

Implementasi Pendeteksi Gesture Tangan dengan Metode Convex Hull

1st Estella, T.

Computer Science Department, School
of Computer Science
Bina Nusantara University
Jakarta, Indonesia
tricia.estella@binus.ac.id

line 1: 2nd Regina, S.

Computer Science Department, School
of Computer Science
Bina Nusantara University
line 4: City, Country
sharlene.regina@binus.ac.id

line 1: 3rd Nathaniel, G.

Computer Science Department, School
of Computer Science
Bina Nusantara University
line 4: City, Country
gilbert.nathaniel@binus.ac.id

Abstract—Gestur tangan merupakan salah satu bentuk komunikasi nonverbal yang sudah sering diterapkan dalam beberapa bidang seperti komunikasi antara orang bisu tuli, kontrol robot, interaksi manusia-komputer (HCI), otomatisasi rumah dan aplikasi medis. Dalam proyek ini, kami mengusulkan sebuah metode untuk mendeteksi gestur tangan dalam video capture menggunakan algoritma convex hull. Convex hull adalah konsep matematika yang menjelaskan himpunan convex (cembung) terkecil yang melingkupi sekumpulan titik. Dengan menerapkan algoritma convex hull ke kumpulan titik berbentuk tangan, kita dapat mendeteksi keberadaan dan perkiraan lokasi tangan dalam sebuah gambar. Kami mendemonstrasikan keefektifan metode kami pada video capture (webcam) dan menunjukkan bahwa metode ini bekerja dengan baik dalam aplikasi video real-time. Selain itu, kami membahas kelebihan, kekurangan serta pengembangan yang dapat dilakukan pada metode convex hull untuk mendeteksi gestur tangan.

Keywords—hand gesture detection, convex hull, computer vision

I. INTRODUCTION

Computer vision adalah bidang kecerdasan buatan yang berfokus pada kemampuan komputer untuk menginterpretasikan dan memahami data visual dari dunia di sekitar mereka [1]. Ini melibatkan pengembangan algoritma dan sistem yang dapat menganalisis dan memahami input visual, seperti gambar dan video, untuk mengekstraksi informasi yang berguna dan membuat keputusan berdasarkan data tersebut.

Computer vision memiliki berbagai aplikasi, termasuk analisis gambar dan video, pengenalan objek, pengenalan wajah, dan robotika. Di setiap area ini, algoritma dan sistem computer vision digunakan untuk menganalisis data visual dan membuat keputusan berdasarkan data tersebut. Misalnya, dalam analisis gambar dan video, algoritma computer vision dapat digunakan untuk mengidentifikasi objek dan orang dalam suatu adegan, mengenali pola dan fitur, serta mengekstrak informasi tentang lingkungan [2]. Dalam pengenalan objek, algoritma computer vision dapat digunakan untuk mengidentifikasi objek tertentu dalam gambar atau video, seperti mobil, hewan, atau manusia. Dalam pengenalan wajah, algoritma visi komputer dapat digunakan untuk mengidentifikasi individu tertentu berdasarkan fitur wajah mereka. Dan dalam robotika, komputer dapat digunakan untuk memungkinkan robot menavigasi dan berinteraksi dengan lingkungannya berdasarkan data visual.

Pada project kali ini, kami ingin mengembangkan aplikasi computer vision untuk mendeteksi gestur tangan menggunakan algoritma convex hull. Metode convex hull adalah teknik yang dapat digunakan dalam computer vision untuk mendeteksi gerakan tangan pada gambar dan video. Convex adalah suatu objek tanpa sudut di dalamnya yang lebih besar daripada 180 derajat. Object yang bukan merupakan convex adalah concave, dan sudut di dalamnya adalah defects. Sedangkan hull adalah eksterior atau bentuk dari objek tersebut. Oleh karena itu, Convex Hull dari suatu bentuk atau sekelompok titik adalah batas cembung yang pas di sekitar titik atau bentuk tersebut [3]. Dengan menerapkan algoritma convex hull ke sekumpulan titik berbentuk tangan pada gambar, kita dapat mendeteksi keberadaan dan perkiraan lokasi tangan pada gambar.

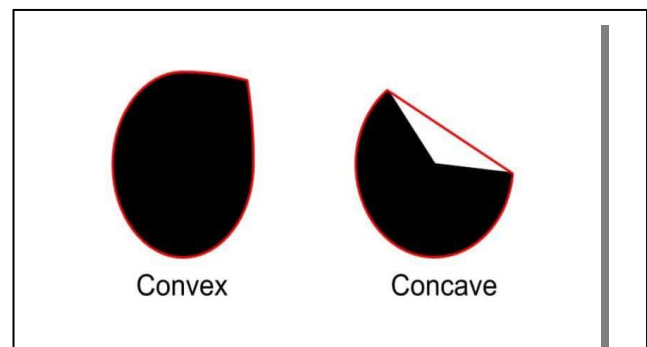


Fig. 1. Convex hull ditandai dengan outline merah, source: learnopencv

Untuk menggunakan metode convex hull untuk deteksi gestur tangan, pertama-tama kita perlu mengekstraksi titik berbentuk tangan dari gambar atau video. Ini dapat dilakukan dengan menggunakan berbagai teknik, seperti deteksi warna kulit, deteksi tepi, atau deteksi blob [4]. Setelah kami mengidentifikasi titik-titik berbentuk tangan, kami dapat menerapkan algoritma convex hull untuk menghasilkan convex hull yang melingkupi titik-titik ini.

Dengan menganalisis bentuk dan orientasi lambung cembung, kita dapat menentukan gestur yang dibuat oleh tangan. Misalnya, jika convex hull lebar dan rata, ini mungkin menandakan tangan terbuka dan rata. Jika lambung cembung lebih bundar dan padat, ini mungkin menunjukkan bahwa tangan terkepal. Namun pendeteksi paling cepat adalah dengan menghitung defects pada convex, dalam hal ini adalah cekungan di antara jari. Dengan menggabungkan metode convex hull dengan teknik lain, seperti machine learning, dimungkinkan untuk mengenali berbagai gerakan tangan dengan akurasi tinggi.

II. RELATED WORKS

Beberapa studi dilakukan menggunakan metode Convex Hull dan Convexity Defects untuk mengenalkan bentuk gestur telapak tangan seseorang. (Ganapathyraju 2013) Sebuah studi mempelajari metode gesture recognition dengan tujuan untuk menggerakkan robot. Terdapat beberapa stage yang perlu dilalui sebelum robot dapat dikendalikan menggunakan hand gesture. Stage tersebut antara lain adalah skin detection, menentukan outline dari telapak tangan (Convex Hull Algorithm), menentukan jumlah jari hasil deteksi (Convexity Hull Algorithm), dan mengirim informasi perhitungan algoritma kepada controller. Hasil akhir yang didapat adalah robot dapat digerakkan berdasarkan jumlah jari yang terdeteksi oleh modul gesture recognition.

(Yuliana, N., & Wardani, K. 2017) Studi lain mempelajari pengenalan hand gesture yang sama namun menggunakan kombinasi dari ibu jari, jari telunjuk dan jari kelingking yang mewakili operasi dasar. Algoritma Freeman Chain Code digunakan untuk pengenalan bentuk dan segmentasi telapak tangan. Algoritma Convex Hull digunakan untuk mendeteksi hull. Algoritma Convexity Defects digunakan untuk menentukan defects pada citra telapak tangan yang digunakan untuk membedakan antara jari. Algoritma terakhir yang dipelajari adalah Maximum Inscribed Circle untuk mendeteksi titik pusat telapak tangan. Hasil deteksi jari menggunakan algoritma - algoritma berikut mencapai nilai akurasi sebesar 90%.

(Wulanningrum et al., 2022) Studi lain juga menggunakan metode Convex Hull untuk mendeteksi 2 jenis hand gesture (terbuka dan terkepal). Proses yang dilakukan adalah: pengenalan telapak tangan menggunakan hardware webcam, thresholding untuk memisahkan telapak tangan dengan background, mengaplikasikan algoritma Convex Hull dan Convexity untuk mendeteksi hull dan defects sebagai data yang digunakan untuk mendeteksi hand gesture.

Kekurangan yang dimiliki oleh beberapa paper adalah seperti pencahayaan yang terlalu berlebihan [5] ataupun kurang [6] sehingga gestur tangan tidak bisa diproses oleh algoritma. Kendala berikutnya yang ditemui oleh studi tersebut adalah kombinasi gestur tangan yang memiliki limitasi sehingga hasil kombinasi gestur tidak banyak [4]. Gestur terbanyak dari eksperimen adalah 8 gestur berbeda dari 1 tangan [5]. Beberapa gestur juga sulit dikenali oleh algoritma terkhususnya keakuratan bentuk ibu jari yang renggang sehingga dianggap tidak valid oleh algoritma [5]. Namun potensi penggunaan metode ini bisa dipakai dalam industri luas seperti dunia robotik [4], telemedicine [4] dan lain sebagainya.

III. METHODOLOGY

Dalam membuat model deteksi gestur tangan, kami menggunakan beberapa algoritma yang sering digunakan pada topik computer vision. Algoritma-algoritma tersebut antara lain adalah grayscale, blur, thresholding, dan convex hull.

A. Grayscale

Grayscale merupakan teknik image conversion dalam fotografi digital. Cara kerja dari Grayscale adalah mengubah warna pada gambar original (RGB) menjadi 2 dimensi warna saja. Piksel yang paling terang ditunjukkan sebagai warna putih, sedangkan piksel yang paling gelap sebagai warna hitam [7].

Grayscale memiliki 3 parameter dasar pada model, yaitu Hue, Saturation dan Brightness. Pada gambar dengan colour model grayscale, hue (colour shade) dan saturasi (colour intensity) pada setiap piksel bernilai 0. Brightness (kecerahan) adalah satu-satunya parameter piksel yang bervariasi nilainya. Brightness memiliki skala dari nilai minimal 0 (hitam) hingga nilai maksimal 100 (putih).

Grayscale cocok diterapkan pada model ini, dikarenakan dapat menyederhanakan dan menghilangkan kompleksitas dalam proses komputasi gambar. Grayscale mengompres gambar original ke piksel yang minimum. Dengan begitu, kinerja model dapat meningkat. Model dapat lebih mudah membedakan antara detail bayangan dengan objek tangan dari video pada persepsi 2 dimensi spasial ketimbang pada 3 dimensi.

B. Gaussian Blur

Gaussian blur adalah salah satu metode yang digunakan untuk image blurring. Edge merupakan konsep penting dalam pengidentifikasian objek pada gambar. Dalam proses image blurring, proses akan melakukan colour transition untuk memperhalus edge dalam gambar.

Untuk menerapkan kernel ke suatu piksel, rata-rata nilai warna piksel yang mengelilinginya dihitung, ditimbang dengan nilai di kernel. Dalam Gaussian blur, piksel yang paling dekat dengan pusat kernel diberi bobot lebih dibandingkan piksel yang jauh dari pusat. Tingkat pengurangan bobot ini ditentukan oleh fungsi Gaussian. Fungsi Gaussian dalam 2D akan menghitung dimensi X dan Y dengan rata-rata piksel μ bernilai 0, dan mewakili bagian tengah dari kernel 2D. Peningkatan nilai σ^2 di salah satu dimensi akan meningkatkan blurring di dimensi tersebut. [8] Pada model ini, gaussian filter digunakan untuk untuk menghilangkan noise dan memperhalus gambar sebelum dilakukannya image processing.

C. Thresholding: Otsu's Binarization Method

Thresholding adalah proses pembuatan binary image dari hasil pengolahan grayscale image. Thresholding biasa digunakan untuk image segmentation antara latar belakang dan objek pada gambar input. Pada Manual Thresholding, proses dilakukan dengan cara mengubah intensitas piksel berdasarkan konstanta threshold yang telah ditentukan. Bila intensitas suatu piksel bernilai lebih kecil dari threshold, maka nilai piksel tersebut diubah menjadi 0, bila sebaliknya maka nilai piksel akan menjadi maximum value.

Pada model ini, kami menggunakan Otsu's Binarization Method, proses thresholding yang lebih adaptif dibandingkan dengan manual thresholding. Bila dibandingkan dengan manual thresholding, konstanta threshold yang digunakan pada proses Otsu's Binarization tidak ditentukan secara manual, sehingga akan disesuaikan secara otomatis dengan gambar input [9]. Tingkat gray level pada gambar berskala dari 0 ke 255 piksel. Otsu's Binarization akan terus melakukan proses pencarian nilai threshold yang paling optimal, dilihat dari hasil intra-class variance yang paling minimum.

Selain itu, model ini juga akan membutuhkan contour berdasarkan hasil thresholding yang sudah dibuat. Contour sendiri adalah garis terluar yang mengelilingi suatu objek, garis tersebut terbentuk dari penyatuan tiap edge terluar dengan nilai intensitas yang sama. Contour dapat

mencerminkan bentuk geometri, serta lokasi objek pada gambar. Contour Detection bekerja dengan menggunakan convex hull, yaitu sekumpulan convex terkecil yang terdiri dari titik piksel yang ada pada objek dalam gambar. [10]

D. Algoritma Convex Hull

Ada beberapa algoritma yang dapat digunakan untuk menghitung convex hull dari sekumpulan titik, termasuk algoritma gift wrapping, algoritma Graham scan, dan algoritma divide and conquer. Adapun pada project ini kami menggunakan library opencv yang memanfaatkan algoritma gift wrapping.

Algoritma gift wrapping, atau Jarvis march algorithm, adalah metode untuk menghitung convex hull dari sekumpulan titik pada sebuah bidang. Algoritma bekerja dengan menambahkan titik pada convex hull, dimulai dari sebuah point pada hull dan menambahkan titik selanjutnya pada set yang paling jauh dari hull sekarang [11]. Pemilihan titik awal pada hull dapat dilakukan dengan memilih poin dengan koordinat x atau koordinat y terkecil, dan menginisialisasi hull tersebut kedalam list. Selanjutnya, kita dapat mengulangi titik yang tersisa pada set dan menemukan titik yang terjauh dari hull saat ini, dan menambahkan titik tersebut ke hull dan memperbarui hull saat ini menjadi convex hull dari titik di hull ditambah titik baru. Proses tersebut diulangi sampai semua titik sudah diperiksa.

Pada project ini, kami menghitung defects pada gambar tangan yang ditangkap untuk mendeteksi berapa jari yang ditunjukkan ke komputer. Oleh karena itu, kita memerlukan convexity defects, yaitu poin yang berada pada convex hull namun bukan bagian dari hull, dan poin-poin tersebut dapat diidentifikasi dari apakah titik tersebut dikeluarkan dari hull dan ditambahkan kembali kedalam hull dalam proses gift wrapping. Adapun defects berada pada lekukan diantara gambar jari-jari tangan yang diambil. Namun, sebuah defects dapat memiliki sudut sampai dengan 180 derajat, sedangkan manusia tidak dapat membuat gestur tangan dengan sudut diantara kedua jarinya sebesar 180 derajat. Oleh karena itu, akan dibutuhkan analisis defects yang cocok untuk dihitung sebagai defects tangan, dengan mengaplikasikan aturan kosinus untuk membuang defects dengan sudut lebih dari 90 derajat. Selain itu, deteksi gestur tangan dengan metode defects convex hull akan mengakibatkan model tidak dapat mendeteksi tangan tanpa defect, seperti kepalan tangan atau gestur angka satu.

IV. EXPERIMENT

Model pendeteksi gestur tangan menggunakan convex hull defects kami buat menggunakan bahasa pemrograman python, dengan menggunakan library cv2 untuk mendukung method yang dibutuhkan. Adapun versi openCV yang digunakan adalah versi keempat.

A. Algoritma

Berikut adalah algoritma dari model yang kami buat:

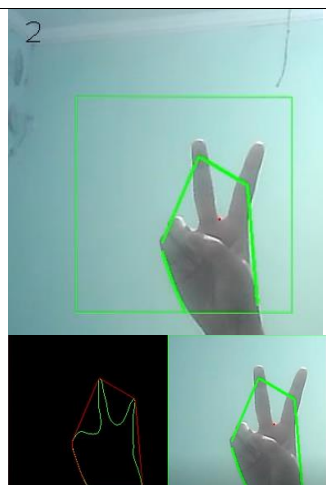
- 1) Membuka video capture dengan cv2.VideoCapture
- 2) Selama video capture terbuka, maka kita akan membaca setiap frame yang ditangkap dari video capture tersebut.

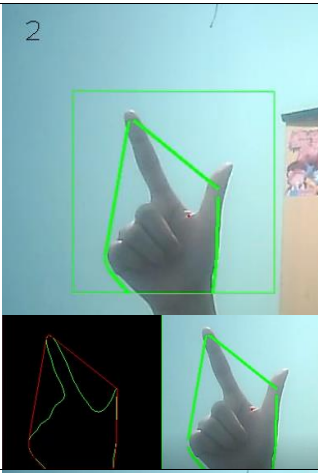
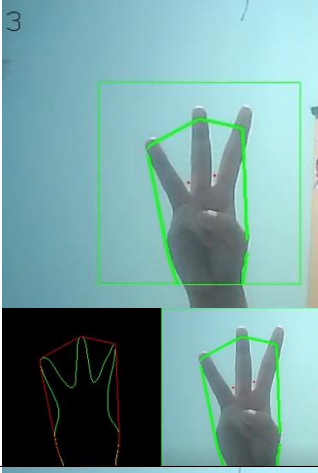
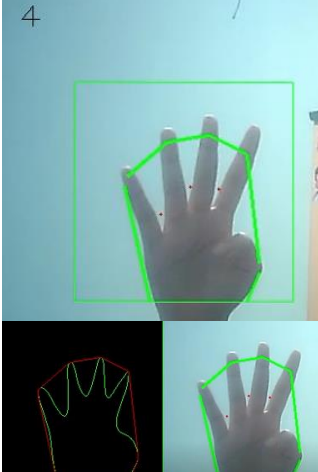
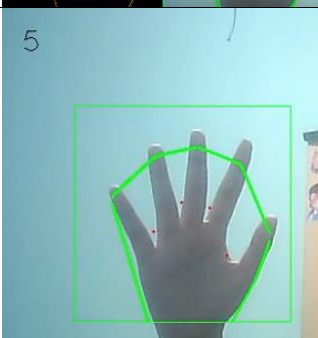
- 3) Kita hanya akan membaca gambar pada window yang ditentukan, maka kita dapat menggambar rectangle pada aplikasi dan memotong gambar sesuai besar window tersebut
- 4) Mengubah gambar yang ditangkap menjadi grayscale
- 5) Menggunakan gaussian blur untuk mengurangi noise
- 6) Menggunakan thresholding otsu's binarization method agar komputer mendeteksi garis pada gambar
- 7) Mencari contours sesuai dengan hasil thresholding
- 8) Mencari contour terbesar, yaitu contour objek (tangan)
- 9) Mencari convex hull berdasarkan contour terbesar, dan menggambarkan contour serta convex hull pada aplikasi untuk kepentingan eksperimen
- 10) Mencari hull
- 11) Mencari defects (cekung antara jari) dari contour dan hull yang ditemukan.
- 12) Menggunakan aturan cosinus untuk menghitung sudut tiap defects, dan menyimpan defects dengan sudut lebih kecil dari 90 derajat, dan menggambarkan defects tersebut serta convex pointnya (ujung jari)
- 13) Menampilkan hasil prediksi pada aplikasi, yaitu angka 2 sampai dengan 5
- 14) Menampilkan seluruh hasil pada aplikasi (imshow).
- 15) Aplikasi akan tertutup jika user menekan tombol x pada keyboardnya.

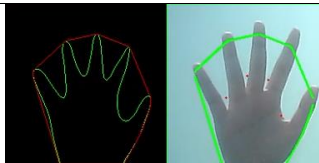
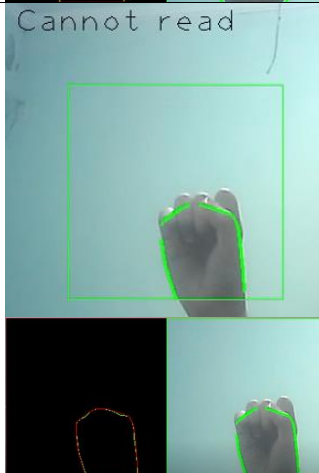
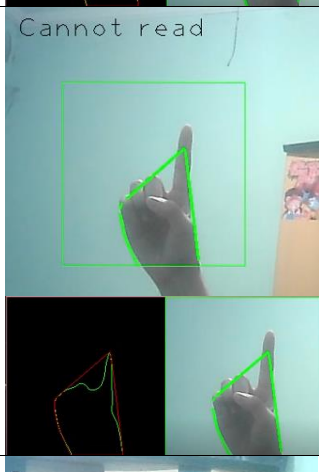
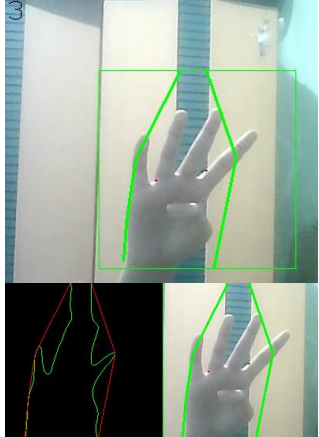
B. Result

Model dapat memberikan hasil prediksi gestur tangan dengan cukup baik, dengan memberikan respons berapa banyak jari yang diangkat oleh user. Adapun model tidak dapat mendeteksi kepalan tangan atau satu jari yang terangkat, juga tidak dapat mendeteksi gestur tangan diluar box dan background yang kurang jelas. Berikut adalah tabel rincian tampilan aplikasi untuk beberapa kasus:

TABLE I. TABLE EXPERIMENT RESULT

No	Gambar	Keterangan
1		Model dapat memprediksi gestur yang menampilkan dua jari dengan baik. Titik merah menandakan terdapat satu defect pada gambar, yang berarti ada dua jari yang diangkat oleh user.

No	Gambar	Keterangan
2		<p>Model dapat memprediksi gestur yang menampilkan dua jari dengan baik.</p> <p>Titik merah menandakan terdapat satu defect pada gambar, yang berarti ada dua jari yang diangkat oleh user.</p>
3		<p>Model dapat memprediksi gestur yang menampilkan tiga jari dengan baik.</p> <p>Dua titik merah menandakan terdapat dua defect pada gambar, yang berarti ada dua jari yang diangkat oleh user.</p>
4		<p>Model dapat memprediksi gestur yang menampilkan empat jari dengan baik.</p> <p>Tiga titik merah menandakan terdapat tiga defect pada gambar, yang berarti ada empat jari yang diangkat oleh user.</p>
5		<p>Model dapat memprediksi gestur yang menampilkan lima jari dengan baik.</p> <p>Empat titik merah menandakan terdapat empat defect pada gambar, yang berarti ada lima jari yang diangkat oleh user.</p>

No	Gambar	Keterangan
		
6	<p>Cannot read</p> 	<p>Model tidak dapat memprediksi gestur kepalan tangan.</p> <p>Hal ini dikarenakan tidak adanya defects pada kepalan tangan.</p>
7	<p>Cannot read</p> 	<p>Model tidak dapat memprediksi gestur yang menampilkan satu jari.</p> <p>Hal ini dikarenakan tidak adanya defects pada gestur tersebut.</p>
8		<p>Model tidak dapat memprediksi gestur dengan background yang tidak polos.</p> <p>Hal ini dikarenakan mesin akan mencari contour semua objek pada gambar, dan mengambil contour terbesar, sehingga background yang 'kotor' dapat mempengaruhi hal tersebut.</p>

CONCLUSION

Algoritma deteksi defects pada convex hull dapat menyelesaikan problem deteksi gesture tangan, yaitu menentukan berapa jumlah jari yang diberikan di depan kamera pada aplikasi. Algoritma ini dipastikan jauh lebih cepat dan mudah untuk dikembangkan modelnya, daripada menggunakan model machine learning yang membutuhkan training set dan waktu pengembangan model. Oleh karena

itu, sumber daya yang dibutuhkan untuk menjalankan model ini juga dapat dipastikan lebih rendah. Sayangnya, ada beberapa keterbatasan pada model defects convex hull. Selain hanya dapat mendeteksi jumlah jari yang diangkat, metode defects convex hull tidak dapat mengenali gestur tangan tanpa defects, seperti pada gestur kepalan tangan maupun gestur angka satu. Model terbatas pada gambar tangan yang memiliki cekung di antara jari (defects). Selain itu, dikarenakan model akan membuat contour secara terus menerus, model hanya dapat mendeteksi tangan pada window yang dibatasi, dan juga pengguna harus memastikan bahwa background dari tangan tidak akan mengganggu pencarian contour pada image. Model ini tidak bisa mencari letak tangan secara otomatis, maupun membedakan tangan dengan backgroundnya, karena model hanya bergantung pada perhitungan sudut. Oleh karena itu, metode convex hull ini dapat dikembangkan seiring dengan bantuan machine learning, sehingga dapat menemukan letak tangan dan membedakan background dengan sumber daya yang lebih kecil pada kasus tertentu.

REFERENCES

- [1] Szeliski, R. (2022). Computer vision: algorithms and applications. Springer Nature.
- [2] Zhang, B. (2010, July). Computer vision vs. human vision. 9th IEEE International Conference on Cognitive Informatics (ICCI'10) (pp. 3-3). IEEE.
- [3] Shrimali, K. (2021, May 04). Convex hull using opencv in C++ and python. Retrieved November 23, 2022, from <https://learnopencv.com/convex-hull-using-opencv-in-python-and-c/>
- [4] Ganapathyraju, S. (2013). Hand gesture recognition using convexity hull defects to control an industrial robot. 2013 3rd International Conference on Instrumentation Control and Automation (ICA). doi:10.1109/ica.2013.6734047
- [5] Yuliana, N., & Wardani, K. R. R. (2016). Metode Convex Hull dan Convexity Defects untuk Pengenalan Isyarat Tangan. Jurnal Telematika, 11(2), 8.
- [6] Wulanningrum, R., Setyawan, D., & Kasih, P. (2022). Implementasi Metode Convex Hull pada Gesture Tangan. Joutica: Journal of Informatic Unisla, 7(1), 519-524.
- [7] Zeger, I., Grgic, S., Vukovic, J., & Sisul, G. (2021). Grayscale image colorization methods: Overview and evaluation. IEEE Access, 9, 113326–113346. <https://doi.org/10.1109/access.2021.3104515>
- [8] Wang, M., Zheng, S., Li, X., & Qin, X. (2014). A new image denoising method based on Gaussian filter. 2014 International Conference on Information Science, Electronics and Electrical Engineering. <https://doi.org/10.1109/infosee.2014.6948089>
- [9] Saddami, K., Munadi, K., Away, Y., & Arnia, F. (2019). Improvement of binarization performance using local Otsu thresholding. International Journal of Electrical and Computer Engineering (IJECE), 9(1), 264. <https://doi.org/10.11591/ijece.v9i1.pp264-272>
- [10] Find and draw contours using opencv: Python. GeeksforGeeks. (2019, April 29). Retrieved November 26, 2022, from <https://www.geeksforgeeks.org/find-and-draw-contours-using-opencv-python/>
- [11] Sugihara, K. (1994). Robust gift wrapping for the three-dimensional convex hull. Journal of Computer and System Sciences, 49(2), 391-407.