GAME SERIUS PENATAAN BARANG DALAM KONTAINER MENGGUNAKAN ALGORITMA *GREEDY*

SKRIPSI

Oleh:
RIZKY PUTRI LUTHFIAH
NIM. 11650075

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2016

GAME SERIUS PENATAAN BARANG DALAM KONTAINER MENGGUNAKAN ALGORITMA *GREEDY*

SKRIPSI

Diajukan Kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
RIZKY PUTRI LUTHFIAH
NIM. 11650075

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2016

HALAMAN PERSETUJUAN

GAME SERIUS PENATAAN BARANG DALAM KONTAINER MENGGUNAKAN ALGORITMA *GREEDY*

SKRIPSI

Oleh :

Nama : Rizky Putri Luthfiah
NIM : 11650075
Jurusan : Teknik Informatika
Fakultas : Sains Dan Teknologi

Telah disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Fachrul Kurniawan, M.MT	Hani Nurhayati, M.T
NIP. 19771020 200901 1 001	NIP. 19780625 200801 2 006

Tanggal, 11 Juli 2016

Mengetahui,
Ketua Jurusan Teknik Informatika

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PENGESAHAN

GAME SERIUS PENATAAN BARANG DALAM KONTAINER MENGGUNAKAN ALGORITMA *GREEDY*

SKRIPSI

Oleh :

RIZKY PUTRI LUTHFIAH
NIM. 11650075

Telah Dipertahankan di Depan Dewan Penguji Skripsi dan
Dinyatakan Diterima Sebagai Salah Satu Persyaratan Untuk
Memperoleh Gelar Sarjana Komputer (S.Kom)

Tanggal 11 Juli 2016

Susunan Dewan Penguji		Tanda Tangan
1. Penguji Utama	: Yunifa Miftachul Arif, M.T NIP. 19830616 201101 1 004	( )
2. Ketua	: Fressy Nugroho, M.T NIP. 19710722 201101 1 001	( )
3. Sekretaris	: Fachrul Kurniawan, M.MT NIP. 19771020 200901 1 001	( )
4. Anggota	: Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )

Mengetahui dan Mengesahkan
Ketua Jurusan Teknik Informatika

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PERNYATAAN
ORISINALITAS PENELITIAN

Saya yang bertanda tangan di bawah ini:

Nama	: RIZKY PUTRI LUTHFIAH
NIM	: 11650075
Fakultas/ Jurusan	: Sains dan Teknologi / Teknik Informatika
Judul Penelitian	: ***GAME* SERIUS PENATAAN BARANG DALAM KONTAINER MENGGUNAKAN ALGORITMA *GREEDY***

Menyatakan dengan sebenar-benarnya bahwa hasil penelitian saya ini tidak terdapat unsur-unsur penjiplakan karya penelitian atau karya ilmiah yang pernah dilakukan atau dibuat oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata hasil penelitian ini terbukti terdapat unsur-unsur penjiplakan, maka saya bersedia untuk mempertanggungjawabkan, serta diproses sesuai peraturan yang berlaku.

Malang, 11 Juli 2016

Yang Membuat Pernyataan,

Rizky Putri Luthfiah

1650075

iv

MOTTO

" Jalan Hidup ini sudah ada yang mengatur, sudah ada yang menentukan Gusti Allah. Jika ada kesusahan berusaha untuk bersabar, jika ada kemudahan sepantasnya untuk bersyukur ".

Oleh karena itu,

*"Jangan pernah takut "**Jatuh**" ketika menempuh perjuangan karena seluruh pengorbanan yang dilakukan akan membentuk pembelajaran penting bagimu dan membuat apa yang nanti kamu dapatkan menjadi sesuatu yang berharga dalam hidupmu".*

- RIZKY PUTRI LUTHFIAH -

HALAMAN PERSEMBAHAN

Dengan rasa syukur seraya mengharap ridho Ilahi kupersembahkan karya ini kepada :

Eyang dan Ibunda tercinta

Hj. Chunnah & Uswatun Hasanah

Yang senantiasa mencurahkan kasih Sayang, perhatian, Doa, dan bimbingannya dalam setiap langkahku.

Semoga Allah SWT melindungi dan menyayangi keduanya.

*Serta yang aku cinta Tanteku **Zakiyah Wifaqi** dan Omku **Imam Zarkasy** yang selalu mendukungku. Keponakanku yang lucu **Maulana Firman Syah**.*

Kepada seseorang yang setia mendukung dan mendampingiku sampai detik ini

Muhammad Mirza Muttaqi

*Terimakasih untuk Bapak/Ibu Dosen **Pak Fachrul** sebagai dosen pembimbing I dan **Ibu Hani** sebagai dosen pembimbing II yang selalu sabar membimbing saya dalam menyelesaikan tugas akhir ini. **Pak Amin** sebagai wali dosen yang senantiasa mengawasi perkembangan perkuliahan saya selama beberapa tahun ini. Terimakasih juga untuk **Pak Fresy, Pak Yunifa, Pak Yaqin, Pak Syahid, Pak Syauqi, Pak Cahyo, Pak Irwan, Bu Aina, Bu Ririn, Bu Linda, Pak Faisal, Pak Fatchur, Ibu Roro** dan **Ibu Ivana**. Untuk **Pak Aziz, Pak Agung, Pak Mujib, Mbak Citra** dan **Mbak Rusiana**.*

Kepada teman seperjuanganku:

Muhammad Mirza M.(TI UIN) , Sigit R.P(TI UIN), Moh. Ali Majedi(TI UIN), Awwib Ahsana(TI UIN), Firdaus I(TI UIN),

yang bersama-sama saling menyemangati satu sama lain dan saling mengingatkan jika lalai.

Kepada para sahabat Putra Shoiba Digital Printing:

***Mas Ghulam, Laili Zakiyah, Mbak Rina, Rizal, Hendry, Idris, Andri, Febri, Mbak Suci** yang selalu sabar , mengerti serta menyemangati selalu.*

*Kepada para **sahabat TI** angkatan 2011 & 2012, yang selalu ada untuk membantu sesama. Dan kepada teman-temanku semua yang tidak bisa kusebutkan satu persatu yang selalu membantuku dan menyemangatiku di saat aku susah dan terpuruk.*

Semoga Allah SWT melindungi, menyayangi dan menempatkan mereka semuanya pada surganya kelak dan melimpahkan rezeki kepada mereka semua…

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Segala puji bagi Allah SWT tuhan semesta alam, karena atas segala rahmat dan karunia-Nya kepada penulis sehingga penulis mampu menyelesaikan skripsi dengan judul "*Game* Simulasi Penataan Barang Dalam Kontainer Menggunakan Algoritma *Greedy*" dengan baik dan lancar. Shalawat dan salam selalu tercurah kepada tauladan terbaik kita Nabi Agung Muhammad SAW yang telah membimbing umatnya dari zaman kegelapan dan kebodohan menuju cahaya islam yang terang *rahmatan lil alamiin* ini.

Dalam penyelesaian skripsi ini, banyak pihak yang telah memberikan bantuan baik secara moril, nasihat dan semangat maupun materiil. Atas segala bantuan yang telah diberikan, penulis ingin menyampaikan doa dan ucapan terimakasih yang sedalam-dalamnya kepada :

1. Prof. DR. H. Mudjia Raharjo, M.Si, selaku rektor UIN Maulana Malik Ibrahim Malang beserta seluruh staf. Dharma Bakti Bapak dan Ibu sekalian terhadap Universitas Islam Negeri Malang turut membesarkan dan mencerdaskan penulis.
2. Dr. Hj. Bayyinatul M., drh., M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang beserta seluruh staf.

Bapak dan ibu sekalian sangat berjasa memupuk dan menumbuhkan semangat untuk maju kepada penulis.

3. Bapak Dr. Cahyo Crysdian, selaku Ketua Jurusan Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang, yang sudah memberi banyak memberi pengetahuan, inspirasi dan pengalaman yang berharga.
4. Bapak Fachrul Kurniawan,M, M.T, selaku dosen pembimbing I yang telah meluangkan waktu untuk membimbing, memotivasi, mengarahkan dan memberi masukan dalam pengerjaan skripsi ini.
5. Ibu Hani Nurhayati, M.T, selaku dosen pembimbing II yang juga selalu memeberi masukan, nasehat serta petunjuk dalam penyusunan laporan skripsi.
6. Eyang, Ibu, Tante, Om serta keluarga besar saya tercinta yang selalu memberi dukungan yang tak terhingga serta doa yang senantiasa mengiringi setiap langkah pengerjaan skripsi ini.
7. Segenap Dosen Teknik Informatika yang telah memberikan bimbingan keilmuan kepada penulis selama masa studi.
8. Teman – teman seperjuangan Teknik Informatika 2011.
9. Para peneliti yang telah mengembangkan Game dengan Engine *Unity3d* yang menjadi acuan penulis dalam pembuatan skripsi ini. Serta semua pihak yang telah membantu yang tidak bisa disebutkan satu satu. Terimakasih banyak.

Berbagai kekurangan dan kesalahan mungkin pembaca temukan dalam penulisan skripsi ini, untuk itu penulis menerima segala kritik dan saran yang membangun dari pembaca sekalian. Semoga apa yang menjadi kekurangan bisa

disempurnakan oleh peneliti selanjutnya dan semoga karya tulis ini bisa bermanfaat dan menginspirasi bagi kita semua. Amin.

Wassalamualaikum Wr. Wb.

Malang, 11 Juli 2016

DAFTAR ISI

BAB V KESIMPULAN DAN SARAN

Daftar Diagram

Daftar Gambar

Daftar Tabel

ABSTRAK

Putri, Rizky. 2016. ***Game* Simulasi Penataan Barang Dalam Kontainer Menggunakan Algoritma *Greedy***. Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Pembimbing : (I) Fachrul Kurniawan,M, M.T, (II) Hani Nurhayati, M.T

Kata Kunci: *Greedy, Kontainer, Game, Game* Simulasi

Game atau Permainan adalah sebuah hiburan 'sepanjang masa' yang bisa dikatakan tak akan lekang oleh waktu. *Game* biasa di peruntukkan untuk hiburan yang menyenangkan seperti jenis game *fighting*, petualangan , *casual game* tetapi terkadang juga *game* ditujukan untuk edukasi, bahkan simulasi strategi. Karena sebenarnya *game* juga penting untuk mengasah perkembangan otak, meningkatkan konsentrasi, dan melatih dalam memecahkan masalah dengan tepat dan cepat. Oleh sebab itu *game* bisa di nikmati oleh semua kalangan baik anak-anak kecil, dewasa bahkan yang sudah tua.

Game simulasi bisa disebut sebagai jenis permainan yang memiliki strategi dalam pemecahan masalahnya. Pada penelitian kali ini juga peneliti merancang sebuah *game* untuk memecahkan masalah dalam proses penataan barang dalam kontainer. *Game* Simulasi Penataan Barang Dalam Kontainer ini dirancang berbasis *mobile*. *Player* di tuntut untuk menemukan strategi dalam menyelesaikan permasalahan penataan barang dalam kontainer berdasarkan lokasi tujuan dan berat barang supaya optimal. Saat ini terdapat beberapa metode optimasi salah satunya adalah Algoritma Greedy.

Pada penelitian ini, *Greedy* akan diterapkan sebagai metode permasalahan optimalisasi apakah barang yang sudah di tata dalam kontainer sesuai lokasi tujuan dan berat barang sudah optimal dan sesuai pada kontainer tersebut. Pengujian dilakukan pada perangkat *mobile* dengan menggunakan *platform* Android.

ABSTRACT

Putri, Rizky. 2016. ***Simulation Game of Goods Structuring in Container using Algorithm of Greedy*.** Thesis. Technology of Informatics Department Faculty of Science and Technology State Islamic University of Malang.

Supervisor : (I) Fachrul Kurniawan,M, M.T, (II) Hani Nurhayati, M.T

Key words: Greedy, Container, Game, Simulation Game

Game is a kind of 'long period' entertainment which will be everlasting thing. Usually some genres of game will be not only entertaining such as fighting game, adventure game, casual game, but also educating even simulating of strategy. It is because that a game has a role to stimulate the brain growth, increase the concentration, and train children in a process of problem solving appropiately. Thus, a game can be used by everyone in all circles including children, youth and oldster.

Simulation game is a kind of game which has strategy in solving the problem. In this research, the researcher designs a game that give a beneficial in structuring process of goods in container. Furthermore, the Simulation Game of Goods Srtucturing in Container is designed in a base of mobile. Player is demanded to find any strategy in order to accomplish the problem that goods can be well-structured in container considering the destination and the weight as well as possible. There are several optimum methods which one of them is Algorithm of Greedy.

In this research, Greedy will be applied as an optimation problem methods that have the structured goods in container appropiated with the destination and the weight or not. The calibration is done by mobile with android platform.

خلاصة

ابنه، سكوت .٢٠١٦-**الإعداد "محاكاة لعبة الأشياء "في الحاوية باستخدام "خوارزمية الجشع** ."أطروحة .
ومن المؤسف إدارة الكمبيوتر كلية الهندسة للعلوم والتكنولوجيا في جامعة الدولة الإسلامية مولانا مالك إبراهيم.

المشرف :(ط) فاتشرول كورنياوان، م.، م .ت.، (ثانيا) هاني نورهاياتي، ت .م.

الكلمات الرئيسية :الجشع، حاويات، والألعاب، وألعاب المحاكاة

ألعاب أو لعبة ترفيه 'كل الوقت 'يمكن أن يقال أن تكون مظلمة بالوقت .الألعاب في بيرونتوكان لمتعة الترفيه أنواع اللعبة مثل القتال، مغامرة، عارضة الألعاب ولكن في بعض الأحيان الألعاب مخصصة للتربية، حتى الاستراتيجيات المحاكاة .نظراً لكون اللعبة مهم أيضا لشحذ نمو الدماغ، وتحسين التركيز، وتدريبهم على حل المشاكل بشكل صحيح وبسرعة .ولذلك يمكن أن يكون الاستمتاع باللعبة من كافة الدوائر سواء القليل للأطفال، حتى الكبار الذين هم القديمة.

ألعاب المحاكاة يمكن أن يشار إليها باسم نوع من اللعبة التي لديها استراتيجية في حل المشكلة .على البحث لهذه المرة أيضا الباحثين تصميم لعبة لحل مشاكل عملية ترتيب البضائع في حاويات .تصميم لعبة "محاكاة الإعداد من العناصر الموجودة "في هذه الحاوية المستندة إلى الجوال .وادعي لاعب لإيجاد استراتيجيات في حل عناصر الإعداد في حاوية استناداً إلى موقع الوجهة ووزن البضائع في النظام الأمثل هذا .هناك حاليا عدة طرق للاستفادة المثلى من واحد هو "خوارزمية الجشع."

في هذه الدراسة، سيتم تطبيق طماع كأسلوب للتحسين مشكلة ما إذا كانت البضاعة بالفعل في الموقع المناسب في حاوية التخطيط الوجهة ووزن البضاعة فعلا الأمثل والمناسب على الحاوية .ويتم اختبار على الأجهزة النقالة باستخدام منصة الروبوت.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sebuah perusahaan yang bergerak di bidang logistik atau ekspedisi dalam mencari sebuah profit harus berfikir keras untuk meningkatkan efisensi dan laba yang di dapat sehingga hal tersebut dapat meningkatkan daya saing terhadap jasa pengiriman barang yang lain. Dapat diambil contoh misalnya permasalahan dalam proses muat barang pada sebuah kontainer, hal tersebut sering terjadi karena penyusunan letak barang dalam sebuah kontainer yang tidak optimal dimana banyak terdapat ruang kosong yang seharusnya itu bisa diletakkan barang lagi dan akan membutuhkan biaya yang berlipat ganda untuk mengangkut sisa barang yang seharusnya bisa dimasukkan hanya dalam satu kontainer saja apabila penyusunan letak barang didalamnya dioptimalkan. Kondisi barang seperti berat dan jenis barang juga menjadi permasalahan tersendiri dalam proses peletakan barang pada sebuah kontainer. Tidak jarang sekali terjadi kerusakan pada barang dikarenakan kurang diperhatikannya berat dari masing-masing barang. Hal tersebut sesuai dengan firman Allah SWT dalam Al-Qur'an Surat Al-Isra' ayat 27 dan Al-Quran Surat Al-Hujarat ayat 6 :

إِنَّ الْمُبَذِّرِينَ كَانُوا إِخْوَانَ الشَّيَاطِينِ وَكَانَ الشَّيْطَانُ لِرَبِّهِ كَفُورًا

"*Sesungguhnya orang-orang yang pemboros itu adalah saudara setan dan setan itu sangat ingkar kepada Tuhannya*" [Q.S.Al Isra' : 27]

يَا أَيُّهَا الَّذِينَ آمَنُوا إِن جَاءكُمْ فَاسِقٌ بِنَبَأٍ فَتَبَيَّنُوا أَن تُصِيبُوا قَوْماً بِجَهَالَةٍ فَتُصْبِحُوا عَلَى مَا فَعَلْتُمْ نَادِمِينَ

"*Hai orang-orang yang beriman, jika dating kepadamu orang fasik membawa suatu berita, maka periksalah dengan teliti agar kamu tidak menimpakan suatu musibah kepada suatu kaum tanpa mengetahui keadaannya yang menyebabkan kamu menyesal atas perbuatanmu itu*" [Q.S.Al Hujurat : 6]

Gambaran kompleksitas kondisi real/nyata yang terdapat pada suatu industri nyata mengenai penataan suatu barang dapat dilihat dari suatu situasi tertentu dimana di dalam kontainer barang-barang dipisahkan berdasarkan tujuan lokasi. Jika terdapat ruang kosong, petugas akan menambahkan barang dengan syarat memiliki tujuan yang sama atau berdekatan dan barang yang dimasukkan memiliki volume yang sesuai dengan ruang kosong yang ada. Jika terdapat barang lebih yang seharusnya barang itu masuk di kontainer dan itu hanya beberapa barang, daripada petugas menggunakan kontainer lain untuk mengangkut barang tersebut, petugas akan mencari cara agar barang itu masuk tetapi jika tidak barang-barang yang tadi sudah tertata di dalam kontainer diturunkan lagi dan ditata ulang.

Proses penataan barang dalam sebuah kontainer adalah hal yang cukup rumit. Terdapat berbagai aspek yang saling berkaitan dan harus dilibatkan dalam proses tersebut antara lain adalah besarnya dimensi ruangan yang digunakan sebagai batasan dari penyusunan peletakan barang-barang, distribusi peletakan barang-barang yang diharapkan memenuhi semua titik koordinat dalam ruangan, kemudian peletakan barang-barang ini akan semakin rumit jika

melibatkan titik berat dari masing-masing barang. Untuk mencapai hal tersebut, perlu adanya pengetahuan dan pemahaman terhadap perkembangan ilmu dan teknologi yang terkait. Sejalan dengan hal itu sekarang ini muncul inovasi metode pembelajaran modern selain model konvensional yaitu pembelajaran yang dilakukan melalui *Game*. *Massachussets Insitute of Technology* (MIT) berhasil membuktikan bahwa *Game* sangat berguna untuk meningkatkan logika dan pemahaman pemain terhadap suatu masalah melalui proyek *Game* yang dinamai *Scratch* (*http://scratch.mit.edu*).

Game serius berbasis *mobile* ini didesain untuk mensimulasikan permasalahan yang ada sehingga diperoleh ilmu yang dapat digunakan untuk menyelesaikan permasalahan tersebut. *Game* serius ini dapat digunakan sebagai salah satu pola pembelajaran *learning by doing*. Berdasarkan pola yang dimiliki oleh *Game* tersebut, pemain dituntut untuk belajar dalam menyelesaikan permasalahan yang ada. Environment tools (status *Game*, instruksi dan tools) yang disediakan pada *Game* akan membimbing sang pemain secara aktif menggali informasi untuk memperkaya pengetahuan dan strategi saat bermain (Hendriyono & Ahmad, 2012).

Dewasa ini kebutuhan akan simulator dan permainan dalam bidang logistik, distribusi dan supply chain mengalami perkembangan yang cukup pesat dengan variasinya sesuai kebutuhan (Alan Rushton et. Al , 2006). Van Horne (2004) menyebutkan tujuan perancangan simulator adalah untuk menganalisa dinamika keputusan logistik, antara lain :

1. Pentingnya informasi yang akurat dan tepat waktu,

2. Memahami efek penundaan waktu,
3. Proses transfer *know how* dari professional ke praktisi bisnis atau sebaliknya dari aktivitas logistik.

Pada penelitian kali ini, simulasi komputer (simulator) yang akan dikembangkan berfokus pada penempatan dan penataan sebuah barang di suatu kontainer. Dimana simulator ini dikemas dalam bentuk *Game Serious*, sehingga diharapkan seorang pemain dapat tertarik untuk memainkan dan menyelesaikan permasalahan yang ada didalamnya. Selain itu, seorang pemain akan mendapatkan sebuah edukasi mengenai permasalahan yang akan disampaikan oleh *Game* yang akan dibuat. Seorang pemain akan dapat mengetahui tingkat kemampuannya berdasarkan level dan nilai yang akan didapatkannya dalam memainkan *Game* ini.

Game serius ini dirancang untuk dapat mensimulasikan suatu kondisi permasalahan yang terjadi dalam proses penataan barang di kontainer. Dalam hal ini pemain diharuskan untuk menyelesaikan dan memberikan solusi terhadap permasalahan tersebut. *Game* ini dirancang sedemikian rupa sehingga dapat membangun pemikiran heuristik dari pemain sehingga dapat mengembangkan imajinasi dan membangun pemikiran secara sistematik terhadap permasalahan yang ada.

Pengembangan dari suatu simulator yang ada ini, dapat direpresentasikan dalam suatu bentuk *Game* serius berbasis *mobile*. Dimana dengan adanya *Game* serius ini dapat dijadikan sebagai suatu alat dalam menghadapi permasalahan penataan barang seperti mengambil keputusan penempatan dan

penataan suatu barang pada kontainer yang di kemas dalam bentuk yang lebih menarik dan menghibur.

1.2 Identifikasi Masalah

Berdasarkan penjelasan pada latar belakang di atas, maka identifikasi masalah dari penilitian ini adalah :

a. Bagaimana merancang suatu *Game* serius mengenai penataan barang dalam kontainer ?
b. Bagaimana mengimplementasikan algoritma *Greedy* pada *Game* serius penataan barang dalam kontainer ?

1.3 Batasan Masalah

Adapun batasan masalah pada penelitian ini adalah sebagai berikut:

a. *Game* ini dibangun dengan *platform* Android.
b. Kontainer yang digunakan hanya kontainer yang memiliki ukuran 20 feet.
c. Barang berbentuk rectangular *box* (balok atau kubus).
d. Data yang digunakan merupakan data produksi yang diambil dari suatu perusahaan logistik di Surabaya.
e. Parameter yang ditentukan dalam proses penyusunan barang dalam kontainer adalah jenis barang, berat barang dan tujuan lokasi pengiriman.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut :

a. Membangun sebuah *Game* serius yang dapat menjadi media latihan dan pembelajaran mengenai problem penataan barang pada kontainer.
b. Mengimplementasikan algoritma *Greedy* sebagai penyelesaian untuk menemukan solusi yang optimal dalam mengatur tata letaknya suatu barang yang masuk dalam kontainer.

1.5 Manfaat penelitian

Manfaat pembuatan aplikasi *Game* ini bagi peneliti adalah dapat memberikan metode alternatif untuk menyelesaikan permasalahan proses muat barang dalam kontainer menggunakan algoritma *Greedy*. Sedangkan manfaat lainnya adalah menjadi pertimbangan suatu perusahaan untuk optimasi proses muat barang, dapat digunakan sebagai referensi dalam membuat suatu metode yang lebih optimal dan lebih singkat dalam penyusunan barang ini, dan dapat menambah pengetahuan mengenai algoritma *Greedy* dan langkah-langkah menyelesaikan masalah ini

BAB II

KAJIAN PUSTAKA

2.1 Teori Penunjang

2.1.1 Permainan (*Game*)

Dalam kamus Bahasa Indonesia "*Game*" diartikan sebagai permaian. Sebuah *Game* merupakan sebuah sistem dimana pemain terlibat dalam konflik buatan, disini pemain berinteraksi dengan sistem dan konflik dalam permainan yang merupakan rekayasa atau buatan. Situasi bermain yang terkait dengan beberapa aturan atau tujuan tertentu. Ada *rule of Games* yang disepakati bersama dan ditentukan dari luar untuk melakukan kegiatan dalam tindakan yang bertujuan untuk membatasi perilaku pemain dan menentukan permaianan (Bettelheim, dalam Hurlock, 1978). *Game* biasanya diperuntukkan untuk hiburan. Tetapi terkadang juga *Game* ditujukan untuk edukasi, karena sebenarnya *Game* juga penting untuk mengasah perkembangan otak, meningkatkan konsentrasi, dan melatih memecahkan masalah dengan tepat dan cepat.

Menurut Mayke S. Tedjasaputra (2001), sebuah permainan bisa dikelompokkan dalam kelompok kegiatan bermain aktif, dimana kegiatan ini memberikan kesenangan dan kepuasan pada anak melalui aktivitas yang mereka lakukan sendiri. Kegiatan ini banyak melibatkan aktifitas tubuh atau gerak-gerakan tubuh.

Sejalan berkembangannya zaman, terdapat jenis-jenis mainan dengan alat bantu alat-alat elektronik seperti komputer, video *Game*s atau play station dan mesin-mesin simulator. Dalam kamus bahasa Inggris Echols (1997) kata video *Game* berasal dari kata video dan *Game*. Sedangkan video adalah penampilan gambar (visual) dengan bantuan alat elektronik. Video *Game* adalah permainan yang dimainkan melawan computer (Bernard Perron et. Al. 2009). *Mobile Game* merupakan sebuah permainan video yang dimainkan pada *mobile* phone, ataupun PDA *(Personal Digital Assistant)* dan bukan pada konsol permainan maupun mesin ding-dong. *Mobile Game* adalah *Game* yang tidak hanya dapat dimainkan melalui *mobile phone*, namun dapat dikembangkan dalam berbagai macam *mobile handset* seperti *PDA, Symbian OS, dan Microsoft's Smartphone* (Lam, 2003).

Menurut Mayke dalam bukunya "Bermain, mainan dan permainan", sebenarnya yang dipicu alat permainan elektronik adalah kemampuan anak yang bereaksi dengan cepat dan dengan latihan yang terus-menerus (*drilling*) anak menjadi tangkas, tetapi belum tentu anak dapat belajar dari kesalahan yang dibuatnya. Berdasarkan platformnya, *Game* dibagi menjadi beberapa jenis, yaitu :

1. Arcade *Game*

   Game ini di Indonesia disebut ding-dong.

2. PC *Game*s

   Game ini dimainkan dengan menggunakan Personal Komputer.

3. Console *Game*s

Game ini dimainkan dengan menggunakan console tertentu, misalnya *Playstation* 2, *Playstation* 3, dan Nintendo Wii.

4. Handheld *Game*s

   Game ini dimainkan menggunakan console khusus yang bisa dibawa kemana-mana, contohnya PSP.

5. *Mobile Game*s

   Jenis *Game* ini hanya dimainkan khusus menggunakan *mobile phone.*

Menurut Philips Hanna, terdapat beberapa jenis *Game* berdasarkan genrenya, yang dikelompokkan sebagai berikut :

1. Game Serius

   Contoh permainan yang termasuk dalam *Game* Serius adalah simulasi konstruksi dan manajemen, simulasi kendaraan seperti yang diterapkan pada permainan balapan, perang, dan luar angkasa.

2. Edukasi

   Contohnya adalah edu*Game*s yang dibuat dengan tujuan spesifik sebagai alat pendidikan, baik itu untuk belajar mengenal warna untuk balita, mengenal huruf dan angka, matematika, sampai belajar Bahasa asing. Developer yang membuatnya, harus memperhitungkan berbagai hal agar *Game* ini benar-benar dapat mendidik, menambah pengetahuan dan meningkatkan ketrampilan yang memainkannya.

3. Entertainment

a. Aksi – Shooting, (tembak-tembakan, atau hajar-hajaran bisa juga tusuk-tusukan, tergantung cerita dan tokoh di dalamnya). *Game* jenis ini sangat memerlukan kecepatan reflex, koordinasi mata-tangan, juga waktu, inti dari *Game* jenis ini adalah tembak-tembakan.

b. Fighting (pertarungan), ada yang mengelompokan *Game* fighting di bagian Aksi, namun peneliti berpendapat berbeda, jenis ini memang memerlukan kecepatan reflex dan koordinasi mata-tangan, tetapi inti dari *Game* ini adalah penguasaan jurus (hafal caranya dan lancer mengeksekusinya), pengenalan karakter dan waktu sangatlah penting. Dan berbeda seperti *Game* Aksi pada umunya hanya melawan Artificial Intellegence atau istilah umumnya melawan computer saja, pemain jenis fighting *Game* ini baruu teruji kemampuan sesungguhnya dengan melawan pemain lainnya.

c. Petualangan, *Game* murni petualangan lenig menekankan pada jalan cerita dan kemampuan berpikir pemain dalam menganalisa tempat secara visual, memecahkan teka-teki maupun menyimpulkan rangkaian peristiwa dan percakapan karakter hingga penggunaan benda-benda tepat pada tempat yang tepat.

d. Role Playing, *Game* jenis ini sesuai dengan terjemahannya, bermain peran, memiliki penekanan pada tokoh/ peran perwakilan pemain di dalam permainan, yang biasanya adalah tokoh utamanya, dimana sering kita memainkannya, karakter tersebut dapat berubah dan berkembang ke arah yang diinginkan pemain (biasanya menjadi semakin hebat, semakin kuat,

semakin berpengaruh, dll) dalam berbagai parameter yang biasanya ditentukan dengan naiknya *Level*.

e. Casual *Game*s, sesuai namanya, *Game* yang casual itu tidak kompleks, mainnya rileks dan sangat mudah untuk dipelajari. Jenis ini biasanya memerlukan spesifikasi komputer yang standart pada jamanya dan ukurannya tidak lebih dari 100 MB karena biasanya dapat di download versi demo-nya di website resminya. Genre permainannya biasanya puzzle atau *action* sederhana.

f. Multiplayer Online, *Game* yang dapat dimainkan secara bersamaan oleh lebih dari 2 orang (bahkan dapat mencapai puluhan ribu orang dalam satu waktu) membuat pemian dapat bermain bersama dalam satu dunia virtual dari sekedar chatting hingga membunuh naga bersama teman yang entah bermain dimana. Umumnya permainan tipe ini dimainkan di PC dan bertema RGP, walau ada juga yang bertema musik atau *action*.

A. Aksi (*Action*)

Jenis *Game* ini pada umumnya membutuhkan reflek yang cepat, akurasi, dan juga waktu untuk menyelesaikan suatu masalah. *Gameplay* jenis *Game* ini ditekankan pada suatu pertempuran atau peperangan. Jenis *Game* ini juga memiliki banyak sub jenis yaitu: *fighting Game* (mortal combat, bloody roar, dll), *platform Game* (crash bondicoot, Mario, dll).

B. Penembak (*Shooter*)

Jenis *Game* ini difokuskan terhadap pertempuran yang kebanyakan menggunakan senjata militer seperti rudal, pistol dan lain-lain. Sub *genre*

pada jenis *Game* ini dibedakan sesuai dengan sudut pandang pemain, yaitu: *Third Player Shooter* (Lost Planet), *First Player Shooter* (Counter Strike), dll.

C. Aksi-Petualangan (*Action-Adventure*)

Jenis *Game* ini memadukan antara dua jenis *Game* yaitu aksi dan petualang yang mana terdapat rintangan-rintangan jangka panjang dan membutuhkan kunci atau barang spesial.

D. Petualangan (*Adventure*)

Jenis *Game* ini tidak mengarah kepada reflek dan tindakan cepat, namun dibutuhkan untuk memecahkan berbagai masalah dengan cara berinteraksi dengan orang-orang dan lingkungan.

E. Peran (*Role Playing*)

Jenis *Game* ini memiliki elemen *Gameplay* yang sangat terkait pada RPG dimana pemain menjadi seseorang yang memiliki kronologi kehidupan, memiliki tujuan utama dan sering juga dijumpai tujuan sampingan. Akuisisi poin seperti tingkat, kekuatan sangat mempengaruhi dalam jenis *Game* ini.

F. Simulasi (*Simulation*)

Jenis *Game* yang memiliki banyak sub jenis yang pada dasarnya jenis *Game* ini mensimulasikan aspek realitas maupun fiksi, salah satu sub jenis nya yaitu simulasi kendaraan dimana pemain dituntut untuk bisa mengoperasikan suatu kendaraan sesuai dengan keadaan yang sesungguhnya (Bus Simulation).

G. Strategi (Strategy)

Jenis *Game* yang mengasah keterampilan dalam berfikir cerdas untuk menyelesaikan suatu tantangan karena jenis *Game* ini difokuskan pada *Gameplay* yang membutuhkan pemikiran dan perencanaan matang. Menurut Andrew Rollings, "pemain diberikan pandangan dewa dari dunia permainan, secara tidak langsung mengendalikan unit di bawah komandonya". Salah satu *Game* strategi yang terpopuler nomor 3 dunia setelah WoW (World Of Warcraft) yaitu DotA 2.

H. Olahraga (*Sport*)

Jenis *Game* yang menuntut memiliki keterampilan pemain dalam melakukan pertandingan olahraga virtual seperti sepak bola, basket, dan lain-lain.

2.1.2 Genre *Game* Serius

Dari semua jenis permainan yang ada, masing-masing memiliki tingkat kesulitan dan kemudahannya, jika bukan algoritmanya maka akan mudah dalam hal animasinya, akan tetapi *Game*s Serius bisa disebut sebagai jenis permainan yang paling sulit, baik algoritma pembuatannya maupun animasinya. Permainan jenis ini juga yang paling membuat pusing dibandingkan dengan permainan jenis lainnya. Algoritmanya sangat sulit sebab harus memperhitungkan semua kejadian dalam kondisi sebenarnya. Berbagai efek animasi yang dibuat tidak cukup bermodalkan ahli grafik dan algoritma saja, tetapi sedikitnya harus mengerti persoalan matematika, teknik dan fisika.

Menurut Philips Hanna, bentuk dari *Game* Serius ini memiliki tujuan sebagai berikut :

1. Bertujuan untuk mensimulasikan kegiatan fisik seperti menerbangkan pesawat terbang (Microsoft Flight Simulator), bermain golf atau sepak bola, dll.
2. Bertujuan untuk memberikan simulasi bentuk manajemen, misalnya *Game* manajemen sepak bola, manajemen kota (SimCity), railroading, dll.
3. Sering kali *Game* serius ini dimaksudkan untuk menjadi menyenangkan sebagai lawan akurat, misalnya Wing Commander dan X3 yang terkenal *Game* serius dengan ruang tempur.

2.1.3 Kontainer

Di Indonesia kontainer dikenal dengan nama peti kemas yang terbuat dari bahan logam dan beberapa macam ukuran dan jenis. Peti kemas atau kontainer dapat dikatakan sebagai "*the moving get down*" yaitu gudang mini yang bergerak dari satu tempat ke tempat lain sebagai akibat dari adanya pengangkutan.

Peti kemas dapat menyederhanakan proses bongkar muat, menurukan biaya dan meningkatkan pengguanaan kapasitas peralatan angkutan. Peti kemas dikatakan sebagai peralatan sistem pengangkutan (*working tool*) yang bersifat padat modal penggunaan peti kemas dapat terjadi kerja sama dan bentuk gabungan berbagai jenis alat angkutan.

A. Jenis Peti Kemas

Menurut (Ella, 2011) peti kemas dapat dibagi dalam empat kelompok, yaitu :

1. *General Cargo*

   General cargo container adalah peti kemas yang dipakai untuk mengangkut muatan umum (*general cargo*), misalnya : kayu, kain, rotan, handicraft, dll.

2. *Thermal*

   Thermal container adalah peti kemas yang dilengkapi dengan pengatur suhu untuk muatan tertentu. Contoh muatanya seperti minuman dingin, buah-buahan, sayur-sayuran, hewan, dll.

3. *Tank*

   Tank container adalah tangki yang ditempatkan dalam kerangka peti kemas yang digunakan untuk muatan cair (*bulk liquid*) maupun gas (*bulk gas*), misalnya : minyak dan gas bumi.

4. *Dry*

   Dry bulk adalah *general purpose container* yang dipergunakan khusus untuk mengangkut muatan curah. Untuk memasukkan muatan melalui lubang bagian atas petikemas sedangkan mengeluarkan muatan melalui lubang atau pintu dibagian bawah peti kemas. Muatannya missal : beras, biji gandum, pupuk urea, gula pasir, bahan baku, kedelai.

B. Ukuran Peti Kemas

Internasional Standard Organization (ISO) telah menetapkan ukuran-ukuran dari peti kemas sebagai berikut :

a. *Container 20' Dry Freight (20 feet)*

Ukuran luar : 20' (p) x 8' (l) x 8'6" (t) atau
: 6.058 x 2.438 x 2.591 m

Ukuran dalam : 5.919 x 2.340 x 2.380 m

Kapasitas : *Cubic Capacity* : 33 Cbm
Pay Load : 22,1 ton

b. *Container 40' Dry Freight (40 feet)*

Ukuran luar : 40' (p) x 8' (l) x 8'6" (t) atau
: 12.192 x 2.438 x 2.591 m

Ukuran dalam : 12.045 x 2.309 x 2.379 m

Kapasitas : *Cubic Capacity* : 67,3 Cbm
Pay Load : 27,396 ton

c. *Container 40' High Cube Dry*

Ukuran luar : 40' (p) x 8' (l) x 9'6" (t) atau
: 12.192 x 2.347 x 2.684 m

Ukuran dalam : 12.056 x 2.347 x 2.684 m

Kapasitas : *Cubic Capacity* : 76 Cbm
Pay Load : 29,6 ton

C. Cara Pemuatan Barang dalam Peti Kemas

Cara pemuatan barang dalam kontainer terbagi menjadi 2 sistem, yaitu :

1. Sistem *Full Container Loaded*

   Dengan sistem FCL dalam kontainer harus dimasukkan atau dipadatkan 1 (satu) partai barang atau lebih, akan tetapi untuk hanya satu alamat penerima di pelabuhan tujuan. Dengan sistem ini seorang *shipper* menyewa kontainer dan mengirim ke alamat untuk 1 penerima, yaitu untuk 1 atau beberapa partai barang yang dipadatkan dalam 1 kontainer. Jadi, untuk cara kondisi pengiriman tersebut, kontainer yang telah berisi muatan dari *bonded warehouse, shipper* akan menuju *Container Yard* dan menunggu pengapalannya saja.

2. Sistem *Less Than Container Loaded*

   Dengan sistem LCL ini maka ke dalam kontainer hanya dapat dipadatkan barang-barang yang terdiri dari beberapa *shipper* dan/atau *consignee* dengan cara dimana *shipper* mengirim barangnya ke CFS yaitu lapangan timbun yang dekat dermaga dimana kapal yang bersangkutan akan bertambat dengan menggunakan truk angkutan lainnya. Setelah semua kontainer terkumpul di CFS dan kemudian melalui prosedur bea-cukai maka barang tersebut dimuat ke dalam kontainer sesuai dengan *destination* dari barang tersebut. Dengan sistem ini "*door to door service"* dapat terlaksana dengan baik.

2.1.4 *NP-Hard dan NP-Complete*

Waktu yang dibutuhkan algoritma terbaik untuk menghasilkan solusi dari banyak problem (permasalahan) dapat dibagi menjadi dua kelompok, yaitu :

- *P (Polynomial Problem)*

  Polynomial Problem adalah suatu permasalahan dimana waktu yang dibutuhkan untuk menghasilkan solusi terbatas pada waktu *polynomial* dalam tingkat kecil. Contohnya : permasalahan evaluasi *polynomial* dengan *O(n),* pengurutan dengan *O(n log n)* dan string editing dengan *O(mn).*

- *NP (Non Polynomial)*

  Dalam pencarian untuk mengembangkan algoritma yang efisien, tidak satupun yang dapat mengembangkan algoritma dengan waktu *polynomial* untuk permasalhan *NP*. Hal ini sangat penting karena algoritma yang waktu pencarian solusinya lebih besar dari *polynomial* (biasanya waktu pencarian adalah eksponensial) membutuhkan waktu yang cukup lama untuk menjalankan permasalhan skala menengah. Contoh *NP* adalah permasalahan *travelling salesman person* dengan $O(n^2 2^n)$) dan permasalahan *knapsack* dengan $O(2^{n/2})$.

  Suatu permasalahan yang termasuk kedalam *NP-Complete* memiliki sifat yang dapat dipecahkan dalam waktu polynomial jika dan hanya jika seluruh permasalahan *NP-Complete* juga dapat dipecahkan dalam waktu *polynomial*. Seluruh permasalahan *NP-Complete* merupakan permasalahan

NP-Hard, tetapi sebagian *NP-Hard* belum tentu menjadi permasalhan *NP-Complete* (Horowitz et. Al. 1998).

2.1.5 Algoritma *Greedy*

Secara harfiah *Greedy* artinya rakus atau tamak, sifat yang berkonotasi negatif. Orang yang memiliki sifat ini akan mengambil sebanyak mungkin atau mengambil yang paling bagus atau yang paling mahal. Sesuai dengan arti tersebut, prinsip dari algoritma *Greedy* sendiri adalah "*take what you can get now*". Algoritma *Greedy* adalah algoritma untuk menyelesaikan suatu permasalahan secara bertahap (Brassard, 1996). Suatu permasalahan dengan *n* masukkan data dilakukan secara bertahap. Pertama dilakukan pemilihan solusi yang mungkin kemudian dari himpunan solusi yang mungkin tersebut akan diperoleh solusi optimal.

Metode ini bekerja secara bertahap dengan memperhatikan setiap input data pada setiap keadaan. Pada setiap tahap, dibuat keputusan dengan memperhatikan ada atau tidak sebuah input data yang memberikan solusi optimal, dan memperhatikan pula urutan data dalam proses pengambilannya. Pendekatan yang digunakan pada algoritma *Greedy* adalah membuat pilihan yang dapat memberikan perolehan yang 10 terbaik yaitu dengan membuat pilihan optimum lokal pada setiap langkah dengan tujuan bahwa sisanya mengarah ke solusi optimum global (Springer V, 2005).

Berikut elemen-elemen yang digunakan dalam penerapan algoritma *Greedy* :

- Himpunan kandidat (*C*)

Himpunan berisi elemen-elemen pembentuk solusi.

- Himpunan solusi (*S*)

  Himpunan ini berisi kandidat-kandidat yang terpilih sebagai solusi dari permasalahan optimasi yang akan diselesaikan.

- Fungsi seleksi (*select()*)

  Fungsi ini akan memilih kandidat yang paling memungkinkan untuk mencapai solusi optimal.

- Fungsi kelayakan (*feasible()*)

  Fungsi ini akan memeriksa apakah suatu kandidat yang dipilih dapat memberikan solusi yang layak. Dalam hal ini yaitu apakah kandidat tersebut bersama dengan himpunan solusi yang sudah terbentuk tidak melanggar *constraints* yang ada.

- Fungsi Solusi (*solution()*)

  Fungsi ini akan mengembalikan nilai Boolean. True jika himpunan solusi yang terbentuk merupakan solusi yang lengkap; False jika himpunan solusi belum lengkap.

- Fungsi obyektif (*objective()*)

  Fungsi yang mengoptimalkan solusi.

Algoritma *Greedy* dapat menyelesaikan beberapa masalah dalam kehidupan nyata, seperti halnya permasalahan dalam bidang transportasi. Salah satu permasalahan di bidang transportasi yang muncul adalah bagaimana suatu perusahaan mengatur produk apa yang harus diangkut agar memperoleh keuntungan yang maksimal, sementara perusahaan

sendiri memiliki problematika atau kendala yaitu kapasitas angkut dari kendaraan yang sangat terbatas. Persoalan optimisasi transportasi seperti itu sering dianalogikan sebagai *Knapsack Problem* (Pisinger D. , Algorithm For Knapsack Problems, 1995).

Terdapat beberapa strategi *Greedy* yang dapat digunakan untuk memilih objek yang akan dimasukkan ke dalam *M* antara lain.

a. *Greedy by profit*

   Algoritma *Greedy by profit*.

   1) tetapkan nilai kapasitas maksimum *knapsack*;
   2) urutkan objek-objek berdasarkan keuntungan (*profit*) dari yang terbesar;
   3) isi *knapsack* dengan objek yang memiliki keuntungan terbesar terlebih dahulu;
   4) ambil satu-persatu objek yang dapat ditampung sampai kapasitas *knapsack* penuh;
   5) hitung jumlah bobot dan keuntungan.

b. *Greedy by weight*

   Algoritma *Greedy by weight*.

   1) tetapkan nilai kapasitas maksimum *knapsack*;
   2) urutkan objek-objek berdasarkan berat dari yang teringan;
   3) isi *knapsack* dengan objek yang memiliki berat teringan terlebih dahulu;

4) ambil satu-persatu objek yang dapat ditampung sampai kapasitas *knapsack* penuh;
5) hitung jumlah bobot dan keuntungan.

c. *Greedy by density*

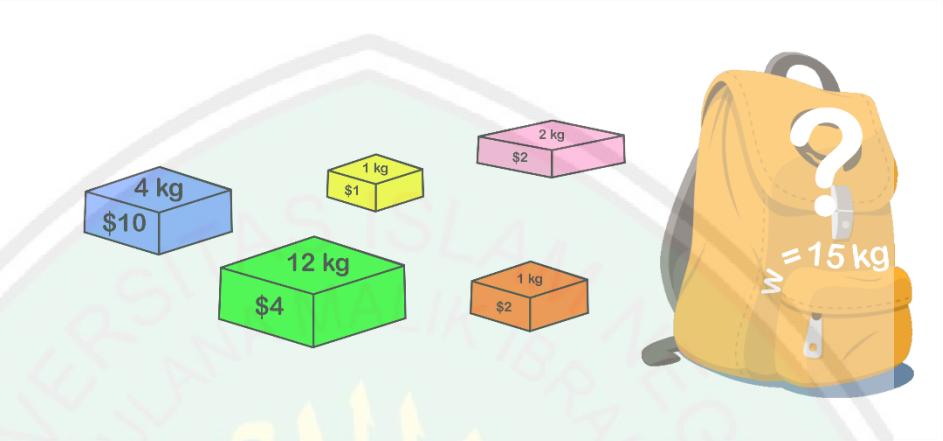
Algoritma *Greedy by density*.

1) tetapkan nilai kapasitas maksimum *knapsack*;
2) hitung rasio (p_i/w_i) dari tiap-tiap barang;
3) urutkan objek-objek berdasarkan rasio (*density*) terbesar terlebih dahulu;
4) ambil satu-persatu objek yang dapat ditampung sampai kapasitas *knapsack* penuh;
5) hitung jumlah bobot dan keuntungan.

2.1.6 *Knapsack Problem*

Knapsack adalah tas atau karung. Karung di gunakan untuk memuat suatu barang atau benda. Dan tentunya tidak semua barang bisa di tamping dalam karung tersebut. Karung tersebut hanya bisa menyimpan beberapa barang dengan total ukurannya (*weight*) lebih kecil atau sama dengan ukuran kapasitas karung. Ilustrasi permasalahan dapat dilihat pada Gambar 2.1. Pada Gambar 2.1 terlihat terdapat sebuah tas berkapasitas 15 kg. Dan terdapat 5 barang dengan berat dan keuntungan masing-masing.

Yang menjadi persoalan adalah barang mana saja yang harus di masukkan ke dalam tas.



Gambar 2.1 Ilustrasi *Knapsack Problem*

Knapsack problem secara matematis dapat ditulis sebagai berikut :

Diberikan bobot *knapsack* adalah M. diketahui n buah objek yang masing-masing bobotnya adalah w1,w2,….,wn. Tentukan nilai bi sedemikian sehingga

$$M = b1w1 + b2w2 + ….. + bnwn$$

Yang dalam hal ini, bi bernilai 0 atau 1. Jika b1=1, berart objek I di masukkan ke dalam *knapsack*, sebaliknya jika b1=0, objek I tidak di masukkan. Berhubung nilai bi 0 dan 1 maka masalah ini sering juga disebut sebagai *Knapsack* 0/1.

Dalam teori algoritma, persoalan knapsack termasuk ke dalam kelompok NP-*Complete*. Persoalan yang termasuk NP-*Complete* tidak dapat di pecahkan dalam orde waktu.

2.2 Penelitian Terkait

Terdapat beberapa penelitian yang terkait dengan penelitian yang dilakukan, yaitu :

a. **Pengembang Rancang Bangun *Game* Edukasi Logistik "STOWA*GAME*" Mengenai Penataan Kontainer Di Bay Kapal**

Penelitian ini dilakukan Hendriono dan ahmad, mahasiswa jurusan Teknik Industri, Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Penelitian ini bertujuan untuk mensimulasikan permasalahan yang terjadi dalam penataan kontainer di bay kapal. Pada penelitian ini, pengembang *Game* edukasi akan fokus pada permasalahan *loading* kontainer di suatu bay kapal. Permasalahan yang dihadapi dalam perencanaan *loading* di kapal biasa disebut dengan *Manster Bay Plan Problem* (MBPP) atau *Stowage Planning Problem* (SPP). Kompleksitas dari permasalahan MBPP dapat dilihat dari konstrain yang ada, yaitu berat kontainer, tujuan kontainer, ukuran kontainer, tipe kontainer, dan aturan peletakan kontainer.

b. **Pengembangan Algoritma *Greedy* Untuk Optimalisasi Penataan Peti Kemas Pada Kapal Pengangkut**

Penelitian ini di lakukan oleh Rinaldi dan Christian, mahasiswa program studi teknik informatika Institut Teknologi Bandung . Penelitian ini bertujuan untuk mengoptimalkan dalam proses bongkar muat kontainer pada kapal pengangkut, sehingga cepat dan tidak mengganggu stabilitas kapal pengangkut. Dalam penelitian ini pengembangan untuk optimalisasi bongkar muat kontainer di terapkan dengan menggunakan algoritma *greedy*.

Hasil dari penelitian ini adalah secara umum, algoritma *greedy* dalam penataan kontainer (peti kemas) sudah di buat cukup baik dan menghasilkan hasil yang tepat guna, namun terdapat kesalahan kecil pada system pengaksesan database, yaitu terlalu seringnya algoritma mengakses database yang menyebabkan algoritma yang di jalankan memerlukan waktu eksekusi yang lebih lama.

c. **Perancangan Program Simulasi Optimasi Penyusunan Barang dalam Kontainer Menggunakan Algoritma *Greedy* (Studi Kasus : Best Global Ekspress)**

Penelitian ini dilakukan oleh Gozali mahasiswa jurusan teknik informatika Binus University pada tahun 2010. Dalam penelitian ini bertujuan untuk membuat sebuah program perangkat lunak yang dapat mengoptimalkan letak penyusunan barang dalam kontainer beserta langkah-langkah penyelesaiannya, sehingga dapat memudahkan penyelesaian masalah yang terjadi pada kenyataan nyata dan juga dapat menekan biaya pengiriman barang.

Penelitian ini disimulasikan dalam ruang tiga dimensi dengan bidang berupa balok yang memiliki panjang, lebar dan tinggi tertentu, dimana dalam proses penyusunan barangnya harus memperhatikan hal-hal seperti beban maksimal yang dapat ditampung oleh kontainer, volume maksimal yang dapat ditampung oleh kontainer, dan keseimbangan posisi barang terhadap barang dibawahnya. Hasil dari penelitian ini adalah dapat diterapkannya algoritma *Greedy* dalam mengatur tata letak suatu barang

dalam suatu kontainer secara optimal. Dalam penelitian ini tidak dilakukan pencarian barang setelah mendapatkan hasil yang maksimal.

d. Pengepakan Pallet dalam Kontainer dengan Forklif Menggunakan Metode Algoritma Genetika

Penelitian ini dilakukan oleh Ira mahasiswi jurusan Teknik Informatika Politeknik Elektronika Negeri Surabaya-ITS. Peneliti menggunakan metode algoritma genetika untuk optimasi pengepakan pallet. Tujuan penelitian ini adalah menyelesaikan optimasi pengepakan pallet dalam kontainer menggunakan algoritma genetika sehingga diperoleh sisa ruang yang minimum. Hasil uji coba menunjukkan bahwa algoritma genetika dapat dijadikan metode alternatif untuk menyelesaikan optimasi pengepakan pallet dalam kontainer.

Pada penelitian ini, peneliti menggunakan batasan masalah dimana setiap kontainer hanya boleh mengangkut untuk satu order dengan ukuran kontainer yang sama (semua pallet yang dikirim tidak disusun berdasarkan jarak pengiriman dan jenis kontainer yang mengangkut satu tiap order), tidak menghitung biaya pengiriman, barang yang dikirim sejenis. Peneliti juga berasumsi semua pallet memiliki ketahanan yang kuat, jumlah kontainer tidak terbatas, pallet tidak dapat dijungkirbalikkan, dan pallet yang diatas memiliki luas penampang yang lebih besar daripada yang dibawah-nya maka pallet akan jatuh.

Perbedaan pada penelitian ini terletak di penggunaan metode dan kategori yang digunakan untuk pencarian optimasi. Kategori penelitian

yang saya lakukan adalah menghitung biaya pengiriman, tujuan barang berbeda kota (dengan syarat kota yang dituju berdekatan), barang yang dikirim tidak sejenis, dan kontainer yang digunakan hanya yang berukuran 20 *feet* dengan volume 33 m^3.

e. **Implementasi Algoritma *Greedy* untuk Menyelesaikan Masalah *Knapsack Problem***

Penelitian ini dilakukan oleh Dian Rahmawati dan Ade Candra mahasiswi jurusan Program Studi Ilmu Komputer Universitas Sumatera Utara pada tahun 2013. Peneliti menggunakan metode algoritma *Greedy* untuk menyelesaikan masalah *knapsack problem*. Tujuan penelitian ini adalah menyelesaikan optimisasi dalam *knapsack problem* yang mempunyai kapasitas muat yang terbatas dengan menggunakan algoritma *Greedy*. Hasil uji coba menunjukkan bahwa algoritma *Greedy* dapat dijadikan metode alternative efisien dan optimal untuk menyelesaikan maslah *knapsack problem*.

Perbedaan pada penelitian ini terletak pada metode yang digunakan dan kasus yang di gunakan yaitu permasalahan *knapsack problem*. Dan pada akhirnya nanti akan menghasilkan solusi yang optimal dalam menyelesaikan permasalahan tersebut.

f. Optimasi Proses Muat Barang Dalam Kontainer Studi Kasus PT ANTESS Menggunakan Algoritma Tabu Search

Penelitian ini dilakukan oleh Rizky Kurnia mahasiswa jurusan teknik informatika UIN Maliki Malang pada tahun 2013. Penelitian ini bertujuan

untuk membuat sebuah aplikasi simulasi untuk optimasi proses muat barang dalam kontainer. Dimana kontainer yang digunakan hanya kontainer yang memiliki ukuran 20 *feet* dengan volume 33.1 m^3. Dalam penelitian ini, peneliti menggunakan salah satu metode algoritma heuristic yaitu tabu search, dimana dalam proses muat barang itu sendiri merupakan problem berklarifikasi *NP-hard problem*. Disini peneliti mengkategorikan beberapa hal yang digunakan dalam penentuan optimasi proses muat barang dalam kontainer, yaitu kategori dimensi, berat, tujuan, dan jenis barang. Hasil dari penelitian ini adalah dapat diterapkannya algoritma *tabu search* dalam optimasi proses muat barang dalam kontainer. Dalam penelitian ini tidak dilakukan pencarian barang setelah mendapatkan hasil yang maksimal.

2.3 METODE PENELITIAN

Peneliti membagi pengerjaan penelitian ini menjadi beberapa tahap, antara lain :

1. Studi literatur

   Pada tahap ini dilakukan berbagai pengumpulan data literatur-literatur terkait penelitian ini sebagai berikut:

   - Literatur di dapatkan dari buku, jurnal, atau skripsi terdahulu
   - Literatur berisi informasi tentang pembuatan *Game* pada android dan juga tentang algoritma *Greedy* yang akan diterapkan pada *Game* nantinya

   Untuk selanjutnya akan dilakukan analisis terhadap hasil pengumpulan data dari literature yang telah didapatkan.

2. Perancangan dan desain aplikasi

   Pada tahap ini, perancangan aplikasi terdiri atas perancangan proses-proses utama dan desain aplikasi yang terdiri atas desain menu *Game* dan desain utama dari *Game* itu sendiri.

3. Pembuatan aplikasi

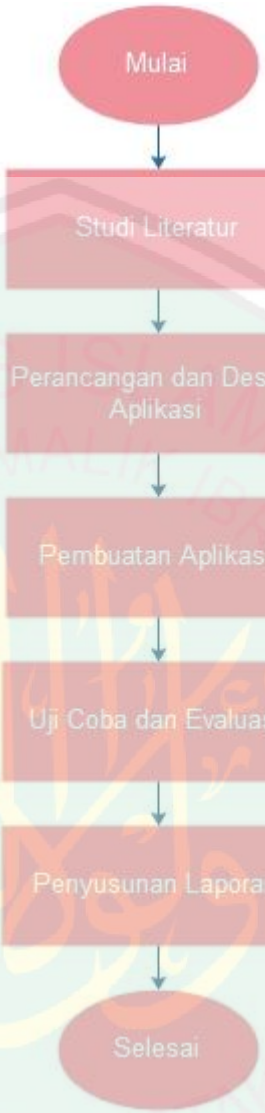
   Pada tahap ini, akan dilakukan pembangunan aplikasi *Game* dengan menuliskan bahasa pemrograman pada compiler sehingga menghasilkan *Game* yang sesuai dengan hasil perancangan.

4. Uji coba dan evaluasi

   Pada tahap ini, akan diketahui hasil dari penataan barang dalam kontainer menggunakan algoritma *Greedy* apakah sudah optimal atau belum. Terjadi kesalahan atau tidak pada penataan barang pada kontainer berdasarkan lokasi tujuan dan berat barang.

5. Penyusunan Laporan

   Dalam pembuatan laporan ini nantinya diharapkan bisa bermanfaat bagi penelitian-penelitian lebih lanjut yang mana penelitian ini berisi hasil dari seluruh dokumentasi pelaksanaan penelitian. Berikut adalah permodelan dari pembagian pengerjaan penelitian.



Gambar 2.2 Alur Penelitian

BAB III

PERANCANGAN DAN *DESIGN GAME*

3.1 Analisis dan Perancangan *Game*

3.1.1 Deskripsi *Game*

Game ini merupakan *Game* ber-genre simulasi. *Game* ini diracang untuk dapat mensimulasikan suatu kondisi permasalahan yang terjadi dalam penataan barang di kontainer. Dengan adanya *Game* serius yang dibuat, *player* dihadapkan pada permasalahan penataan kontainer yang relatif mirip dengan keadaan sebenarnya. Dalam menjalankan permainan, rancangan penataan kontainer yang diberikan oleh *player* dievaluasi oleh *Game*. Evaluasi yang dilakukan oleh *Game* adalah menampilkan skor dari barang yang masuk pada kontainer.

Sasaran pengguna *Game* ini adalah mahasiswa-mahasiswi yang mendalami bidang manajemen logistik dan manajemen supply chain atau juga bisa juga untuk khalayak umum. *Game* ini merupakan *Game single player* dan dibangun 3D. Manfaat yang bisa di ambil dari *Game* ini adalah melatih otak pikiran untuk menyelesaikan suatu permasalahan yang ada di dalam, dengan kata lain bermanfaat untuk melatih kemampuan logika.

3.1.2 *Storyline Game*

Pada proses *Game* serius ini player memilih salah satu level dalam hal ini memilih salah satu dari 3 level, dimana didalam level yang dipilih terdapat beberapa misi yang harus diselesaikan oleh *player* dengan jenis kontainernya

berbeda - beda, ada General Cargo, Thermal Cargo, dan Dry Cargo. Dalam misi ini *player* akan masuk dalam proses utama *Game* ini, yaitu peletakan barang (*box*) dalam ruang 3 dimensi atau kontainer itu sendiri. Informasi data barang yang sudah disediakan akan memudahkan *player* melihat lebih detail tentang barang-barang mana yang sesuai untuk di letakkan pada kontainer tersebut. Detail informasi data barang disini meliputi kode barang, jenis barang, berat barang dan tujuan lokasi. Sedangkan detail informasi data kontainer meliputi kode kontainer, jenis kontainer, berat maksimum kontainer. Dalam proses peletakan barang (*box*) *player* harus memperhatikan beberapa hal yaitu beban maksimal (*pay load*) yang dapat ditampung oleh kontainer, keseimbangan posisi barang agar tidak roboh dan menyebabkan kehancuran barang yaitu di sesuaikan dari jenis barang dan berat, serta tujuan lokasi barang yang dituju. *Player* juga harus memperhatikan batasan waktu dan batasan perubahan dalam penataan barang (*box*) yang sudah ditentukan. *Player* meletakkan barang (*box*) tersebut dengan cara menjalankan barang (*box* atau balok) dengan navigasi button. Navigasi button ini meliputi navigasi arah kanan, kiri, depan, belakang, atas dan bawah yang kemudian diletakkan pada posisi yang menurut player sudah sesuai di ruang 3D (kontainer). *Player* harus bisa membedakan mana barang (*box*) yang harus diletakkan di bagian depan atau bagian belakang sesuai dengan tujuan lokasi yang paling jauh dan terdekat. Player juga harus memperhatikan berat tiap barang (*box*) dalam peletakaannya, yang pasti berat yang ringan tidak akan di letakkan di bawahnya *box* yang memiliki beban berat, karena pada kenyataannya akan menimbulkan kerusakaan pada barang (*box*)

tersebut. Jika semua barang (*box*) sudah dimuat dalam kontainer dan dirasa sudah sesuai dan optimal, maka *player* langsung mengklik *button* selesai, yang nantinya akan di eksekusi menggunakan algoritma greedy dan menampilkan *score*.

3.1.3 Skenario *Game*

Pada proses *Game* serius ini menceritakan tentang *player* yang sedang menyelesaikan permasalah pada setiap level yaitu permasalahan dalam proses penataan barang (*box*) pada ruang 3d (kontainer). Dalam *Game* serius ini terdapat 3 level. Setiap level, memiliki misi yang berbeda – beda yang harus di selesaikan oleh player, diantaranya :

1. Cargo 1 :

   Terdapat 2 misi yang harus di selesaikan, dengan jenis kontainer yang sama tetapi ukurannya berbeda tiap misinya.
2. Cargo 2 :

   Terdapat 3 misi yang harus di selesaikan, dengan jenis dan ukuran kontainer yang berbeda-beda tiap misinya.
3. Cargo 3 :

   Terdapat 3 misi yang harus di selesaikan, dengan jenis kontainer yang berbeda-beda tetapi ukurannya sama tiap misinya.

Game serius ini memiliki 3 parameter yang di gunakan sebagai acuan untuk menyelesaikan tiap-tiap misi dalam level *Game* ini seperti yang terlihat pada Tabel 3.1.

3 parameter itu sebagai berikut :

1. Jenis barang
2. Berat barang
3. Lokasi tujuan.

Tabel 3.1 Level Dan Konstrain pada *Game*

Level	***Konstrain***					
	Barang			**Tujuan Lokasi**	**Kontainer**	
	Jumlah Barang	*Jenis Barang*	*Berat Barang*		*Jenis Kontainer*	*Ukuran*
(Cargo 1)						
Misi 1	25	Berbeda-beda	Berbeda-beda	3	General Cargo	20”
Misi 2	25	Berbeda-beda	Berbeda-beda	5	General Cargo	40”
(Cargo 2)						
Misi 1	42	Berbeda-beda	Berbeda-beda	5	Thermal	20”
Misi 2	25	Berbeda-beda	Berbeda-beda	6	Dry	20”
Misi 3	42	Berbeda-beda	Berbeda-beda	10	Thermal	40”
(Cargo 3)						
Misi 1	42	Berbeda-beda	Berbeda-beda	10	Thermal	40”
Misi 2	25	Berbeda-beda	Berbeda-beda	15	General Cargo	40”

Misi 3	25	Berbeda-beda	Berbeda-beda	8	Dry	40”

3.1.4 *Finite State Machine* (FSM)

3.1.4.1 FSM *Game*

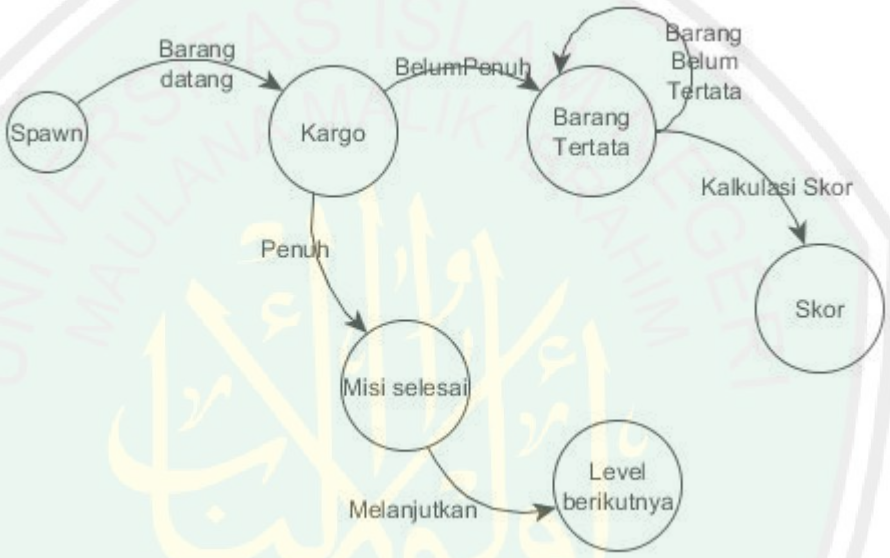


Diagram 3.1 FSM *Game* Level

3.1.5 *Scoring* (Penilaian)

Sistem penilaiannya berasal dari sisa waktu yang digunakan, kesempatan perubahan yang digunakan, dan juga berapa barang yang bisa di letakkan pada ruang 3D sehingga bisa optimal. Optimalnya disini meliputi :

- Jenis kontainer sesuai dengan **jenis barang** yang akan dimuat
- Keseimbangan posisi barang dari **berat barang**
- Peletakan barang dari **lokasi** yang dituju

3.1.6 Konten-Konten Pada *Game*

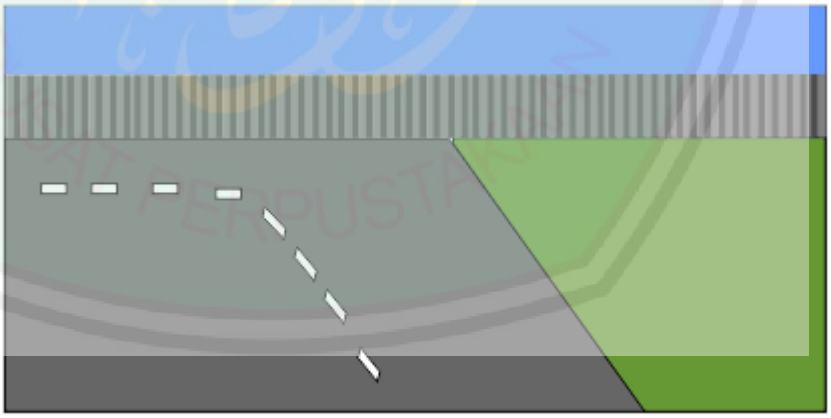
Konten-konten yang terdapat pada *Game* ini adalah :

- Keterangan Barang dan Kontainer (Notif)

Gambar 3.1 Notif Barang

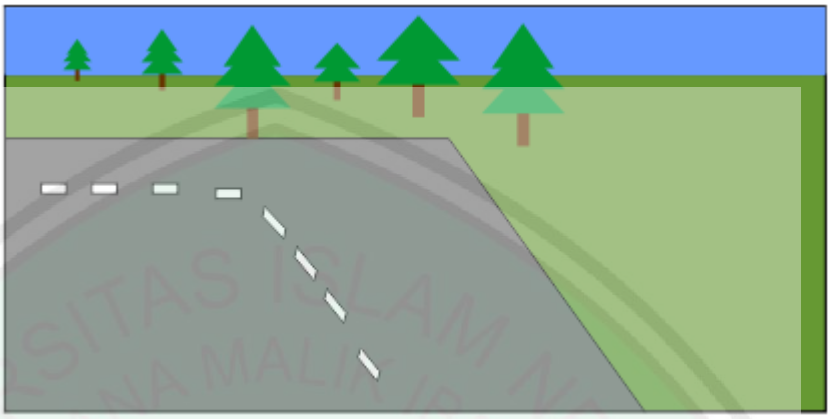
- Latar Belakang (*Background*)

  a. *Background* I



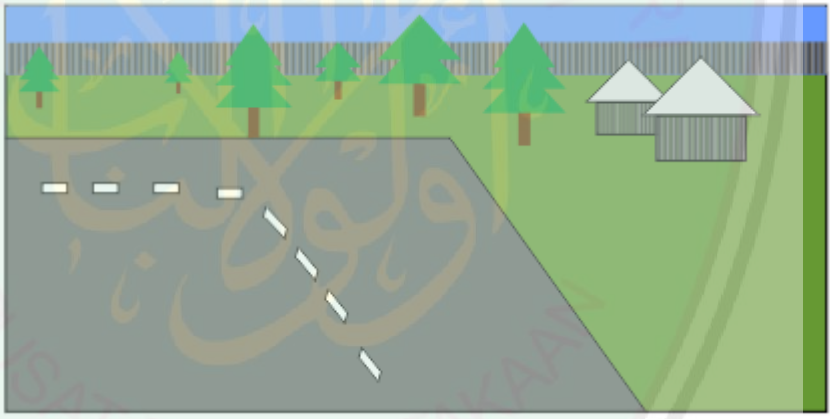
Gambar 3.2 *Background* Cargo 1

b. *Background* II



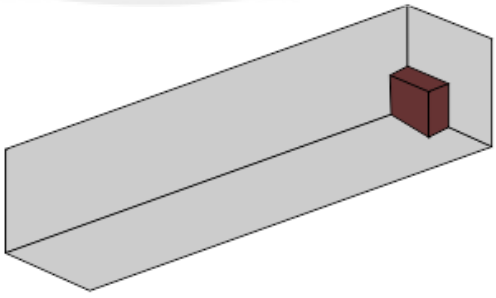
Gambar 3.3 *Background* Cargo 2

c. *Background* III



Gambar 3.4 *Background* Cargo 3

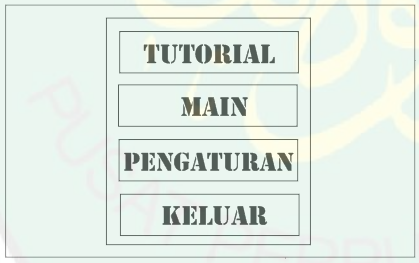
- Ruang 3D Kontainer

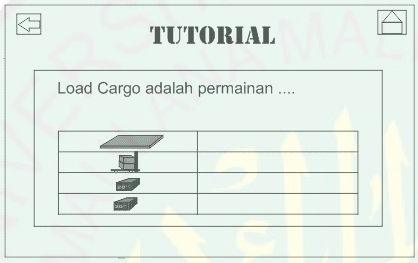
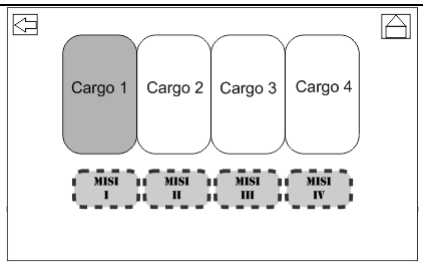


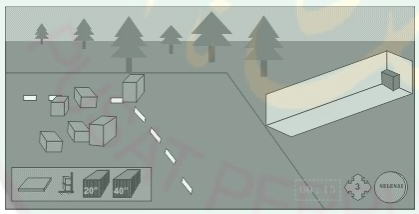
Gambar 3.5 Ruang 3D Kontainer

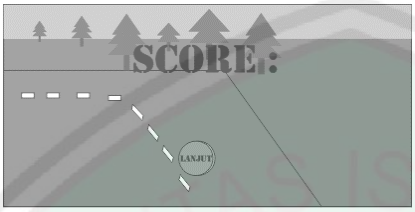
3.1.7 *Story Board Game*

Tabel 3.2 *StoryBoard Game*

No	User Interface	Isi	Keterangan
1.		Tampilan awal / *Splash Screen*	Ini adalah tampilan awal *Game*, dimana terdapat gambar icon *Game*.
2.		Tampilan menu, berisi : - *Button* tutorial - *Button* main - *Button* pengaturan - *Button* keluar	- *Button* tutorial : berfungsi sebagai pentunjuk dalam memainkan *Game*. - *Button* main : berfungsi untuk memulai *Game* yang nantinya user akan menginputkan nama terlebih dahulu. - *Button* pengaturan : berfungsi untuk mengatur suara seperti musik atau sound dari *Game*.

			- *Button* keluar : berfungsi jika *user* ingin keluar dari *Game*.
3.		Tampilan menu tutorial, berisi : - *Button* kembali - *Button home*	Pada tampilan menu tutorial menampilkan petunjuk dalam memainkan *Game* ini serta terdapat penjelasan mengenai fungsi dari item-item yang ada pada *Game* ini.
4.		Tampilan menu main, berisi : - Inputan nama *player* - *Button* OK	Pada tampilan ini *player* diharuskan menginputkan nama terlebih dahulu. *Button* ok akan menujukan pada tampilan selanjutnya.
5.		Tampilan *level* permainan,	cargo 1,2,3 dan 4 merupakan tingkat *level*, dimana setiap cargo terdapat beberapa misi

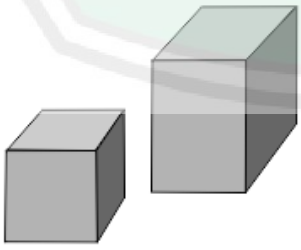
		berisi konten : - cargo 1,2,3 dan 4 - misi 1,2,3 dan 4	yang harus diselesaikan oleh *player*.
6.		Tampilan *Game screen*, berisi : -  item -  waktu -  perubahan -  *Box*/Karton -  Kontainer -  *finish*	Saat *player* memulai *Game*, *player* akan dibantu dengan adanya beberapa item yang disajikan, dan juga batasan waktu akan terus berjalan. *Box* / karton merupakan barang yang harus diletakkan dalam kontainer.

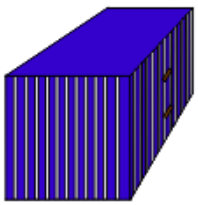
7.		Tampilan nilai, berisi : - *Score* - *Button* lanjut	Pada tampilan ini menampilkan *score* yang di dapat oleh *player*. *Button next* berfungsi untuk menampilkan halaman *level* untuk *player* melanjutkan misi selanjutnya.
8.		Tampilan menu pengaturan, berisi : - Check*box* Musik - Check*box* Suara	- Check*box* musik berfungsi untuk mengaktifkan dan menonaktifkan musik pada saat memulai *Game*. - Check*box* suara juga berfungsi untuk mengaktifkan dan menonaktifkan suara pada *Game*.

9.		Tampilan menu keluar, berisi : - icon  or	Pada tampilan menu keluar, *player* akan di tujukan pertanyaan mengenai " apakah anda yakin untuk keluar dari *Game* ? " dan untuk menjawabnya terdapat icon "iya" untuk keluar dan "tidak" untuk tetap bermain.

3.1.8 *Item Game*

Tabel 3.3 *Item Game*

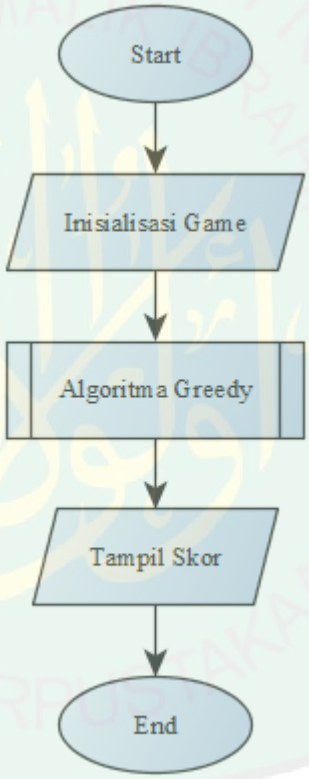
Item	Nama	Fungsi
	*Box* / Karton	*Box* / Karton ini berfungsi sebagai barang yang akan di masukkan dalam kontainer. Barang tersebut dibentuk seperti *box* /karton sesuai kekuatan dan ukuran barang.

	Kontainer 20 *feet*	Kontainer ukuran 20 *feet* berfungsi untuk mengangkut barang-barang sesuai kapasitas yang ditentukan berdasarkan ISO.
	Kontainer 40 *feet*	Kontainer ukuran 40 *feet* berfungsi untuk mengangkut barang-barang sesuai kapasitas yang ditentukan berdasarkan ISO.
	*Thermal*	Kontainer jenis thermal ini berfungsi untuk mengangkut muatan / barang yang membutuhkan pengaturan suhu.
	*Dry*	Kontainer jenis dry ini berfungsi untuk mengangkut muatan curah .
	*General cargo*	Kontainer jenis general cargo berfungsi untuk mengangkut barang-barang pada umumnya.

3.1.9 Perancangan Algoritma *Greedy* pada *Game*

3.1.9.1 Flowchart

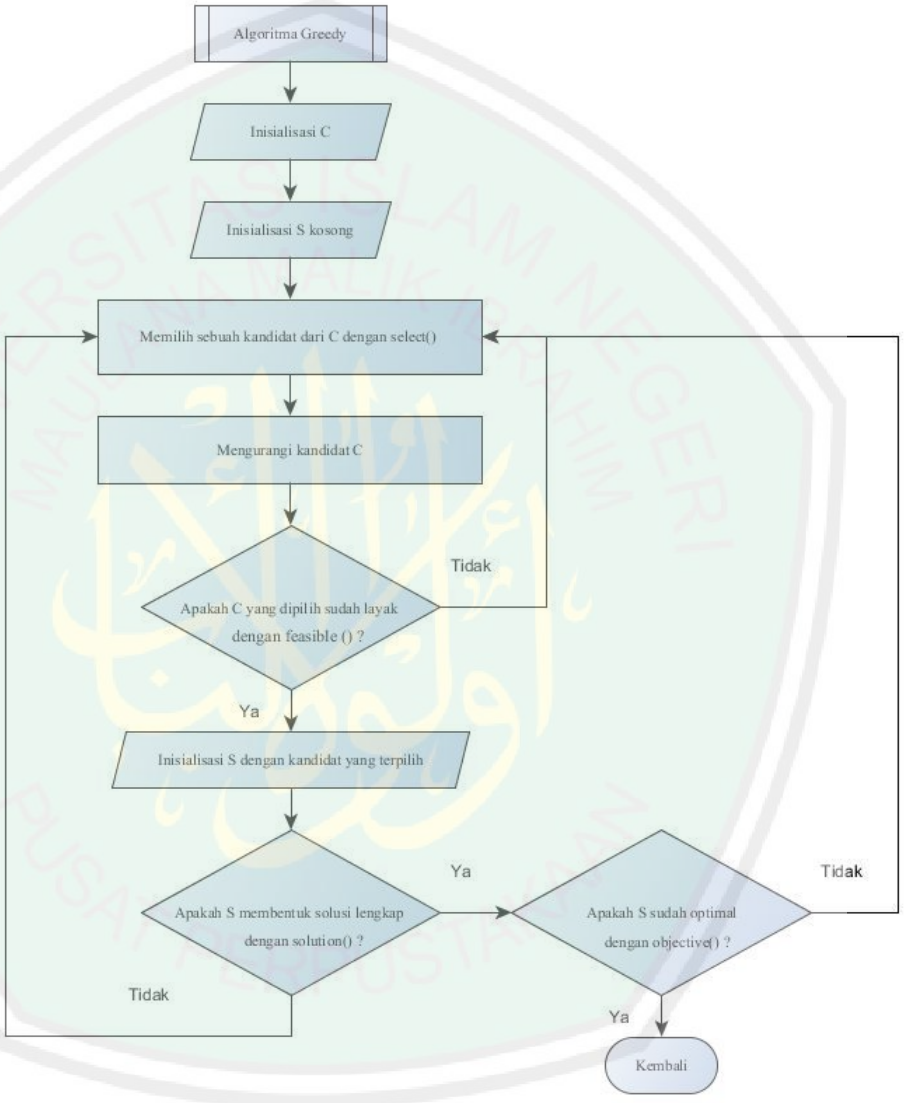
Perancangan algoritma di awali inisialisasi *Game* terlebih dahulu dan kemudian akan di evaluasi dengan menggunakan algoritma *Greedy* dan akan menampilkan skor dari proses penataan *box* tersebut, seperti yang ditunjukkan pada Gambar 3.6



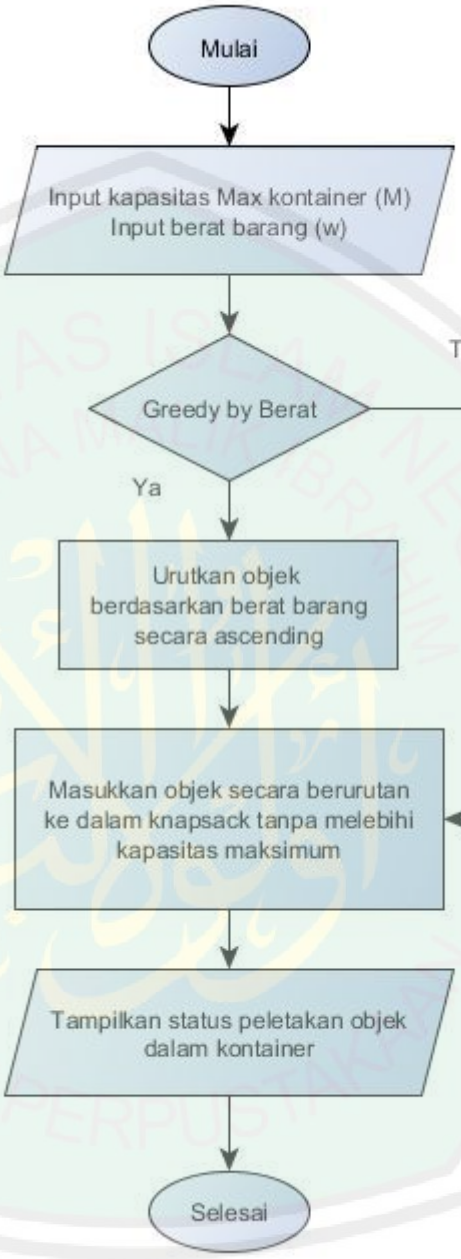
Gambar 3.6 Flowchart *Game*

Pada saat penataan *box* pada ruang 3d (kontainer). Penataan *box* yang sudah di mainkan oleh player akan di evaluasi dengan algortima *Greedy* berdasarkan jenis *box*, berat *box*, dan tujuan lokasi apakah barang yang diletakkan sesuai berdasarkan ketentuan-ketentuan yang

sudah di tentukan. Alur algoritma *Greedy* secara umum di tunjukkan pada Gambar 3.7



Gambar 3.7 Flowchart Umum Algortima *Greedy*



Gambar 3.8 Flowchart Algortima *Greedy* pada *Game*

3.1.9.2 Perhitungan Manual Algoritma *Greedy*

Data pada Tabel 3.1 merupakan data barang yang akan di ambil untuk di inisialisasi dalam proses perhitungan algoritma greedy. Data pada tabel ini berupa nama barang, dimensi barang, berat dan jenis barang, tetapi yang di gunakan untuk inisialisasi dalam listing program hanya jenis barang, berat barang dan dimensi barang. Pada tabel di bawah ini, data barang sudah di golongkan berdasarkan jenis kontainer yang akan di gunakan dalam *Game* serius ini. Dimana setiap jenis kontainer ini memiliki kapasitas maksimum masing-masing. Data barang yang akan di proses terdapat pada Tabel 3.4.

- **General Cargo**

  Lokasi Awal : Malang

  Untuk General Cargo ***20"*** (20 feet) , Kapasitas Muat (M) = 500

  Untuk General Cargo ***40"*** (40 feet) , Kapasitas Muat (M) = 800

Tabel 3.4 Data Barang Pada General Cargo

No	Nama Barang	Dimensi			Berat (w_i)	Tujuan Lokasi (p)	Bobot Tujuan Lokasi (p_i)	Jenis
		Panjang	Lebar	Tinggi				
1	Paket1	4	4	4	20	Surabaya	89	Elektronik
2	Paket2	5	1	5	10	Gresik	107	Elektronik
3	Paket 3	7	2	3	10	Sidoarjo	66	Elektronik
4	Paket4	4	5	8	10	Mojokerto	89	Elektronik
5	Paket5	5	5	5	20	Jombang	119	Elektronik
6	Paket6	3	5	7	10	Bojonegoro	197	Elektronik

7	Paket7	15	4	4	3	Lamongan	134	Furniture
8	Paket8	10	3	5	10	Tuban	192	Furniture
9	Paket9	2	2	2	18	Madiun	178	Makanan
10	Paket10	3	3	3	45	Ngawi	190	Makanan
11	Paket11	3	2	3	90	Magetan	202	Kertas
12	Paket 12	3	3	2	25	Ponorogo	195	Kertas
13	Paket13	2	2	2	40	Pacitan	290	Elektronik
14	Paket 14	4	4	4	20	Kediri	100	Makanan
15	Paket15	4	2	2	40	Nganjuk	26	Makanan
16	Paket 16	6	4	3	10	Tulungagung	111	Pakaian
17	Paket17	5	5	6	10	Blitar	78	Pakaian
18	Paket18	6	6	6	15	Trenggalek	142	Kain
19	Paket19	5	3	7	3	Pasurun	55	Pakaian
20	Paket20	7	2	3	30	Probolinggo	94	Pakaian
21	Paket21	20	2	4	20	Lumajang	117	Furniture
22	Paket22	20	3	5	20	Bondowoso	194	Furniture
23	Paket23	20	3	5	20	Situbondo	189	Furniture
24	Paket24	20	3	2	20	Jember	192	Furniture
25	Paket25	20	3	5	20	Banyuwangi	259	Elektronik
26	Paket26	20	3	5	20	Bangkalan	117	Elektronik
27	Paket27	20	3	2	20	Sampan	179	Elektronik
28	Paket28	10	3	5	10	Pamengkasan	212	Elektronik
29	Paket29	10	5	5	10	Sumenep	264	Furniture
30	Paket30	10	3	5	10	Tuban	192	Furniture

31	Paket31	10	3	5	10	Lumajang	117	Furniture
32	Paket32	10	3	5	10	Bangkalan	117	Furniture
33	Paket33	3	3	5	5	Jember	192	Furniture
34	Paket34	5	2	2	15	Surabaya	89	Makanan
35	Paket35	5	5	3	10	Mojokerto	89	Makanan
36	Paket36	1	1	1	5	Surabaya	89	Furniture
37	Paket37	5	5	5	25	Blitar	78	Makanan

Data dari tabel di atas akan di olah menggunakan algoritma greedy. Terdapat 3 strategi greedy pada perhitungan algoritma greedy, yaitu strategi greedy by berat, strategi greedy by ukuran, dan stategi greedy by lokasi tujuan. Untuk langkah perhitungannya sebagai berikut :

1. **Perhitungan Strategi *Greedy by Weight* (Berat) pada Kontainer 20”**

   Langkah pertama yang harus dilakukan adalah mengurutkan berat barang berdasarkan dari berat teringan sampai terbesar.

   1. Memasukkan data awal berat barang (w_i) : [20 10 10 10 20 10 3 10 18 45 90 25 40 20 40 10 10 15 3 30 20 20 20 20 20 20 20 10 10 10 10 10 5 15 10 5 25]

      Dengan persamaan sebagai berikut :

      $Z = \sum_{i=1}^{n} w_i \, x_i$

   2. Input kapasitas maksimum kontainer 20” (W) : 500 , dimana $M \in N$, $w_i \leq M$, $i \in N$,

3. Melakukan pengurutan barang (*sorting*)

   $W = sort\ (w_i);$

   W = [3 3 5 5 10 10 10 10 10 10 10 10 10 10 10 10 10 15 15 18 20 20 20 20 20 20 20 20 20 20 25 25 30 40 40 45 90]

4. Kemudian di ambil satu – persatu barang yang dapat di tampung oleh kontainer dengan membandingkan berat barang (w_i) dengan bobot maksimal kontainer (M) sampai kontainer penuh atau sudah tidak ada lagi yang bisa di masukkan.

   M = 500

   Perhitungan *Greedy by Weight* (Berat)

   (w_i) <= M dengan batasan $\sum_{i=1}^{n} w_i x_i \leq M$

2. Perhitungan Strategi *Greedy by Location* (Tujuan Lokasi) pada Kontainer 20”

Langkah pertama yang harus dilakukan adalah mengurutkan nilai tujuan lokasi berdasarkan dari nilai terbesar (lokasi terjauh) sampai nilai terkecil (lokasi terdekat).

1. Memasukkan data awal nilai tujuan lokasi (p_i) : [89 107 66 89 119 197 134 192 178 190 202 195 290 100 26 111 78 142 55 94 117 194 189 192 259 117 179 212 264 192 117 117 192 89 89 89 78]

Dengan persamaan sebagai berikut :

$Z = \sum_{i=1}^{n} p_i \, x_i$

2. Melakukan pengurutan tujuan lokasi (*sorting*)

   P = sort (p_i);

   P = [290 264 259 212 202 197 195 194 192 192 192 192 190 189 179 178 142 134 119 117 117 117 117 111 107 100 94 89 89 89 89 89 78 78 66 55 26]

3. Kemudian di ambil satu – persatu barang dengan nilai lokasi terjauh terlebih dahulu sampai yang terdekat, dan juga sekalian membandingkan berat barang apakah sudah sesuai dengan kapasitas maksimum barang.

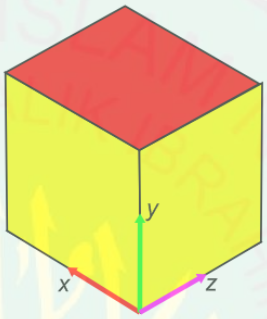
   Perhitungan *Greedy by Location* (Tujuan Lokasi)

   (w_i) <= *M* dengan batasan $\sum_{i=1}^{n} w_i x_i \leq M$ dan $\sum_{i=1}^{n} p_i x_i$

3.1.10 Perhitungan Visualisasi untuk Kontainer dan *Box*

- Jumlah Barang **(n)**
- Dimensi Barang **(Di)**

Panjang, Lebar, Tinggi = (Xi,Yi,Zi) ====> satuan Scale



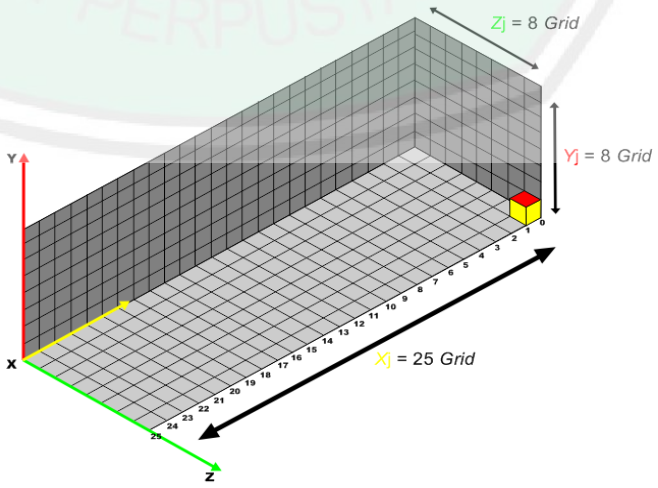
Gambar 3.9 Ukuran *Box* 3D

Di = (2pl +2pt + 2lt)

- Volume Barang **(Vi)**

Vi = Xi*Yi*Zi (*pl*t*)**

- Dimensi Kontainer **(Dj)**



Gambar 3.10 Ukuran Kontainer 20” (Ruang 3D)

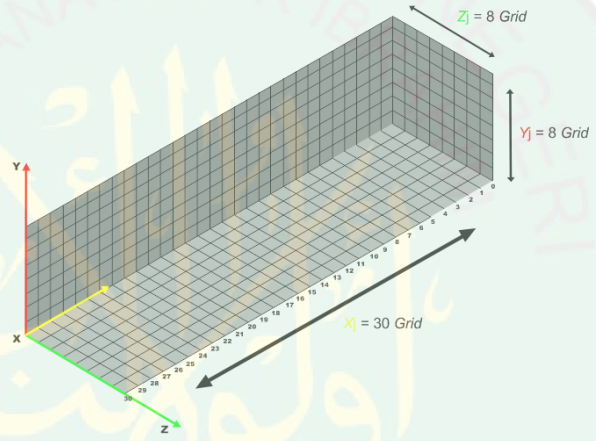
Panjang, Lebar, Tinggi = (Xj,Yj,Zj) ====> satuan Scale

Dimensi Kontainer 20”

***P* = *Xj* = 25 Grid**

***L* = *Yj* = 8 Grid**

***T* = *Zj* = 8 Grid**



Gambar 3.11 Ukuran Kontainer 40” (Ruang 3D)

Dimensi Kontainer 40”

***P* = *Xj* = 30 Grid**

***L* = *Yj* = 8 Grid**

***T* = *Zj* = 8 Grid**

- Volume Kontainer **(Vj)**

Vj = Xj*Yj*Zj (*p*l***t*)**

BAB IV

HASIL DAN PEMBAHASAN

4.1 Implementasi

Pada bab ini membahas tentang implementasi dari perencanaan yang telah dibuat. Serta melakukan pengujian terhadap *Game* untuk mengetahui apakah *Game* tersebut telah berjalan sesuai dengan yang diharapkan.

4.1.1 Kebutuhan Perangkat Keras

Perangkat keras yang diperlukan untuk uji coba perangkat lunak dari aplikasi *Game* ini, sebagai berikut:

Tabel 4.1 Kebutuhan Perangkat Keras

No.	*Perangkat Keras*	*Spesifikasi*
1.	Processor	Quadcore 2.1 Ghz
2.	RAM	4 Gb
3.	VGA	Radeon Dual Graphics (4Gb)
4.	HDD	500 Gb
5.	Monitor	14’
6.	Speaker	On
7.	Mouse & Keyboard	On

4.1.2 Kebutuhan Perangkat Lunak

Perangkat keras yang diperlukan untuk mengimplementasikan perangkat lunak dari aplikasi *Game* ini, sebagai berikut:

Tabel 4.2 Kebutuhan Perangkat Lunak

No.	*Perangkat Lunak*	*Spesifikasi*
1.	Sistem Operasi	Windows 7 32 Bit
2.	*Game Engine*	*Unity3d* 5.2
3.	Konsep desain 2D	Corel Draw X7
4.	*Script Writer*	Mono Develop

4.2 Implementasi Algoritma *Greedy*

Berikut tabel 4.3 akan menjelaskan penggunaan *method* dan fungsi yang ada pada listing program di unity, dimana listing program di unity ini di gunakan untuk control barang pada kontainer dan menghasilkan skor pada tiap barang (*box*).

Tabel 4.3 Variabel, Method pada Listing Program

No	Variabel	Penjelasan	Method	Penjelasan
1.	`public static int gridX = 8; public static int gridY = 8; public static int gridZ = 25;`	- gridX, gridY, gridZ untuk inisialisasi ukuran kontainer 20 feet, dimana	`public void tampilKet () { GameObject jenisbrg = GameObject.Find ("JenisBrg_txt"); Text text`	Method ini di fungsikan untuk menampilkan keterangan barang (*box*),

	public int score = 40; public static int currentScore3d = 0;	P = 25 , gridZ L = 8 , gridY T = 8 , gridX - int score = 40, variabel nilai yang ditentukan untuk tiap barang (*box*). - Int currentScore 3d = 0, variabel untuk menampilkan skor pertama saat mulai bermain.	JnsBrg = jenisbrg .GetComponent<Text> (); textJnsBrg.text = FindObjectOfType<KontenBrang> ().JenisBarang; *Game*Object tujlok = *Game*Object.Find ("LokasiTuj_txt"); Text text lokasi = tujlok.GetComponent<Text> (); textlokasi.text = FindObjectOfType<KontenBrang> ().LokasiTujuan; *Game*Object beratbrg = *Game* Object.Find ("Berat_txt");	yaitu dengan mengambil konten variabel dari barang tersebut yang ada pada kelas KontenBrang yaitu - jenis barang, - tujuan lokasi - dan berat barang yang nantinya akan di tampilkan pada gui *Game*.

			Text textbrat = beratbrg.GetComponent<Text> (); textbrat.text = FindObjectOfType<KontenBrang> ().BeratBarang.ToString(); }	
			public bool CheckAboveGrid3d (Cargo3d *box*){ for(int x = 0; x < gridX; ++x){ foreach(Transform mino in *box*.transform){ Vector3 pos = Roundd3d (mino.position); i	Method untuk mengecek *box* jika sudah memenuhi grid kontainer.

			f(pos.z > gridZ - 1){ return true; } if(pos.y > gridY - 1){ return true; } } } return false; }	
			public void UpdateGrid3d (Cargo3d *box*) { for(int z = 0; z < gridZ; ++z){	Method supaya *box* tidak saling tembus jika bersentuhan.

```
                for(i
nt x = 0; x < gri
dX; ++x){

                    f
or(int y = 0; y <
 gridY; ++y){

    if (grid3d [x,
y,z] != null) {

        if (grid3d
 [x,y,z].parent =
= box.transform)
{

            grid3d
 [x,y,z] = null;

        }

    }
                }
            }
```

			} foreach (Transform mino in *box*.transform) { Vecto r3 pos = Round3d (mino.position); if(po s.z < gridZ){ g rid3d [(int)pos.x , (int)pos.y, (in t)pos.z] = mino; } } }	
			public void Spawn Next*Box*3d(){ if (!mula imain) {	Method untuk menampilkan barang (*box*) secara random pada posisi

			mulai main = true; nextBrang = (*Game*Object)Instantiate (Resources.Load (getRandom*Box*3d (), typeof(*Game*Object)), new Vector3 (5.0f,0.0f, 26.0f), Quaternion.identity); *previewBrang* = (*Game*Object)Instantiate (Resources.Load (getRandom*Box*3d (), typeof(*Game*Object)), *previewBrang*Posisi, Quaternion.identity); *previewBrang*.GetComponent<Cargo3d> ().enabled = false;	titik awal yang sudah di tentukan. Dan juga menampilkan *preview* barang (*box*) yang akan di tampilkan selanjutnya.

			} else { *previewBrang*.transform.localPosition = new Vector3 (5.0f,0.0f, 26.0f); nextBrang = *previewBrang*; nextBrang.GetComponent<Cargo3d> ().enabled = true; *previewBrang* = (*GameObject*)Instantiate (Resources.Load (getRandom*Box*3d (), typeof(*GameObject*)), *previewBrangPosisi*, Quaternion.identity); *previewBrang*.GetCompon	

			ent<Cargo3d> ().enabled = false; } tampilKet (); }	
			public bool cekGrid3d (Vector3 pos){ return ((int)pos.x >= 0 && (int)pos.x < gridX && (int)pos.y >= 0 && (int)pos.z >= 0); } public Vector3 Round3d (Vector3 pos){ return new Vector3 (Mathf.Round(pos.x), Mathf.Round(pos.y), Mathf.Round(pos.z	Method untuk mengatur barang (*box*) yang dating supaya jika di control dalam peletakan barangnya sesuai batas grid yang sudah di tentukan yaitu sesuai ukuran kontainer.

			)); } 	
2.	public bool allowRotation3d = true; public bool limitRotation3d = false;	Variabel di samping merupakan variabel yang di letakkan pada tiap-tiap barang (*box*) dalam bentuk prefabs. Dimana variabel ini nantinya berfungsi untuk mengatur rotasi barang (*box*).	void NavigasiKontrol(){ if(Input.GetKeyDown(KeyCode.RightArrow)){ transform.position += new Vector3 (1,0,0); if (CekFalidPosisiGrid ()) { FindObjectOfType<*Game*3d> ().UpdateGrid3d (this); } else {	Method ini merupakan method navigasi, dimana fungsinya adalah untuk mengotrol barang (*box*). Dalam method ini terdapat 7 navigasi control, di antaranya : ***1. Kontrol Kanan*** ***2. Kontrol Kiri*** ***3. Kontrol Atas*** ***4. Kontrol Bawah***

			t ransform.position += new Vector3 (-1,0,0); }	***5. Kontrol Depan*** ***6. Kontrol Belakang*** ***7. Kontrol Rotasi*** untuk navigasi kontrol kanan, disini berpacu dari vector 3 sumbu x. dimana sumbu x kontrol kanan ini di beri nilai = 1. Dan untuk arah sebaliknya x = -1.
			} else if (Input. GetKeyDown(KeyCod e.LeftArrow)){ trans form.position +=	Navigasi untuk kontrol *box* kekiri dan sebaliknya, sama seperti kontrol *box* ke

			new Vector3 (-1,0,0); if (CekFalidPosisiGrid ()) { FindObjectOfType<*Game*3d> ().UpdateGrid3d (this); } else { transform.position += new Vector3 (1,0,0); }	kanan menggunakan sumbux namun bedanya disini x di beri nilai -1. Dan untuk arah sebaliknya x = 1.
			} else if (Input.GetKeyDown(KeyCode.UpArrow)){	Navigasi untuk kontrol *box* ke depan dan

			transform.position += new Vector3 (0,0,1); if (CekFalidPosisiGrid ()) { FindObjectOfType<*Game*3d> ().UpdateGrid3d (this); } else { transform.position += new Vector3 (0,0,-1); }	sebaliknya, dengan vector 3 sumbu z, dengan nilai z = 1. Dan untuk arah sebaliknya z = -1.
			} else if (Input.GetKeyDown(KeyCode.DownArrow)){	Navigasi untuk kontrol *box* ke belakang dan sebaliknya,

			trans form.position += new Vector3 (0,0, -1); if (C ekFalidPosisiGrid ()) { F indObjectOfType<*G* *ame*3d> ().UpdateG rid3d (this); } els e { t ransform.position += new Vector3 (0,0,1); / / (setelah *box* tertata sebaris dibwah *box* tidak hilang dan score	dengan vector 3 sumbu z, dengan nilai z = -1. Dan untuk arah sebaliknya z = 1.

			tidak muncul dan bisa menampilkan *box* selanjutny) F indObjectOfType<*G ame*3d> ().UpdateS core3d (); / / jika *box* mem enuhi grid maka l angsung *Game* over *part9 i f (FindObjectOfTy pe<*Game*3d> ().Che ckAboveGrid3d (th is)) { FindObjectOfTy pe<*Game*3d> ().*Gam e*Over (); } F indObjectOfType<*G*	

			*ame*3d> ().SpawnNextBox3d (); enabled = false; }	
			} else if (Input.GetKeyDown(KeyCode.CapsLock)){ if(allowRotation3d){ if (limitRotation3d) { if (transform.rotation.eulerAngles.y >= 90) { transform.Rotate (0, -90, 0);	Navigasi untuk kontrol rotasi *box*.

			} else { transform. Rotate (0, 90, 0) ; } } else { transform.Rota te (0,-90,0); } }	
			} else if (Input. GetKeyDown(KeyCod e.A)){ trans form.position += new Vector3 (0,1,	Navigasi untuk kontrol *box* ke atas dan sebaliknya, dengan vector 3 sumbu y,

			0); if (CekFalidPosisiGrid ()) { FindObjectOfType<*Game*3d> ().UpdateGrid3d (this); } else { transform.position += new Vector3 (0,-1,0); }	dengan nilai y = 1. Dan untuk arah sebaliknya y = -1.
			} else if (Input.GetKeyDown(KeyCode.Z)){ trans	Navigasi untuk kontrol *box* ke bawah dan sebaliknya, dengan vector

			form.position += new Vector3 (0,-1,0); if (CekFalidPosisiGrid ()) { FindObjectOfType<*Game*3d> ().UpdateGrid3d (this); } else { transform.position += new Vector3 (0,1,0); } }	3 sumbu y, dengan nilai y = -1. Dan untuk arah sebaliknya y = 1.

Sedangkan pada Tabel 4.4 merupakan penjelasan dari beberapa method untuk simulasi perhitungan algoritma greedy pada Matlab. Dimana dalam proses

perhitungannya sudah di jelaskan pada sub bab sebelumnya yaitu berdasarkan strategi greedy by berat yang juga menampilkan nilai total dari hasil perhitungan tersebut.

Tabel 4.4 Method Algoritma *Greedy* di Matlab

<table>
<tr><th colspan="3">Variabel</th></tr>
<tr><td colspan="3">Dalam method di bawah ini memiliki 5 inputan variabel, yaitu :

location=[] : inputan untuk array nilai jarak lokasi tujuan

weight=[] : inputan untuk array berat barang

M : berat maksimum kontainer

n : panjang dari array location

m : panjang dari array weight</td></tr>
<tr><th colspan="3">Greedy By Weight (Berat)</th></tr>
<tr><th>No</th><th>Method – Method pada Listing Program</th><th>Keterangan</th></tr>
<tr><td>1.</td><td>%clc
clear all
%input data
location=[89 107 66 89 119 197 134 192 178 190 202 195 290 100 26 111 78 142 55 94 117 194 189 192 259 117 179 212 264 192 117 117 192 89 89 89 78];%input ('Masukkan keuntungan = ');
weight=[20 10 10 10 20 10 3 10 18 45 90 25 40 20 40 10</td><td>Berikut adalah inputan data,

location=[89 107 66 89 119 197 134 192 178 190 202 195 290 100 26 111 78 142 55 94 117 194 189 192 259 117 179 212 264 192 117 117 192 89 89 89 78]: inputan data lokasi barang
weight=[20 10 10 10 20 10 3 10 18 45 90 25 40 20 40 10 10 15 3 30 20 20 20 20 20 20 20</td></tr>
</table>

	10 15 3 30 20 20 20 20 20 20 20 20 10 10 10 10 10 5 15 10 5 25];%input ('Masukkan berat = '); M = 500; %input ('Masukkan Kapasitas Kontainer = '); n=length(location); m=length(weight); weigt0=weight; M0=M; n0=n; m0=m; location0=location; maksm = sum(weight); if M>=maksm disp ('Semua barang terangkut'); untung = sum(location); disp (['jumlah keuntungan = ', num2str(untung)]); break end if n~=m disp ('Error, matrik profit dan weight tidak sama'); break end	10 10 10 10 10 5 15 10 5 25] : inputan data berat barang M = kapasitas maksimal muat barang pada kontainer

2.	%disp('pilih 1 : Greedy by Berat'); %disp('pilih 2 : Greedy by TujuanLokasi'); %pilih=input('Masukkan pilihan 1-2 = '); %switch pilih %case 1 %Greedy by Berat %weight yang sudah urut dari kecil ke besar W = sort (weight); %menyesuaikan urutan profit dan urutan barang for i =1:n for j = 1:m a=W(i); b= weight(j); if a==b P(i) = location(j); urutan(i)=j;	Pada fungsi method di samping yaitu berfungsi untuk mengeksekusi data barang yang sudah ada dengan rumus greedy by berat. - *W = sort (weight)* Mengurutkan berat barang dari yang terkecil ke besar - *for I = 1:n* *for j = 1:m* *a=W(i);* *b=weight(j)* *if a==b* *P(i) = location(j);* *urutan(i)=j;* *weight(j)=-b;* *break;* *end* *end* *end*

	weight(j)=-b; break; end end end berat=0; i=0; while berat<=M i=i+1; berat=berat+W(i); end if berat > M berat=berat-W(i); i=i-1; end disp('Greedy by Weight'); disp('Hasil');	berfungsi untuk melakukan perulangan dalam indeks array location dan weight, dimana weight disini sudah terurut dan indeks juga ikut urut. - *If berat > M* *Berat=berat-W(i);* *I=i-1;* *end* berfungsi untuk membandingkan berat yang sudah di urutkan tadi dengan beban maksimum kontainer, apakah berat dari barang ini masih bisa di masukkan atau tidak. - *disp([‘……’)])* berfungsi untuk menampilkan hasil dari sebuah inputan yang sudah di proses. Dalam

	disp(['Data Berat Barang Awal : ', num2str(W)]) disp(['Data Berat Barang Yang Di Muat: ', num2str(W(1:i))]) disp(['Data Barang Indeks ke - :', num2str(urutan(1:i))]) disp(['Total Berat Barang Yang di Angkut: ',num2str(berat)])	perhitungan Greedy by Berat, untuk Outputnya menampilkan 4. Indeks Barang 5. Berat barang 6. Total berat barang - *num2str(urutan(1:i)* berfungsi untuk memanggil indeks array barang dimana indeks ini sudah sesuai dengan berat barang dari yang terkecil ke besar - *num2str(W)* berfungsi menampilkan berat barang yang muat dalam kontainer sesuai urutan barang - *num2str(berat)* berfungsi menampilkan total berat keseluruhan dari berat barang yang sudah di urutkan,

		dan dari situ kita dapat melihat bahwa berat yang dimuat tidak melebihi berat maksimum kontainer 20" yang ada.
	***Greedy By Location* (Tujuan Lokasi)**	
3.	% %case 2 % %Greedy by TujuanLokasi weight=weigt0; M=M0; n=n0; m=m0; location=location0; P=sort(location); %tujuan lokasi yang sudah urut dari jarak terjauh ke dekat P=P(n:-1:1); %menyesuaikan urutan weight dan urutan barang for i = 1:n for j =1:m	Pada fungsi method di samping berfungsi untuk mengeksekusi data barang yang sudah ada dengan greedy by tujuan lokasi. - *P=sort(location);* berfungsi untuk mengurutkan jarak lokasi tujuan dari jarak terjauh ke terdekat, karena dalam proses penataan barang pada kontainer jarak terjauh akan terlebih dahulu di letakkan di bag. belakang, sedangkan semakin kedepan jarak lokasi barang yang dituju semakin dekat.

	a=P(i); b=location(j); if a==b W(i)=weight(j); urutan(i)=j; location(j)=-b; break; end end end disp('Greedy By Tujuan Lokasi'); disp('Hasil') disp(['barang indeks ke - : ',num2str(urutan(1:i))]) disp(['jarak lokasi : ', num2str(P)]) %end	- *P=P(n:-1:1);* berfungsi menyesuaikan urutan berat dan urutan indeks barang - *for i = 1:n* *for j = 1:m* *a=P(1);* *b=location(j);* *if a==b* *W(i)=weight(j);* *urutan(i)=j;* *location(j)=-b;* *break* *end* *end* berfungsi untuk urutan jarak lokasi yang sudah di proses di awal dengan indeks yang

		diurutkan berdasarkan nilai jarak tersebut. - *disp([‘’])* berfungsi menapilkan output dari proses inputan - *num2str(ururtan(1:i))* berfungsi untuk memenggil indeks jarak lokasi dari lokasi yang sudah di ururtkan - *num2str(P)* berfungsi untuk memanggil data jarak lokasi dari yang terjauh ke terdekat.

4.2 Hasil Pengujian Algoritma *Greedy*

- **Pengujian pada Matlab**

KONTAINER 20”

Sesuai listing program yang ada di Matlab di dapatkan pengujian nya pada Tabel 4.5.

Jumlah Objek = 37

Kapasitas Kontainer = 500

Tabel 4.5 Hasil Output Greedy By (*Weight*) Berat

No	Dimensi			Berat (w_i)	Bobot Tujuan Lokasi (p_i)	Status
	Panjang	Lebar	Tinggi			
7	15	4	4	3	134	1
19	5	3	7	3	55	1
33	3	3	5	5	192	1
36	1	1	1	5	89	1
2	5	1	5	10	107	1
3	7	2	3	10	66	1
4	4	5	8	10	89	1
6	3	5	7	10	197	1
8	10	3	5	10	192	1
16	6	4	3	10	111	1
17	5	5	6	10	78	1
28	10	3	5	10	212	1
29	10	5	5	10	264	1
30	10	3	5	10	192	1
31	10	3	5	10	117	1
32	10	3	5	10	117	1
35	5	5	3	10	89	1
18	6	6	6	15	142	1
34	5	2	2	15	89	1
9	2	2	2	18	178	1
1	4	4	4	20	89	1
5	5	5	5	20	119	1
14	4	4	4	20	100	1

21	20	2	4	20	117	1
22	20	3	5	20	194	1
23	20	3	5	20	189	1
24	20	3	2	20	192	1
25	20	3	5	20	259	1
26	20	3	5	20	117	1
27	20	3	2	20	179	1
12	3	3	2	312	195	1
37	5	5	5	25	78	1
20	7	2	3	30	94	1
13	0	0	0	0	0	0
15	0	0	0	0	0	0
10	0	0	0	0	0	0
11	0	0	0	0	0	0
Total				474		

Keterangan :

1 = dipilih

0 = tidak dipilh

Hasil Output Listing Program MATLAB :

```
Greedy by Weight

Hasil

Data Berat Barang Awal : 3   3   5   5  10  10  10  10
10  10  10  10  10  10  10  10  10  15  15  18  20  20
20  20  20  20  20  20  20  20  25  25  30  40  40  45
90

Data Berat Barang Yang Di Muat: 3   3   5   5  10  10
10  10  10  10  10  10  10  10  10  10  10  15  15  18
20  20  20  20  20  20  20  20  20  20  25  25  30
```

```
Data Barang Indeks ke - :7  19  33  36   2   3   4   6
8  16  17  28  29  30  31  32  35  18  34   9   1   5
14  21  22  23  24  25  26  27  12  37  20

Total Berat Barang Yang di Angkut: 474
```

Pada pengujian di atas, maka pada kontainer 20" dengan makimum kapasitas kontainer 500, berhasil membuat barang sebanyak 34 barang dengan berat barang yang sudah di seleksi berdasarkan greedy by Berat. Dan total bobot dari ke 34 barang tersebut adalah 474 kg.

Tabel 4.6 Hasil Output Greedy By *Location* (Tujuan Lokasi)

No	Bobot Tujuan Lokasi (p_i)	Status
13	290	0
29	264	1
25	259	1
28	212	1
11	202	0
6	197	1
12	195	1
22	194	1
8	192	1
24	192	1
30	192	1
33	192	1
10	190	0
23	189	1
27	179	1
9	178	1
18	142	1

7	134	1
5	119	1
21	117	1
26	117	1
31	117	1
32	117	1
16	111	1
2	107	1
14	100	1
20	94	1
1	89	1
4	89	1
34	89	1
35	89	1
36	89	1
17	78	1
37	78	1
3	66	1
19	55	1
15	26	0

Keterangan :

1 = dipilih

0 = tidak dipilh

Hasil Output Listing Program MATLAB :

```
Greedy By Tujuan Lokasi

Hasil

barang indeks ke - : 13  29  25  28  11   6  12  22   8
24  30  33  10  23  27   9  18   7   5  21  26  31  32
16   2  14  20   1   4  34  35  36  17  37   3  19  15

jarak lokasi : 290  264  259  212  202  197  195  194
192  192  192  192  190  189  179  178  142  134  119
```

```
117   117   117   117   111   107   100    94    89    89    89
89    89    78    78    66    55    26

>
```

Sedangkan pada *Greedy By Location* sudah diurutkan barang berdasarkan lokasi terjauh sampai dengan yang terdekat dengan masing-masing memiliki berat dari masing-masing barang yang sudah diseleksi pada *Greedy by Weight*.

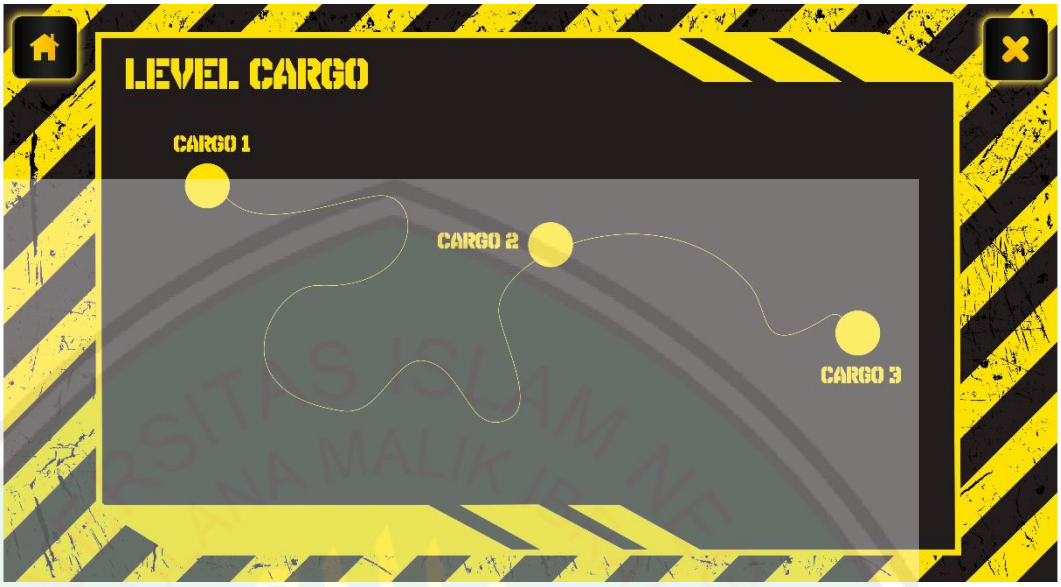
4.3 Implementasi Aplikasi *Game*

Berikut adalah tampilan *Game* yang telah selesai dibuat

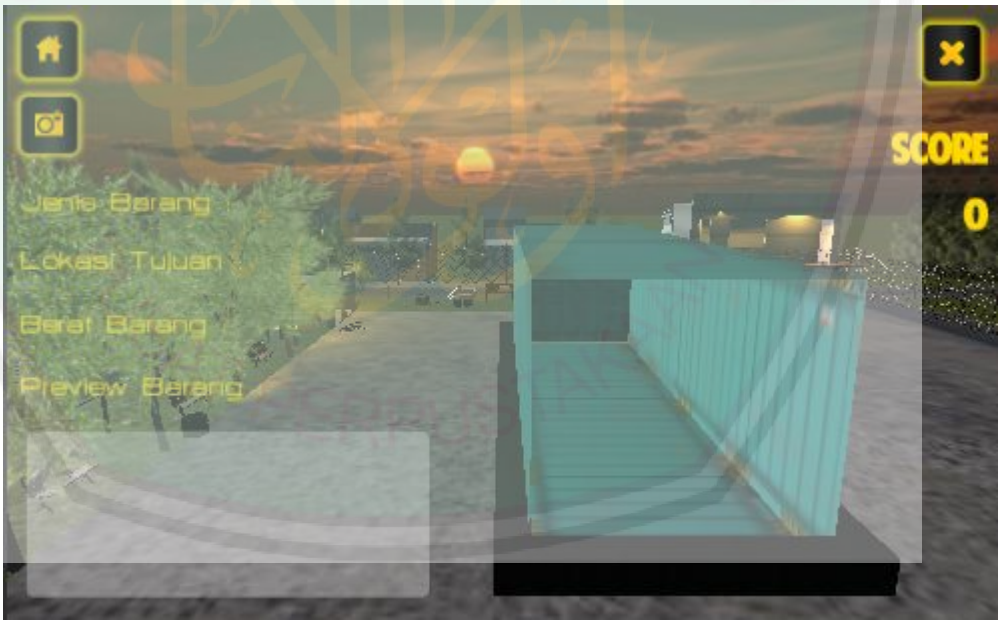
Gambar 4.1 SplashScreen

Gambar 4.2 Tampilan Main Menu

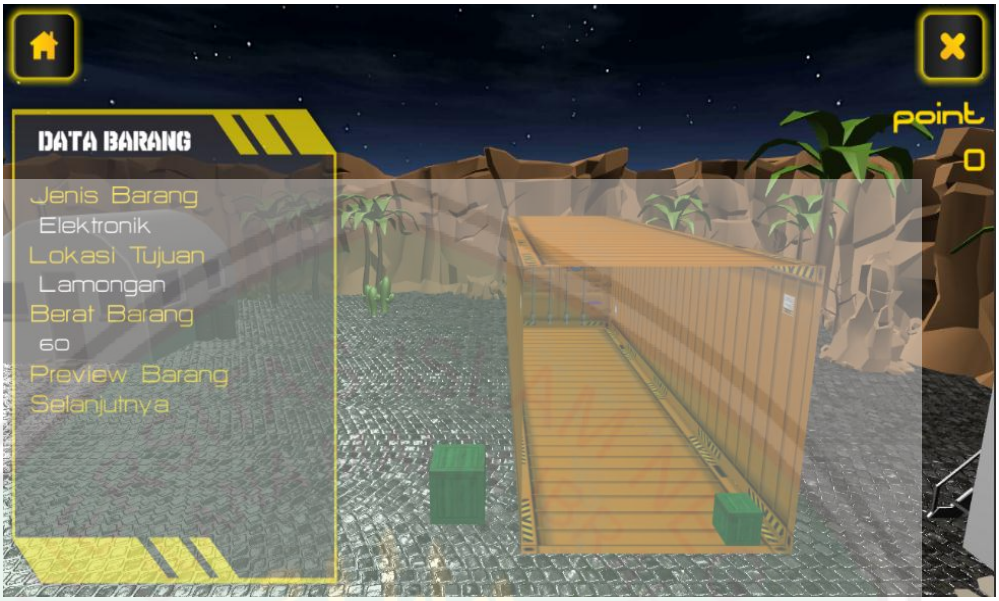
Gambar 4.3 Tampilan Menu Tutorial



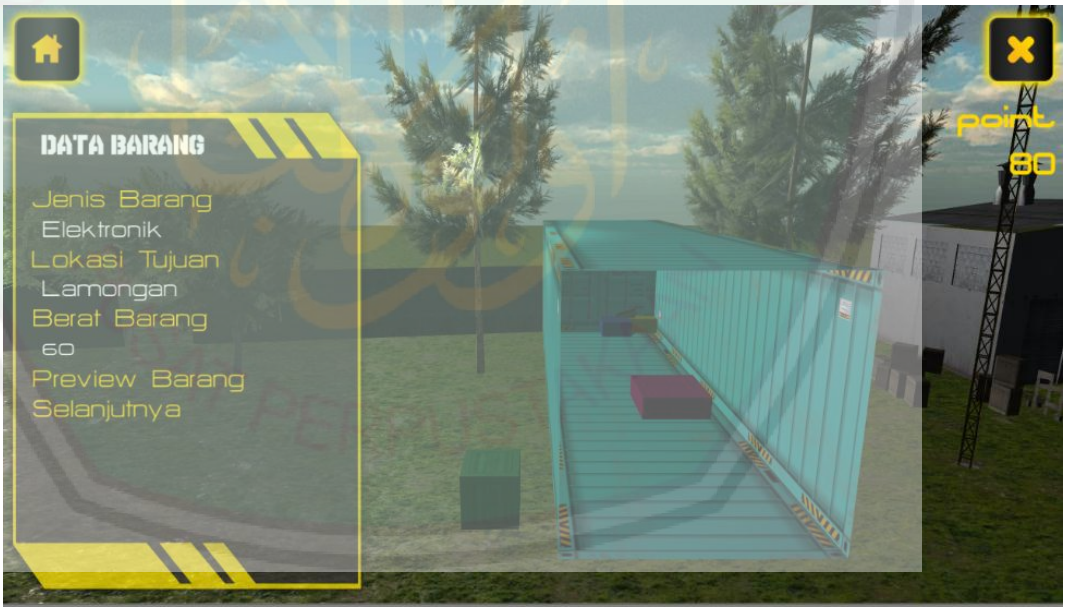
Gambar 4.4 Tampilan Level Cargo



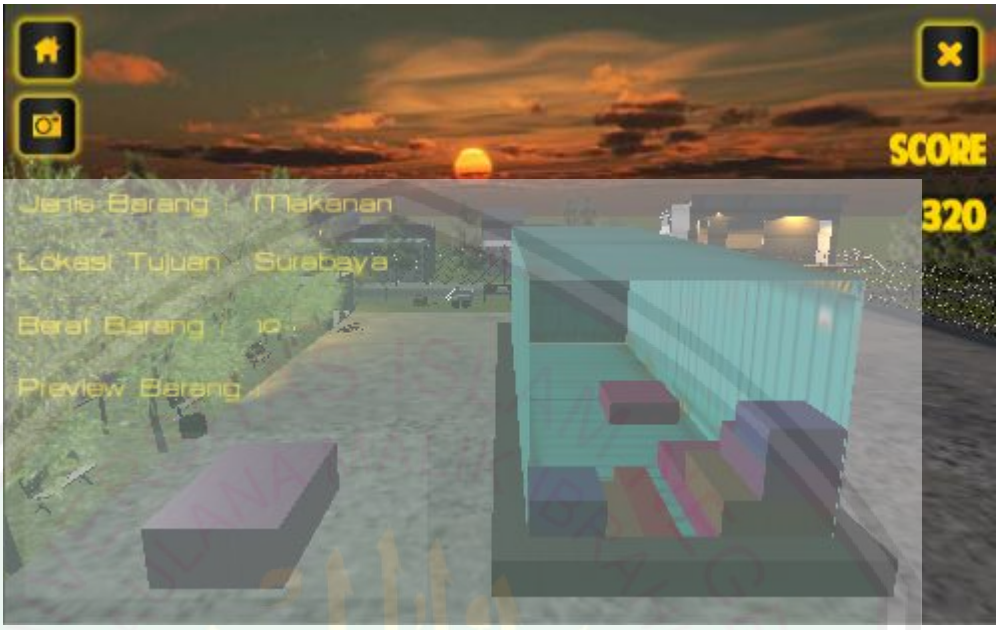
Gambar 4.5 Tampilan Area General cargo



Gambar 4.6 Tampilan Area Dry Cargo



Gambar 4.7 Tampilan Area Thermal Cargo



Gambar 4.8 Tampilan Kontrol *Box* di Kontainer

Merupakan tampilan area bermain *Game* serius Penataan Barang dalam Kontainer pada Area General Cargo. Dimana setiap barang (*box*) yang tampil sudah langsung tampil keterangan barang tersebut. Yang mana pada gambar tersebut *box* warna ungu yang ada dalam kontainer dan masih dalam proses kontol barang, memiliki keterangan yaitu :

Jenis Barang : Makanan

Tujuan Lokasi : Surabaya

Berat Barang : 10

Preview Barang :

Pada *preview* barang, menampilkan barang selanjutnya setelah barang (*box*) warna ungu tersebut sudah ditentukan letak posisinya, maka selanjutnya akan tampil barang (*box*) yang ada pada *preview* barang. Dan setiap barang yang sudah di letakkan pada posisi yang sudah player inginkan akan mendapatkan skor seperti pada tampilan skor pada Gambar 4.5 di atas.

4.4 Integrasi Dalam Islam

Seperti yang di jelaskan pada bab sebelumnya, yaitu dalam hal logistik, ekspedisi ataupun perniagaan butuh ketelitian dan berfikir keras dalam hal penataan barang dan pengirimannya sehingga dapat meningkatkan efisiensi dan laba perusahaan itu sendiri. Tertulis jelas pada surah Al-Isra' Ayat 27 dimana ayat tersebut dengan tegas melarang kita menjadi manusia yang pemboros. Hal ini menunjukkan bahwa dalam kegiataan logistik, ekspedisi atau peniagaan ini di upayakan dalam proses muat barang tidak menjumpai ruang kosong pada kontainer / cargo dimana ruang kosong ini seharusnya bisa di muat barang lagi dan yang pasti tidak membutuhkan biaya yang berlipat.

Kegiatan logistik pastinya mencakup jasa pengiriman barang yang mana barang tersebut menjadi tanggung jawab perusahaan tersebut untuk menjaga agar tidak terjadi kerusakan, dan barang sampai pada penerima dengan kondisi yang baik-baik saja. Di dalam islam, titipan barang di sebut juga wadi'ah. Wadi'ah dapat di artikan pelimpahan kekusaan oleh seorang penitip kepada orang yang menjaga hartanya. Dalam firman Allah SWT QS Al Hujurat Ayat 6 :

يَا أَيُّهَا الَّذِينَ آمَنُوا إِن جَاءكُمْ فَاسِقٌ بِنَبَإٍ فَتَبَيَّنُوا أَن تُصِيبُوا قَوْماً بِجَهَالَةٍ فَتُصْبِحُوا عَلَى مَا فَعَلْتُمْ نَادِمِينَ

"*Hai orang-orang yang beriman, jika dating kepadamu orang fasik membawa suatu berita, maka periksalah dengan teliti agar kamu tidak menimpakan suatu*

musibah kepada suatu kaum tanpa mengetahui keadaannya yang menyebabkan kamu menyesal atas perbuatanmu itu" [Q.S.Al Hujurat : 6]

Yang mana dalam surah ini menekankan kita sebagai manusia harus benar-benar teliti dalam melakukan sesuatu sehingga tidak menimbulkan kerugian atau musibah untuk orang lain yang membuat diri kita menyesal di kemudian hari.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dari implementasi dan pengujian yang dilakukan peneliti, maka dapat ditarik kesimpulan bahwa :

1. *Game* serius ini dapat menjadi sebuah simulasi penataan barang dalam kontainer ini seolah olah sesuai dengan kenyataannya.
2. Algortima *Greedy* adalah salah satu algoritma sederhana yang dapat menyelesaikan permasalahan dan memberikan solusi pada proses pemuatan barang (*box*) pada kontainer (ruang3d).
3. Algoritma *Greedy* ini diterapkan ke dalam AI dengan melakukan evaluasi pada saat player bermain dengan tingkat kesulitan yang berbeda-beda.
4. Peneliti mampu membuat *Game* serius ini dengan prosentase keberhasilan 95%, diluar prosentase tersebut masih terdapat beberapa konten yang tidak berjalan baik.

5.2 Saran

Penulis sadar, dalam pembuatan *Game* serius penataan barang pada kontainer berbasis *mobile* ini masih banyak kekurangan yang nantinya perlu untuk dilakukan pengembangan, oleh karena itu penulis menyarankan beberapa hal diantaranya yaitu :

1. Menambahkan fitur untuk pemilihan jenis kontiner dan barang agar lebih menarik

2. Variabel yang di pakai pada *Game* penataan barang pada kontainer ini berdasarkan berat saja, jadi untuk pengembangan selanjutnya peneliti lain bisa menambahkan variabel lain untuk menambah tingkat untuk pemula atau bahkan tingkat yang lebih sulit.
3. Simulasi dalam *Game* Serius ini perlu dikembangkan lagi supaya terlihat lebih nyata.

DAFTAR PUSTAKA

25 November 2014.http://scratch.mit.edu.

Alan, R., Phil, C., & Peter, B. (2010). *The handbook of logistics and distribution management 4th.* London: KoganPage Ltd.

Bernard, P., & Mark, J. (2009). *Video Game Theory Reader 2.* New York: Routledge.

Bishoff, & Marriot. (1990). A Comparative Evaluation of Heuristics for Container Loading. *Journal of Operational Research*, 267-276.

Brassard, G. (1996). *Fundamentals of algorithmics.* New Jersey: Prentice-Hall.

Ella, E. (2011). *Proses Pemuatan Barang Ke Dalam Container (Stuffing) pada CV. Manggala Java di Klaten.* Surakarta: Perpustakaan Universitas Sebelas Maret.

George, & Robinson. (1980). *A Heuristic for Packing Boxes into a Container.* 147-156: Computers and Operations Research.

Rinaldi, & Christian. (2012). Pengembangan Algoritma Greedy Untuk Optimalisasi Penataan Peti Kemas Pada Kapal Pengangkut. Bandung : Instotut Teknologi Bandung.

Manik, gazali. (2010). Perancangan Program Simulasi Optimasi Penyusunan Barang Dalam Kontainer Menggunakan Algoritma *Greedy*. Jakarta : Universitas Bina Nusantara.

Ira.(2012). Pengepakan Pallet Dalam Kontainer Dengan Forklif Menggunakan Metode Algoritma Genetika. Surabaya : ITS.

Rahmawati, Dian & Ade Candra. (2013). Implementasi Algoritma *Greedy* untuk Menyelesaikan Masalah *Knapsack Problem*. Medan : Universitas Sumatra Utara.

Rizky. (2013). optimasi proses muat barang dalam kontainer studi kasus pt antess menggunakan algoritma tabu search. Malang : UIN Malang.

Gilmore, & Gomory. (1965). *Mutistage cutting stock problems of two and more dimensions.* 94-120: Operations Research.

Hendriyono, R., & Ahmad, R. (2012). *Pengembangan Rancang Bangun Game Edukasi Logistik "STOWAGAME" Mengenai Penataan Kontainer di Bay Kapal.* (ITS) Surabaya: Jurusan Teknik Industri.

Horowitz, E., & Sahni, S. (1974). Computing partitions with applications to the knapsack problem. *Journal of the Association for Computing Machinery 21*, 277–292.

Martello, S., & Toth, P. (1990). *Knapsack Problems: Algorithms and Computer Implementation.* Manag: John Wiley and Sons.

Mayke, S. (2001). *Bermain, Mainan dan Permainan.* Jakarta: Grasindo.

Philips, H. (n.d.). *Java Games Programing.* Ireland: Queen's University.

Pisinger, D. (1995). *Algorithm For Knapsack Problems.* Denmark: Copenhagen.

Pisinger, D. (1998). A Tree Search Heuristic For The Container Loading Problem. *Ricerca Operative*, 31-48.

Pisinger, D. (1998). A Tree Search Heuristic For The Container Loading Problem. pp. 31-48.

Rinaldi, M. (2008). *Diktat Kuliah IF3051 Strategi Algoritma "Algoritma Greedy".* Bandung: Sekolah Teknik Elektro dan Informatika ITB.

Silvano. (1990). *Knapsack problem : Algorithm and Computer Implementation.* ISBN : 0-471-92420.

Sudijono, S., & Sariyanto, S. (2009). *Transportasi Ekspor dan Tatalaksana Kepabeanan.* Surakarta: Fakultas Ekonomi Universitas Sebelas Maret.

V. Springer. 2005. *Knapsack 0-1 Problem*. Yogyakarta: Graha Ilmu.