RANCANG BANGUN GAME STICKMAN DENGAN METODE QUADTREE

[1]Rico Fernando, [2]Jusia Amanda Ginting

Program Studi Teknik Informatika, Fakultas Teknik dan Desain, Universitas Bunda Mulia
Jl. Lodan Raya No: 2, Ancol, Jakarta Utara 14430
e-mail: [1]32190002@student.ubm.ac.id, [2]gintingjusia2@gmail.com

Abstrak

Metode quadtree merupakan metode pembagian wilayah menjadi 4. Metode quadtree dapat diterapkan untuk mempermudah collision detection pada object collidable. Collision detection akan dilakukan pada wilayah terkecil dari pembagian wilayah quadtree karena terbatasnya kecepatan dalam collision detection. Masalah tersebut menjadi masalah dimana untuk menghasilkan game dengan pendeteksian secara optimal dilakukan pemecahan wilayah dengan metode quadtree dengan membagi wilayah menjadi 4 hingga object collidable pada tiap wilayah tidak melebihi batasan yang ditentukan, sehingga memperkecil bagian yang perlu di periksa tanpa harus memeriksa secara keseluruhan. Metode yang digunakan dalam pengembangan adalah quadtree. Metode ini cocok diterapkan pada game yang memerlukan collision detection didalamnya. Dari penelitian ini dihasilkan bahwa metode quadtree dengan membatasi area deteksi tubrukan dapat menghasilkan kecepatan lebih optimal yang dilihat dari frame per second yang dihasilkan. Game yang dibangun diharapkan dapat mengoptimalkan permainan sejenisnya yang menggunakan collision detection didalamnya. Disarankan melakukan penelitian lebih lanjut dengan variasi jenis game lainnya.

Kata Kunci: *Game Stickman*, *Quadtree*, *Game Platformer*, *Collision Detection.*

Abstract

Quadtree method is a method of dividing region into 4. Quadtree method can be applied to make collision detection easier for collidable objects. Collision detection will be done in the smallest region of the quadtree division because of the limited speed in collision detection. This problem becomes a problem where to produce a game with optimal detection is done by dividing the region with quadtree method by dividing the region into 4 until the collidable objects in each region does not exceed the predetermined limits, thus reducing the parts that need to be checked without having to check as a whole. The method used in development is quadtree. This method is suitable to be applied to games that require collision detection in it. From this study it is obtained that the quadtree method by limiting the collision detection area can produce an optimal speed seen from the frame per second produced. The game built is expected to be able to optimize similar games that use collision detection in it. It is suggested to do further research with various types of other games.

Keywords: Stickman Game, Quadtree, Platformer Game, Collision Detection.

Pendahuluan

Game atau yang biasa disebut dengan video game merupakan salah satu hiburan dengan aturan tertentu yang diminati banyak orang. Ada banyak jenis game, dari game puzzle hingga game perang. Game dapat dimainkan dari berbagai kalangan dari orang dewasa hingga anak-anak. Dalam penggunaannya, video game adalah permainan elektronik dan visual, dan video game menggunakan media visual dengan multimedia yang menggabungkan animasi, video, gambar, dan suara. Pada masa kini masih banyak game yang memiliki performa kurang baik seperti contohnya game Way Of The Warrior dimana mendapat kritik atas performa yang buruk, salah satu penyebabnya adalah pada pendeteksian tubrukan antar objek yang dideteksi menggunakan metode yang kurang optimal terutama pada objek dinamis, maka dilakukan pengujian untuk menghasilkan performa lebih baik yang diterapkan pada game stickman.

Game memiliki beragam jenis, beberapa diantaranya adalah game strategi, game teka-teki, game action, game adventure dan game platformer. Game masa kini yang banyak diminati adalah game-game online bergenre strategi, namun pada beberapa kalangan juga menyukai game offline dengan genre game platformer seperti game Mario Bross, serta salah satu diantaranya adalah game Stickman.

Stickman merupakan salah satu game yang akan dibuat menggunakan algoritma *collision* detection dengan metode *quadtree*. Tokoh utama game ini diambil karena keunikan dengan bentuk seperti tongkat. Game ini bertujuan untuk bertahan hidup selama mungkin untuk meraih score dengan mengatasi setiap rintangan yang ada dan mendapatkan skor tertinggi atau *High Score* lalu masuk ke pintu menuju level selanjutnya. Karakter dapat melompat dan menembak untuk menghindari rintangan yang akan diatasi dan dapat mengambil beberapa item pendukung seperti item bantuan yang disertakan dalam setiap perjalanan. Jika karakter jatuh saat menghadapi rintangan atau mati karena serangan musuh, permainan akan berakhir. Jika diibandingkan terhadap game sejenis lainnya, game ini memiliki perbedaan pada karakter yaitu menggunakan stickman sebagai karakter utama, dan game ini tidak hanya dimainkan dengan melewati rintangan dengan berlari dan melompat, game ini juga memiliki variasi menggunakan senjata sebagai bantuan yang mana tubrukan terjadi antara peluru terhadap musuh.

Algoritma *collision* detection adalah sebuah proses pendeteksian antara dua buah objek atau lebih mengalami bentrokan atau tidaknya, *collision detection* ini juga biasa disebut dengan sebuah teknik pendeteksian tubrukan antar objek untuk mengetahui tubrukan pada bidang wilayah tertentu (Haekal Fasha & Gufroni, 2018). Dalam penelitian ini, metode yang digunakan adalah metode pemisahan *quadtree* untuk mendeteksi tumbukan dari pengelompokan setiap objek yang dibatasi pada setiap node dalam permainan stickman 2D.

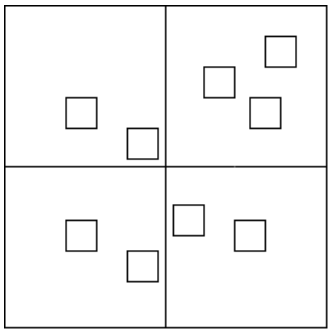
Quadtree adalah sistem pembagian wilayah menjadi 4 bagian seperti halnya percabangan pada pohon yang dapat mempermudah pendeteksian tubrukan objek pada tiap wilayahnya (Tandean et al., 2019), jika sebuah game 2 dimensi, pada pendeteksiannya menggunakan metode bruteforce atau yang biasa tertera secara default pada editor akan memakan waktu lama karena terjadinya pendeteksian secara menyeluruh tanpa terfokuskan, sedangkan metode quadtree lebih cocok digunakan pada pendeteksian game 2 dimensi karena pada dasarnya game 2 dimensi memiliki 2 daerah sumbu x dan y saja sedangkan untuk game 3 dimensi akan lebih baik menggunakan metode *octree* degan 3 sumbu x,y dan z. pada game stickman 2D metode ini akan diterapkan untuk mengoptimasi pendeteksian dari algoritma *collision detection* untuk memperkecil jangkauan yang perlu dilakukan pengecekan tubrukan antar objek seperti peluru terhadap musuh.

Pembahasan

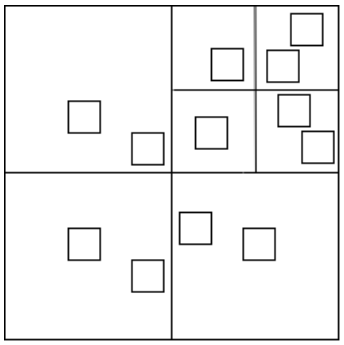
Pada hasil Penelitian terdahulu disebutkan bahwa kondisi dimana mengetahui objek yang saling bertuburukan atau bersentuhan satu sama lain disebut collision detection, serta menghasilkan bagaimana quadtree dapat digunakan sebagai sebuah metode untuk mengoptimasi collision detection pada proses perhitungan untuk mengetahui tubrukan (Indra et al., 2019).

Metode quadtree merupakan sebuah pohon m-ray yang setiap simpulnya memiliki 4 cabang, metode quadtree berawal dengan membagi area menjadi 4 bagian atau biasa disebut node, kemudian akan dideteksi objek yang ada pada bagian node yang terbagi dengan batasan objek yang diperbolehkan pada satu node. Jika pada node mempunyai

objek lebih daripada yang diperbolehkan maka node tersebut akan membentuk 4 bagian, skema ini akan dilakukan secara berulang seperti pada gambar 1 dan 2.



Gambar 1. Parent Quadtree

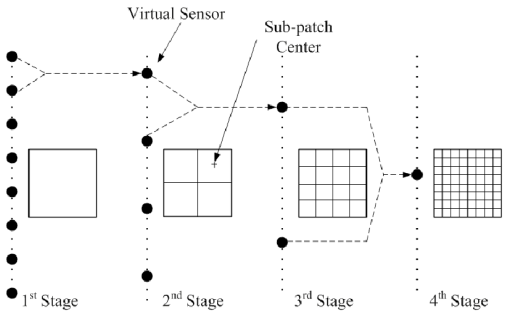


Gambar 2. Child Quadtree

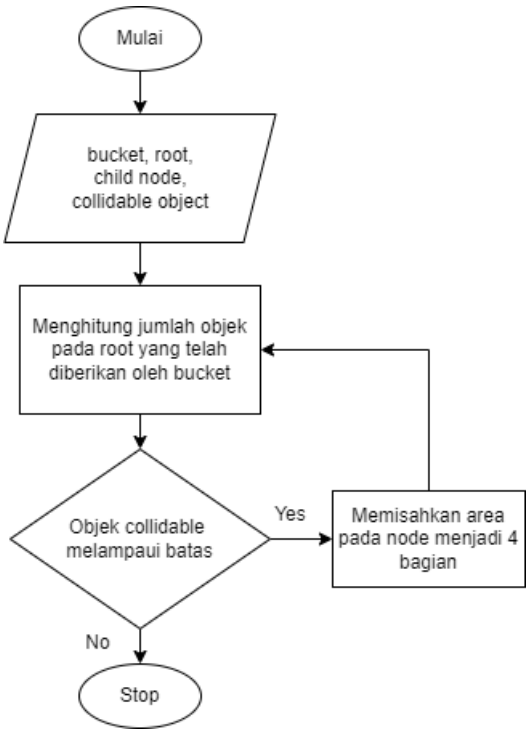
Pada gambar dijelaskan bahwa objek yang terbagi di beberapa node dengan kapasitas 2 objek, node dengan objek melebihi kapasitas akan terbagi menjadi child node dengan 4 daerah untuk menyimpan objek yang ada.

Perancangan proses

Permainan game stickman membutuhkan *collision detection* untuk mendeteksi tabrakan antara satu objek dengan objek lain, pendeteksian objek yang bertabrakan disederhanakan dengan menggunakan *quadtree*, pohon bercabang ini akan membagi 4 wilayah (cabang) dan cabang dari induk akan terus bercabang 4 (anak) sehingga tubrukan pada objek dapat terdeteksi dengan lebih optimal. Untuk lebih jelasnya dapat melihat gambar 3.



Gambar 3. Gambaran cara kerja quadtree



Gambar 4. Flowchart quadtree

Seperti yang tertera pada gambar flowchart 4, untuk membuat pemisahan *quadtree* diperlukan sebuah *collidable* objek, dalam hal ini objek yang dimaksud adalah player, npc, ataupun *obstacle*, kemudian dibutuhkan posisi objek untuk mengetahui dan mengatur di *node* mana objek tersebut akan disimpan.

Perancangan Tampilan

Tampilan dari antarmuka game yang dirancang:

1. Tampilan Menu

Gambar 5. Menu Game

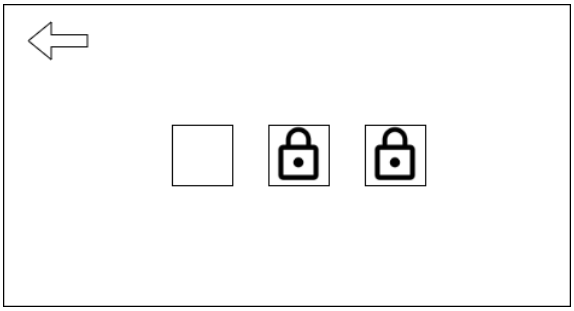
perancangan halaman menu berfungsi sebagai halaman pertama dari aplikasi, pada halaman ini terdapat beberapa tombol yaitu tombol start yang akan mengarahkan kepada halaman utama game/arena, tombol about yang merngarahkan kepada halaman tentang game, dan tombol exit untuk keluar dari aplikasi game stickman.

2. Tampilan About

Gambar 6. About Game

perancangan halaman about berfungsi untuk menampilkan berbagai informasi mengenai aplikasi, versi game serta informasi pengembang dari game stickman, pada halaman ini terdapat tombol back untuk kembali ke halaman menu.

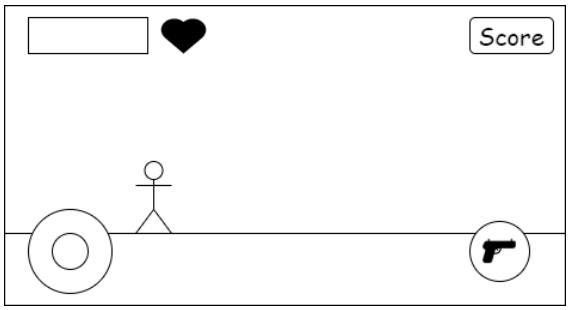
3. Tampilan Level



Gambar 7. Level Game

perancangan halaman level berfungsi untuk menampilkan berbagai level dari game, tingkatan dari tiap level akan berbeda beda, pada halaman ini terdapat tombol back untuk kembali pada menu utama game stickman.

4. Tampilan Utama Game

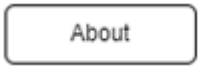
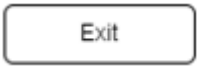


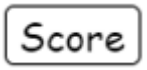
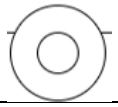
Gambar 8. Tampilan Utama Game

perancangan halaman tampilan utama game berfungsi sebagai arena permainan dimana game ini akan dimainkan. Pada halaman ini diberi sebuah kontroler untuk menggerakkan karakter dan sebuah tombol tembak untuk menyerang musuh, selain itu health/nyawa dari karakter ditampilkan dengan bar pada pojok kiri atas, dan score ditampilkan pada papan skor sebelah kanan atas.

Komponen-Komponen Game yang dirancang :

Tabel 1. Komponen-komponen game

No.	Komponen	Keterangan
1.	Tombol Play 	Tombol untuk memulai permainan.
2.	Tombol About 	Tombol untuk menampilkan mengenai game.
3.	Tombol Exit 	Tombol untuk keluar dari game.
4.	Tombol Back 	Tombol untuk kembali ke halaman sebelumnya
5.	Pemilihan Level 	Field pemilihan level pada game.
6.	Health Bar 	Tampilan nyawa player.
7.	Score Text	Papan skor pada game.

No.	Komponen	Keterangan
		
8.	Joypad 	Joypad untuk menggerakan karakter.
9.	Button Tembak 	Tombol tembak untuk menembak.
10.	Karakter 	Bentuk karakter utama dan musuh.
11.	Peluru 	Peluru yang dikeluarkan saat menekan tombol tembak.

Sumber: Data diolah

Hasil dan Pembahasan

Hasil yang didapatkan berupa penerapan antarmuka, metode dan hasil dari pengujian metode yang diterapkan dengan memperlihatkan perbandingan antara pendeteksian tanpa metode (default) dengan pendeteksian dengan metode.

Implementasi Antarmuka Pengguna

Implementasi antar muka (*User Interface*) ini adalah tampilam yamh akan ditampilkan oleh aplikasi kepada user, berikut adalah tanpilan antar muka dari aplikasi game stickman.

1. Tampilan Main Menu

Gambar 9. Tampilan Main Menu

Tampilan menu utama dimana terdapat beberapa button diantaranya:

- Button info yang menyajikan informasi mengenai pengembang aplikasi.

- Button play untuk memulai permainan.
- Button options untuk mengatur tinggi rendahnya backsound.
- Button exit untuk keluar dari permainan.

2. Tampilan About

Gambar 10. Tampilan About

Tampilan About yang menyajikan informasi pengembang dimana terdapat sebuah button:

- Button untuk kembali ke menu utama.

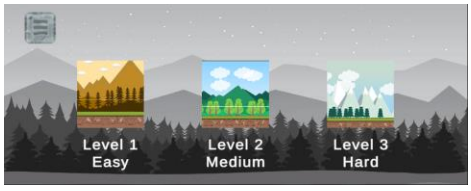
3. Tampilan Options

Gambar 11. Tampilan Options

Tampilan Options dimana berisi pengatur tinggi rendahnya suara game dengan sebuah slidebar :

- Slidebar untuk mengatur suara

4. Tampilan Select Level



Gambar 12. Tampilan Select Level

Tampilan select level yang menyajikan pilihan level dengan tingkatan kesulitan dalam game dengan menampilkan button diantaranya:

- Button kembali yang mengarah kembali ke menu utama.

- Button level 1 dengan tingkatan mudah akan mengarahkan pada arena level 1 game.
- Button level 2 dengan tingkatan menengah akan mengarahkan pada arena level 2 game.
- Button level 3 dengan tingkatan sulit akan mengarahkan pada arena level 3 game.

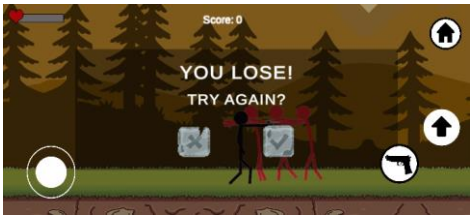
5. Tampilan Arena Game

Gambar 13. Tampilan Arena Game

Tampilan arena game yang menyajikan tampilan utama dari game, dimana terdapat karakter dan arena permainan dengan tampilan health bar serta score dan beberapa button diantaranya:

- Button fire untuk menembakkan peluru.
- Joystick pad untuk menggerakkan karakter.

6. Tampilan Kalah

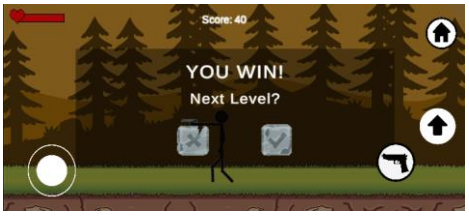


Gambar 14. Tampilan Kalah

Tampilan kalah yang menyajikan panel yang menyatakan bahwa pemain telah kalah dikarenakan karakter yang mati dengan nyawa 0, terdapat beberapa button diantaranya:

- Button silang yang akan mengarahkan ke select level.
- Button centang yang akan mengulang permainan pada level tersebut.

7. Tampilan Menang

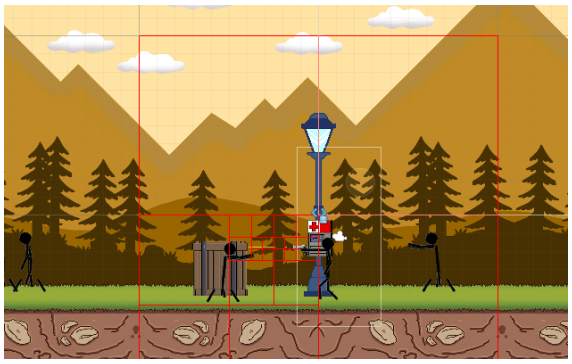


Gambar 15. Tampilan Menang

Tampilan panel yang menyatakan kemenangan pemain, dimana pemain diberikan pilihan untuk lanjut ke level selanjutnya atau kembali ke menu select level, panel ini terdiri dari beberapa button diantaranya:

- Button silang yang akan mengarahkan pada select level.
- Button centang yang akan mengarahkan pada level selanjutnya.

4.2 Implementasi Metode



Gambar 16. Gambaran Quadtree Pada Game

Gambar 16 memperlihatkan pemisahan quatree pada peluru yang mana daerah yang telah dilewati akan dihapus nodenya sehingga pada peluru yang bergerak dinamis dapat terdeteksi. Pada saat collidable object bertabrakan, metode OnCollision akan dipanggil. Quadtree memberi tahu posisi musuh dan peluru saat metode collision memeriksa apakah tabrakan perlu dipanggil atau tidak.

4.5 Pengujian Keberhasilan Metode

Pada subab ini dilakukan pengujian dalam *Collision Detection* pada Quadtree yang sudah diimplementasikan sebelumnya, pengujian yang dilakukan adalah *stress testing* pada collision detection yang diimplementasikan pada quadtree. Selain itu dilakukan *stress testing* terhadap metode *bruteforce*. Pengujian ini dilakukan untuk

mengetahui efektivitas dari penerapan metode quadtree untuk mengoptimalkan pendeteksian tubrukan.

Tabel 2. Pengujian Collision Detection dengan Quadtree

Pengujian ke-	Jumlah objek Collidable	Frame per second
1	2	193.9
2	4	149.1
3	6	120.8

Gambar 17. Pengujian 2 Object Quadtree

Pada gambar 17 dapat dilihat pengujian dengan menggunakan 2 collidable objek menghasilkan 193.9 *frame per second* pada area *quadtree*.

Gambar 18. Pengujian 4 Object Quadtree

Pada gambar 18 dapat dilihat pengujian dengan menggunakan 4 collidable objek menghasilkan 149.1 *frame per second* pada area *quadtree*.

Gambar 19. Pengujian 6 Object Quadtree

Pada gambar 19 dapat dilihat pengujian dengan menggunakan 6 collidable objek menghasilkan 120.8 *frame per second* pada area *quadtree*.

Tabel 3. Pengujian Collision Detection Dengan Bruteforce

Pengujian ke-	Jumlah objek Collidable	Frame per second
1	2	76.1
2	4	63.2
3	6	55.4

Sumber: Data diolah

Gambar 20 Pengujian 2 Object Bruteforce

Pada gambar 20 dapat dilihat pengujian dengan menggunakan 2 collidable objek menghasilkan 76.1 *frame per second* pada area *game.*

Gambar 21 Pengujian 4 Object Bruteforce

Pada gambar 21 dapat dilihat pengujian dengan menggunakan 2 collidable objek menghasilkan 63.2 *frame per second* pada area *game*.

Gambar 2 Pengujian 6 Object Bruteforce

Pada gambar 22 dapat dilihat pengujian dengan menggunakan 2 collidable objek menghasilkan 55.4 *frame per second* pada area *game*.

Kesimpulan

Pengujian yang dilakukan memperlihatkan frame per second (FPS) yang dihasilkan dari penerapan metode quadtree pada pendeteksian algoritma collision detection secara optimal. Frame per second digunakan sebagai acuan hasil dari penerapan metode quadtree terhadap metode bruteforce. Algoritma *collision detection* yang diterapkan pada metode *quadtree* dengan memanggil algoritma *collision detection* hanya pada saat node *quadtree* memiliki member *collidable object* yaitu antara peluru dan musuh. Semakin banyak objek peluru pada area *quadtree* maka semakin banyak pula pembagian area hingga semakin mengecil area tiap node yang diperiksa pertumpukannya. Karena *collision detection* dilakukan hanya pada node tertentu saja maka semakin kecil wilayah node akan semakin singkat deteksi pertumpukan object yang menghasilkan penggunaan resource lebih minim dan menyebabkan kenaikan *frame per second* yang dihasilkan.

Referensi

Haekal Fasha, L., & Gufroni, M. (2018). IMPLEMENTASI ALGORITMA COLLISION DETECTION PADA GAME SIMULATOR DRIVING CAR. In *Jurnal String* (Vol. 3, Issue 1).

Indra, A., Wicaksono, L., Muhammad, E., Jonemaro, A., & Akbar, M. A. (2019). *Optimasi Collision Detection Pada 2D Spaceship Game Menggunakan Metode Quadtree* (Vol. 3, Issue 9). http://j-ptiik.ub.ac.id

Tandean, J., Endry, Wijaya, A., Gunawan, W., & Harahap, M. (2019). Vehicle Collision Detection Application Through Collision Video Files with Quadtree Algorithms. *Journal of Physics: Conference Series*, *1230*(1). https://doi.org/10.1088/1742-6596/1230/1/012016