IMAGE PROCESSING DETEKSI ANGKA PADA JARI MENGGUNAKAN LIBRARY MEDIAPIPE DAN DETEKSI CANNY

ABSTRAK

PENDAHULUAN

Pengolahan citra digital yaitu bidang ilmu yang mempelajari mengenai bagaimana suatu citra itu dibentuk, diolah dan dianalisis sehingga dapat menghasilkan sebuah informasi yang bisa dipahami oleh manusia.Dapat pula di jelasakan bahwa pengolahan citra digital adalah manipulasi dan interprestasi digital dari citra dengan bantuan komputer. Citra sendiri merupakan fungsi dari intensitas cahaya yang direpresentasikan ke dalam bidang 2 dimensi.

Pengolahan Citra Digital Menurut Suhandy (2003):

Pengolahan citra digital adalah sebuah teknologi visual yang dipakai untuk mengamati dan menganalisis sebuah objek tanpa berhubungan secara langsung dengan objek yang diamati itu. Teknologi ini bisa dipakai untuk mengevaluasi mutu suatu produk tanpa merusak produk itu sendiri.

MediaPipe adalah Kerangka Kerja untuk membangun alur kerja pembelajaran mesin untuk memproses data deret waktu seperti video, audio, dll. Kerangka Kerja lintas platform ini berfungsi pada Desktop/Server, Android, iOS, dan perangkat tertanam seperti Raspberry Pi dan Jetson Nano.

Deteksi tepi Canny adalah teknik yang digunakan dalam pemrosesan gambar untuk menemukan tepi pada gambar. Dinamai sesuai dengan penciptanya, John Canny, yang mengembangkan teknik ini pada tahun 1986. Algoritma deteksi tepi Canny digunakan untuk menemukan tepi pada gambar dengan mencari maxima lokal pada turunan pertama gambar.

RUMUSAN MASALAH

1. Bagaimana cara mengintegrasikan library Mediapipe dan algoritma deteksi tepi Canny untuk mendeteksi angka yang ditunjukkan oleh jari secara akurat?
2. Bagaimana sistem ini dapat diimplementasikan secara real-time dengan mempertimbangkan berbagai kondisi lingkungan?

TUJUAN

1. Mengembangkan sistem real-time untuk mendeteksi angka pada jari menggunakan Mediapipe dan algoritma deteksi tepi Canny.
2. Mengevaluasi akurasi dan keandalan sistem dalam berbagai kondisi lingkungan.

Alasan pengambilan judul

METODE

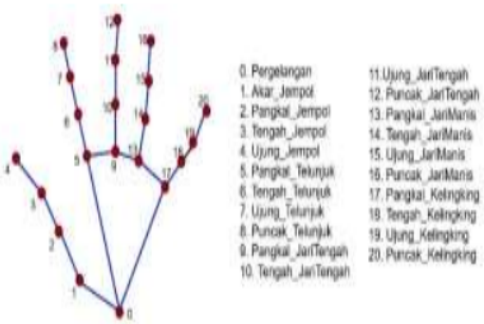
1. MediaPipe

Pengenalan gerakan tangan dengan MediaPipe adalah suatu teknik untuk mengubah input video dengan mempertahankan informasi penting tentang gerakan tangan, sementara gaya atau tekstur visual dari video tersebut diubah [10]. Informasi penting tersebut terkait dengan pengenalan gestur tangan, baik oleh manusia maupun mesin, dan dapat

dimanfaatkan dalam berbagai aplikasi seperti interaksi manusia dan komputer.

A. Pengumpulan Data

Pemilihan dataset merupakan langkah awal yang penting dalam penelitian ini. Dataset yang digunakan merupakan hasil rekaman gesture tangan dengan menggunakan MediaPipe. MediaPipe menyediakan model untuk mendeteksi landmark tangan, dimana model ini mendeteksi posisi tangan dengan 21 keypoint. Keypoint inilah yang disimpan sebagai dataset untuk melatih model agar bisa mendeteksi gesture tangan apa yang sedang dibuat. Ilustrasi keypoint dalam mediapipe ditunjukan pada Gambar 1.



Gambar 1 : *Keypoint Hand Gesture MediaPipe*

Proses pengumpulan data melibatkan beberapa variasi gesture. Setiap gesture disimpan keypoint-nya, masing-masing gestur direkam untuk tangan kanan dan tangan kiri. Gestur yang dipilih ditunjukan pada Gambar2.

Gambar 2 : *Sampel Gesture yang Digunakan*

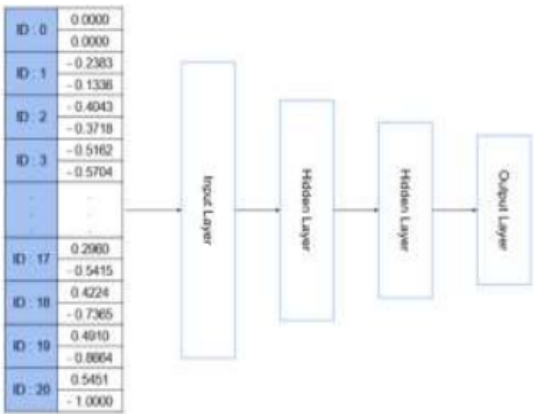
Saat pengumpulan data, setiap gestur diberi label dari 0 sampai 3 sebagai index untuk kemudian diberi label nama gesturnya yaitu Open, Close, Pointer, dan Peace. Total dataset yang dikumpulkan dan digunakan untuk klasifikasi gestur ini sebanyak 6226 baris data.

B. Model

Dalam penelitian ini diperlukan model untuk mengenali gesture dengan label yang sesuai dengan yang sudah didefinisikan. Diagram blok pada Gambar 3 menjelaskan proses keseluruhan pengenalan gestur tangan dalam penelitian ini. Proses dimulai dari pengambilan data gerakan tangan menggunakan MediaPipe, diikuti oleh ekstraksi fitur berupa koordinat keypoint, pelatihan model neural network, hingga klasifikasi gestur berdasarkan probabilitas output.

Gambar 3 : *Proses Pattern Recognition*

Model untuk mengenali gestur dan labelnya dibuat dengan neural network yang ditunjukan pada Gambar 4. Ketika pengumpulan data, satu gesture data yang disimpan itu merupakan koordinat dari semua keypoint dari keypoint 0 hingga 20. Data dari keypoint ini yang dijadikan sebagai input untuk train model.



Gambar 4 : *Struktur Model*

Dataset yang digunakan untuk training dan menguji model merupakan data keypoint dari gestur yang didapatkan dari MediaPipe yang disimpan dalam csv saat pengumpulan data. Data train yang digunakan sebanyak 75% dan data test sebanyak 25% dari dataset yang ada. Proses pelatihan model dimulai dengan membangun arsitektur jaringan saraf menggunakan TensorFlow Keras. Model dibuat dalam bentuk sekuensial dengan beberapa lapisan untuk memproses input data. Lapisan pertama menerima vektor dengan dimensi 21×2, yang merupakan data koordinat x dan y dari 21 keypointtangan. Untuk mencegah overfitting, dua lapisan dropout diterapkan dengan probabilitas masing -masing 0,2 dan 0,4. Model juga memiliki dua lapisan dense (fully connected) dengan masing-masing 20 dan 10 neuron, menggunakan fungsi aktivasi ReLU untuk menangkap pola non-linear. Lapisan keluaran menggunakan fungsi aktivasi softmax dengan jumlah neuron yang sesuai dengan jumlah kelas yaitu 4, untuk mengklasifikasikan gestur tangan berdasarkan probabilitas. Model dikompilasi menggunakan Adam optimizer, yang merupakan algoritma pembaruan bobot yang efisien, serta fungsi loss sparse categorical crossentropy, karena data label diberikan dalam format integer. Metrik accuracy dipilih untuk memantau performa model selama pelatihan. Proses pelatihan dilakukan hingga maksimal 1000 epoch, dengan dua callback yang digunakan dalam proses ini adalah ModelCheckpoint untuk menyimpan bobot model terbaik secara otomatis setelah setiap epoch, dan EarlyStopping dengan toleransi 20 epoch untuk menghentikan pelatihan jika performa validasi tidak meningkat, sehingga menghindari overfitting. Dengan konfigurasi ini, model dapat dilatih secara optimal untuk mengenali gestur tangan berdasarkan dataset

yang tersedia.

2. DETEKSI CANNY

Operator Canny, yang dikemukakan oleh John Canny pada tahun 1986, terkenal sebagai operator deteksi tepi yang optimal. Algoritma ini memberikan tingkat kesalahan yang rendah, melokalisasi titik-titik tepi (jarak piksel-piksel tepi yang ditemukan deteksi dan tepi yang sesungguhnya sangat pendek), dan hanya memberikan satu tanggapan untuk satu tepi. Menurut Green (2012), terdapat enam langkah yang dilakukan untuk mengimplementasikan deteksi tepi Canny. Keenam langkah tersebut dijabarkan berikut ini :

Pertama, dilakukan penipisan terhadap citra dengan tujuan untuk menghilangkan derau. Hal ini dapat dilakukan dengan menggunakan filter Gaussian dengan cadar sederhana. Cadar yang digunakan berukuran jauh lebih kecil daripada ukuran citra. Contoh ditunjukkan pada Gambar 1 :

1 / 115

2	4	5	4	2
4	9	12	9	4
5	12	15	12	5
4	9	12	9	4
2	4	5	4	2

Setelah penghalusan gambar terhadap derau dilakukan, dilakukan proses untuk mendapatkan kekuatan tepi (edge strength). Hal ini dilakukan dengan menggunakan operator Gaussian. Selanjutnya, gradien citra dapat dihitung melalui rumus :

$$|G| = |G_x| + |G_y|$$

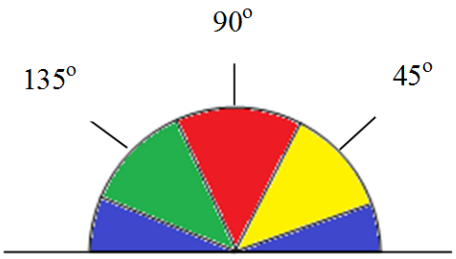
Langkah ketiga berupa penghitungan arah tepi. Rumus yang digunakan untuk keperluan ini :

$$theta = tan^{-1}(G_y, G_x)$$

Setelah arah tepi diperoleh, perlu menghubungkan antara arah tepi dengan sebuah arah yang dapat dilacak dari citra. Sebagai contoh, terdapat susunan piksel berukuran 5 x 5 seperti terlihat pada Gambar 2. Dengan melihat piksel “a” tampak bahwa a hanya memiliki 4 arah berupa 0 derajat, 45 derajat, 90 derajat, dan 135 derajat.

X	X	X	X	X
X	X	X	X	X
X	X	**a**	X	X
X	X	X	X	X
X	X	X	X	X

Selanjutnya, arah tepi yang diperoleh akan dimasukkan ke dalam salah satu kategori dari keempat arah tadi berdasarkan area yang tertera pada Gambar 3. Berikut adalah aturan konversi yang berlaku :



- Semua arah tepi yang berkisar antara 0 dan 22,5 serta 157,5 dan 180 derajat (warna biru) diubah menjadi 0 derajat.
- Semua arah tepi yang berkisar

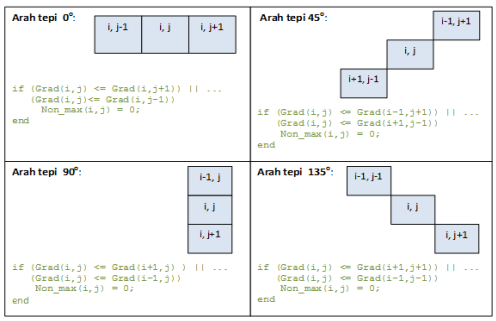
antara 22,5 dan 67,5 derajat (warna kuning) diubah menjadi 45 derajat.

- Semua arah tepi yang berkisar antara 67,5 dan 112,5 derajat (warna merah) diubah menjadi 90 derajat.
- Semua arah tepi yang berkisar antara 112,5 dan 157,5 derajat (warna hijau) diubah menjadi 135 derajat.

Setelah arah tepi diperoleh, penghilangan non-maksimum dilaksanakan dilaksanakan. Penghilangan non-maksimum dilakukan di sepanjang tepi pada arah tepi dan menghilangkan piksel-piksel (piksel diatur menjadi 0) yang tidak dianggap sebagai tepi. Dengan cara seperti itu, diperoleh tepi yang tipis.

Langkah keenam berupa proses yang disebut *hysteresis*. Proses ini menghilangkan garis-garis yang seperti terputus-putus pada tepi objek. Caranya adalah dengan menggunakan dua ambang T1 dan T2. Lalu, semua piksel citra yang bernilai lebih besar daripada T1 dianggap sebagai piksel tepi. Selanjutnya, semua piksel yang terhubung dengan piksel tersebut dan memiliki nilai lebih besar dari T2 juga dianggap sebagai piksel tepi.

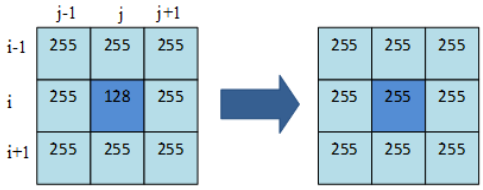
Bagian penting yang perlu dijelaskan adalah penghilangan non-maksimum dan peng-ambangan histeresis. Penghilangan non-maksimum dilakukan dengan mula-mula menyalin isi larik Grad (yang berisi besaran gradien) ke Non_max. Selanjutnya, penghilangan non-maksimum dilaksanakan dengan memperhatikan dua titik tetangga yang terletak pada arah tepi (yang tersimpan dalam Theta). Misalnya, arah tepi adalah 0. Apabila titik yang menjadi perhatian mempunyai koordinat (r, c), dua titik tetangga berupa (r, c-1) dan (r, c+1). Apabila gradien titik perhatian lebih besar daripada gradien kedua tetangga, nilainya akan dipertahankan. Sebaliknya, jika nilai titik perhatian lebih kecil daripada nilai salah satu atau kedua gradien tetangga, nilainya akan diabaikan (diubah menjadi nol).



Peng-ambangan histeresis dilakukan dengan melibatkan dua ambang T1 (ambang bawah) dan ambang T2 (ambang atas). Nilai yang kurang dari T1 akan diubah menjadi hitam (nilai 0) dan nilai yang lebih dari T2 diubah menjadi putih (nilai 255). Lalu, bagaimana nilai yang lebih dari atau sama dengan T1 tetapi kurang dari T2? Oleh karena itu, untuk sementara nilai pada posisi seperti itu diberi nilai 128, yang menyatakan nilai abu-abu atau belum jelas, akan dijadikan 0 atau 255.

Selanjutnya, dilakukan pengujian untuk mendapatkan kondisi seperti tercantum pada Gambar 5. Apabila kondisi seperti itu terpenuhi, angka 128 diubah menjadi 255. Proses pengujian seperti itu dilakukan sampai tidak ada lagi perubahan dari nilai 128 menjadi 255. Tahap selanjutnya, semua piksel yang bernilai 128 yang

tersisa diubah menjadi nol.



HASIL DAN PEMBAHASAN

Daftar Pustaka

https://www.temukanpengertian.com/2013/08/pengertian-pengolahan-citra-digital.html
https://learnopencv.com/introduction-to-mediapipe/
https://medium.com/@andreryan33/memahami-metode-deteksi-tepi-canny-menggunakan-bahasa-pemrograman-python-5413192eb0e7
https://www.kitainformatika.com/2015/07/segmentasi-citra-deteksi-tepi_79.html
https://jurnal.unsur.ac.id/mjinformatika