BAB 2
LANDASAN TEORI

2. 1 Landasan Teori

Landasan teori menjelaskan beberapa teori yang berkaitan dengan permasalahan yang dibahas sebagai dasar pemahaman dalam sebuah sistem serta metode yang dipakai dalam sistem yang akan dibangun.

2.1. 1 Bencana Alam

Bencana alam adalah bencana yang diakibatkan oleh peristiwa atau serangkaian peristiwa yang disebabkan oleh alam antara lain berupa gempa bumi, tsunami, gunung meletus, banjir, kekeringan, angin topan, dan tanah longsor [3].

2.1. 2 Gunung Meletus

Gunung dalam istilah asing disebut "volcano". Istilah ini berasal dari nama kepulauan kecil yang ada di Laut Mediterania yang bernama "Vulcano" [4]. Berabad-abad yang lalu orang-orang yang tinggal di sekitar kepulauan ini percaya bahwa Vulcano adalah cerobong asap dari pandai besi dewa-dewa Romawi yang bernama Vulcan. Mereka mempercayai bahwa lava dan debu panas dari erupsi Vulcano berasal dari tempat kerja Vulcan yang sedang membuat senjata untuk Jupiter (raja para dewa) dan Mars (dewa perang). Ada banyak mitos mengenai keberadaan gunung api, tetapi untuk saat ini diketahui bahwa erupsi gunung api tidak berkaitan dengan mitos-mitos tersebut dan bisa dipelajari serta diinterpretasi oleh ilmu pengetahuan.

Banyak kejadian tragis menimpa manusia akibat erupsi . Gunung Vesuvius di Italia mengubur kota Pompeii dan Herculaneum beserta isinya pada tahun 79 M. Erupsi Gunung Galunggung pada tahun 1982 menyebabkan evakuasi 35,000 penduduk yang tinggal di sekitar gunung tersebut, sekitar 94 ribu hektar lahan pertanian terkena dampak letusan, tiga juta penduduk yang tinggal dalam radius 100-150 km merasakan debu vulkanik, dan total kerugian material diperkirakan sekitar 80 milyar rupiah. Erupsi

Gunung Papandayan pada tahun 2002 menimbulkan awan tebal dari debu vulkanik setinggi 500 meter yang menyebabkan warga yang tinggal di sekitarnya mengungsi.

Begitu dahsyatnya akibat yang ditimbulkan oleh suatu erupsi gunung api, tetapi ironisnya, tanah vulkanik hasil aktifitas gunung api tetap mengundang manusia untuk hidup di sekitarnya. Bahkan seiring dengan kemajuan teknologi dan meningkatnya kebutuhan energi, gunung api dipelajari sedemikian untuk keperluan energi geotermal, energi ramah lingkungan yang dapat terbarukan (*renewable resources*) sebagai energi alternatif.

Lingkup studi mengenai gunung api meliputi petrologi, mitigasi dan evaluasi bencana, survei pemetaan geologi, pemantauan/mitigasi erupsi, tata guna lahan, pertanian, dan eksplorasi sumber daya alam termasuk energi geotermal. Dalam mempelajari gunung api ada beberapa aspek keilmuan penting yang harus dipelajari secara terpadu yaitu: pembentukan magma, akumulasi dan diferensiasi dalam dapur magma, erupsi, metoda analisa statistik, proses fisika dan kimia, dan hidrovolkanisme.

Tujuan paling akhir dalam mempelajari gunung api adalah mampu mengetahui dan merencanakan penggunaan lahan di sekitar daerah gunung api dengan sebaik-baiknya serta kemungkinannya untuk eksplorasi geotermal.

Gunung api bisa merupakan rangkaian pegunungan, tetapi sangat berbeda dengan gunung lainnya. Gunung api tidak dibentuk oleh perlipatan, erosi ataupun pengangkatan, tetapi membentuk tubuhnya sendiri oleh adanya pengumpulan bahan erupsinya, seperti lava, jatuhan dan aliran piroklastik. Gunung api aktif dan dorman (mati) terletak di sepanjang jalur yang bersamaan dengan daerah gempa bumi.

Kegiatan merupakan suatu proses yang tidak random, sehingga dapat diamati/dipantau dengan metode pengamatan geologi/geokimia ataupun dengan menggunakan peralatan geofisika dan geodesi.

Metode penyelidikan lapangan daerah gunung api, meliputi persiapan kerja lapangan, studi literatur, peta topografi, citra satelit, foto udara, peta kepemilikan lahan/peta tata guna lahan. Analisa vulkanostratigrafi sangat penting sebagai dasar untuk membantu studi lainnya seperti petrologi, geokimia, keadaan suhu, dan kerangka struktur geologi. Stratigrafi di daerah gunung api dapat dibuat

atas dasar penelitian sebelumnya dari hasil analisa citra satelit, foto udara, maupun peta topografi.

2.1.2. 1 Gunung Api di Indonesia

Indonesia memiliki gunung api yang terbanyak di dunia yaitu 129 gunung api aktif atau sekitar 15% dari seluruh gunung api yang ada di bumi [5]. Meskipun demikian, sangat sedikit sekali orang Indonesia yang ingin mendalami ilmu vulkanologi.

Penyebaran gunung api di Indonesia dapat dikelompokan sebagai berikut:

1. Kelompok sunda, mulai dari pulau Weh, Sumatra, Jawa, Bali, Sumbawa, Flores dan beberapa pulau di sebelah utara dan timurnya.
2. Kelompok Banda, teletak di beberapa pulau di Laut Banda bagian tengah dan selatan.
3. Kelompok Sulawesi-Sangihe tersebar mulai dari Teluk Tomini, Sulawesi Utara sampai dengan bagian utara Kepulauan Sangihe.
4. Kelompok Halmahera, tersebar di beberapa pulau di Halmahera bagian barat dan utara.

Di Indonesia umumnya gunung api bertipe strato dengan komposisi batuan intermedier, terdapat kawah atau kubah lava dengan ketinggian antara 2000-3000 m di atas permukaan laut. Daerah di sekitar puncak sejauh 5-15 km adalah daerah utama yang terkena pengaruh bencana yang mematikan. Daerah di sekitar gunung api, biasanya merupakan daerah yang sangat subur, sehingga banyak penduduk yang bermukim di sekitarnya.

Tabel 2.1 dan Tabel 2.2 memperlihatkan informasi mengenai penyebaran gunung api di Indonesia yang diklasifikasikan berdasarkan sejarah erupsinya.

Tabel 2. 1 Informasi Umum Gunung Api Indonesia

Tipe Gunung api			Daerah bahaya (km^2)	Penduduk yg terancam (jiwa)	Penyebaran gunung api aktip di Indonesia (Tipe A,B dan C)				
A	B	C			Sumatra	Jawa	Bali & NTT	Maluku	Sulawesi
76	29	24	17,000	4,000,000	30	35	30	16	18

Tabel 2. 2 Gunung Api Aktif Indonesia

S U M A T R A		
Tipe A	**Tipe B**	**Tipe C**
1. Silawih Agam 2. Peuetsagoe 3. Bur Ni Telong 4. Sorik Marapi 5. Marapi 6. Tandikat 7. Talang 8. Kerinci 9. Sumbing 10. Kaba 11. Dempo 12. Krakatau	1. Bur Ni Geureudong 2. Sibayak 3. Sinabung 4. Pusuk Buhit 5. Bual-Buali 6. Talakmau 7. Kunyit 8. Blerang Beriti 9. Bukit Daun 10. Lumut Balai 11. Sekicau Belirang 12. Rajabasa	1. Pulu Weh 2. Gyolesten 3. Helatoba Tarutung 4. Marga Bayur 5. Pematang Bata 6. Hulubelu

J A W A		
Tipe A	**Tipe B**	**Tipe C**
1. Salak 2. Gede 3. Tangkubanperahu 4. Papandayan 5. Guntur 6. Galunggung 7. Cereme 8. Slamet 9. Butak Petarangan 10. Dieng 11. Sundoro 12. Sumbing 13. Merbabu 14. Merapi 15. Kelud 16. Arjuno Welirang 17. Semeru 18. Bromo 19. Lamongan 20. Raung 21. Ijen	1. Pulosari 2. Karang 3. Patuha 4. Wayang Windu 5. Talagabodas 6. Ungaran 7. Lawu 8. Wilis 9. Iyang Argopuro	1. Kiaraberes Gagak 2. Perbakti 3. Kawah Manuk 4. Kawah Kamojang 5. Kawah Karaha

NUSA TENGGARA TIMUR		
Tipe A	**Tipe B**	**Tipe C**

1. Inie Lika 2. Inie Rie 3. Ebuloho 4. Iya 5. Kelimutu 6. Rokatenda 7. Egon 8. Lewotobi Laki-laki 9. Lewotobi Perempuan 10. Ili Boleng 11. Lereboleng 12. Lewotolo 13. Ili Werung 14. Batutara 15. Sirung 16. Hobal	1. Ili Muda 2. Ili Labalekan 3. Jersey	1. Waisano 2. Pocoleok 3. Sokoria 4. Ndetu Soko 5. Riangkotang 6. Mataloko/Bajawa

NUSA TENGGARA BARAT		
Tipe A	**Tipe B**	**Tipe C**
1. Rinjani 2. Sangeangapi 3. Tambora		
B A L I		
Tipe A	**Tipe B**	**Tipe C**
1. Agung		

2. Batur		

SULAWESI - SANGIHE		
Tipe A	**Tipe B**	**Tipe C**
1. Una-Una 2. Ambang 3. Soputan 4. Lokon 5. Mahawu 6. Tongkoko 7. Ruang 8. Karangetang 9. Banua Wuhu 10. Awu 11. Submarin 1922	1. Sempu 2. Klabat	1. Batukolok 2. Tempang 3. Tampusu 4. Lahendong 5. Sarongsong

M A L U K U		
Tipe A	**Tipe B**	**Tipe C**
1. Emperor of China 2. Nieuwerkerk 3. Gunung api 4. Wurlali (P. Damar) 5. Serawerra (P. Teon) 6. Laworkawra(P. Nila) 7. Legatala (P.Serua) 8. Banda Api 9. Dukono 10. Malupang Walirang	1. Manuk 2. Todoko 3. Ibu	

11. Gamkonora 12. Gamalama 13. Kie Besi(Makian)		

Di Indonesia terdapat 129 gunung api aktif yang 10-15 diantaranya dikategorikan sebagai gunung api kritis atau sangat mungkin untuk meletus. Bentuk ancaman dari bencana akibat meletusnya gunung api adalah korban jiwa dan kerusakan pemukiman/harta/benda, akibat aliran lava, lemparan batu, abu, awan panas, gas-gas beracun, dan lain lain. Frekuensi letusan gunung api di Indonesia tercatat antara 3 sampai 5 kali pertahun.

2. 2 Penanggulangan Bencana

Letusan gunung api memberikan catatan sejarah tersendiri terhadap kebencanaan di Indonesia. Beberapa letusan dahsyat tidak hanya berdampak di wilayah Indonesia tetapi juga wilayah-wilah di benua lain. Letusan gunungapi dahsyat antara lain mengguncang Gunung Toba, Tambora (1815), dan Krakatau (1883). Indonesia yang dilewati oleh barisan gunungapi atau lebih dikenal dengan cincin api memiliki 29 gunungapi aktif [6].

Menghadapi ancaman letusan gunungapi, Anda memiliki lebih banyak waktu karena aktivitas letusan mengalami proses yang dapat dideteksi oleh para ahli dan pihak berwenang. Masyarakat yang hidup di sekitar gunungapi aktif mungkin akan melihat pergerakan binatang-binatang yang menjauh karena suhu yang memanas, getaran gempa, maupun bau sulfur.

Apa yang dilakukan sebelum terjadi letusan gunung api :

1. Memperhatikan arahan Pusat Vulkanologi dan Mitigasi Bencana Geologi (PVMBG) terkait dengan perkembangan aktivitas gunung api.
2. Persiapkan masker dan kacamata pelindung untuk mengantisipasi debu vulkanik.
3. Mengetahui jalur evakuasi dan shelter yang telah disiapkan oleh pihak berwenang.

4. Mempersiapkan skenario evakuasi lain apabila dampak letusan meluas di luar prediksi ahli.
5. Persiapkan dukungan logistik:
    a. Makanan siap saji dan minuman
    b. Lampu senter dan baterai cadangan
    c. Uang tunai secukupnya
    d. Obat-obatan khusus sesuai pemakai

Apa yang dilakukan pada saat terjadi letusan gunung api :

1. Pastikan anda sudah berada di shelter atau tempat lain yang aman dari dampak letusan.
2. Gunakan masker dan kacamata pelindung
3. Selalu memperhatikan arahan dari pihak berwenang selama berada di shelter.

Apa yang dilakukan sesudah terjadi letusan gunung api :

1. Apabila Anda dan keluarga harus tinggal lebih lama di shelter, pastikan kebutuhan dasar terpenuhi dan pendampingan khusus bagi anak-anak dan remaja diberikan. Dukungan orangtua yang bekerjasama dengan organisasi kemanusiaan dalam pendampingan anak-anak dan remaja sangat penting untuk mengurangi stres atau ketertekanan selama di shelter.
2. Tetap gunakan master dan kacamata pelindung ketika berada di wilayah yang terdampak abu vulkanik.
3. Memperhatikan perkembangan informasi dari pihak berwenang melalui radio atau pengumuman dari pihak berwenang.
4. Waspada terhadap kemungkinan bahaya kedua atau secondary hazard berupa banjir lahar dingin. Bencana ini dipicu oleh curah hujan tinggi dan menghanyutkan material vulkanik maupun reruntuhan kayu atau apapun sepanjang sungai dari hilir ke hulu. Perhatikan bentangan kiri dan kanan dari titik sungai mengantisipasi luapan banjir lahar dingin.

2. 3 Multimedia

2.3. 1 Pengertian Multimedia

Multimedia adalah suatu kombinasi dari berbagai medium, dimana kombinasi tersebut dapat digunakan untuk tujuan pembelajaran. Multimedia juga dapat diartikan sebagai gabungan dari teks, suara, gambar, animasi dan video dengan alat bantu (*tool*) dan koneksi (*link*) sehingga pengguna dapat bernavigasi, berinteraksi, berkarya dan berkomunikasi [7].

Multimedia berasal dari kata multi dan media. Multi berarti banyak, dan media berarti tempat, sarana atau alat yang digunakan untuk menyimpan informasi. Jadi berdasarkan kata‘multimedia’ dapat diasumsikan sebagai wadah atau penyatuan beberapa media yang kemudian didefinisikan sebagai elemen-elemen pembentukan multimedia. Elemen-elemen tersebut berupa teks, gambar, suara, animasi, dan video. Multimedia merupakan suatu konsep dan teknologi baru bidang bidang teknologi informasi, dimana informasi dalam bentuk teks, gambar, suara, animasi, dan video disatukan dalam komputer untuk disimpan, diproses, dan disajikan baik secara linier maupun interaktif.

Menurut Arsyad multimedia adalah berbagai macam kombinasi grafik, teks, audio, suara, dan animasi. Penggabungan ini merupakan suatu kesatuan yang secara bersama-sama menampilkan informasi, pesan, atau isi pembelajaran [8]. Sedangkan Gayeskimengartikan multimedia ialah suatu sistem hubungan komunikasi interaktif melalui komputer yang mampu menciptakan, menyimpan, memindahkan, dan mencapai kembali data dan maklumat dalam bentuk teks, grafik, animasi, dan sistem *audio* [9].

Penyajian dengan menggabungkan seluruh elemen multimedia tersebut menjadikan informasi dalam bentuk multimedia yang dapat diterima oleh indera penglihatan dan pendengaran, lebih mendekati bentuk aslinya dalam dunia sebenarnya. Multimedia interaktif adalah bila suatu aplikasi terdapat seluruh elemen multimedia yang ada dan pemakai (*user*) diberi kebebasan/kemampuan untuk mengontrol dan menghidupkan elemen-elemen tersebut.

2.3. 2 Karakteristik Media dalam Multimedia

Berikut ini terdapat beberapa karakteristik meid adalam multimedia, di antaranya adalah sebagai berikut :

1. Text

Text mungkin bukan merupakan media paling kuno yang digunakan oleh manusia dalam menyampaikan informasi, suara (*sound*) adalah media yang lebih dahulu digunakan di dalam menyampaikan informasi. Para filusuf Yunani, bahkan para Nabi menggunakan suara sebagai media utama untuk menyebarkan ajarannya. Namun di dalam penggunaannya di dalam komputer text adalah media yang paling awal dan juga paling sederhana. Di awal-awal perkembangan teknologi komputer text adalah media yang dominan (bahkan satu-satunya).

Hal yang sama juga berlaku di dalam perkembangan internet. Ketika internet masih bernama ARPANET di awal tahun 1970 an text merupakan satunya-satunya media. Kini ketika perkembangan teknologi komputer telah demikian maju text bukan lagi media yang dominan.

2. Audio

Socrates pernah berujar bahwa suara adalah imitasi terbaik bagi pikiran maka suara adalah media terbaik untuk menyampaikan informasi. Bagi Socrates text adalah imitasi dari suara, dengan demikian sebagai penyampai pikiran text bukanlah media yang ideal karena ia hanyalah imitasi dari suatu imitasi. Pendapat Socrates mungkin ada benarnya karena suara adalah media yang secara natural telah dimiliki oleh manusia sehingga suara adalah media yang paling alami.

3. Graphics

“A picture is worth a thousand words”, Peribahasa ini menunjukkan bahwa penggunaan gambar di dalam pembelajaran mampu menjelaskan banyak hal bila dibandingkan dengan media text. Berikut ini merupakan kelebihan dari media gambar diantaranya adalah :

1) Lebih mudah mengidentifikasi objek-objek.
2) Lebih mudah dalam mengklasifikasi objek.
3) Mampu menunjukkan hubungan spatial dari suatu objek.
4) Membantu menjelaskan konsep abstrak menjadi konkret.

4. Animasi

Animasi sendiri berasal dari bahasa latin yaitu "anima" yang berarti jiwa, hidup, semangat. Sedangkan karakter adalah orang, hewan maupun objek nyata lainnya yang dituangkan dalam bentuk gambar 2D maupun 3D, sehingga karakter animasi dapat diartikan sebagai gambar yang memuat objek yang seolah-olah hidup, disebabkan oleh kumpulan gambar itu berubah beraturan dan bergantian ditampilkan. Objek dalam gambar bisa berupa tulisan, bentuk benda, warna dan spesial efek. Berdasarkan teknik pembuatannya animasi dibedakan menjadi sepuluh jenis yaitu :

1) Animasi *Cel*

   Animasi *cel* berasal dari kata "*celluloid*", yaitu bahan dasar dalam pembuatan animasi jenis ini ketika tahun-tahun awal adanya animasi. Animasi *cel* merupakan lembaran-lembaran yang membentuk animasi tunggal, masing-masing *cel* merupakan bagian yang terpisah sebagai objek animasi. misalnya ada tiga buah animasi *cel*, *cel* pertama berisi satu animasi karakter, *cel* kedua berisi animasi karakter lain, dan *cel* terakhir berisi latar animasi. Ketiga animasi *cel* ini akan disusun berjajar, sehingga ketika dijalankan animasinya secara bersamaan, terlihat seperti satu kesatuan. Contoh animasi jenis ini adalah film kartun seperti Tom and Jerry, Mickey Mouse dan Detectif Conan.

2) Animasi *Frame*

   Animasi *frame* merupakan animasi yang paling sederhana, dimana animasinya didapatkan dari rangkaian gambar yang bergantian ditunjukan, pergantian gambar ini diukur dalam satuan fps (*frame per second*). Contoh animasi ini adalah ketika kita membuat rangkaian gambar yang berbeda pada tepian sebuah buku, kemudian kita buka

buku tersebut sedemikian rupa menggunakan jempol, maka gambar akan terlihat bergerak. Dalam Macromedia Flash, animasi ini dibuat dengan teknik animasi *keyframe*, teknik ini sering digunakan untuk mendapatkan animasi objek yang tidak bisa didapatkan dengan teknik animasi *tween*, teknik animasi *path* dan teknik animasi *script*.

3) Animasi *Sprite*

Pada animasi ini setiap objek bergerak secara mandiri dengan latar belakang yang diam, setiap objek animasi disebut "sprite". Tidak seperti animasi *cel* dan animasi *frame*, setiap objek dalam animasi *sprite* bergerak tidak dalam waktu bersamaan, memiliki besar fps yang berbeda dan pengeditan hanya dapat dilakukan pada masing-masing objek *sprite*. Contoh animasi ini adalah animasi rotasi planet, burung terbang dan bola yang memantul. Penggunaan animasi jenis ini sering digunakan dalam Macromedia Director.

4) Animasi *Path*

Animasi *path* adalah animasi dari objek yang gerakannya mengikuti garis lintasan yang sudah ditentukan. Contoh animasi jenis ini adalah animasi kereta api yang bergerak mengikuti lintasan rel. Biasanya dalam animasi *path* diberi perulangan animasi, sehingga animasi terus berulang hingga mencapai kondisi tertentu. Dalam Macromedia Flash, animasi jenis ini didapatkan dengan teknik animasi *path*, teknik ini menggunakan *layer* tersendiri yang didefinisikan sebagai lintasan gerakan objek.

5) Animasi *Spline*

Pada animasi *spline*, animasi dari objek bergerak mengikuti garis lintasan yang berbentuk kurva, kurva ini didapatkan dari representasi perhitungan matematis. Hasil gerakan animasi ini lebih halus dibandingkan dengan animasi *path*. Contoh animasi jenis ini adalah

animasi kupu-kupu yang terbang dengan kecepatan yang tidak tetap dan lintasan yang berubah-ubah. Dalam Macromedia Flash, animasi jenis ini didapatkan dengan teknik animasi *script*, teknik ini menggunakan *actionscript* yang membangkitkan sebuah lintasan berbentuk kurva dari persamaan matematis.

6) Animasi *Vektor*

Animasi vektor mirip dengan animasi *sprite*, perbedaannya hanya terletak pada gambar yang digunakan dalam objek *sprite*-nya. Pada animasi *sprite*, gambar yang digunakan adalah gambar bitmap, sedangkan animasi vektor menggunakan gambar vektor dalam objek *sprite*-nya. Penggunaan vektor ini juga mengakibatkan ukuran file animasi vektor menjadi lebih kecil dibandingkan dengan file animasi *sprite*.

7) *Morphing*

Morphing adalah mengubah satu bentuk menjadi bentuk yang lain. *Morphing* memperlihatkan serangkaian *frame* yang menciptakan gerakan halus dari bentuk pertama yang kemudian mengubah dirinya menjadi bentuk yang lain. Dalam Macromedia Flash animasi jenis ini dilakukan dengan teknik *tweeningshape*.

8) Animasi *Clay*

Animasi ini sering disebut juga animasi *doll* (boneka). Animasi ini dibuat menggunakan boneka-boneka tanah liat atau material lain yang digerakkan perlahan-lahan, kemudian setiap gerakan boneka-boneka tersebut difoto secara beruntun, setelah proses pemotretan selesai, rangkaian foto dijalankan dalam kecepatan tertentu sehingga dihasilkan gerakan animasi yang unik. Contoh penerapan animasi ini adalah pada film Chicken Run dari Dream Work Pictures. Teknik

animasi inilah yang menjadi cikal bakal animasi 3 Dimensi yang pembuatannya menggunakan alat bantu komputer.

9) Animasi *Digital*

Animasi *digital* adalah penggabungan teknik animasi *cell* (Hand Drawn) yang dibantu dengan komputer. Gambar yang sudah dibuat dengan tangan kemudian dipindai, diwarnai, diberi animasi, dan diberi efek di komputer, sehingga animasi yang didapatkan lebih hidup tetapi tetap tidak meninggalkan identitasnya sebagai animasi 2 dimensi. Contoh animasi jenis ini adalah film Spirited Away dan Lion King.

10) Animasi Karakter

Animasi karakter biasanya digunakan dalam film kartun berbasis 3 dimensi, oleh karena itu ada juga yang menyebutnya sebagai animasi 3D. Pada animasi ini setiap karakter memiliki ciri dan gerakan yang berbeda tetapi bergerak secara bersamaan. Dalam pengerjaannya, animasi jenis ini sangat mengandalkan komputer, hanya pada permulaan saja menggunakan teknik manual, yaitu pada saat pembuatan sketsa model atau model patung yang nantinya di-*scan* dengan *scanner* biasa atau 3D *scanner*. Setelah itu proses pembuatan objek dilakukan di komputer menggunakan perangkat lunak 3D modelling and animation, seperti Maya Unlimited, 3ds max dan lain sebagainya. Setelah itu dilakukan editing *video*, penambahan spesial efek dan sulih suara menggunakan perangkat lunak terpisah, bahkan ada beberapa animasi dengan teknik ini yang menggunakan alam nyata sebagai latar cerita animasi tersebut. Contoh animasi dengan teknik ini adalah Film yang berjudul Finding Nemo, Toy Story dan Moster Inc.

5. Simulasi

Media simulasi mirip dengan animasi, tetapi ada satu perbedaan yang menonjol. Bila dalam animasi kontrol dari pengguna hanyalah sebatas memutar ulang maka di dalam simulasi kontrol pengguna lebih luas lagi. Pengguna bisa memasukkan variabel-varibel tertentu untuk melihat bagaimana besarnya variabel berpengaruh

terhadap proses yang tengah dipelajari. Sebagai contoh pada simulasi pembentukan bayangan oleh suatu lensa, pengguna dapat mengubah sendiri nilai indeks bias dan kelengkungan lensa sehingga pengguna dapat melihat secara langsung bagaimana variabel-variabel tersebut berpengaruh terhadap pembentukan bayangan, berikut mafaat dari media simulasi :

1) Menyediakan suatu tiruan yang bila dilakukan pada peralatan yang sesungguhnya terlalu mahal atau berbahaya (misal simulasi melihat bentuk tegangan listrik dengan simulasi *oscilloscope* atau melakukan praktek menerbangkan pesawat dengan simulasi penerbangan).
2) Menunjukkan suatu proses abstrak di mana pengguna ingin melihat pengaruh perubahan suatu variabel terhadap proses tersebut (misal perubahan frekwensi tegangan listrik bolak balik yang melewati suatu kapasitor atau induktor).

6. Video

Video adalah gambar-gambar yang saling berurutan sehingga menimbulkan efek gerak. Pembuatan video dalam tampilan multimedia bertujuan untuk membuat tampilan yang dihasilkan akan lebih menarik. Kelebihan-kelebihan video di dalam multimedia adalah :

1) Memaparkan keadaan real dari suatu proses, fenomena atau kejadian.
2) Sebagai bagian terintegrasi dengan media lain seperti teks atau gambar, video dapat memperkaya pemaparan.
3) Pengguna dapat melakukan *replay* pada bagian-bagian tertentu untuk melihat gambaran yang lebih fokus. Hal ini sulit diwujudkan bila *video* disampaikan melalui media seperti televisi.
4) Sangat cocok untuk mengajarkan materi dalam ranah perilaku atau psikomotor.
5) Kombinasi video dan *audio* dapat lebih efektif dan lebih cepat menyampaikan pesan dibandingkan media text.

6) Menunjukkan dengan jelas suatu langkah prosedural (misal cara melukis suatu segitiga sama sisi dengan bantuan jangka).

Sementara kelemahan-kelemahan dari media video di dalam multimedia adalah :

1) Video mungkin saja kehilangan detil dalam pemaparan materi karena siswa harus mampu meningat detil dari *scene* ke *scene*.
2) Umumnya pengguna menganggap belajar melalui video lebih mudah dibandingkan melalui text, sehingga pengguna kurang terdorong untuk lebih aktif di dalam berinteraksi dengan materi.

2. 4 Game

Dalam bagian ini akan dibahas pengertian game, serta klasifikasi game berdasarkan platform maupun berdasarkan genre.

2.4. 1 Pengertian Game

Istilah "*video game*" awalnya mengacu pada jenis spesifik perangkat, yaitu sebuah sistem komputer yang menciptakan sinyal tampilan video untuk televisi, tapi kini istilah tersebut telah menjadi istilah yang mencakup segala jenis perangkat [10].

Video game dikendalikan oleh komputer melalui interaksi pengguna dan imersi audio-visual, video game memungkinkan pemain untuk menanggapi situasi yang menantang dalam dunia fantasi. Sangat penting untuk dicatat bahwa "fantasi" merujuk pada fakta bahwa dunia yang dimaksud adalah maya, dan tidak nyata, misalnya pemain dapat menjelajahi fantasi bermain di liga utama bisbol.

2.4. 2 Klasifikasi Game

Game bisa diklasifikasikan kedalam beberapa hal, diantaranya klasifikasi berdasarkan *platform*, dan berdasarkan *genre*.

2.4.2. 1 Berdasarkan Platform

Tiga kategori utama platform pada game adalah *personal computer (PC), console, dan mobile* [10].

1. *Personal computer (PC)*

   PC game adalah game yang dibuat untuk computer baik berbasis Windows, Mac, ataupun Linux. *PC* menyediakan kekuatan grafis dan pemrosesan yang kuat yang memungkinkan pengembang untuk membuat game yang mutakhir. Tetapi kelemahan game *PC* adalah mahal karena pengguna harus menggunakan hardware yang up-to-date untuk memainkan game *PC* dengan baik. Selain itu, kelemahan game *PC* bagi pengembang adalah banyaknya varian dari konfigurasi *PC* membuat sulit bagi pengembang untuk memastikan game tersebut berjalan dengan benar pada semua setup *PC*.

2. *Console*

   Console adalah *hardware* yang dibuat oleh pihak ketiga seperti Sony, Microsoft, dan Nintendo. *Console* terhubung ke televisi dan tujuan utamanya adalah untuk bermain game. *Game Console* sangat menarik bagi game pengembang karena mereka hanya perlu memikirkan satu konfigurasi *hardware* ketika membuat *software* untuk konsol. Sangat kontras dengan *PC* yang memiliki opsi konfigurasi yang tak terbatas.

3. *Mobile*

   Mobile platform terdiri atas sesuatu yang portable dan bisa digenggam, termasuk ponsel, PDA, iPods, dan handheld game seperti Nintendo DSi atau Sony PSP. Game mobile memiliki kontrol yang sederhana (terutama jika dibandingkan dengan PC).

2.4.2. 2 Berdasarkan Genre

Menurut buku "Fundamentals of Game Design" [10], genre game bisa dibagi menjadi 9, yaitu:

1. *Action*

   Action game adalah game dimana kebanyakan dari tantangan yang disajikan merupakan dari tes *physical skill* dan koordinasi pemain. Salah satu sub-genre *action game* adalah *shooters game*, baik yang 2D maupun 3D seperti *First Person Shooters* (FPS).

2. *Strategi*

   Strategi game menantang pemain untuk mencapai kemenangan dengan perencanaan, khususnya melalui perencanaan serangkaian tindakan yang dilakukan melawan satu lawan atau lebih. Kemenangan diraih dengan perencanaan matang dan pengambilan keputusan yang optimal.

3. *Role Playing Game (RPG)*

   RPG adalah game dimana pemain mengontrol satu atau lebih karakter yang biasanya di desain oleh pemain itu sendiri, dan memandu mereka melewati berbagai rintangan yang diatur oleh komputer. Perkembangan karakter dalam hal kekuatan dan kemampuannya adalah kunci dari game jenis ini.

4. *Sports*

   Sports game mensimulasikan berbagai aspek dari olahraga atletik nyata maupun imajiner, apakah itu memainkan pertandingan, *me-manage* tim dan karir, atau keduanya. Salah satu contoh game jenis ini adalah *Pro Evolution Soccer 2012* (PES 2012), dimana pemain bisa memainkan pertandingan, menjadi manajer tim, maupun menjadi pemain dan mengembangkan karirnya sendiri.

5. *Vechicle Simulation*

   Vechicle simulation membuat feeling mengendarai kendaraan, baik

kendaraan nyata maupun kendaraan imajiner. Performa dan karakteristik kasar mesin harus menyerupai kenyataan, kecuali jika yang didesain adalah kendaraan imajiner.

6. *Construction and Management Simulation*

   CMS game adalah game tentang proses. Tujuan pemain bukan untuk mengalahkan musuh, tetapi membangun sesuatu dengan konteks proses yang sedang berjalan. Semakin pemain mengerti dan mengontrol proses, semakin sukses sesuatu yang ia bangun. Game seperti ini biasanya menyediakan dua jenis permainan, yaitu mode bebas dimana pemain bebas membangun sesuatu, dan mode misi dimana terdapat skenario hal apa yang harus dibangun oleh pemain.

7. *Adventure*

   Adventure game adalah cerita interaktif tentang karakter protagonis yang dimainkan oleh pemain. Penyampaian cerita dan eksplorasi adalah elemen inti dari game ini. Penyelesaian teka-teki dan tantangan konseptual adalah bagian besar dari permainan.

8. *Artificial Life and Puzzle Game*

   Artificial Life game adalah game yang membuat tiruan dari kehidupan sebenarnya. Biasanya ada dua jenis game ini, tiruan kehidupan manusia, contohnya The SIMS, dan tiruan kehidupan binatang, contohnya Tamagochi.

9. *Online Game*

   Istilah *online game* disini mengacu kepada multiplayer game dimana mesin dari para pemain terhubung dengan jaringan.

2. 5 AI (Artificial Intelligence)

Artificial Intelligence (AI) atau kecerdasan buatan merupakan sub-bidang pengetahuan komputer yang khusus ditujukan untuk membuat *software* dan *hardware* yang sepenuhnya bisa untuk menirukan beberapa fungsi otak manusia. Dengan demikian

diharapkan komputer bisa membantu manusia dalam memecahkan berbagai masalah yang lebih rumit. AI juga didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan diciptakan dan dimasukkan kedalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Diantara yang menggunakan kecerdasan buatan antara lain sistem pakar, games, logika fuzzy, jaringan syaraf tiruan dan robotika.

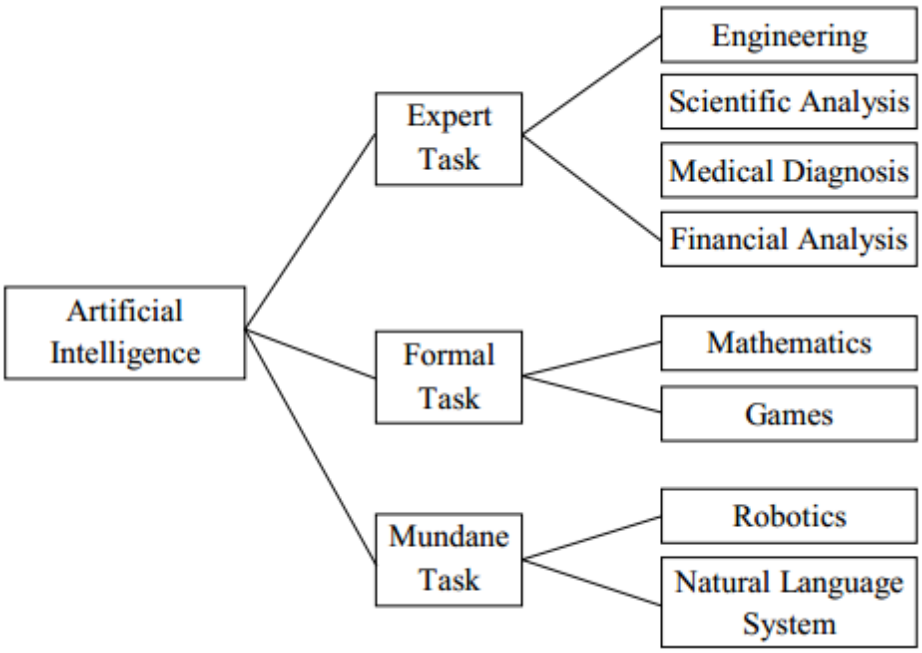
Ada beberapa pengertian Artificial Intelligence diantaranya sebagai berikut :

1. Suatu cara yang sederhana untuk membuat komputer dapat "berpikir" secara *intelligent*.
2. Bagian dari ilmu komputer yang mempelajari perancangan sistem komputer yang *intelligent*, yaitu suatu sistem yang memperlihatkan karakteristik yang ada pada tingkah laku manusia.
3. Suatu studi bagaimana membuat komputer dapat mengerjakan sesuatu, yang pada saat ini orang dapat mengerjakan lebih baik.
4. Bidang ilmu komputer yang memungkinkannya untuk memahami, bernalar dan bertindak.

Dari beberapa pengertian diatas, definisi AI dapat disimpulkan kedalam empat kategori, yaitu :

1. Sistem yang dapat berfikir seperti manusia "*Thinking humanly*".
2. Sistem yang dapat bertingkah laku seperti manusia "*Acting Humanly*".
3. Sistem yang dapat berfikir secara rasional "*Thinking rationally*".
4. Sistem yang dapat bertingkah laku secara rasional "*Acting rationally*".

AI seperti bidang ilmu lainnya juga memiliki sejumlah sub-disiplin ilmu yang sering digunakan untuk pendekatan yang esensial bagi penyelesaian suatu masalah dan dengan aplikasi bidang AI yang berbeda. Gambar 2.1 merupakan sejumlah bidang-bidang tugas (*task domains*) dari AI.



Gambar 2. 1 Bidang – Bidang Tugas (*Task Domaons*) dari AI

Aplikasi penggunaan AI dapat dibagi ke dalam tiga kelompok, yaitu :

1. Mundane task

Secara harfiah, arti mundane adalah keduniaan. Di sini, AI digunakan untuk melakukan hal-hal yang sifatnya duniawi atau melakukan kegiatan yang dapat membantu manusia. Contohnya :

a) Persepsi (*vision & speech*).
b) Bahasa alami (*understanding, generation & translation*).
c) Pemikiran yang bersifat *commonsense*.
d) *Robot control*.

2. *Formal task*

AI digunakan untuk melakukan tugas-tugas formal yang selama ini manusia biasa lakukan dengan lebih baik. Contohnya :

a) Permainan/*games.*
b) Matematika (geometri, logika, kalkulus, integral, pembuktian).

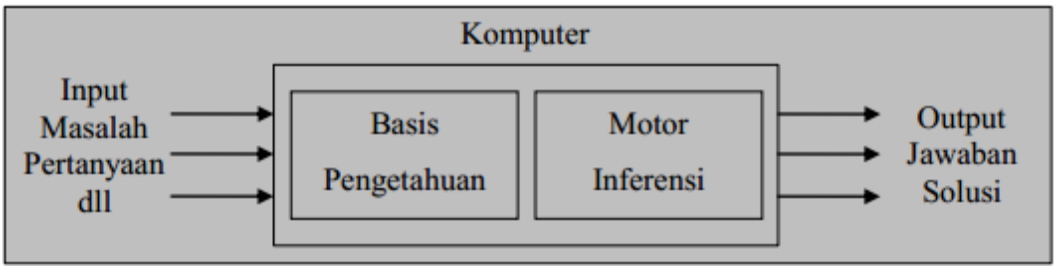
3. Expert task

AI dibentuk berdasarkan pengalaman dan pengetahuan yang dimiliki oleh para ahli. Penggunaan ini dapat membantu para ahli untuk menyampaikan ilmu-ilmu yang mereka miliki. Contohnya :

a) Analisis finansial
b) Analisis medikal
c) Analisis ilmu pengetahuan
d) Rekayasa (desain, pencarian, kegagalan, perencanaan, manufaktur)

Aplikasi *Artificial Intelegent* memiliki dua bagian utama, yaitu :

1. Basis Pengetahuan (*Knowledge Base*) : berisi fakta-fakta, teori, pemikiran dan hubungan antara satu dengan lainnya.
2. Motor Inferensi (*Inference Engine*) : kemampuan menarik kesimpulan berdasarkan pengalaman.



Gambar 2. 2 Penerapan Konsep Kecerdasan Buatan di Komputer [11]

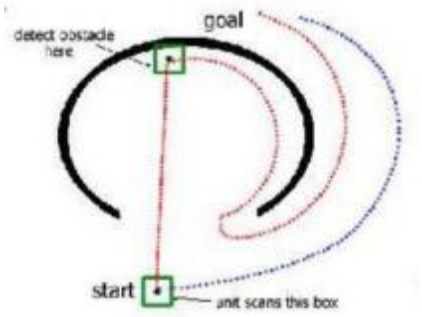
2.5. 1 Teknik-Teknik Dasar Pencarian

Pencarian atau pelacakan merupakan salah satu teknik untuk menyelesaikan permasalahan AI. Keberhasilan suatu sistem salah satunya ditentukan oleh kesuksesan dalam pencarian dan pencocokan. Teknik dasar pencarian memberikan suatu kunci bagi banyak sejarah penyelesaian yang penting dalam bidang AI.[11]

2.5. 2 Pathfinding

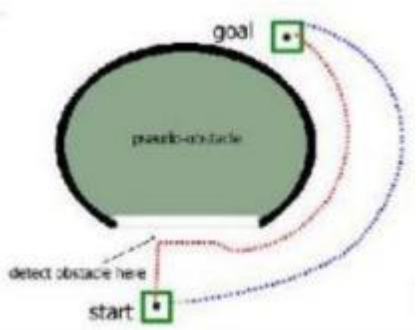
Pathfinding (pencarian jalan/*rute*) adalah salah satu bidang penerapan yang sering ditangani oleh kecerdasan buatan khususnya dengan menggunakan algoritma pencarian. Penerapan yang dapat dilakukan dengan *pathfinding* antara lain adalah pencarian *rute* dalam suatu game dan pada suatu peta. Algoritma pencarian yang dipakai harus dapat mengenali jalan dan elemen peta yang tidak dapat dilewati.

Sebuah algoritma *pathfinding* yang baik dapat bermanfaat untuk mendeteksi halangan/rintangan yang ada pada medan dan menemukan jalan untuk menghindarinya, sehingga jalan yang ditempuh lebih pendek daripada yang seharusnya bila tidak menggunakan algoritma *pathfinding*. Lihat ilustrasi pada Gambar 2.3.



Gambar 2. 3 Penentuan Rute Tanpa Pathfinding

Pada Gambar 2.5, dari start menuju *goal*, tanpa algoritma *pathfinding*, unit hanya akan memeriksa lingkungan sekitarnya saja (dilambangkan dengan daerah di dalam kotak hijau). Unit tersebut akan maju terus ke atas untuk menca pai tujuan, baru setelah mendekati adanya halangan, lalu berjalan memutar untuk menghindarinya. Sebaliknya, penentuan *rule* dengan algoritma *pathfinding* pada Gambar 2.4 akan memprediksi ke depan mencari jalan yang lebih pendek menghindari halangan (dilambangkan garis biru) untuk mencapai tujuan, tanpa pernah mengirim unit ke dalam "perangkap" halangan berbentuk U. Karena itu peran algoritma *pathfinding* sangat berguna untuk memecahkan berbagai permasalahan dalam penentuan *rute*.



Gambar 2. 4 Penentuan Rute dengan Pathfinding

2.5. 3 Algoritma Pencarian (*Search Algorithms)*

Penerapan kecerdasan buatan *(Artificial intelligence)* untuk pemecahan masalah (*problem solving*) dalam bidang ilmu komputer telah mengalami perkembangan yang pesat dari tahun ke tahun seiring perkembangan kecerdasan buatan itu sendiri. Permasalahan yang melibatkan pencarian (*searching*) adalah salah satu contoh penggunaan kecerdasan buatan yang cukup populer untuk memecahkan berbagai macam permasalahan.

Penerapannya bermacam-macam, mulai masalah dunia nyata, seperti penetuan rute pada suatu peta, *travelling salesman problem* (TSP), penentuan urutan perakitan (*assembly sequencing*) oleh robot, sampai penerapan dalam dunia *game*, seperti membuat komputer dapat bermain catur layaknya manusia ataupun penentuan pengambilan jalan karakter dalam sebuah *game*.

Dalam cara kerjanya, sistem menggunakan algoritma tertentu untuk mencapai kondisi yang diinginkan atau menemukan hasil yang dicari dari kondisi atau input yang ada sekarang. Dalam algoritma pencarian, dikenal istilah "*state*" yang berarti kondisi. Kondisi akhir yang hendak dituju dikenal dengan istilah "*goal state*". Contoh *state* antara lain, dalam game catur misalnya, adalah letak tiap buah catur pada papan. *Goal state* dalam kasus ini biasanya kondisi raja terskak mati.

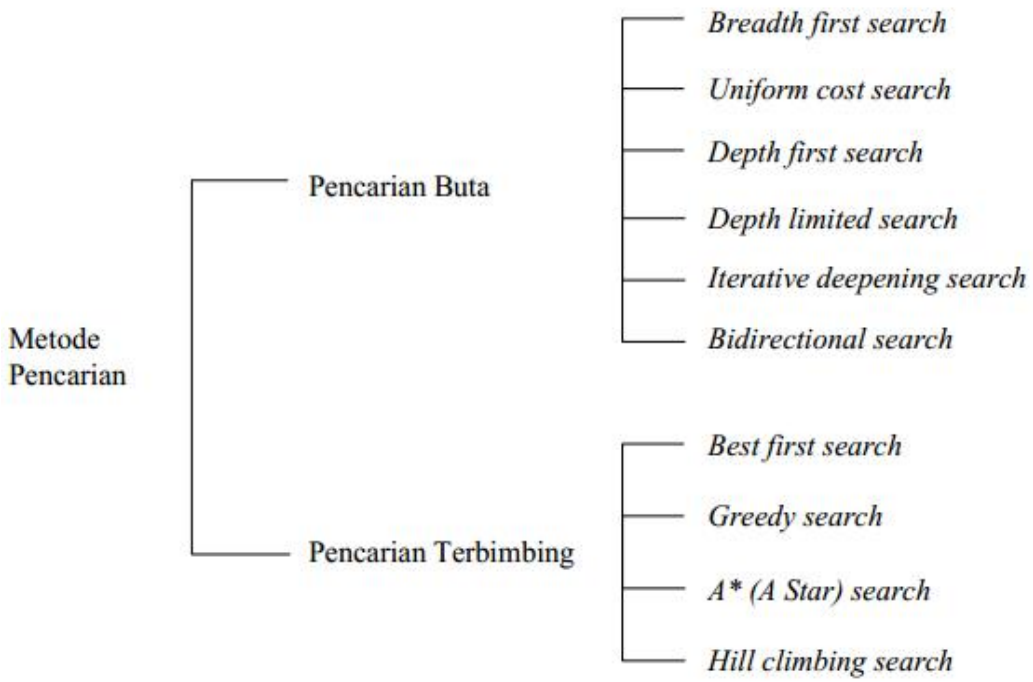
Pada umumnya, algoritma pencarian bekerja dengan mengembangkan berbagai kemungkinan *state* yang mungkin dicapai dari *state* sekarang. State dalam proses pencarian biasa disebut dengan istilah *node*. Kumpulan *node* akan terus dikembangkan sampai ditemukan *node* yang merupakan *goal state* atau bila sudah tidak ada lagi *node* yang dikembangkan (berarti tidak ditemukan solusi). Rangkaian kumpulan *node* dari state

awal sampai goal state yang terbaiklah yang akhirnya diambil menjadi solusi. Kriteria terbaik disini tergantung pada kasus yang dihadapi. Pada penetuan rute misalnya, terbaik biasanya adalah solusi yang memberikan jalan terpendek atau tercepat untuk mencapai tujuan.

Berbagai algoritma untuk pencarian (*searching algorithm*) yang ada berbeda satu dengan yang lain dalam hal pengembangan kumpulan *node* untuk mencapai *goal state*. Perbedaan ini terutama dalam hal cara dan urutan pengembangan *node*, dan sangat berpengaruh pada kinerja masing-masing algoritma. Empat kriteria yang menjadi ukuran algoritma pencarian adalah :

1. *Completeness*
   apakah algoritma pasti dapat menemukan solusi?
2. *Time Comlexity*
   berapa lama waktu yang dibutuhkan untuk menemukan sebuah solusi?
3. *Space Complexity*
   berapa memori atau resource yang diperlukan untuk melakukan pencarian?
4. *Optimality*
   apakah algoritma tersebut dapat menemukan solusi yang terbaik jika terdapat beberapa solusi yang berbeda?

Namun perlu diingat, algoritma pencarian yang dikatakan terbaik sekalipun belum tentu sesuai untuk semua jenis kasus/masalah. Harus dipilih algoritma pencarian yang sesuai dengan kebutuhan kasus yang dihadapi. Menurut cara algoritma mengembangkan *node* dalam proses pencarian, terdapat dua golongan, yakni *Uninformed Search/Blind Search* dan *Informed Search/Heuristic Search.*



Gambar 2. 5 Bagan Metode Pencarian (*Searching*)

Dari Gambar 2.3 bagan metode pencarian/penelusuran dibagi menjadi dua golongan, yakni pencarian buta (*blind search*) dan pencarian terbimbing (*heuristic search*). [11]

A. Pencarian Buta (*Blind Search*)

Blind Search adalah pencarian solusi tanpa adanya informasi yang dapat mengarahkan pencarian untuk mencapai *goal state* dari *current state* (keadaan sekarang). Informasi yang ada hanyalah definisi *goal state* itu sendiri, sehingga algoritma dapat mengenali *goal state* bila menjumpainya.

Dengan ketiadaan informasi, maka *blind search* dalam kerjanya memeriksa/mengembangkan *node-node* secara tidak terarah dan kurang efisien untuk kebanyakan kasus karena banyaknya *node* yang dikembangkan.

Beberapa contoh algoritma yang termasuk *blind seacrh* antara lain adalah *Breadth First Search, Uniform Cost Search, Depth First Search, Depth Limited Search, Iterative Deepening Search*, dan *Bidirectional Search.* [12]

B. Pencarian Terbimbing (*Heuristic Search*)

Berbeda dengan *blind search*, *heuristic search* mempunyai informasi tentang *cost*/biaya untuk mencapai *goal state* dari *current state*. Dengan informasi tersebut, *heuristic search* dapat melakukan pertimbangan untuk mengembangkan atau memeriksa *node-node* yang mengarah ke *goal state*. Misalnya pada pencarian *rute* pada suatu peta, bila berangkat dari kota A ke kota tujuan B yang letaknya di Utara kota A, dengan *heuristic search*, pencarian akan lebih difokuskan ke arah Utara (dengan informasi *cost* ke *goal*), sehingga secara umum, *heuristic search* lebih efisien daripada *blind search*.

Heuristic search untuk menghitung (perkiraan) *cost* ke *goal state*, digunakan fungsi *heuristic*. Fungsi *heuristic* berbeda daripada algoritma, dimana *heuristic* lebih merupakan perkiraan untuk membantu algoritma, dan tidak harus valid setiap waktu. Meskipun begitu, semakin bagus fungsi *heuristic* yang dipakai, semakin cepat dan akurat pula solusi yang didapat. Menentukan *heuristic* yang tepat untuk kasus dan implementasi yang ada juga sangat berpengaruh terhadap kinerja algoritma pencarian.

Beberapa contoh algoritma pencarian yang menggunakan metode *heuristic* search adalah : *Best First Search, Greedy Search, A* (A Star) Search, dan Hill Climbing Search.*

Tabel 2. 3 Kegunaan *Heuristic Search*

Algoritma Pencarian	**Kegunaan**	**Implementasi**
Best First Search	Penelusurannya hanya menggunakan *estimasi cost* / jarak ke *node* tujuan, $h(n)$ akibatnya pencarian tidak menyeluruh.	Sistem pakar, penjadwalan, pencarian *rute* pada peta geografis dan lain-lain
Greedy Search	Pada *greedy search* ide utamanya adalah	Masalah penukaran uang, minimisasi waktu di dalam

	mengembangkan *node* dengan nilai estimasi biaya ke *goal* yang paling kecil (berarti *node* yang paling dekat ke tujuan).	sistem (penjadwalan), memilih beberapa jenis investasi (penanaman modal), memilih jurusan di perguruan tinggi, masalah *knapsnak* dan lainnya.
A (A Star) Search*	Menjumlahkan jarak sebenarnya dengan estimasi jaraknya dan pencariannya menyeluruh, tetapi akan memakai memori yang cukup besar untuk menyinpan *node* sebelumnya	Pencarian jalur terpendek pada peta, pencarian jalur angkutan kota, pencarian jalan pada *game*, menggerakkan karakter *game*, pencarian jalan untuk permainan *Lose Your Marble*, penyelesaian permasalahan 8 Puzzle dan lain-lain.
Hill Climbing Search	*Hill climbing* sering digunakan ketika fungsi *heuristic* yang bagus tersedia untuk mengevaluasi *state* tapi ketika tidak ada yang perlu dievaluasi maka fungsi ini tidak ada.	Sistem pakar, pencarian lokasi pada peta, dan lain-lain.

2.5. 4 *Best-First Search*

Sesuai dengan namanya, *best-first search* merupakan sebuah metode yang membangkitkan simpul dari sebuah simpul sebelumnya (yang sejauh ini terbaik di antara semua simpul yang pernah dibangkitkan). Penentuan simpul terbaik dilakukan dengan menggunakan sebuah fungsi yang disebut fungsi evaluasi f(n) [13]. Pada *best-first search* jika *successor* digunakan, maka dikatakan algoritma mengembangkan simpul tersebut.

Setiap sebuah simpul dikembangkan, algoritma akan menyimpan setiap *successor* simpul *n* sekaligus dengan harga (*cost*) dan petunjuk pendahulunya yang disebut "*parent*". Algoritma akan berakhir pada simpul tujuan, dan tidak ada lagi pengembangan simpul. Fungsi evaluasi pada *best-first search* dapat berupa informasi biaya perkiraan dan suatu simpul menuju ke simpul tujuan atau gabungan antara biaya sebenarnya dan biaya perkiraan tersebut. Biaya perkiraan dapat diperoleh dengan menggunakan suatu fungsi yang disebut fungsi *heuristic*. Pada strategi *best-first search*, *cost* sebenarnya yaitu dari simpul awal ke simpul *n*, dinotasikan dengan *g(n*) dan fungsi *heuristic* yang digunakan yaitu perkiraan/estimasi nilai dari simpul *n* ke simpul tujuan dinotasikan dengan *h*(*n*). Algoritma yang menggunakan metode *best-first search*, yaitu :

1. Greedy Best-First Search
2. Algoritma A*

Berikut akan diberikan beberapa istilah yang sering digunakan pada metode *best-first search* :

- *Start Node* adalah sebuah terminologi untuk posisi awal sebuah pencarian.
- *Current Node* adalah simpul yang sedang dijalankan (yang sekarang) dalam algoritma pencarian jalan terpendek.
- Kandidat (*successor*) adalah simpul-simpul yang berbatasan dengan *current node*, dengan kata lain simpul-simpul yang akan diperiksa berikutnya.
- Simpul (*node*) merupakan representasi dari area pencarian
- *Open list* adalah tempat menyimpan data simpul yang mungkin diakses dari *starting node* maupun simpul yang sedang dijalankan.
- *Closed list* adalah tempat menyimpan data simpul yang juga merupakan bagian dari jalur terpendek yang telah berhasil didapatkan.
- *Goal node* yaitu simpul tujuan.
- *Parent* adalah *current node* dari *successor*/kandidat.

2.5. 5 *Greedy Best First Search*

Salah satu algoritma yang termasuk kedalam kategori *heuristic search* adalah *gready best first search* yang dikenal juga dengan *greedy search*. Secara harfiah *greedy* artinya rakus atau tamak, sifat yang berkonotasi negatif. Sesuai dengan arti tersebut, prinsip *greedy* adalah mengambil keputusan yang dianggap terbaik hanya untuk saat itu saja yang diharapkan dapat memberikan solusi terbaik secara keseluruhan. Oleh karena itu, pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan. Keputusan yang telah diambil pada suatu langkah tidak dapat diubah lagi pada langkah selanjutnya.

Greedy Best First Search seperti halnya algoritma yang menggunakan strategi *best-first search* lainnya mempunyai sebuah fungsi yang menjadi acuan kelayakan sebuah simpul yaitu fungsi evaluasi *f*(*n*). Pada *Greedy Best First Search* fungsi evaluasi tidak bergantung pada *cost* sebenarnya, tetapi hanya tergantung pada fungsi *heuristic* itu sendiri. Jika pada algoritma *Dijkstra* pencarian yang dilakukan tergantung pada *cost* sebenarnya dari sebuah simpul yaitu *g*(*n*), pada *Greedy best first search* fungsi evaluasi hanya tergantung pada fungsi *heuristic h*(*n*) yang mengestimasikan arah yang benar, sehingga pencarian jalur dapat berlangsung dengan sangat cepat

Berikut langkah-langkah pencarian lintasan terpendek yang dilakukan

Greedy Best-First Search:

1. Masukkan simpul awal ke dalam *open list*.
2. Open berisi simpul awal dan *closed list* masih kosong.
3. Masukkan simpul awal ke *closed list* dan suksesornya pada *open list.*
4. Ulangi langkah berikut sampai simpul tujuan ditemukan dan tidak ada lagi simpul yang akan dikembangkan.
    - Hitung nilai *f* simpul-simpul yang ada pada *open list*, ambil simpul terbaik (*f* paling kecil).
    - Jika simpul tersebut sama dengan simpul tujuan, maka sukses
    - Jika tidak, masukan simpul tersebut ke dalam *closed*
    - Bangkitkan semua *successor* dari simpul tersebut
    - Untuk setiap *successor* kerjakan :

- Jika *succesor* tersebut belum pernah dibangkitkan, evaluasi *successor* tersebut, tambahkan *open*, dan catat "parent"-nya.
- Jika *successor* tersebut sudah pernah dibangkitkan, ubah parent-nya. Jika jalur melalui parent ini lebih baik dari jalur melalui parent yang sebelumnya. Selanjutnya, perbarui biaya untuk successor tersebut.

2.5. 6 Algoritma A* (*A Star*)

Algoritma A* (*A Star*) adalah algoritma yang menggabungkan algoritma *Djikstra* dan algoritma *Greedy Best First Search*. Selain menghitung biaya yang diperlukan untuk berjalan dari simpul satu ke simpul lainnya, algoritma A* juga menggunakan fungsi *heuristic* untuk memprioritaskan pemeriksaan simpul-simpul pada arah yang benar, sehingga algoritma A* mempunyai efisiensi waktu yang baik dengan tidak mengorbankan perhitungan biaya sebenarnya.

2.5. 7 Sejarah Algoritma A* (*A Star*)

Penggunaan informasi *heuristic* untuk meningkatkan efisiensi pencarian telah dipelajari dalam berbagai bidang. Penggunaan fungsi evaluasi pada masalah pencarian pada graft dikemukakan oleh Lin pada tahun 1965 yang digunakan pada masalah *Traveling Salesman Problem* (TSP) [13]. Pada 1966 Doran dan Michie merumuskan dan mencoba dengan algoritma *best-first search* yang menggunakan perkiraan jarak dan *current node* ke simpul tujuan [14]. Hart, Nilsson dan Raphael memperkenalkan penggunaan sebuah algoritma dalam masalah optimasi yaitu algoritma A* (*A star*). Versi *bi-directional* algoritma A*, yang secara bersamaan mencari dari simpul awal dan simpul tujuan diperkenalkan oleh Pohl pada tahun 1971 [15], yang selanjutnya diteliti oleh de *Champeaux* dan Sint pada tahun 1977 [16].

2.5. 8 Algoritma A* dalam Pencarian Rute Terpendek

Beberapa perbedaan strategi *best-first search* hanya pada bentuk fungsi evaluasi yang digunakan. Pada strategi *best-first search*, simpul yang terpilih untuk dikembangkan

adalah simpul dengan nilai fungsi evaluasi terendah dan ketika dua lintasan mengarah pada simpul yan g sama, simpul dengan nilai paling besar akan dibuang, secara matematis.

Pencarian menggunakan algoritma A* mempunyai prinsip yang sama dengan algoritma BFS, hanya saja dengan dua faktor tambahan [17].

1. Setiap sisi mempunyai "*cost*" yang berbeda-beda, seberapa besar *cost* untuk pergi dari satu simpul ke simpul yang lain.
2. *Cost* dari setiap simpul ke simpul tujuan bisa diperkirakan. Ini membantu pencarian, sehingga lebih kecil kemungkinan mencari ke arah yang salah. *Cost* untuk setiap simpul tidak harus berupa jarak. *Cost* bisa saja berupa waktu bila ingin mencari jalan dengan waktu tercepat untuk dilalui. Sebagai contoh, bila berkendaraan melewati jalan biasa bisa saja merupakan jarak terdekat, tetapi melewati jalan tol biasanya memakan waktu lebih sedikit.

Algoritma bekerja dengan prinsip yang hampir sama dengan BFS, kecuali dengan dua perbedaan, yaitu :

1. Simpul-simpul di *list* "terbuka" diurutkan oleh *cost* keseluruhan dari simpul awal ke simpul tujuan, dari *cost* terkecil sampai *cost* terbesar. Dengan kata lain, menggunakan *priority queue* (antrian prioritas). *Cost* keseluruhan dihitung dari *cost* dari simpul awal ke simpul sekarang (*current node*) ditambah *cost* perkiraan menuju simpul tujuan.
2. Simpul di *list* "tertutu" bisa dimasukkan ke *list* "terbuka" bila jalan terpendek (*cost* lebih kecil) menuju simpul tersebut ditemukan.

Karena *list* "terbuka" diurutkan berdasarkan perkiraan *cost* keseluruhan, algoritma mengecek simpul-simpul yang mempunyai perkiraan *cost* yang paling kecil terlebih dahulu, jadi algoritmanya mencari simpul-simpul yang kemungkinan mengarah ke simpul tujuan. Karena itu, lebih baik perkiraan costnya, lebih cepat pencariannya. *Cost* dan perkiraannya ditentukan oleh kita sendiri. Bila *cost*-nya adalah jarak, akan menjadi mudah.

Cost antara simpul adalah jaraknya, dan perkiraan *cost* dari suatu simpul ke simpul tujuan adalah penjumlahan jarak dari simpul tersebut ke simpul tujuan. Atau agar lebih mudahnya bisa ditunjukkan seperti berikut ini.

$$F(n)=g(n)+h(n) \quad (2.1)$$

Dengan :

$f(n)$ = Fungsi evaluasi

$g(n)$ = Biaya (*cost*) dari keadaan awal (*start node*) sampai keadaan *n*

$h(n)$ = Estimasi biaya dari keadaan n sampai tujuan (*goal node*)

Perhatikan bahwa algoritma ini hanya bekerja bila cost perkiraan tidak lebih besar dari cost yang sebenarnya. Bila cost perkiraan lebih besar, bisa jadi jalan yang ditemukan bukanlah yang terpendek.[17]

Node dengan nilai terendah merupakan solusi terbaik untuk diperiksa pertama kali pada $g(n) + h(n)$. Dengan fungsi *heuristic* yang memenuhi kondisi tersebut, maka pencarian dengan algoritma A* dapat optimal.

Algoritma A* menggunakan dua buah *list* yaitu *Open List dan Closed List*. Seperti halnya *best-first search* yang lain kedua list mempunyai fungsi yang sama. Pada awalnya *Open List* hanya berisi satu simpul yaitu simpul awal dan *Closed List* masih kosong. Perlu diperhatikan setiap simpul akan menyimpan petunjuk "*parent*"nya sehingga setelah pencarian berakhir lintasan juga akan didapatkan.

Berikut adalah langkah-langkah algoritma A* (A star) :

1. Masukkan simpul awal ke *Open List*.
2. Ulangi langkah berikut sampai pencarian berakhir.
   - Cari node n dengan nilai $f(n)$ paling rendah, dalam *Open List*. *Node* ini akan menjadi current *node*.
   - Keluarkan current *node* dari *Open List* dan masukkan ke *Closed List*.

- Untuk setiap *successor* dari *current node* lakukan langkah berikut :
    a) Jika sudah terdapat dalam *Closed List*, abaikan, jika tidak lanjutkan.
    b) Jika belum ada pada *Open List*, masukkan ke *Open List*. Simpan *current node* sebagai *parent* dari *successor-successor* ini. Simpan *cost* masing-masing simpul.
    c) Jika belum ada dalam *Open List*, periksa jika simpul *successor* ini mempunyai nilai lebih kecil dibanding *successor* sebelumnya. Jika lebih kecil, jadikan sebagai *current node* dan ganti *parent node* ini.
- Walaupun telah mencapai simpul tujuan, jika masih ada *successor* yang memiliki nilai yang lebih kecil, maka simpul tersebut akan terus dipilih sampai bobotnya jauh lebih besar atau mencapai simpul a khir dengan bobot yang lebih kecil dibanding dengan simpul sebelumnya yang telah mencapai simpul tujuan.
- Pada setiap pemilihan simpul berikutnya, nilai $f(n)$ akan dievakuasi, dan jika terdapat nilai $f(n)$ yang sama maka akan dipilih berdasarkan nilai $g(n)$ terbesar.

Metode A* mirip dengan algoritma pencarian graph yang berpotensial mencari daerah yang luas pada sebuah peta. Metode A* mempunyai fungsi *heuristic* untuk memandu pencarian ke depan sampai tujuan. Metode A* dapat melakukan *backtracking* jika jalur yang ditempuh ternyata salah. Metode A*

dapat melakukannya karena menyimpan jejak / *track* yang mungkin sebagai jalur

yang optimal. Sebagai contoh, jika kita sedang menuju suatu kota dan sampai pada persimpangan jalan, dan memutuskan untuk belok kiri daripa da ke kanan, dan ternyata bila jalan yang dipilih ternyata salah, kita akan kembali ke persimpangan dan mengambil jalan satunya. Itulah yang dilakukan metode A* ini.

Performansi algoritma A* dapat diketahui dengan melihat perbandingan metode dari Tabel 2.4.

Tabel 2. 4 Perbedaan Ketiga Algoritma

No.	Nama Algoritma	Metode	Keterangan
1	*Optimal Search* *(Djikstra's Algorithm*)	$g(n_i) = g(n) + c(n,n_i)$	$g(n)$ adalah *cost* dari IS ke *node* n $c(n,n_i)$ adalah *cost* dari *node* n ke n_i
2	*Best First Search*	$h(n)$	$h(n)$ adalah *estimed cost* dari jalur terpendek dari *node* n ke GS
3	*A**	$f(n) = g(n) + h(n)$	$f(n)$ adalah fungsi *heruistic* $g(n)$ adalah *cost* dari IS ke *node* n $h(n)$ adalah *estimed cost* dari jalur terpendek dari *node* n ke GS

Dari informasi pada Tabel 2.2 di atas dapat dibuktikan bahwa bila :

a. $g(n) = 0$, maka $f(n) = h(n)$ sehingga algoritma A* akan bertingkah laku sebagaimana *Best First Search*.
b. $h(n) = 0$, maka $f(n) = g(n)$ sehingga algoritma A* akan bertingkah laku sebagaimana *Optimal Search* (*Dijkstra's Algorithm*).

Dengan demikian dapat diambil kesimpulan bahwa algoritma A* mengkombinasikan kelebihan dari algoritma *Optimal Search dan Best First Search.* Dari informasi ini, dapat menganalisa dan membandingkan *cost* ketiga algoritma tersebut.

2.5. 9 *Heuristic Best First Seach*

Fungsi *heuristic* *h*(*n*) merupakan estimasi *cost* dari *n* ke simpul tujuan. Sangat penting untuk memilih fungsi *heuristic* yang baik. Misalkan h*(n) merupakan *cost* sebenarnya dari simpul *n* ke simpul tujuan, maka pada algoritma A* terdapat kemungkinan yang terjadi pada pemilihan fungsi *heuristic* yang digunakan yaitu (Amit Gaming):

1. Jika $h(n) - 0$, sehingga hanya $g(n)$ yang terlibat maka A* akan bekerja seperti halnya algoritma *Djikstra*.
2. Jika $h(n) \leq h^*(n)$, maka A* akan mengembangkan titik dengan nilai paling rendah dan algoritma A* menjamin ditemukannya lintasan terpendek. Nilai $h(n)$ terendah akan membuat algoritma mengembangkan lebih Banyak simpul. Jika $h(n) \leq h^*(n)$, maka $h(n)$ dikatakan *heuristic* yang *admissible*.
3. Jika $h(n) = h^*(n)$, maka A* akan mengikuti lintasan terbaik dan tidak akan mengembangkan titik-titik yang lain sehingga akan berjalan cepat. Tetapi hal ini tidak akan terjadi pada semua kasus. Informasi yang baik akan mempercepat kinerja A*.
4. Jika $h(n) \geq h^*(n)$, maka A* tidak menjamin pencarian rute terpendek, tetapi berjalan dengan cepat.
5. Jika $h(n)$ terlalu tinggi relative dengan $g(n)$ sehingga hanya $h(n)$ yang bekerja maka A* berubah jadi *Greedy Best First Search*.

   Berikut beberapa *heuristic* yang biasa digunakan yaitu :

2.5. 10 *Manhattan Distance*

Manhattan distance atau sering disebut *Taxicab Geometry, city block distance*, diperkenalkan oleh Hermann Minkowski pada abad ke -19. Manhattan

distance merupakan *heuristic standar*. Fungsi *heuristic* ini digunakan untuk kasus dengan pergerakan pada peta hanya lurus x,y (horisontal atau vertikal), tidak diperbolehkan pergerakan diagonal. *Manhattan distance* antara dua *vektor* p,q pada

sebuah dimensi *n* adalah penjumlahan panjang proyeksi garis antara dua objek. Secara formal perhitungan nilai *heuristic* untuk simpul ke-*n* menggunakan

Manhattan distance adalah sebagai berikut :

$$d(p,q) = |p - q| = \sum_{i=1}^{n} |p_i - q_i| \quad (2.2)$$

Pada kasus dua dimensi dan pada peta geografis *Manhattan Distance* adalah sebagai berikut :

$$h(n) = (abs(n.x - tujuan.x + abs(n.y - tujuan.y)) \quad (2.3)$$

Dengan :

h(*n*) = nilai *heuristic* untuk simpul *n*

n.x = nilai koordinat *x* dari simpul *n*

n.y = nilai koordinat *y* dari simpul *n*

x-tujuan = nilai koordinat *x* dari simpul tujuan

y-tujuan = nilai koordinat *y* dari simpul tujuan

2.5. 11 *Euclidean Distanc*

Euclidean distance didefinisikan sebagai panjang dari garis lurus yang menghubungkan posisi dua buah objek. Secara logis diketahui bahwa jarak terpendek antara dua titik adalah garis lurus antara kedua titik tersebut. *Euclidean distance* digunakan jika proses dapat bergerak ke segala arah. *Heuristic* ini akan menghitung jarak berdasarkan panjang garis yang dapat ditarik dari dua buah titik. Perhitungannya dapat ditulis sebagai berikut:

$$h(n) = \sqrt{(n.x - tujuan.x)^2 + (n.y - tujuan.y)^2} \quad (2.4)$$

Dalam kasus ini, skala relatif nilai g mungkin akan tidak sesuai lagi dengan nilai fungsi *heuristic h*. Karena *Euclidean distance* selalu lebih pendek dari *Manhattan*

distance, maka dapat dipastikan selalu akan didapatkan jalur terpendek, walaupun secara komputasi lebih berat.

Dalam beberapa literatur juga disebutkan jika nilai g adalah 0, maka lebih baik jika ongkos komputasi operasi pengakaran pada *heuristic Euclidean distance*

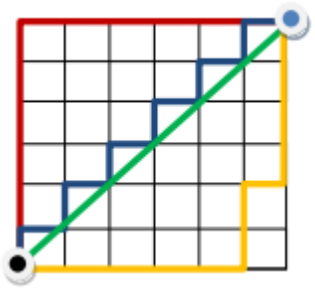
dihilangkan saja, menghasilkan rumus sebagi berikut:

$$h(n) = (n.x\text{-}tujuan.x)^2 + (n.y\text{-}tujuan.y)^2 \quad (2.5)$$

Hal lain yang harus diperhatikan adalah seberapa cepat fungsi *heuristic* dapat dikomputasi. Selalu akan ada *tradeoff* antara akurasi dari fungsi *heuristic* dan waktu yang dibutuhkannya untuk mengomputasinya. Nampaknya bagus jika fungsi *heuristic* yang digunakan sangat akurat, dilihat dari berbagai macam percobaan bahwa jika *heuristic* yang digunakan sempurna maka A* akan selalu melewati jalur yang tepat dan akan selalu memberikan optimum global. Namun, *heuristic* yang sempurna semacam itu tidak ada (dan tidak akan pernah ada), dan bahkan untuk mendekatinya saja akan memerlukan tambahan komputasi yang tidak ringan. Seringkali dalam aplikasinya *heuristic* yang memberikan hasil yang sangat akurat namun lambat, kurang disenangi dibanding *heuristic* yang tidak begitu optimal namun memberikan hasil dengan cepat. Maka dari itu, pemilihan *heuristic* sangat bergantung pada tujuan penggunaan A*. Jika hasil yang dibutuhkan adalah optimum global, maka fungsi *heuristic* yang digunakannya haruslah "sempurna", sedang jika yang dibutuhkan adalah hasil yang cepat dan tidak harus jalur terpendek, maka lebih bijak menggunakan *heuristic* yang lebih ringan.

Euclidean distance bertujuan untuk memprioritaskan *node-node* yang berada dekat garis lurus antara simpul awal dan simpul tujuan. Pendekatan ini dapat sangat membantu algoritma A* karena nilainya yang tidak pernah akan melebihi nilai sebenarnya. Namun pendekatan ini dapat tidak berpengaruh ataupun malah memperlambat kinerja algoritma A*.

Karena *Euclidean distance* lebih pendek dari *Manhattan distance atau diagonal distance* akan didapat lintasan terpendek, namun waktu yang dibutuhkan akan bertambah.



Gambar 2. 6 Euclidean Distance dan Manhattan Distance

Gambar 2.4 memperlihatkan pendekatan *Euclidean distance* dan *Manhattan distance*. Merah, biru dan kuning mempunyai nilai yang sama yaitu 12 merupakan hasil *Manhattan distance* untuk rute yang sama dengan *Euclidean distance*, warna hijau yang mempunyai panjang $6 \times \sqrt{2} \approx 8{,}48$ yang lebih kecil dari *Manhattan distance*.

2.5. 12 *Collision Detection*

Collision detection adalah komponen penting pada sebuah *game*. Disadari atau tidak, komponen ini sering kali membuat *game* menjadi realistik, ada beberapa teknik *collision detection*, mulai dari *bounding box*, *reduce bounding box, multi bounding box, circle*, sampai penggunaan sudut dalam pendeteksian tabrakan.

Bounding box, meski memiliki algoritma yang super cepat, dan metode penyimpanan data yang tidak terlalu susah, namun *bounding box* memiliki kelemahan yang pokok, yakni menganggap area kosong sebagai *solid area*. hal ini memberikan efek kesalahan deteksi *collision*. Kesalahan ini dikurangi dengan menggunakan *reduce bounding box*, namun sayang penggunaan *reduce bounding box* juga masih menyebabkan permasalahan yang hampir sama.

Penggunaan multi *bounding box* dirasa cukup bagus, di mana setiap area tubuh diberikan *bounding box*. seperti tangan, kaki, kepala, dan badan diberikan *bounding box*-nya masing masing, cara ini pasti lebih lama dari *bounding box*, namun dirasa cukup mengurangi area kosong yang dianggap objek.

Cara lain selain *bounding box* adalah penggunaan lingkaran, penggunaan lingkaran memiliki kelebihan *bound* memiliki jarak yang sama dengan pusat, sehingga dengan memanfaatkan sifat ini dapat dibuat algoritma yang lebih cepat dari deteksi

bounding box, namun sayang algoritma ini masih kurang bagus dalam mengatasi area kosong.

Penggunaan cara lain adalah dengan pendeteksian garis yang berpotongan, dan penggunaan sudut. untuk cara yang ini, objek akan pertama kali di *list* area *edge* terluar, untuk membatasi area badan, dan dunia luar.

2. 6 Tools

Pada sub bab ini akan dijelaskan tools yang digunakan dalam pembangunan game strategi gunung meletus ini.

2.6. 1 Diagram Konteks

Diagram konteks adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem [18]. Diagram konteks merupakan level tertinggi dari DFD yang menggambarkan seluruh input ke sistem atau output dari sistem. Diagram konteks merupakan arus data yang berfungsi untuk menggambarkan keterkaitan aliran-aliran data antar sistem dengan bagian luar (kesatuan luar). Kesatuan luar ini merupakan sumber arus data atau tujuan data yang berhubungan dengan sistem informasi tersebut.

2.6. 2 Data Flow Diagram (DFD)

Data Flow Diagram (DFD) adalah alat pembuatan model yang memungkinkan profesional sistem untuk menggambarkan sistem sebagai suatu jaringan proses fungsional yang dihubungkan satu sama lain dengan alur data, baik secara manual maupun komputerisasi [19]. DFD ini sering disebut juga dengan nama *Bubble chart*, *Bubble* diagram, model proses, diagram alur kerja, atau model fungsi. DFD ini adalah salah satu alat pembuatan model yang sering digunakan, khususnya bila fungsi-fungsi sistem merupakan bagian yang lebih penting dan kompleks dari pada data yang dimanipulasi oleh sistem. Dengan kata lain, DFD adalah alat pembuatan model yang memberikan penekanan hanya pada fungsi sistem. DFD ini merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk

penggambaran analisa maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program.

Simbol-simbol yang digunakan dalam Data Flow Diagram menurut notasi Yourdan adalah sebagai berikut :

1. Proses

   Proses adalah simbol pertama data flow diagram. Proses dilambangkan dengan lingkaran, dimana proses ini menunjukan bagian dari sistem yang mengubah satu atau lebih input dan output. Nama proses dituliskan dengan satu kata, singkatan atau kalimat sederhana.

2. Aliran Data

   Aliran Data digambarkan dengan tanda panah. Aliran data juga digunakan untuk menunjukan bagian-bagian informasi dari satu bagian ke bagian lain. Pembagian nama untuk aliran ini menunjukan sebuah arti untuk sebuah aliran. Untuk kebanyakan sistem yang dibuat, aliran data sebenarnya mengambarkan data yakni angka, huruf, pesan, floating point, dan macammacam informasi lainnya.

3. Simpanan Data

   Simpanan data digunakan sebagai penyimpanan bagi paket - paket data. Notasi penyimpanan data digambarkan dengan garis horizontal yang pararel. Simpanan data merupakan simpanan data dari data yang berupa suatu file atau database di sistem komputer ataupun berupa arsip atau catatan manual. Nama dari simpanan data menunjukan nama filenya.

4. Terminator

   Terminator digambarkan dengan sebuah kotak yang menggambarkan kesatuan luar (eksternal entitty) yang berhungan dengan sistem. Kesatuan luar merupakan kesatuan (entity) dilingkungan luar sistem yang dapat berupa orang, Organisasi

atau sistem lainnya yan g berada di lingkungan luarnya yang akan memberikan input atau output dari sistem.

2.6. 3 Adobe Flash

Adobe Flash dahulu bernama Macromedia Flash adalah salah satu perangkat lunak komputer yang merupakan produk unggulan Adobe Systems. Adobe Flash digunakan untuk membuat gambar *vektor* maupun animasi gambar tersebut. Berkas yang dihasilkan dari perangkat lunak ini mempunyai *fileextension* .swf dan dapat diputar di penjelajah web yang telah dipasangi Adobe Flash Player. Flash menggunakan bahasa pemrograman bernama *Action Script* yang muncul pertama kalinya pada Flash 5.

Sebelum tahun 2005, Flash dirilis oleh Macromedia. Flash 1.0 diluncurkan pada tahun 1996 setelah Macromedia membeli program animasi *vektor* bernama FutureSplash. Versi terakhir yang diluncurkan di pasaran dengan menggunakan nama 'Macromedia' adalah Macromedia Flash 8. Pada tanggal 3 Desember 2005 Adobe Systems mengakuisisi Macromedia dan seluruh produknya, sehingga nama Macromedia Flash berubah menjadi Adobe Flash.

Adobe Flash merupakan sebuah program yang didesain khusus oleh Adobe dan program aplikasi standar *authoringtoolprofessional* yang digunakan untuk membuat animasi dan bitmap yang sangat menarik untuk keperluan pembangunan situs web yang interaktif dan dinamis. Flash didesain dengan kemampuan untuk membuat animasi 2 dimensi yang handal dan ringan sehingga flash banyak digunakan untuk membangun dan memberikan efek animasi pada website, CD Interaktif dan yang lainnya. Selain itu aplikasi ini juga dapat digunakan untuk membuat animasi logo, *movie*, *game*, pembuatan navigasi pada situs web, tombol animasi, banner, menu interaktif, interaktif form isian, *e-card, screensaver* dan pembuatan aplikasi-aplikasi web lainnya.

Dalam Flash, terdapat teknik-teknik membuat animasi, fasilitas *actionscript, filter, customeasing* dan dapat memasukkan *video* lengkap dengan fasilitas *playbackFLV*. Keunggulan yang dimiliki oleh Flash ini adalah ia mampu diberikan sedikit code pemograman baik yang berjalan sendiri untuk mengatur animasi yang ada didalamnya atau digunakan untuk berkomunikasi dengan program lain seperti HTML, PHP, dan

Database dengan pendekatan XML, dapat dikolaborasikan dengan web, karena mempunyai keunggulan antara lain kecil dalam ukuran file outputnya.

Movie-movie Flash memiliki ukuran *file* yang kecil dan dapat ditampilkan dengan ukuran layar yang dapat disesuaikan dengan keingginan. Aplikasi Flash merupakan sebuah standar aplikasi industri perancangan animasi web dengan peningkatan pengaturan dan perluasan kemampuan integrasi yang lebih baik. Banyak fitur-fitur baru dalam Flash yang dapat meningkatkan kreativitas dalam pembuatan isi media yang kaya dengan memanfaatkan kemampuan aplikasi tersebut secara maksimal. Fitur-fitur baru ini membantu kita lebih memusatkan perhatian pada desain yang dibuat secara cepat, bukannya memusatkan pada cara kerja dan penggunaan aplikasi tersebut.

Flash juga dapat digunakan untuk mengembangkan secara cepat aplikasi-aplikasi web yang kaya dengan pembuatan *script* tingkat lanjut. Di dalam aplikasinya juga tersedia sebuah alat untuk men-*debugscript*, dengan menggunakan *codehint* untuk mempermudah dan mempercepat pembuatan dan pengembangan isi *Action Script* secara otomatis [20].

2.6. 4 Action Script

Salah satu kelebihan *Adobe Flash Professional CS5* dibanding perangkat lunak animasi yang lain yaitu adanya *Action Script. Action Script* adalah bahasa pemrograman Adobe Flash yang digunakan untuk membuat animasi atau interaksi. *ActionScript* mengizinkan untuk membuat intruksi berorientasi *action*(lakukan perintah) dan instruksi berorientasi *logic* (analisis masalah sebelum melakukan perintah).

Sama dengan bahasa pemrograman yang lain, *Action Script* berisi banyak elemen yang berbeda serta strukturnya sendiri, kita harus merangkainya dengan benar agar *Action Script* dapat menjalankan dokumen sesuai dengan keinginan. Jika tidak merangkai semuanya dengan benar, maka hasil yang didapat kan akan berbeda atau *file* flash tidak akan bekerja sama sekali. *Action Script* juga dapat diterapkan untuk *action* pada *frame*, tombol, *movieclip*, dan lain-lain. *Actionframe* adalah *action* yang diterapkan pada *frame* untuk mengontrol navigasi *movie*, *frame*, atau objek lain-lain [20].

Salah satu fungsi *Action Script* adalah memberikan sebuah konektivitas terhadap sebuah objek, yaitu dengan menuliskan perintah-perintah di dalamnya. Tiga hal yang harus diperhatikan dalam *Action Script* yaitu [20]:

1. *Event*

   Event merupakan peristiwa atau kejadian untuk mendapatkan aksi sebuahobjek. *Event* pada *Adobe Flash Professional CS5* ada empat, yaitu:

   a. *Mouse Event*

      Event yang berkaitan dengan penggunaan *mouse*

   b. Keyboard Event

      Kejadian pada saat menekan tombol *keyboard.*

   c. Frame event

      Event yang diletakan apda *keyframe.*

   d. Movie Clip event

      Event yang disertakan pada *movie clip*

2. Target

   Target adalah objek yang dikenai aksi atau perintah. Sebelum dikenai aksi atau perintah, sebuah objek harus dikonversi menjadi sebuah simbol dan memiliki nama instan. Penulisan nama target pada skrip harus menggunakan tanda petik ganda (” ”).

3. *Action*

   Pemberian *action* merupakan lagkah terakhir dalam pembuatan interaksi antar objek. *Action* dibagi menjadi dua antara lain :

   a. *Action Frame* : adalah *action* yang diberikan pada *keyframe*. Sebuah *keyframe* akan ditandai dengan huruf a bila pada *keyframe* tersebut terdapat sebuah *action*

b. *Action Objek* : adalah *action* yang diberikan pada sebuah objek, baik berupa tombol maupun *movie clip*.

2.6. 5 Adobe Photoshop

Adobe Photoshop, atau biasa disebut Photoshop adalah perangkat lunak editor citra buatan Adobe System yang dikhususkan untuk pengeditan foto/gambar dan pembuatan efek. Perangkat lunak ini banyak digunakan oleh fotografer digital dan perusahaan iklan sehingga dianggap sebagai pemimpin pasar (*marketleader*) untuk perangkat lunak pengolah gambar/foto, dan bersama Adobe Acrobat dianggap sebagai produk terbaik yang pernah diproduksi oleh Adobe System. Versi kedelapan aplikasi ini disebut dengan nama Photoshop CS (Creative Suite), versi kesembilan disebut Adobe Photoshop CS2, versi kesepuluh disebut Adobe Photoshop CS3, versi kesebelas disebut Adobe Photoshop CS4, versi keduabelas disebut Adobe Photoshop CS5, dan versi yang terakhir (ketigabelas) adalah Adobe Photoshop CS6.

Photoshop tersedia untuk Microsoft Windows, Mac OS X, dan Mac OS, versi 9 ke atas juga dapat digunakan oleh sistem operasi lain seperti Linux dengan bantuan perangkat lunak tertentu seperti *CrossOver* [21].

2.6.5. 1 Format File

Photoshop memiliki kemampuan untuk membaca dan menulis gambar berformat *raster* dan *vektor* seperti *.png*, *.gif*, *.jpeg*, dan lain-lain. Photoshop juga memiliki beberapa *formatfile* khas [22]:

1. **PSD** (*Photoshop Document*) format yang menyimpan gambar dalam bentuk *layer*, termasuk teks, *mask*, *opacity*, *blend mode*, channel warna, channel alpha, *clipping paths*, dan *settingduotone*. Kepopuleran photoshop membuat format file ini digunakan secara luas, sehingga memaksa programer program penyunting gambar lainnya menambahkan kemampuan untuk membaca format PSD dalam perangkat lunak mereka.
2. **PSB'** adalah versi terbaru dari PSD yang didesain untuk *file* yang berukuran lebih dari 2 GB.

3. **PDD** adalah versi lain dari PSD yang hanya dapat mendukung fitur perangkat lunak PhotshopDeluxe.

2.6. 6 Logic Pro

Logic pro adalah sebuah digital audio workstation (DAW) yang artinya sistem elektronik yang dirancang semata-mata untuk keperluan merekam, mengedit, dan memutar ulang audio digital yang berplatform Mac OS X. Awalnya dibuat oleh pengembang Emagic Jerman, kemudian pada tahun 2002 dibeli Apple dan menjadi bagian dari Apple Logic Studio bundle, aplikasi musik profesional. Logic pro menyediakan instrument perangkat lunak, synthesizer, efek audio, dan fasiitas perekam untuk sintesis musik [23].

Pada pertengahan 1980-an, *Gerhard Lengeling* dan *Chris Adam* mengembangkan Midi Sequencer program untuk Atari ST platform yang di panggil sebagai *Sang Pencipta*. Ketika kemampuan notasi musik ditambahkan, ini menjadi *Notator*, dan kemudian *Notator SL*. Untuk kesederhanaan ketiga secara kolektif disebut sebagai Notator.

Versi awal Logic pro adalah versi 5 yang terakhir dirilis untuk windows, dari versi 6 dan seterusnya logic pro hanya tersedia untuk sistem Apple Mac. Pada bulan maret 2004 Apple merilis Logic Pro 6, kemudian selanjutanya berkembang hingga sampai ke versi 9 yang dirilis pada tanggal 23 juli 2009.

2.6. 7 Propellerhead Reason 5

Reason adalah sebuah perekam musik yang terintegrasi dan studio produksi dengan track audio yang tak terbatas yang menyertakan koleksi suara yang banyak. *Reason* dilengkapi dengan instrumen, efek dan alat pencampuran yang dibutuhkan untuk menulis, merekam, remixing dan memproduksi lagu [24]. Dengan bank yang tersedia dan aliran intuitif, *Reason* membantu anda menjadi lebih kreatif dalam membuat lagu. *Reason* adalah pembuat musik perangkat lunak yang membuat anda tetap fokus pada musik dan memungkinkan anda untuk menyelesaikan lagu itu. Jika anda membutuhkan lebih banyak pilihan kreatif, Rack Extensions memperluas alasan dan memungkinkan anda menambahkan lebih banyak instrumen dan efek, diciptakan oleh Propellerhead dan lain-lain.