# BAB 2

# TINJAUAN PUSTAKA

## 2.1 Hutan

Hutan menurut undang-undang nomor 41 tahun 1999 adalah suatu kawasan ekosistem berupa hamparan lahan berisi sumber daya alam hayati yang didominasi pepohonan dalam persekutuan alam lingkungan, yang satu dengan lainnya tidak dapat dipisahkan. Perkiraan tidak resmi dari bank dunia menyatakan bahwa setiap tahunnya Indonesia kehilangan 1,5 juta Ha hutan selama 12 tahun terakhir dan apabila ini dibiarkan maka akan terjadi ketidakstabilan/ketidakseimbangan lingkungan yang akan berdampak buruk bagi kehidupan manusia [1].

## 2.2 Polisi Hutan

Polisi Hutan adalah pejabat fungsional tertentu dilingkungan Departeman Kehutanan yang oleh atau atas kuasa undang-undang memiliki wewenang kepolisian terbatas dibidang kehutanan. Polisi Hutan bertugas dalam menyelenggaraan perlindungan hutan dan konservasi alam bertujuan untuk menjaga hutan, kawasan hutan dan lingkungannya, agar fungsi lindung, fungsi konservasi, dan fungsi produksi, dapat tercapai secara optimal dan lestari [1].

## 2.3 Game

*Video game* merupakan salah satu media hiburan yang paling popular untuk semua kalangan usia. Sejak pertama kali ditemukan sampai saat sekarang, teknologi *game* telah mengalami kemajuan yang terbilang pesat. Hal ini dapat dilihat dengan berkembangnya jenis, produk, alat, dan jenis interaksi *game* dengan penggunaan yang semakin beragam.

### 2.3.1 Pengertian Game

Dalam kamus bahasa Indonesia *game* diartikan sebagai permainan. Permainan merupakan bagian dari bermain dan bermain juga bagian dari permainan keduanya saling berhubungan. Permainan adalah kegiatan yang kompleks yang didalamnya terdapat peraturan, permainan dan budaya. Sebuah permainan adalah sebuah sistem dimana pemain terlibat dalam konflik buatan, disini pemain berinteraksi dengan sistem dan konflik dalam permainan merupakan rekayasa atau buatan, dalam permainan terdapat peraturan yang bertujuan untuk membatasi perilaku pemain dan menentukan permainan. *Game* bertujuan untuk menghibur, biasanya *game* banyak disukai oleh anak – anak hingga orang dewasa. *Game* sebenarnya penting untuk perkembangan otak, untuk meningkatkan konsentrasi dan melatih untuk memecahkan masalah dengan tepat dan cepat karena dalam *game* terdapat berbagai konflik atau masalah yang menuntut kita untuk menyelesaikannya dengan cepat dan tepat. Tetapi *game* juga bisa merugikan karena apabila kita sudah kecanduan *game* kita akan lupa waktu dan akan mengganggu kegiatan atau aktifitas yang sedang kita lakukan.

*Game* berasal dari kata bahasa Inggris yang berarti dasar permainan. Permainan dalam hal ini merujuk pada pengertian kelincahan intelektual *game* yang akan dibuat ( *Intellectual Playability Game*) yang juga bisa diartikan sebagai arena keputusan dan aksi pemainnya. Dalam *game* ada target-target yang ingin dicapai pemainnya.

Berdasarkan representasi *visual*, *game* dapat dibedakan menjadi 2 jenis, yaitu game 2 dimensi (2D) dan 3 dimensi (3D). Game 2D adalah *game*, yang secara matematis, hanya melibatkan 2 elemen koordinat kartesius, yaitu *x* dan *y*, sehingga konsep kamera pada *game* 2D hanya menentukan “gambar” mana pada *game* yang dapat dilihat oleh pemain. Sedangkan *game* 3D adalah *game* yang selain melibatkan elemen *x* dan *y*, juga melibatkan elemen *z* pada perhitungannya, sehingga konsep kamera pada *game* 3D benar-benar menyerupai konsep kamera

pada kehidupan nyata, yaitu selain digeser (seperti pada *game* 2D), juga dapat diputar dengan sumbu tertentu [4].

### 2.3.2 Sejarah Singkat Game

Di antara jajaran nama tokoh penemuan penting, nama besar seorang penemu permainan komputer, atau lebih lazim kita menyebutnya dengan *game*, jarang sekali disebutkan. Namun bila penemuannya sampai sekarang masih digunakan dan bahkan sangat diminati dan dinikmati. Sebagian orang bahkan saat ini mengangap *game* sebagai sebuah kebutuhan yang harus terpenuhi.

*Game* pertama kali dibuat pada tahun 1966 oleh Ralph Baer bersama dengan timnya yang berjumlah 500 orang yang terdiri dari para insinyur dan teknisi. Permainan yang pertama sekali ditemukan hanya dapat dimainkan oleh komputer seharga US$ 40.000. Dimana pengembangan dan pembuatan *game* tersebut didanai oleh Pentagon.

Pada tahun 1965, pihak militer datang kepada Bear dan meminta menciptakan simulasi komputer yang dapat membantu pasukan untuk belajar strategi dan mengukur kemampuan reflek. Proyek ini dikerjakan dengan tingkat keamanan yang sangat ketat di tengah situasi perang dingin.

Setelah sebulan bekerja keras, Bear berhasil menampilkan dua titik putih yang berkejar-kejaran di layar. Hal ini membuat pihak militer merasa kagum dan memberikan dana yang lebih besar lagi sehingga ia dapat menyewa asisten lebih banyak. Tim ini kemudian berhasil membuat permainan antara papan dan bola yang pada akhir tahun 1966 dipresentasikan di depan para pejabat Pentagon. Namun sayang, ternyata mereka tak merasa tertarik dengan penemuan Bear. Hingga akhirnya ia berusaha memperoleh izin agar dapat memproduksi mesin permainan secara komersial.

Tahun 1970, Bill Enders yang tergabung dalam sebuah organisasi dengan nama Megnavox mencoba menyakinkan edukatif Megnavox untuk memberikan

kesempatan pada Bear dan mesin permainannya. Hasilnya adalah munculnya *video game* komersial pertama, yaitu Magnavox Odyssey.

Tahun 1972, muncul nama baru dalam dunia *game*. Nolan Bushnel. Pada tanggal 27 Juni 1972, ia mendirikan perusahaan Atari, dan membuat *game arcade* Pong. Era baru dalam perkembangan dunia *game* terjadi pada tahun 1988, yang di dominasi oleh perusahaan di Jepang. Nintendo, yang awalnya hanya memproduksi mesin fotocopi, beralih memproduksi *game*.

Saat ini permainan komputer telah berkembang sedemikian pesatnya seiring dengan perkembangan *hardware* yang mendukung. Dengan program permainan yang lebih kompleks dan tampilan grafis 3D yang luar biasa [4].

### 2.3.3 Game Komputer

Pengertian dari *game* komputer adalah sebuah interaktif yang dijalankan menggunakan komputer. Program komputer menerima *input* dari si pemain melalui pengendalian dan menampilkan lingkungan buatan melalui TV atau layar monitor.

Sejarah teknologi *game* komputer secara langsung berhubungan dengan perkembangan komputer itu sendiri. Komputer dengan kecepatan *processor* tinggi, grafis yang lebih mendekati realita, dan media penyimpanan yang lebih besar sebenarnya dimaksudkan untuk memenuhi kebutuhan dalam bermain *game*.

### 2.3.4 Jenis – Jenis Game

a. *Arcade Games*

Jenis *game* yang sering disebut ding-dong di Indonesia, biasanya berada di daerah / tempat khusus dan memiliki box atau mesin yang memang khusus di rancang untuk jenis *video games* tertentu dan tidak jarang bahkan memiliki fitur yang dapat membuat pemainnya lebih merasa masuk dan menikmati, seperti

pistol, kursi khusus, sensor gerakan, sensor injakan dan stir mobil (beserta transmisinya tentunya).

b. *PC Games*

*Game* yang dimainkan pada PC (*Personal Computer*) yang memiliki kelebihan yaitu memiliki tampilan antarmuka yang baik untuk input maupun *output, output visual* kualitas tinggi karena layar komputer biasanya memiliki resolusi yang jauh lebih tinggi dibandingkan dengan layar televisi biasa.

c. *Console Games*

Jenis *video game* yang dimainkan menggunakan *console* tertentu, seperti Playstation 2, Playstation 3, XBOX 360, dan Nintendo Wii.

d. *Handheld Games*

Jenis *game* yang dimainkan di *console* khusus *video game* yang dapat dibawa kemana-mana, contoh Nintendo DS dan Sony PSP.

e. *Mobile Games*

Jenis *game* yang dapat dimainkan atau khusus untuk *mobile phone* atau PDA.

**2.3.5 Genre - Genre Game**

a. *Action Game*

Adalah tipe game dengan fitur utama berupa banyaknya aksi di mana pemain harus memiliki keterampilan reaksi yang cepat untuk menghindari musuh atau menghindari rintangan. Pengembang game tipe ini perlu memastikan game yang dibuat dioptimasi sehingga pemain memiliki pengalaman bermain yang baik, yang tidak terganggu oleh delay proses yang lama, contohnya : Metal Gear Solid.

b. *Adventure Game*

Adalah tipe game yang umumnya membuat pemain harus berjalan mengelilingi suatu tempat yang terkondisi, seperti sebuah istana, gua yang berkelok, dan planet yang jauh. Pemain melakukan navigasi suatu area, mencari pesan-pesan rahasia, memperoleh obyek yang memiliki kemampuan yang bervariasi, bertempur dengan musuh, dan lain-lain. Untuk membuat game ini, diperlukan perencanaan yang akurat sehingga memiliki alur cerita yang menarik bagi pemain, contohnya : Tomb Raider.

c. *Sport Game*

Adalah tipe game yang berupa kompetisi antara dua pemain atau lebih, dimana pemain dapat berupa individual atau tim. Contoh game tipe ini antara lain sepakbola, bola basket, tenis, dan bilyard. Tergantung seberapa cepat permainan yang terjadi, aplikasi game perlu dioptimalkan, contohnya : Pro Evolution Soccer.

d. *RPG (Role Playing Game)*

Adalah tipe game yang seringkali berupa multi karakter game di mana setiap pemain memiliki karakter dengan kemampuan, kekuatan, dan kelemahan yang spesifik. Para pemain saling berkompetisi, berinteraksi, dan bertempur satu sama lain. Tampilan grafis yang khas untuk setiap karakter pemain ditambah dengan storyline yang mendebarkan akan sangat menarik dan memberikan pengalaman yang berbeda di dalam bermain, contohnya : Final Fantasy.

e. *Platform Game*

Adalah tipe game yang mengharuskan pemain mengarahkan suatu obyek dengan melalui berbagai tahap atau tingkatan area untuk menyerang musuh dan menghindar terhadap serangan. Tipe game ini sedikit serupa dengan action game, tetapi aksinya tidak secepat action game. Teknik collision detection sangat sering dimanfaatkan pada tipe game ini, contohnya : Sonic The Hedgehog, dan Mario Bros.

f. *Puzzle Game*

Adalah tipe game yang umumnya membuat pemain menggunakan kemampuan berpikirnya sebagai pengganti keterampilan reaksi yang cepat karena terdapat rahasia yang perlu dipecahkan. Game ini lebih bersifat statis dibanding action game. Pembuatan game tipe ini seringkali ditunjang dengan AI (Artificial Intelligence), contohnya : 7 Wonders of The Ancient World.

g. *Sandbox Game*

Adalah tipe game yang umumnya ditujukan untuk menjelajahi suatu kota atau tempat dan bebas berinteraksi dengan objek disekitarnya. Mungkin game ini bisa di bilang mirip RPG tapi Sandbox sangat berbeda dengan RPG game, perbedaannya adalah tidak adanya peningkatan level karakternya, contohnya : GTA, dan Bully.

h. FPS (*First Person Shooter*)

Adalah tipe game yang mengunakan sudut pandang orang pertama untuk membidik atau membunuh musuh, sehingga kita hanya melihat tangannya saja dan tidak melihat tubuh karakter yang dimainkan, contohnya : Call of Duty.

i. TPS (*Third Person Shooter*)

Adalah tipe game yang mirip dengan FPS, tapi bedanya tipe game ini mengunakan sudut pandang orang ke-3, sehingga kita bisa melihat seluruh tubuh karakter yang dimainkan, contohnya : Gears of Wars.

j. *Music Game*

Adalah tipe game yang menuntut pemainnya untuk menekan tombol sesuai dengan tombol yang ada di layar dengan diiringi dengan musik, contohnya : Guitar Hero, dan Dance Dance Revolution.

k. *Fighting Game*

Adalah tipe game yang intinya harus menjatuhkan lawan tandingnya, entah itu dengan pukulan, tendangan, combo, maupun dengan jurus special, contohnya : Tekken & Mortal Kombat.

l. *Strategy*

Adalah tipe game yang mengharuskan pemainnya menggunakan taktik dan strategi untuk jeli dalam melihat setiap peluang, kelemahan musuh dan bijaksana dalam menggunakan sumber daya yang ada, contohnya : Command & Conquer.

m. *Simulation Game*

Adalah tipe game yang memberikan pengalaman atau interaksi sedekat mungkin dengan keadaan yang aslinya, meskipun terkadang keadaan tersebut masih eksperimen atau bahkan fiktif, tapi ada penekanan khusus pada detail dan pengalaman realistik menggunakan keadaan tersebut, contohnya : Gran Turismo.

n. *Racing Game*

Adalah tipe game yang tujuannya adalah mencapai garis finish dari suatu race, dalam game ini biasanya pemain dapat memilih dan membeli kendaraan, mendandani, melakukan upgrade mesin , contohnya : Need For Speed.

### 2.3.6 *Game Tower Defense*

Permainan *Tower Defense* (TD) merupakan permainan strategi pengaturan *tower* (bangunan, senjata, naga, tanaman, dll) yang bertujuan menghentikan musuh yang akan melintas. *Tower* ini akan menembaki musuh dalam radiusnya. *Tower* memiliki beberapa tipe dan level dengan kemampuan, biaya pembelian, biaya penggunaan, dan biaya upgrade yang berbeda. Untuk setiap musuh yang berhasil dikalahkan, pemain akan mendapatkan uang untuk meng-*upgrade* towers.

Strategi dalam permainan ini mencakup pemilihan *tower* yang akan di-*upgrade*. Permainan ini biasanya memiliki misi untuk meminimalkan jumlah musuh yang lolos atau menjaga sejumlah barang. Pemain dianggap menang jika jumlah musuh yang lolos kurang dari batas yang ditentukan atau barang yang dijaga masih ada.[7]




Gambar 2.1. Contoh *Game Tower Defense*.

## 2.4 *Game Maker*

Aplikasi ini adalah aplikasi yang dapat Anda gunakan untuk membuat beragam macam game untuk diinstall di PC. Mulai dari game berbentuk maze (labirin dengan konsep permainan bak puzzle), tembak-menembak, hingga permainan bertampilan tiga dimensi, dan first person shooter.

Aplikasi ini menjanjikan trik pembuatan game dengan sistem drag-and-drop, sehingga untuk pembuatan game sederhana kita tidak perlu menggunakan bahasa pemrograman. Namun, ketika ingin membuat permainan yang lebih kompleks, kemampuan pemrograman juga diperlukan di sini.

Aplikasi ini juga menyediakan beragam komponen untuk pendukung pembuatan beragam jenis permainan yang dikehendaki. Komponen atau resources ini bahkan ada yang disediakan oleh para pengguna Game Maker lainnya. Anda bisa mengunduh background, musik latar, skrip pemrograman, banner, hingga fasilitas untuk membuat cheat codes di game.[3]

### 2.4.1. Game Maker Studio 1.0

Biasanya dalam hidup, semuanya terbuat dari *objects, objects* baik yang terlihat atau tidak terlihat, dan *objects* melakukan *actions*. *Actions* yang harus dilakukan yang dilakukan pada acara-acara biasa, kesempatan ini disebut peristiwa. Sebuah object dapat membuat tindakan ketika dibuat, sebelum hancur, ketika bertabrakan dengan *object*, ketika tombol ditekan, ketika mouse mengacu di atas *object*, dan banyak lagi.

Tapi di mana akan object melakukan tindakan mereka? Tindakan yang dilakukan di Room. *Room* adalah datar, dan tidak 3D, ini berarti bahwa permainan berlangsung dalam dunia 2 dimensi. Namun ini tidak berarti bahwa permainan tidak dapat memiliki grafis 3D. Di dalam *Room*, kita dapat menempatkan *object*. *Room* ini sangat disesuaikan, Anda dapat mengatur latar belakang *room*, dan Anda dapat memiliki lebih dari satu *room*.

Grafis dalam permainan yang baik latar belakang yang ditetapkan untuk sebuah Room, atau sprite, sprite animasi grafis, yang dapat dilampirkan ke *object*, dan mereka akan mewakili *object*, atau mereka dapat bergantung, dan object dapat memanggil beberapa tindakan tertentu untuk menarik sprite tanpa membuat sprite mewakili object.

Suara dapat ditetapkan sebagai suara latar belakang, suara normal, atau bahkan suara 3D suara yang baik dimainkan oleh suatu benda, atau dalam kode penciptaan room / object.

Game Maker adalah alat yang ampuh yang memiliki banyak tindakan, salah satu tindakan yang menggambar teks. Teks dapat diambil dengan font yang berbeda, dan format. Ketika menggambar teks, pengguna dapat mengatur font untuk teks yang akan ditulis dengan [3].

#### 2.4.1.1 Event

Untuk membuat event click "Add Event" di tengah bawah dari layar-objek. Anda kemudian dapat Pilih jenis event dan event tertentu.

***1. Create Event***

*Create event* adalah peristiwa di mana *actions* dipanggil pada saat objek dibuat (dibuat dalam *room*). Ini berarti bahwa menginterupsi *current event*. Jadi jika membuat sebuah instance dari sebuah objek dan kemudian menetapkan variabel dalam objek yang segera setelah itu (dalam *event* sama), create event untuk *instance* sudah dieksekusi.

***2. Destroy Event***

*Destroy event* adalah peristiwa di mana event dijalankan ketika objek dihancurkan (dihapus dari *room*).

***3. Alarm Event***

*Alarm event* dijalankan ketika variabel alarm yang sesuai global mencapai 0 (misalnya, jika alarm [1] diatur ke 5 dalam hal menciptakan alarm 1 event akan memicu 5 langkah kemudian).

***4. Step Event***

*Step event* dijalankan setiap langkah. Sebuah langkah peristiwa dalam Game Maker terdiri dari tiga sub-peristiwa:

a. Begin Step: Dijalankan sebelum semua peristiwa dijalankan..
b. Normal Step: Dijalankan ketika kasus atau kondisi terpenuhi.
c. End Step: Dijalankan setelah semua peristiwa dijalankan.

### *5. Collision Event*

Dalam *collision event*, *actions* yang dilakukan ketika suatu objek bertabrakan dengan benda tertentu. Ketika mengklik tombol "Collision Event", Anda akan melihat menu yang berisi nama-nama dari semua benda. Anda memilih salah satu objek, dan itu akan menjadi *collision event* dengan benda itu.

Berikut adalah contoh:

Object1 adalah peluru dan object2 adalah musuh. Dalam kode object2 harus dihancurkan ketika bertabrakan dengan object1. Di dalam properti obyek dari object2 (musuh) klik pada tombol *collision event* dan Anda akan melihat daftar semua benda, pilih object1 (peluru), ini berarti bahwa object2 akan melakukan daftar tindakan ketika object1 bertabrakan dengan itu. Selanjutnya Anda dapat menempatkan tindakan yang dibutuhkan dalam kotak di sebelah kiri.

### *6. Keyboard Event*

*Keyboard event* dijalankan ketika tombol *keyboard* ditekan. Tombol bisa berupa pesan, tombol fungsi, nomor, arah, atau tombol yang terletak di keyboard. Jika Anda terus menekan tombol, event akan tetap melaksanakan sampai tombol dilepaskan.

### *7. Key press Event*

*Key press event* dijalankan hanya bila *key* pertama ditekan. Itu berarti bahwa *event* dieksekusi ketika tombol ditekan.

### *8. Key release Event*

*Key release event* dijalankan hanya bila tombol dilepaskan. Itu berarti bahwa *event* dieksekusi ketika tombol dilepaskan.

### *9. Mouse Event*

**Mouse Button Event**

Mouse events memiliki tiga jenis:

a. *Mouse button*: Dipanggil ketika tombol mouse ditekan dan mengacu objek, seperti event keyboard.

b. *Mouse press*: Dipanggil ketika tombol diklik, agar dapat dipanggil lagi, mouse harus dibebaskan dan ditekan lagi.

c. *Mouse release*: Dipanggil ketika mouse dilepaskan.

Note: *mouse button events* terjadi hanya ketika tombol mouse di atas sprite yang berhubungan dengan object.

***Mouse Enter and Leave Events***

*Mouse enter* dan *leave* events dipanggil ketika mouse memasuki atau meninggalkan objek. Yaitu, ketika mouse sprite terkait dengan benda atau berhenti mengacu pada objek.

***Mouse Wheel Events (Up and Down)***

*Mouse wheel events* akan dieksekusi ketika roda mouse menggulir ke atas atau bawah.

***Global Mouse Events***

*Global mouse* mencakup banyak sub-peristiwa yang hampir identik dengan peristiwa tombol mouse, kecuali bahwa mereka dieksekusi terlepas dari lokasi mouse.

***10. Other Events***

*Other event* memiliki banyak sub-peristiwa di dalamnya:

a. *Outside Room*: Dijalankan ketika objek berada di luar ruangan

b. *Intersect Boundary*: Dijalankan ketika objek menyentuh batas ruangan.

c. *Game Start*: Dijalankan ketika pertandingan dimulai.

d. *Game End*: Dijalankan ketika permainan berakhir.

e. *Room Start*: Dijalankan ketika room dimulai.

f. *Room End*: Dijalankan ketika room berakhir.

g. *No More Lives*: Dijalankan ketika pemain tidak memiliki *lives* lagi.

h. *No More Health*: Dijalankan ketika pemain tidak memiliki kesehatan.

i. *Animation End*: Dijalankan ketika sprite yang terkait dengan objek selesai *animating*.

j. *End of path*: Dijalankan ketika objek tersebut berhasih melaksanakan akhir jalurnya.

k. *User defined*: Pengguna didefinisikan.

**2.4.1.2 Actions**

***1. The actions***

***Start moving in fixed direction*** 

*Action* ini akan membuat objek mulai bergerak dalam arah tertentu dalam kecepatan tertentu. Klik pada arah yang Anda inginkan objek untuk pergi ke suatu arah, dan menentukan kecepatan. Tombol di tengah membuat objek diam. Jika Anda mengklik dua tombol yang berbeda, objek akan secara acak memilih satu posisi untuk pergi.

***Move in any direction*** 

*Action* ini akan memungkinkan Anda untuk mengubah arah dan kecepatan gerak benda tanpa kendala dalam arah.

***Move in the direction of a point*** 

*Action* ini akan membiarkan gerakan objek ke arah titik tertentu yang ditetapkan oleh Anda. Benda tersebut akan bergerak dalam kecepatan didefinisikan oleh programmer.

***Set the horizontal speed*** 

*Action* ini menetapkan kecepatan horizontal obyek. Setting ke 0 akan mengakibatkan tidak ada gerakan horisontal. Setting ke 1 + akan menghasilkan bergerak ke kanan. Dan pengaturan ke -1 - akan menghasilkan bergerak ke kiri.

***Set the vertical speed*** 

*Action* ini menetapkan kecepatan vertikal objek. Setting ke 0 akan mengakibatkan tidak ada gerakan vertikal. Setting ke 1 + akan menghasilkan gerakan ke arah bawah. Dan pengaturan ke -1 - akan menghasilkan bergerak ke atas.

***Set the gravity*** 

*Action* ini menetapkan kekuatan gravitasi dan arahnya. Kekuatan gravitasi adalah berapa banyak benda yang menuju ke arah gravitasi. Mengatur arah gravitasi ke 0 berarti arahnya adalah ke kanan, pengaturan untuk 90 berarti arahnya adalah ke atas, pengaturan untuk 180 bergerak ke kiri dan itu berarti bahwa pengaturan ke 270 akan bergerak ke bawah.

***Reverse horizontal direction*** 

Ini akan membalikkan arah horizontal dari objek. Jika arah horisontal adalah ke kiri, menjadi ke kanan.

***Reverse vertical direction*** 

Ini akan membalikkan arah vertikal dari obyek. Jika arah objek itu ke atas, menjadi ke bawah. Menggunakan "arah horisontal Reverse" dan "arah vertikal Reverse" bersama-sama akan membuat objek membalikkan arah sepenuhnya, dan membentuk gerakan reaksi.

***Set the friction*** 

Ini akan mengatur kekuatan gesekan yang mempengaruhi suatu objek dalam setiap langkah (millisecond). Lupakan semua tentang matematika! Gesekan di Game Maker sangat sederhana, jika gesekan suatu benda adalah ".1" kecepatan (horisontal dan vertikal) dikurangi dengan ".1" sampai kecepatan mencapai 0. Perhatikan bahwa jika kecepatan adalah negatif, ".1" akan ditambahkan ke dalamnya.

***Jump to a given position*** 

Tindakan ini akan mengubah posisi objek langsung ke posisi baru yang ditetapkan oleh x dan y.

***Jump to the start position*** 

Tindakan ini akan mengubah posisi objek langsung ke posisi yang mulai. Anda perlu mengatur apa-apa, objek secara otomatis akan memindahkan posisi di mana ia pertama kali diciptakan.

***Jump to a random position*** 

*Object* akan melompat ke arah yang acak. Tapi Anda masih harus mengatur Snap Horisontal dan Vertikal Snap.

***Snap to a grid*** 

Ini akan memindahkan objek ke garis grid terdekat. Lebar dan tinggi dari grid didefinisikan oleh pengguna. Bagi anda yang tidak tahu apa grid adalah, grid adalah semacam jaring, objek yang dapat disesuaikan.

***Wrap around the screen*** 

Dilakukan pada *instance*, jika berjalan di luar ruangan, datang kembali di sisi lain. Anda dapat mengatur apakah hanya melakukannya horizontal, vertikal, atau keduanya

***Move to a contact position*** 

Dengan aksi ini Anda dapat memindahkan contoh dalam arah tertentu sampai posisi kontak dengan obyek tercapai. Jika sudah ada adalah tabrakan pada posisi saat ini contoh tidak bergerak. Jika tidak, misalnya ditempatkan tepat sebelum tabrakan terjadi. Anda dapat menentukan arah tetapi juga jarak maksimal untuk bergerak.

***Bounce against objects*** 

Ini akan membuat bouncing objek (bereaksi) terhadap objek. Ada dua hal yang harus diatur:

a. *Whether or not the bouncing is precise*: Jika Anda mengatur *Precise* ke *false*, objek akan bereaksi terhadap dinding horizontal atau vertikal baik saja. Jika Anda mengatur Precise ke true, objek akan bereaksi terhadap semua benda benar.

b. Bouncing terhadap benda padat saja atau semua benda?

Tindakan ini harus diletakkan dalam *collision event.*

***Create an instance of an object***

Tindakan ini akan membuat objek tertentu dalam posisi tertentu didefinisikan oleh *programmer*.

***Create an instance of an object with a motion***

Tindakan ini akan membuat objek tertentu dalam posisi tertentu, namun objek yang akan dibuat dengan gerak. Itu berarti Anda harus mengatur kecepatan dan arah dari objek yang baru dibuat. Arah adalah 0-360, 0: Kanan, 90: Up, 180: Kiri, 270: Bawah.

***Change the instance***

Tindakan ini akan mengubah objek ke objek lain. Anda perlu mengatur objek dan apakah atau tidak untuk melakukan merusak acara untuk objek lama dan menciptakan acara untuk objek baru.

***Destroy the instance***

Tindakan ini akan menghancurkan *object.*

***Destroy instances at position***

Ini akan menghancurkan semua contoh (*objects*) yang terletak di posisi tertentu didefinisikan oleh *programmer*.

***Change the sprite***

Ini akan mengubah sprite benda (atau sprite yang mewakili objek). Anda harus mengatur sprite, gambar sub (-1 untuk semua), dan kecepatan gambar (1 = normal).

***Transform the sprite***

Tindakan ini akan mengubah bentuk sprite. Anda harus mengubah "XScale" dan ini adalah jumlah skala untuk lebar sprite itu. Jika "XScale" adalah 1, lebar sprite itu tidak akan mengubah ukurannya. Kemudian Anda harus mengatur "yscale". Kemudian sudut gambar (rotasi). Jika sudut gambar adalah 0, sprite tidak akan diputar. Akhirnya, Anda akan diminta untuk memilih cermin, apakah Anda ingin sprite untuk memiliki mirroring tidak, cermin horisontal, vertikal, atau keduanya. Di bawah ini, ada meja menggambarkan x dan y skala.

Tabel 2.1 *Transform the sprite*

| **Scale** | **Width/Height change** | **Mirrored** |
|---|---|---|
| Less than -1 | Larger | Yes |
| -1 | Normal | Yes |
| 0 to -1 | Smaller | Yes |
| 0 | None | No |
| 0 to 1 | Smaller | No |
| 1 | Normal | No |
| More than 1 | Larger | No |

***Set sprite blending***

Ini akan mengatur pencampuran warna / alpha dari sprite objek. Pencampuran adalah apa warna objek harus memudar ke. Putih adalah campuran normal untuk sebuah objek. The alpha gambar transparansi objek. Jika alpha adalah 1 objek akan terlihat. Jika alpha adalah 0 objek akan terlihat. Jika itu .5 objek akan semitransparan.

***Play a sound***

Tindakan ini juga akan memainkan suara sekali, atau loop suara.

***Stop a sound***

Tindakan ini juga akan menghentikan suara yang tidak selesai diputar, atau akan menghentikan suara looping.

***Set an alarm***

Tindakan ini menetapkan alarm khusus untuk jumlah tertentu milidetik. Untuk mempelajari lebih lanjut tentang peristiwa alarm lihat dokumentasi sebelumnya.

***Sleep for a while***

Tindakan ini akan menghentikan permainan untuk sejumlah tertentu milidetik. Anda harus memeriksa apakah akan atau tidak untuk redraw layar.

***Set a timeline***

Tindakan ini akan menetapkan garis waktu. Posisi adalah dari ketika timeline akan dimulai. Jika Anda memilih 0, garis waktu akan mulai dari awal.

***Set timeline position***

Tindakan ini akan mengatur posisi batas waktu tertentu. Posisi garis waktu dibahas sebelumnya.

***Display a message***

Tindakan ini akan menampilkan message.

***Show the game info***

Tindakan ini akan menampilkan informasi permainan. Informasi permainan di mana Anda menulis bantuan permainan dan komentar lainnya.

***Show a video***

Tindakan ini akan menampilkan video dalam permainan.

***Restart the game***

Tindakan ini akan me-restart permainan.

***End the game***

Tindakan ini akan berakhir / berhenti permainan.

***Save the game***

Tindakan ini akan menyimpan semua perubahan permainan ke dalam satu set file tertentu oleh programmer.

***Load the game***

Tindakan ini akan memuat semua perubahan permainan diselamatkan dari save sebelumnya dari file tertentu yang ditetapkan oleh programmer.

***Replace a sprite from a file***

Tindakan ini akan mengubah sprite. Penggantian sprite akan diimpor dari file. Anda harus mengatur sprite yang akan diubah, dan nama dari file (.gif / .bmp). Dan kemudian Anda harus memasukkan jumlah gambar (gambar sub sebenarnya) bahwa pengguna harus mengganti (1 untuk gambar diam).

***Replace a sound from a file***

Tindakan ini akan menggantikan suara dari file.

***Replace a background from a file***

Tindakan ini akan menggantikan latar belakang dari file.

***Call the inherited event***

Ini adalah tindakan yang entah bagaimana kompleks yang akan dipahami nanti.

***Call a script***

Tindakan ini akan menjalankan script dengan semua argumen.

***Put some comment***

Tindakan ini tidak ada tindakan. Itu hanya akan menempatkan beberapa komentar sehingga Anda dapat mengatur pikiran Anda. Sebuah komentar dalam tindakan Drag & Drop setara dengan frase dikelilingi oleh / * dan * / atau satu baris kode yang memiliki / / di awal (teks dalam komentar / / harus menempati satu baris saja).

### *2. Introduction to If Conditions and blocks*

Sampai saat ini, Anda mungkin berpikir bahwa Anda bisa membuat permainan profesional, tetapi kenyataannya adalah sulit untuk melakukan itu sekarang. Anda perlu memeriksa kondisi yang lebih kompleks. Jadi permainan akan membuat satu hal jika ekspresi adalah benar. Ini adalah Laporan Jika (dan kadang-kadang disebut Jika Kondisi). Jika Anda ingin melakukan satu tindakan jika ekspresi benar, Anda akan menempatkan tindakan setelah tindakan kondisi Jika. Anda dapat melakukannya menggunakan "Jika Pernyataan" dengan cara yang sama seperti di bawah ini:

*If Condition – If something is true,* variabel sama dengan nilai misalnya (lihat di bawah untuk info tentang variabel). *Action – if the statement above is true,* tindakan ini dilakukan.

Namun, jika Anda ingin melakukan banyak tindakan jika ekspresi tertentu adalah benar, itu tidak akan mudah untuk mengulang hal yang sama Jika kondisi beberapa kali untuk hasil tersebut. Sebaliknya Anda dapat menempatkan " Start of

block " dan " End of block " tindakan. Dan Anda yang sederhana bisa membuat beberapa tindakan:

If Condition

Start of block

Action

Action

Action

End of block

Dengan cara ini Anda dapat membuat game profesional yang banyak dengan Game Maker. Namun, apa yang akan Anda lakukan jika Anda ingin tindakan untuk melakukan sesuatu jika ekspresi adalah benar, dan sesuatu yang lain jika ekspresi tidak benar? Ada dua cara, baik mengulangi hal yang sama jika kondisi dengan kotak centang " Not " diperiksa, dan ini akan mengevaluasi kebalikan dari ekspresi. Namun tindakan yang lebih baik untuk menggunakan adalah " Else " tindakan, di mana ia benar-benar bekerja seperti Kondisi Jika tapi harus ditempatkan setelah Jika kondisi ini berakhir. Itu baik setelah akhir blok Jika Kondisi, atau setelah tindakan seperti yang ditunjukkan dalam contoh pertama

If Condition

Start of block

Action

Action

Action

Action

End of block

Else

Start of block

Action

Action

End of block

***If a sound is playing***

Merupakan kondisi Jika yang memeriksa apakah suara tertentu bermain atau tidak.

***If previous room exists***

Tindakan ini merupakan Jika Kondisi yang memeriksa apakah atau tidak ada ruang sebelumnya.

***If next room exists***

Tindakan ini merupakan Jika Kondisi yang memeriksa apakah ada atau tidak ada kamar sebelah.

***If a position is collision free***

Sebuah Kondisi Jika yang memeriksa apakah atau tidak posisi tertentu mengandung obyek di dalamnya. (Ia melakukan tindakan jika posisi tidak mengandung objects).

***If there is a collision at position***

Sebuah Jika Kondisi yang memeriksa apakah atau tidak bertabrakan posisi tertentu dengan benda apapun. (Lakukan tindakan jika tidak).

***If there is an object at position***

Sebuah Jika Kondisi yang memeriksa apakah atau tidak objek tertentu terletak di posisi tertentu.

***If the number of instances is a value***

Sebuah Jika Kondisi yang memeriksa jumlah benda-benda tertentu di dalam ruangan. Jika jumlah contoh adalah sama dengan jumlah yang ditentukan oleh pengguna, tindakan akan dilakukan.

***With a chance perform the next action***

Tindakan ini melakukan tindakan dengan kesempatan pihak-pihak tertentu.

***If the user answers yes to a question***

Kondisi ini Jika akan menampilkan pesan yang meminta suatu "Ya / Tidak" pertanyaan. Jika pemain memilih ya, tindakan untuk jika pernyataan akan dilakukan.

***If an expression is true***

Ini adalah Kondisi Jika yang memeriksa apakah pernyataan tertentu adalah benar atau tidak. Lebih belajar tentang pernyataan diperlukan sebelum menggunakan tindakan ini benar.

***If a mouse button is pressed***

Jika tombol mouse ditekan, tindakan akan dilakukan.

***If an instance is aligned with a grid***

Ini akan melakukan tindakan jika contoh tertentu sejajar (snapped) ke grid.

***Exit this event***

Tindakan ini akan menghentikan pelaksanaan event.

***3. Loops***

***Repeat next action***

Tindakan ini akan mengulangi tindakan berikutnya beberapa kali didefinisikan oleh pengguna. Anda dapat menggunakan blok dalam aksi ini.

***4. Introduction to variables***

Variabel adalah nilai-nilai yang dapat berubah. Seperti "*fast*", cepat adalah variabel, cepat dapat menjadi 0, 1, 2, 3, atau apa pun sama sekali. Sementara jumlah "5" hanya 5! Sebuah nama variabel bisa apa saja, tapi tidak boleh berisi:

| | | | | |
|---|---|---|---|---|
| . | ! | @ | # | $ |
| % | ^ | & | * | ( |
| ) | - | = | + | \ |
| < | > | / | ? | <space> |
| { | } | [ | ] | , |
| \| | ` | ~ | " | ' |

Untuk mempermudah, nama variabel hanya dapat berisi huruf dan angka dan garis bawah (_). Hal ini dapat dimulai dengan huruf, dan garis bawah, tetapi tidak dapat dimulai dengan angka.

Menetapkan variabel mudah, Anda menggunakan drag dan tindakan drop, yang cukup sederhana atau menggunakan bentuk kode:

```
//setting a variable to 1

variable=1;
```

Variabel dalam Game Maker yaitu "*real*" atau "*strings*".

***Real***

Nilai real untuk variabel angka. 1, 2, 3, atau 10000 semua nilai riil. Saat membuat nilai Anda cukup memasukkan nilai.

***String***

Strings adalah jenis besar nilai pada variabel. "Halo" bisa berupa string "123 ABC" adalah string. Sebuah string adalah jenis teks yang dikelilingi oleh kutipan ("s) atau ('s). Ketika menetapkan nilai variabel ke string, Anda mengelilingi nilai dengan tanda kutipan.

***5. Variables: Global & Local***

Misalkan Anda memiliki dua objek, variabel satu objek (bahagia) diatur ke 1, bahagia variabel objek lain diatur ke 0. Dengan cara ini setiap objek memiliki bahagia sendiri. Tapi bagaimana jika Anda ingin semua benda memiliki nilai yang sama? Dengan cara ini Anda dapat menggunakan variabel global. Tujuan pertama adalah obyek yang menginisialisasi segalanya, ia meminta pemain untuk ya / tidak pertanyaan dan variabel set. misalnya:

If the user answers yes to a question: "Are you happy"

Start of a block

Set a variable: happy to 1

End of a block

Else

Start of a block

Set a variable: happy to 0

End of a block

Dengan cara ini menginisialisasi objek akan mengenali "bodoh" variabel, tetapi benda-benda lainnya wont. Ini adalah ketika Anda menggunakan: global. variabel. The terletak antara global dan nama variabel (.) Memisahkan jenis variabel dari nama variabel. Jika itu tidak ada (.) Di tempat pertama, maka akan dianggap sebagai variabel lokal. Berikut adalah penjelasan untuk variabel global:

***Global Variables***

*Global variables* adalah variabel yang tidak terbatas pada objek tertentu atau kamar. Variabel global selalu ada. Jika Anda mengatur global.stupid di room0 oleh object0, object1 di ruangan yang sama dan object2 di room1 mampu mengenali global.

***Local Variables***

*Local variables* adalah variabel untuk objek yang sama.

***All variables***

Semua variabel (*all whatever*) menetapkan variabel lokal untuk semua objek di dalam ruangan.

***6. Draw Actions***

***Draw a sprite***

Tindakan ini akan membiarkan objek menggambar sprite tertentu dengan gambar sub tertentu (-1 menarik menjiwai semua) dalam posisi tertentu.

***Draw a background image***

Tindakan ini akan menarik latar belakang tertentu dalam posisi tertentu.

***Draw a text***

Tindakan ini akan *draw* (menulis) teks dalam posisi tertentu.

***Draw text transformed***

This action will draw (write) text in a certain position. Tetapi dengan skala x tertentu dan skala y, dan rotasi tertentu (angle).

***Draw a rectangle***

Tindakan ini akan menggambar persegi panjang. Berikut adalah apa yang Anda harus memasukkan: x1: Posisi x di mana persegi panjang akan dimulai. y1: Posisi y mana persegi panjang akan dimulai. x2: Posisi x di mana bagian kanan bawah dari persegi panjang akan berakhir. y2: Posisi y mana bagian kanan bawah dari persegi panjang akan berakhir. Diisi: Jika diatur ke benar, persegi panjang tidak akan berongga.

***Draw a horizontal gradient, Draw a vertical gradient***

Tindakan ini menarik persegi panjang dengan efek yang bagus.

***Draw an ellipse***

Gambar oval dengan argumen yang sama seperti persegi panjang.

***Draw a gradient ellipse***

Tindakan ini menarik oval dengan efek yang bagus.

***Draw a line***

Tindakan ini akan menarik garis pergi dari x1,y1 to x2,y2.

***Draw an arrow***

Tindakan ini akan menarik panah pergi dari x1, y1 ke x2, y2, dan Anda harus mengatur ukuran ujung panah.

***Set the color***

Tindakan ini akan mengatur warna gambar.

***Set the font***

Ini akan mengatur font teks gambar yang akan digunakan dan horisontal menyelaraskan untuk teks.

***Change full screen mode***

Tindakan ini juga akan berubah menjadi layar penuh. Atau mengubah ke tampilan normal. Atau beralih antara keduanya.

***Take a snapshot of the image***

Ini akan membuat screenshot dari jendela dan simpan ke file tertentu.

***7. Path Actions***

***Set path for the instance***

Ini akan membuat obyek mulai jalan (*path*).

***End the path for an instance***

Ini akan mengakhiri objek jalan.

***Set the position on the path***

Ini akan membuat obyek menetapkan posisi jalan.

***Set the speed for the path***

Ini akan membuat objek mengatur kecepatan jalan.

***8. Motion Actions***

***Perform a step towards a point***

Ini akan membuat gerakan objek ke suatu titik.

***Step towards a point avoiding objects***

Ini akan membuat gerakan objek menuju titik menghindari benda. Entah untuk menghindari semua benda, atau benda padat.

### 2.4.2 *Game Maker Language*

Game Maker Language (GML) adalah bahasa scripting utama yang digunakan dalam gamemaker. Hal ini memungkinkan pengguna untuk lebih meningkatkan dan mengendalikan desain permainan mereka melalui pemrograman konvensional, yang bertentangan dengan sistem drag-and-drop. Sintaks dari GML meminjam aspek dari bahasa lain seperti C, C + + dan Java, secara efektif memanfaatkan Object-oriented programming.

Awalnya, GML dirancang untuk melengkapi antarmuka drag-and-drop, yang memungkinkan pengguna tingkat lanjut untuk menambahkan fungsionalitas yang lebih besar untuk game atau program. Versi yang lebih baru dari gamemaker benar-benar menggunakan GML sebagai basis mereka, dengan semua drag-and-drop fungsi sebagai pra-ditulis script GML.

GML adalah bahasa pemrograman ditafsirkan bukan bahasa yang dikompilasi. Ketika gamemaker menciptakan permainan yang berdiri sendiri, semua GML (termasuk fungsi drag dan drop) dikemas ke dalam executable permainan. Setiap kali program yang dihasilkan dieksekusi, seorang penerjemah termasuk melaksanakan perintah ditunjukkan oleh kode GML, ini membuat gamemaker-menciptakan permainan dapat dengan mudah decompiled dan membalikkan direkayasa. (Permainan pembuat ingin membatasi ini harus menggunakan enkripsi.) Menjadi bahasa ditafsirkan, GML biasanya secara signifikan lebih lambat dari bahasa dikompilasi seperti C + + atau Delphi.

GML juga termasuk dukungan untuk memuat dan menggunakan DLL. Hal ini memungkinkan fungsi dan rutinitas yang akan ditulis dalam bahasa pemrograman lain seperti C + +, Delphi, Pascal, dan lain-lain [3].

## 2.5 Kecerdasan Buatan / AI (*Artificial Intelligence*)

Kecerdasan Buatan adalah bidang studi yang berhubungan dengan penangkapan, pemodelan dan penyimpanan kecerdasan manusia dalam sebuah sistem teknologi informasi sehingga sistem tersebut dapat memfasilitasi proses pengambilan keputusan yang biasanya dilakukan oleh manusia.

Definisi kecerdasan buatan (*Artificial Intelligence*) atau disingkat AI, adalah suatu pengetahuan yang membuat komputer dapat meniru kecerdasan manusia sehingga diharapkan komputer dapat melakukan hal-hal yang memerlukan kecerdasan seperti melakukan penalaran untuk mencapai suatu kesimpulan atau melalukan translasi dari satu bahasa manusia ke bahasa manusia yang lain.

Dalam katagori AI ada yang dikatakan sebagai AI tingkat rendah dan AI sebagai tingkat tinggi. Misalnya permainan game seperti tic-tac-toe (permainan tiga dara), permainan catur dan permainan komputer yang lainnya, permainan ini dapat dikatan sebagai AI yang tingkatannya masih rendah karena keputusan yang diambilnya masih dapat diprediksi. Adapun AI sebagai tingkat tinggi misalnya pernahkah anda mendengar suatu cerita atau menyaksikan film-film fiksi yang menunjukkan suatu komputer dengan kepintaran yang melebihi manusia, yang dapat bercakap-cakap dengan manusia dan memberikan keputusan serta pendapat yang cerdik. Hal inilah yang dikatakan sebagai AI tingkat tinggi yang masih diimpikan oleh para ahli untuk mewujudkannya [5].

Menurut O'Brien atribut perilaku cerdas adalah :

a. Berfikir dan Bernalar

b. Memakai penalaran untuk memecahkan persoalan

c. Menyerap dan menerapkan pengetahuan

d. Meperlihatkan kreativitas dan imajinasi

e. Bekerja dengan situasi yang kompleks dan membingungkan

f. Melakukan tanggapan dengan cepat dan berhasil terhadap situasi baru

g. Mengenali elemen-elemen yang relatif penting dalam suatu situasi

h. Menangani informasi yang rancu, tak lengkap atau salah

Ruang lingkup penelitian dan pengembangan AI sekarang ini dapat dikelompokkan kedalam beberapa bidang yaitu:

a. Game Playing (Permainan game)

b. General problem solving (Pemecahan masalah secara umum)

c. Natural language recognition (Pengenalan bahasa alamiah)

d. Speech recognition (Pengenalan percakapan)

e. Visual recognition (Pengenalan penglihatan)

f. Robotics

g. Expert System (Sistem ahli)

AI atau Artificial Intelligence mempunyai beberapa kelebihan dibandingkan dengan kecerdasan alami (Kecerdaran Manusia). Kelebihan tersebut dipaparkan sebagai berikut:

a. AI Lebih Bersifat Permanen

Berbeda dengan AI, kecerdasan Alami yang dipunyai oleh seseorang tidak dapat disimpan. Ketika orang tersebut pindah kerja, pengetahuan yang dimilikinya ikut terbawa. AI lebih bersifat permanen karena tetap ada sepanjang sistem komputer dan program masih terpelihara.

b. AI Menawarkan Kemudahan untuk digandakan dan disebarkan

Pemindahan pengetahuan dari satu orang ke orang lain memerlukan waktu yang panjang dan bahkan mungkin pengetahuan itu tidak dapat diduplikasi secara lengkap. Adapun pengetahuan dalam sistem komputer mudah sekali untuk disalin dan dipidahkan ke sistem lain.

c. AI dapat lebih murah daripada kecerdasan Alami.

Telah banyak dibuktikan bahwa biaya membeli jasa dengan komputer lebih murah daripada biaya untuk membiayai manusia yang melaksanakan tugas yang sama.

d. AI bersifat konsisten dan teliti.

e. AI dapat didokumentasi.

### 2.5.1 Algoritma Fuzzy

#### 2.5.1.1 Pengertian Algoritma Fuzzy

*Fuzzy* secara bahasa diartikan sebagai kabur atau samar-samar. Suatu nilai dapat bernilai besar atau salah secara bersamaan. Dalam fuzzy dikenal derajat keanggotaan yang memiliki rentang nilai 0 (nol) hingga 1(satu). Berbeda dengan himpunan tegas yang memiliki nilai 1 atau 0 (ya atau tidak).

*Algoritma Fuzzy* merupakan seuatu *logika* yang memiliki nilai kekaburan atau kesamaran (*fuzzyness*) antara benar atau salah. Dalam teori *Algoritma Fuzzy* suatu nilai bias bernilai benar atau salah secara bersama. Namun berapa besar keberadaan dan kesalahan suatu tergantung pada bobot keanggotaan yang dimilikinya. *Algoritma Fuzzy* memiliki derajat keanggotaan dalam rentang 0 hingga 1. Berbeda dengan logika digital yang hanya memiliki dua nilai 1 atau 0. *Algoritma Fuzzy* digunakan untuk menterjemahkan suatu besaran yang diekspresikan menggunakan bahasa (*linguistic*), misalkan besaran kecepatan laju kendaraan yang diekspresikan dengan pelan, agak cepat, cepat, dan sangat cepat. Dan *Algoritma Fuzzy*

menunjukan sejauh mana suatu nilai itu benar dan sejauh mana suatu nilai itu salah. Tidak seperti *logika klasik* (scrisp) tegas, suatu nilai hanya mempunyai 2 kemungkinan yaitu merupakan suatu anggota himpunan atau tidak. Derajat keanggotaan 0 (nol) artinya nilai bukan merupakan anggota himpunan dan 1 (satu) berarti nilai tersebut adalah anggota himpunan.

*Algoritma Fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang *input* kedalam suatu ruang *output*, mempunyai nilai kontinyu. *Fuzzy* dinyatakan dalam derajat dari suatu keanggotaan dan derajat dari kebenaran. Oleh sebab itu sesuatu dapat dikatakan sebagian benar dan sebagian salah pada waktu yang sama.

*Algoritma Fuzzy* memungkinkan nilai keanggotaan antara 0 dan 1, tingkat keabuan dan juga hitam dan putih, dan dalam bentuk *linguistik*, konsep tidak pasti seperti "sedikit", "lumayan" dan "sangat".

Ada beberapa hal yang perlu diketahui dalam memahami sistem fuzzy, yaitu:

1. Variabel fuzzy

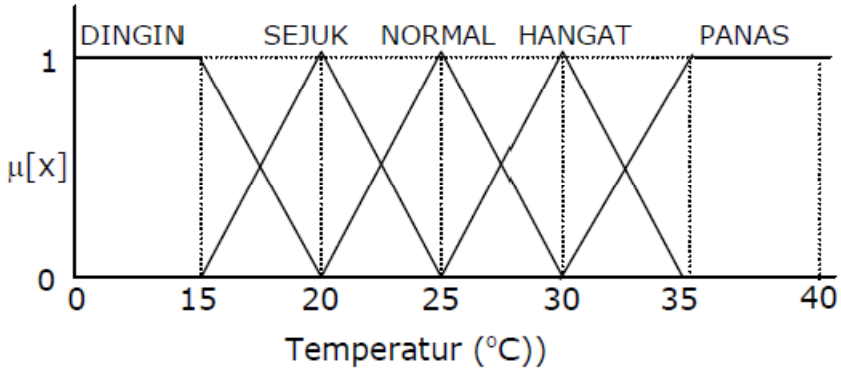
   Variabel fuzzy merupakan variabel yang hendak dibahas dalam suatu sistem fuzzy. Contoh: umur, temperatur, permintaan, dsb.

2. Himpunan fuzzy

   Himpunan fuzzy merupakan suatu grup yang mewakili suatu kondisi atau keadaan tertentu dalam suatu variabel fuzzy.

Contoh:

Variabel temperatur, terbagi menjadi 5 himpunan fuzzy, yaitu: DINGIN, SEJUK, NORMAL, HANGAT, dan PANAS.




Gambar 2.2 Himpunan fuzzy pada variabel temperature.

Kelebihan dari teori *Algoritma Fuzzy* adalah kemampuan dalam proses penalaran secara bahasa (*linguistic reasoning*). Sehingga dalam perancangannya tidak memerlukan persamaan matematik [5].

## 2.6 Adobe Photoshop CS

Photoshop adalah software yang digunakan untuk memodifikasi gambar atau foto secara profesional baik meliputi modifikasi obyek yang sederhana maupun yang sulit sekalipun. Photoshop merupakan salah satu *software* yang berguna untuk mengolah gambar berbasis bitmap, yang mempunyai tool dan efek yang lengkap, sehingga dapat menghasilkan gambar atau foto yang berkwalitas tinggi [6].

Kelengkapan fitur yang ada di dalam Photoshop inilah yang akhirnya membuat software ini banyak digunakan oleh desainer grafis profesional. Dan mungkin juga sampai saat ini masih belum ada *software* desain grafis lain yang bisa menyamai kelengkapan fitur dalam Photoshop.