IMPLEMENTASI HAND GESTURE RECOGNITION MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK PADA GAME

Mukhammad Zainul Musyafa', Nurlaily Vendyansyah, Febriana Santi Wahyuni
Teknik Informatika, Institut Teknologi Nasional Malang
Jalan Raya Karanglo km 2 Malang, Indonesia
2118050@scholar.itn.ac.id

ABSTRAK

Industri game terus berkembang seiring kemajuan teknologi, salah satunya melalui penerapan hand gesture recognition yang memanfaatkan visi komputer untuk mengenali gerakan tangan sebagai input permainan. Di sisi lain, penggunaan kontroler konvensional dalam jangka panjang cenderung membatasi pergerakan fisik pemain dan berpotensi menimbulkan masalah kesehatan seperti gangguan otot dan postur tubuh. Tujuan penelitian ini untuk mengimplementasikan sistem pengenalan gestur menggunakan CNN sebagai kontrol game. Sehingga didapatkan solusi untuk mengembangkan sistem *hand gesture recognition* berbasis CNN sebagai alternatif kontrol dalam permainan digital. Hasil implementasi menunjukkan bahwa model memiliki akurasi 70% yang cukup mampu mengenali gestur dasar. Berdasarkan pengujian, model dapat mendeteksi tangan dengan baik pada tingkat kecerahan 50–170 piksel, dengan respon cepat 40–50 ms pada intensitas cahaya di atas 120 piksel. Namun, pencahayaan rendah dan pergerakan tangan yang cepat menyebabkan keterlambatan deteksi. Implementasi game berjalan optimal pada perangkat dengan RAM minimal 8GB dan sistem operasi Windows 10 ke atas. Dan Berdasarkan pengujian responden didapat 55% indeks atau sistem hand gesture pada game tersebut dapat difungsikan secara cukup baik.

Kata kunci : *convolutional neural network, game, hand landmark, hand gesture recognition, machine learning*

1. PENDAHULUAN

Industri game terus mengalami perkembangan pesat dengan berbagai inovasi dalam teknologi interaksi, termasuk metode pengendalian yang lebih natural dan intuitif. Salah satu inovasi yang menjanjikan adalah penggunaan *hand gesture recognition* berbasis convolutional *neural network* (CNN), yang memungkinkan pemain mengontrol game hanya dengan gerakan tangan tanpa memerlukan perangkat keras tambahan. Teknologi ini menawarkan pendekatan yang fleksibel dibandingkan penggunaan kontroler konvensional.

Kontroler tradisional seperti *keyboard*, *mouse*, atau *joystick* masih menjadi standar utama dalam pengendalian game. Namun, penggunaan perangkat tersebut sering kali menyebabkan pemain duduk dalam waktu lama di depan layar, yang dapat meningkatkan risiko gangguan kesehatan seperti nyeri otot, kelelahan mata, hingga gangguan postur tubuh. Menurut artikel yang dipublikasikan oleh Sehat Negeriku (Kementerian Kesehatan RI), game daring yang dimainkan dalam waktu lama tanpa gerakan aktif dapat menyebabkan berbagai masalah kesehatan, termasuk gangguan sirkulasi darah. [1]

Sebagai alternatif terhadap kontroler tradisional, sistem hand gesture recognition memungkinkan interaksi yang lebih aktif secara fisik. Dengan mengenali gerakan tangan sebagai input perintah, pemain dapat melakukan navigasi, menyerang, atau berinteraksi dengan elemen dalam game secara lebih alami. Selain memberikan pengalaman bermain yang lebih imersif, pendekatan ini juga berpotensi mengurangi risiko cedera akibat posisi duduk yang berkepanjangan.

Salah satu metode yang efektif dalam pengenalan gestur adalah *convolutional neural network* (CNN), yang mampu mengenali pola visual dengan akurasi tinggi. Teknologi ini memungkinkan sistem untuk membedakan berbagai gestur tangan secara *real-time* dan menerjemahkannya menjadi perintah dalam game, menggantikan fungsi tombol fisik.

Oleh karena itu, penelitian ini akan fokus pada implementasi hand gesture recognition berbasis CNN sebagai sistem kontrol pada game *puzzle and shooter* yang interaktif dan inovatif. Tujuan utamanya adalah menghadirkan solusi pengendalian yang lebih terjangkau, dan ergonomis, serta memberikan kontribusi terhadap pengembangan pengalaman bermain game yang lebih alami dan sehat bagi pengguna [9].

2. TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Pengembangan Aplikasi Pengenalan Bahasa Isyarat Abjad SIBI Menggunakan Metode *Convolutional Neural Network* (CNN) bertujuan untuk mengembangkan aplikasi berbasis web yang mampu mengenali peragaan bahasa isyarat abjad SIBI secara real-time. Aplikasi ini diharapkan dapat membantu proses pembelajaran bahasa isyarat bagi penyandang tunarungu. Dapat disimpulkan bahwa aplikasi yang dibuat dengan metode CNN mampu melakukan klasifikasi peragaan bahasa isyarat abjad SIBI dengan akurasi sebesar 80,76%, sehingga dapat meningkatkan efektivitas pembelajaran dan pemahaman bahasa isyarat bagi penyandang tunarungu. [1]

Penerapan *Hand Gesture Recognition* sebagai Media Kontrol Presentasi Aplikasi *PowerPoint*

bertujuan untuk mempermudah kontrol aplikasi *PowerPoint* dengan menggunakan pengenalan gestur tangan sebagai pengganti perangkat kendali seperti keyboard dan mouse [7]. Solusi yang diusulkan melibatkan penggunaan kamera sebagai perangkat input real-time, dengan memanfaatkan bahasa pemrograman Python dan pustaka seperti OpenCV untuk pemrosesan video, *Mediapipe* untuk mendapatkan landmark tangan, dan *CVZone* untuk klasifikasi gestur tangan. Hasil pengujian menunjukkan bahwa sistem mampu merespons perintah gestur dengan waktu rata-rata sekitar 2 detik, baik untuk berpindah slide maupun mengatur mode layar penuh. [2]

Penerapan sistem pengenalan bentuk tangan menggunakan *Convolutional Neural Network* (CNN) dibuat dengan tujuan mengurangi tingkat komputasi dan upaya persiapan perangkat yang biasanya merepotkan pada sistem serupa. Sistem ini menerima *input* berupa citra tangan dan menghasilkan *output* berupa label dari gestur yang dikenali. Metodologi yang diterapkan melibatkan proses pelatihan model hingga tahap pengujian, di mana model CNN yang dibangun berhasil mencapai akurasi klasifikasi sebesar 88% berdasarkan pengujian terhadap 2142 citra, dan performanya diukur menggunakan *confusion matrix*. Hasil ini menunjukkan bahwa pendekatan ini cukup efektif dan efisien untuk aplikasi pengenalan gestur tangan berbasis citra. [3]

2.2. Hand gesture

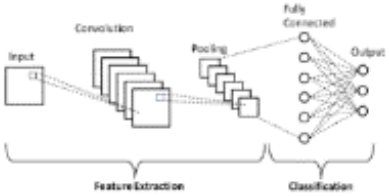
Hand gesture recognition merupakan proses identifikasi gerakan tangan manusia oleh komputer yang memungkinkan interaksi tanpa kontak fisik antara manusia serta mesin. Sistem ini biasa digunakan dalam bidang *Human Computer Interaction*, realitas virtual, serta alat bantu komunikasi bagi penyandang disabilitas. Pada umumnya pembuatan *hand gesture* akan melibatkan beberapa metode *Neural Network* seperti *Recurrent Neural Network* sebagai pendeteksi posisi tangan serta *Convolutional Neural Network* sebagai pengklasifikasi bentuk *hand gesture* [4].

2.3. Hand Landmark

Gambar 1. *Hand landmark*

Hand landmark merupakan titik-titik kunci pada tangan manusia yang digunakan untuk mengenali dan menganalisis berbagai gestur tangan. Pada gambar 1 menunjukkan model bekerja dengan mendeteksi 21 titik penanda yang tersebar di berbagai bagian tangan, seperti ujung jari, buku-buku jari, dan pergelangan tangan. Dengan adanya titik-titik ini, sistem dapat memahami pergerakan serta posisi tangan dalam ruang tiga dimensi, memungkinkan aplikasi seperti pelacakan gerakan, interaksi berbasis gestur, serta pengenalan bahasa isyarat. Terdapat model *hand landmark* yang biasa digunakan yaitu seperti yang dimiliki *Google mediapipe* yang biasa digunakan dalam pembuatan *hand gesture.*

2.4. Convolutional Neural Network



Gambar 2. Proses *Convolutional Neural Network*

Convolutional Neural Network (CNN) merupakan algoritma *deep learning* yang digunakan untuk mengolah data secara 2 dimensi [8]. Selain itu *Convolutional Neural Network* digunakan untuk membedakan data yang berlabel seperti menggunakan metode *supervised learning* [4] . *Long Short Term Memory (LSTM) Network* merupakan algoritma yang memanfaatkan sejumlah slot memori di luar proses perulangan untuk menyimpan potongan-potongan representasi dari input. Algoritma ini juga menggunakan memori untuk memperkuat kemampuan jaringan berulang dalam mengingat informasi serta, yang terpenting, dalam mengidentifikasi hubungan antar data. Sementara itu, pada Gambar 2 ditunjukkan bahwa *Convolutional Neural Network* (CNN) memiliki beberapa tahap atau lapisan yang saling terstruktur.[5]

2.5. Convolutional Layer

Convolution Layer merupakan proses utama pada CNN. Konvolusi meripakan mengaplikasikan sebuah fungsi pada *output* fungsi lain secara berulang, dengan kata lain konvolusi berarti mengaplikasikan sebuah kernel dengan ukuran yang lebih kecil pada sebuah citra yang berguna untuk memperoleh suatu piksel yang didasarkan pada nilai piksel itu sendiri dan nilai tetangganya dengan rumus seperti pada persamaan (1).

$$O = \sum_{i=1}^{m} . \sum_{j=1}^{n} . I_{i,j} . K_{i,j} + b \quad (1)$$

Keterangan :

O, adalah *output* dari operasi konvolusi.

I adalah citra masukan dengan ukuran m x n.

K, adalah *filter* konvolusi yang digunakan untuk mengekstraksi fitur

b, adalah bias yang membantu menyesuaikan nilai *output*

i, indeks baris dalam matriks citra masukan I dan kernel K

j, indeks kolom dalam matriks citra masukan I dan kernel K

2.6. Maxpooling Layer

Subsampling merupakan proses mereduksi ukuran sebuah data citra. *Subsampling* juga biasa digunakan untuk meningkatkan invariasi posisi fitur. Dalam CNN *subsampling* yang biasa digunakan adalah *Max Pooling*. Cara kerja fungsi ini adalah dengan membagi *output* dari *convolution layer* dan mengambil nilai maksium dari jendela *pooling* yang biasanya berukuran 2x2. dengan rumus *max pooling* seperti pada persamaan (2).

$$O = max\ (I_{i:i+2,\ j:j+2}) \quad (2)$$

2.7. Flatten Layer

Flatten layer merupakan lapisan konvolusi yang digunakan untuk menghubungkan ke lapisan *fully connected* atau *dense layer* dengan cara mengubah nilainya menjadi *vector* 1 dimensi.

2.8. Dense Layer

Dense Layer atau *Fully-Connected Layer* merupakan sebuah layer yang digunakan untuk menghubungkan setiap satuan input ke setiap satuan *output*. Pada lapisan ini setiap *neuron* akan melakukan operasi perkalian matriks vektor, dengan rumus perhitungan *neuron* seperti pada persamaan (3).

$$O = f\ (W\ X + B) \quad (3)$$

Keterangan :
O, adalah *output* dari *neuron*
W, adalah bobot yang dipelajari selama pelatihan
X, adalah *input* dari lapisan sebelumnya
B, adalah bias untuk mengatur nilai aktivasi
f, adalah fungsi aktivasi ReLU yang digunakan untuk mengintroduksi *non-linearitas*

2.9. Output Layer

Output layer atau *Activation layer* merupakan lapisan terakhir yang akan menghasilkan prediksi akhir berdasarkan fitur yang telah diproses pada lapisan sebelumnya. Lapisan ini biasanya akan menggunakan fungsi aktivasi untuk mengubah *output* menjadi nilai yang dapat dipahami. Terdapat beberapa fungsi yang biasa digunakan meliputi *sigmoid* (untuk klasifikasi biner), *softmax* (untuk klasifiksi *multi class*), *Linear* (untuk regresi), dan *ReLU* (untuk unit linear yang diperbaiki). seperti pada persamaan (4).

$$P(y = i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \quad (4)$$

Keterangan :
P(y=i), adalah probabilitas bahwa *input* termasuk dalam kelas *i*

2.10. Puzzle Game

Puzzle game merupakan sebuah game yang mengharuskan pemain untuk mencari jalan keluar agar dapat menyelesaikan permainan [6]. Pada penelitian yang dilakukan game akan dibuat menjadi 2 jenis permainan sebagai bentuk implementasi *hand gesture* pada. Permainan sepenuhnya akan mengimplementasikan gestur menunjuk sebagai kontroler untuk menyentuh tombol. Pada game pertama merupakan game yang akan mengimplementasikan gerakan mengambil dan pemain akan diminta untuk mengambil objek yang ada hingga skor tertentu. Pada game kedua merupakan game yang akan mengimplementasikan gearakan menembak dan pemain akan diminta untuk mengalahkan musuh dengan pistol yang hanya dapat digunakan ketika pemain menggunakan gestur menembak.

3. METODE PENELITIAN

3.1. Analisis Kebutuhan

Analisis dilakukan untuk memahami struktur data masukan, parameter yang digunakan dalam pelatihan model, serta bagaimana pengolahan data dilakukan agar dapat diterapkan dalam lingkungan game. Terdapat beberapa variabel yang digunakan meliputi :

a. Data_dir, *inputan* untuk tempat penyimpanan dataset
b. Datagen, *inputan* untuk mengelola dataset pada data_dir
c. *Train_generator*, inputan untuk dataset yang akan dilatih
d. *Test_generator*, inputan untuk dataset yang akan ditesting
e. *History*, *output* hasil pelatihan
f. *Input_shape*, ukuran gambar yang akan diproses bisa dengan gambar rgb
g. *Maxpooling2D*, *layer* yang mengambil nilai maksium dalam ukuran kernel tertentu
h. *Conv2D*, *layer* untuk mengerkstrak fitur pada gambar
i. *Flatten*, membuat *output* dari 3D menjadi 1D
j. *Dense*, menghasilkan *output* akhir yang dapat ditampilkan pada variabel *history*

3.2. Prancangan Model CNN

Arsitektur CNN yang digunakan terdiri dari beberapa lapisan utama yang bertujuan untuk mengekstraksi fitur dari gambar *inputan* dan melakukan klasifikasi berdasarkan pola yang terdeteksi [10] . Model yang dibuat memiliki beberapa lapisan ar cnn yang meliputi seperti berikut.

a. *Layer* Konvolusi
Ukuran kernel 3x3, jumlah *filter* 32 dan 64, fungsi aktivasi ReLU, digunakan untuk mengekstrasi fitur pada citra masukan, dengan rumus aktivasi ReLU seperti pada persamaan (5).

$$Conv2D = \sum_{i=1}^{m} . \sum_{j=1}^{n} . Input\_shape_{i,j} \ . Conv2D_{i,j} + bias \quad (5)$$

Keterangan :
I dan j, indeks baris dalam matriks citra masukan *I* dan kernel *K*

b. *Layer Maxpooling*
Ukuran *pooling* 2x2, digunakan untuk mengurangi ukuran citra masukan, dengan rumus seperti pada persamaan (6).

$$Maxooling2D = max\,(Conv2D_{i:i+2,\ j:j+2}) \quad (6)$$

c. *Flatten Layer*
Mengubah matriks hasil lapisan sebelumnya menjadi vektor 1 dimensi

d. *Dense Layer*
Jumlah *neuron* 128, fungsi aktivasi ReLU, digunakan untuk menghubungkan fitur yang telah diekstraksi oleh konvolusi menjadi representasi *vector*, dengan rumus perhitungan *neuron* seperti pada persamaan (7).

$$Dense = f\,(Dense_{Weight}\,.Flatten + bias) \quad (7)$$

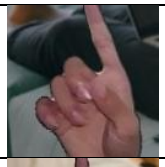
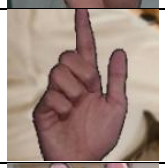
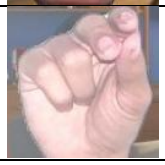
e. *Output Layer*
Jumlah kelas yang diklasifikasi 4 jenis gestur, fungsi aktivasi *Softmax*, digunakan untuk menghasilkan probabilitas untuk setiap kelas gestur yang dikenali oleh model dengan rumus seperti persamaan (8).

$$P(Dense_{\ Output} = i) = \frac{e^{Dense_i}}{\sum_j e^{Dense_j}} \quad (8)$$

3.3. Bentuk Gestur

Model yang akan dibuat akan mengklasifikasikan beberapa bentuk *hand gesture* yang terdiri dari 4 jenis, yaitu gestur menunjuk, gestur menembak 1, gestur menembak 2, serta gestur mengambil.

Tabel 1. Macam *Hand Gestur*

Gestur	Gambar
Gestur menunjuk	
Gestur Menembak 1	
Gestur Menembak 2	
Gestur Mengambil	

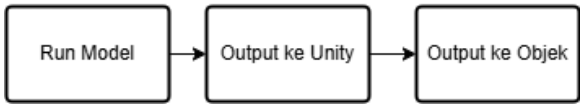
3.4. Labeling

Gambar 2. *Hand landmark*

Setelah didapatkan data *hand gesture* yang dibutuhkan kemudian dilakukan pelabelan gambar sesuai dengan pada gambar 3 *hand lanmark* yang telah ditentukan. Terdapat 21 titik x, 21 titik y, serta 21 titik z.

4. HASIL DAN PENGUJIAN

4.1. Proses integrasi



Gambar 3. Proses Integrasi sebagai kontrol game

Ketika game dijalankan, proses pertama adalah menjalankan model CNN yang telah dilatih sebelumnya. Model ini menerima input berupa data visual dari kamera, kemudian memprosesnya untuk mendeteksi posisi tangan dan mengenali jenis gestur. *Output* yang dihasilkan terdiri dari 64 nilai, yang mencakup 21 koordinat x, 21 koordinat y, 21 koordinat z, dan 1 label klasifikasi gestur. Model ini dijalankan secara *real-time* menggunakan script

Python yang berjalan paralel dengan aplikasi game.

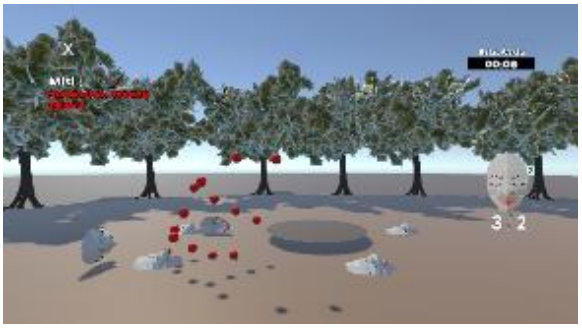
Gambar 4. Implementasi pada game

Output yang diterima Unity digunakan untuk memanipulasi objek tertentu di dalam game. Misalnya, jika output klasifikasi menunjukkan gestur "menunjuk", maka objek target akan diberi tanda atau dipindahkan. Jika gestur "mengambil" terdeteksi, maka objek dalam game dapat berubah status atau berpindah posisi. Dengan demikian, data landmark dan klasifikasi gestur secara langsung memengaruhi perilaku objek dalam game secara dinamis.

4.2. Hasil Implementasi pada Game

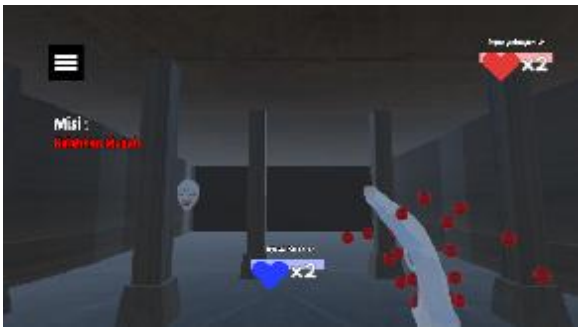
Gambar 5. Implementasi Gestur Menunjuk

Pada gambar 5 model mengimplementasikan gestur menunjuk sebagai cara untuk menekan tombol



Gambar 6. Implementasi Gestur Mengambil

Pada gambar 6 model mengimplementasikan gestur mengambil sebagai cara untuk mengambil objek yang ada pada game



Gambar 7. Implementasi Gestur Menembak

Pada gambar 7 model mengimplementasikan gestur menembak 1 dan gestur menembak 2 sebagai cara untuk menembak musuh atau objek pada game.

4.3. Hasil Model CNN

Pada model memiliki jumlah data yang digunakan sebanyak 2000 data yang terdiri dari 1600 data latih dan 400 gambar validasi. Pengulangan atau jumlah pelatihan (*epoch*) yang dilakukan sebanyak 20 kali.

Gambar 8. Kurva akurasi model

Pada Gambar 8 menunjukkan hasil akurasi model yang memiliki hasil yang cukup baik dengan memiliki akurasi 70% dari 20 kali pengulangan pelatihan model. Akurasi model dihitung berdasarkan persamaan (9).

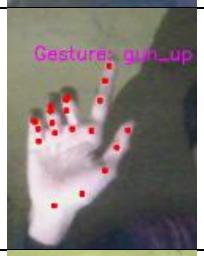
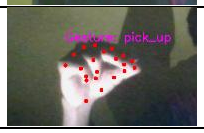
$$\text{Akurasi} = \frac{jumlah\ prediksi\ benar}{jumlah\ total\ data\ yang\ di\ evaluasi} \quad (9)$$

Akurasi pelatihan dapat terus meningkat secara konsisten meskipun terdapat beberapa kali mengalami penurunan yang menandakan model dapat mempelajari data pelatihan dengan baik. Namun perbedaan antara akurasi dan dengan validasi akurasi masih cukup jauh yang bisa menandakan model masih mengalami *overfitting* atau model masih terlalu mengikuti data latihan dan kurang mampu menerima data baru.

4.4. Pengujian gestur

Pada tahap ini dilakukan pengujian terhadap model yang telah dikembangkan untuk memastikan bahwa model mampu mengenali setiap jenis gestur. Pengujian dilakukan dengan 4 jenis gestur yang setiap gesturn nantinya akan dapat dilihat apakah sesuai dengan *ouput* yang seharusnya

Tabel 2. Pengujian *Hand Gesture*

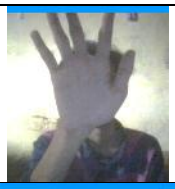
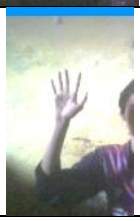
Jenis	Gestur	Persentase keberhasilan
menunjuk		95%
Menembak 1		90%
Menembak 2		90%
Mengambil		85%

Berdasarkan hasil pengujian pada tabel 2 pengujian dilakukan sebanyak 20 kali dengan akurasi sebesar 90%. Gestur yang berhasil dikenali meliputi gestur menunjuk, gestur menembak 1, gestur menembak 2, serta gestur mengambil. Gestur tersebut dapat dikenali dengan respon yang baik namun tidak bisa selalu dapat mengenali jenis gestur. hal ini diakibatkan karena akurasi model yang masih belum cukup bagus untuk dapat mendeteksi jenis gestur.

4.5. Pengujian terhadap faktor jarak

Pada tahap ini dilakukan pengujian jarak jangkau yang dapat dideteksi model untuk memastikan batasan jarak yang harus dilakukan oleh pengguna.

Tabel 3. Pengujian Jarak

Tampilan	Jarak (cm)	Hasil (20 percobaan)
	25	100% Tidak terdeteksi
	30	90% Terdeteksi
	45	90% Terdeteksi
	60	100% Tidak terdeteksi

Berdasarkan hasil pengujian pada tabel 3 dari 20 kali percobaan didapat bahwa model hanya dapat mendeteksi tangan pada jarak antara 30 cm hingga 45 cm dari kamera dengan akurasi 90%. Jika jarak kurang dari 30 cm atau lebih dari 45 cm, maka deteksi tangan tidak berfungsi dan *hand landmark* tidak akan ditampilkan oleh model.

4.6. Pengujian Waktu Respon Model Dengan Game

Pada tahap ini dilakukan pengujian terhadap waktu respon pada *output* model dengan game untuk memastikan batasan gerakan yang dapat dilakukan

Tabel 4. Waktu Respon

skenario	Respon model (ms)	Respon game (ms)
Menjalankan awal	20-30	30-50
tangan diam	30-50	30-50
tangan bergerak lambat	30-50	60-100
Tangan bergerak cepat	200-600	300-900
Tangan berinteraksi dengan objek pada game	30-50	60-100

Berdasarkan hasil pengujian pada tabel 4 dari 20 percobaan pada skenario awal sampai lambat memiliki rata rata 90 ms, dan dapat berjalan dengan baik terutama pada gerakan biasa hingga lambat. Namun, saat tangan bergerak lebih cepat, model mengalami keterlambatan (*delay*) yang menyebabkan kegagalan dalam mendeteksi keberadaan tangan dengan rata rata 900 ms.

4.7. Pengujian Terhadap Perangkat

Pada tahap ini dilakukan pengujian perangkat terhadap implementasi game yang telah dibuat untuk memastikan model yang telah dibuat dapat diimplementasikan dengan baik pada sebuah game. Pada pengujian ini dibatasi hanya dapat dilakukan pada perangkat dengan spesifikasi ram 8gb dengan OS *Windows* 10 atau lebih. Hasil pengujian perangkat ditunjukkan pada Tabel 5.

Tabel 5. Pengujian Perangkat

Perangkat	Spesifikasi	*Output*
perangkat 1	Ram : 8gb Cpu : 15-8265U Windows : 10	Berjalan dengan baik
perangkat 2	Ram : 8gb Cpu : Athlon 3050U Windows : 10	Berjalan dengan baik
perangkat 3	Ram : 8gb Cpu : Ryzen R5-5600H Windows : 11	Berjalan dengan baik
perangkat 4	Ram : 8gb Cpu : Ryzen R5-4500U Windows : 11	Berjalan dengan baik
perangkat 5	Ram : 8gb Cpu : Intel I5-1135G7 Windows : 11	Berjalan dengan baik
perangkat 6	Ram : 8gb Cpu : Intel i5-10210U Windows : 11	Berjalan dengan baik

Berdasarkan hasil pengujian pada tabel 5 dari 6 perangkat yang diuji didapat hasil 100% perangkat menunjukkan bahwa implementasi game dapat bekerja dengan baik pada perangkat komputer dengan penggunaan ram 8gb dan *windows* 10 hingga selebihnya. Jika kurang dari spesifikasi tersebut program tidak bisa berjalan dengan baik.

4.8. Pengujian Terhadap Responden

Pada tahap ini dilakukan pengujian terhadap responden untuk memastikan apakah model dapat mendeteksi tangan dan apakah respon pengguna dapat sesuai dengan yang diharapkan

Tabel 6. Pengujian Responden

No	Pertanyaan	SST	TS	CS	S	SS
1	Seberapa mudah Anda memahami cara menggunakan sistem gesture ini dalam game?	1	5	5	8	1
Nilai Akhir		1	10	15	32	5

No	Pertanyaan	SST	TS	CS	S	SS
2	Seberapa akurat sistem mengenali gesture tangan Anda saat bermain game?	2	5	7	5	1
Nilai Akhir		2	10	21	20	5
3	Apakah ada keterlambatan (lag) saat sistem mengenali gesture?	3	10	3	4	
Nilai Akhir		3	20	9	16	
4	Seberapa puas Anda dengan pengalaman bermain game menggunakan gesture tangan?	2	3	13	1	1
Nilai Akhir		2	6	39	4	5
5	Apakah penggunaan gesture tangan membuat game menjadi lebih interaktif dan menarik?	2	3	8	6	1
Nilai Akhir		2	6	24	24	5
6	Menurut Anda, dibandingkan dengan kontrol tradisional (mouse, keyboard, touch), penggunaan gesture pada game ini	2	6	8	3	1
Nilai Akhir		2	6	24	12	5

Berdasarkan hasil kuisioner yang ada selanjutnya setiap pertanyaan akan dianalisis menggunakan skor yang sesuai dengan skala penilaian. Setiap skor akan dijabarkan seperti berikut

a. Sangat Tidak Setuju (STS) = 1
b. Tidak Setuju (TS) = 2
c. Cukup Setuju (CS) = 3
d. Setuju (S) = 4
e. Sangat Setuju (SS) = 5

Kemudian untuk proses perhitungan dengan skala likert ditunjukkan sebagai berikut:

a. Interpretasi Skor Perhitungan
   Y = Skor tertinggi × jumlah responden × jumlah pertanyaan
   = 5 × 20 × 6 = 600
   X = Skor terendah × jumlah responden × jumlah pertanyaan
   = 1 × 20 × 6 = 120
b. Rumus Interval
   I = 100 / Jumlah skor
   = 100 / 5 = 20

Berikut adalah kriteria interpretasi skor berdasarkan interval:

A. Angka 0% - 19,99% = Sangat tidak setuju/Tidak puas/Buruk/Kurang sekali
B. Angka 20% - 39,99% = Tidak setuju/Tidak puas/Kurang Baik
C. Angka 40% - 59,99% = Cukup puas/Cukup baik/Netral
D. Angka 60% - 79,99% = Setuju/Baik/Puas/Suka
E. Angka 80% - 100% = Sangat setuju/Baik/Puas/Suka

c. Penyelesaian Akhir
   Rumus Indeks % = Total Skor / Y × 100
   = 335 / 600 × 100
   = 55,8% (Sangat Baik)

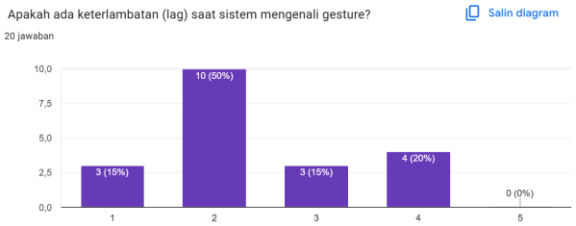
Dari hasil perhitungan tersebut didapat kesimpulan bahwa sistem yang dikembangkan memiliki hasil yang cukup baik. Ini membuktikan bahwa sistem hand gesture pada game tersebut dapat difungsikan secara cukup baik. Setelah didapat hasil secara menyeluruh tersebut dihitung untuk tingkat kepuasan pengguna pada setiap pertanyaan

Gambar 9. Kurva akurasi model

Pada gambar 9 Merupakan grafik yang menampilkan hasil pada pertanyaan pertama. Hasil grafik menunjukkan dari 20 responden sebanyak 8 atau 40% pengguna setuju dengan kemudahan dalam memahami cara menggunakan sistem gesture ini dalam game.

Gambar 10. Pertanyaan Pertama

Pada gambar 10 Merupakan grafik yang menampilkan hasil pada pertanyaan pertama. Hasil grafik menunjukkan dari 20 responden sebanyak 7 atau 35% pengguna setuju atau dirasa cukup akurat dengan keakuratan sistem mengenali gestur tangan saat bermain game



Gambar 11. Pertanyaan Pertama

Pada gambar 11 Merupakan grafik yang menampilkan hasil pada pertanyaan pertama. Hasil grafik menunjukkan dari 20 responden sebanyak 10 atau 50% pengguna mengalami *lag* atau *delay* dengan waktu respon atau keterlambatan yang cukup tinggi saat sistem mengenali gestur

Gambar 12. Pertanyaan Pertama

Pada gambar 12 Merupakan grafik yang menampilkan hasil pada pertanyaan pertama. Hasil grafik menunjukkan dari 20 responden sebanyak 13 atau 65% pengguna merasa cukup puas dengan kepuaasan pengguna dalam pengalaman bermain

Gambar 13. Pertanyaan Pertama

Pada gambar 13 Merupakan grafik yang menampilkan hasil pada pertanyaan pertama. Hasil grafik menunjukkan dari 20 responden sebanyak 8 atau 40% pengguna cukup setuju dengan penggunaan gestur tangan dapat membuat game lebih interaktif dan menarik

Gambar 14. Pertanyaan Pertama

Pada gambar 14 Merupakan grafik yang menampilkan hasil pada pertanyaan pertama. Hasil grafik menunjukkan dari 20 responden sebanyak 8 atau 40% pengguna cukup setuju bahwa game dengan kontrol hand gesture dirasa cukup baik dibandingkan dengan penggunaan kontrol tradisional

5. KESIMPULAN DAN SARAN

Kesimpulan yang didapat pada penelitian yang telah dibuat ini yaitu implementasi hand gesture dengan CNN pada game dapat dilakukan dengan akurasi deteksi sebesar 70%. Pada pengujian jarak memiliki jarak deteksi antara 30 hingga 45 cm dari tangan ke kamera. Hasil respon model berkisar 30-50 ms namun ketika dijalankan dengan game hasil respon menjadi lambat berkisar 60-100 ms. Dan berdasarkan hasil pengujian pada pengguna diperoleh bahwa 40% pengguna menilai sistem mudah dipahami dan 35% menilai akurasi pengenalan gestur cukup baik. Tetapi, 50% responden mengalami lag atau *delay* saat sistem mengenali gestur. 65% responden merasa puas terhadap pengalaman bermain. Selain itu, 40% menyatakan penggunaan gestur membuat game lebih interaktif dan menarik, serta 40% lainnya menilai kontrol berbasis hand gesture cukup baik dibandingkan kontrol tradisional. Saran dari penelitian ini adalah agar pengembangan selanjutnya memungkinkan model deteksi untuk mengenali dua tangan secara bersamaan, serta menambahkan variasi gestur (sebutkan jenis gestur tambahan apa saja) guna meningkatkan interaktivitas dalam implementasinya. model juga perlu dioptimalkan lebih lanjut untuk mengurangi waktu respons, sehingga dapat memberikan pengalaman bermain yang lebih baik dan responsif. Dan juga Model dapat dikembangkan pada permainan perangkat *Virtual Reality* (VR).

DAFTAR PUSTAKA

[1] Kementerian Kesehatan Republik Indonesia, "Cedera akibat game daring," *Sehat Negeriku*, Jan. 9, 2024. [Online]. Available: https://sehatnegeriku.kemkes.go.id/baca/blog/20240109/3344664/cedera-akibat-game-daring/. [akses: Feb. 5, 2024].

[2] S. Munir, S. Mushtaq, A. Nadeem, and S. Zahra, "Hand gesture Recognition: A Review," *International Journal of Scientific & Technology Research*, vol. 10, 2021.

[3] M. Sholawati, K. Auliasari, and F. X. Ariwibisono, "Pengembangan Aplikasi Pengenalan Bahasa Isyarat Abjad Sibi Menggunakan Metode Convolutional Neural Network (CNN)," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 6, no. 1, 2022.

[4] R. A. Pratama, S. Achmadi, and K. Auliasari, "Penerapan Metode Convolutional Neural Network Pada Aplikasi Deteksi Wajah Pengunjung Perpustakaan," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 6, no. 1, 2022.

[5] S. Satwikayana, S. A. Wibowo, and N. Verdyansyah, "Sistem Presensi Mahasiswa Otomatis Pada Zoom Meeting Menggunakan Face Recognition Dengan Metode Convolutional Neural Network Berbasis Web," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 5, no. 2, 2021.

[6] C. W. Wiguna, J. D. Irawan, and M. Orisa, "Penerapan Metode Convolutional Neural Network Pada Aplikasi Deteksi Wajah Buronan Berbasis Web," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 6, no. 2, 2023.

[7] D. Khairianto and R. Firdaus, "Penerapan Hand Gesture Recognition Sebagai Media Kontrol Presentasi Aplikasi Powerpoint," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 2, 2024.

[8] M. R. S. Alfarizi *et al*., "Penggunaan Python Sebagai Bahasa Pemrograman untuk Machine Learning dan Deep Learning," *Karimah Tauhid (Karya Ilmiah Mahasiswa Bertauhid)*, vol. 2, no. 1, 2023.

[9] W. Diharjo, "Game Edukasi Bahasa Indonesia Menggunakan Metode Fisher Yates Shuffle Pada Genre Puzzle Game," *INTEGER (Journal of Information Technology)*, vol. 5, no. 2, 2020.

[10] P. Kanani and M. Padole, "Deep Learning to Detect Skin Cancer using Google Colab," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 8, no. 6, 2019.