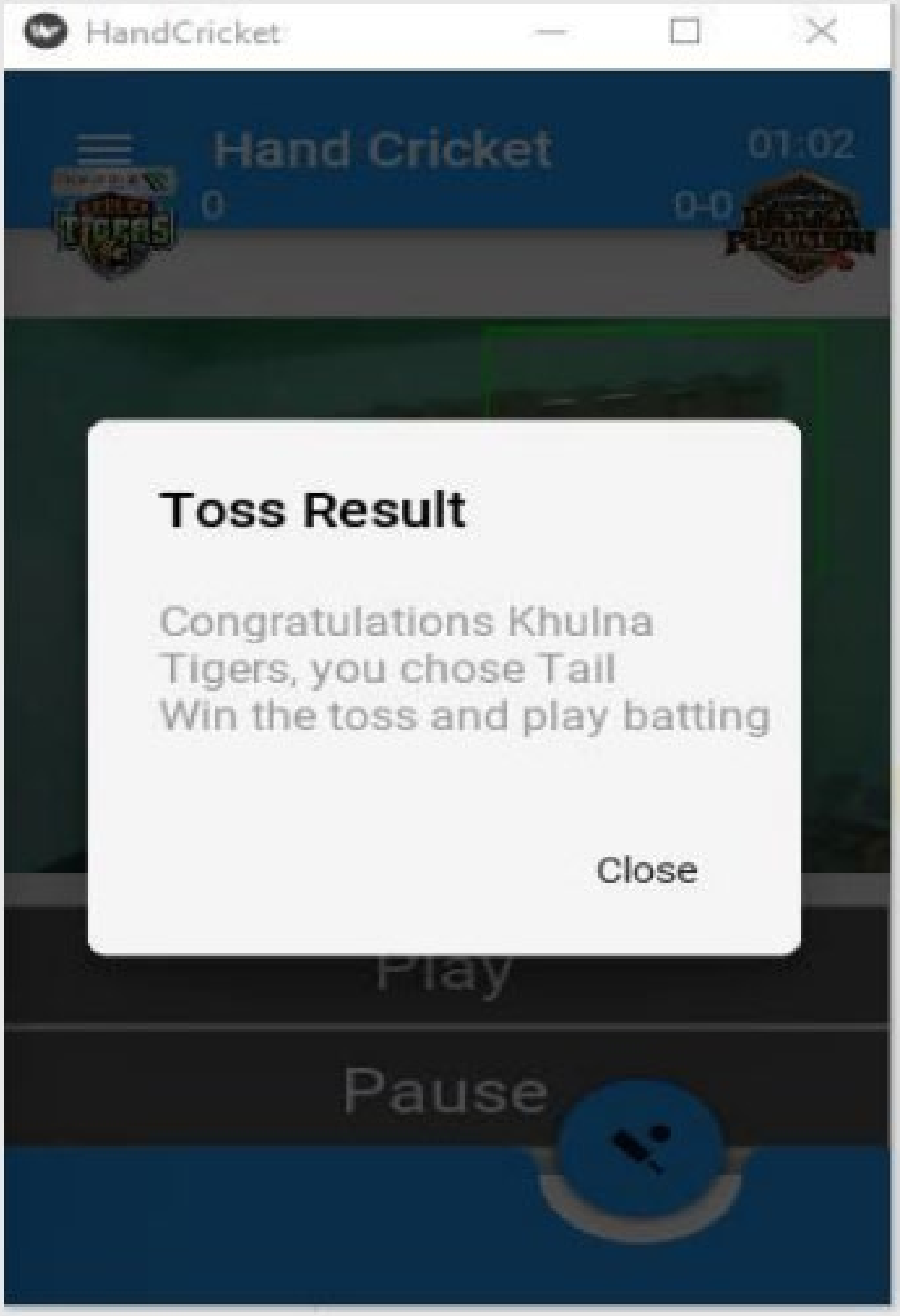

Wicket
One
Two
Three
Four
Six
Back

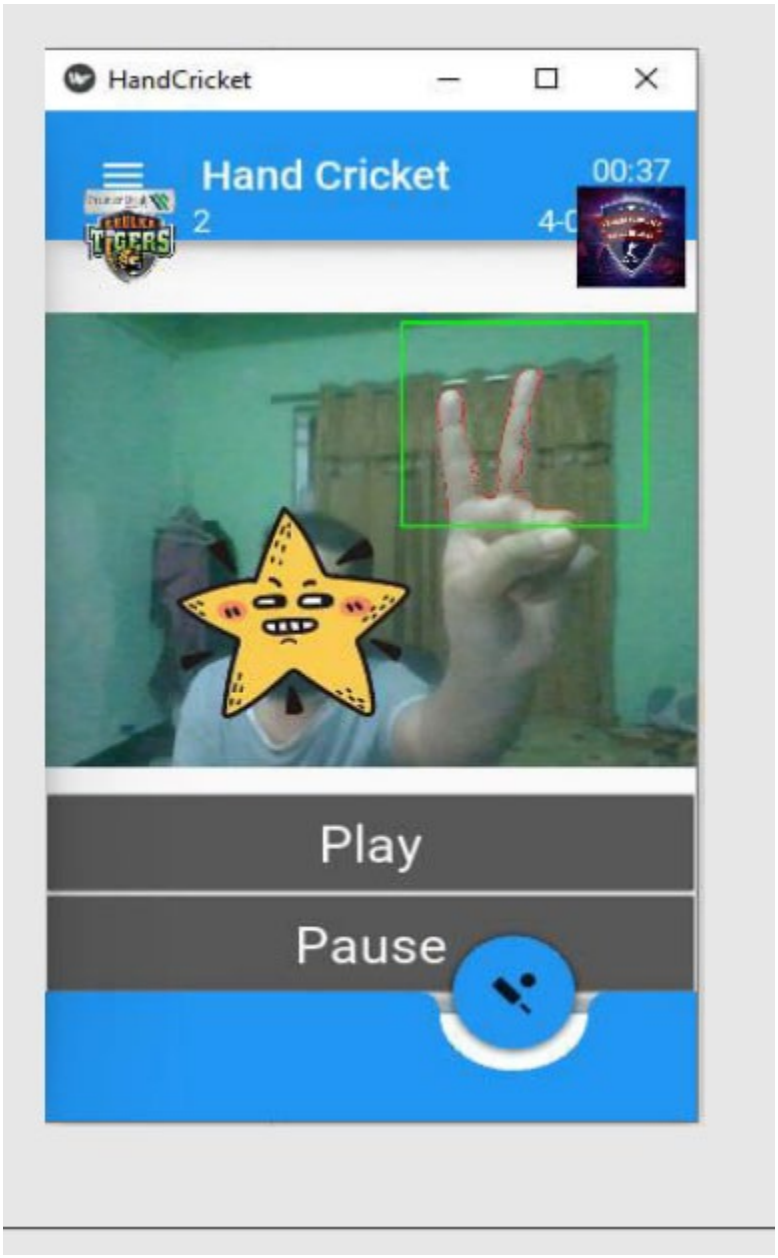

Video Feed
5
Video Feed
4
Video Feed
3
Video Feed
2
Video Feed
1

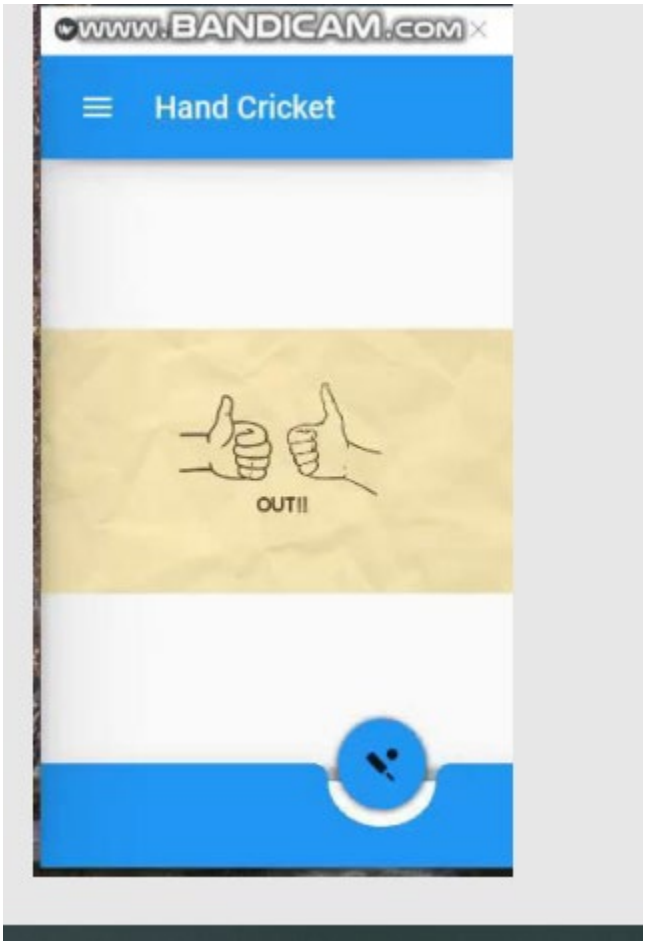

HandCricket
Hand Cricket
OUT!!


HandCricket
HAND
CRICKET
Hand Cricket
Home
Game
Exit


HandCricket
Hand Cricket
Batting
Bowling
Head
Tail
Number of wickets
1
Play
Quit


HandCricket
Hand Cricket
01:02
0
0-0
Toss Result
Congratulations Khulna Tigers, you chose Tail
Win the toss and play batting
Close
Play
Pause


HandCricket
Hand Cricket
00:37
2
Play
Pause


www.BANDICAM.com
Hand Cricket
OUT!!