SERIOUS GAME PENGOLAHAN TANAH MENGGUNAKAN BAJAK SINGKAL BERBASIS FUZZY by agung Hendra gunawan.doc

by - -

Submission date: 16-May-2024 01:05PM (UTC+0100)

Submission ID: 233944013

File name: SERIOUS_GAME_PENGOLAHAN_TANAH_MENGGUNAKAN_BAJAK_SINGKAL_BERBASIS_FUZZY_by_agung_Hendra_gunawan.doc (5.94M)

Word count: 5548

Character count: 34031

BAB I

PENDAHULUAN

1.1. Latar Belakang

Pengolahan tanah dilakukan pada lahan pertanian untuk meningkatkan sifat biologis dan fisik tanah pada kedalaman tertentu agar lebih menguntungkan bagi pertumbuhan tanaman. Pengolahan tanah (*soil tillage*) adalah sejenis pertanian darat yang menggabungkan metode tradisional dan kontemporer. Meskipun traktor digunakan sebagai tenaga penarik dalam pengolahan lahan kontemporer, ternak masih digunakan dalam pengolahan lahan tradisional untuk membantu mendorong bajak. Pengolahan tanah digunakan untuk menghilangkan gulma dari tanah dan melonggarkannya. Traktor yang menarik bajak telah membantu petani mengolah lahan mereka dengan lebih mudah.[1]. Lebar deviasi terbesar yang dapat dicapai traktor saat berjalan di lintasan dihitung ketika beroperasi di lintasan lurus.[2]

Pentingnya bagaimana mengetahui sifat tanah untuk *Serious Game* karena tanah merupakan komponen krusial yang membantu pengguna terhubung dengan subjek bajak singkal dan merasa lebih membumi. *Serious Game* sendiri diciptakan dengan dua prinsip utama: memberi semangat pada pembelajaran dan menumbuhkan lingkungan belajar yang menarik. Jika suatu tindakan menyenangkan dan menyenangkan, itu dianggap sebagai *Games*. Ciri-ciri semacam ini dapat digunakan dalam lingkungan pendidikan sebagai *intervensi* pembelajaran yang sesuai dalam berbagai keadaan tertentu. Sementara kata *Serious* dimaksudkan untuk menciptakan setting atau suasana hati yang menyerupai keadaan sebenarnya yang terlihat di dunia nyata [3].

Pada *Game* Agar tanah tampak lebih nyata dan realistis, pengolahan kualitas tanah menjadi penekanan utama dalam penelitian ini. Penelitian ini bertujuan untuk mengkaji porositas tanah dengan mempertimbangkan pengaruh kecepatan pembajakan, kedalaman, dan sudut pemotongan vertikal. Selanjutnya akan disortir dan ditentukan nilai porositasnya. Jumlah tanah yang ditempati oleh udara dan air disebut porositas, dan jumlah keseluruhan ruang pori mencakup ruang antara agregat tanah dan partikel pasir, debu, dan tanah liat.

Porositas merupakan istilah yang digunakan untuk menggambarkan besar kecilnya persentase rongga pada bahan tanah. Yang dimaksud dengan “terbuka” adalah bagian luar dari bahan tanah jika rongga tersebut dapat dihubungkan dengan bagian lain; Yang dimaksud dengan “tertutup” adalah bagian terluar dari bahan tanah yang tidak dapat diakses atau dihubungkan dengan bagian lai. [3].

Fuzzy Logic sendiri Pendekatan ini dipilih karena kemampuan adaptasi dan toleransinya terhadap data yang sudah ada. Keunggulan metode ini antara lain prediksi yang lebih jelas, komputasi lebih cepat, dan penerimaan luas [4]. Oleh karena itu, model tersebut dapat digunakan untuk memprediksi secara nyata berdasarkan teori dan data. Sehingga nanti nya *Game* tersebut dapat meprediksi dengat tepat.

Untuk menerapkan algoritma tersebut deperlukan *Game Engine*. Pada penelitian ini akan digunakan engine *Unity*, *Unity* sendiri dipilih karena fitur dalam engine ini dikemas dengan sederhana, membuat sebuah *workflow* pengembangan *Game* menjadi lebih mudah.

1.2. Rumusan Masalah

Rumusan masalah dihasilkan sebagai berikut, dengan

mempertimbangkan konteks topik yang telah dibahas sebelumnya

1. Kurangnya alat pembelajaran bagi para petani dalam mengolah tanah nya.
2. Bagaimana membuat media pembelajaran untuk pengolahan tanah yang tepat.
3. Bagaimana membuat prediksi pengolahan tanah dengan media pembelajaran *Serious Game* .

1.3. Batasan Masalah

Rumusan masalah dibuat sebagai berikut, dengan memperhatikan latar belakang topik yang telah disebutkan sebelumnya :

1. para pengguna yaitu petani hanya dapat mengetahui informasi prediksi porositas tanah.
2. Sistem prediksi hanya menggunakan metode *Fuzzy*.

1.4. Tujuan

Berikut adalah tujuan yang ingin dicapai oleh desain penelitian ini, sebagaimana ditunjukkan oleh rumusan masalah yang telah dijelaskan sebelumnya :

1. Digunakannya *Serious Game* Sebagai simulasi bajak Singkal untuk alat pembelajaran.
2. membuat media pembelajaran pengolahan tanah yang tepat.
3. *Game* dapat memprediksi porositas tanah sebagai media pembelajaran bagi para petani.

1.5. Manfaat

Manfaat yang diharapkan dari pembuatan *Serious Game* bajak singkal ini adalah:

1. Mempermudah untuk para petani mengetahui bagaimana mengolah tanah menggunakan bajak singkal.
2. Membuat para petani tidak jenuh untuk belajar.
3. Untuk meningkatkan kualitas para petani kedepannya.

1.6. Sistematika Penulisan

Berikut ini adalah bab-bab yang menyusun Sistematika Penulisan Laporan Kerja Praktek .

BAB 1 PENDAHULUAN

Isi bab ini akan mencakup sejarah, rumusan masalah,

batasan masalah, tujuan, kelebihan, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Isi bab ini memuat informasi yang menjadi sumber dan masukan bagi penelitian ini.

BAB III METODOLOGI PENELITIAN

Pada bab ini akan memberikan alur dari penelitian, mulai dari pengumpulan data dan informasi yang diperlukan dalam penelitian nantinya.

BAB IV HASIL DAN PEMBAHASAN

Untuk memastikan bahwa game berfungsi sebagaimana mestinya, bab ini mencakup pengujian dan implementasi game sesuai desain.

BAB V KESIMPULAN DAN SARAN

Sementara itu, bab ini berupaya memberikan rekomendasi dari para peneliti beserta temuan-temuan dari apa yang telah dievaluasi dan dilakukan dalam penelitian ini.

DAFTAR PUSTAKA

Pada bagian ini merupakan sumber-sumber referensi

yang menjadi acuan dalam penelitian ini.

BAB II

TINJAUAN PUSTAKA

2.1. *Serious Game*

Kata "*Serious*" biasanya mengacu pada barang-barang yang sering digunakan dalam bidang-bidang seperti teknik, perencanaan kota, politik, sains, pendidikan, militer, dan kesehatan. *Serious Game* bukan sebuah genre tapi kategori yang didalamnya dapat termasuk jenis *adverGame, political Game, educational Game*. *Game* bisnis telah banyak berkembang dipasaran, beberapa yang cukup familiar adalah *Lemonade Tycoon, Sim City, Transportation Tycoon, Roller-coaster Tycoon, Big Business Social Game* dan lain sebagainya. Membuat simulasi kehidupan buatan dengan mensimulasikan aktivitas dunia nyata di lokasi tertentu selalu menjadi salah satu konsep yang dieksplorasi.[5]

2.2. Bajak Singkal

Pengolahan tanah adalah upaya menggunakan kekuatan mekanis untuk mengubah tanah guna menyediakan kondisi yang mendukung pertumbuhan tanaman. membajak tanah adalah salah satu metode mengolah tanah. Sebagai alat

pengolahan tanah, bajak singkal dipandang sebagai mesin mekanis yang tujuan utamanya adalah mengembangkan sistem mekanis yang dapat mengatur penerapan gaya sehingga mengakibatkan terjadinya perubahan pada tanah antara lain menggemburkan, memutar, dan memotong serta pergerakan tanah. [6]

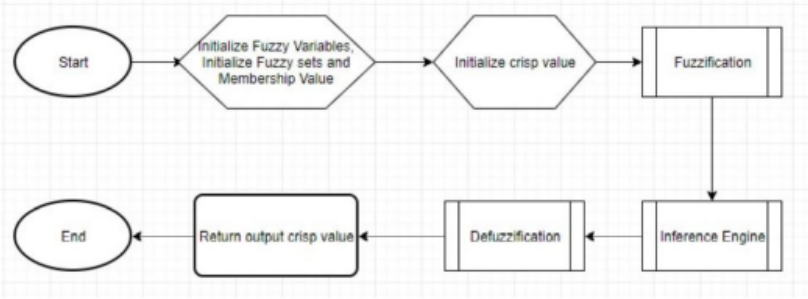
2.3. Kecerdasan Buatan

Kecerdasan buatan, atau sekadar kecerdasan buatan, adalah penambahan kecerdasan pada suatu sistem yang dapat dikontrol dalam kerangka ilmiah (bahasa Inggris: *Artificial Intelligence*) atau hanya disingkat *AI*, didefinisikan sebagai kecerdasan entitas ilmiah. Kecerdasan buatan didefinisikan sebagai "kapasitas suatu sistem untuk secara akurat memahami masukan dari luar, belajar dari data tersebut, dan menggunakan pembelajaran tersebut untuk mencapai tugas dan tujuan tertentu melalui perubahan yang dapat disesuaikan" oleh Michael Haenlein dan Andreas Kaplan. Sistem ini sering disebut dengan komputer. Kecerdasan ditambahkan pada komputer agar dapat melaksanakan pekerjaan yang dapat dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer (*Games*), logika *Fuzzy*, jaringan saraf

tiruan dan robotika. (Wikipedia)

2.4. *Fuzzy Logic*

Beginilah cara kerja metode *Fuzzy Inference* untuk menghitung nilai rem. Pada awalnya, program akan menginisialisasi *Fuzzy* Variable yang digunakan, termasuk *Fuzzy* Variable rem sebagai variable *output* yang menjadi tolak ukur nilai rem. Semua *Fuzzy* Sets beserta Membership *Value*-nya juga telah diinisialiasi oleh program. Setelah itu, program akan menerima *Input* berupa *crisp* value. *Crisp value* akan dimasukkan ke dalam proses Fuzzification dimana menggunakan *Fuzzy* Inference Engine untuk menghasilkan *Fuzzy Variable* baru yang *Fuzzy* Sets-nya sesuai dengan masing-masing *value Fuzzy* Sets. Setelah proses *Fuzzification*, *Fuzzy* Variable tersebut akan dimasukkan kedalam proses *Defuzzification* dimana akan melakukan proses hingga menghasilkan sebuah *output crisp* value berupa nilai rem.[7]



Gambar 2.1. Alur *Fuzzy*

2.5. *Unity*

Unity merupakan suatu aplikasi yang digunakan untuk mengembangkan *Game multi platform* yang didesain untuk mudah digunakan. *Unity* itu bagus dan penuh perpaduan dengan aplikasi yang profesional. *Editor* pada *Unity* dibuat degan user interface yang sederhana. *Editor* ini dibuat setelah ribuan jam yang mana telah dihabiskan untuk membuatnya menjadi nomor satu dalam urutan ranking teratas untuk editor *Game*. *Software Unity3D* adalah *Game Engine* yang dapat mengolah gambar, *Input*, animasi, teks, dan lain-lain untuk membuat suatu *Game* atau aplikasi. *Unity* merupakan *Game Engine* yang ber-multiplatform. *Unity* membolehkan pengguna untuk membuat *Game* dalam format *Standalone* (.exe), berbasis web, *Android*, *IOS Iphone*, *XBOX*, dan *PS3*. Dengan *Unity3D* kita dapat membuat *Game 3D*, FPS dan *2D Game* maupun *Game Online*. Kelebihan dari *Game Engine* ini adalah bisa membuat *Game* berbasis *3D* maupun *2D*, dan lebih mudah digunakan dibandingkan dengan *Game Engine* lainnya.[8]

2.6 FSM(*Finite State Machine*)

Finite state machine merupakan merupakan metodologi perancangan *system control* menggambarkan tingkah laku atau

prinsip kerja system dengan menggunakan tiga hal berikut: *State* (Keadaan), *Event* (kejadian) dan *action*(aksi). Sistem akan berada dalam salah satu fase aktif untuk jangka waktu tertentu. Sebagai respons terhadap kejadian atau masukan tertentu, baik dari komponen sistem internal maupun perangkat eksternal, *system* dapat berubah atau bertransisi ke keadaan baru. Biasanya, sistem merespons masukan dengan melakukan tindakan tertentu bersamaan dengan pergeseran keadaan ini. Langkah-langkah yang diambil dapat terdiri dari satu aktivitas langsung atau sejumlah operasi yang rumit [10].

2.7 Flowchart

Flowchart dapat dilihat sebagai seperangkat instruksi untuk memecahkan suatu masalah yang dinyatakan dalam simbol-simbol tertentu. Diagram alir ini akan menunjukkan alur logis program". *Flowchart* ini diperlukan tidak Masukkan Proses Keluaran hanya sebagai alat komunikasi tetapi juga sebagai pedoman, dan sebelum komponenkomponennya dapat lebih dipahami, perlu dikomunikasikan aturan-aturan desain *org chart*, yaitu: 1. *Flowchart* digambarkan dengan toporientasi ke bawah dan kiri ke kanan. 2. Setiap prosedur atau tindakan dalam bagan organisasi harus diungkapkan secara ringkas dan jelas. 3. Harus ada satu atau lebih status terminal/akhir/halt di

akhir setiap diagram alur, yang dimulai dengan prefiks atau status awal. 4. Gunakan konektor Page State dan Page Exit yang memiliki nama yang sama untuk menunjukkan bahwa hubungan antar algoritme terputus, misalnya, akibat relokasi atau perubahan halaman. Diagram alur digunakan untuk mewakili proses atau penyelesaian masalah dengan cara yang mudah dipahami, terorganisir, dan jelas dengan menggunakan sejumlah simbol umum.[11]

2.8 C# (C Sharp)

C# atau yang dibaca C *sharp* adalah bahasa pemrograman yang lugas dengan beragam aplikasi. Ini dapat digunakan untuk membuat aplikasi desktop dan seluler, permainan skrip, memprogram konten sisi server untuk situs web, dan banyak lagi. Selain itu C# juga bahasa pemrograman yang berorientasi objek, jadi C# juga mengusung konsep objek seperti *inheritance*, *class*, *polymorphism* dan *encapsulation*. C# merupakan salah satu aplikasi yang memiliki kemampuan dalam penguatan *Framework.NET*. C# sendiri dikembangkan oleh *Microsoft*. Dalam penerapannya C-*Sharp*(C#) memberikan produktifitas, fleksibilitas serta kemudahan yang ada dari aplikasi sebelumnya yaitu *Visual Basic, Java* dan C++. C# mengadopsi kemampuan dari penggabungan aplikasi

sebelumnya yaitu C, *Microsoft* membuat C# seiring dengan pembuatan *Framwork.NET*.[12]

2.9 Microsoft Visual Studio

Microsoft Visual Studio adalah keseluruhan rangkaian alat yang dapat digunakan untuk membuat aplikasi komponen perusahaan, pribadi, atau aplikasi, dalam bentuk aplikasi *console*, aplikasi *Windows*, ataupun aplikasi *Web*. *Visual Studio* mencakup kompiler, *SDK*, *Integrated Development Environment* (*IDE*), dan dokumentasi (umumnya berupa *MSDN Library*). Kompiler yang dimasukkan ke dalam paket *Visual Studio* antara lain *Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro, dan Visual SourceSafe*. *Microsoft Visual Studio* dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan di atas *Windows*) ataupun *managed code* (dalam bentuk *Microsoft Intermediate Language* di atas *.NET Framework*). Selain itu, *Visual Studio* juga dapat digunakan untuk mengembangkan aplikasi *Silverlight*, aplikasi *Windows Mobile* (yang berjalan di atas *.NET Compact Framework*). *Visual Studio* kini telah menginjak versi *Visual Studio* 9.0.21022.08, atau dikenal dengan sebutan *Microsoft Visual Studio* 2008 yang diluncurkan

pada 19 November 2007, yang ditujukan untuk platform *Microsoft .NET Framework* 3.5. Versi sebelumnya, *Visual Studio* 2005 ditujukan untuk platform *.NET Framework* 2.0 dan 3.0. *Visual Studio* 2003 ditujukan untuk *.NET Framework 1*.1, dan *Visual Studio* 2002 ditujukan untuk *.NET Framework* 1.0. Versi-versi tersebut di atas kini dikenal dengan sebutan *Visual Studio .NET*, karena memang membutuhkan *Microsoft .NET Framework*. Sementara itu, sebelum muncul *Visual Studio .NET*, terdapat *Microsoft Visual Studio 6.0 (VS1998*).[13]

2.10. *Blender*

Aplikasi *Blender* adalah program yang sering digunakan untuk membuat animasi dua dan tiga dimensi. Tidak hanya membutuhkan aplikasi untuk membuat animasi saja, imajinasi dan kreativitas pembuatnya juga sama pentingnya. Agar hasil akhir animasi lebih menarik dan mampu memberikan informasi kepada penonton, maka kesesuaian materi animasi dan informasi yang ingin disampaikan harus saling melengkapi. Tidak ada keraguan bahwa hasil animasi 2 dan 3 dimensi sangat bervariasi dalam hal model, gerakan, dan video akhir. Jelasnya, animasi tiga dimensi menghasilkan hasil yang lebih unggul dibandingkan animasi dua dimensi. Misalnya, dalam animasi dua dimensi, karakter hanya boleh

bergerak dalam dua koordinat sisi x dan y.[14]

2.11. *Adobe Illustrator*

Biasanya, program ini digunakan untuk mengembangkan berbagai persyaratan desain, termasuk karya seni digital, situs web, grafik, logo, dan materi promosi. *Adobe Illustrator* banyak digunakan untuk pemrosesan karya seni digital profesional serta tugas desain grafis yang berkaitan dengan pemasaran dan periklanan.

Ilustrator dapat digunakan untuk menghasilkan berbagai gambar digital dan media cetak, menurut desainer grafis. Kegunaan utama *Adobe Illustrator* adalah untuk desain dan gambar ilustrasi vektor. Selain itu, *Adobe Illustrator* dapat digunakan untuk mewarnai ulang item, membuat jejak pada objek gambar, dan bahkan membuat foto tampak samar.

Dengan program ini, Anda dapat membuat desain multimedia dan mengubah teks selain membuat gambar sketsa. Kemampuan untuk mengembangkan dan membangun desain mock-up sebagai templat ikon atau bentuk desain yang disajikan dalam aplikasi dan situs web juga tersedia bagi pengguna *Adobe Illustrator*.

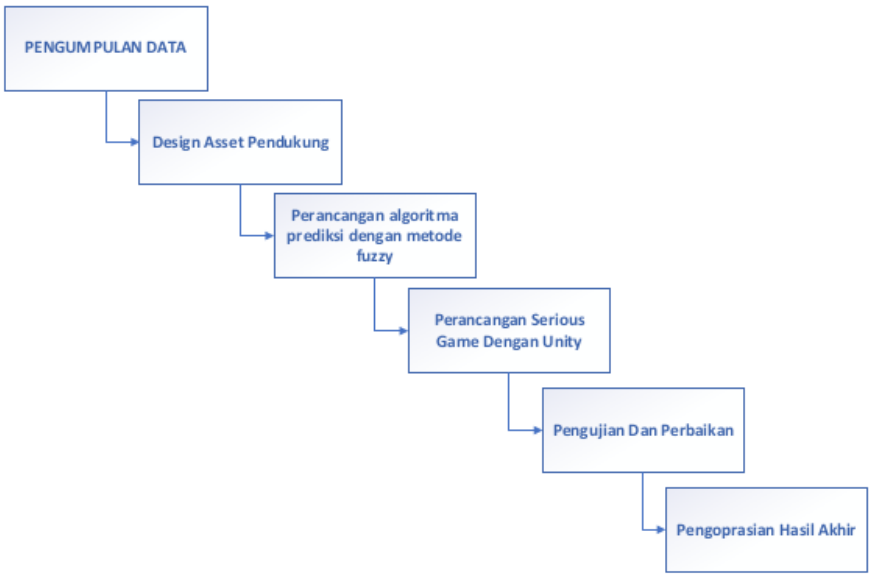
Berbicara mengenai kualitas grafis vektor yang dapat

dihasilkan oleh *Adobe Illustrator*, secara tidak langsung program ini berperan dalam menghasilkan gambar dengan kualitas yang tinggi. *Adobe Illustrator* adalah program yang sangat serbaguna dengan beragam aplikasi, tergantung pada pengetahuan dan keterampilan desainer grafis yang ingin mengolah gambar desain.[15]

BAB III

METODE PENELITIAN

Dalam memulai penelitian tentang" Perancangan *Serious Game* Pengolahan Tanah Menggunakan Bajak Singkal Berbasis *Fuzzy*"diperlukan nya tahapan tahapan atau alur penelitian



Gambar 3.1 Alur Penelitian

3.1. Pengumpulan data

Studi kasus ini mencakup penyiapan tanah menggunakan bajak *moldboard* dan pengumpulan data aktual melalui pengujian dengan alat yang dikenal sebagai wadah tanah. Tujuan dari masukan kajian yang terdiri dari kecepatan potong vertikal, kedalaman, dan sudut adalah untuk mengetahui bagaimana faktor-faktor tersebut mempengaruhi porositas tanah [9]. pada Tabel 3.1.

Tabel 3.1 data yang di ambil sebagai dasar

	SPEED	DEPTH	VERTICAL CUTTING ANGLE	CHANGES IN POROSITY
1	6.808	3.5	60	1
2	6.808	3.5	65	3
3	6.808	3.5	70	4
4	6.808	7	60	2
5	6.808	7	65	3
6	6.808	7	70	4
7	10.169	3.5	60	4
8	10.169	3.5	65	6
9	10.169	3.5	70	7
10	10.169	7	60	5
11	10.169	7	65	7
12	10.169	7	70	8
13	19.917	3.5	60	8
14	19.917	3.5	65	8
15	19.917	3.5	70	10
16	19.917	7	60	10
17	19.917	7	65	11

Pada data yang telah ditentukan untuk acuan prediksi di Tabel 3.1 dan dibuat fungsi keaanggotaan dari *variable Input* dan *output* pada Tabel 3.2 sampai Tabel 3.5.

Tabel 3.2 Speed

No	*Variable*	*Range*
1	Lambat	1 < 6.808

2	Sedang	6.808 < 10.169
3	Cepat	10.169 < 19.917

Tabel 3.3 *Depth*

No	*Variable*	*Range*
1	Dangkal	0 <= 3.5
2	Dalam	3.5 <= 7

Tabel 3.4 *Vertical cutting angel*

No	*Variable*	*Range*
1	60	55 < 60
2	65	60 < 65
3	70	65 < 70

Tabel 3.5 *Porosity*

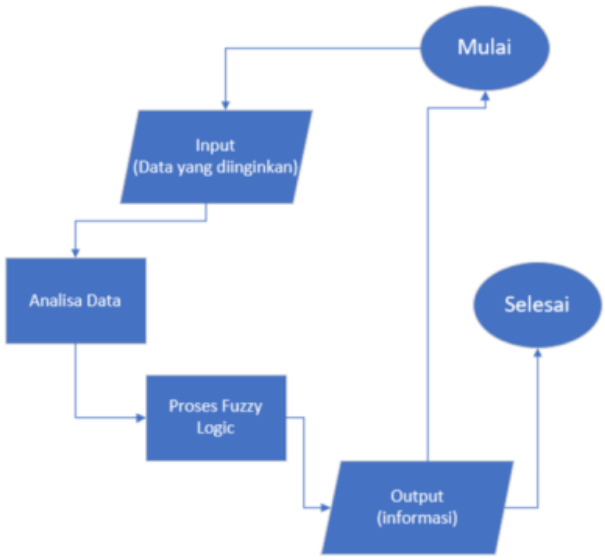
No	*Variable*	*Range*
1	Belum gembur	0 < 4
2	Cukup gembur	4 < 7
3	Sangat gembur	7 < 11

3.2. Design *asset 3D* pendukung

Mendesign berbagai *asset* dalam menunjang terbentuk nya proses perubahan tanah yang nantinya akan mendukung masuk nya data dari perhitungan *Fuzzy Logic*, seperti tanah, dan segala *asset* dalam pembajakan sawah.

3.3. Perancangan Algoritma prediksi dengan metode *Fuzzy*

Dalam memprediksi porositas tanah diperlukan nya sebuah metode yaitu *Fuzzy* yang merupakan pengambilan data yang sudah diteliti sebelumnya lalu diproses agar dapat memprediksi keluaran, meskipun masukan nya tidak ada di data awal. Proses perhitunga *Fuzzy* terdapat pada Gambar 2.1. dan sebagai acuan untuk proses untuk alur sistem yang akan dirancang pada penelitian ini seperti pada Gambar 2.1



Gambar 3.2 Alur Sistem

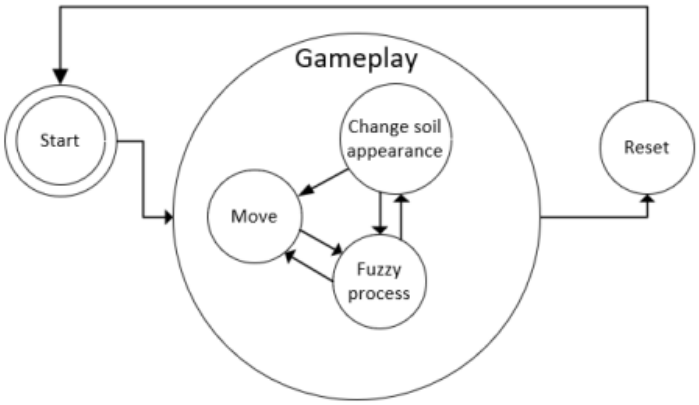
Selanjutnya Menerapkan tahap tahap *Fuzzy Logic* pada *Unity* yang nanti nya akan di sinkron kan pada *asset asset* yang sudah dibuat sebelumnya, Menggunakan Bahasa *C#* sesuai *engine* yaitu *Unity*.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Skenario Permainan

4.1.2 FSM *Gameplay* (*Finite State Machine*)



Gambar 3.3 FSM *Gameplay*

Menekan tombol WASD pada keyboard memungkinkan pemain untuk bergerak bebas saat permainan baru dimulai, lalu Ketika traktor bergerak menuju tanah sawah yang akan di bajak nanti nya 3 *variable* yang telah di *set* oleh *player* akan di proses dengan menggunakan algoritma *fuzzy* . setelah meperoleh keluaran *porositas* nanti nya akan di

lanjutkan dengan perubahan penampilan pada tanah apakah sedikit gembur ,gembur,atau sangat gembur. Lalu jika *player* ingin mengulangi kembali *player* dapat menekan *reset* untuk mengulangi nya kembali.

Skenario dari "*Serious Game* Pengolahan Tanah Menggunakan Bajak Singkal Berbasis *Fuzzy*" ini memiliki konsep tentang mensimulasikan yang diperuntukan untuk orang-orang yang membutuhkan pembelajaran tentang apa saja faktor apa saja yang menyebabkan tanah menjadi tidak gembur, gembur dan sangat gembur. Di dalam game/permainan ini berlatar lokasi di sawah. Dalam permainan ini *Player* memeran kan sebuah traktor tangan yang nanti nya akan berjalan di sawah.

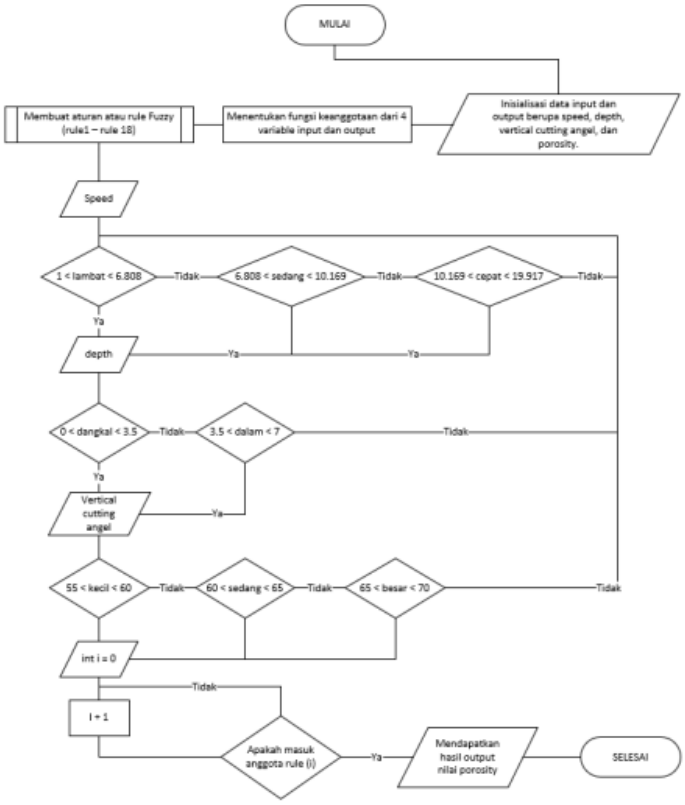
Dan nantinya Player dapat mengontrol 3 *variable* yaitu *Speed* (kecepatan), *Depth* (kedalaman), *vertical cutting angel*(titik potong vertikal).

Permainan ini nantinya akan menghasilkan output tingkat kegemburan tanah, berdasarkan 3 *variable* yang telah di tentukan di dalam game oleh Player. Tingkat kegemburan tanah ditentukan oleh 3 *variable* Input yang diproses oleh algoritma *Fuzzy* yang nantinya akan menentukan tingkat kegemburan dari tanah tersebut. Nantinya hasil dari gembur

atau tidak gembur, gembur dan sangat gembur akan menampilkan perubahan pada tanah yang telah dilewati oleh traktor. Sehingga *Player* dapat melihat perubahan pada tanah secara langsung, tingkat kegemburan akan ditampilkan pada uji *game*.

4.2 Desain dan *flowchart*

4.2.1 *Flowchart Fuzzy Logic*



Gambar 4.1 *Flowchart Fuzzy logic*

Flowchart alur (Gambar 4.1) menunjukkan dari awal proses menggunakan *Fuzzy logic* yaitu pengumpulan data *Input* dan *output*.

Variabel Input

- Kecepatan (*Speed*) : Mengukur tingkat kecepatan bajak singkal.

- Kedalaman Singkal (*Depth*): Parameter ini menentukan seberapa dalam alat bajak memasuki tanah.

- *Vertical Cutting Angle* : sudut mata pisau bajak singkal.

Variabel Output

Kegemburan Tanah: Berdasarkan *Input*, output akan menentukan apakah kondisi tanah ideal untuk diolah atau membutuhkan perubahan parameter.

Algoritma *Fuzzy* Logic

Fuzzy Logic yang diimplementasikan dalam game ini didasarkan pada teori *Fuzzy* set yang diperkenalkan oleh Zadeh pada tahun 1965 (Zadeh, 1965). Algoritma ini terdiri dari beberapa tahapan:

Fuzzification: Mengubah *variabel Input* menjadi tingkat

keanggotaan *Fuzzy*. Misalnya, kadar air tanah bisa menjadi "Rendah", "Sedang", atau "Tinggi".

Inference: Menggunakan aturan *Fuzzy* untuk memproses *Input*.

Defuzzification: Mengubah *variable* output *Fuzzy* kembali menjadi angka konkret untuk tindakan selanjutnya dalam *game*.

4.2.2 *Rules*/Aturan *Fuzzy*

1. Jika *Speed* adalah "Lambat" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Kecil," maka *Porosity* adalah "Sedikit Gembur."

2. Jika *Speed* adalah "Lambat" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Sedang," maka *Porosity* adalah "Sedikit Gembur."

3. Jika *Speed* adalah "Lambat" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Besar," maka *Porosity* adalah "Sedikit Gembur."

4. Jika *Speed* adalah "Lambat" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Kecil," maka *Porosity* adalah "Sedikit Gembur."

5. Jika *Speed* adalah "Lambat" dan *Depth* adalah "Dalam" dan

Vertical Cutting Angle adalah "Sedang," maka *Porosity* adalah "Sedikit Gembur."

6. Jika *Speed* adalah "Lambat" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Besar," maka *Porosity* adalah "Gembur."

7. Jika *Speed* adalah "Sedang" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Kecil," maka *Porosity* adalah "Sedikit Gembur."

8. Jika *Speed* adalah "Sedang" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Sedang," maka *Porosity* adalah "Gembur."

9. Jika *Speed* adalah "Sedang" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Besar," maka *Porosity* adalah "Gembur."

10. Jika *Speed* adalah "Sedang" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Kecil," maka *Porosity* adalah "Sedikit Gembur."

11. Jika *Speed* adalah "Sedang" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Sedang," maka *Porosity* adalah "Gembur."

12. Jika *Speed* adalah "Sedang" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Besar," maka *Porosity* adalah "Sangat Gembur."

13. Jika *Speed* adalah "Cepat" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Kecil," maka *Porosity* adalah "Sedikit Gembur."

14. Jika *Speed* adalah "Cepat" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Sedang," maka *Porosity* adalah "Sedikit Gembur."

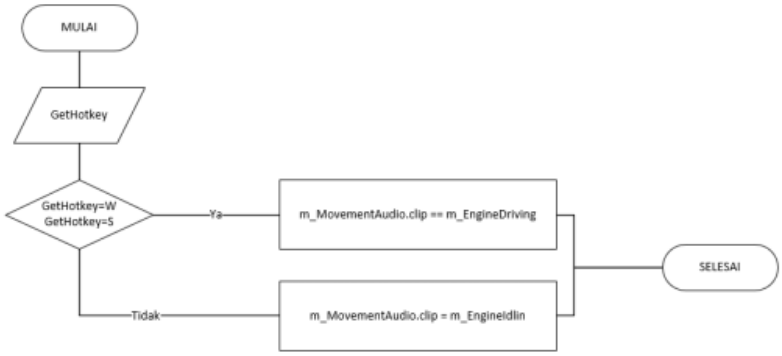
15. Jika *Speed* adalah "Cepat" dan *Depth* adalah "Dangkal" dan *Vertical Cutting Angle* adalah "Besar," maka *Porosity* adalah "Gembur."

16. Jika *Speed* adalah "Cepat" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Kecil," maka *Porosity* adalah "Gembur."

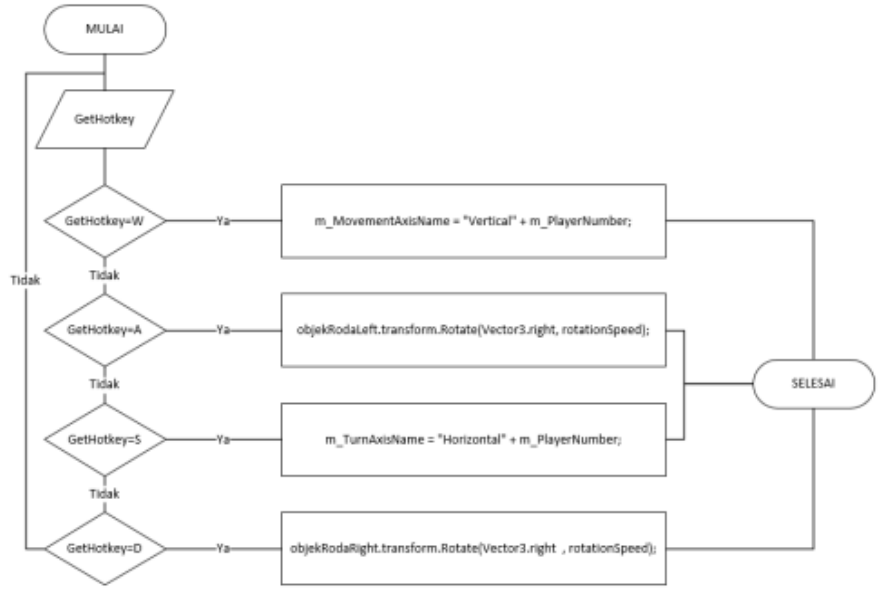
17. Jika *Speed* adalah "Cepat" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Sedang," maka *Porosity* adalah "Gembur."

18. Jika *Speed* adalah "Cepat" dan *Depth* adalah "Dalam" dan *Vertical Cutting Angle* adalah "Besar," maka *Porosity* adalah "Sangat Gembur."

4.2.3 *Flowchart movement*/pergerakan



Gambar 4.2.1 *Flowchart movement*/pergerakan dan audio



Gambar 4.2.2 *Flowchart movement*/pergerakan dan audio

Pada gambar 4.2.2 merupakan *Flowchart* yang menganalisa masukan *hotkey* pada *keyboard* WASD yang nantinya akan di proses dalam perpindah *vertex* ke traktor pada *Unity engine*. Pada gambar 4.2.1 Jika *hotkey* yang di masukkan berupa W dan S akan berbunyi traktor yang sedang berjalan. Sedangkan jika tidak memasukkan *hotkey* apa apa maka akan terdengar suara traktor yang sedang tidak berjalan.

4.3 *Desain Interface* Permainan

UI, lingkungan, *gameplay*, traktor, dan aset tambahan lainnya semuanya harus dirancang pada saat ini untuk

mendukung pengoperasian game. Pada perancangan *Serious Game* ini mengambilan sumber dari daerah di sekitar peneliti yaitu di pulau jawa. Desain area dan asset yang akan di buat akan berupa foto contoh di sekitar peneliti terlebih dahulu yang nanti nya di buat kan versi 3 dimensi nya.

Gambar 4.3 gambar tanah petak

Pada gambar 4.3 menunjukkan tanah kotak dari skitar wilayah peneliti. Ditambah beberapa pohon di sekitar.

Gambar 4.4 Traktor tangan

Lalu pada gambar 4.4 merupakan traktor yang di gunakan setiap hari oleh petani untuk membajak sawah yang berguna untuk mengemburkan tanah.

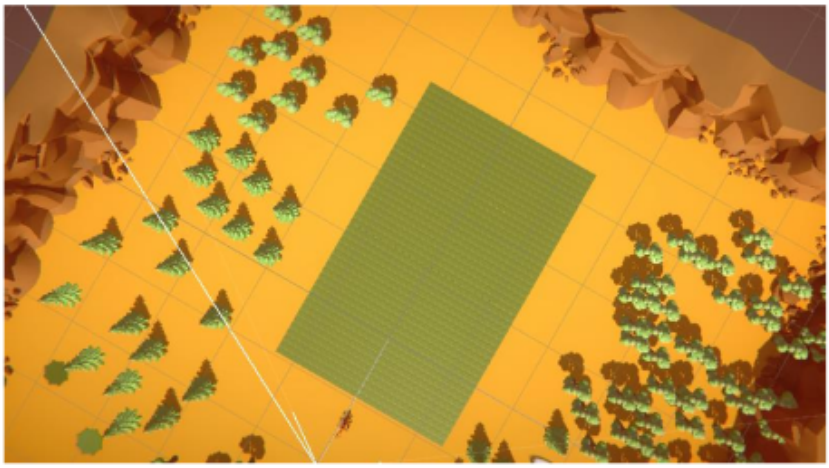
Gambar 4.5 Tata Letak

Kami mulai merancang ruang lingkup game dalam tiga dimensi pada tahap desain tata letak ini dengan menggunakan skenario desain tampilan lingkungan game yang telah ditentukan. Terdapat beberapa tempat untuk persawahan, traktor, dan peralatan pendukung lainnya pada Gambar 4.13.

Gambar 4.6 Desain Tatap Muka atau HUD

Disini tedapat beberapa desain tatap muka dalam *game* yang bisa dikontrol dan ditampilkan ditatap muka *game* nanti yang berupa *Speed Depth* dan traktor kontrol yang dapat dilihat ditatap muka sehingga *Player* dapat memahami hanya dengan melihat tatap muka tanpa menggunakan tutorial.

4.4 Implementasi Objek 3d

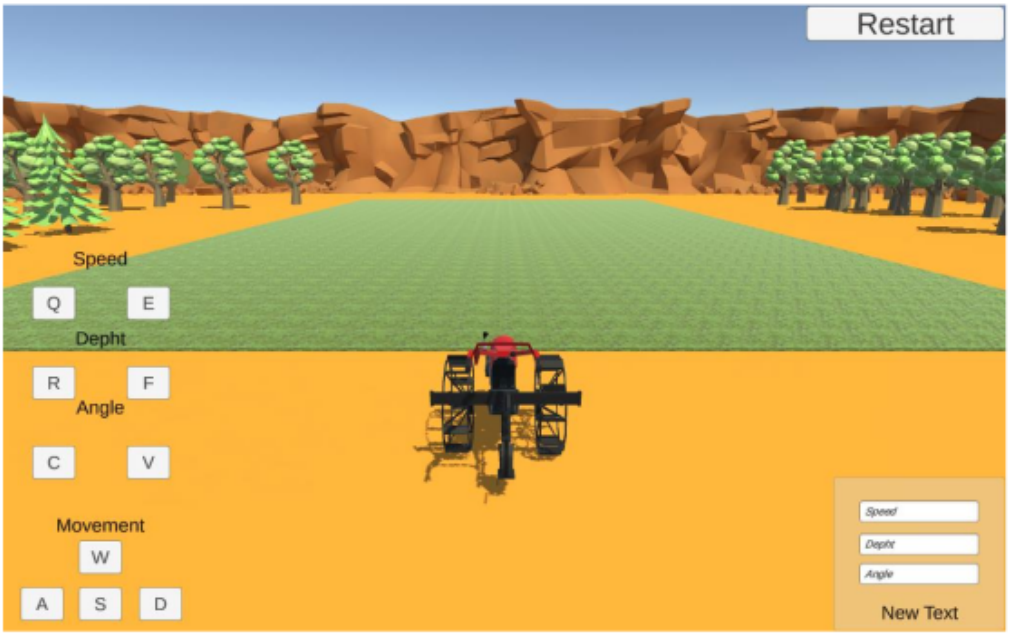


Gambar 4.7 Desain 3D Sawah

Pada Gambar 4.7 sudah dibuatkan nya 1 petak sawah dengan tambahan pohon-pohon dan bebatuan sebagai tambahan inverontment agar lebih terlihat sedikit real.

Gambar 4.8 traktor 3d

Dan pada gambar 4.8 terlihat penggambaran traktor dalam bentuk 3d yang sudah jadi.



Gambar 4.9 di dalam game

Pada Gambar 4.9 terlihat sudah berbentuk game, terdapat *Speed*, *Depth* dan vertical cutting angel yang bisa di atur oleh *Player*. Terdapat juga ui WASD sebagai cara bergeraknya tractor dan di kanan bawah Gambar 4.9 terdapat 3 variable value dan output.

4.5 Implementasi dan Uji Coba *Serious Game*

Implementasi *Game* adalah proses pembuatan game menggunakan desain game dan skenario yang telah ditetapkan.

Pada tahapan ini memulai mengimplementasikan proses pembuatan *game* berupa pembuatan koding yang dibutuhkan agar game dapat berjalan sesuai skenario. Kemudian memasukkan desain 3d dan UI/HUD yang dibutuhkan sesuai desain yang telah direncanakan.

4.5.1 *Source Code Fuzzy* Logic

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
using UnityEngine.SceneManagement;
public class UseIfLogic : MonoBehaviour
{
    public double TestSpeed;
    public double TestDepth;
    public double TestAngle;
    public string ResultTextPorsity;
    public TextMeshProUGUI ResultTMP;
    public TMP_InputField InputFieldSpeed;
    public TMP_InputField InputFieldDepth;
    public TMP_InputField InputFieldAngle;
    public Material MatSedikitGembur;
```

```
public Material MatGembur;
public Material MatSangatGembur;
public void RestartScene()
{
    SceneManager.LoadScene(0);
}
private void Start()
{
    TestSpeed = 6.808;
    TestDepth = 3;
    TestAngle = 60;
}
private void Update()
{
    SpeedUpdate();
    dephtUpdate();
    angleUpdate();
    OnCalculate();
}
private void SpeedUpdate()
{
    if (Input.GetKey(KeyCode.Q))
```

```
    {
        TestSpeed -= 0.01;
        if (TestSpeed <= 1) TestSpeed = 1;
    }
    if (Input.GetKey(KeyCode.E))
    {
        TestSpeed += 0.01;
        if (TestSpeed >= 19.917) TestSpeed = 19.917;
    }
    InputFieldSpeed.text = TestSpeed.ToString();
}
private void dephtUpdate()
{
    if (Input.GetKey(KeyCode.R))
    {
        TestDepth -= 0.01;
        if (TestDepth <= 0) TestDepth = 0;
    }
    if (Input.GetKey(KeyCode.F))
    {
        TestDepth += 0.01;
        if (TestDepth >= 7) TestDepth = 7;
```

```
    }
    InputFieldDepth.text = TestDepth.ToString();
}
private void angleUpdate()
{
    if (Input.GetKey(KeyCode.C))
    {
        TestAngle -= 0.01;
        if (TestAngle <= 55) TestAngle = 55;
    }
    if (Input.GetKey(KeyCode.V))
    {
        TestAngle += 0.01;
        if (TestAngle >= 70) TestAngle = 70;
    }
    InputFieldAngle.text = TestAngle.ToString();
}
public string ResultSpeed(double value)
{
    string result = "";
    if (value >= 1 && value <= 6.808)
    {
```

```
            result = "Lambat";
        }
        else if (value > 6.808 && value <= 10.169)
        {
            result = "Sedang";
        }
        else if (value > 10.169 && value <= 19.917)
        {
            result = "Cepat";
        }
        return result;
    }
    public string ResultDepth(double value)
    {
        string result = "";
        if (value >= 0 && value <= 3.5)
        {
            result = "Dangkal";
        }
        else if (value > 3.5 && value <= 7)
        {
            result = "Dalam";
```

```
    }
    return result;
}
public string ResultAngle(double value)
{
    string result = "";
    if (value >= 55 && value <= 60)
    {
        result = "Kecil";
    }
    else if (value > 60 && value <= 65)
    {
        result = "Sedang";
    }
    else if (value > 65 && value <= 70)
    {
        result = "Besar";
    }
    return result;
}
public float raycastDistance = 5f;
public void OnCalculate()
```

```
    {
        ResultTextPorsity = "";


        CalculationPorosity();
        Ray ray = new Ray(transform.position, Vector3.down);
        RaycastHit hit;
        if (Physics.Raycast(ray, out hit, raycastDistance))
        {
            if (hit.collider.CompareTag("box"))
            {
                MeshRenderer boxRenderer =
hit.collider.GetComponent<MeshRenderer>();
                if (boxRenderer != null)
                {
                    switch (ResultTextPorsity)
                    {
                        case "Sedikit Gembur":
                            boxRenderer.material =
MatSedikitGembur;
                            break;
                        case "Gembur":
                            boxRenderer.material = MatGembur;
```

```
                    break;
                case "Sangat Gembur":
                    boxRenderer.material                    =
MatSangatGembur;
                    break;
                default:
                    break;
            }
        }
    }
}
[ContextMenu ("Test Result")]
public void CalculationPorosity()
{
    string Speed = ResultSpeed(TestSpeed);
    string Depth = ResultDepth(TestDepth);
    string angle = ResultAngle(TestAngle);
    if (Speed == "Lambat" && Depth == "Dangkal" &&
angle == "Kecil")
    {
        ResultTextPorsity = "Sedikit Gembur";
```

```
        }
        else if (Speed == "Lambat" && Depth == "Dangkal"
&& angle == "Sedang")
        {
            ResultTextPorsity = "Sedikit Gembur";
        }
        else if (Speed == "Lambat" && Depth == "Dangkal"
&& angle == "Besar")
        {
            ResultTextPorsity = "Sedikit Gembur";
        }
        else if (Speed == "Lambat" && Depth == "Dalam" &&
angle == "Kecil")
        {
            ResultTextPorsity = "Sedikit Gembur";
        }
        else if (Speed == "Lambat" && Depth == "Dalam" &&
angle == "Sedang")
        {
            ResultTextPorsity = "Sedikit Gembur";
        }
        else if (Speed == "Lambat" && Depth == "Dalam" &&
```

```
angle == "Besar")
      {
         ResultTextPorsity = "Gembur";
      }
      else if (Speed == "Sedang" && Depth == "Dangkal"
&& angle == "Kecil")
      {
         ResultTextPorsity = "Sedikit Gembur";
      }
      else if (Speed == "Sedang" && Depth == "Dangkal"
&& angle == "Sedang")
      {
         ResultTextPorsity = "Gembur";
      }
      else if (Speed == "Sedang" && Depth == "Dangkal"
&& angle == "Besar")
      {
         ResultTextPorsity = "Gembur";
      }
      else if (Speed == "Sedang" && Depth == "Dalam" &&
angle == "Kecil")
      {
```

```
            ResultTextPorsity = "Sedikit Gembur";
        }
        else if (Speed == "Sedang" && Depth == "Dalam" &&
angle == "Sedang")
        {
            ResultTextPorsity = "Gembur";
        }
        else if (Speed == "Sedang" && Depth == "Dalam" &&
angle == "Besar")
        {
            ResultTextPorsity = "Sangat Gembur";
        }
        else if (Speed == "Cepat" && Depth == "Dangkal" &&
angle == "Kecil")
        {
            ResultTextPorsity = "Sedikit Gembur";
        }
        else if (Speed == "Cepat" && Depth == "Dangkal" &&
angle == "Sedang")
        {
            ResultTextPorsity = "Sedikit Gembur";
        }
```

```
        else if (Speed == "Cepat" && Depth == "Dangkal" &&
angle == "Besar")
        {
            ResultTextPorsity = "Gembur";
        }
        else if (Speed == "Cepat" && Depth == "Dalam" &&
angle == "Kecil")
        {
            ResultTextPorsity = "Gembur";
        }
        else if (Speed == "Cepat" && Depth == "Dalam" &&
angle == "Sedang")
        {
            ResultTextPorsity = "Gembur";
        }
        else if (Speed == "Cepat" && Depth == "Dalam" &&
angle == "Besar")
        {
            ResultTextPorsity = "Sangat Gembur";
        }
        ResultTMP.text = ResultTextPorsity;
    }
```

```
}
```

Gambar 5.0 *source code Fuzzy* beserta *Input keybind*

Pada gambar 5.0 merupakan *source code* yang berguna menjalan kan algoritma *Fuzzy* pada game sehingga game dapat berjalan dengan baik, dari mulai pengelompokan keanggotaan hingga penentuan aturan/*Rules* hingga hasil/*Output*. Pada *source code* tersebut juga terdapat tedapat keybind atau tombol yang dapat di klik oleh *Player* untuk menaikkan dan menurunkan *variable* dari *Speed, Depth, dan vertical cutting angel*, sehingga *Player* dapat melihat secara langsung perubahan angka *Input* yang berubah dan dapat disesuaikan dengan *output* apa yang diinginkan oleh *Player, Source code* tersebut juga dapat menampilkan secara langsung kepada *Player* dalam bentuk perubahan pada tanah, karena *source code* tersebut tersambung dengan *assets* tanah pada game, apakah sedikit gembur, gembur, atau sangat gembur, seperti yang di tampilkan pada gambar 5.2 sampai gambar 5.5.

4.5.2 *Source code* Pergerakan traktor dan *Input* suara

```
using UnityEngine;
```

```
namespace Complete

{

    public class TankMovement : MonoBehaviour

    {

        public int m_PlayerNumber = 1;

        public float m_Speed = 12f;

        public float m_TurnSpeed = 180f;

        public AudioSource m_MovementAudio;

        public AudioClip m_EngineIdling;

        public AudioClip m_EngineDriving;


        private string m_MovementAxisName;

        private string m_TurnAxisName;

        private Rigidbody m_Rigidbody;

        private float m_MovementInputValue;

        private float m_TurnInputValue;
```

```
private ParticleSystem[] m_particleSystems;

public GameObject objekRodaLeft;

public GameObject objekRodaRight;
7
private void Awake ()

{

    m_Rigidbody = GetComponent<Rigidbody> ();

}

private void OnEnable ()

{

    m_Rigidbody.isKinematic = false;


    m_MovementInputValue = 0f;

    m_TurnInputValue = 0f;


    m_particleSystems =
```

```
GetComponentsInChildren<ParticleSystem>();

        for (int i = 0; i < m_particleSystems.Length; ++i)

        {

            m_particleSystems[i].Play();

        }

    }

    private void OnDisable ()

    {

        m_Rigidbody.isKinematic = true;

        for(int i = 0; i < m_particleSystems.Length; ++i)

        {

            m_particleSystems[i].Stop();

        }

    }

    private void Start ()

    {
```

```
        m_MovementAxisName = "Vertical" +
m_PlayerNumber;

        m_TurnAxisName = "Horizontal" +
m_PlayerNumber;

    }

    public float rotationSpeed ;
    4
    private void Update ()

    {

        m_MovementInputValue = Input.GetAxis
(m_MovementAxisName);

        if (m_MovementInputValue > 0)

        {

            objekRodaRight.transform.Rotate(Vector3.right  ,
rotationSpeed);



            objekRodaLeft.transform.Rotate(Vector3.right,
rotationSpeed);
```

```
        }

        m_TurnInputValue = Input.GetAxis
(m_TurnAxisName);

        EngineAudio ();

    }

    private void EngineAudio ()

    {

        if (Mathf.Abs (m_MovementInputValue) < 0.1f &&
Mathf.Abs (m_TurnInputValue) < 0.1f)

        {

            if (m_MovementAudio.clip == m_EngineDriving)

            {

                m_MovementAudio.clip = m_EngineIdling;

                m_MovementAudio.Play ();

            }

        }

        else
```

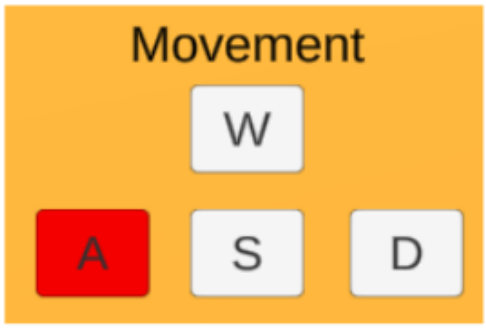
```
        {
            if (m_MovementAudio.clip == m_EngineIdling)
            {
                m_MovementAudio.clip = m_EngineDriving;
                m_MovementAudio.Play();
            }
        }
    }
4
    private void FixedUpdate ()
    {
        Move ();
        Turn ();
    }
    private void Move ()
    {
        Vector3 movement = transform.forward *
```

```
m_MovementInputValue * m_Speed * Time.deltaTime;

        m_Rigidbody.MovePosition(m_Rigidbody.position +
movement);

    }

    private void Turn ()

    {

        float turn = m_TurnInputValue * m_TurnSpeed *
Time.deltaTime;



        Quaternion turnRotation = Quaternion.Euler (0f, turn,
0f);

        m_Rigidbody.MoveRotation (m_Rigidbody.rotation
* turnRotation);

    }

  }

}
```

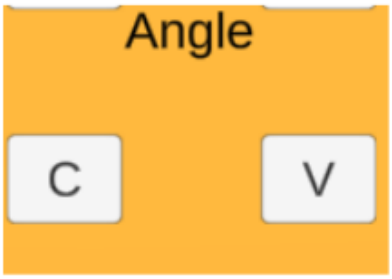
Gambar 5.1 *Source code* Pergerakan traktor dan *Input* suara

Pada gambar 5.1 yang merupakan *Source code* pergerakan dari traktor sekaligus merupakan tempat dimana suara traktor saat sedang diam dan sedang berjalan sehingga *Player* dapat merasakan secara nyata mungkin dengan ada nya suara traktor.

4.6 Implementasi *UI/HUD* dan Uji Coba *Serious Game*

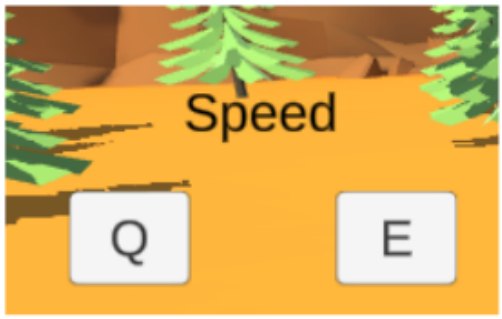


Gambar 5.2 *movement*



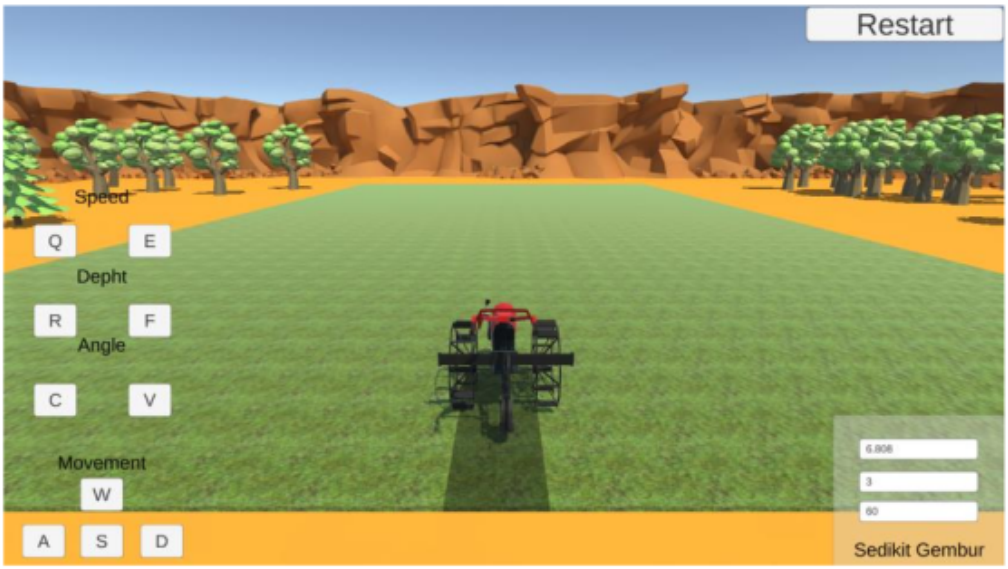
Gambar 5.3 *vertical cutting angel*

Gambar 5.4 *Depth*



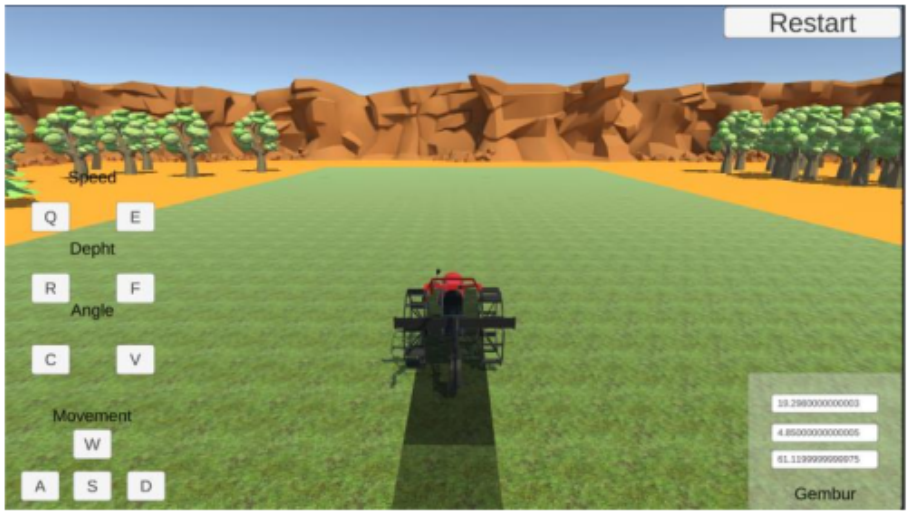
Gambar 5.5 *Speed*

Bisa kita lihat pada gambar 5.2 kita menggunakan huruf WASD pada *keyboard* untuk menggerakan traktor, W untuk maju, A untuk belok kiri, D untuk belok kanan dan S untuk membuat traktor mundur. Terdapat juga gambar 5.3 C dan V, C untuk menurunkan *vertical cutting angel* dan V untuk menaikkan *vertical cutting angel*. Lalu gambar 5.4 R dan F, huruf R untuk menurunkan nilai *Depth* dan huruf F untuk menaikkan nilai *Depth*. Yang terakhir pada gambar 5.5 terdapat huruf Q dan E, huruf Q untuk menurunkan nilai *Speed* dan huruf E untuk menaikkan nilai *Speed*.



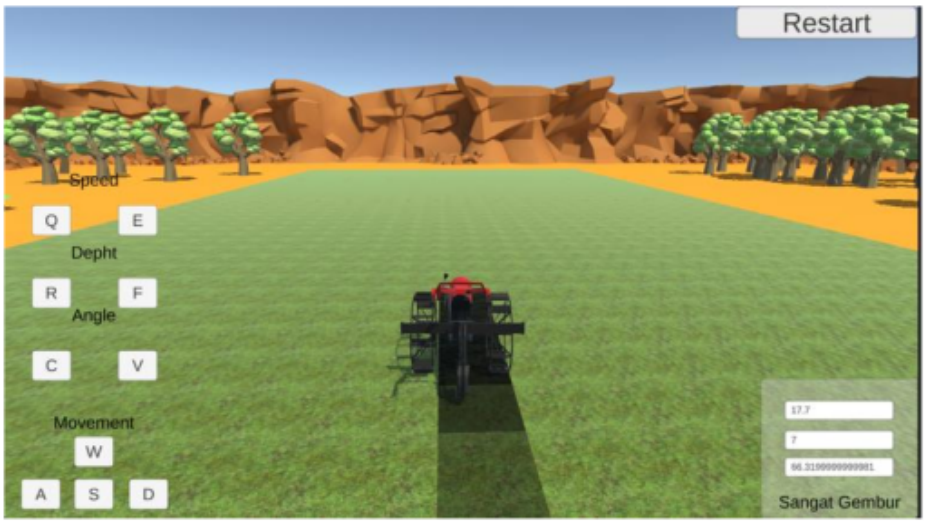
Gambar 5.6 tidak gembur

Pada gambar 5.6 pada kanan bawah gambar ketika diproses *Fuzzy* dan hasil keluarannya tidak gembur maka tanah akan sedikit tidak gelap warna nya.



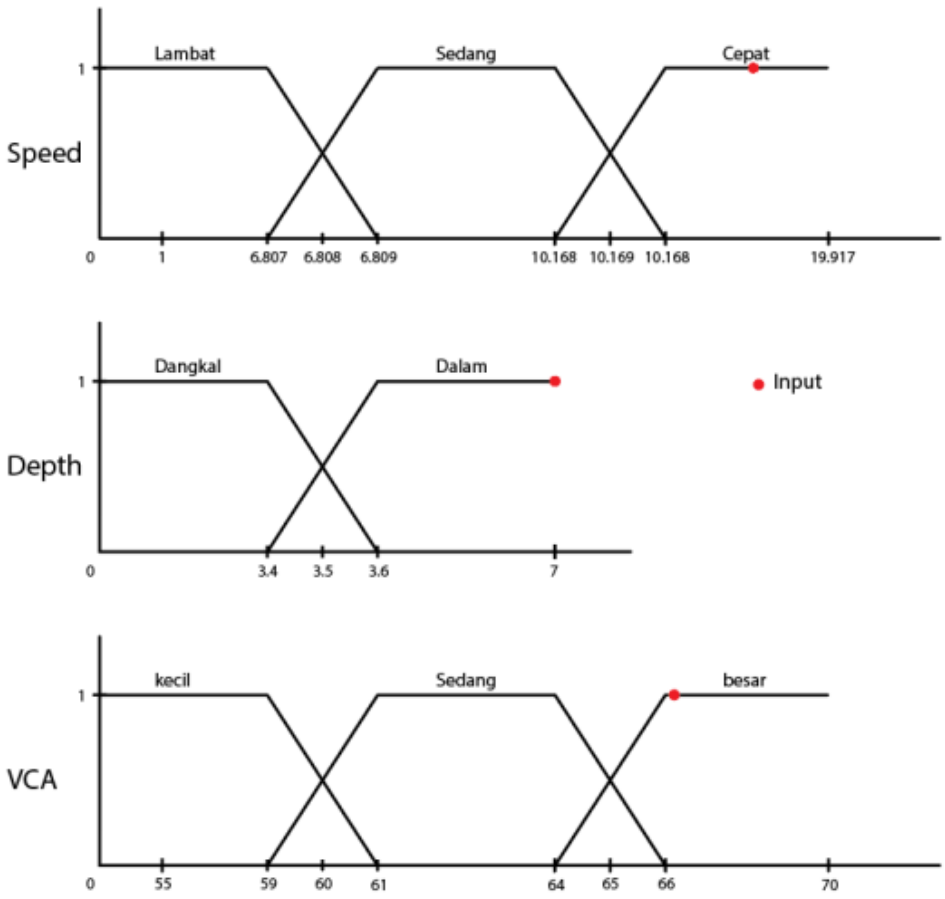
Gambar 5.7 gembur

Pada gambar 5.7 pada kanan bawah gambar ketika diproses *Fuzzy* dan hasil keluarannya tidak gembur maka tanah akan sedikit gelap warnanya. Berbeda dengan warna sebelum nya yang tidak terlalu gelap.



Gambar 5.8 sangat gembur

Pada gambar 5.8 pada kanan bawah gambar ketika diproses *Fuzzy* dan hasil keluarannya tidak gembur maka tanah akan sangat gelap warna nya. Berbeda dengan 2 warna sebelum nya yang tidak terlalu gelap.



Gambar 5.9 Grafik proses *fuzzy* 5.8 manual

$$[X]=\begin{cases} 0, & X \le X\min \\ \dfrac{X-X\min}{X\max-X\min}, & X\min \le X \le X\max \\ 1, & X \ge X\max \end{cases}$$

Gambar 6.0 Rumus *Fuzzy MIN MAX*

Terlihat pada Gambar grafik pada Gambar 5.9 merupakan proses *fuzzyfikasi* dari gambar 5.8. titik merah merupakan nilai *input* yang dicoba pada percobaan Gambar 5.8 . Dari grafik pada Gambar 5.9 tersebut nilai dari *variable speed* adalah 17.7, 17.7 masuk dalam fungsi keanggotaan dari cepat yang merupakan *MIN*=10.168 sampai *MAX*=19.917. pada nilai *variable depth* adalah 7, 7 masuk dalam fungsi keanggotaan dari dalam yang merupakan *MIN*=3.4 sampai *MAX*=7. Pada *variable* terakhir yaitu *vertical cutting angel* nilainya adalah 66.32, 66.32 masuk dalam fungsi keanggotaan dari besar yang merupakan *MIN*=64 sampai *MAX*=70. Sesuai dengan rumus pada Gambar 6.0. Setelah menganalisa nilai terhadap fungsi keanggotaan dari masing-masing *variable* nantinya akan dicari kesamaan dari rules yang telah di buat.

Tabel 4.1 Hasil Ujicoba *fuzzy* pada *game*

NO	Input			Output	
	speed	depth	Vertical Cutting Angel	Manual	Serious Game
1	7.35	3.12	58.2	Sedikit gembur	Sedikit gembur
2	12.25	4	62	Gembur	Gembur
3	8.5	5.4	64.5	Gembur	Gembur
4	13.4	6.1	66.8	Sangat Gembur	Sangat Gembur
5	5.3	4.6	67.4	Gembur	Gembur
6	4.9	5	62.2	Sedikit gembur	Sedikit gembur
7	10.4	2.5	68.8	Gembur	Gembur
8	7.3	6.2	66.6	Sangat Gembur	Sangat Gembur
9	4.4	5.7	60.5	Sedikit gembur	Sedikit gembur
10	8.1	6	67.7	Sangat Gembur	Sangat Gembur

Pada Tabel 4.1 merupakan 10 kali percobaan dari permainan bajak singkal yang telah dibuat, dengan memasukkan masukan yang ingin dicari dari *speed, depth,* dan *vertical cutting angel*. Pada tabel 4.1 dari 10 kali percobaan semuanya mendapatkan hasil keluaran yang sama dengan perhitungan manual yang telah dicoba sebelumnya.

Jika ingin mengulangi *game Player* dapat mengeklik *restart* pada kanan atas gambar 5.8. untuk mengulangi dari awal.

Dari percobaan di atas dapat dilihat hasil dari *Input* yang berupa 3 variable tersebut dapat berjalan dan menghasilkan hasil seperti pada gambar 5.7 sampai gambar 5.8 , Dimana tanah akan berubah sesuai dengan kegemburan tanah yang dihasilkan oleh algoritma *Fuzzy* yang dijalankan oleh *game* tersebut.

Kinerja algoritma *Fuzzy* Logic dievaluasi berdasarkan sejauh mana algoritma ini mampu menghasilkan keputusan yang akurat dalam konteks *game* pengolahan tanah. Beberapa indikator yang digunakan untuk mengukur keakuratan meliputi:

- Kecocokan dengan Data Empiris: Algoritma diuji dengan

data empiris yang mencerminkan situasi dalam game. Hasil dari algoritma harus sesuai dengan hasil yang diharapkan berdasarkan data tersebut.

- Kemampuan Menangani Variabilitas: Algoritma harus mampu menangani variasi *Input*, seperti kecepatan, kedalam, dan faktor lainnya.

- Kesesuaian dengan Aturan *Fuzzy*: Keputusan yang dihasilkan oleh algoritma harus sesuai dengan aturan *Fuzzy* yang telah ditentukan.

Selain keakuratan, efisiensi algoritma juga sangat penting dalam game. Evaluasi efisiensi melibatkan beberapa aspek berikut:

Waktu Eksekusi: Algoritma harus bekerja dengan cepat dan memberikan respons yang hampir instan kepada pemain. Lamanya waktu eksekusi algoritma harus sesuai dengan ekspektasi pengguna.

Penggunaan Sumber Daya: Algoritma harus dapat beroperasi tanpa menghabiskan terlalu banyak sumber daya komputer seperti CPU dan RAM.

Kemampuan Penyesuaian: Algoritma harus dapat mengatur

dirinya sendiri sesuai dengan tingkat kesulitan atau variasi yang ada dalam permainan.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

1. Game ini bisa menjadi sarana pembelajaran para petani sebagai media pengetahuan dalam mengelola tanah nya.
2. Meggunakan *engine* yang tepat serta metode perhitungan yang sudah dipakai lama oleh orang dalam menyelesaikan suatu pemmbuatan keputusan seperty *fuzzy logic*.
3. Dari *Serious Game* yang telah di buat pada penelitian ini, bahwa menggunakan *Fuzzy* logic untuk menentukan kegemburan tanah pada game bisa memperlihat kan bagaimana tanah berubah secara real time terhadap perhitungan *Fuzzy*. Sehingga dapat mmenjadi media pebelajaran *Serious Game* .

5.2 Saran

Rekomendasi untuk menambah atau memperbaiki konten

edukatif dalam *game*.

Saran untuk penelitian selanjutnya dalam meningkatkan atau memodifikasi algoritma *Fuzzy Logic* untuk simulasi yang lebih realistis atau efisien.

Rekomendasi untuk melakukan evaluasi pada kelompok usia atau latar belakang yang berbeda.