Intellecta : Journal of Artificial Intelligence
Volume. 1 Issue 1 December 2025
Page No: 1-16

Controlling Computer Device Cursors Using Hand Gestures by Utilizing OpenCV, MediaPipe, and PyAutoGui

Muhammad Juzairi Safitli[1], Fattachul Huda Aminuddin[2], Gustina[3]
Nurdin Hamzah University, Indonesia[123]
Coresspondent : fattachulhuda@unh.ac.id [2]

Received : October 10, 2025
Accepted : November 13, 2025
Published : December 30, 2025

Citation: Safitrli, M, J., Aminuddin, F, H., & Agustina. (2025). Controlling Computer Device Cursors Using Hand Gestures by Utilizing OpenCV, MediaPipe, and PyAutoGui. Intellecta : Journal of Artificial Intelligence, 1(1), 1-16.

ABSTRACT: This research presents an innovative interactive solution for controlling computer device cursors by utilizing hand movements using OpenCV, MediaPipe, and PyAutoGui. By combining these three libraries, this program is able to detect and analyze hand movements in real-time, allowing users to control the cursor and click with hand movements. Through image processing methods and hand landmark analysis, this program succeeded in achieving its designed goals. The main goal of the research is to create interactive solutions that are responsive and efficient in controlling computer device cursors. Test results on computer devices with certain specifications show success in detecting hand movements and controlling the cursor well. However, this research still has the potential for further development, such as adding complex hand gesture recognition features and integration with certain applications to improve program functionality and performance.

Keywords: OpenCV, Python, MediaPipe, Pyautogui, Controlling the Cursor.

INTRODUCTION

Python was created by Guido van Rossum in the Netherlands in 1990 and its name was taken from Guido's favorite television show Monty Python's Flying Circus. Van Rossum developed Python as a hobby, then Python became a programming language that is widely used in industry and education because it is simple, concise, has intuitive syntax and has an extensive library (Romzi & Kurniawan, 2020b)

Python is an interpretive programming language that is considered easy to learn and focuses on code readability. In other words, Python is claimed to be a programming language that has programming codes that are very clear, complete, and easy to understand (Enterprise, 2019). Jubilee (2019) added that Python is a simple programming language for creating artificial intelligence-based applications (*artificial intelligence*).

*Artificial Intelligence*or artificial intelligence is a computer system capable of performing tasks that normally require human intelligence. This technology can make decisions by analyzing and using the data available in the system. The processes that occur in Artificial Intelligence include:*learning*, *reasoning*, and*self-correction*. This process is similar to humans conducting analysis before making a decision (Lubis, 2021).

Some Python Language functions are usable in*server* to create web applications, Python can be used alongside software to create workflows, and can be connected to systems*database*, and can read and modify*file*. Python can be used to handle big data and perform complex mathematics, and can be used for rapid prototyping, or for production-ready software development(Ma'Arif, 2020). In everyday life, Python can also be used in industrial fields such as game development, data science, and machine learning (Wali M, Nengsih TA, Hts DI, Choirina P, Awaludin AA, Yusuf M, Aminuddin FH, 2023, p. 143).

In Python, there is*library*. *Library* Python will handle various existing functions. So when you want to use a function, you have to call it*library* who supervised him first. The library can be likened to a house, if we want to meet someone we have to go to their house first.*Library* in Python it is a combination of groups*package* and*module* with the same functionality with the aim of making it easier to create an application, without having to write a lot of code.*Library* on Python has more than 140,000 libraries that are developed through*open-source* projects. By using libraries in Python, you can generate code efficiently and save time without having to write entire scripts. The library is also reusable, which means it can be used many times, anywhere and at any time. Some examples*library* commonly used by data practitioners, namely TensorFlow, Scikit-Learn, Numpy, Keras, PyTorch, LightGBM, Eli5, SciPy, Theano, and Pandas (Miftah, 2021).

The importance of libraries in the Python ecosystem cannot be ignored. Therefore, this research uses three main libraries, namely OpenCV, MediaPipe, and PyAutoGui, to design a program that allows controlling the cursor of a computer device by utilizing hand movements.

Open Computer Vision (OpenCV) is*library open-source* whose purpose is specifically for image processing. Image processing in question is so that computers have capabilities similar to the way visual processing is done in humans(fattachulhudacom, 2023). OpenCV has provided many basic computer vision algorithms. OpenCV also provides an object detection module that uses methods*computer vision* (Zulkhadi et al., 2019).

Library The next thing to use is MediaPipe. MediaPipe is a framework (*framework*) which is used to build machine learning networks to process time series data, including video. The use of MediaPipe in several studies has also shown promising results (Rasyid et al., 2022).

Library The last one to use is PyAutoGui. By using*library* here, console*command prompt* can control*mouse* and*keyboard*. This capability is used to interact with other connected applications*engine design*. *Library* This is part of Python so it is compatible with various operating systems such as Windows, Mac, and Linux (Pradono et al., 2020).

Based on previous research conducted by Jawas, N (2017) regarding hand movement tracking for gesture recognition using the Shi and Tomasi algorithm to determine corner points in images, this research will also apply the Shi and Tomasi algorithm to provide an interactive, innovative solution control the cursor with hand movements(Jawas, 2017). Concludes that an algorithm is a method or series of efficient, written, and sequential steps that contains a set of instructions for completing a task, where the task must be completed systematically, using logical language, and with clear goals(Harahap, 2023).

By combining these three libraries, this research seeks to create an innovative interactive solution for controlling the cursor of a computer device using hand movements. The results of implementing this application are also an implementation of learning from computer graphics and image processing courses at Nurdin Hamzah University.

METHOD

This research is focused on developing an interactive solution for controlling computer device cursors using hand movements. By using three main libraries, namely OpenCV, MediaPipe, and PyAutoGui, it is possible to implement innovative programs. In system development, the camera object will process starting from initialization*library* The detection object is a hand as the source object. Next process*frame* The video will live and detect hand movements on the camera. Detected objects are in the form of points which will later be recognized by MediaPipe(Parashar et al., 2023; Samaan et al., 2022; Subramanian et al., 2022). Each object will be captured and detected into an image. In more detail, this process will be explained using*flowchart. Flowchart* (flow chart) is an illustration in the form of a flow diagram of the algorithms in a program, which states the direction of flow of the program (Dr. Aneu Yulianeu & Oktamala, 2022). As for*flowchart* for the program in this research can be found in picture 1.

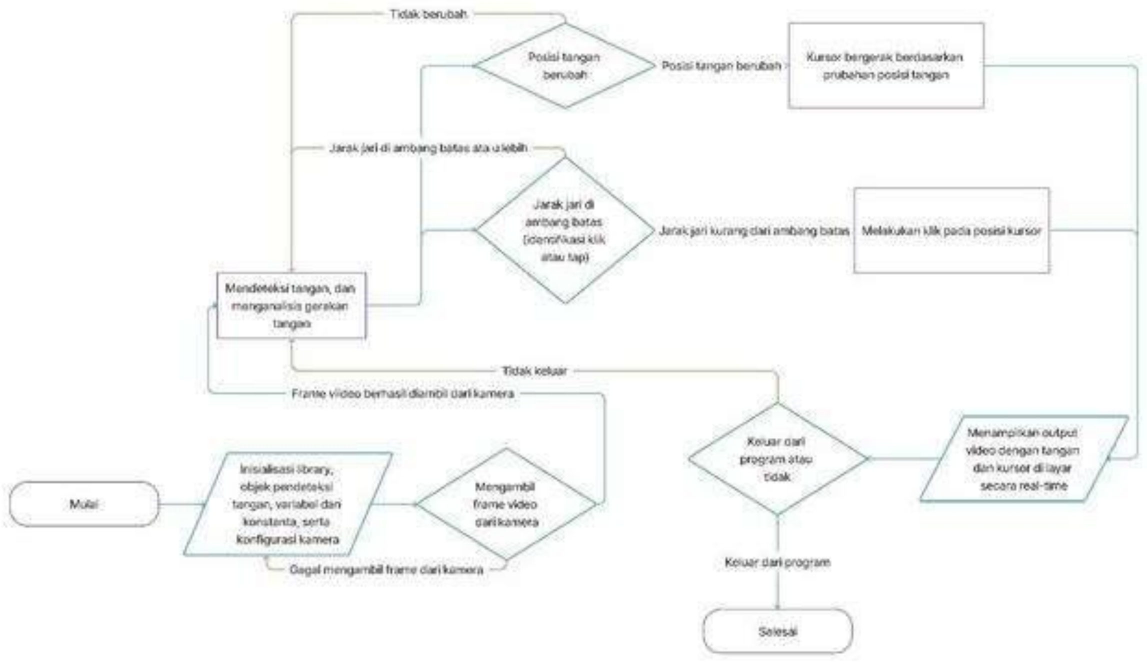
Tidak berubah
Posisi tangan berubah
Posisi tangan berubah
Kursor bergerak berdasarkan prubahan posisi tangan
Jarak jari di ambang batas ata u lebih
Jarak jari di ambang batas (identifikasi klik atau tap)
Jarak jari kurang dari ambang batas
Melakukan klik pada posisi kursor
Mendeteksi tangan, dan menganalisis gerakan tangan
Tidak keluar
Frame video berhasil diambil dari kamera
Keluar dari program atau tidak
Menampilkan output video dengan tangan dan kursor di layar secara real-time
Mulai
Inisialisasi library, objek pendeteksi tangan, variabel dan konstanta, serta konfigurasi kamera
Mengambil frame video dari kamera
Gagal mengambil frame dari kamera
Keluar dari program
Selesai

Figure 1. Program Workflow Flowchart

In detail, explanation*flowchart* in Figure 1 above can be explained as follows.

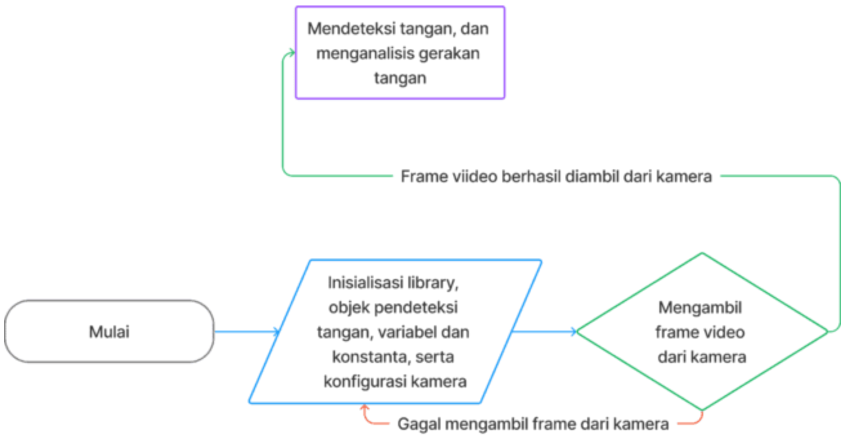
Mendeteksi tangan, dan menganalisis gerakan tangan
Frame viideo berhasil diambil dari kamera
Mulai
Inisialisasi library, objek pendeteksi tangan, variabel dan konstanta, serta konfigurasi kamera
Mengambil frame video dari kamera
Gagal mengambil frame dari kamera

Figure 2. Initial Program Workflow Flowchart

Start

In this step, the program will start and execute according to the program flow that has been created.

Initialize Library

This step will do*import* and initialize OpenCV, MediaPipe, and PyAutoGui libraries. This step also ensures that all required modules and functions are ready to use. Before*library* recognized, the installation and configuration process is carried out from*library* which will be used first.
The stages in the OpenCV Python installation process are as follows (fatachulhuda.com, 2023).
a. Download and install Idle python on the official site atwww.python.org
b. Check using*command prompt* (CMD) on Windows.
c. Install OpenCV with the command: 'pip install-opencv-python'
d. Then the OpenCV library will be downloaded automatically.
e. To initialize and*import* do the command with 'import cv2 as cv'.

Next is to carry out the MediaPipe and PyAutoGui installation process, as for the second installation stage*library* these are as follows.

a. Open*command prompt* (CMD) on Windows.
b. Install MediaPipe with the command: 'pip install mediapipe' (MediaPipe, 2023).
c. The MediaPipe library will be downloaded automatically.
d. With the same CMD, the next step is to install PyAutoGui with the command: 'pip install pyautogui' (PyAutoGui, 2019).
e. The PyAutoGui library will be downloaded automatically.
f. To initialize and*import* This is done with the 'import mediapipe as mp' command for MediaPipe, and the 'import pyautogui as pygui' command for PyAutoGui.

Camera Configuration

The third step is to set the camera to gain*input* video online*real-time*. *Real-time* is a very fast control mechanism, data recording, processing so that*output* results can be received in relatively the same time (Rahman & Bambang, 2021).

Hand Detection Object Initialization

In this step, the program will create a hand detection object using MediaPipe(Han et al., 2022; Kim et al., 2023). This step will also initialize detection parameters such as static or dynamic mode, number of detected hands, and detection confidence level.

Initializing Variables and Constants

The next step is to define the variables necessary to track the cursor position and click status. This step also determines constants such as movement threshold values, detection intervals, and so on.

Video Capture Loop

The sixth step of the program has entered*loop* or main loop.*Loop* is a useful and frequently used feature in all modern programming languages, especially Python.*Loop* useful when wanting to automate certain tasks . In this program,(Lemonaki, 2022)*loop* used for continuous taking*frame* from camera.

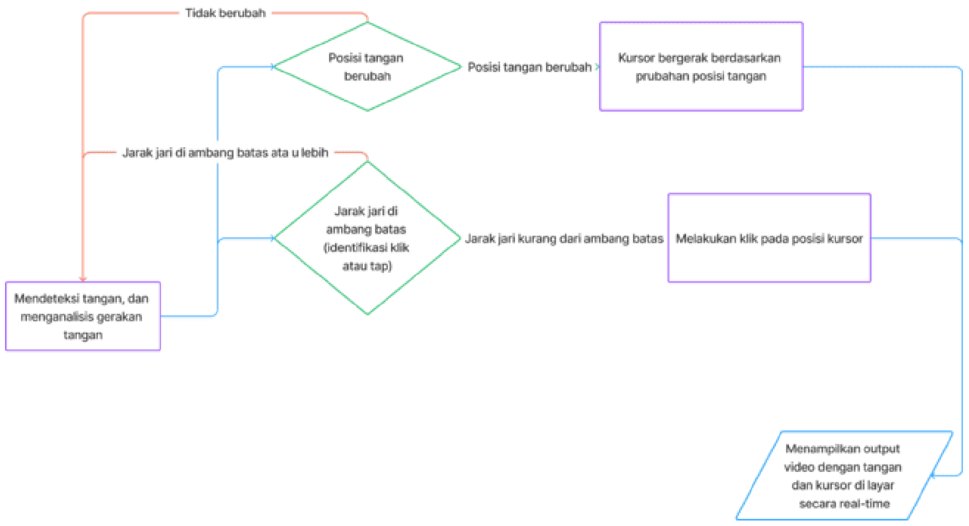
Tidak berubah
Posisi tangan berubah
Posisi tangan berubah
Kursor bergerak berdasarkan prubahan posisi tangan
Jarak jari di ambang batas ata u lebih
Jarak jari di ambang batas (identifikasi klik atau tap)
Jarak jari kurang dari ambang batas
Melakukan klik pada posisi kursor
Mendeteksi tangan, dan menganalisis gerakan tangan
Menampilkan output video dengan tangan dan kursor di layar secara real-time

Figure 3. Flowchart of Hand Detection and Finger Analysis Program Workflow

Hand Detection

The seventh step the program will take*frame* video from the camera and uses a hand detection object to detect the hand inside*frame*.

Hand Movement Analysis

In the program results, if a hand is detected, the program will get hand landmarks (hand frame points). Landmarks or handframe points are objects that contain the coordinates of skeleton points recognized by Mediapipe.

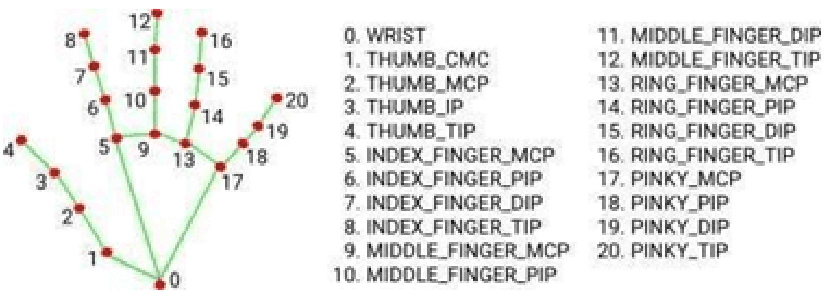

Figure 4. MediaPipe hand framework points (Suyudil et al., 2022)

Next, it will calculate the change in hand position from the previous frame. This aims to adjust the cursor position based on changes in hand position.

Cursor Control

In the next step, the program will use PyAutoGui to control the cursor based on changes in hand position, and move the cursor on the screen according to changes in hand position.

Click Detection

The program will calculate the monitor distance between fingertips to detect clicks or*tap* (touch). If the distance is less than the predetermined threshold from step five, it will be identified as a click or*tap*. Then the program will use PyAutoGui to click at the cursor position.

Show Video Output

The program will display*output* video with hand and cursor on screen.*Frame* Pre-processed videos will also be displayed automatically*real-time*.

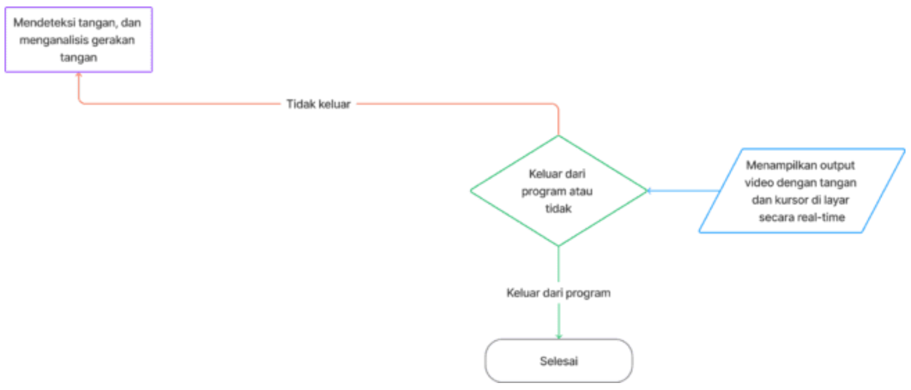
Mendeteksi tangan, dan menganalisis gerakan tangan
Tidak keluar
Keluar dari program atau tidak
Menampilkan output video dengan tangan dan kursor di layar secara real-time
Keluar dari program
Selesai

Figure 5. Final Program Workflow Flowchart

Stop

This process indicates stopping or ending a program. For the exit process,*user* or the user must press the exit button then the program. Then the program will automatically exit and do the work*loop* or looping and ending the program (stopping).

RESULT AND DISCUSSION

The program was created and designed using Visual Studio Code as*code editor*. *Source code editor*, or short*code editor*, is one type of *text editor* which are available.*Code editor* this is*text editor* which is devoted to writing kdoe-code for the software being developed.*Code editor* this allows developers to write and read*source code* more easily, because*code editor* this provides a kind of*highlight* different for each code element so that the codes can be easier to see, so*code editor* easier to use in software development than*text editor* normal There are many*code editor* that can be used, but there are three*code editor* which are relatively popular, namely Sublime, Atom, and Visual Studio Code (Dea, 2020).

Visual Studio Code is one*code editor* which was developed by a giant company in the technology sector, namely Microsoft. Visual Studio Code is capable of operating on desktop devices based on Mac OS, Windows and Linux. Visual Studio Code is*code editor* Which*powerful* because it can be used for editing*source code* various languages such as Typescript, Javascript, PHP, to Python (Patria, 2023)

Additionally, Visual Studio Code can be installed*extension*which functions to facilitate the creation and design of programming, especially Python. To be able to run Python code, installation is required*extension* Python. The way to do this is to click the tab*Extensions*, type Python, and select Python, then click Install (Romzi & Kurniawan, 2020a)

The following is*source code* or program source code moves the cursor using hand movements based on the flow (*flowchart*) program that has been planned in picture 1.

```
import cv2 as cv
import mediapipe as mp
import pyautogui as pygui

camera = cv.VideoCapture(0)
mp_hands = mp.solutions.hands
tangan = mp_tangan.Hands(static_image_mode=False, max_num_hands=1, min_detection_confidence=0.5,
min_tracking_confidence=0.5)

try:
    count = 0
    DETECTION_INTERVAL = 5
    MOVEMENT_THRESHOLD = 20
    CLICK_THRESHOLD = 15

    prev_position_x, prev_position_y = 0, 0
    clicked = False

    while True:
        baca, frame = camera.read()
        if not baca:
            break

        count += 1
        if count % DETECTION_INTERVAL == 0:
            frame_rgb = cv.cvtColor(frame, cv.COLOR_BGR2RGB)
            result = hand.process(frame_rgb)
            if hasil.multi_hand_landmarks:
                for landmarks in hasil.multi_hand_landmarks:
                    for id, landmark in enumerate(landmarks.landmark):
                        posisi_x, posisi_y = int(landmark.x * frame.shape[1]), int(landmark.y * frame.shape[0])
                        cv.circle(frame, (x_position, y_position), 5, (0, 255, 0), -1)

                    posisi_x = int(landmarks.landmark[mp_tangan.HandLandmark.INDEX_FINGER_TIP].x * frame.shape[1])
                    posisi_y = int(landmarks.landmark[mp_tangan.HandLandmark.INDEX_FINGER_TIP].y * frame.shape[0])

                    distance = ((position_x - previous_position_x)**2 + (position_y - previous_position_y)**2)**0.5
                    if jarak > MOVEMENT_THRESHOLD:
                        pygui.moveTo(x_position, y_position)

                    if not clicked and jarak < CLICK_THRESHOLD:
                        pygui.click()
                        clicked = True
                    elif clicked and jarak >= CLICK_THRESHOLD:
                        clicked = False

                    prev_position_x, prev_position_y = position_x, position_y

        cv.imshow('Kamera', frame)
        if cv.waitKey(1) & 0xFF == ord('q'):
            break
finally:
    camera.release()
    cv.destroyAllWindows()
```

1. Import Library

```
import cv2 as cv
import mediapipe as mp
import pyautogui as pygui
```

In this section, three will be imported*library* used in the program. As for the third*library* are as follows.

a. First, 'cv2' is used for reading and processing*frame* from the camera.
b. Second, 'mediapipe' is used for detection*landmarks* on hand
c. Third, 'pyautogui' is used to control movement*mouse* and click.

2. Initialize Camera and Medipipe

```
camera = cv.VideoCapture(0)
mp_hands = mp.solutions.hands
tangan = mp_tangan.Hands(static_image_mode=False, max_num_hands=1, min_detection_confidence=0.5,
min_tracking_confidence=0.5)
```

The program begins by initializing the 'camera' object using 'cv.VideoCapture(0)' to access the camera. Next, the 'mp_hand' and 'hand' objects are initialized to use the hand detection module from*library* MediaPipe.

3. Initializing Variables and Constants

```
count = 0
   DETECTION_INTERVAL = 5
   MOVEMENT_THRESHOLD = 20
   CLICK_THRESHOLD = 15

   prev_position_x, prev_position_y =
0, 0
   clicked = False
```

Variables and constants are declared to organize several aspects of detection and control*mouse*. 'count' is used to count iterations, while 'DETECTION_INTERVAL', 'MOVEMENT_THRESHOLD', and 'CLICK_THRESHOLD' are used to control hand detection and action from*mouse*. ’prev_position_x’ and ’prev_position_y’ store positions*mouse* on*frame* before, and ’clicked’ marks whether*mouse* has been clicked or not.

Iteration is used to optimize program performance by performing hand detection on only a few*frame* certain of each*frame* read from the camera, with the aim of optimizing performance, processing and effective data filtering. This is proven in research conducted by Prihatiningsih, M, Andriani, & Nugraha (2019), that performance is influenced by the number of iterations(Prihatiningsih et al., 2019).

4. Loop (Repetition)

```
while True:
```

Entering program*loop* which continues to run as long as the camera is still reading*frame*. Each iteration of*loop* will read*frame* from camera and hand detection only every few*frame*, corresponds to ’DETECTION_INTERVAL’ from the previous section (initialization of variables and constants).

5. Reading*Frame* and Hand Detection

```
baca, frame = camera.read()
      if not baca:
         break

      count += 1
      if count % DETECTION_INTERVAL == 0:
         frame_rgb = cv.cvtColor(frame, cv.COLOR_BGR2RGB)
         result = hand.process(frame_rgb)
```

In each iteration that meets the condition 'DETECTION_ITERVAL",*frame* from the camera is read, and hand detection is carried out using*library* MediaPipe. The detection results are then used to obtain coordinates *landmarks* on hand(MediaPipe, 2023)

6. Detection*Landmarks* Control*Mouse*

```
for landmarks in hasil.multi_hand_landmarks:
                for id, landmark in enumerate(landmarks.landmark):
                   posisi_x, posisi_y = int(landmark.x * frame.shape[1]), int(landmark.y * frame.shape[0])
                   cv.circle(frame, (x_position, y_position), 5, (0, 255, 0), -1)
```

```
          posisi_x = int(landmarks.landmark[mp_tangan.HandLandmark.INDEX_FINGER_TIP].x * frame.shape[1])
          posisi_y = int(landmarks.landmark[mp_tangan.HandLandmark.INDEX_FINGER_TIP].y * frame.shape[0])

          distance = ((position_x - previous_position_x)**2 + (position_y - previous_position_y)**2)**0.5
          if jarak > MOVEMENT_THRESHOLD:
              pygui.moveTo(x_position, y_position)

          if not clicked and jarak < CLICK_THRESHOLD:
              pygui.click()
              clicked = True
          elif clicked and jarak >= CLICK_THRESHOLD:
              clicked = False
```

This part involves iterating through each detected hand and*landmarks* on hand. The position of the index finger tip ('INDEX_FIGER_TIP') is taken and used to measure the distance the hand moves. If the movement exceeds a certain threshold, the program uses 'pyautogui' to move*mouse* to the appropriate position. In addition, the program also controls clicks*mouse* based on movement threshold.

7. Displaying*Frame* and Exit the Program

```
cv.imshow('Kamera', frame)
        if cv.waitKey(1) & 0xFF == ord('q'):
            break
```

Each*frame* which has been processed is displayed using OpenCV. The program also checks whether the 'q' or exit key was pressed, and if so, the program exits*loop*.

8. The program closes

```
finally:
    camera.release()
    cv.destroyAllWindows()
```

The last part of the program is used to release the resources in use, namely the camera, and close all OpenCV windows after the program has successfully run. This serves to ensure that system resources are properly freed.

Implementation of the program is carried out using a computer device that has a camera with sufficient lighting. As for installing Python, you need to use version 3.12, while for specifications you need to use a computer or laptop*operation system* (os) Windows 11 with CPU (*Central Processing Unit*) Intel Core i3 1005G1. The camera used is an external camera from Jovitech CM08 Full HD Webcam with a resolution of 1920x1080. Detailed specifications of the devices used are in table 1.

Model	ACER 215-52
YOU	Windows 11
CPU	Intel Core i3 1005G1
Memory	Tipe : DDR4 Size : 12 GB Channel : Dual
Grapics	Intel® 630 UHD Graphics
Camera	Jovitech CM08 Full HD Webcam

Table 1. Specifications of the Devices Used

The program managed to run well using computer equipment according to the specifications in table 1. As seen in figure 6, the program was successful*input* video online*real-time* and constantly taking*frame* from the user's camera or*user*. The program also managed to detect the hand that was in*frame* and get*landmark* hand containing outline points recognized by MediaPipe.

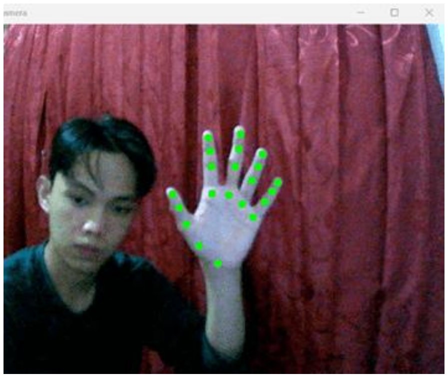

Figure 6. Landmarks or points of the hand framework

Using PyAutuGui was also successful in controlling the cursor based on changes in hand position. As seen in Figure 7, Figure 8 and Figure 9, the cursor (yellow and in the form of a sum symbol) will always follow the landmark or tip of the user's finger that is captured in the real-time video input frame. The cursor also clicks or taps based on the distance between the user's fingertips.

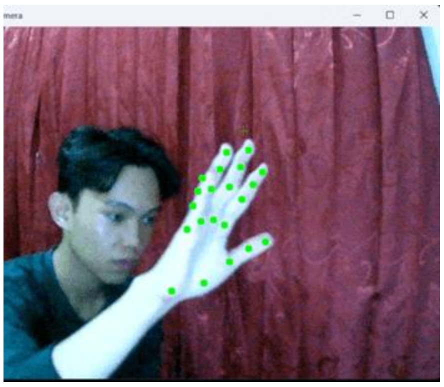

Figure 7. The cursor follows a landmark or point on the fingertip

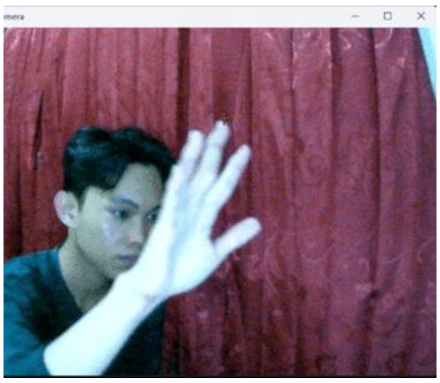

Figure 8. View without landmarks or frame points

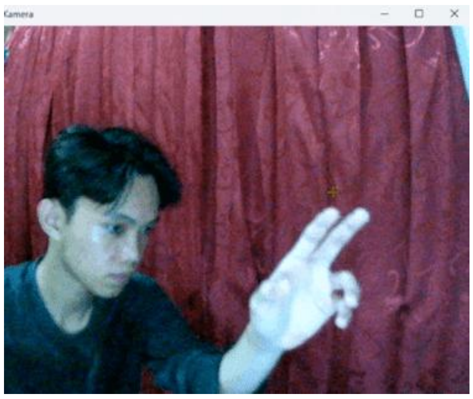

Figure 9. The cursor clicks or taps based on the distance from the fingertip

Next is testing the use of two hands to control the cursor. The first test will be carried out with different distances between hands. The distance of both hands from the camera can be seen in table 2.

	Left Hand Distance	Distance Right hand
Testing 1	75 cm	45 cm
Testing 2	65 cm	50 cm
Testing 3	78 cm	47 cm

Table 2. Hand distance from the camera

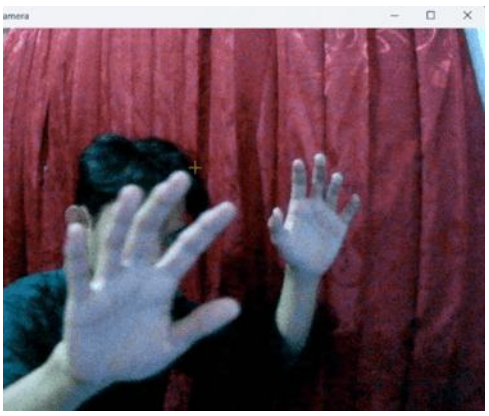

Figure 10. First test using two hands

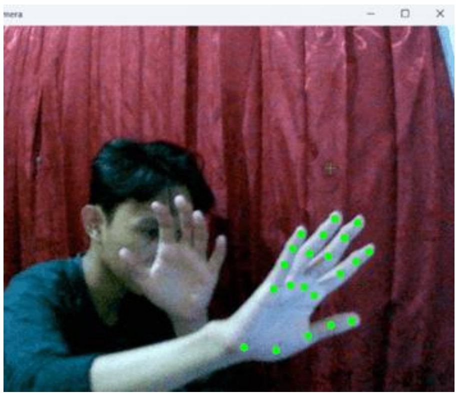

Figure 11. The second test uses two hands with landmarks

As seen in Figure 10 and Figure 11, the cursor only follows the movements of the right hand which is closer than the left hand. Even if the hand position is changed, the program will still only detect the hand closest to the camera.

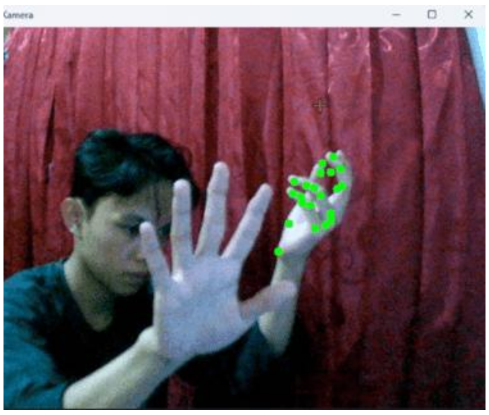

Figure 11. The third test uses two hands with landmarks

However, different from the previous test, in the third test (figure 11), the program follows the hand that is furthest away (the left hand). This shows the program's ability to identify the relevant ones, although there are some cases where the hand priority may change.

Although this program has achieved its goal of controlling the cursor of a computer device with hand movements, there is still potential for further development. Additional features, such as more complex hand gesture recognition or integration with certain applications, can increase the functionality of these programs.

CONCLUSION

This research succeeded in developing an innovative interactive solution for controlling computer device cursors by utilizing hand movements. The implementation of this program is carried out by combining three main libraries, namely OpenCV, MediaPipe, and PyAutoGui, within the scope of Python programming.

Through testing carried out on computer devices with certain specifications, the program succeeded in obtaining*input* video online*real-time* from the user's camera. Detection of hands and hand landmarks using MediaPipe works well, allowing the program to identify the position of the fingertips.

The use of PyAutoGui succeeded in controlling the computer cursor with a good response to changes in hand position. The program can also detect clicks or taps based on the distance between fingertips, adding interactive functionality to the program.

Testing with two hands demonstrated the program's ability to recognize the hand closest to the camera. Despite some situations where hand priorities may change, the overall performance of the program provides satisfactory results.

However, this research still has the potential for further development. Adding features such as more complex hand gesture recognition or integration with certain applications can improve the functionality and performance of these programs.

REFERENCE

Jubilee Enterprise. (2019). *Python for Beginner Programmers.* Jakarta: PT Elex Media Komputindo.

Lubis, M. S. (2021). Implementation of Artificial Intelligence in Integrated Manufacturing Systems.*Posing of the UISU National Engineering Seminar*, 1-7. Retrieved January 26, 2024, from https://jurnal.uisu.ac.id/index.php/semnastek/article/download/4134/2966

PyAutoGui. (2019).*Installation.* Retrieved January 25, 2024, from PyAutoGUI: https://pyautogui.readthedocs.io/en/latest/install.html

Wali, M., Nengsih, T. A., Hts, D. I. G., Choirina, P., Awaludin, A. A. R., Yusuf, M., ... & Baradja, A. (2023). INTRODUCTION TO THE 15 BEST PROGRAMMING LANGUAGES OF THE FUTURE (Reference & Coding For Beginners). PT. Sonpedia Publishing Indonesia.

Dea, G. (2020). *January 2).Get to know the Code Editor, and the Three Code Editors.* https://vantura.id/blog/mengenal-code-editor-dan-tiga-code-editor-terpopular

Dr. Aneu Yulianeu, S., & Oktamala, R. (2022). *Web-Based Public Transport Route Information System in Tasikmalaya City.Journal of Information Engineering* (pp. 125–134). http://jurnal.stmik-dci.ac.id/index.php/jutekin/

Enterprise, J. (2019). *Python for Beginner Programmers.* PT Elex Media Komputindo.

fattachulhudacom. (2023). *Image Processing Using OpenCV Integrated with Python.* https://www.fattachulhuda.com/2023/11/pengolahan-citra-using-opencv.html

Han, J.-S., Lee, C.-I., Youn, Y.-H., & Kim, S.-J. (2022). A Study on Real-time Hand Gesture Recognition Technology by Machine Learning-based MediaPipe. *Journal of System and Management Sciences*, *12*(2), 468–482. https://doi.org/10.33168/JSMS.2022.0225

Harahap, H. R. (2023). *Concepts and Functions of Programming Algorithms.Journal of Mathematics and Natural Science* (pp. 254–257). https://doi.org/10.59581/konsanta-widyakarya.v1i4.1875

Jawas, N. (2017). *Hand Movement Tracking for Gesture Recognition.Informatic Technique Journal* (pp. 13–23). https://www.e-journal.potensi-utama.ac.id/ojs/index.php/ITJournal/article/download/364/337

Kim, J.-W., Choi, J.-Y., Ha, E.-J., & Choi, J.-H. (2023). Human Pose Estimation Using MediaPipe Pose and Optimization Method Based on a Humanoid Model. *Applied Sciences (Switzerland)*, *13*(4). https://doi.org/10.3390/app13042700

Lemonaki, D. (2022). *Loops in Python - While True Loop Statement Example*.

Ma'Arif, A. (2020). *Programming Language Advanced Programming Textbook*. https://eprints.uad.ac.id/32743/1/buku%20python.pdf

MediaPipe. (2023). *September 13).Project Mediapipe*. https://pypi.org/project/mediapipe/

Miftah, S. (2021). *Mei 17).Python Library Get to know the differences between modules, packages and libraries in Python* (A. W. Davita, Ed.). https://dqlab.id/library-python-kenali-perbedaan-module-package-dan-library-pada-python

Parashar, D., Mishra, O., Sharma, K., & Kukker, A. (2023). Improved Yoga Pose Detection Using MediaPipe and MoveNet in a Deep Learning Model. *Revue d'Intelligence Artificielle*, *37*(5), 1197–1202. https://doi.org/10.18280/ria.370511

Patria, R. (2023). *Studio Code Complete Definition, Features*. Advantages! https://www.domainesia.com/berita/visual-studio-code/#Visual_Studio_Code_dengan_Code_Editor

Pradono, K. A., Safitri, Y. D., Adhiatma, B. S., Hestrio, Y. F., Soleh, M., Gunawan, H., & Sunarmodo, W. (2020). *Development of Automatic Landsat Data Download Engine.IOP Conf.* Materials Science and Engineering. https://doi.org/10.1088/1757-899X/1007/1/012109

Prihatiningsih, S., M, N. S., Andriani, F., & Nugraha, N. (2019). Performance Analysis of Handwritten Number Recognition Based on the Number of Iterations Using the Convolutional Neural Network. *Method.Scientific Journal of Technology and Engineering*, 58–66. https://ejournal.gunadarma.ac.id/index.php/tekno/article/download/1934/1644

Rahman, M. F., & Bambang. (2021). Trash Detection in Real-time Video using the Faster R-CNN. *Method.Applied Technology and Computing Science Journal*, 117–125. https://doi.org/10.33086/atcsj.v3i2.1846

Rasyid, M. F., Mustafa, M. S., & Suradi, A. A. (2022). *Eye Detection in Smartphone Videos Using Mediapipe Python.JOINTECS* (pp. 49–56). Journal of Information Technology and Computer Sciense. https://www.researchgate.net/publication/371968761_Eye

Romzi, M., & Kurniawan, B. (2020a). *Implementation of Python programming using Visual Studio Code.Journal of Informatics and Computers, 1-9.* https://jurnal.unmaha.ac.id/index.php/jik/article/view/11

Romzi, M., & Kurniawan, B. (2020b). Learning Python Programming with an Algorithmic Logic Approach.JTIM. *Mahakarya Information Engineering Journal*, 37–44. https://journal.unmaha.ac.id/index.php/jtim/article/view/6

Samaan, G. H., Wadie, A. R., Attia, A. K., Asaad, A. M., Kamel, A. E., Slim, S. O., Abdallah, M. S., & Cho, Y.-I. (2022). MediaPipe's Landmarks with RNN for Dynamic Sign Language Recognition. *Electronics (Switzerland)*, *11*(19). https://doi.org/10.3390/electronics11193228

Subramanian, B., Olimov, B., Naik, S. M., Kim, S., Park, K.-H., & Kim, J. (2022). An integrated mediapipe-optimized GRU model for Indian sign language recognition. *Scientific Reports*, *12*(1). https://doi.org/10.1038/s41598-022-15998-7

Suyudil, I., Sudadio, S., & Suherman, S. (2022). *Introduction to Indonesian Sign Language using Mediapipe with Random Forest and Multinomial Logistic Regression Models.Jurnal Ilmu Siber dan Teknologi Digital (JISTED* (pp. 65–80). https://doi.org/10.35912/jisted.v1i1.1899

Zulkhadi, T. C.-S., Maria, E., & Yulianto. (2019). *Facial Shape Pattern Recognition with OpenCV.JURTI, 181-186.* https://e-journals.unmul.ac.id/index.php/INF/article/view/4033