

Game Real-Time Strategy *Galactic Defense* Berbasis Ios

Bobby William Therry¹, Justinus Andjarwirawan², Liliana³

Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra

Jl. Siwalankerto 121 – 131 Surabaya 60236

Telp. (031) – 2983455, Fax. (031) – 8417658

E-mail : ggypro@gmail.com¹, justin@petra.ac.id², lilian@petra.ac.id³

ABSTRAK

Dengan semakin cepatnya perkembangan teknologi, kebutuhan akan hiburan semakin meningkat. Di sisi lain kesibukan orang-orang akan pekerjaan dan kegiatan sehari-hari juga semakin bertambah. Adanya *smartphone* membantu begitu banyak orang untuk meningkatkan produktivitas. Selain kegunaannya yang sesuai kebutuhan pengguna telepon genggam pada umumnya, *smartphone* memiliki fitur-fitur tambahan untuk mendapatkan informasi dan hiburan lebih.

Android dan iOS merupakan 2 jenis sistem operasi yang sekarang marak digunakan di khalayak umum. Dengan menggunakan *game engine* bernama Cocos2d, *game* ini akan dibuat untuk sistem operasi iOS. Kecerdasan buatan dengan metode *Finite State Machine* juga diterapkan kepada 5 NPC karena permainan ini ditujukan untuk *single-player*.

Dengan menggunakan SpriteBuilder sebagai *editor* untuk *user interface*, Objective-C sebagai bahasa pemrograman dan Xcode sebagai IDE. Video *game* bernama *Galactic Defense* telah dibuat dengan mengadaptasi permainan tradisional "bentengan" yang telah dimodifikasi untuk menyesuaikan media *handheld device*. *Game* ini memiliki 2 tingkat kesulitan dan fitur untuk menyimpan *state* permainan (*save game*).

Kata Kunci: Cocos2d, *Real-Time Strategy*, iOS *Game Development*

ABSTRACT

With the rapid development of technology, the need of entertainment keeps going up. On the other hand activities and businesses among the vast majority of people are increasing. The existence of smartphone helps a lot of people to increase their productivity. Besides the general function of it, smartphone can also be used for gathering information and as an entertainment system.

Android and iOS are the top 2 operating system that most people used. Using a game engine called Cocos2d, a game will be made for iOS operating system. Artificial intelligence with Finite State Machine method will be applied to this game on 5 NPCs, since the game is intended for single-player.

With SpriteBuilder as an editor for user interface, Objective-C as the programming language and Xcode as IDE. A video game called Galactic Defense has been created by adapting it's concept from a traditional game called "Bentengan" (or Fortress in English) which has been modified to adjust the handheld device. This game has 2 difficulty levels and a save game feature.

Keywords: Cocos2d, *Real-Time Strategy*, iOS *Game Development*

1. LATAR BELAKANG

Dengan semakin majunya teknologi di era modern seperti ini, bermain *game* merupakan salah satu kegiatan yang dilakukan setiap orang sehari-hari. Namun karena tingginya pula tingkat produktivitas setiap orang, bermain *game* di depan layar kaca seperti TV dengan menggunakan *console* ataupun bermain *game* melalui PC merupakan hal yang tidak mungkin dilakukan setiap saat.

Smartphone adalah salah satu perkembangan yang dapat menjawab permasalahan diatas, karena perkembangan teknologi yang begitu cepat, memungkinkan setiap orang untuk bermain *game* menggunakan ponsel. Selain dapat dilakukan di waktu-waktu luang, bermain melalui ponsel juga berguna saat *stress* dan mengisi waktu saat menunggu hal-hal tertentu.

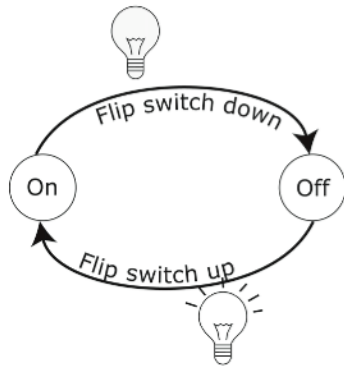
Beberapa jenis *smartphone* yang beredar di masyarakat sekarang membuat adanya berbagai macam permainan yang dapat dimainkan melalui genggam tangan kita. Android dan iOS merupakan 2 jenis sistem operasi yang paling banyak digunakan masyarakat umum. Berfokus terhadap iOS *game* bernama *Galactic Defense* akan dibuat memanfaatkan teknologi *Game Engine* Cocos2D menggunakan Sprite Builder IDE dan Xcode.[2][3][10][11] *Game* ini merupakan *game* yang diadaptasi dari permainan tradisional "Bentengan".[9] permainan klasik yang biasa dimainkan anak-anak sekolah dasar sampai sekolah menengah atas. *Player* akan menggerakkan salah satu karakter di dalam permainan dan dapat berganti karakter lain (*Role-playing*).[6] Permainan bentengan ini akan dimodifikasi dengan menjadi 5 *characters / team*.

2. ARTIFICIAL INTELLIGENCE

Artificial Intelligence adalah sebuah entitas yang dimasukkan ke dalam sebuah komputer yang dapat membuat komputer tersebut melakukan pekerjaan seperti yang dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer, logika *fuzzy*, jaringan syaraf tiruan dan robotika.

Banyak hal yang kelihatannya sulit untuk kecerdasan manusia, tetapi untuk Informatika relatif tidak bermasalah. Seperti contoh: mentransformasikan persamaan, menyelesaikan persamaan integral, membuat permainan catur atau Backgammon. Di sisi lain, hal yang bagi manusia kelihatannya menuntut sedikit kecerdasan, sampai sekarang masih sulit untuk direalisasikan dalam Informatika Seperti contoh: Pengenalan Obyek/Muka, bermain sepak bola.

Dengan adanya AI terciptalah NPC atau yang biasa disebut *Non-Player Character*. [6][7] NPC adalah karakter dalam sebuah *game* yang tidak digerakkan oleh manusia melainkan oleh komputer, dan pada umumnya setiap NPC memiliki cara berpikir masing-masing untuk memproses *input* dan menghasilkan *output* yang diberikan oleh *player*. Berbagai macam metode telah ditemukan untuk menerapkan sebuah kecerdasan buatan pada *game*, beberapa metode yang cukup terkenal adalah *minimax* dan *finite-state machine*.



Gambar 1. Saklar lampu merupakan sebuah Finite State Machine yang simple

Pada mulanya FSM merupakan sebuah alat yang digunakan oleh para ahli matematika untuk menyelesaikan sebuah masalah. [1][4] Yang paling terkenal pada saat itu adalah buatan dari Alan Turing yaitu the Turing Machine. Definisi dari *Finite-state Machine* sendiri adalah sebuah *device* atau alat yang dapat diberi kondisi-kondisi secara terbatas pada suatu waktu tertentu dan dapat beroperasi dengan 2 cara yaitu :

- Menerima *input* dan membuat suatu transisi dari satu kondisi ke kondisi yang lain.
- Menghasilkan *output*

3. CHARACTERS SPECIALITY

Dalam permainan bentengan di kehidupan nyata, setiap pemain akan memiliki keistimewaan masing-masing, contohnya : Akan ada pemain yang memiliki postur tubuh tinggi sehingga langkahnya lebih besar sehingga bisa berlari lebih jauh, akan ada pula pemain yang memiliki badan lebih kecil akan tetapi larinya begitu cepat, ada juga yang berbadan besar yang bisa menghalangi lawan yang akan lewat dan membantu timnya. Demikian pula pada penerapan *game* ini menjadi sebuah *game* digital, akan ada 4 jenis karakter yang memiliki kemampuan spesial dan 1 karakter yang tidak memiliki kemampuan apapun. [8]

Dalam permainan ini semua karakter akan memiliki beberapa *main attribute* yaitu *stamina*, *speed*, *power* dan *lives*. *Stamina* adalah *attribute* yang akan berkurang terus menerus sesuai dengan berjalannya waktu, sedangkan *Speed* itu relatif terhadap *stamina* yang dimiliki karakter, apabila karakter memiliki *stamina* yang tinggi maka *speed* dari karakter tersebut juga akan menjadi tinggi. *Power* adalah *attribute* yang didapatkan oleh tiap karakter ketika mereka kembali ke *base*, sehingga siapapun yang baru saja kembali ke *base* akan memiliki *power* yang lebih tinggi dibandingkan karakter yang belum kembali ke *base*. *Lives* adalah nyawa karakter, jika karakter ditabrak oleh lawan maka *life* akan berkurang bila *power* lawan lebih besar dari pada *power*. [5]

3.1 The Runner

Runner merupakan karakter yang memiliki *stamina* dan *life standard* akan tetapi bisa menggunakan *skill* untuk menambah kecepatan di atas rata-rata.

Untuk tingkat kesulitan *Easy*, *Runner* akan bergerak secara *random*, dan untuk tingkat kesulitan *Hard*, *Runner* akan bergerak menuju posisi tertentu sebelum akhirnya mencoba menyentuh base lawan.

3.2 The Tanker

Tanker merupakan sebuah AI yang memiliki kemampuan untuk hidup 2 kali. Ketika karakter ini berhasil ditangkap maka dia akan kembali *respawn* ke base untuk ditangkap ke 2 kalinya.

Pada tingkat kesulitan *Easy*, *Tanker* akan bergerak menyerang musuh sedangkan *Hard* akan menjaga benteng.

3.3 The Healer

Healer merupakan sebuah AI yang memiliki kemampuan untuk merestore *stamina/energy* dari pesawat kawan. Akan tetapi setelah penggunaan ini *Healer* akan menjadi sangat lambat dan harus kembali ke base

Healer pada tingkat kesulitan *Easy* tidak akan mengikuti anggota timnya, sedangkan pada kesulitan *Hard*, *Healer* akan selalu mengikuti.

3.4 The Phaser

Phaser merupakan sebuah AI yang memiliki kemampuan untuk menghilang sesaat, dimana pada saat itu *Phaser* tidak akan dapat mengalami tabrakan dengan lawan.

Phaser pada tingkat kesulitan *easy* akan menggunakan kekuatannya secara sembarangan, sedangkan pada tingkat kesulitan *hard* dia akan menggunakannya secara dalam strategi.

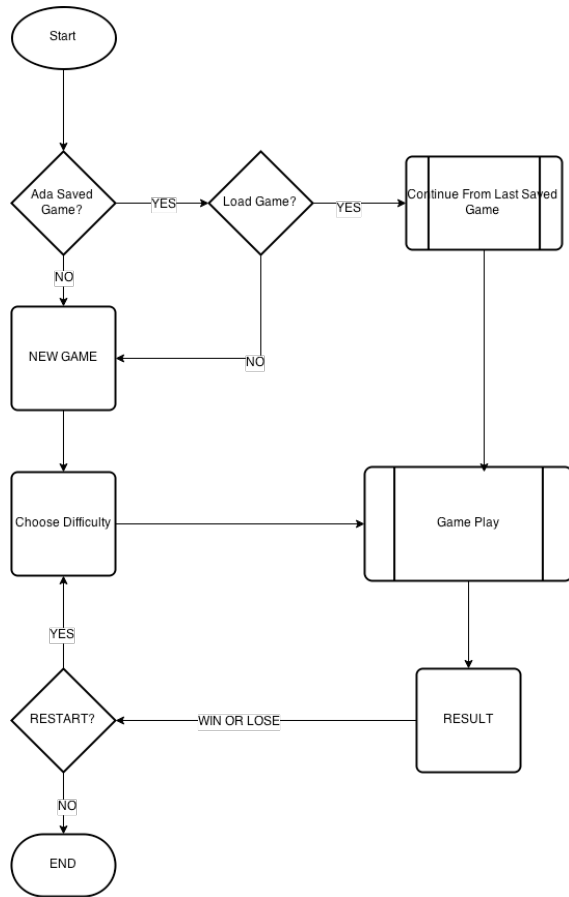
3.5 The Decoy

Decoy merupakan sebuah AI yang tidak memiliki kemampuan apapun tapi di tingkat kesulitan *Hard*, *Decoy* memiliki pola gerak khusus yang dapat membuat *player* berpikir dua kali untuk langsung menyerang

Di tingkat kesulitan *Easy*, akan bergerak secara *random* dan dalam tingkat kesulitan *Hard*, *Decoy* akan menuju titik terkanan untuk mencari celah yang bisa dimanfaatkan untuk memperoleh kemenangan.

4. DESAIN SISTEM

Bagian ini akan menggambarkan garis besar sistem yang akan dibuat, hubungan yang terjadi antara proses 1 sistem dengan lainnya, garis besar cara kerja sistem, dan desain tampilan *software* yang akan dibuat. Proses dan *flow* dari sistem ini akan dijelaskan di bawah ini.



Gambar 2. Cara Kerja Aplikasi

Berikut dapat kita lihat melalui *flowchart* di atas tahap apa saja yang harus dilakukan sebelum memulai permainan. Pada saat aplikasi ini dibuka, *user* akan diberi 2 pilihan yaitu untuk melanjutkan permainan yang sudah di-*save* (disimpan), atau memulai permainan baru. Apabila *user* memilih untuk me-*load* permainan yang sebelumnya maka *user* akan langsung diarahkan ke *gameplay scene*. Apabila *user* memilih untuk *start a new game* maka *user* akan diberi 2 opsi pilihan tingkat kesulitan yaitu *Easy* atau *Hard*, baru setelah itu *user* akan diarahkan ke *gameplay scene*. Setelah menyelesaikan permainan, *user* akan diberi opsi untuk mengulangi permainan atau berhenti.

5. PENGUJIAN

AI akan diuji kemampuannya untuk deteksi apakah ada pesawat lain, *collision*, dan penggunaan *power*. Strategi yang digunakan AI juga akan diuji apakah dapat dikalahkan atau tidak.

Tipe Pengujian yang akan dilakukan antara lain:

- Pengujian *scannedShip*
- Pengujian *Collision*
- Pengujian penggunaan *power*
- Pengujian jalannya strategi

5.1 *scannedShip*

Dalam Subbab ini akan ditunjukkan kemampuan AI untuk mendeteksi pesawat lain dan apa yang akan dilakukan pada saat menemukan ship tersebut.

Tabel 1. Pengujian *scannedShip*

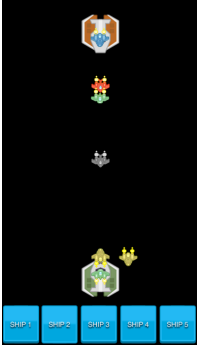
Kondisi	Visual dalam game
1. Gambar pertama menunjukkan <i>Ship Tanker</i> milik <i>Ally</i> yang telah keluar dari base, berarti <i>stamina</i> dari <i>Ship</i> tersebut maksimal	
2. Karena <i>Ship Tanker</i> memiliki tenaga yang maksimal pada saat <i>Ship</i> ini maju, <i>Ship Runner & Tanker</i> lawan mundur pada saat melakukan <i>detection</i> .	

5.2 Pengujian *Collision*

Dalam Subbab ini akan ditunjukkan kemampuan AI untuk mendeteksi adanya *collision* terhadap pesawat lain, apabila terjadi *collision* antara yang lebih kuat atau yang lebih lemah apa akibatnya.

Tabel 2. Pengujian *Collision*

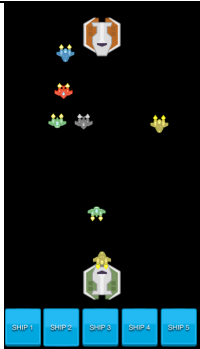
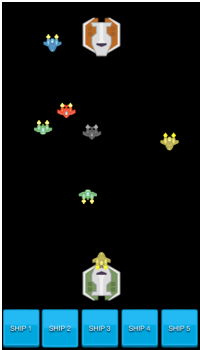
Kondisi	Visual dalam game
1. Pada gambar pertama dapat kita lihat bahwa <i>Tanker</i> milik <i>User</i> sedang maju untuk menyerang,	

<p>2. Akan tetapi dapat kita lihat di gambar ke 2, karena <i>tanker</i> dari musuh telah melakukan <i>recharge</i>, maka <i>tanker</i> dari <i>Ally</i> mati terbunuh dan akhirnya <i>respawn</i> di <i>base</i> lagi</p>	
---	---

5.3 Pengujian Penggunaan Power

Pada Subbab ini akan ditunjukkan special skill salah satu *ship* yakni menggunakan *phase power* pada *phaser*. Hal ini terjadi karena *phaser* mendeteksi adanya musuh di dalam radarnya.

Tabel 3. Pengujian Penggunaan Power

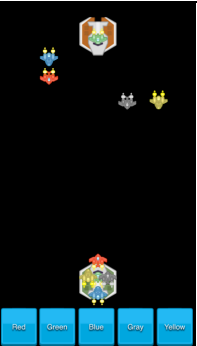
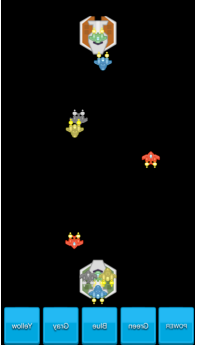
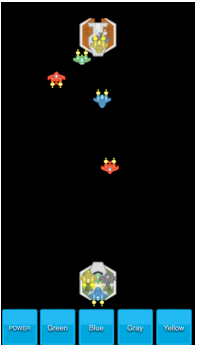

Kondisi	Visual dalam Game
<p>1. Gambar pertama menunjukkan <i>Tanker</i> yang akan mendekati ke <i>Phaser</i> lawan</p>	
<p>2. Di gambar 2 <i>Phaser</i> sudah mendeteksi adanya <i>Tanker</i> lawan sehingga <i>phase power</i> pun digunakan oleh <i>phaser</i>. (<i>opacity</i> berkurang)</p>	

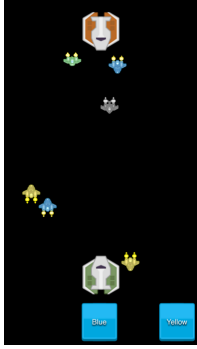
5.4 Pengujian Jalannya Strategi

Pada Subbab ini akan ditunjukkan kemampuan AI dalam *difficulty Hard* yang memiliki strategi. Dalam strategi ini, *tanker*, *healer*, dan *decoy* memiliki gerakan khusus. *Tanker* akan menjaga *base* dari serangan lawan, sedangkan *Healer* akan terus mengikuti

salah satu *ally* yang menjadi *target*, sedangkan *Decoy* akan terus berjalan disebelah kanan sambil mencari celah untuk menyerang.

Tabel 4. Pengujian Strategi

Kondisi	Visual dalam Game
<p>1. Gambar pertama menunjukkan <i>start position</i> mereka dimana mereka menuju posisi masing-masing. <i>Tanker</i> juga terlihat <i>standby</i>.</p>	
<p>2. Lalu di gambar ke 2 terlihat <i>Runner</i> pemain menyerang dari arah kiri dengan tujuan untuk mencari celah yang kosong.</p>	
<p>3. Di gambar berikutnya <i>user</i> meluncurkan <i>Runner</i> yang mengakibatkan <i>Tanker</i> yang menjaga <i>base</i> untuk maju dan menangkap <i>Runner</i> pemain.</p>	
<p>4. Di gambar ke-3 ini pemain melakukan serangan dengan beberapa unit di sebelah kiri, akan tetapi <i>decoy</i> dari tim lawan yang menyerang dari sebelah kanan sudah berada di posisi siaga.</p>	

<p>5. Ketika seluruh unit pemain di arahkan ke lawan, <i>decoy</i> dengan mudah menyerang dari kanan.</p>	
---	---

6. KESIMPULAN

Berdasarkan hasil pengujian dapat disimpulkan beberapa hal sebagai berikut.

- Penggunaan *Finite State Machine* cukup terbatas pada pengetahuan dari pembuat program dan limitasi-limitasinya
- *Finite State Machine* merupakan *AI* yang hampir tidak bercelah apabila didesain dengan benar
- Fungsi *scannedShip* yang di-*running* oleh *game engine* memiliki keterbatasan dalam pengecekan dimana apabila pengecekan dilakukan setiap saat maka akan terjadi semacam "*infinite loop*" yang membuat *AI* berhenti total, tetapi membatasi dengan waktu pun terkadang membuat *AI* menjadi sedikit lebih lambat.
- Strategi yang diterapkan membuat permainan menjadi lebih seru, akan tetapi karena keterbatasan *user* dalam memainkan 5 unit *AI* membuat permainan menjadi lebih susah.

7. DAFTAR REFERENSI

- [1] Buckland, M. 2002. *AI Techniques for Game Programming*.
- [2] Xcode Documentation. 2014. URL=<https://developer.apple.com/library/ios/documentation/T>

oolsLanguages/Conceptual/Xcode_Overview/chapters/about.html

- [3] Sprite Builder. 2014. URL=<https://spritebuilder.com/>
- [4] Buckland, M. 2005. *Programming Game AI by Example*.
- [5] Millington, I. 2006. *Artificial Intelligence for Games*.
- [6] Nendya, M. B., Gunanto, S. G., & Santosa, R. G. journal: *Pemetaan Perilaku Non-Playable Character Pada Permainan Berbasis Role Playing Game Menggunakan Metode Finite State Machine*. URL=http://www.academia.edu/5184151/Pemetaan_Perilaku_Non-Playable_Character_Pada_Permainan_Berbasis_Role_Playing_Game_Menggunakan_Metode_Finite_State_Machine
- [7] Safadi, F., Fonteneau, R., & Ernst, D 2015. Research Articles: Artificial Intelligence in Video Games: Towards a Unified Framework. *International Journal of Computer Games Technology*. Volume 2015, Article ID 271296.
- [8] Wu, B., & Wang, A. I 2012. Review Articles: A Guideline for Game Development-Based Learning: A Literature Review. *International Journal of Computer Games Technology*. Volume 2012, Article ID 103710.
- [9] Wibawanto, W. journal: *Pengenalan Kembali Permainan Tradisional "BENTENGAN" Melalui Media Game Digital*. URL=https://www.academia.edu/13397272/PENGENALAN_KEMBALI_PERMAINAN_TRADISIONAL_BENTENGAN_MELALUI_MEDIA_GAME_DIGITAL
- [10] Rahamathunnisa, U., & Pragadeeswaran, S 2012. journal: *Collision Detection Game Using Cocos2dx-a Cross Platform*. *International Journal of Engineering Research and Applications*.
- [11] Vaidya, A. H., & Naik, S. 2013. journal: *Comprehensive Study and Technical Overview of Application Development in iOS, Android and Window Phone 8*. *International Journal of Computer Applications*. Volume 64.