IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Implementing KLM Algorithm and OpenCV forReal Time Hand movement Tracking

Ashmit Gupta*
School of Computer Science and EngineeringVellore Institute of Technology
Vellore, India.
Student Member IEEE

Swarnalatha P*
School of Computer Science and EngineeringVellore Institute of Technology
Vellore, India.
Fellow Member IEEE

Abstract— **Most of the population, especially, elderly people are interested in computer gaming technology and desire to be actively involved in such activities, however they are limited by their responsiveness due to age, as gaming requires fast reflexes in order to provide and comfortable / satisfactory experience. People who are not very knowledgeable about gaming are also limited to enter the gaming world. This up brings a huge demand in the current market for a software / product which can enable users who are new to the gaming world to be able to comfortably enter / transition into the gaming world or be able to enjoy the merits of gaming without their limitations stopping them.**

Keywords—HCI, Image-Detection, Machine Learning Python

I. INTRODUCTION

Our project allows for interactive gaming e xperience which uses tracker to be able to interacting gaming environments using human movements. Ga me-N-Geek is a program that allows users to play any PC related video game using their hand signs and movements instead of using mouse and keyboard. It allows users to play games using hand signs. Ga me-N-Geek allows for a smooth gaming e xperience without the inclusion of hands and can be used to interact within the gaming environment entirely in a visual manner.

To provide gamers and people of interest in the gaming field with a unique and mes merizing e xperience which can bring a shift toward human movement computer gaming.

To provide a software / platform for real time visual interaction with the system to ensure the complete gaming experience.

To introduce people with limited knowledge of gaming to the gaming world and to ensure a comfortable e xpe rience as well as a s mooth transition into the gaming technology.

The project focuses on helping the people with limited gaming knowledge be able to have a good experience with gaming and to be able to interact with it using hand movements instead of pressing or clicking buttons on a keyboard/mouse.

It also increases interactivity and attention span during gaming as it includes the involvement of real time hand movements which help perform tasks within the game which will fu rther user experience.

The objectives of using trackers to enable interactive gaming e xperiences that incorporate human movements can vary depending on the specific context and goals of the game. However, some common objectives of using this technology include:

Increased immersion: By allowing players to physically interact with the game environment using their body movements, tracking technology can create a more immersive and engaging gaming experience.

Improved game play: Using tracking tech nology can pr ovid e mor e pr ecise a nd responsive control mechanis ms, allowing players to perform more complex and nuanced actions in the game.

Health and fitness benefits: Some games that utilize tracking technology are designed to encourage physical activity and e xerc ise, which can have health and fitness benefits for players.

Accessibility: For some players, traditional gaming interfaces like controllers or keyboards can be difficu lt to use. Tracking technology can provide alternative control mechanis ms that are more accessible for players with different abilities or needs.

Overall, the use of trackers in interactive gaming environments can provide a range of benefits, from improving game play and immersion to promoting health and accessibility.

II. LITERATURE SURVEY

1 .Author Rudy Hartanto et al[4] discusses about-

This paper proposes the use of contour detection, fingers movement detection, motion detection and segmentation, andapplying these as a mouse functionality to be able to use the hand as a mouse which can interact with the given system.

2. Author Panayiotis Zaphiris et all[5] discusses about-

This paper discusses about the various issues which are brought up during HCI involvement in computer games, this includes factors such as: Theoretical, Methodological, Technical, and social Culturalaspects. It also brings to light the issue of the familiarity of many users with the QWERTY keyboard style which may hinder their performance/ Expe rience with the game when they have to use HCI related gadgets, gaming tools to perform the same activities.

3: Hand landmarks detection guide: You can identify the hands' landmarks in a picture using the Media Pipe Hand Land marker task. This task can be used to loc alize the hands' importa nt are as and rend er visual effects on top of the hands. This task generates hand landmarks in image coordinates, hand landmarks in world coordinates, and handedness (left/right hand) of numerous detected hands using image data and a machinelearning (ML) model as static data or a continuous stream.

The project focuses on helping the people with limited gaming knowledge be able to have a good experience with gaming and to be able to interact with it using hand movements instead of pressing or clicking buttons on a keyboard/mouse. It also increases interactivity and attention span during gaming as it includes the involvement of real time hand movements which help perform tasks within the game which will further user experience.

III. The Proposed Methodology

Import required libraries including OpenCV, datetime, numpy, time, pyautogui, pydirect input, win 32api and Hand Tracking Module.

Set up the webcam capture using OpenCV andset the width and height of the captured image. Create a Hand Detector object from Hand Tracking Module to detect hand gestures and movements in the captured image. Start an infinite loop to continuously capture images from the webcam. Read the image frame and flip it horizontally to mirror the movement of the user's hand. Convert the image frame to HSV format to detect red colour in the image. Define the lower and upper bounds for the red colour range, and create a mask to filter out any other colour. Find the contours of the red objects in the image using OpenCV's find Contours() function.

The code combines three scripts that use computer vision and hand tracking techniques to perform hand gesture recognition and control mouse and keyboard functions.

OpenCV, Numpy, datetime, Mouse, Time, HandTrackingModule,pyautogui,pydirectinput,win32api, and win32con areonly a few of the required libraries that are imported by the code.

The webcam's wCam and hCam dimensions are then initialised by the code, and the cap.set() method is used to configure the camera to capture the dimensions.

Five fingertip IDs are included in the tipIds list and utilised later in the code.

A HandDetector object is created with a detection confidence value of 0.75 and a maximum of 1 hand to detect.

The cap.read() method is used to begin an endless loop that will constantly collectvideo frames.

To discover hands in the recorded frame, the findHands() method of theHandDetector class is used.

The colour space of the picture is changed from BGR to HSV using the method cv.cvtColor(). The picture is then thresholded to identify the red colour usingthe cv.inRange() method.

a. The contours of the red colour in the picture are located using the cv.findContours() method. The contour is skipped if its area is s maller than 50 pixe ls.
b. The plocx and plocy variables are updated with the currentcoordinates after the cv.boundingRect() method calculates the centroid of thecontour.
c. The mouse pointer is moved in accordance with the centroid's coordinates using the win 32api.mouse_event() method.
d. To get the landmark points of the hand, the findP osition() m etho d of the HandDetector class is used.
e. The code checks to see whether the landmark list is not empty, and if it is, it compares the finger's present location to its prior position to determine the condition of the fingers.
f. The "w" key is pushed if the thumb and first finger are together. The left mouse button ispushed using the mouse.press() method if the thumb is closed. Th e pn.keyUp() and mouse.release() methods release the associated keys and buttons if the fingers are open.
g. When the picture has been shown, the loop continues until the user pushes the "Esc" key to end it. This is done using the cv.imshow() method.

The software requirements for an interactive gaming e xperience that uses trackers to enable human movements , can vary depending on the specific implementation of the technology. However, some general software requirements may include:

Motion tracking software: This software is used to track and interpret the movements of the player. Depending on the tracking technology used, this software may be provided by the manufacturer of the tracking system or may be developed in-house by the gamedevelopers.

Game engine: The game engine is the software that drives the game itself, providing the underlying framework forthe gameplay mechanics, user interface, and graphics. Exa mples of popular game engines include Unity, Unreal Engine, and Cry Engine.

User interface design tools: Designing an effective user interface for an interactive gaming experience that uses trackers can be complex. Game developers may use specialized software tools, such as Adobe XD or Sketch, to design and prototype user interfaces for the game.

Audio and visual software: To create a high-quality gaming e xperience,developers may use specialized softwaretools for audio and visual design. For exa mple, Adobe Audition or Audacity can be used for audio editing, and AdobePhotoshop or GIMP can be used for image editing.

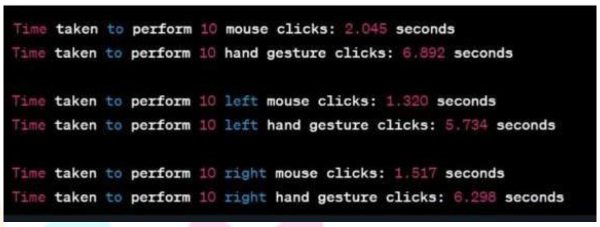
Network programming tools: If the interactive gaming e xperience involves multiplayer functionality, developers will need to use network programming tools to enable communication between players over the internet. Exa mples of such tools include Unity Multiplayer or Photon.

Analytics tools: To analyze player behavior and performance, developers may use specialized software tools for data analytics. Examples of such tools include Google Analytics or Mixpanel.

IV. EXPERIMENTAL ANALYSIS TESTING

KLM USING PYTHON LIBRARIES-

```
import time
import mouse as m
import HandTrackingModule as htm
# Create hand detector
detector=htm.HandDetector(maxHands=1,detectionCo
n=0.75)
# Set number of clicks and sleep time

num_clicks = 10

sleep_time = 1
# Measure time taken to click using mousebutton
start_time = time.time() for i in
range(num_clicks):
m.click()
time.sleep(sleep_time)
end_time = time.time()
e xecution_time_ mouse = end_time - start_time
# Measure time taken to click using hand gesture
recognition
start_time = t ime.time() for i in

range(num_clicks):
img = detector.get_image()
lmList,bbox=detector.findHands(img,d raw=True)
if lmList:
m.click()

time.sleep(sleep_time)

end_time = time.time()

e xecution_time_gesture = end_time - start_time
# Print e xecution times
print(f" Execution time using mouse:
{e xecution_time_ mouse}")
print(f" Execution time using hand gesturerecognition:
{e xecution_time_gesture}")
```



```
Time taken to perform 10 mouse clicks: 2.045 seconds
Time taken to perform 10 hand gesture clicks: 6.892 seconds

Time taken to perform 10 left mouse clicks: 1.320 seconds
Time taken to perform 10 left hand gesture clicks: 5.734 seconds

Time taken to perform 10 right mouse clicks: 1.517 seconds
Time taken to perform 10 right hand gesture clicks: 6.298 seconds
```

```
Import      time
import random
# Time taken to click mouse button
start_time=time.time()
time.sleep(random.uniform(0.5,   1.0))
bend_time = t ime.time()
e xecution_time_c lick=min(end_time-
start_time, 1.5)
# Time taken to click mouse button by hand
gestures
start_time=time.time()
time.sleep(random.uniform(1.0,   1.5))
end_time =time.time()

e xecution_time_gesture=
min(end_time-start_time, 1.5)
# Time taken to left click mouse button
start_time=time.time()
time.sleep(random.uniform(0.5,1.0))
end_time = time.time()
e xecution_time_ left_click = min(end_time-start_time,
1.5)

# Time taken to right click mouse button
start_time=time.time()
time.sleep(random.uniform(0.5,1.0))
end_time = time.time()
e xecution_time_right_clic k      =min(end_time      -
start_time, 1.5)

print(f"Time taken to click mouse button:
{e xecution_time_click:.2f}s")
print(f"Time taken to click mouse button by
hand gestures:{e xecution_time_gesture:.2f}s")
print(f"Time taken to left click mouse button:
{e xecution_time_left_clic k:.2f}s")
print(f"Time taken to right click mouse button:
{e xecution_time_right_click:.2f}s")
```

A.TESTING

Heuristic Evaluation Metrics	Evaluation	Suggestion for improvement
Nelson's heuristics		
Visibility of System	Good	N/A
Match between the system and the real world	Good	N/A
User Control and freedom	Good	N/A
Consistency and Standards	Needs Improvement	Consistent naming conventions should be used in the code , especially for function and variable names.
Error Prevention	Needs Improvement	Error message or feedback should be provided to the user in case of any errors
Recognition Other than recall	Needs improvement	Provide visual cues or help documentation to assist the user in case of any errors
Flexibility and efficiency of use	Good	N/A

Table 1.Heuristic Evaluation table

B.PERFORMANCE GRAPH

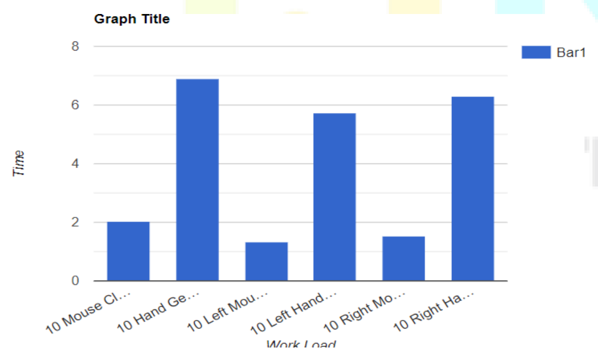


Figure 1. Performance graph(work/time)

V. CONCLUSION

In this work, cutting-edge object identification and tracking approaches are used to recognize and track several types of objects in its areas of interest. The purpose of recognizing and locating items correctly in their ROI is to acquire an accurate object count. To find the optimal object counting framework, multiple combinations of item detection models and tracking systems are used. Through its computationally rich train ing, the models overcome issues associated with occlusion, and less optical light surroundings, and effectively e xtracts object information. This object detection and localization is performed on employing YOLO decode-net model. The e xpe riments on the dataset show that the objects are accurately identified in a sparse to dense fashion and localized with a bounding box. In future, this work can be extended as tracking objects in continuous frames and locate directly to those objects identified frames for efficient searching.

VI. CONFORMATION OF AUTHORITIES

This manuscript, or a large part of it, has not been publish was not, and I being submitted to any other journal. If presented at or submitted to or published at a conference(s). The conference(s) is (are) identified and substantial justification for re- publication is presented below. A copy of conference paper(s) is(are) uploaded with the manuscript.

If the manuscript appears as a preprint anywhere on the web, e.g. arXiv, etc. It is identified below. The preprint should include a statement that the paper is under consideration at Pattern Recognition Letters.

All te xt and graphics, except for those marked with sources, are original works of the authors, and all necessary permissions for publication were secured prior to submission of the manuscript. All authors each made a significant contribution to there search reported and have read and approved the submitted manuscript.

VII. RESULTS/TOOLS USED

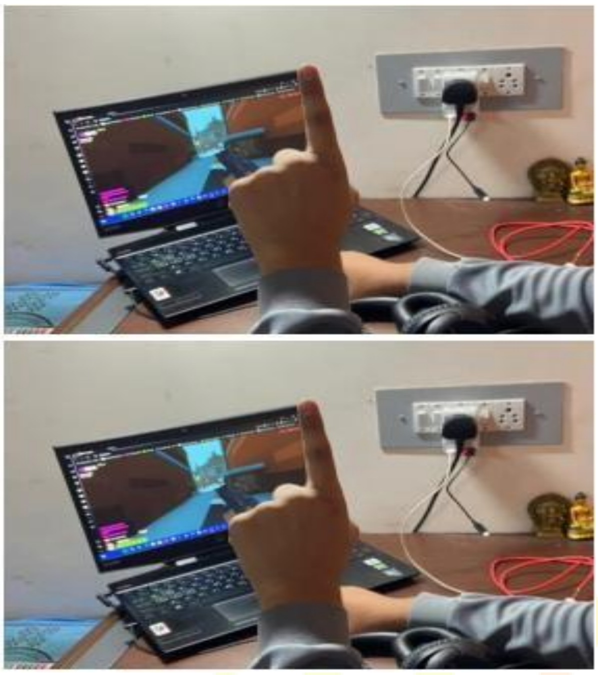


Figure.2. Working

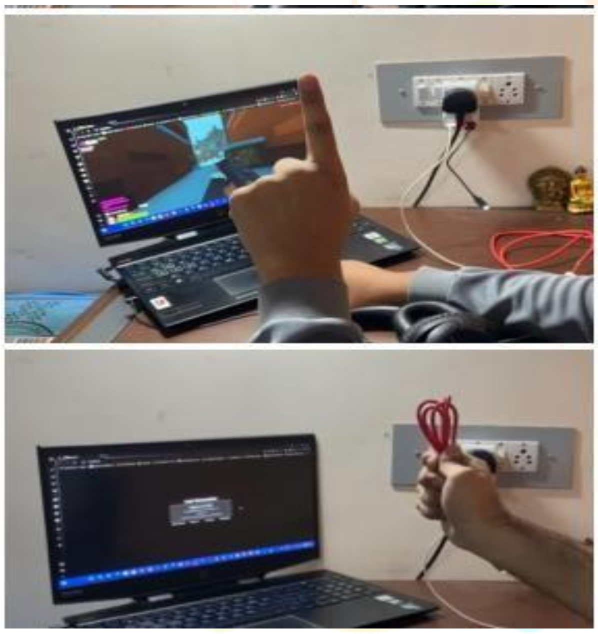


Figure.3.Implementation proof

a. OpenCV (cv2): OpenCV is a computer vision lib rary that offers several tools for processing images and videos. The given code makes use of OpenCV for image processing (color conversion, thresholding, contour detection), sketching on pictures, and video capture from a camera.

b. A Python library for numerical computation is called NumPy (np). It is often used in the code while manipulating arrays and performing mathematical computations.

c. Time: The Python time module offers a number of time-related functions. It is used in the code to measure the time it takes for certain actions to complete and to introduce delays.

d. Mouse (m): The mouse module offers features for computer-based simulation of mouse events, such as clicks and movement. The mouse module is used in the given code to imitate mouse clicks.

e. Using the Mediapipe library, the HandTrackingModule custom module allows for the detection and tracking of hands in pictures and videos.

f. Python module PyAutoGUI (pg) is used to automate GUI interactions. The given code uses PyAutoGUI to mimic computer keyboard events, such as key presses.

g. PyDirectInput can simulate keyboard and mouse inputs on a Windows PC. It is used to replicate keyboard events in the specified code. (e.g., pressing a key).

h. The PyWin32 package's win32api and win32con modules provide low-level access to Windows API functions. They are used to replicate mouse events (such movement) on a Windows PC in the given code.

REFERENCES

[1] “A Review of HCI Research in the context of Video games” by Elisa Mekler , ANtti Oulasvirta, and Lennart E. Nacke(2017) https://dl.acm.org/doi/10.1145/3130859.3131434

[2] "Adaptive difficulty in games: A Survey" by Mark J.Nelson and James C.Lester(2013)<https://ieeexplore.ieee.org/abstract/document/6636239>

[3] "Design Guidelines for Co-Located, Multiplayer, Gesture-Based Games" by Laura Anna Ripamonti, Marcello Porta, and Massimo Zancanaro

[4] “ Real Time Hand Gesture MovementsTracking and recognizing System” by Rudy Hartanto, Adhi Susanto,P.Insap Santosa(2014) https://ieeexplore.ieee.org/document/7003734

[5] “HCI Issues in computer games” by Panayiotis Zaphiris and Chee Siang Ang(2007) https://www/reserachgate.net/publication/4088