Quality of Service Analysis in Hand Gesture Identification Application Based on Convex Hull Algorithm to Control a Learning Media

Mitodius Nicho Swacaesar Setiawan[1], Nurul Hidayati[2]*, Rachmad Saptono[3]
[1,3] Digital Telecommunication Network Study Program,
Department of Electrical Engineering, State Polytechnic of Malang, Malang City, 65141, Indonesia
[2] Telecommunication Engineering Study Program,
Department of Electrical Engineering, State Polytechnic of Malang, Malang City, 65141, Indonesia
[1]mito.nicho@gmail.com, [2]nurulhid8@polinema.ac.id, [3]rachmad.saptono@polinema.ac.id

Abstract— **The technique of human interaction with computers has developed very rapidly. Body movement is the easiest and most expressive way and hand movements are flexible. We can use gestures as a simple identification command. This research discusses the application of hand gesture reading as a remote control for learning media and analyzes its quality of service using Wireshark software. This system uses a Raspberry Pi as a computing center. Raspberry Pi reads hand gestures using the Convex Hull algorithm, which can read the number of fingers raised on the hand by taking the outermost point in the contour scan of the hand. Each gesture in the form of the number of fingers was allocated to the keyboard commands used to select answers on the learning media. On the learning media side, a quiz system was created that uses keyboard commands to select the answers. Then the Raspberry Pi is connected to the laptop using a third-party application called VNC. Based on QoS measurement results, the throughput result is 62 kbps, which is included in the very good category. Packet Loss of 0%, which is also included in the very good category. The delay of 52.2 ms, which is included in the very good category. Thus, the overall quality of the network can be categorized as very good.**

Keywords— ***Convex Hull, Hand Gesture Detection, Learning Media, OpenCV, Quality of Service.***

I. INTRODUCTION

The interaction between people and computers is always changing as technology advances. At first it still used a keyboard and mouse, but now it can already use touch screen technology. Now that interactions are more natural and users may communicate with one another through body language, speech, facial expressions, and eye contact, the next revolution is beginning to take shape.

The computer is able to easily understand the user's intentions with the support of advanced sensors and algorithms. This may increase the system's activity and interaction. Users and computers still interact using mouse in their communication. Given that it is simpler to recognize hand gestures, hand gesture recognition can be utilized to facilitate interaction rather than using a mouse.

There are various situations that occur during lectures that require lecturers to stay in the lecturer's room while teaching students in class. Lecturers can present material via Google Meet or Zoom and quizzes may be given by lecturers at certain points. There are a number of mobile applications that can be used as the quiz system itself, however the system necessitates that students bring their own smartphones, and sometimes there are devices that are incompatible with the program. Another issue is the inconsistent and sometimes missing internet connectivity between students.

A presentation control system was developed by M.H. Khoirul using hand gesture recognition. The Support Vector Machine Classification technique is used by the system to classify skin tones according to background color. The camera for this experiment was a Logitech C270 webcam, and the data processing system was an Intel NUC5i7RYH. There are only five keyboard commands available for the system [1].

A hand gesture identification system that D.R.M. Harika developed can be used to control the mouse pointer. In this work, the mouse pointer was smoothed using the Kalman filter method and hand movements were identified using the color detection method [2].

An android-based system for hand gesture recognition was developed by M. Anshary. To recognize hand motions, the system use the convex hull method and convexity defects. The background color and light intensity in this system are adjustable in order to determine the background color and light intensity that work best for identifying hands [3].

A. R. Adnan used PoseNet and the K Nearest Neighbor algorithm to construct a system for identifying human arm movements. This motion identification is utilized to regulate the flame of two switches connected over the IoT network [4]. Another example is the Convolutional Neural Network (CNN)-based hand gesture detection system developed by A. Adi. This system is implemented in wheeled robots, where there are 12 hand gestures to control the 12 maneuvers of wheeled robots [5]. A finger tracking device was also developed by Alan Tompunu using a Raspberry Pi 3. OpenCV and Python were used in this research. An accuracy rate of 75% and an average video processing time of 9.2 seconds were the result of this study [6].

In this project, a hand gesture identification system was developed for learning media in the form of a quiz program. Convex hull [7] and convexity defect methods [8] are used in this identification system together with the Python [9] programming language and OpenCV plugins [10].

For accuracy smoothing and enhancement, the system employs an external webcam camera. Hand gestures taken by the webcam camera will be converted into RGB color [11] types so that they can be processed by OpenCV. After that, the background color changes to green. After that, the background color changes to green. From these results, the algorithms for convex hull and convexity defect are applied. By seeking out the fingertip point with the least amount of skin tone, the algorithm

* Coresponding author

locates the fingertips' points. Students' answers to the questions displayed are identified by the hand movements they make.

The factors examined in this study were website system success, accuracy, hand gesture recognition process time [12][13], and Quality of Service (QoS) [14]. The performance of transmission system could be seen by Quality of Service measurement, including delay, throughput, packet loss and jitter [14][15]. This measurement utilizes a wireshark software.

II. METHOD

A. *Block Diagram System*

Fig. 1 depicts a system overview that describes the hardware process of the system.

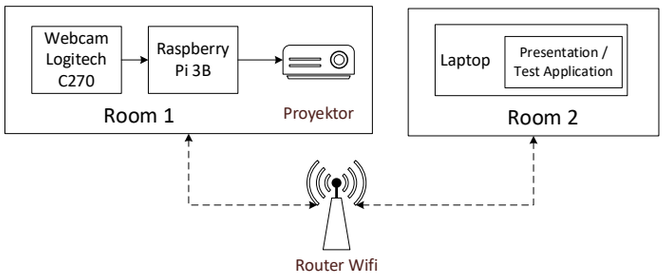

Figure 1. Block Diagram System

Hand gestures are performed by users who are in Room 1 and the webcam captures the movements and hand gestures demonstrated by the user. The program is a hand gesture detection system and is created through the Thonny IDE. In the program, there is a convex hull algorithm, and this program is tasked with computing the input video data into a collection of dots that form the hand. From these dots, the gesture and the number of fingers of the hand that will be used for the output process will be determined. The output of the number of fingers indicates the answer choice chosen by the user. The Raspberry Pi is a platform where video input data is converted into output data in the form of answer choices chosen by the user.

B. *System Planning*

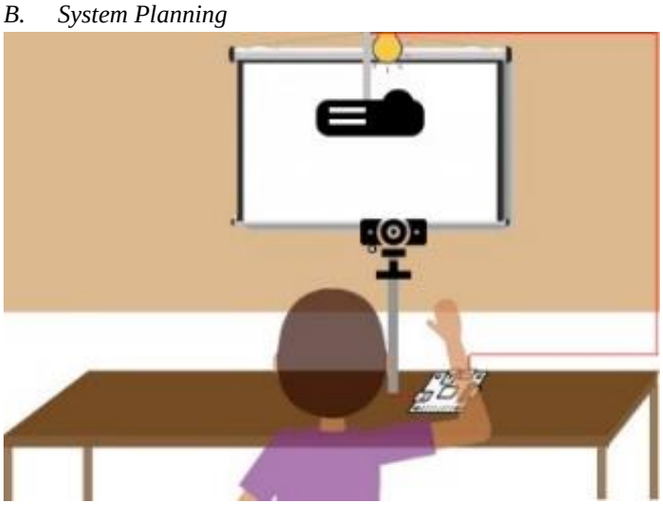

Figure 2. User's position from behind

In order to ensure that the convex hull only selected up the hand part, the camera capture was divided by a bulkhead. This sealing is carried out because the convex hull method can only differentiate between the outermost ends of the set of dots, making it unable to distinguish between hands and faces at this time. Inside the area, there is also another seal on the upper right and lower right sides. If there is one finger detected in the upper right area, then the presentation will proceed to the next question, while if there is one finger detected in the lower right area, then the presentation will proceed to the previous question. Fig. 2 and Fig. 3 shows how the system work.

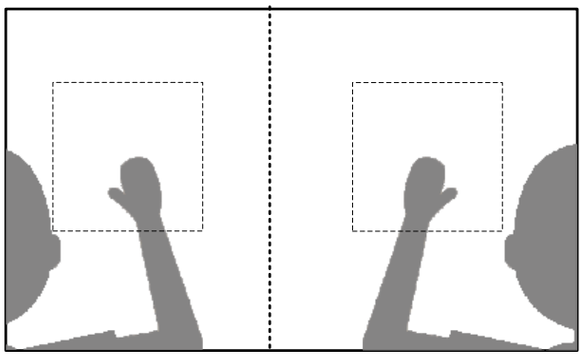

Figure 3. Webcam capture display design

C. *Flowchart Convex Hull Algorithm*

The convex hull of any point P is denoted by the notation CH(P). In simple terms, we can suppose that point P is a collection of nails stuck in a tree or wall and the convex hull is the smallest area like rubber made in a circle where every point P is inside the rubber area. Convex Hull is formed from lines connecting the points P which are in the outermost area. In the implementation, the convex hull has several methods that can be used, namely Jarvis Wrap. In simple terms the Jarvis Wrap method can be explained in the following steps:

- Take one of the points that is located on the outermost, for example, which has the highest or lowest x coordinate, and it could also be the one with the highest or lowest y-coordinate.
- Find the nearest line that makes the largest angle when drawn by the line from the first point
- The second step is done continuously until it returns to the starting point.

To accomplish the convex hull algorithm, theory convexity defects is needed. The use of the convex hull approach is thought to be less exact since one finger might occasionally be identified at both ends. To identify it accurately, we need an assistance algorithm. As a result, the convexity defect method is employed in this work. The convex hull technique is utilized to detect knuckles, which makes determining the number of elevated fingers more particular and exact. These convexity defects are the deepest concavities or deviations between two points convex hull. This theory calculates the angle generated between 2 fingers. Angles are calculated using the cosine Equation 1.

$$c = \sqrt{a^2 + b^2 - 2ab.\cos(\alpha)} \quad (1)$$

where a is length of finger 1, b is length of finger 2, c is distance between finger 1 and finger 2, d is the angle of finger 1 and finger 2. Thus, to calculate α use the following Equation 2.

$$\alpha = a\ cos\left(\frac{a^2+b^2-c^2}{2ab}\right) x \frac{180}{\pi} \quad (2)$$

If the angle α is less than 90 degrees, then α is a convexity defect. The number of convexity defects is the number of fingers raised minus 1. Figure 4 depicts the flowchart of Convex Hull Algorithm process.

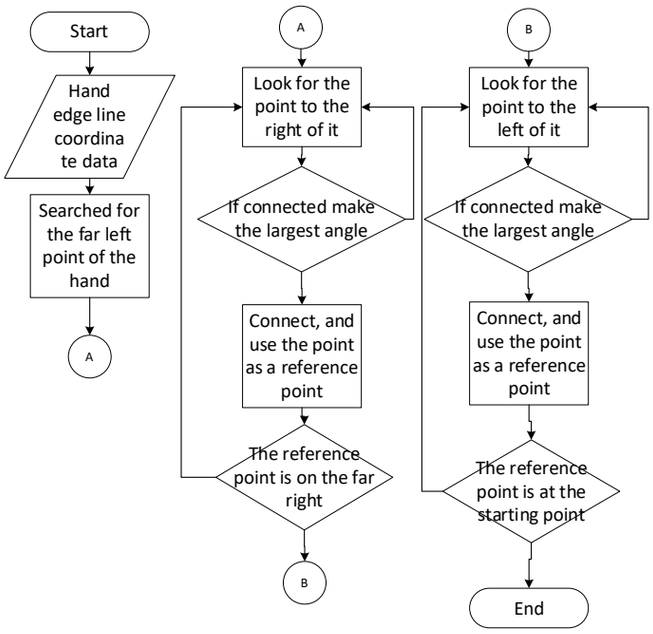

Figure 4. Convex Hull Algorithm

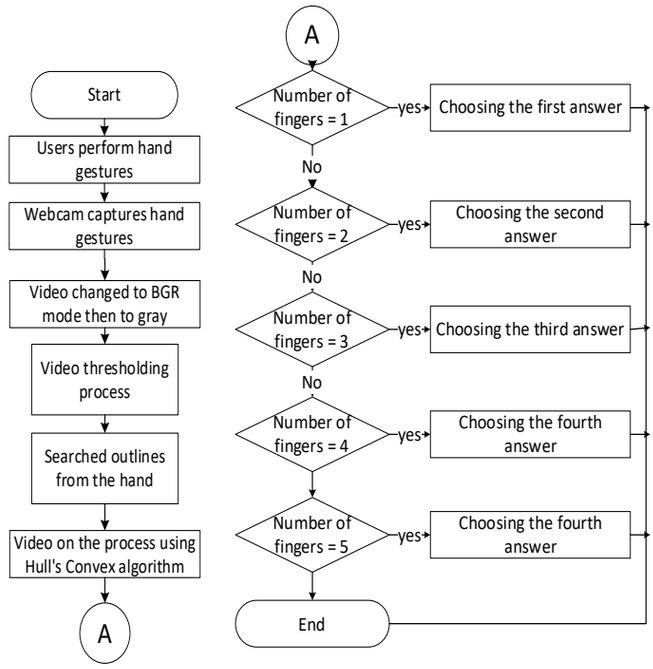

Figure 5. System Flowchart Diagram

The convex hull algorithm are: (1) The initial data needed is the border point data or hand contours scanned through the HSV color filter and commanded drawContour; (2) The data will be scanned using the convex Hull command; (3) the first thing to do is take the starting point on the far left of the object (the biggest X value); (4) Create a reference line, namely line x = x coordinate of the starting point. Find all points with the closest x coordinate and y coordinate above that point, connect the first point and all the points closest to that point, if the angle between the two lines these (reference lines and connection lines) form the largest angle, then the closest point that forms the line is the hull point. Then the reference point is changed to that point and the reference line is changed to the line that connects the starting point and the reference point, then the steps are repeated until the point is far right; (5) The same thing is done after the point is on the far right, looking for the closest point and measuring the largest angle between the 2 reference lines; (6) This is repeated until the reference point is at the starting point.

D. System Flowchart Diagram

Figure 5 illustrates the flow of system performance depicted in the form of a flowchart.

When the user makes a gesture with their hands, the camera will record it in real time. The webcam video will be automatically converted to BGR format (Blue, Green, and Red). After that, a convex hull algorithm will be used to process the video and calculate how many fingers there are based on the hand gesture that was recorded. The number of fingers on the hand gesture that has been captured will be identified, which will later be the output of the answer from the quiz. The table below lists the inputs and outputs for hand gestures, as depicted in Table I.

TABLE I
COMMAND AND HAND GESTURES

Command	Hand Gestures
Answer A	Identified 1 finger
Answer B	Identified 2 finger
Answer C	Identified 3 finger
Answer D	Identified 4 finger
Answer E	Identified 5 finger

E. Web Quiz System

Web Quiz is made by PHP and JavaScript language. The data is stored in PhpMyAdmin local database. This web is using radio button that assigned to keyboard press for check the radio button, as shown in Fig. 6 and Fig. 7.

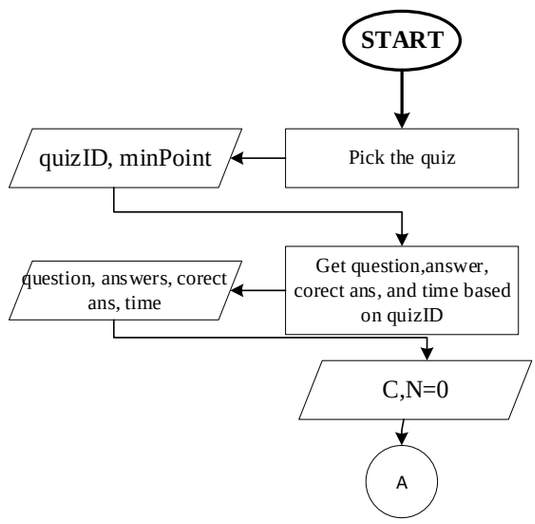

Figure 6. depicts the flowchart of web quiz system

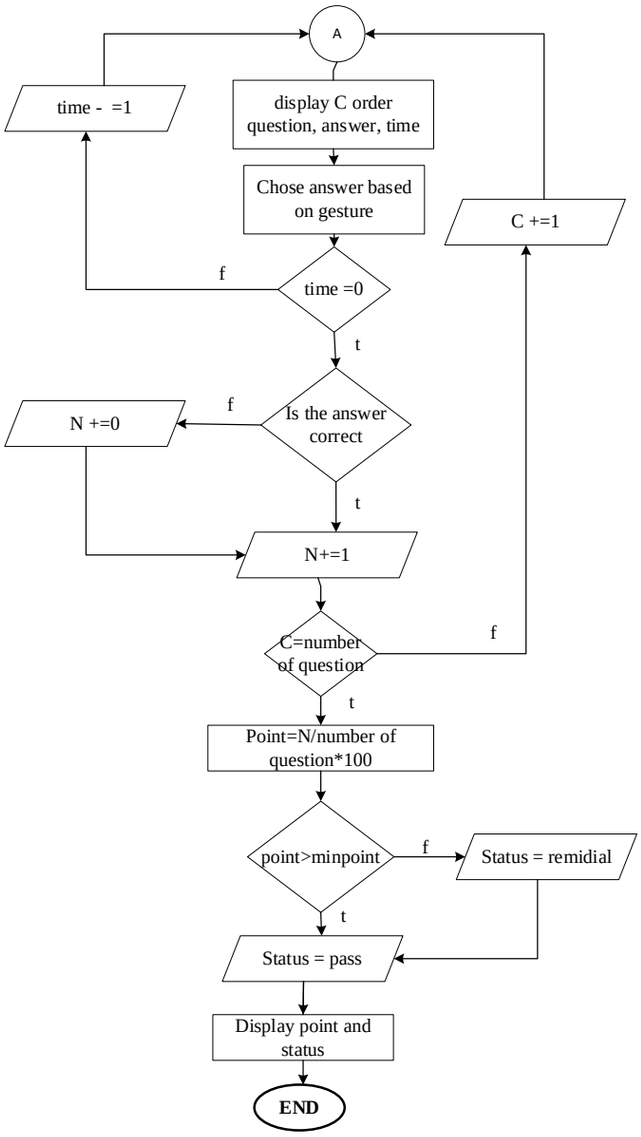

Figure 7. Web Quiz Flowchart

Fig 6 and 7 show flowchart of web quiz. The user has selected a quiz on the main page and received data in the form of a quiz and value limit. Retrieving all questions, answers 1-5, correct answers, and time according to the id quiz in the database so that you get data in the form of questions, answers 1-5, correct answers, and time. Then declare the variable C=0 (count), which is useful for the index of questions to be displayed, as well as N=0 (grades) for initial grades, and J=0 (answers) to find out the number of students who have answered a question. Displays the (C+1) question. Students choose answers according to the questions that appear.

If the answer chosen is in accordance with the correct answer, then N is increased by 1, but if the answer chosen is not appropriate then the value of J will be checked first. If the value of J ≠ 1 then the time will be reset, the value of J will increase by 1 and other students can answer again. If the value of J = 1 then it will proceed to the next question without adding value. On the other hand, if the value of C is not the same as the number of questions, then the value of C will increase by 1 and the process will return to displaying the questions and answers to (C + 1) and the value of J will be reset, but if the value of C is equal to the number of questions, then the process will continue to value calculation. Finally, value is calculated with the formula Value=N/number of questions*100 and presented on display.

III. RESULTS AND DISCUSSION

A. *Callibration of Sensors*

Light Intensity Sensor (BH1750) is used in this research. Testing of the light intensity sensor was carried out using two measuring instruments, the Digital Lux Meter AS803 and the BH1750 sensor. This test was carried out in 2 conditions, room conditions and conditions when exposed to direct light. The results of the calculation of light intensity will be displayed on the Thonny terminal. In the BH1750 datasheet it is written that the intensity that can be measured is 0.11 lux to 100000 lux. The average measurement error stated on the datasheet is ± 20%. The results of testing the light intensity sensor error rate can be calculated using the following formula:

$$Error\ (\%) = \frac{Lux\ Meter\ AS803\ Value - BH1750\ Value}{Lux\ Meter\ AS803\ Value} \times 100\%$$

The error percentage calculation is performed between the values obtained from the BH1750 sensor and the AS803 luxmeter, which are presented in the Table II.

TABLE II
MEASUREMENT RESULTS OF THE BH1750 SENSOR AND AS803 LUXMETER

No	BH1750 (lx)	AS803 (lx)	Deviation (lx)	Error (%)
1	30.83	31	0.17	0.6%
2	30	31	1	3.2%
3	30	30	0	0%
4	30	30	0	0%
5	30	30	0	0%
6	30	31	1	3.2%
7	30	30	0	0%
8	30	31	1	3.2%
9	275	278	3	1.07%
10	274.17	278	3.83	1.3%
11	270.83	280	9.17	3.2%
12	271.67	277	5.33	1.92%
13	271.67	277	5.33	1.92%
14	271.67	278	6.33	2.27%
15	273.33	278	4.67	1.68%
16	273.33	277	3.67	1.32%
Average			2.78	1.56%

Based on Table II, the average value of the difference is 2.78 lux, and the average error value is 1.56%. According to the datasheet of the BH1750 sensor, this sensor has a tolerance value of ±20%. Therefore, it can be said that the BH1750 sensor used is accurate and can be used because it does not exceed the tolerance value.

B. *Hardware Implementation Result*

There are few hardware components in this system. In the student side there are Raspberry pi, 2 webcam, BH1750 light sensor, and monitor to display raspberry pi. In the lecturer side just only a laptop. This laptop is the source of the web quiz, and the raspberry pi is just a tool to recognize the hand gesture that student do in the other room, as depicted in Fig. 8 and Fig. 9.

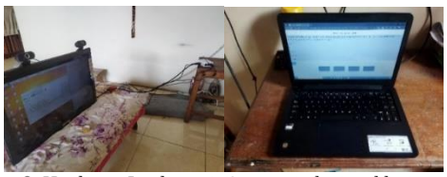

Figure 8. Hardware Implementation on student and lecturer side

Figure 9. Display when webquiz and hand gesture recognition running

C. *Software Implementation Result*

This software is made by PHP and JavaScript language. PhpMyAdmin database is the database that used in this web quiz. The information below provides documentation of the software in the form of webpage based on quiz. This webpage was created using visual studio code software, and this web is installed only on lecturer laptop.

In testing the login page will be carried out by opening the login page and filling in the username and password. If the registered username and password match then it will change to the main page. If the username or password is not registered, a warning will appear, as shown in Fig. 10.

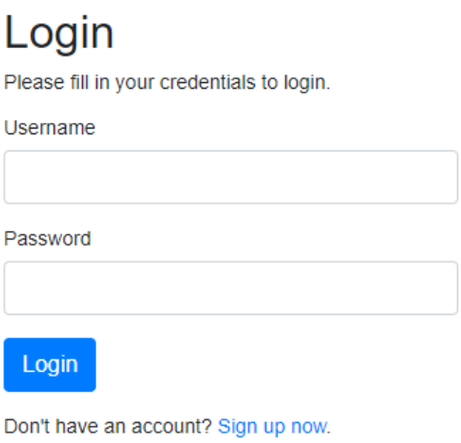

Figure 10. Login page

On the main page there are 4 components that can be pressed, namely: choice of quiz title, add quiz schedule, score recap, and sign out. The homepage is depicted in Fig. 11.

Figure 11. Homepage

A pop-up page will appear, and the title, class, and quiz id input fields will be filled in automatically according to the selected quiz, as presented in Fig. 12.

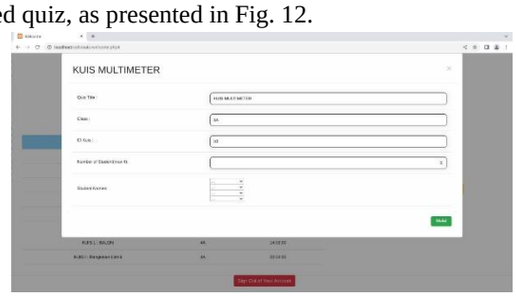

Figure 12. Pop Up page to set the number of students to take the quiz

If the answer is correct then the box will change color to green and it will say "Answer (name of student) is correct". If the answer is wrong then the box will change color to red and it will say "Incorrect answer (name of student), as shown in Fig. 13.

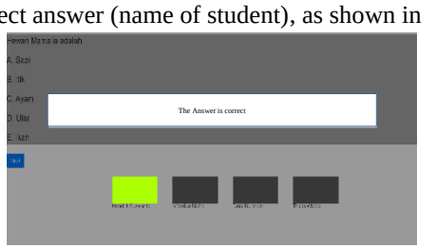

Figure 13. Quiz Page

Displays the initial value recap page which contains a select class column. The value of each student will be displayed according to the order of the highest score, as highlighted in Fig. 14.

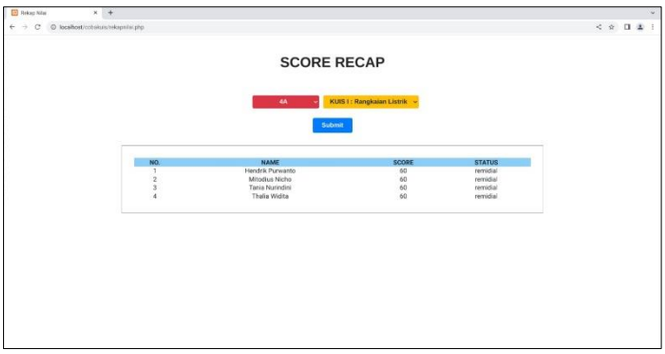

Figure 14. Point Recap

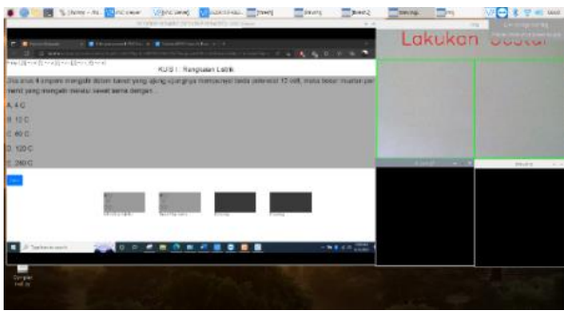

Figure 15. Hand Gesture Recognition

D. *System Testing for Hand Gesture Recognition*

This hand gesture recognition accuracy test is conducted at 3 various distances (30 cm, 50cm, and 80cm) and 3 various light intensity (<50 lux, 50-100 lux, and >100lux). The accuracy calculation will be calculated by the success and failure of hand

gesture identification. The table III shows the result of the recognition

TABLE III
EXAMPLE RESULT OF HAND GESTURE RECOGNITION

Image	Description
	1 finger up
	2 finger up
	3 finger up
	4 finger up
	5 finger up

The calculation of the accuracy of hand gestures is done after getting results at each distance of 30 cm, 50 cm, and 80 cm with three different light intensities, namely <50 lux, 50-100 lux, and >100 lux. The Table IV presents the results of calculating the accuracy.

TABLE IV
HAND GESTURE RECOGNITION ACCURACY RESULT

No.	Distance	Light Intensity (Lux)	Accuracy (%)	Avg accuracy (%)
1.	30 cm	<50	88%	96%
		50 -100	100%	
		>100	100%	
2.	50 cm	<50	92%	97%
		50 -100	100%	
		>100	100%	
3.	80 cm	<50	88%	96%
		50 -100	100%	
		>100	100%	

After the testing, it is known that the accuracy percentage of the <50 lux condition is below the other light intensity. And we can see, the distance does not affect the accuracy of this hand gesture recognition. This test gets 100% accuracy on every distance when the light intensity is above 50 lux. But if the light intensity below 50 lux, the accuracy become lower, 88%-92% in every distance.

E. Delay testing for hand gesture recognition

This hand gesture recognition accuracy test is conducted at 3 various distances (30 cm, 50cm, and 80cm) and 3 various light intensity (<50 lux, 50-100 lux, and >100lux). The delay calculation will be calculated by time that raspberry pi gets the gesture recognition, minus by time that raspberry pi gets the frame from webcam. The Table V depicts the results of calculating the delay.

TABLE V
DELAY TEST RESULT

No.	Light Intensity	Distance	Average Delay per Light Intensity and Distance	Average Delay per Light Intensity
1	<50	30	0.0273	0.0280
		50	0.0293	
		80	0.0275	
2	50-100	30	0.003	0.0032
		50	0.0036	
		80	0.003	
3	>100	30	0.0033	0.004
		50	0.003	
		80	0.0056	

From these results, it can be seen that the highest average delay is found in hand gesture testing at light intensities below 50 lux. This is due to the lack of light, so that the raspberry pi takes longer to carry out the identification process.

F. Quiz Web Test Using Hand Gestures Results

This test aims to find out whether the answers on the quiz webpage can change according to the hand gesture input performed. This test is carried out after the Raspberry Pi has been connected to the laptop and the laptop is in the state of opening the web page and answering questions, as highlighted in Fig. 16, Fig. 17, Fig. 18 and Fig. 19.

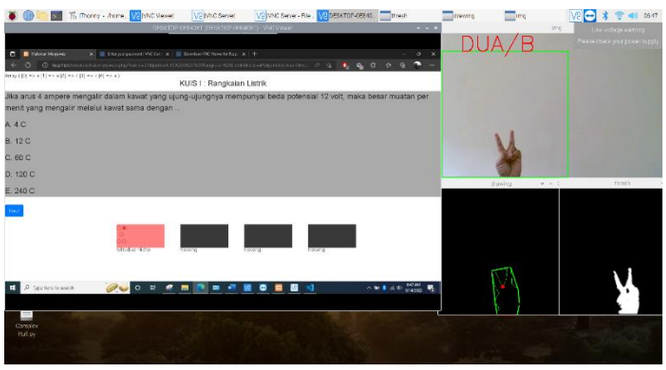

Figure 16. Hand Gesture Recognition on 1 student

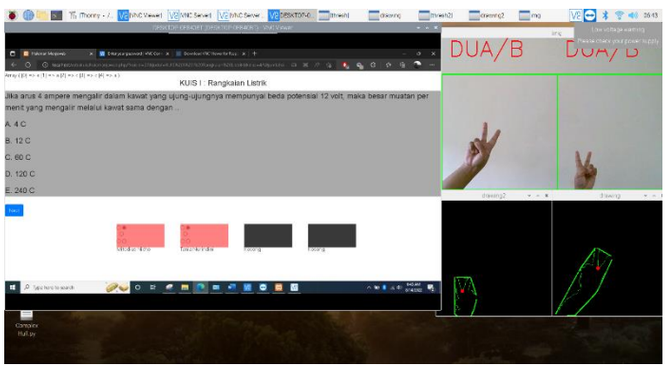

Figure 17. Hand Gesture Recognition on 2 students

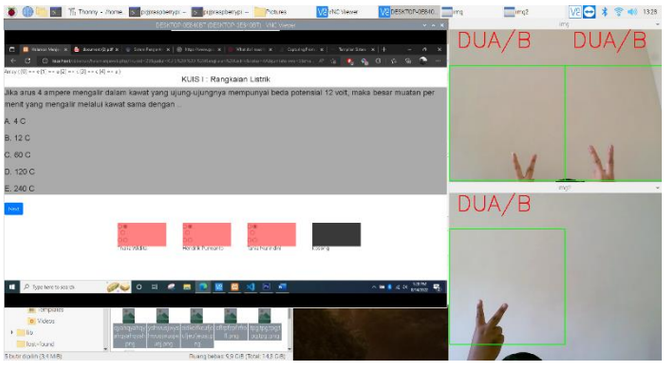

Figure 18. Hand Gesture Recognition on 3 students

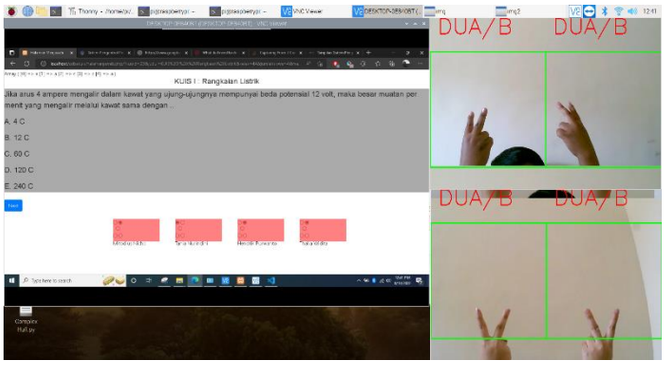

Figure 19. Hand Gesture Recognition on 4 students

After testing, it can be seen that the overall quiz web system is able to recognize answers based on hand gestures.

G. *Quality of Service (QoS) Analysis*

This QoS test is conducted in order to assess the network quality utilized by laptops and Raspberry Pi. Throughput, Packet Loss, and Delay are the variables that were examined in this study. The wireshark program was used to conduct this test on the laptop side.

1) Throughput Measurement: Throughput is the total number of packets that have successfully arrived during a certain period of time divided by the duration of that period.

$$Throughput = \frac{\Sigma delivered\ packet}{lenght\ of\ measurement}$$

$$Throughput = \frac{22479930}{2883.350} = 7.796\ KBps = 62.371\ kbps$$

Based on Wireshark application, throughput measurement can be shown in Fig. 20

Measurement	Captured	Displayed	Marked
Packets	55221	55221 (100.0%)	—
Time span, s	2883.350	2883.350	—
Average pps	19.2	19.2	—
Average packet size, B	407	407	—
Bytes	22479930	22479930 (100.0%)	0
Average bytes/s	7796	7796	—
Average bits/s	62 k	62 k	—

Figure 20. The result of throughput measurement

From the throughput results obtained, it can be seen that the throughput values obtained are included in the very good category (> 100bps), according to the TIPHON standard.

2) Packet Loss Measurement: The number of packets lost compared to packets sent.

$$Packet\ Loss = \frac{\Sigma \text{sent packet} - \Sigma delivered\ packet}{\Sigma \text{sent packet}}$$

$$Packet\ Loss = \frac{55221 - 55221}{55221} x100\% = 0\%$$

Based on Wireshark application, packet loss measurement can be shown in Fig. 21

Measurement	Captured	Displayed	Marked
Packets	55221	55221 (100.0%)	—
Time span, s	2883.350	2883.350	—
Average pps	19.2	19.2	—
Average packet size, B	407	407	—
Bytes	22479930	22479930 (100.0%)	0
Average bytes/s	7796	7796	—
Average bits/s	62 k	62 k	—

Figure 21. The result of packet loss measurement

The packet loss test results are 0%, which means there are no failed packets. The value of 0% packet loss is included in the very good category (<3%), according to TIPHON.

3) Delay Measurement: Delay is the amount of time it takes for a packet to travel from origin to destination. Delay is obtained from the division between Time Span and Packets. Delay is affected by distance, physical media and congestion.

$$Delay = \frac{\text{timespan}}{\text{packets}}$$

$$Delay = \frac{2883.35}{55221} = 0.0522s = 52.2\ ms$$

Based on Wireshark application, packet loss measurement can be shown in Fig. 22

Measurement	Captured	Displayed	Marked
Packets	55221	55221 (100.0%)	—
Time span, s	2883.350	2883.350	—
Average pps	19.2	19.2	—
Average packet size, B	407	407	—
Bytes	22479930	22479930 (100.0%)	0
Average bytes/s	7796	7796	—
Average bits/s	62 k	62 k	—

Figure 22. Delay measurement

The delay obtained is 52.2 ms, this value is included in the very category good (<150ms), according to the TIPHON standard.

IV. CONCLUSION

Following the design, implementation, and testing phases, it can be concluded. The system is made up of several hardware and software components. The Raspberry Pi, a webcam, a light intensity sensor, and a laptop make up the hardware. The software is a web-based application that was created using the PHP. Through testing, a total accuracy of 96.5% was obtained based on 2 parameter such as distance and light density. In that test, can be concluded that brighter light and closer distance will make the accuracy higher. In this research, a third-party program called VNC was used to connect the quiz web system (a laptop) and Raspberry Pi It was discovered through testing in chapter 4 that the Raspberry Pi can provide keyboard commands in response to movements made. The keyboard commands that are then delivered are also consistent with the chosen answer option. Based on QoS measurement results, the throughput result is 62 kbps which is included in the very good category. Packet Loss of

0% which is also included in the very good category. Delay of 52.2 ms which is included in the very good category. Thus, the overall quality of the network can be categorized as very good.

REFERENCES

[1] M.H.Khoirul, "Sistem Pengontrol Presentasi Menggunakan Pengenalan Gestur Tangan Berbasis Fitur pada Contour Dengan Metode Klasifikasi Support Vector Machine," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 4, no. 4, pp. 1083-1089, 2020.

[2] D. R. M. Harika, "Rancang Bangun Pengontrol Presentasi Berbasis Slide dengan Teknik Analisis Gerakan Jari dan Tangan," *JOIN (Jurnal Online Informatika)*, vol. 1, no. 2, 2016.

[3] M. Anshary, "Prototype Program Hand Gesture Recognize Using the Convex Hull Method and Convexity Defect on Android," *JOIN (Jurnal Online Informatika)*, vol. 5, no. 2, pp. 205 - 2011, 2020.

[4] A.R.Adnan, "Klasifikasi Gestur Lengan Manusia Menggunakan Metode KNN Untuk Kendali Stop Kontak Pintar Berbasis Internet of Things," *e-Proceeding of Engieering Telkom University*, vol. 8, no. 1, pp. 9-16, 2021.

[5] I. C. H.A. Adi, "Sistem Pengenal Isyarat Tangan Untuk Mengendalikan," *Indonesian Journal of Electronics and Instrumentation Systems (IJEIS)*, vol. 9, no. 2, pp. 193-202, 2019.

[6] S. Qin, "Real-time Hand Gesture Recognition from Depth Images," *J Sign Process Syst*, 2018.

[7] A. Tompunu, "FINGER TRACKING AND RECOGNITION USING OPENCV RASPBERRY PI 3," *Proceeding Forum in Research, Science, and Technology (FIRST)*, 2017.

[8] Wilkinson, "A Raspberry Pi-based camera system and image processing procedure for low cost and long-term monitoring of forest canopy dynamics.," *Methods Ecol*, vol. 12, pp. 1316-1322, 2021.

[9] J. Minichino, "Learning OpenCV 3 Computer Vision with Python Second Edition," in *Preface*, Brimingham, UK, Packt Publishing, 2015, p. vii.

[10] K. &. Atul, "The AI Learner," 9 November 2020. [Online]. Available: https://theailearner.com/2020/11/09/convexity-defects-opencv/. [Accessed 12 Agustus 2022].

[11] Bhowmik, Interactive Display : Natural Human - Interface Technologies, John Wiley & Sons, 2015.

[12] A. Skuric, V. Padois, N. Rezzoug and D. Daney, "On-Line Feasible Wrench Polytope Evaluation Based on Human Musculoskeletal Models: An Iterative Convex Hull Method," in IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 5206-5213, April 2022, doi: 10.1109/LRA.2022.3155374.

[13] S. -J. Horng, D. -T. Vu, T. -V. Nguyen, W. Zhou and C. -T. Lin, "Recognizing Palm Vein in Smartphones Using RGB Images," in IEEE Transactions on Industrial Informatics, vol. 18, no. 9, pp. 5992-6002, Sept. 2022, doi: 10.1109/TII.2021.3134016.

[14] P. E. Mas'udia, C. A. Pratama, D. Purwati, Y. Ratnawati, M. Sarosa, and N. Hidayati, "Rancang Bangun dan Analisis QoS pada Sistem Informasi Penjualan Obat dengan Layanan Antar-Jemput Berbasis Android," Techno.Com, vol. 21, no. 3, pp. 633–643, 2022, doi: 10.33633/tc.v21i3.6209.

[15] D. Priadi, "Pengukuran Quality of Service (QoS) Pada Aplikasi File Sharing dengan Metode Client Server Berbasis Android", Jartel, vol. 6, no. 1, pp. 39-49, May 2018.