ANTIVIRUS: Jurnal Ilmiah Teknik Informatika (p – ISSN: 1978 – 5232; e – ISSN: 2527 – 337X)
Vol. 16 No. 2 November 2022, pp. 223 – 232

PENGENALAN GESTUR GERAKAN JARI UNTUK MENGONTROL VOLUME DI KOMPUTER MENGGUNAKAN *LIBRARY* OPENCV DAN MEDIAPIPE

Saiful Nur Budiman[1], Sri Lestanti[2)], Suji Marselius Evvandri [3)], Rahma Kartika Putri [4)]
[1, 2, 3, 4)] Program Studi Teknik Informatika, Fakultas Teknologi Informasi dan Universitas Islam Balitar
[1, 2, 3, 4)] Jalan Majapahit No. 4, Kec. Sananwetan, Kota Blitar, Jawa Timur, Indonesia, kode pos: 66131
e-mail: sync.saifulnb@gmail.com [1)], lestanti85@gmail.com [2)], marseloky268@gmail.com [3)] ,
rahmakartika22@gmail.com [4)]

Abstrak **:** *Gesture recognition merupakan salah satu bagian yang ada dalam kecerdasan buatan dibidang computer vision. Dengan gesture recognition tersebut komputer mampu memahami gerakan yang tertangkap pada kamera/webcam. Manfaat dari gesture recognition ini banyak sekali, salah satunya yang sedang peneliti lakukan mengenai handtracking gesture recognition jari tangan kanan manusia untuk mengantur kontrol volume pada komputer atau laptop. Berdasarkan latar belakang tersebut, penelitian ini dimaksudkan untuk menerapkan machine learning yang dikembangkan dari library OpenCV dan MediaPipe untuk melakukan proses pelatihan dan testing gesture jari tangan sebagai isyarat untuk mengontrol salah satu fungsi yang ada di Windows, salah satunya adalah mengontrol volume. Proses ini menggunakan Library OpenCV dan MediaPipe karena mampu melakukan multiprocessing dengan data yang real-time, sehingga proses identifikasi gesture lebih cepat dan akurat. Ketika kamera/webcam menangkap frame pergerakan gesture jari tangan kanan manusia, maka dilakukan proses augmentasi dan pemberian landmark keypoint localization dari setiap ruas jari. Pada penelitian ini, jari yang dikenali hanya landmark ujung jempol dan landmark ujung jari telunjuk. Machine learning akan melakukan perhitungan dari jarak ujung jempol dengan ujung telunjukn yang mana digunakan untuk menentukan perubahan besaran volume suara. Dari hasil pengujian sebanyak sembilan kali uji coba dengan pose-pose jari berbeda diperoleh 88,89%. Satu dari hasil pengujian tidak berhasil membaca gesture gerakan jari, dikarenakan posisi landmark ujung jari telunjuk yang tertutup dengan jari lainnya.*

Kata Kunci – MediaPipe, Gesture Recognition, Machine Learning

Abstract **:** *Gesture recognition is a part of artificial intelligence in the field of computer vision. With gesture recognition, the computer is able to understand the movements captured on the camera/webcam. The benefits of gesture recognition are many, one of which is what researchers are doing regarding hand-tracking gesture recognition of the human right-hand finger to adjust the volume control on a computer or laptop. Based on this background, this research is intended to apply machine learning developed from the OpenCV and MediaPipe libraries to carry out the process of training and testing finger gestures as gestures to control one of the functions in Windows, one of which is volume control. This process uses the OpenCV Library and MediaPipe because they are capable of multiprocessing with real-time data, so the gesture identification process is faster and more accurate. When the camera/webcam captures the frame of the movement of the human's right-hand finger gesture, an augmentation process is carried out and the provision of keypoint localization landmarks is carried out for each knuckle. In this study, only the fingertip landmarks and index finger landmarks were recognized. Machine learning will perform calculations from the distance between the tip of the thumb and the tip of the forefinger which is used to determine changes in the volume of the sound. From the test results of nine trials with different finger poses, 88.89% was obtained. One of the test results failed to read finger movement gestures, due to the landmark position of the tip of the index finger which was closed with the other fingers.*

Kata Kunci – MediaPipe, Gesture Recognition, Machine Learning

I. PENDAHULUAN

MEDIAPIPE merupakan sebuah *framework* yang dirancang dengan cara mengimplementasikan kecerdasan buatan kedalam aplikasi yang akan dibangun. MediaPipe tersebut digunakan oleh Google dalam menentukan karakteristik yang ada dalam video. Sebagai contohnya untuk menentukan *thumbnail* video atau mengidentifikasi *copyright* video. Selain itu, bisa digunakan untuk menentukan *object* video yang tidak diizinkan seperti hal-hal yang berbau porno, kekerasan dan iklan. Hal yang dilakukan oleh Google dengan mengumpulkan semua dataset kemudian memberikan label serta mengidentifikasi setiap *landmark*-nya secara manual.

DOI: https://doi.org/10.35457/antivirus.v16i2.2508

Kecerdasan buatan pada pengaplikasiannya secara garis besar terbagi tujuh cabang, yaitu *machine learning, natural language processing, expert system, vision, speech, planning* dan *robotics*. Cabang dari kecerdasan buatan tersebut dimaksudkan untuk ruang lingkup saat pengembangan atau belajar *artificial intelligence* karena pada dasarnya kecerdasan buatan memiliki ruang lingkup yang sangat luas dan beragam [1]. Salah satu hal yang menarik penulis untuk melakukan penelitian terhadap kecerdasan buatan ini adalah dibidang vision yaitu *hand gesture recognition*. Hal yang bisa dilakukan dengan *hand gesture* antara lain adalah berinteraksi dengan komputer, menggerakan mouse ataupun keyboard. Proses pengenalan *hand gesture* secara *real-time* masih banyak mengalami kendala di beberapa bagian antara lain proses pelacakan posisi tangan dan metode pengenalan konvensional dimana tingkat akurasi dari pengenalannya belum bisa optimal. Untuk mengatasi hal tersebut, proses *hand gesture* dikembangkan melalui tiga tahap yakni akusisi, pelatihan dan identifikasi. Proses akuisisi *hand gesture* terdiri dari segmentasi, dilasi, erosi, *convex hull* dan *convexity defect* untuk mendapatkan bentuk dan posisi tangan yang dilakukan secara terus menerus [2]. Kumpulan posisi tangan kemudian diproses dan dinormalisasi menjadi citra yang kemudian digunakan dalam proses pelatihan dan identifikasi.

Salah satu cabang kecerdasan buatan yang menangani masalah citra adalah *computer vision*. Dengan *computer vision* citra akan ditangkap oleh kamera/webcam secara analog kemudian dikonversi ke digital untuk diolah menggunakan komputer. Pengolahan citra ini bertujuan untuk mengidentifikasi citra atau melakukan perbaikan kualitas citra. Untuk mempermudah pengolahan citra, Intel membuat *library open source* yakni OpenCV [3]. Dengan memakai OpenCV maka pendeteksian objek pada citra yang diidentifikasi ataupun diperbaiki kualitas citranya bisa dilakukan secara cepat dan *real-time*. Jika Intel memiliki OpenCV maka Google memiliki MediaPipe. Kelebihan dari MediaPipe bisa menggunakan data t*ime series* dan menggunakan *pipeline configuration* sehingga bisa dilakukan *multiprocessing* secara paralel [4].

Berdasarkan latar belakang kecerdasan buatan pada *computer vision, hand recognition*, OpenCV dan MediaPipe maka penulis bertujuan untuk mengenali pola *gesture* gerakan jari tangan kanan manusia, khususnya jari jempol dan jari telunjuk. Gerakan *gesture* yang dimaksudkan disini digunakan untuk mengontrol *volume* media suara pada laptop atau komputer secara *real-time*. OpenCV dan MediaPipe dengan bahasa python digunakan sebagai library pengenalan pola *landmark* model jari-jari tangan kanan manusia yang terdeteksi pada kamera/webcam secara *real-time*.

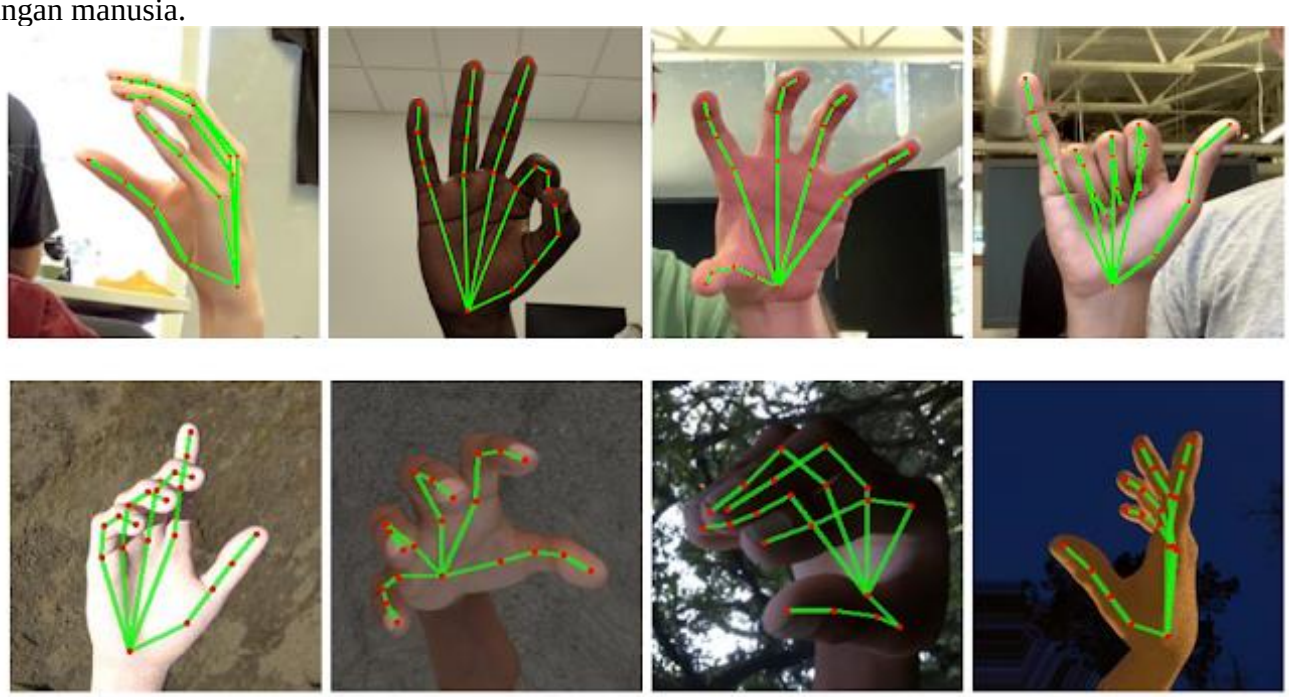
II. TINJAUAN PUSTAKA

OpenCV (*Open Computer Vision*) merupakan API yang bersifat open source yang bisa digunakan untuk *image processing*. Pada tahun 1999 OpenCV dikembangkan oleh Garry Bradsky dan baru dirilis tahun 2020. Beberapa algoritma didukung oleh OpenCV untuk jenis penelitian yang behubungan dengan *computer vision* dan *machine learning*. Selain itu OpenCV juga bisa dikembangkan dengan bahasa pemrograman C++, Python, Java dan lain sebagainya [5]. Tidak hanya itu, OpenCV juga tersedia dalam berbagai platform, seperti Windows, Linux, OSX, Android, IOS, dan lain sebagainya. OpenCV juga dapat melakukan operasi high-speed GPU dengan menggunakan *interface* berbasis CUDA, CuDnn serta OpenCL. Kombinasi terbaik untuk dapat melakukan operasi berkecepatan tinggi tersebut adalah dengan perpaduan antara OpenCV, C++ API dan bahasa pemrograman Python [6].

MediaPipe juga merupakan *framework* yang ditulis menggunakan bahasa C++ dan bisa digunakan untuk mengembangkan aplikasi *cross-platform*, seperti windows, linux, macintosh, android bahkan website. Dengan menggunakan *multithreading* dan GPU *acceleration*, membuat MediaPipe sangat cepat dalam memproses banyak data yang masuk. MediaPipe mempunyai *calculator* yang akan berjalan ketika program dimulai dan akan berhenti ketika program selesai. *Calculator* tersebut digunakan MediaPipe untuk konsep *multithreading*. MediaPipe menggunakan *machine learning* secara *real-time* (*ML solution for live and streaming media*). *Machine learning* yang ditawarkan oleh MediaPipe antaralain *Face Detection, Face Mesh, Iris, Hands, Pose, Holistic, Hair Segmentation, Object Detection, Box Tracking, Instant Motion Tracking, Objectron, KNIFT* [7]. Dari banyaknya solusi *machine learning* yang ditawarkan MediaPipe, maka penelitian ini memfokuskan penggunaan OpenCV dan MediaPipe untuk mengenali tangan kanan manusia dan melakukan segmentasi khususnya untuk jari jempol dan jari telunjuk saja. Hal yang dilakukan pertama kali setelah tangan terdeteksi oleh kamera/webcam, maka komputer akan memberikan landmark pada tangan. Proses pembentukan landmark ini didasarkan dari

DOI: https://doi.org/10.35457/antivirus.v16i2.2508

model yang dibuat oleh *machine learning* MediaPipe dan OpenCV. Contoh dari *landmark-landmark* model tangan yang ditangkap kamera/webcam menggunakan MediaPipe dan OpenCV dapat dilihat pada Gambar 1. Dengan mengetahui *landmark* tersebut, maka bisa dilakukan pengenalan *gesture* gerakan jari tangan manusia.



Gambar 1. Hand Landmark [7]

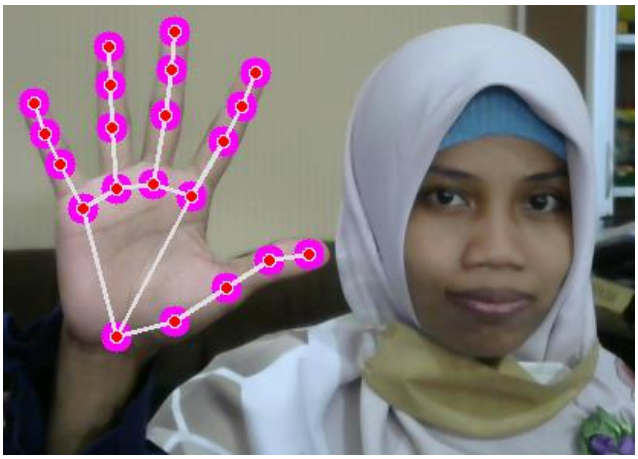
Hand gesture recognition sendiri merupakan teknologi yang mampu membaca gerak tangan kemudian dirubah menjadi teks, suara ataupun perintah. *Gesture recognition* merupakan topik dalam *computer science* dan *language technology* bertujuan agar komputer bisa memahami gerakan manusia pada umumnya melalui *gesture* tangan ataupun *gesture* wajah [8].

Terdapat dua macam teknologi yang bisa mengenali *gesture* antara lain *vision based* dan *glove based*. Metode penggunaan *vision based* membutuhkan kamera/webcam untuk mengambil gambar yang akan diproses. *Vision based* termasuk metode yang sederhana namun memiliki tantangan seperti faktor pencahayaan, warna objek yang dideteksi dengan warna di sekitar dan pendeteksian *background*. Selain itu dibutuhkan kecepatan, ketepatan serta efisiensi sistem pendeteksian. Contoh hasil pendeteksian menggunakan *vision-based hand recognition* dapat diamati pada Gambar 2. Contoh lain dari *vision based* adalah pengenalan ekspresi wajah menggunakan DCT dan LDA untuk memutar musik [9]. Setiap citra wajah diklasifikasikan berdasarkan ekspresinya, seperti ekpresi netral, senang, sedih, kaget, jijik, takut, marah [6]. Metode pengenalan *gesture* lainnya dengan menggunakan *glove based* (sarung tangan) membutuhkan sensor untuk menangkap koordinat tangan dan perpindahannya. Pengguna diharuskan menggunakan peralatan tambahan. Keuntungan menggunakan metode ini dapat mempermudah penerimaan informasi seperti koordinat telapak tangan, jari, dan orientasi. Kekurangan metode ini memerlukan koneksi langsung antara pengguna dengan sistem, serta biaya pembuatan alatnya cukup mahal. Sehingga pada penelitian ini, penulis hanya menggunakan *vision based* untuk melakukan pengenalan *gesture* tangan kanan khususnya jari melalui *hand tracking recognition* untuk mengontrol volume suara pada laptop atau komputer.

Pada dasarnya untuk mendeteksi pergerakan citra pada *vision based* direkam dengan cara *frame by frame* yang membentuk pergerak dinamis [10]. Setiap perubahan yang terjadi pada citra *frame by frame* tersebut menjadi tantangan untuk dapat mengimplementasikan algoritma yang tepat untuk situasi yang *real-time*. Solusi pelacakan tangan atau *handtracking* terdiri dari dua model yang bekerja secara bergantungan satu sama lain [11] yakni deteksi telapak model (*palm*) dan *model landmark* (*keypoint localization*). Model deteksi telapak memberikan gambar telapak tangan yang terpotong secara akurat. Selanjutnya hasil *croping* ini diteruskan ke *model landmark*. Proses tersebut dapat mengurangi augmentasi data rotasi, *flipping*, *scaling* yang dilakukan dalam *model deep learning*. Augmentasi citra dilakukan untuk mengatasi masalah pengenalan pola telapak tangan dan jari tangan. Sebagai contoh

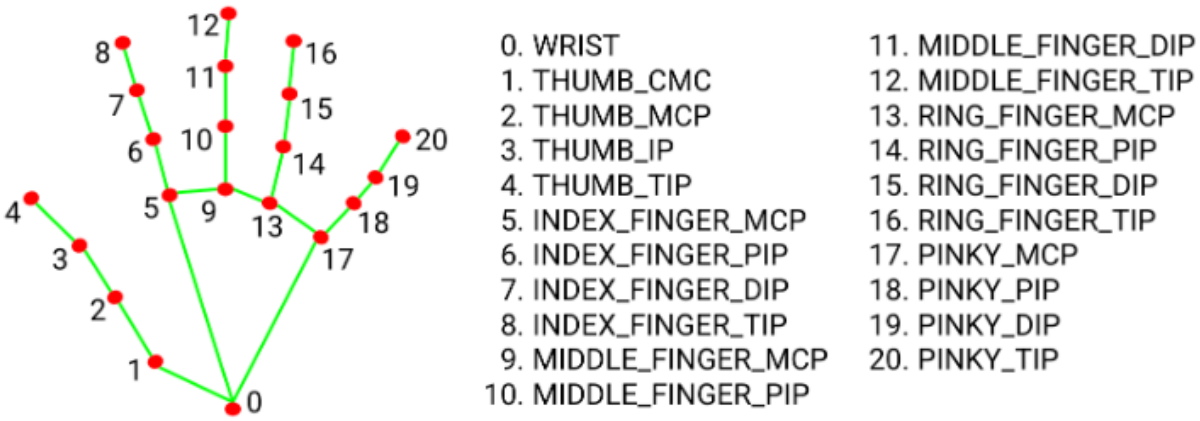
DOI: https://doi.org/10.35457/antivirus.v16i2.2508

model yang dibuat dari *machine learning* adalah data gambar tangan manusia secara utuh nyata terlihat jelas dan normal. Pada saat testing atau pengujian, tangan kanan manusia tersebut berubah-ubah bentuknya dan tertangkap oleh kamera/webcam maka *machine learning* tersebut haruslah bisa mengenali pola tersebut merupakan pola yang sama dengan pola tangan yang dilatih sebelumnya. Ini semua bisa dilakukan dengan cara mengkombinasikan OpenCV dan MediaPipe untuk membuat *machine learning* yang handal. Pada gambar 2, dilakukan pengetesan hasil pergerakan tangan manusia. Disaat kamera/webcam menangkap pergerakan tangan, maka dengan mudah *machine learning* yang telah dilatih bisa mengetahui lokasi-lokasi *landmark keypoint localization* pada tangan tersebut.



Gambar 2. Hasil Deteksi Dengan *Vision Based*

Berdasarkan hasil penelitian dan pengembangan yang dilakukan oleh tim Google terhadap library mereka yaitu MediaPipe maka *landmark* keypoint localization-nya dibagai menjadi 21 poin seperti yang ditunjukkan pada Gambar 3. Untuk penelitian ini landmark yang berasal dari ujung jari jempol (keypoint 2) dan ujung dari telunjuk (keypoint 8). Secara logika untuk mengecilkan volume media suara, maka pergerakan *gesture* jempol dan jari telunjutk dirapatkan seperti gerakan mencubit. Untuk memperbesar volume media suara berarti dilakukan sebaliknya yaitu dengan melebarkan pegerakan antaran jempol dengan jari telunjuk.



Gambar 3. Finger Hand Landmarks [7]

III. METODE PENELITIAN

Pada tahap ini penulis melakukan pengumpulan dengan cara sebagai berikut:

A. Pengumpulan Dataset

Penulis mengumpulkan dataset dari berbagai kondisi tangan kanan responden dengan pose yang berbeda-beda. Model dari tiap tangan kanan akan direkam menggunakan kamera/webcam. Nantinya data ini digunakan untuk membuat model pada *machine learning*. Sebagai tahap testing akan dilakukan perhitungan jarak *landmark point* dari ujung jempol dan ujung jari telunjuk. Jarak tersebut kemudian

DOI: https://doi.org/10.35457/antivirus.v16i2.2508

diinterpolasikan ke besaran skala suara, semakain besar hasil skalanya maka volume suaranya akan dimaksimalkan, begitupula sebaliknya.

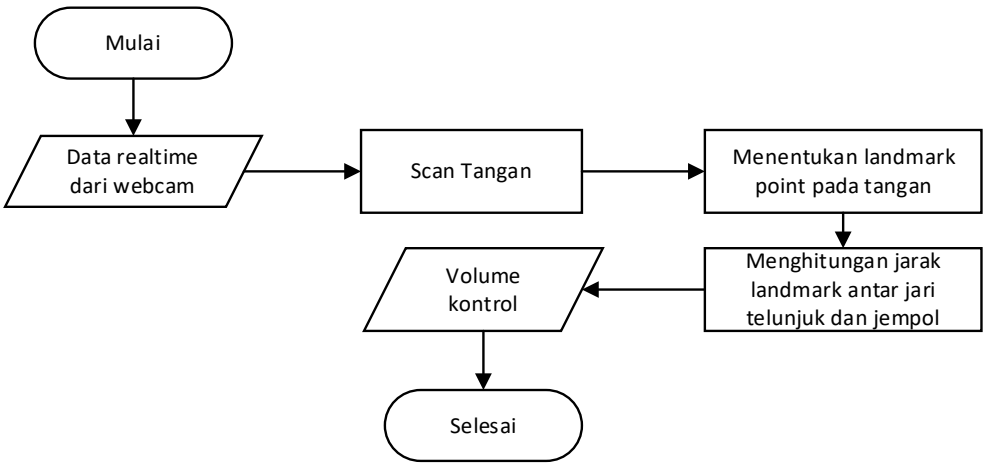
B. Jenis Penelitian

Teknik pengumpulan studi pustaka dilakukan dengan mempelajari beberapa topik yang berhubungan dengan penelitian ini. Referensi sumber yang digunakan berasal dari buku, jurnal, internet, prosiding dan sebagainya. Pada penelitian ini menggunakan metode R&D (*Research and Development*), dimana metode penelitian ini akan menghasilkan produk perangkat lunak tertentu untuk diujikan efektifitasnya [12]. Dapat dikatakan bahwa tujuan penelitian R&D adalah menginformasikan proses pengambilan keputusan sepanjang pengembangan dari suatu produk menjadi berkembang dan kemampuan pengembang untuk menciptakan berbagai hal dari jenis ini pada situasi kedepan. Pada Gambar 4, ditunjukkan langkah-langkah yang harus dilakukan untuk prosedur metode penelitian [12].

C. Flowchart Handtracking Gesture

Flowchart sistem digunakan untuk menunjukkan alur kerja dari penelitian ini dimana sistem secara keseluruhan menjelaskan prosedur urutannya. Berikut pada Gambar 5 ditunjukkan alur mengenai *hand recognition* sampai proses analisis *landmark*. Data dari *webcam* akan dibaca menggunakan *library* OpenCV untuk dianalisis. Proses *scan* tangan dilakukan secara *real-time* dan akan menghasilkan *landmark point*. Proses analisis ini menggunakan *library* dari MediaPipe dan OpenCV dengan bahasa pemrograman Python.

Gambar 4. Langkah-langkah *R&D*



Gambar 5. Flowchart Hand Recognition dengan *MediaPipe*

Jarak dari poin *landmark* jempol dan jari telunjuk dihitung untuk menentukan kontrol *volume*. Perhitungan jarak menggunakan rumus *hypotenuse atau rumus Pythagoras* yang ditunjukan pada persamaan 1. Dari hasil perhitungan jarak dilakukan interpolasi linear untuk menentukan *volume* suara pada laptop atau komputer.

$$c = \sqrt{a^2 + b^2} \qquad (1)$$

Keterangan:

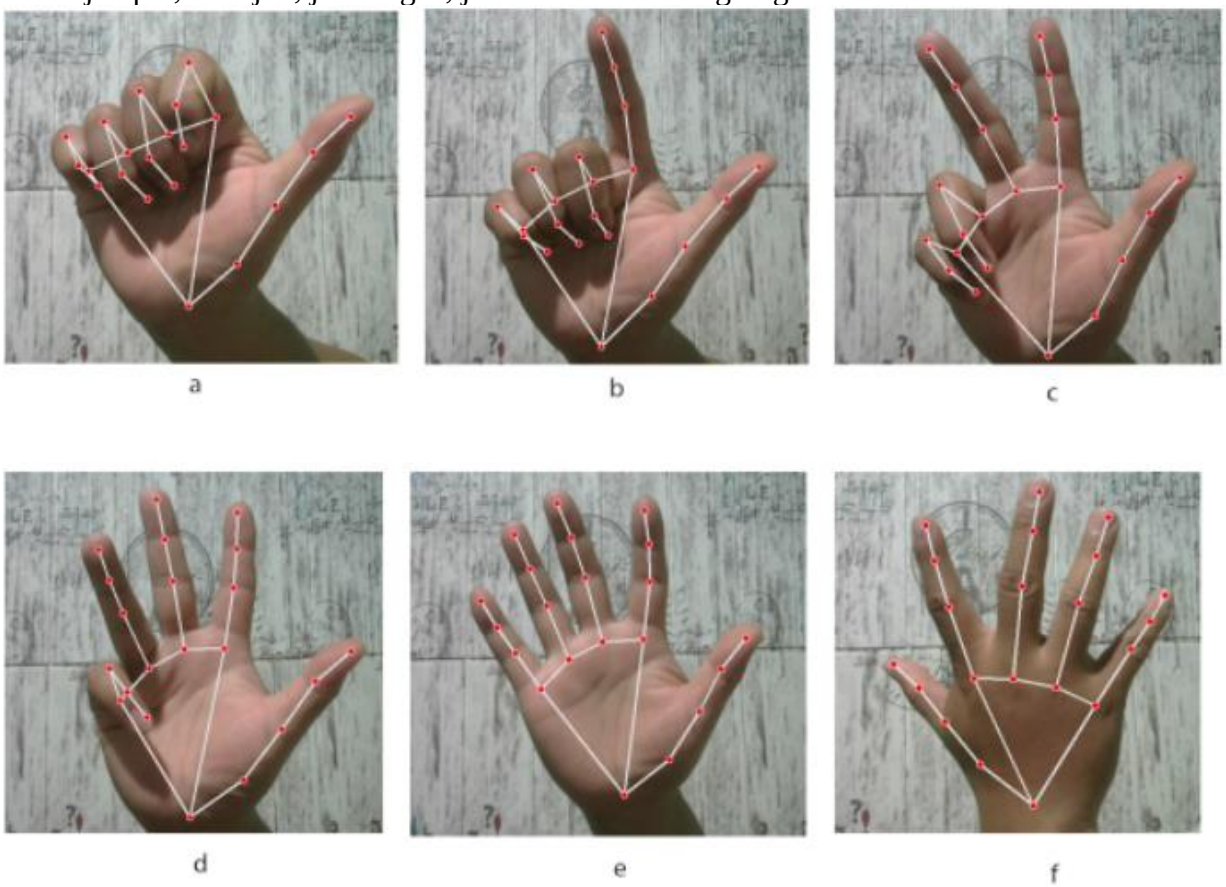
a = koordinat dari ujung telunjuk dikurangi dengan ujung jempol
b = koordinat dari ujung telunjuk dirkurangi dengan ujung jempol
c = Jarak antara ujung jempol dengan ujung telunjuk

DOI: https://doi.org/10.35457/antivirus.v16i2.2508

IV. HASIL DAN PEMBAHASAN

A. *Pengambilan Data Tangan*

Berdasrkan *flowchart* yang dijelaskan pada bab sebelumnya, maka langkah pertama penelitian ini adalah mengambil sampel data tangan secara *real-time* dengan berbagai pose. Gambar tangan yang dipakai adalah tangan kanan manusia. Dipilihnya tangan kanan karena kebiasaan dari orang Indonesia yang menggunakan tangan kanan untuk melakukan berbagai kegiatan dan pekerjaan. Pose berbagai macam gesture gerakan jari tangan kanan manusia diilustrasikan pada Gambar 6. Setiap *landmark keypoint localization* dapat ditangkap kamera dan digambar ulang secara jelas oleh machine learning yang dibuat. Dari Gambar 6a-6e ditunjukkan gambar posisi kamera/webcam dalam pengambilan gerakan *gesture* jari yang benar yakni posisi telapak tangan (*palm*) terlihat jelas menghadap kamera/webcam. Saat kondisi telapak tangan dibalik seperti pada Gambar 6f, ternyata *machine learning* masih bisa mengenali pola *landmark* yang ada pada jari dengan baik, yang artinya bisa membedakan letak semua jari dengan benar mulai dari jempol, telunjuk, jari tengah, jari manis dan kelingking.

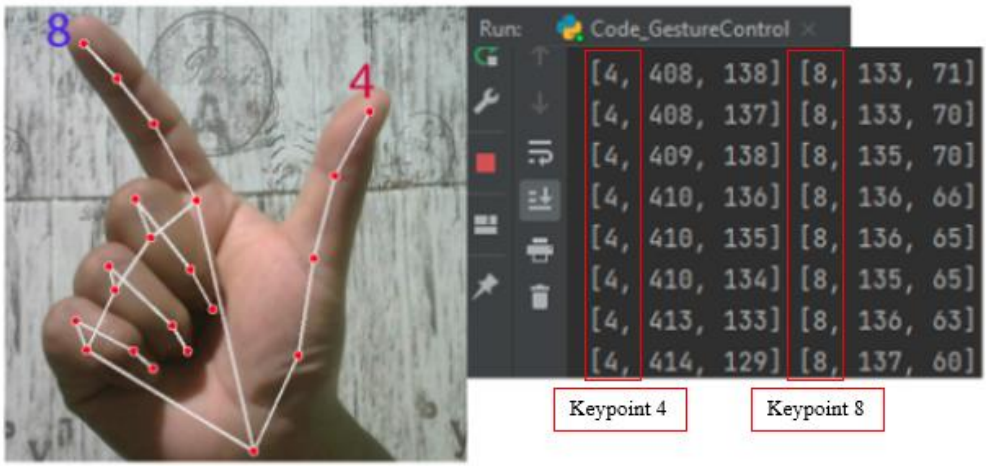


Gambar 6. Pose Tangan Kanan

B. *Menentukan jari jempol dan jari telunjuk*

Langkah selanjutnya melakukan *mapping* terhadap lokasi dari jari jempol dan telunjuk saja. Dengan menentukan *landmark point* 1,2,3,4 (*thumb finger*) dapat diketahui posisi jempol. Untuk *mapping* jari telunjuk dengan menentukan *landmark point* 5, 6, 7, 8 (*index finger*). Setiap posisi koordinat x dan y pada *frame* masing-masing *thumb finger* dan *index finge*r hanya ditangkap bagian ujung-ujungnya dari setiap *landmark* yang ada. Sebagai contoh terlihat pada Gambar 7, dimana *landmark keypoint localization* nomor 4 saja yang digunakan untuk menentukan ujung jempol, dan *landmark keypoint localization* nomor 8 digunakan untuk menentukan ujung jari telunjuk. Setiap sedikit saja terjadi perubahan frame, maka lokasi koordinat keypoint 4 dan 8 diubah sesuai lokasinya. Hal ini bisa dilihat dari Gambar 7 untuk data pertama sampai data kedepan, posisi koordinatnya berubah-ubah.

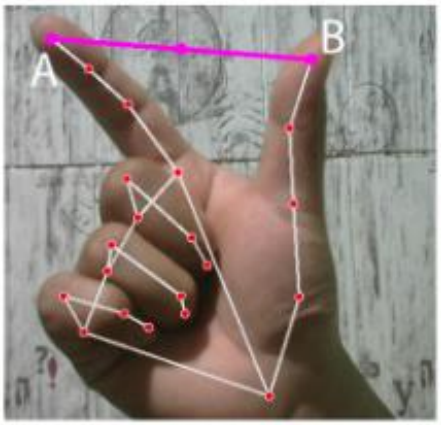
DOI: https://doi.org/10.35457/antivirus.v16i2.2508



Gambar 7. Landmark Jari Telunjuk dan Jempol

C. Menghitung Jark Jari

Dimisalkan pada Gambar 8 di bawah ini *landmark* A adalah ujung dari jari telunjuk, sedangkan *landmark* B adalah ujung dari jempol. Untuk menghitung jarak dari posisi *real-time* A dengan B, dilakukan dengan rumus pada persamaan 1. Perhitungan jarak ini selalu berubah-ubah secara *real-time* dikarenakan adanya pergerakan dari tangan yang memang tidak bisa stabil diam ditempat. Sedikit pergeseran posisi jari tangan akan mempengaruhi nilai x dan y dari *frame* yang ditangkap kamera/webcam tersebut.



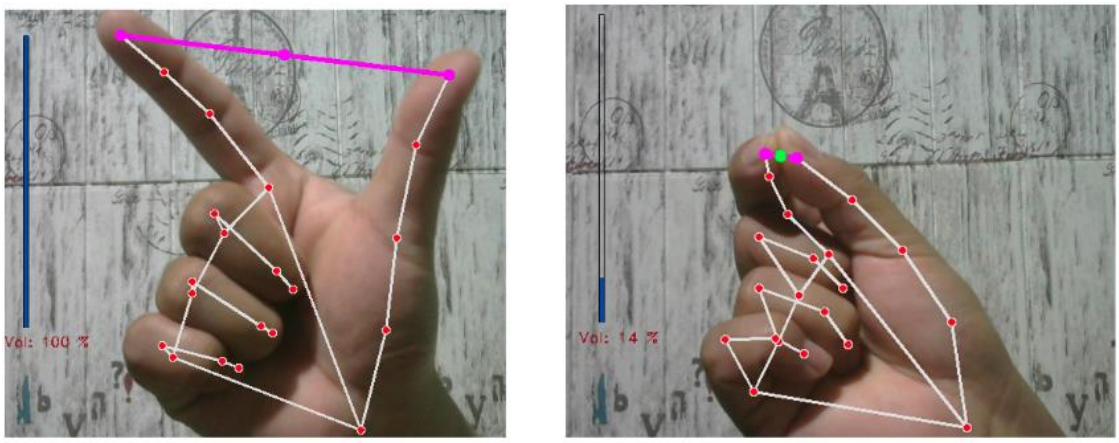
Gambar 8. Jarak A ke B

Hasil dari perhitungan beberapa kali pose jari antara jarak A dan B ditunjukkan pada Tabel 1. Dari jarak dilanjutkan ke *mapping* kontrol *volume*. Jika hasil perhitungan jarak antara ujung telunjuk dengan ujung jempol kecil, maka *volume* diseting rendah. Begitu pula sebaliknya, jika hasil jaraknya besar maka *volume* suara akan diperbesar. Pada Gambar 9a ditunjukan ketika kamera/webcam menangkap pergerakan *gesture* jari tangan dan menghasilkan *volume* suara maksimal. Sedangkan Gambar 9b ditunjukkan ketika *volume* suara mengecil ditandai dengan jarak yang paling mininum atau titik warna hijau.

Tabel 1. Hasil Perhitungan Jarak A ke B

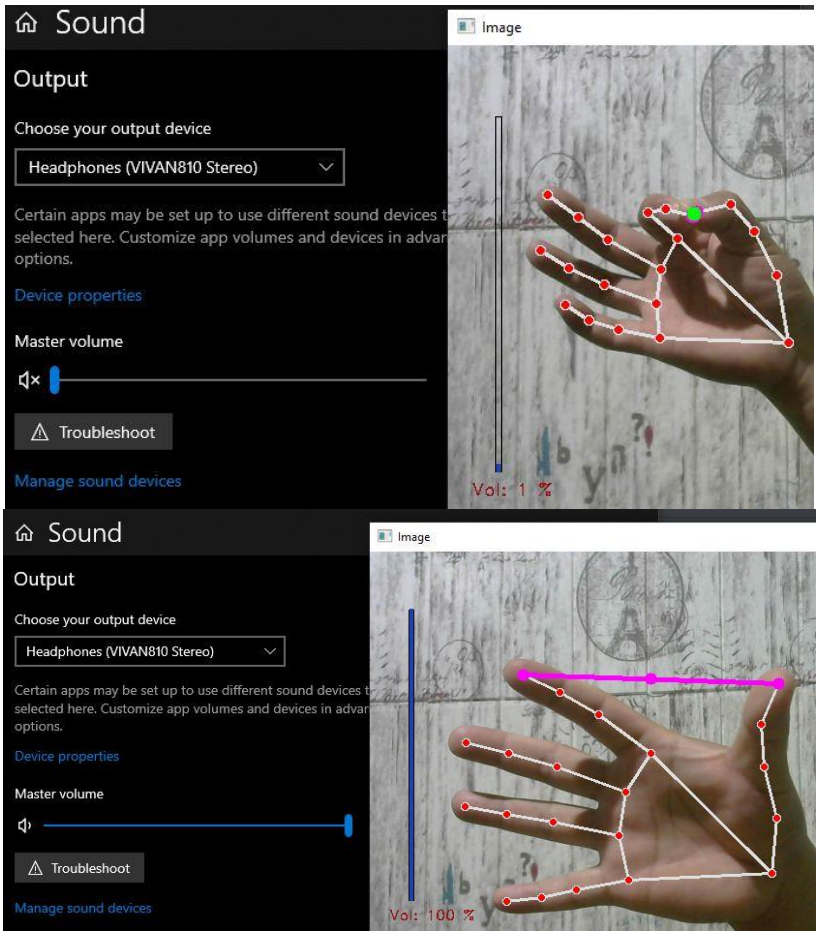
X1	X2	Y1	Y2	Jarak
435	201	170	125	238
442	205	167	123	241
442	208	165	121	238
442	213	164	116	234
453	217	160	112	241
456	219	158	111	242
461	223	159	110	243
462	226	160	110	241
463	226	160	110	242
462	226	163	115	241

DOI: https://doi.org/10.35457/antivirus.v16i2.2508



Gambar 9. Perubahan Volume Suara Berdasarkan Gesture Jari Tangan

Hasil yang diterapkan pada kontrol volume di di laptop/komputer untuk sistem operasi Windows dapat dilihat pada Gambar 10.

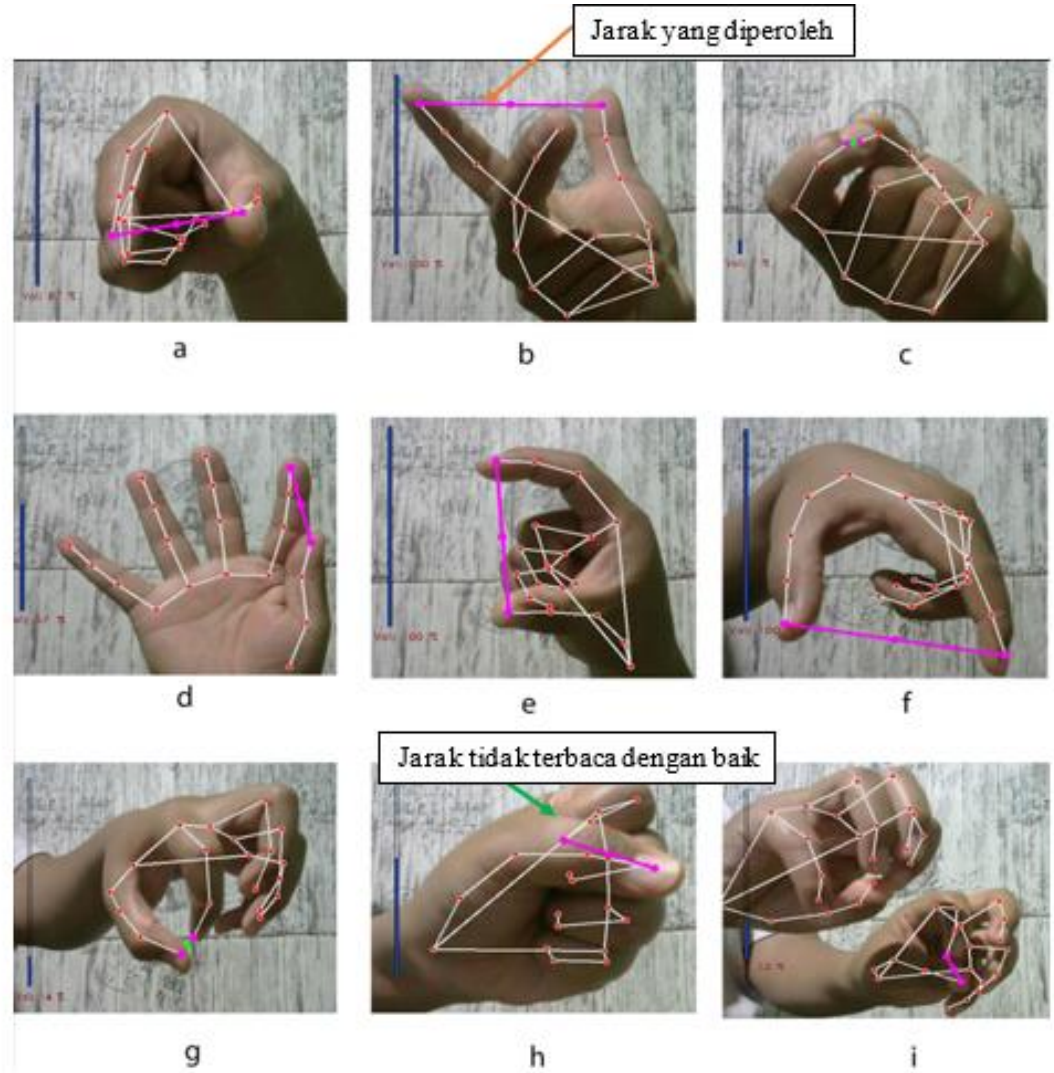


Gambar 10. a) hasil kontrol volume dengan kondisi 1%. b) hasil kontrol volume dengan kondisi 100%

Untuk mengetahui seberapa akurat OpenCV dan MediaPipe melakukan *handtracking gesture* ini, maka peneliti melakukan pengujian dengan berbagai pose jari tangan yang ditunjukkan pada Gambar 11. Dilakukan sebanyak sembilan pose tangan yang berbeda dan *machine learning* menghasilkan nilai akurasi sebesar 88,89%. Nilai akurasi ini diperoleh dari hasil testing 9 pose yang mana hanya 8 pose saja yang menunjukan hasil pembacaan *gesture* gerakan jari dengan tepat. Saat kondisi ujung jempol dan ujung jari telunjuk terlihat dengan jelas seperti yang ditunjukkan Gambar 11a-i (kecuali 11h), maka jarak yang dibuat pun terlihat dengan jelas, bisa dilihat pada Gambar 11 warna pink tebal. Akibatnya volume media suara pada komputer/laptop bisa diperbesar atau diperkecil suaranya seacra tepat melalui *gesture* gerakan mencubit atau merenggangkan jari jempol dan telunjuk. Namun bisa dilihat pada Gambar 11h, dimana

DOI: https://doi.org/10.35457/antivirus.v16i2.2508

saat kondisi jari tangan mencoba menggenggam, *keypoint localization* antara ujung jempol dan ujung telunjuk terhalangi. Terdapat satu pose tangan yang tidak bisa diidentifikasi polanya dengan benar yakni di Gambar 11h bahwa pose tersebut tampak ujung jari telunjuk yang tidak bisa dikenali. Sehingga, *volume* suara yang dihasilkan berubah secara drastis, terkadang *volume* naik menjadi maksimal ataupun turun menjadi minimal.



Gambar 101. Hasil Pengujian Akurasi

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Dari hasil pembahasan, *hand tracking recognition* dari *gesture* jari jempol dan jari telunjuk serta hasil pengujian, maka pada penlitian ini dapat diambil kesimpulan sebgai berikut:

1. Dari berbagai macam pose tangan yang ditangkap oleh kamera, OpenCV dan MediaPipe bisa melakukan *mapping landmark keypoint localization* jari tangan secara akurat dan *real-time* khusunya jari tangan kanan untuk jempol dan jari tulunjuk saja.
2. Perhitungan jarak menggunakan interpolasi linear dari hasil perhitungan jarak *Pythagoras*. Jika jarak yang dihasilkan dari *gesture* kecil, maka *volume* suara akan dikecilkan, begitu pula sebaliknya. Semua proses pose tangan bisa terdeteksi jaraknya, namun terdapat kendala jika jangkauan kamera terlalu dekat dengan objek yakni tangan, maka hasil interpolasinya akan buruk. Akibatnya *volume* suara tidak bisa diseting dengan benar. Hasil dari pengujian akurasinya dengan berbagai pose gerakan jari maka diperoleh nilai 88,89% dimana hanya satu dari sembilan testing yang gagal membaca gesture gerakan jari tangan kanan. Ini disebabkan *landmark point localization* ujung jari telunjuk yang tertutup oleh kepalan jari jempol.

DOI: https://doi.org/10.35457/antivirus.v16i2.2508

B. Saran

Pada aplikasi yang telah dibuat oleh penulis terdapat ketidak sempurnaan. Sehinga, penulis memiliki saran guna menyempurnakan penelitian ini dimasa nanti yakni posisi kamera terhadap tangan yang mempengaruhi pengukuran jarak jari jempol ke telunjuk. Alangkah baiknya jika pada kamera diberikan penanda minimal jarak yang harus digunakan untuk menangkap gambar jari, misalkan 30 cm dari jarak kamera. Dengan begitu semua gambar jari yang ditangkap akan memiliki standard yang sama.

DAFTAR PUSTAKA

[1] M. E. A. Rivan and A. Setiawan, "Pengenalan Gestur Angka Pada Tangan Menggunakan Arsitektur AlexNet Dan LeNet Pada Metode Convolutional Neural Network," *Jurnal Sistem Komputer*, vol. 11, pp. 19-28, 2022.

[2] H. Yunita and E. Setyanti, "Hand Gesture Recognition Sebagai Pengganti Mouse Komputer Menggunakan Kamera," *Jurnal ELTIKOM*, vol. 3, pp. 64-76, 2019.

[3] T. C. A.-S. Zulkhaidi, E. Maria, and Yulianto, "Pengenalan Pola Bentuk Wajah dengan OpenCV," *Jurnal Rekayasa Teknologi Informasi (JURTI)*, vol. 3, pp. 181-185, 2019.

[4] P. Priyonggo, A. Kusumah, A. Khumaidi, M. B. Rahmat, and J. Endrasmono, "Sistem Tracking Posisi Kamera Menggunakan Pengolahan Citra Untuk Pemusatan Posisi Pengambilan Video di Automation Academy," *Jurnal Teknik Elektro dan Komputer TRIAC*, vol. 9, 2022.

[5] T. C. A.-S. Zulkhaidi, E. Maria, and Yulianto3, "Pengenalan Pola Bentuk Wajah dengan OpenCV," *JURTI (Jurnal Rekayasa Teknologi Informasi)*, vol. 3, 2019.

[6] F. Damatraseta, R. Novariany, and M. A. Ridhani, "Real-time BISINDO Hand Gesture Detection and Recognition With Deep Learning CNN," *Jurnal Informatikan Kesatuan (JIKES)*, vol. 1, 2021.

[7] F. Zhang *et al.*, "MediaPipe Hands: On-Device Real-Time Hand Tracking," *ArXiv*, vol. abs/2006.10214, 2020.

[8] M. R. Azharfianto, N. C. Basjaruddin, and E. Rakhman, "Pengenalan Gestur Tangan Berbasis Augmented Reality dan Metode Logika Fuzzy Untuk Mengendalikan Kendaraan," *Industrial Research Workshop and National Seminar*, vol. 9, 2018.

[9] I. G. P. S. Wijaya, A. A. Firdaus, A. P. J. Dwitama, and Mustiari, "Pengenalan Ekspresi Wajah Menggunakan DCT dan LDFA Untuk Aplikasi Pemutar Musik (Moodsic)," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, vol. 5, pp. 559-565, 2018.

[10] H. M. Putri, F. Fadlisyah, and W. Fuadi, "Pendeteksian Bahasa Isyarat Indonesia Secara Real-Time Menggunakan Long Short-Term Memory (LSTM)," *Jurnal Teknologi Terapan and Sains 4.0*, vol. 3, no. 1, pp. 13-25, 2022.

[11] A. Halder and A. Tayade, "Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning," *International Journal of Research Publication and Reviews (IJRPR)*, vol. 2, no. 5, pp. 9-17, 2021.

[12] Sugiono, *Penelitian kuantitatif, kualitatif dan R&D*. Bandung: Alfabeta, 2014.

DOI: https://doi.org/10.35457/antivirus.v16i2.2508