ANALISIS DAN IMPLEMENTASI DETEKSI TUBRUKAN PADA KARAKTER GAME 2D MENGGUNAKAN MULTILINE (STUDI KASUS: FIGHTING GAME)

Ni Luh Putu K.[1], Agung Toto Wibowo[2], Retno Novi Dayawati[3]

[1]Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom

Abstrak

Pada sebuah game biasanya terdapat collision detection dan pemilihan terhadap metode collision yang digunakan mempengaruhi dari segi efisiensi yang bisa dilihat dari nilai rata-rata FPS yang melebihi 60 dan persentase penggunaan CPUyang kurang dari 50% dan efektifitas yang dilihat dari sisa piksel pembungkus yang berada di bawah 50% serta coverage area dari piksel pembungkus yang tidak kurang dari 100%. Pada tugas akhir ini digunakan proses collision detection menggunakan multiline dengan step antar titik yang bervariasi. Tujuan dari tugas akhir ini adalah selain membuktikan teori mengenai rata-rata nilai FPS dan waktu komputasi serta dapat disimpulkan bahwa belum tentu multiline dengan step terkecil merupakan multiline yang terbaik.

Kata Kunci : collision, multiline, FPS, step, komputasi, sisa piksel

Abstract

In a game, there is usually collision detection and a collision detection method that chosen can give effect on the efficient side of the game that can be seen from average FPS that is above 60 and computational time that can be seen from CPU usage under 50% and from the effective side can be seen from the amount of empty space that exists between boundary and real character that must be under 50%, and also the coverage area of a boundary must be 100%. In this bachelor thesis, will be used multiline collision detection method with various step between nodes. Besides proving the condition of computational time and FPS, and can conclude that the smallest step taken musn't always be the best multiline.

Keywords : collision, multiline, FPS, step, computation, empty space

1. Pendahuluan

1.1 Latar Belakang

Pada *game* action, banyak terdapat interaksi antara karakter yang satu dengan yang lainnya. Jika antara karakter terjadi interaksi maka seharusnya terdapat respon yang cepat dan tepat dari masing-masing karakter. Adanya deteksi tubrukan (collision detection) menyebabkan interaksi antara karakter menjadi terlihat lebih nyata. Terjadinya collision detection pada *game* action seharusnya menggunakan rentang waktu yang cukup singkat dan tepat.

Terdapat beberapa metode yang bisa digunakan untuk mendeteksi terjadinya tubrukan, yaitu metode irisan antar objek, multi-pass atau hybrid, rectangle, circle, ellips, multibounding box, dan multiline. Metode irisan antara objek merupakan metode yang paling sering digunakan. Tetapi hasilnya kurang detail sehingga masih terdapat sisa jarak ketika karakter saling bertabrakan. Kemudian terdapat metode multi-pass atau hybrid, yaitu sebuah metode yang mengeliminasi bagian karakter yang memiliki jarak yang jauh dengan bagian karakter lainnya.[13]

Metode Rectangle merupakan sebuah metode collision detection di mana pada setiap karakternya terdapat batas luar berupa segi empat. Tabrakan antar karakter terjadi apabila batas luar yang terdapat pada masing-masing karakter saling bersinggungan. Kekurangan yang terdapat pada teknik ini adalah masih terdapat banyak sisa antara karakter dengan batas luar sehingga tubrukan yang terjadi tidak terlihat secara detail. Metode lainnya yang juga banyak dipakai adalah Circle. Metode ini menggunakan lingkaran sebagai batas luar dari sebuah karakter. Kekurangan pada metode ini adalah tidak cocok untuk digunakan pada karakter yang panjang dan tipis karena mudah terjadi kesalahan pada saat pendeteksian tubrukan dan juga terdapat sisa antara karakter dengan batas luar sehingga tubrukan yang terjadi tidak terlihat secara detail.

Jika pada *game* 3D, elips yang kemudian berubah menjadi ellipsoid, banyak digunakan maka pada *game* 2D elips jarang digunakan karena banyak terdapat sisa ruang antara karakter dengan batas luar yang berbentuk elips. Sedangkan untuk metode multibounding box adalah metode di mana terdapat banyak bounding box dan berukuran lebih kecil serta disesuaikan dengan ukuran karakter. Kekurangan dari metode multibounding box adalah masih terdapat sisa ruang antara karakter dengan batas luar tersebut. Metode Multiline atau Polygon merupakansebuah metode dengan membuat sebuah garis pada sebuah sisi dari karakter. Garis tersebut kemudian menjadi batas luar dari karakter tersebut. Proses tubrukan pada Multiline dihitung jika terdapat irisan antara garis pada karakter satu dengan karakter yang lainnya.[5,6,11]

Karena pada *game* fighting, dimana banyak terdapat interaksi antar karakter, dibutuhkan sebuah collision detection yang detail dan tepat serta memiliki waktu komputasi yang cepat maka digunakan metode multiline. Selain itu, metode multiline merupakan metode yang memiliki waktu komputasiyang kecil karena perhitungan interaksi antar karakter adalah dengan perhitungan garis singgung antara batas luar karakter tersebut.[2,3]

Oleh karena itu, pada tugas akhir ini penulis memilih menggunakan metode multiline dan mengangkat studi kasus *Game* Fighting untuk menjadi objek penelitian penulis.

1.2 Perumusan Masalah

Permasalahan yang menjadi objek penelitian penulis, yaitu:
1. Berapa besar jarak yang tersisa antara objek sebenarnya dengan polygon yang membungkusnya?
2. Berapa besar waktu komputasi jika line diperbanyak?
3. Berapa besar waktu komputasi jika semakin banyak objek yang ada?
4. Berapa besar *step* ketika batasan *frame per second* berada dibawah 24 *frame per second* dan berapa waktu komputasinya?

1.3 Tujuan

Tujuan dari tugas akhir ini adalah:
1. Membuktikan bahwa penambahan objek dan berkurangnya *step* akan membuat rata-rata FPS menurun serta penggunaan CPU meningkat.
2. Dari banyak multiline dengan *step* yang berbeda-beda bisa didapatkan multiline terbaik berdasarkan *coverage* terhadap piksel sebenarnya, rata-rata FPS dan rata-rata penggunaan CPU dalam keadaan jumlah objek paling sedikit hingga jumlah objek paling banyak.
3. Membuktikan apakah hipotesis yang menyatakan bahwa multiline dengan step yang semakin kecil adalah multiline yang terbaik.

1.4 Batasan Masalah

Yang menjadi batasan masalah dari tugas akhir ini adalah:
1. Waktu yang dihitung adalah waktu terjadinya *collision* dan pada saat rendering gambar karakter tersebut.
2. Penambahan objek atau *line* berhenti hingga waktu komputasi mencapai 1 detik.
3. Perhitungan waktu komputasi hanya pada satu computer dengan satu kartu grafik dan prosesor dengan satu inti.
4. Waktu komputasi yang dihitung akan direpresentasikan dalam bentuk persentase pengunaan CPU.

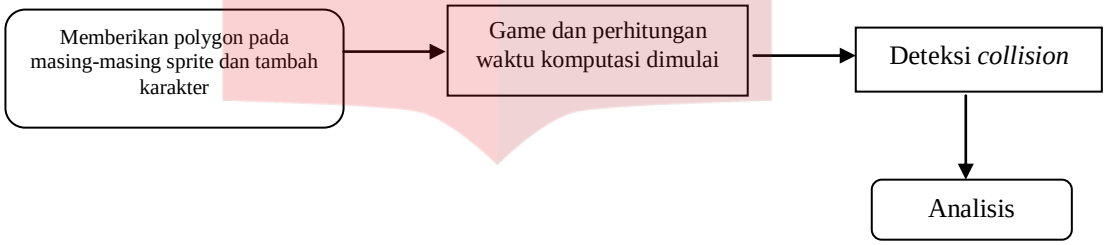
1.5 Metodologi Penyelesaian Masalah

Metodologi yang digunakan dalam menyelesaikan tugas akhir ini, yaitu:
1. Studi kepustakaan, yaitu dengan mempelajari literatur-literatur yang ada, yang berkaitan dengan permasalahan yang meliputi:
    1. Konsep *collision detection.*
    2. Konsep *collision detection* menggunakan *multiline.*
2. Perumusan masalah, yaitu dengan mendalami materi yang digunakan untuk mendefinisikan masalah, yaitu:
    1. Perhitungan jarak pada bagian depan yang langsung terkena *collision*.

2. Besar waktu komputasi jika objek ditambah atau line ditambah.
3. Besar penggunaan CPU tertinggi, serta FPS dibawah 24 terdapat pada penambahan objek atau line keberapa.

3. Analisa Kebutuhan Sistem dan Perancangan Perangkat Lunak, yaitu melakukan analisa terhadap model implementasi yang dibangun dengan tujuan memahami secara jelas proses yang dilakukan pada sistem tersebut, serta perancangan dengan menggunakan konsep analisis dan desain yang berorientasi objek.

Deskripsi Sistem



Pada sistem ini, pertama kali sistem menentukan titik-titik pada setiap karakter untuk dijadikan polygon yang kemudian akan disipan di dalam memori. Setelah karakter memiliki titik-titik tersebut, permainan dimulai dan pada saat itu perhitungan waktu komputasi, persentase penggunaan CPU, serta FPS dimulai. Kemudian proses akan dilakukan berulang-ulang hingga besar jarak antar titik adalah 5.

4. Implementasi Perancangan Perangkat Lunak, yaitu implementasi secara *coding* berdasarkan analisis dan desain dengan menggunakan bahasa pemrograman Java dibantu dengan *framework* GTGE. Dengan spesifikasi komputer sebagai berikut:
   a. RAM 2 GB
   b. Hardisk 160 GB
   c. Prosesor Intel Celeron M 1.8 GHz
   d. Kartu Grafis Intel 965 GM Express Chipset Family
   e. *Sound card* Realtek
   f. Operating System Microsoft Windows XP MSDNAA
5. Uji coba dan analisa system, yang meliputi:
   1. Mengimplementasikan *multiline* pada setiap objek yang ada pada studi kasus tersebut.
   2. Menghitung seberapa besar jarak antara bentuk karakter yang sebenarnya.
   3. Menghitung seberapa besar waktu komputasi, persentase CPU, serta FPS jika *line* ditambahkan.
   4. Menghitung seberapa besar waktu komputasi, persentase CPU, serta FPS jika objek ditambahkan.
   5. Pada penambahan objek keberapa atau line keberapa sehingga *frame per second* dibawah 30 fps serta hitung besar waktu komputasi.
6. Penyusunan laporan tugas akhir dan kesimpulan akhir.

1.6 Sistematika Penulisan

Tugas Akhir ini disusun dengan sistematika penulisan sebagai berikut:

BAB I Pendahuluan

Bab ini berisi latar belakang, perumusan masalah, tujuan, hipotesis, batasan masalah, metodologi penyelesaian masalah, serta sistematika penulisan buku tugas akhir

BAB II Landasan Teori

Bab ini berisi uraian teori mengenai *Collision Detection*, Metode pada *Collision Detection*, Waktu Komputasi, dan *Frame Rate*.

BAB III Perancangan dan Implementasi

Bab ini berisi rancangan algoritma yang digunakan untuk membuat polygon, mendeteksi tubrukan, serta perhitungan waktu komputasi serta *frame rate*.

BAB IV Analisis Hasil Implementasi

Bab ini berisi uraian hasil implementasi serta menganalisis parameter tingkat komputasi tertinggi, terendah, serta parameter *frame rate*.

BAB V Kesimpulan dan Saran

Bab ini berisi tentang kesimpulan yang didapat dari pelaksanaan tugas akhir ini serta saran-saranyang diperlukan untuk perbaikan maupun pengembangannya lebih lanjut.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan maka didapatkan kesimpulan sebagai berikut:

a. *Collision detection* dengan metode Multiline cocok digunakan dalam lingkungan game yang memiliki jumlah objek yang banyak dan diperlukan collision yang cukup detail. Selain itu, dengan menggunakan metode ini bisa dipilih multiline yang terbaik dari sekumpulan multiline yang bisa dibuat berdasarkan FPS, komputasi, coverage pembungkus, serta sisa jarak pada pembungkus.
b. *Multiline* yang memiliki step paling kecil, pada kasus ini, tidak menjadi *multiline* yang terbaik.
c. Semakin banyak objek yang berada dalam game maka rata-ratapersentase penggunaan CPU semakin meningkat dan rata-rata FPS menurun.
d. Semakin kecil *step* pada multiline maka nilai rata-rata FPS menurun dan rata-rata persentase penggunaan CPU meningkat.

5.2 Saran

Berikut ini merupakan saran penelitian yang bisa dilakukan lebih lanjut, yaitu:

a. Collision detection untuk fighting game yang menambahkan proses broad phase untuk mendapatkan persentase penggunan CPU yang lebih kecil.
b. Collision detection untuk fighting game dengan membandingkan dari segi *hardware* untuk mendapatkan penggunaan CPU yang terbaik.

Referensi

[1] Abrash, Michael. 1991. *"The Polygon Primeval"*. Dr. Dobb's Journal.

[2] Collision Detection, 2010, http://faculty.cs.**tamu**.edu/schaefer, didownload pada tanggal 11 November 2010.

[3] CPU usage, http://www.terryscomputertips.com/computers/what-is-cpu-usage-and-how-can-i-reduce-it/ diakses pada tanggal 30 Januari 2011.

[4] DeLoura, Mark A. 2001. *"Game Programming Gems 2"*. Cengage Learning.

[5] Fares, Charbel, Hamam, Yskandar. 2005. "*Collision Detection for Rigid Bodies: A State of the Art Review*".

[6] Faure, Fancois. Barbier Sebastien. Allard, Jeremie. Falipou, Florent. 2008. "*Image-based Collision Detection and Response between Arbitrary Volume Objects.*".

[7] FPS comparison, http://www.boallen.com/fps-compare.html diakses pada tanggal 2 Februari 2011.

[8] Govindaraju, Naga K. Redon, Stephane. Lin, Ming C. Manocha, Dinesh. 2003. "*CULLIDE: Interactive Collision Detection between Complex Models in Large Environments using Graphics Hardware*"

[9] Harbour. Jonathan S. 2004. "*Game Programming All in One Second Edition"*. Boston: Thomson Course Technology.

[10] Human Eye Frame Rate, 2011, http://www.newton.dep.anl.gov/askasci/gen01/gen01025.htm, didownload pada tanggal 01 Januari 2011.

[11] Intersection Point of Two Lines, 2010, http://local.wasp.uwa.edu.au/~pbourke/geometry/lineline2d/, didownload pada tanggal 2 Desember 2010.

[12] Jimenes. Juan Jose, et al. 2004. "Efficient Collision Detection between 2D Polygons". Plzen: Science Press.

[13] Larimer, James, Jeniffer Gille, 2004, "5.1: Visual Performance Depends Upon Signal Resolution: Frame Rate, Dot Pitch, & Bit Depth Guidelines". Journal of NASA Ames Research Center.

[14] Mendoza, C. O'Sullivan, C. "*Interruptible Collision Detection for Demorfale Objects*", in *ScienceDirect Computer and Graphics* 30(2006), pp 432-436.

[15] Simple Algorithms-Intersection of Lines, 2011, http://www.geog.ubc.ca/courses/klink/gis.notes/ncgia/u32.html, didownload pada tanggal 01 Januari 2011.

[16] Suryadi. 1993. "Pengantar Analisis Algoritma Seri Diktat Kuliah". Jakarta: Gunadarma.

[17] Fighting Game – Definition. http://www.wordiq.com/definition/Fighting_game, didownload pada tanggal 5 Februari 2011.