Jurnal String Vol. 3 No.1 Agustus 2018 **p-ISSN: 2527 - 9661**
e-ISSN: 2549 -2837

IMPLEMENTASI ALGORITMA COLLISION DETECTION PADA GAME SIMULATOR DRIVING CAR

Lui Haekal Fasha[1], Fauziah[2], M. Gufroni[3]
Teknik Informatika, Fakultas Teknologi Komunikasi dan Informatika, Universitas Nasional
Jalan Sawo Manila, Pasar Minggu, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12520
Telp (021) 7806700 Faks. (021) 7802718
luihaekal26@gmail.com[1], fauziah@civitas.unas.ac.id[2], m.guroni@gmail.com[3]

Abstrak

Saat ini banyak sekali *game engine* yang dapat digunakan untuk merancang dan membuat game salah satunya adalah *game engine* Unity 3D, Unity 3D merupakan sebuah *game Engine* yang banyak sekali digunakan. Saat ini untuk perancangan atau pembuatan *game* tidak hanya personal bahkan beberapa *depelover*pun menggunakannnya untuk sebuah produksi *game* nya. Algoritma *Collision Detector* adalah sebuah metode pendeteksian ketika dua objek atau lebih yang bertabrakan dan akan menimbulkan sebuah reaksi, metode ini sangat banyak digunakan dalam dunia pembuatan animasi maupun pembuatan *game*. Penulis menggunakan metode ini untuk perancangan sebuah game Simulasi berkendara, ketika player menabrak sebuah objek *obstacle* / penghalang yang sudah disusun pada sebuah lintasan agar mendapatkan sebuah reaksi pada salah satu objek yang ditentukan. Dari hasil penelitian ini dapat disimpulkan bahwa *Fps* yang didapatkan pada game ini mencapai titik terbaik dikisaran 60 *Frame Per Second* dan titik terburuknya berada dikisaran 24,6 *Frame Per Second.*
Kata Kunci :*Untiy 3D, Collision Detector, FPS*

Abstract

Currently, there are so many game engines that can be used to design and create games. One of them is Unity 3D. It is now a game engine widely used not only by a person to design or create games but also by some developers to produce games. Collision Detector Algorithm is a method of detecting reaction produced by a collision between two or more objects. The method is primarily used in the animation and game production. The researcher uses this method to design a driving simulation game, in which a player hitting obstacles that have been stacked on a track will experience reaction from one of the specified obstacles. From the results of this study, it can be concluded that the Fps obtained in this game reaches its best point at around 60 Frame Per Second and its worst point at around 24.6 Frame Per Second.
Keywords: *Game, Untiy 3D, Collision Detector, FPS*

1. PENDAHULUAN

Seiring berkembangnya kemajuan teknologi. *Game-game* menjadi sangat mudah didapatkan melalui internet, *game* menjadi pilihan utama untuk mengisi waktu senggang atau untuk sekedar melepas ketegangan setelah bekerja.Dalam suatu *game*, biasanya dikembangkan untuk kalangan remaja bahkan dewasa sekalipun untuk hiburan. *Game* biasanya dimainkan lebih dari satu pemain dalam waktu yang bersamaan (*Game Online*) tetapi juga bisa dimainkan sendiri (*Stand Alone Game*). Metode *collision detection* merupakan salah satu metode yang bisa digunakan untuk memeriksa suatu kondisi objek apakah terjadi tubrukan objek atau tidak, ketika terjadi tubrukan objek maka beberapa proses bisa dieksekusi dan menghasilkan kembali berupa informasi yang diketahui oleh pengguna sistem [1][2]

Setiap permainan menerapkan *collision detection* (deteksi tabrakan), baik itu dalam hal tabrakan antara *sprite* dengan

*sprite*maupun antara *sprite* dengan peluru dan lain-lain. Proses *collision* dapat dibagi menjadi dua kategori dasar, yaitu *collision detection* dan *collision response*, dengan jarak respon yang telah diaplikasikan secara spesifik. Terdapat banyak sekali jenis dari *collision detection* itu sendiri [3].

Sekarang ini banyak sekali *software* - *software* yang dirancang untuk membantu dalam proses pembuatan maupun pengembangan game khususnya pada pembuatan game 3D yang bias adisebut *game engine*. Dengan bantuan *game engine* seseorang dapat membuat game sesuai dengan kemauan sendiri. Keuntungan lainnya dari game engine adalah tidak diperlukann yatim dalam pembuatannya dan dana yang besar. Salah satu game engine yang sering digunakan saat ini adalah unity 3D tidak hanya perorangan namun beberapa developer pun menggunakan game engine ini untuk projek game mereka walaupun dengan versi yang berbeda. Pada proses pembuatan game, sebaiknya menggunakan system informasi supaya nantinya game akan berbobot dan akan lebih terspesifik sehingga pada saat pembuatan game tidak melewati tahapan - tahapan yang semestinya. Tahapan itu terdiri dari analisa system kebutuhan, perancangan desain, pengkodean, pengujian dan maintaincenya.

Tujuan penelitian ini adalah merancang sebuah *game* simulator berkendara berbasis pc dengan *software* unity3D, dimana sang *user/player* harus menghindari beberapa *obstacle* yang ada demi mencapai *finish*. *Game* ini juga memberikan kesempatan penulis untuk mengembangkan kemampuan berkreatifitas ditengah perkembangan dunia *game*. Penerapan konsep algoritma *Collision Detection* diharapkan dapat menjadi pemicu reaksi dari NPC *(Non Player Character)* dan objek penghalangnya.

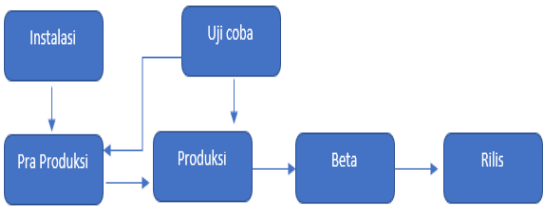
Dalam menentukan perilaku harus berdasarkan dari nilai probabilitas penghitungan bayesian. Jika nilai probabilitas yang paling tinggi dimiliki oleh kategori *smash* maka perilaku NPC *(Non Player Character)* akan masuk kategori *smash*. Tapi jika nilai probabilitas yang paling tinggi dimiliki oleh kategori *kick* maka perilaku tidak akan memilih perilaku *kick*, melainkan perilaku akan mengecek sesuai dengan metode *collision detection*, apakah *collider* yang dimiliki NPC *(Non Player Character)* menyentuh *collider player*, jika *collider* NPC *(Non Player Character)* terbukti menyentuh *collider player*, NPC*(Non Player Character)*akan mengambil keputusan untuk memilih perilaku *kick*[6].

Unity3D ini akan berjalan maksimal jika spesifikasi laptop atau pc yang di gunakan sesuai, spesifikasi minimal yang dibutuhkan *software / game engine* ini sebagai berikut : *Processor Intel Core i3-2370M. Microsoft Windows 10 64-Bit. Graphic Card n Vidia Geforce 610M 5.1.1f1. Memory (RAM) 2 GB* [5].

*Game*ini di rancang penulis agar *user* dapat menyelesaikan permasalahan yang ada, karena *game*s angat mudah di terima di semua kalangan.

2. METODE PENELITIAN

A. Kerangka Penelitian

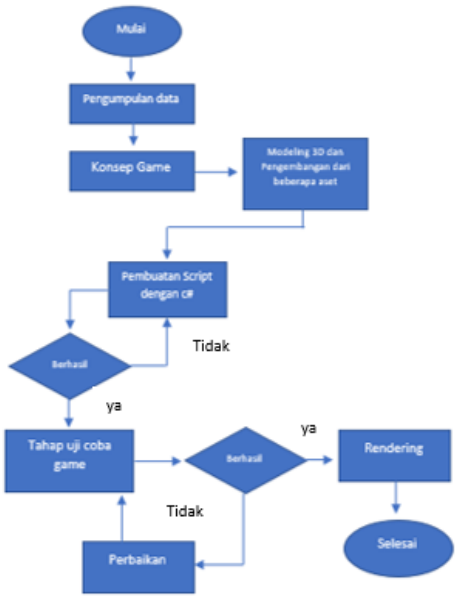


Gambar 1.Tahapan GDLC

Pada gambar 1 merupakan tahapan yang digunakan dalam pembuatan *game* ini penulis menggunakan metode GDLC

(*Game Development Life Cycle*) yang tahapan pertamanya berupa inisiasi yang merupakan perumusan ide dari sebuah *game* nya, lalu tahapan selanjutnya adalah Pre- Produksi yang merupakan tahapan dari *desain game*. Setelah Pre – Produksi selesai dilakukan masuk ketahap Produksi dimana proses mulai pengkodingan, alur dari sebuah gamenya agar dapat dimainkan. Setelah selesai maka akan masuk ketahap uji coba untuk menemukan *buggame* dan kekurangan – kekurangan lainnya agar diperbaiki. Tahapan akhirnya adalah rilis *game* yang sudah selesai dibangun melalui tahapan – tahapan sebelumnya.

B. Tahapan Perancangan *Game*



Gambar 2.*Flowchart* Perancangan *Game*

Untuk merancang sebuah *game* membutuhkan beberapa tahapan yang harus dilaksanakan dengan benar agar nantinya *game* yang dibuat sesuai dengan rancangan yang ada dan berjalan dengan baik. Dalam pembuatan *game simulator driving car* ini melalui beberapa tahapan seperti perumusan ide, desain *game*, pengkodingan, Pembuatan alur perjalan gamenya, tahap uji coba, tahap perbaikan *bug*, dan rendering *game* itu sendiri. Gambar 2 merupakan gambaran dari diagram tahapan game yang dibuat untuk menghasilkan *gamesimulator driving car.*

C. Algoritma *Collision Detector*

Algoritma *collision detection*merupakan proses pengecekan apakahbuah objek spasial salingbertumpuk atau tidak. Jika ternyata adapaling sedikit dua buah objek yangbertumpuk, maka kedua objek tersebutdikatakan saling bertumpukkan. Pada ruangspasial dua dimensi objek yang bertumpukberarti objek spasialnya beririsan. *Collision detection* merupakan teknik deteksi tabrakan untuk mengetahui objek-objek apa saja yang bersentuhan dalam bidang koordinat tertentu. [2]

Metode ini menggunakan sensor pemicu (*triger*) untuk mendeteksi keberadaan objek ketika berada di lingkungan virtual. Mekanisme yang mendukung fitur ini adalah area pemicu (sensor) yang ditambahkan di sekeliling mobil. Sensor yang ditambahkan akan dihitung secara matematis untuk menentukan apakah sensor mengenai objek lain seperti tembok (*unwalkable object*), kendaraan lain, atau objek-objek lain yang berada di lingkungan virtual. [5]

D. Unity 3D

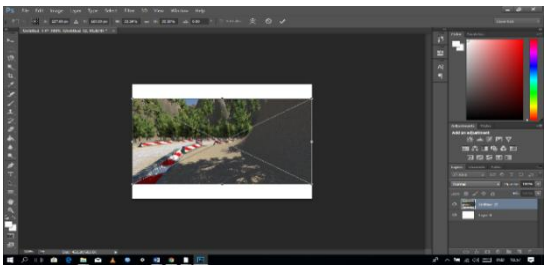
Unity 3D adalah sebuah *game engine* yang sangat banyak digunakan saat ini bahkan beberapa *developer game* juga menggunakan *software* ini, Unity 3D sudah memiliki menu *packages* yang cukup lengkap. Penulis menggunakan Untiy 3D untuk medesain latar tempat dan beberapa bagian *game* lainnya.

Gambar 3.Software Unity3D

Gambar 3 adalah salah satu contoh tampilan menu *packages* yang terdapat dalam *software* unity3D pada proses desain t*errain* / latar.

E. Adobe Photoshop CC

Adobe Photoshop merupakan sebuah *software desain*, kali ini penulis menggunakan *software* ini untuk membuat *desain background* menu dan desain menu yang nantinya akan di import kedalam *software* Unity3D .

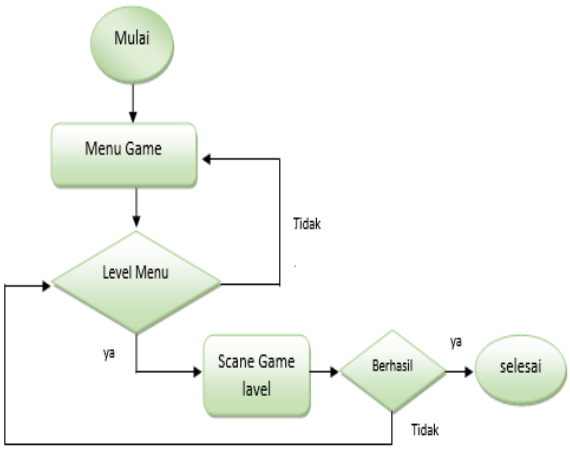


Gambar 4.*Softwar*e AdobePhotoshopCC

Pada gambar 4 merupakan tampilan dari *software Adobe Photoshop* CC yang digunakan dalam tahap desain *background* untuk tampilan menu pada *game.*

F. Alur *Game*

Game ini dibuat agar *user* dapat melewati dan menghindari beberapa *obstacle* / penghalang yang disusun oleh sang creator dan sampai di garis finish, namun untuk melewati *obstacle* yang ada *user* harus mengendarai mobil yang disediakan *creator* dan mengatur kecepatan untuk mencapai garis *finish* sebelum masuk kedalam *scane* simulator player akan melewati tahapan seperti *flowchart* pada gambar 5.



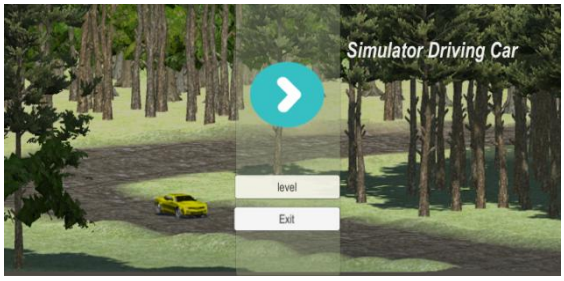
Gambar 5.*Flowchart Game*

Flowchart pada gambar 5 merupakan alur dari *game* ini, dimana setelah melewati *scene loading player* akan masuk ke *scene* berikutnya yaitu *scene* menu *game* yang di dalamnya terdapat 2 *button* yaitu menu *level* dan *exit.* Setelah masuk dalam *scene level* menu, *player* akan diberi 3 pilihan *level* mulai dari *easy, medium* dan *hard* saat masuk dalam salah satu *scene gamelevel,player* diharuskan menyelesaikan *test*. Ketika berhasil melewati setiap tahapan *level* maka *game* ini bisa dikatakan selesai.

3. HASIL DAN PEMBAHASAN

A. Desain Menu

Gambar 4 adalah tampilan menu dari *game* yang di buat, di dalam tampilan menu awal terdapat 2 *button* Level and *Exit*. Button Level adalah button untuk menampilkan pilihan level di tampilan menu berikutnya. Sedangkan *buttonExit* berfungsi untuk keluar dari *game* tersebut.



Gambar 6.Tampilan Menu game

Gambar 6 merupakan tampilan dari menu *game* awal yang di dalamnya terdapat *button level* dan *Exit*.

B. Skema Permainan

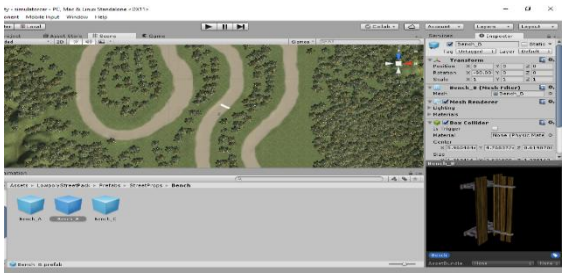
Tujuan dari *game* ini adalah simulasi berkendara, untuk menyelesakan 1 *level player* diharuskan mencapai *finish* dengan cara melewati dan menghindari *obstacle* yang ada padalintasan, jika *playe*r menabrak atau mengenai halangan / *obstacle* yang ada *player* akan kembali ke *start* awal dan begitu seterusnya. Untuk *levelgame* Ada 3 *level* yang di buat yaitu *Easy, Medium, Hard*.

Gambar 7.Tampilan *Level Game*

Gambar 7 adalah tampilan dari salah satu *game level* yang terdapat pada game ini.

C. *Pengembangan Asset*

Dalam software unity3D memiliki fitur asset store dimana user dapat mengunduh propertis / asset free maupun tidak, yang dibutuhkan dalam projek *game* nya. Dalam pembuatan *game* ini ada beberapa componnet yang digunakan dari asset yang diunduh melalui aset strore yang ada di unity untuk memperindah tampilan dari game ini agar terlihat lebih real.



Gambar 8.Objek unduhan *Asset Store*

Gambar 8 adalah beberapa *modeling* 3D objek yang saya ambil dari *asset store* di unity3D untuk melengkapi objek yang dibutuhkan dalam *game* ini.

D. *Analisis Kebutuhan Perangkat*

Dalam merancang atau membuat *game* pasti dibutuhkan perangkat – perangkat pendukungnya. Dalam penelitian ini penulis menggunakan beberapa perangkat keras dan perangkat lunak dalam proses perangcangan *game*nya, yaitu sebagai berikut :

- Kebutuhan Perangkat Lunak

Tabel 1.Tabel Spesifikasi Perangkat Lunak

NamaPerangkat
Unity 3D 5.6.1
Adobe Photoshop cc
Mono Develop
Windows 10 Home 64-bit

Tabel 1 adalah spesifikasi perangkat lunak apa saja yag dibutuhkan dalam game ini mulai dari tahapan desain game bahkan sampai tahap pengkodingan.

- Kebutuhan Perangkat Keras

Tabel 2. Tabel Spesifikasi Perangkat Keras

Perangkat	Spesifikasi
Prosesor	Intel Core i5 7200U 2.71GHz
RAM	4 *Gigabyte*
VGA	2 *Gigabyte*

Tabel 2 adalah spesifikasi perangkat keras yang dibutuhkan untuk perancangan game Simulator Driving Car ini.

E. Pengujian Fungsional Game

Uji Fungsional *game* dilakukan bertujuan agar kita dapat mengetahui apakah *game* sudah sesuai dengan rancangan yang ada dan jika saat melalui tahap uji coba terdapat beberapa masalah fungsi dapat segera diperbaiki dengan cepat. Dalam Penelitian ini ada 3 tahapan pengujian mulai dari pengujian alur *game* pada setiap *scenenya* agar mengetahui alur *game* berjalan dengan baik dan benar. Tahapan kedua adalah memeriksa grafik *frame* pada setiap *device* dengan menggunakan tools mini profiler, Tahapan terakhir pengujian *kontrollergame* pada setiap fungsi *button* dan kontrol agar tidak terjadi *crash* saat dimainkan.

- *Pengujian Delay scene Respon*

Pengujian delay scene respon dilakukan secara manual dengan menggunakan stopwach untuk mengetahui *dellay* dari *scane* 1 ke *scene* lainnya di setiap perangkat yang berbeda.

Tabel 3 Estimasi Waktu Respon Perangkat

Tipe	Processor	RAM	Estimasi waktu
Acer E451G	AMD A8	4 GB	1 Second
Asus A456U	Core i5	4GB	0,57 Second
Asus X550Z	AMD R7	4GB	0,35 second

Tabel 3 adalah tabel hasil estimasi respon perangkat untuk pergantian pada setiap scene yang ada dalam *game* ini.

- *Penelitian FPS (Frame Per Second)*

Dalam penelitian ini untuk mengukur frame per second penulis menggunakan asset mini profiler yang terdapat dalam asset store, kemudian melihat data yang muncul pada setiap device yang berbeda.

Gambar 8.Tools Mini Profiler

Berdasarkan standar baru yang dibuat oleh ATSC(Advance Television Systems Committee), FPS terbaik yang didapatkan dalam standar video 1080p berada pada 50 sampai 60 frames persecond.[7]

Tabel 4. Spesifikasi penelitian FPS Device

Tipe	Processor	RAM	Sistem Operasi
Acer E451G	AMD A8	4 Gb	Windows 7 64-bit
Asus A456U	Core i5	4Gb	Windows 10 Home 64-bit
Asus X550Z	AMD R7	4GB	Windows 10 Home 64-bit

Tabel 4 merupakan tabel spesifikasi perangkat keras yang digunakan dalam pengujian game ini dari sisi *Frame Per Secondn*ya.

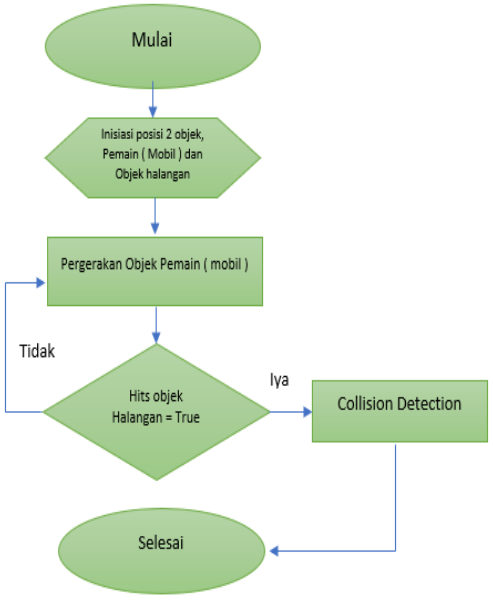
Tabel 5. Hasil Percobaan FPS Device

Tipe	FPS Terburuk	FPS Tebaik	Sistem Operasi
Acer E451G	24,6	58,7	Windows 7 64-bit
Asus A456U	44,3	59,4	Windows 10 Home 64-bit
Asus X550Z	51,9	60,0	Windows 10 Home 64-bit

Dari Tabel diatas adalah hasil dari tahapan pengujian fungsional *Frame Per Second* pada setiap perangkat yang tertera.

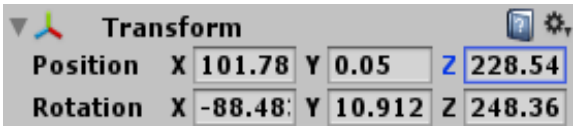
- *Pengujian Algoritma Collision Detection*

Dalam *game* ini penerapan algoritma *Collision Detection* terdapat pada dua buah objek agar menghasilkan reaksi untuk kembali ketitik *start*. Objek yang dipilih adalah NPCnya yaitu mobil sebagai objek utamanya dan *Obstacle* nya.



Gambar 8. Flowchart Algoritma *Collision Detection*

Untuk memastikan algoritma *Collision Detection* dapat berfungsi dengan baik maka penulis melakukan pengujian dengan Percobaan responsif objek di setiap kordinatnya ketika berbenturan.



Gambar 9. Kordinat Objek Di Unity3D

Untuk mengetahui letak kordinat objek dalam game ini peneliti menggunakan kordinat berdasarkan tools yang di siapkan oleh unity3D dan Gambar 9 merupakan contoh letak salah satu objeknya.

Tabel 6. Hasil Percobaan Responsif Objek

No	Kordinat Objek 1 (Mobil)	Kordinat Objek 2 (Road Block)	Reaksi kordinat yang di hasilkan
1	X (246.5), y (0.12), z (328.3)	X (246.5), y (0.12), z (328.3)	X (179.7), y (-0.198), z (317.98)
2	X (315.2), y (-0.1), z (347.1)	X (315.2), y (-0.1), z (347.1)	X (179.7), y (-0.198), z (317.98)
3	X (341.3), y (-0.1), z (296.1)	X (341.3), y (-0.1), z (296.1)	X (179.7), y (-0.198), z (317.98)
4	X (373.6), y (0.20), z (236.6)	X (373.6), y (0.20), z (236.6)	X (179.7), y (-0.198), z (317.98)
5	X (345.5), y (0.34), z (155.7)	X (345.5), y (0.34), z (155.7)	X (179.7), y (-0.198),z (317.98)
6	X (212.6), y (-0.16), z (164.3)	X (212.6), y (-0.16), z (164.3)	X (179.7), y (-0.198), z (317.98)
7	X (130.9), y (0.11), z (152.6)	X (130.9), y (0.11), z (152.6)	X (179.7), y (-0.198), z (317.98)
8	X (101.7), y (0.05), z (228.5)	X (101.7), y (0.05), z (228.5)	X (179.7), y (-0.198), z (317.98)

Tabel 6 merupakan tabel hasil pengujian responsif objek terhadap algoritma yang diterapkan pada setiap titik letak kordinatnya untuk mengetahui reaksi yang ditimbulkan ketika 2 objek tesebut bebenturan. Pengujian ini dilakukan sebanyak 8 kali pada setiap titik kordinat objek yang berbeda.

```
4
5 public class TabrakObjek : MonoBehaviour {
6
7     [SerializeField] public Transform karakter;
8     [SerializeField] public Transform CekPoint;
9
10    void OnTriggerEnter (Collider other)
11    {
12        karakter.transform.position = CekPoint.transform.position;
13    }
```

Gambar 10.Script Fungsi Algoritma

Gambar diatas merupakan potongan script fungsional algoritma *Collision Detection* yang menghasilkan reaksi kembalinya player ketitik awal ketika menabrak sebuah objek penghalang yang ada pada area lintasan.

4. SIMPULAN

Mengacu jurnal – jurnal sebelumnya dan hasil dari penelitian perancangan game simulator driving car dan melalui beberapa tahapan pengujian, Penulis menyimpulkan :

1. *Game SimulatorDriving Car* sudah dapat berjalan sesuai dengan perancangannya.
2. Penerapan Algoritma *Collison Detection* pada setiap objek halangan atau *obstacle* sudah berhasil menimbulkan reaksi, yaitu kembalinya *player* ke titik *start* untuk mengulangi tes *driving* seperti hasil yang dihasilkan pada tabel 6 .
3. Untuk *standart* yang mengacu pada ATSC*(Advance Television Systems Committee*), game ini sudah memenuhi *standart* karena fps yang dihasilkan mencapai titik 60 *frame per second*.
4. Penerapan fungsional script objek telah berhasil menimbulkan reaksi pada setiap objeknya.
5. Untuk estimasi waktu respons per *scane* dari awal menu sangat bergantung pada spesifikasi Ram di setiap *Devicenya* dalam pengujian perangkat keras peneliti menggunakan perangkat dengan spesifikasi 4 GB seperti pada tabel 4.

DAFTAR PUSTAKA

[1] Avril, Q. Gouranton, V. & Arnaldi, B. (2014). Collision Detection: Broad Phase Adaptation From. *Journal Of Virtual Reality And Broadcasting, Volume N(200n), No. N*, 1-14.

[2] Bade, A., Ping, C. S., & Tanalol, S. H. (2015). Collision Detection For Cloth Simulation Using Bounding Sphere Hierarchy. 1-5.

[3] Bhaskara, S. G., Buana, P. W., & Purnawan, I. K. (2017). Permainan Edukasi Labirin Dengan Metode Collision Detection Dan Stereoscopic. *Lontar Komputer Vol. 8, No. 2*, 65-76.

[4] Heidelberger, B., Teschner, M., & Gross, M. (T.Thn.). Detection Of Collisions And Self-Collisions. 7-8.

[5] Wibawanto, W. (2017). Metode Trigger Detection Untuk Gerakan Kendaraan NPC Dalam Game. *Journal of Animation and Games Studies, Vol.3 No.1*, 32

[6] Asmiatun, S. (2016). Penerapan Algoritma Collision Detection Dan Bayesian Pada Npc (Non Player Character) Menggunakan Unity 3d. *Jurnal Transformatika, Volume 14*, 6-11.

[7] ATSC. (2015). "ATSC Standard A/72 Part 1 – VideoSystem Characteristics of AVC in the ATSC DigitalTelevision System".[Online] Tersedia:https://www.atsc.org/standard/a72-parts- 1-2- and-3/.[23 November 2017].