p-ISSN : 2443-2210
e-ISSN : 2443-2229

Implementasi Algoritma $P *Point Cloud Recognizer* pada Pengenalan Angka Berbasis *Game*

http://dx.doi.org/10.28932/jutisi.v8i3.5472

Riwayat Artikel
Received: 29 September 2022 | *Final Revision*: 05 Desember 2022 | *Accepted*: 05 Desember 2022

Muhammad Farid Athar[✉ #1], Yohannes [#2], Yoannita [#3]

[#1,2,3] *Program Studi Informatika , Universitas Multi Data Palembang*
Jalan Rajawali No. 14, Palembang, Indonesia

[1]mfa4th4r@gmail.com

[2]yohannesmasterous@mdp.ac.id

[3]yoannita@mdp.ac.id

[✉]Corresponding author: mfa4th4r@gmail.com

Abstrak — **Semakin banyak perangkat memiliki masukan berupa gestur. Mengenali gestur tersebut menjadi semakin dibutuhkan untuk mengembangkan aplikasi. Salah satu metode yang digunakan dalam mengenal gestur adalah Point Cloud Recognizer atau $P. Pengenalan gestur dapat digunakan untuk mengenali tulisan seperti angka atau. Hasil pengenalan ini dapat digunakan untuk edukasi tentang menulis melalui aplikasi, seperti game. Penelitian dilakukan dengan mengimplementasikan $P pada game untuk menunjukkan $P dapat menjadi saran metode yang dapat digunakan dalam pengembangan game yang memerlukan fitur tersebut. Dalam penelitian $P diimplementasi dengan menggunakan game engine Unity dan bahasa pemrograman C#. Data yang digunakan ada 3 set numeral 1 sampai 10 dengan konfigurasi jumlah point pada $P sejumlah 32. Total 100 pengujian dilakukan pada game dengan hasil akurasi 99% dalam pengenalan, menunjukkan bahwa metode $P dapat mengenali gestur tulisan dengan baik.**

Kata kunci—**Implementasi; Pengenalan *gestur*; *Point Cloud*.**

Implementation of $P Point Cloud Recognizer for Game-based Numeral Recognition

Abstract — ***More devices use gestures as their input method. Recognizing these gestures becomes more important for app development. One of the methods used for gesture recognition is Point Cloud Recognizer or $P. Gesture recognition can be used to recognize written characters like numbers or letters. Result of this recognition can be used for education involving apps, like games. This study is done by implementing $P in games to show that $P can be used as one of the methods for gesture recognition when developing games that need such features. In this study $P is implemented with the help of the game engine Unity with C# programming language. 3 sets of numerals 1 to 10 are used as data with $P configured to use 32 points. Total of 100 tests are done in the game resulting in 99% accuracy, showing $P is able to recognize the gesture well.***

Keywords— ***Gesture Recognizer; Implementation; Point Cloud***

I. PENDAHULUAN

Bertambahnya penggunaan layar sentuh pada perangkat mobile membuat penggunaan gestur sebagai input semakin relevan. Bahkan perangkat seperti laptop memiliki input gestur pada touchpadnya. Hal ini membuat pengenalan gestur semakin penting Pengenalan gestur merupakan sistem yang memungkinkan komputer menginterpretasi gerakan manusia termasuk gerakan tangannya. Gestur tersebut nantinya dapat digunakan sebagai input dalam komputer [1].

Pengenalan gestur sentuh merupakan metode klasifikasi gerak dari alat tunjuk atau jari pada layar. Pengenalan gestur dapat berbentuk 2D seperti pada layar sentuh maupun 3D. Pengenalan gestur tulisan 2D bertujuan mengenali gestur dengan membandingkan template gestur yang telah direkam ke dalam training set dari pengenal sebelumnya [2].

Metode Point Cloud Recognizer (disimbolkan $P) merupakan pengenal gestur yang merepresentasikan gestur tulisan dalam sekumpulan titik yang disebut point cloud. Huruf P pada $P berasal dari kata Point Cloud sedangkan simbol dollar karena $P termasuk dari famili $ recognizer maka $P disebut Point Cloud Recognizer. $P berbeda dari pendahulunya, $1 *Unistroke Recognizer* dan $N *Multistroke Recognizer* karena $P merupakan *point-based-recognizer* dan bukan *stroke-based-recognizer*. Dalam *stroke-based-recognizer* seperti $N tulisan yang memiliki urutan, jumlah, ataupun arah tulisan berbeda dianggap gestur yang berbeda. Pada *point-based-recognizer* jumlah, urutan, maupun arah penulisan menjadi tidak relevan dalam pengenalan gestur [3]. *Point Cloud Recognizer* telah dipakai sebagai pengenalan gestur pada beberapa aplikasi seperti "Ombre Fabula" di 2014 yang menggunakan $P sebagai pengenalan kontrol gestur pada cerita interaktifnya. $P juga telah sukses dipakai pada game "*City Protector*" dimana pengenalan gestur dipakai sebagai input dari gameplay [4]. Dibanding metode pengenal gestur lain seperti HMM, ekstraksi fitur ataupun kombinasi *classifier* lain *Point Cloud Recognizer* punya keuntungan mudah diimplementasikan dalam aplikasi namun tetap memiliki performa yang baik serta dapat dibuat untuk mengenali gestur 3D karena representasi point cloud yang simpel [3].

Pengenalan angka atau numerasi merupakan kemampuan menggunakan simbol matematis untuk memecahkan masalah sehari hari dan kemampuan mengenali informasi secara spasial untuk mengambil keputusan. Numerasi berguna untuk anak melakukan perencanaan keputusannya dalam hidup berdasarkan informasi yang didapatkan dan ditafsirkan ke dalam hubungan matematis yang sesuai dengan konteks masalahnya [5]. Edukasi numerasi menggunakan game pernah diteliti sebelumnya dan berdasarkan observasi beserta tes tertulis sebelum dan sesudah menggunakan aplikasi terlihat perkembangan pesat pada kemampuan menjawab pertanyaan tentang pengenalan angka dan berhitung [6].

Pengembangan game merupakan proses kompleks terdiri dari beberapa disiplin ilmu seperti desain grafis, rekayasa perangkat lunak, manajemen dan sebagainya [7]. Penggabungan disiplin-disiplin ini memberikan game kemampuan untuk menyajikan konsep dengan cara yang menarik. Selain itu pengembangan game berbeda dengan perangkat lunak lainnya karena diperlukan aspek keseruan dalam menggunakan game [8]. Fokus kepada aspek tersebut membuat game menjadi sarana yang bagus sebagai alat untuk mendukung proses belajar dengan cara yang seru dan kreatif [9].

Sistem numerasi merupakan aturan yang digunakan untuk menuliskan suatu bilangan dengan menggunakan lambang bilangan. Lambang untuk menyatakan suatu bilangan disebut sebagai numeral. Satu lambang menggambarkan satu bilangan namun satu bilangan dapat memiliki banyak nama mengikuti aturan tertentu. Terdapat banyak sistem numerasi yang dipakai dalam sejarah dalam berbagai tempat, beberapa masih dipakai hingga sekarang [10]. Sistem numerasi yang sering dipakai sekarang adalah sistem numerasi Hindu-Arab. Sistem numerasi Hindu-Arab memiliki 10 digit yang berulang. Angka pada sistem Hindu-Arab disusun dari kanan ke kiri dan angka pertama dari kanan disebut satuan, angka kedua dari kanan disebut puluhan dan seterusnya [11].

Salah satu penelitian terdahulu tentang implementasi algoritma pengenalan gestur adalah penelitian menggunakan metode JST *backpropagation* oleh Syahrudin [12]. Penelitian menguji gestur penulisan alfabet dengan total 26 karakter. Pengujian dilakukan pada game edukasi. Tiap alfabet merupakan *level* di dalam game. Penelitian menemukan hasil akurasi rata-rata 91%.

Penelitian lain menguji metode *Restricted Boltzmann Machines*(RBM) untuk tulisan tangan oleh Susilawati [13]. Penelitian dilakukan pada *dataset* angka tulisan tangan dari MNIST. Numeral yang dilakukan pengujian terdiri dari angka 0 sampai angka 9. Penelitian menunjukkan metode dapat mengenali 93,42% data uji dengan tepat menggunakan *learning rate* 0,05 dan *momentum* 0,7.

Penelitian lain menguji tulisan angka dengan metode CNN oleh Ahlawat dkk [14]. Penelitian dilakukan pada *dataset* angka tulisan tangan dari MNIST. Penelitian menguji akurasi pengenalan dari CNN 3 *layer* dan 4 *layer*, serta beberapa *optimizer* seperti Sgdm, Adam, Adagrad dan Adadelta. Penelitian menemukan hasil terbaik terdapat pada CNN 3 *layer* menggunakan *optimizer* Adam. Akurasi pengenalan dari kombinasi ini sebesar 99,89%.

Sementara penggunaan $P pada penelitian sebelumnya ada dalam pengenalan aksara Jepang oleh Yogi Udjaja [15]. Penelitian ini berbentuk pengembangan game edukasi dan metode yang digunakan gabungan dari $1 dan $P. Metode ini digabung karena perlunya mengecek urutan tulis pada aksara Jepang. Dalam prosesnya $1 digunakan untuk mengecek satu per satu goresan, lalu $P dipakai untuk mengecek apakah keseluruhan aksara cocok dengan data. Hasil penelitian ini

merupakan keberhasilan edukasi aksara dengan perkembangan pemahaman tulisan aksara sebesar 20% hingga 100% dalam penggunaan satu minggu.

Penggunaan $P dalam game sebelumnya dapat dilihat pada game *arcade* oleh Danawan dkk [4]. $P digunakan sebagai *input* untuk mencocokkan gestur pada layar *smartphone* ke musuh di dalam game yang memerlukan gestur yang tepat untuk dimusnahkan. Terdapat 8 gestur berbeda yang digunakan dalam studi ini. Hasil studi berupa survei dari 8 pemain, dengan 90,6% responden mengatakan dalam pengalaman bermain gestur yang digambar sesuai dengan yang ditunjukkan dalam layar.

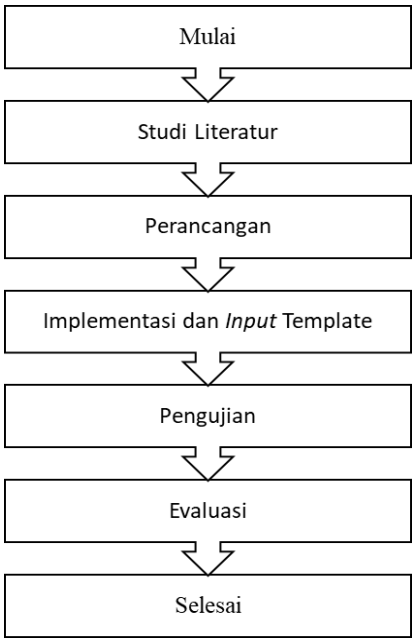
Penggunaan lain untuk $P ada pada pengenalan ekspresi melalui alat *Kinect* oleh [16]. Pada penelitian ini $P digunakan untuk mengenali *point cloud* dalam tiga dimensi(3D). Terdapat 7 *template* ekspresi yang dipakai dengan jumlah *point* dalam tiap *template* yaitu 121 *point*. Hasil penelitian menunjukkan 94,28% ekspresi dikenali dengan tepat.

Dibandingkan metode terdahulu metode $P yang akan dipakai pada penelitian ini tidak memerlukan data input yang banyak karena menggunakan *template matching*, juga tidak memerlukan training yang lama. Metode $P juga lebih mudah diterapkan karena proses yang digunakan lebih simpel dibandingkan metode lain seperti JST dan dapat lebih mudah dimodifikasi sesuai dengan kebutuhan.

Penelitian ini bertujuan untuk mengetahui performa $P dalam mengenali tulisan angka pada aplikasi berbasis game, mengetahui numeral mana yang paling mudah dikenali dan sulit dikenali oleh $P, serta mengukur performa $P terhadap metode pengenalan sebelumnya.

II. METODE PENELITIAN

Gambar 1 menunjukkan tahapan yang diperlukan dalam penelitian untuk mengimplementasikan $P *Point Cloud Recognizer* berbasis game. Terdapat lima tahapan implementasi yaitu studi literatur, perancangan, implementasi, pengujian, dan evaluasi.



Gambar 1. Metodologi Penelitian

A. Studi Literatur

Dilakukan pencarian jurnal tentang penerapan algoritma pada game edukasi pengenalan angka dan jurnal tentang algoritma pengenalan angka terkait. Pencarian literatur bertujuan mencari dasar untuk memakai algoritma yang dipilih.

B. Perancangan

Pada tahap ini dilakukan perancangan implementasi berupa pemilihan gestur yang akan diinput serta menyiapkan antarmuka dan navigasi pada game yang telah disediakan untuk implementasi. Implementasi dilakukan dalam *game* engine Unity versi 2020.1.4f1 menggunakan Bahasa C#. Bentuk game didasarkan pada game edukasi angka untuk anak anak dan dibuat dalam bentuk dua dimensi(2D). Bagian pengenalan angka dibuat berdasarkan referensi dari demo $P pada website *University of Washington* dengan perubahan untuk menyesuaikan tema game. Kode algoritma didapatkan dari *source code* $P dengan bahasa C# di website yang sama.

Pengenalan gestur pada $P dapat dibagi menjadi beberapa proses. Proses *Gesture Template* merupakan proses mengambil template dari setiap numeral(1,…,10). Setiap numeral akan diambil *template* sebanyak 3 kali. Proses *pre-processing* dijalankan saat gestur diinput, baik yang dijadikan template maupun yang ingin dilakukan pengenalan. *Pre-processing* terdiri dari 3 tahap, diantaranya :

1) Resampling merupakan tahap dimana gestur diubah menjadi sejumlah titik yang jaraknya sama. Pada tahap ini gestur yang akan dibandingkan telah memiliki titik rujukan yang konsisten yaitu jumlah titik setelah resampling.

2) Scale merupakan tahap dimana ukuran gestur diubah terhadap nilai tertentu. Pada tahap ini jarak antar titik pada gestur akan semakin kecil.

3) Translate merupakan tahap dimana posisi gestur diubah ke nilai tertentu. Pada tahap ini posisi antar gestur akan menjadi berdekatan dan akan memudahkan algoritma pengenalan.

Proses *template matching* merupakan proses mencocokkan gestur dengan template yang paling mirip dalam data. Dalam proses ini pencocokan dilakukan dengan menghitung jarak titik di gestur ke titik di template yang paling dekat. Pencocokan diulang terus hingga semua titik punya pasangan dan dihitung *closeness score* nya, Setelah ini *matching* diulang lagi ke template lain. Terakhir diambil template dengan *closeness score* paling besar sebagai output. Alur dari proses pengenalan ini dapat dilihat pada Gambar 2.

$P membandingkan dua titik pada *point cloud* dengan memakai pendekatan *Hungarian Algorithm* melalui *heuristic* Greedy-X. Artikel dari Vatavu et al., pada tahun 2012 [3]menyebutkan versi terbaik *heuristic* ini ada pada Greedy-5. Greedy-5 membandingkan poin di *cloud* pertama (C_i) dengan titik pada template terdekat yang belum dicocokkan, lalu dilanjutkan ke titik selanjutnya ($C_i + 1$) sampai seluruh poin ($i = 1 \dots n - 1$) sudah di cocokkan. Setelahnya di lakukan pencarian jarak *Euclidean* seperti pada persamaan 1.

$$\sum_i \boldsymbol{w_i} \cdot \left\| \boldsymbol{C_i} - \boldsymbol{T_j} \right\| = \sum_{i=k}^{n} \boldsymbol{w_i} \cdot \left\| \boldsymbol{C_i} - \boldsymbol{T_j} \right\| + \sum_{i=1}^{k-1} \boldsymbol{w_i} \cdot \left\| \boldsymbol{C_i} - \boldsymbol{T_j} \right\| \quad (1)$$

Keterangan :
w_i = *weight*
k = nilai pada *point* permulaan
n = jumlah iterasi
C_i = *Point* ke-i dalam C
T_j = *Point* ke-j pada T
*C_i dan T_j disebut sebagai *cloud pair*.

Poin yang telah dicocokkan dengan template tidak dimasukkan kembali pada Greedy-5 untuk mempercepat komputasi. Karena hal ini hasil pencocokan selanjutnya kurang pasti dibandingkan dengan pencocokan saat langkah pertama. Sebagai gantinya diberikan nilai *weight* yang nilainya semakin mengurang dengan bertambahnya langkah pencocokan. Perhitungan *weight* dilakukan seperti persamaan 2.

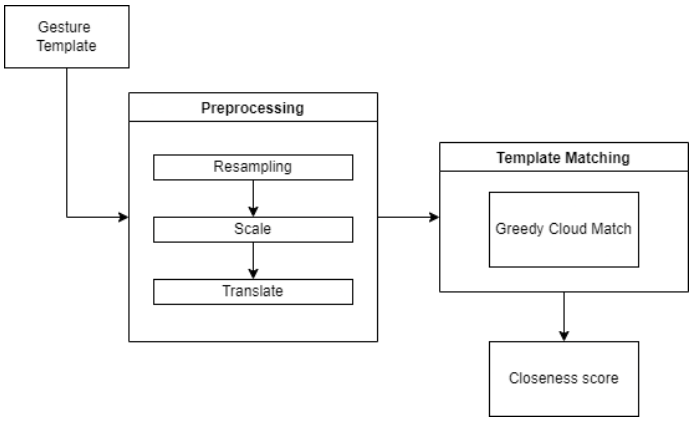
$$\boldsymbol{w_i} = \mathbf{1} - \frac{\boldsymbol{i-1}}{\boldsymbol{n}} \quad (2)$$

Keterangan :
$i = 1 \dots n$, dimana n merupakan langkah algoritma ke-n.

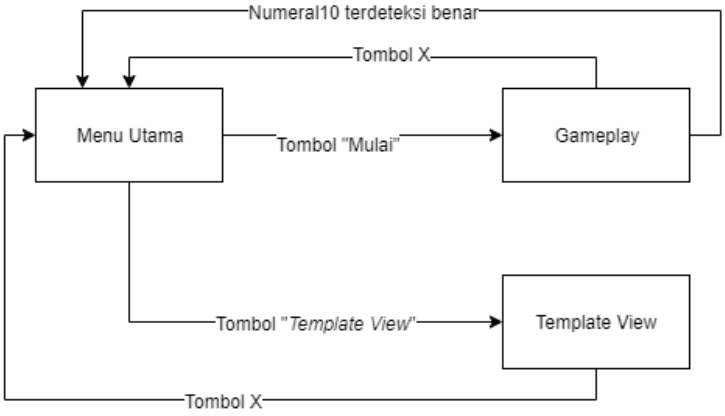
Perancangan alur permainan dimulai dari Menu Utama. Dalam Menu Utama terdapat tombol “Mulai” dan tombol “*Template View*”. Bila tombol “Mulai” ditekan maka akan berpindah ke *gameplay* sedangkan “*Template View*” berisi menu untuk melihat *template* yang dipakai. Dalam menu *gameplay* terdapat *progress bar*, jendela penulisan dan 3 tombol. Tombol paling atas berbentuk bulat berfungsi kembali Menu Utama dari tengah *gameplay*,tombol selanjutnya di tengah kanan layar berfungsi untuk menghapus seluruh tulisan pada jendela penulisan. Tombol di bawahnya bertulisan “CHECK” berfungsi untuk menguji tulisan.

Perancangan *gameplay* dilakukan dengan menyiapkan *array* berisi *string* angka “1”, ”2” dan seterusnya hingga “10”. *Array* ini digunakan sebagai bentuk stage. Pengenal akan mulai dari angka 1 dan bila tulisan benar akan maju ke angka 2 dan seterusnya hingga 10. Bila benar di angka 10, akan ada *popup* tulisan selamat telah memainkan game dan tombol untuk

kembali ke Menu Utama. *Template* diberi nama sama dengan anggota *array*, misalkan template numeral 1 diberi nama "1" untuk memudahkan pengenalan. *Output* dari fungsi $P adalah nama *template* terdekat dan *closeness score*. Nama dan *closeness score* ini diubah menjadi string dan ditampilkan pada *Dialog Window* yang muncul saat tombol "CHECK" ditekan. Gambar 3 menunjukkan grafik alur permainan.



Gambar 2. Rancangan Program *Recognizer*



Gambar 3. Alur Permainan

C. *Implementasi dan Input Template*

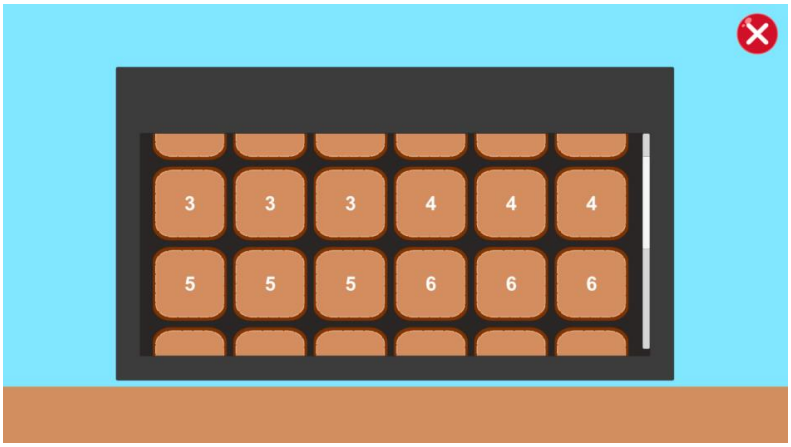
Pada tahap ini dilakukan implementasi algoritma pada game yang telah dirancang dalam *game engine* Unity dan dilakukan proses memasukkan template. *Template* disimpan dalam bentuk file .xml yang berisi point-point pada gestur serta label gesture tersebut. File-file .xml ini nantinya akan dipanggil untuk dibandingkan dengan gestur yang ditulis di dalam game dan ditentukan template mana yang paling mirip serta tingkat kemiripannya. Total 30 *template* di-*input* ke dalam game yang terdiri dari 3 data untuk setiap numeral dari 1 sampai 10. Gambar 4 menunjukkan data *template* yang berbentuk .xml. Data disimpan di dalam folder game dan dapat dimunculkan melalui menu *template view* dari layar utama. Data dimunculkan di dalam game melalui *Line Renderer* dari Unity.

Assets > Resources > GestureSet > Numbers > 1-132919564957517859.xml

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<Gesture Name = "1">
    <Stroke>
        <Point X = "284.5" Y = "-525.75" T = "0" Pressure = "0" />
        <Point X = "284.5" Y = "-525.75" T = "0" Pressure = "0" />
        <Point X = "284.5" Y = "-525.75" T = "0" Pressure = "0" />
        <Point X = "284.5" Y = "-525.75" T = "0" Pressure = "0" />
        <Point X = "284.5" Y = "-525.75" T = "0" Pressure = "0" />
        <Point X = "284.5" Y = "-525.75" T = "0" Pressure = "0" />
        <Point X = "287" Y = "-527" T = "0" Pressure = "0" />
        <Point X = "299.5" Y = "-538.25" T = "0" Pressure = "0" />
        <Point X = "319.5" Y = "-558.25" T = "0" Pressure = "0" />
        <Point X = "350.75" Y = "-585.75" T = "0" Pressure = "0" />
        <Point X = "369.5" Y = "-602" T = "0" Pressure = "0" />
        <Point X = "384.5" Y = "-613.25" T = "0" Pressure = "0" />
        <Point X = "390.75" Y = "-624.5" T = "0" Pressure = "0" />
        <Point X = "395.75" Y = "-630.75" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-633.25" T = "0" Pressure = "0" />
        <Point X = "399.5" Y = "-628.25" T = "0" Pressure = "0" />
        <Point X = "399.5" Y = "-607" T = "0" Pressure = "0" />
        <Point X = "399.5" Y = "-567" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-515.75" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-468.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-423.25" T = "0" Pressure = "0" />
        <Point X = "398.25" Y = "-388.25" T = "0" Pressure = "0" />
        <Point X = "399.5" Y = "-367" T = "0" Pressure = "0" />
        <Point X = "400.75" Y = "-358.25" T = "0" Pressure = "0" />
        <Point X = "400.75" Y = "-355.75" T = "0" Pressure = "0" />
        <Point X = "400.75" Y = "-352" T = "0" Pressure = "0" />
        <Point X = "402" Y = "-349.5" T = "0" Pressure = "0" />
        <Point X = "403.25" Y = "-348.25" T = "0" Pressure = "0" />
        <Point X = "403.25" Y = "-348.25" T = "0" Pressure = "0" />
        <Point X = "403.25" Y = "-348.25" T = "0" Pressure = "0" />
        <Point X = "403.25" Y = "-348.25" T = "0" Pressure = "0" />
        <Point X = "403.25" Y = "-348.25" T = "0" Pressure = "0" />
    </Stroke>
</Gesture>
```

Gambar 4. Struktur Data Template

Gambar 5 menunjukkan halaman dari menu *template view*. *Template* disusun dalam ikon kotak bertulisan nama template. Ikon disusun dalam 6 kolom dan disediakan fitur *scroll* untuk mencari *template* yang ingin ditampilkan. Jumlah isi ikon di dalam menu akan otomatis berubah bila *template* ditambahkan atau dihapus. Gambar 6 menunjukkan *template* yang digunakan dalam pengujian. Data yang dimunculkan berbentuk titik titik putih yang menunjukkan posisi tiap *point* dalam *template*, dengan total 32 *point* dalam *template*.

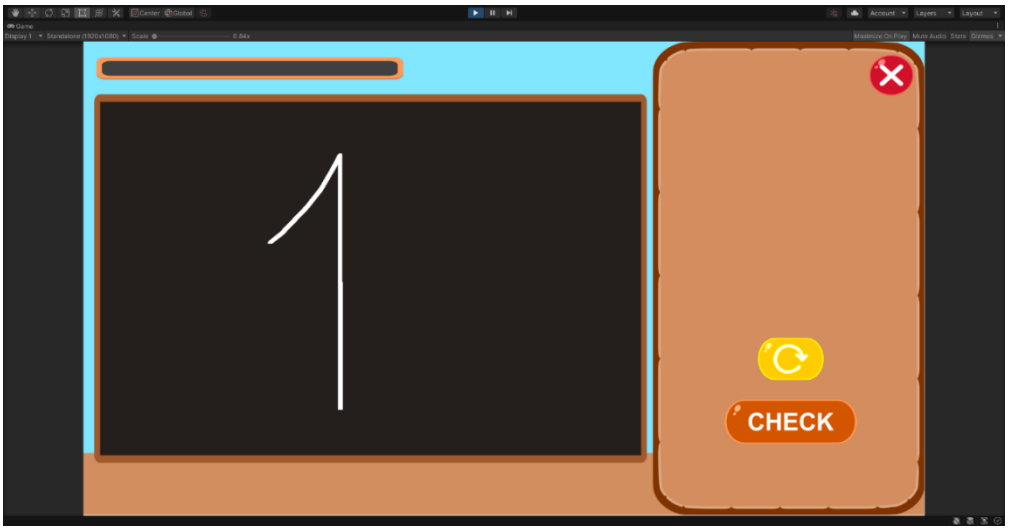


Gambar 5. Halaman *Template View*

Gambar 6. Template Numeral yang dipakai

D. Pengujian

Pada tahap ini dilakukan pengujian algoritma dengan cara menjalankan game dan menggambar gestur. Lingkungan pengujian terdiri dari Laptop dengan Unity 2020 beserta *Pen Tablet* sebagai alat inputnya. Pengujian dilakukan dari angka 1(satu) hingga angka 10(sepuluh) secara urut mulai dari 1. Bila pemain memasukkan nomor secara tidak urut akan tetap dimunculkan angka apa yang dikenali tapi *progress bar* tidak maju. *Progress bar* ditunjukkan pada UI game seperti pada Gambar 7. Setiap selesai pengenalan akan ditunjukkan akurasi dari *recognizer* dan nama template yang dianggap paling dekat menurut algoritma.Gambar 8 menunjukkan kondisi apabila angka urut dan angka berhasil dikenali.



Gambar 7. Halaman Gameplay

Gambar 8. Kondisi Pengenalan Berhasil

E. Evaluasi

Pada tahap ini hasil pengujian dibandingkan dengan penelitian sebelumnya dan diambil kesimpulan penelitian. Karena pengujian dilakukan dengan menjalankan game, iterasi yang didapatkan hanyalah sedikit maka hasil dibuat menjadi tabel lalu dilakukan perhitungan metrik dari tabel tersebut.

Metrik yang digunakan untuk menilai $P terdiri dari Akurasi, *Precision*, dan *Recall*. Rumus untuk menghitung ketiga metrik tersebut dapat dilihat pada persamaan (3), (4), dan (5) berikut. Akurasi akan digunakan sebagai pembanding performa

$P dengan metode yang digunakan sebelumnya.

$$Accuracy = \frac{TP}{TP+TN+FP+FN} \quad (3)$$

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

Keterangan :
TP/*True Positive* = Data benar yang diprediksi benar.
TN/*True Negative* = Data salah yang diprediksi salah.
FP/*False Positive* = Data salah yang diprediksi benar.
FN/*False Negative* = Data benar yang diprediksi salah.

III. HASIL DAN PEMBAHASAN

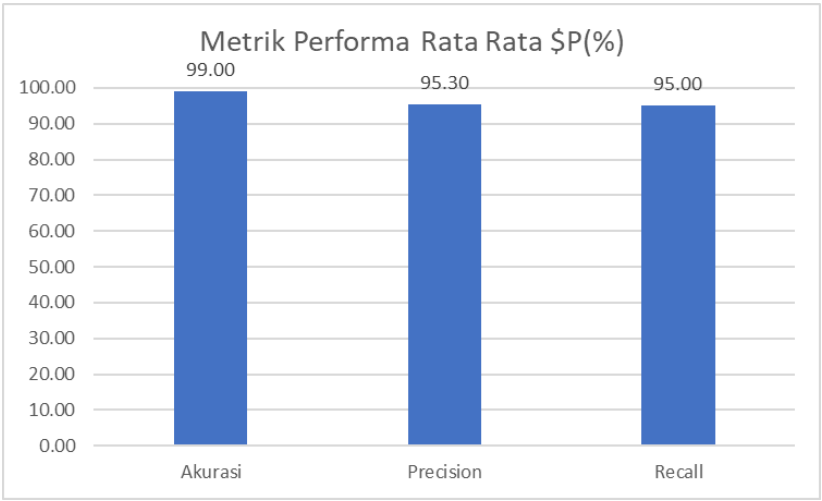
Hasil penelitian didapatkan dengan menjalankan permainan dan menulis angka 1 sampai 10 sebanyak sepuluh kali permainan. *Resampling* gestur diatur menjadi 32 titik, *Scale* hingga ukuran gestur ada pada jarak 0 hingga 1, dan *Translate* ke titik *origin*(0,0). Tabel 1. menunjukkan hasil pengujian $P pada game. Dari total 100 pengujian ditemukan 5 eror, dua pada numeral 3, satu pada numeral 5, satu pada numeral 7 dan satu pada numeral 9. Numeral 3 memiliki dua eror dimana numeral dikenali menjadi numeral 9, sebaliknya numeral 9 satu kali eror dikenali menjadi numeral 3. Numeral 5 memiliki satu eror dikenali menjadi numeral 6, dan Numeral 7 memiliki satu eror dikenali menjadi numeral 8. Total numeral dikenali 95 kali dari 100 pengujian dengan tingkat pengenalan pada pengujian 95%.

TABEL 1
DATA PENGUJIAN PENGENALAN ANGKA

Numeral	Jumlah dikenali										Jumlah Error	Recognition Rate
	1	2	3	4	5	6	7	8	9	10		
1	10	0	0	0	0	0	0	0	0	0	0	100%
2	0	10	0	0	0	0	0	0	0	0	0	100%
3	0	0	8	0	0	0	0	0	2	0	2	80%
4	0	0	0	10	0	0	0	0	0	0	0	100%
5	0	0	0	0	9	1	0	0	0	0	1	90%
6	0	0	0	0	0	10	0	0	0	0	0	100%
7	0	0	0	0	0	0	9	1	0	0	1	90%
8	0	0	0	0	0	0	0	10	0	0	0	100%
9	0	0	1	0	0	0	0	0	9	0	1	90%
10	0	0	0	0	0	0	0	0	0	10	0	100%
Total Error											5	95%
Total Benar											95	

Setelah dilakukan pengujian maka dilakukan proses menghitung metrik seperti yang telah dibahas pada bab sebelumnya. Grafik pada Gambar 9 menunjukkan metrik dari tiap kelas dalam pengujian dari 1 sampai 10 dan grafik pada Gambar 10 menunjukkan metrik rata rata pengujian untuk dijadikan metrik $P.

Gambar 9. Grafik Metrik per Kelas



Gambar 10. Grafik Metrik $P

Pengenalan angka banyak dilakukan dalam citra komputer. Metode metode yang sering dipakai seperti CNN menggunakan gambar citra digital bukan *Point Cloud* seperti $P. Dengan pertimbangan perbedaan bentuk data, diambil metode CNN sebagai pembanding dalam pengenalan angka. Tabel 2. menunjukkan perbandingan $P dengan metode CNN dari penelitian Ahlawat [14].

TABEL 2

Perbandingan Akurasi Metode dari Penelitian Ahlawat [14]

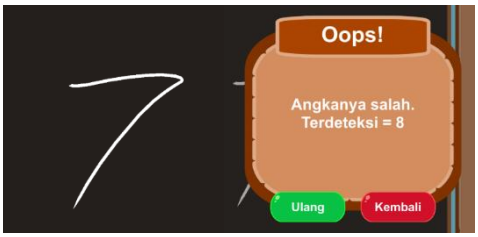
	CNN	**$P Recognizer**
Akurasi	99,89%	99%

B. Pembahasan

Dari pengujian ditemukan bahwa numeral 3 memiliki kesalahan pengenalan terbanyak dengan nilai pengenalan 80%. Kedua kasus salah pengenalan menunjukkan numeral 9 sebagai template terdekat. Sebaliknya numeral 9 memiliki 1 kasus yang menunjukkan numeral 3 sebagai template terdekat. Hal ini dapat terjadi karena ada template label 3 dan 9 yang bentuknya dekat dalam data atau jumlah point dalam *resampling* kurang tepat untuk membedakan kedua numeral tersebut.

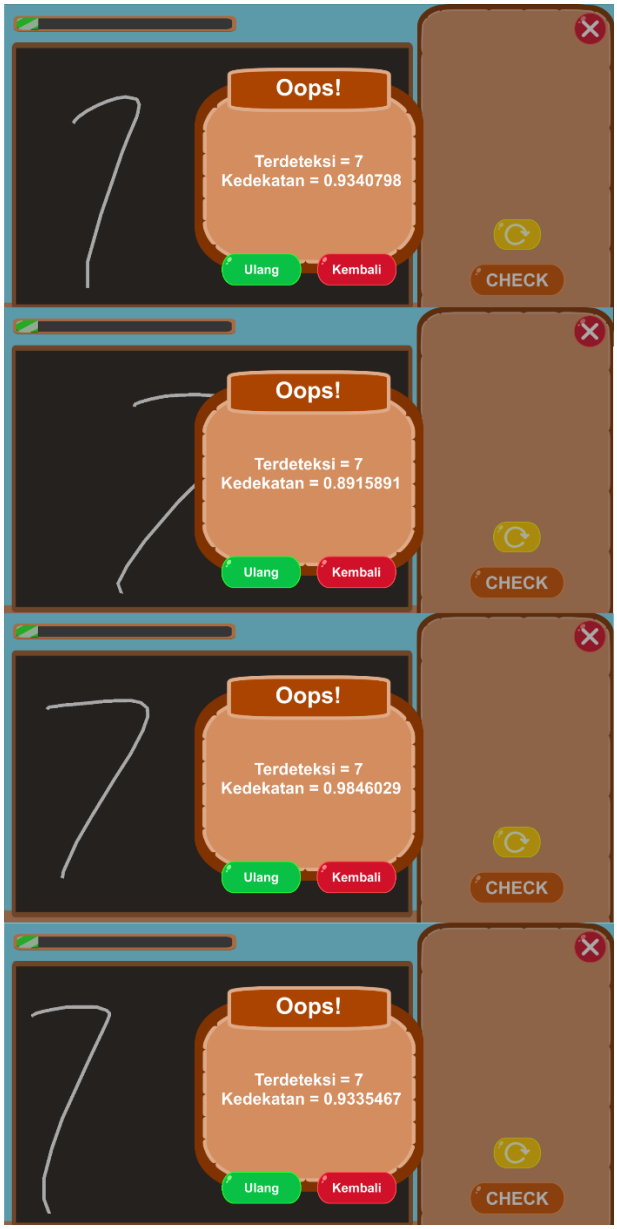
Numeral 5 memiliki satu kasus salah pengenalan menjadi numeral 6. Untuk kasus sebaliknya, numeral 6 tidak memiliki kasus salah pengenalan ke template lain. Kasus menarik terjadi pada pengenalan numeral 7. Terdapat satu kasus dimana pengenal melabel numeral 7 menjadi 8 seperti pada Gambar 11. Melihat template kedua numeral dan gestur input belum terlihat penyebab kasus ini, terutama karena secara visual kedua numeral ini tidak mirip. Dugaan terkuat ada pada garis awal pada input yang terlalu melengkung, karena pada template numeral 8 terdapat garis dengan kemiringan mirip dengan numeral 7, sedangkan garis bagian atasnya lengkung, sedangkan di template numeral 7 garis ini lurus.

Untuk mengatasi kasus numeral 7 ditambahkan 4 *template* numeral 7 yang dapat dilihat pada Gambar 12. Dua *template* yang ditambahkan memiliki karakteristik garis atas dibuat lebih melengkung untuk menguji dugaan pada paragraf sebelumnya. Hasil tes ulang berhasil menghilangkan kasus tersebut. Gambar 13 menunjukkan beberapa hasil tes setelah penambahan *template*. Tulisan "*Oops!*" muncul karena pengujian ulang langsung melihat numeral 7.



Gambar 11. Kasus Error Numeral 7

Gambar 12. Tambahan *template* numeral 7



Gambar 13. Beberapa hasil tes dengan *template* tambahan

IV. SIMPULAN

Berdasarkan penelitian yang telah dilakukan, diperoleh kesimpulan metode berhasil diimplementasikan, dengan rata-rata akurasi 99%, *precision* 95,3%, dan *recall* 95%. Hasil ini menunjukkan akurasi yang sedikit lebih kurang akurat dari metode sebelumnya yaitu CNN dengan akurasi 99,89% Dari 10 iterasi game ditemukan numeral dengan akurasi paling kecil adalah numeral 3 dan 9 dengan akurasi 97%, dan terdapat beberapa nilai akurasi 100% yaitu pada numeral 1, 2, 4, 6, 8, dan 10. Saran untuk penelitian selanjutnya dapat dilakukan penyempurnaan pada game dengan menambahkan *voice line* dan efek suara, serta menyempurnakan bagian desain permainan seperti menambahkan aturan permainan, kondisi menang dan kalah, *gameplay loop*, serta *goal* permainan. Untuk pengembangan metode dapat dilakukan dengan menambah *template* yang lebih bervariasi bentuknya, dapat juga menggunakan metode untuk pengenalan numeral lain seperti numeral romawi, juga melihat perubahan apabila angka *resample* diubah untuk melihat pengaruh jumlah poin terhadap performa, lalu dapat dilakukan pengujian dengan alat input lain seperti *smartphone*.

DAFTAR PUSTAKA

[1] I. Wijayanto and E. Susatio, "Identifikasi Pergerakan Dasar pada Game Untuk Pengembangan Gesture Recognition Berbasis Kinect Identification of Basic Movement on The Game for A Gesture Recognition Based on Kinect," *eProceedings of Engineering*, vol. 4, no. 2, p. 1988–1995, 2017.

[2] N. Magrofuoco, P. Roselli and J. Vanderdonckt, "Two-dimensional Stroke Gesture Recognition: A Survey," *ACM Comput Surv*, vol. 54, no. 7, 2022.

[3] R. D. Vatavu, L. Anthony and J. O. Wobbrock, "Gestures as point clouds: A $p recognizer for user interface prototypes," in *Proceedings of the ACM International Conference on Multimodal Interaction*, 2012.

[4] A. Danawan, J. Pragantha and D. A. Haris, "Pembuatan Game Arcade 'City Protector' Dengan Menggunakan Touch Gesture Recognizer," *Jurnal Ilmu Komputer dan Sistem Informasi*, vol. 8, no. 2, p. 217–223, 2021.

[5] C. Nofita and N. Amalia, "Studi Pemanfaatan Game Edukasi Belajar Angka dan Mudah Berhitung (Bamber) Berbasis Android Terhadap Kemampuan Berhitung Anak Autis," *Jurnal Pendidikan Khusus*, vol. 11, no. 3, 2019.

[6] B. B. Marklund, H. Engström, M. Hellkvist and P. Backlund, "What Empirically Based Research Tells Us About Game Development," *The Computer Games Journal*, vol. 8, no. 3–4, p. 179–198, Dec 2019.

[7] M. Borg, V. Garousi, A. Mahmoud, T. Olsson and O. Stalberg, "Video Game Development in a Rush: A Survey of the Global Game Jam Participants," *IEEE Trans Games*, vol. 12, no. 3, p. 246–259, Sep 2020.

[8] M. M. Attoyibi, F. E. Fikrisa and A. N. Handayani, "The Implementation of A Star Algorithm (A*) In the Game Education About Numbers Introduction," in *Proceedings of the 2nd International Conference on Vocational Education and Training (ICOVET 2018)*, 2019.

[9] N. L. G. K. Widiastuti, Konsep Dasar Matematika SD untuk PGSD, Jakarta: Kencana, 2019, p. 123–126.

[10] H. F. H. Su, F. Bell, D. Gates, J. Haramis, K. Hjerpe, C. Manigat and L. D. Silva, "Using Number Properties to Inspire Teaching and Learning in The K-12 Classrooms," *Transformations*, vol. 5, no. 1, 2019.

[11] I. Syahrudin, "Pengembangan Game Android sebagai Media Belajar Tulis dan Mengenal Huruf untuk Anak Menggunakan Pengenalan Pola Gesture dengan Metode Jaringan Saraf Tiruan Backpropagation," 2017. [Online]. Available: http://eprints.uty.ac.id/686/.

[12] Susilawati, "Algoritma Restricted Boltzmann Machines (RBM) untuk Pengenalan Tulisan Tangan Angka," in *Seminar Nasional Teknologi Informatika*, 2017.

[13] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh and B. Yoon, "Improved handwritten digit recognition using convolutional neural networks (Cnn)," *Sensors (Switzerland)*, vol. 20, no. 12, p. 1–18, Jun 2020.

[14] Y. Udjaja, "Gamification Assisted Language Learning for Japanese Language Using Expert Point Cloud Recognizer," *International Journal of Computer Games Technology*, 2018.

[15] M. Jazouli, A. Majda and A. Zarghili, "A $P Recognizer for Automatic Facial Emotion Recognition using Kinect Sensor," *Intelligent Systems and Computer Vision*, p. 1–5, 2017.

[16] KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI, "Modul Literasi Numerasi Di Sekolah Dasar," 2021. [Online]. Available: https://bersamahadapikorona.kemdikbud.go.id/tingkat-sd-modul-belajar-literasi-numerisasi/.