BAB III

METODOLOGI DAN PERANCANGAN APLIKASI

3.1 Metodologi Penelitian

Metode penelitian yang digunakan dalam perancangan dan pembangunan *game* simulasi pembelajaran mekanika klasik berbasis *virtual reality* dengan algoritma *Perlin Noise* antara lain sebagai berikut.

1. Studi Literatur

Tahap ini mencakup pembelajaran terhadap teori-teori desain *game*, pembelajaran materi mekanika klasik, dan pembelajaran teknik pembangunan aplikasi pada *virtual reality headset* Google Cardboard. Pada tahap ini juga dilakukan wawancara terhadap dosen Fisika Universitas Multimedia Nusantara untuk pembelajaran lebih lanjut tentang materi Fisika. Dari hasil wawancara ditentukan bahwa media *game* simulasi cocok untuk mengajarkan materi fisika. Narasumber juga mengusulkan memasukkan hambatan udara dan rotasi bola dalam beberapa *level* di *game* simulasi, akan tetapi usulan ini tidak digunakan karena keterbatasan waktu penelitian. Hasil wawancara terdapat pada Lampiran 1.

2. Perancangan *Game*

Pada tahap ini dilakukan perancangan *gameplay*, pembuatan *Game Design Document* (penjabaran *formal* dan *dramatic elements* dari *game*), pembuatan *flowchart*, serta desain aset visual dan audio untuk *game*.

3. Pembangunan *Game*

Pada tahap ini dilakukan pembuatan *game* dengan *Game Engine Unity 3D* dengan target sistem operasi Android, sesuai dengan rancangan dalam *Game Design Document*.

4. Pengujian *Game*

Pengujian dilakukan dengan metode *playtesting* oleh sejumlah responden yang dipilih secara acak. Responden kemudian akan mengisi kuesioner mengenai pengalaman selama memainkan *game* untuk menilai *usability game* simulasi. Menurut Sugiyono (2014), ukuran sampel paling sedikit yang layak dalam penelitian adalah 30.

5. Evaluasi

Evaluasi *game* dilakukan melalui analisis hasil kuesioner SUS dari tiap responden. Hasil evaluasi akan disimpulkan melalui nilai SUS untuk mengukur *usability game.*

3.2 Struktur Game

Judul *game*: Parabolix

Penjabaran *formal elements* dalam *game* simulasi ini antara lain sebagai berikut.

a. *Players*

Jumlah pemain pada *game* yaitu satu pemain (*single player*). Interaksi pemain yang terjadi yaitu interaksi *player versus game,* dimana seorang pemain berusaha mengalahkan tantangan-tantangan dalam *game*.

b. *Objective*

Tujuan dari *game* yaitu menggerakkan bola ke area *goal,* dengan menembakkan bola. Tujuan tambahan *game* yaitu meminimalisir berapa kali bola ditembakkan sebelum mencapai area *goal*.

c. *Procedures*

Aksi yang dapat dilakukan pemain dalam *puzzle mode* antara lain sebagai berikut.

1. Berjalan dan melompat, untuk menelusuri *terrain*.
2. Menggerakkan kamera sesuai gerakan kepala, untuk menyelidiki dunia *game*.
3. Menembakkan bola ke arah tertentu dengan jumlah gaya tertentu, untuk menggerakkan bola menuju area *goal*.
4. Melihat besaran-besaran fisika dalam dunia *game* pada tampilan layar, untuk membantu menentukan arah dan kuatnya tembakan yang dilakukan.
5. Menghentikan waktu dalam *game* untuk sementara, untuk mempermudah melihat besaran-besaran fisika dalam dunia *game* pada *timeframe* tertentu.
6. Melihat rumus-rumus fisika yang relevan untuk menentukan arah dan kuatnya tembakan pada layar.

Aksi yang dapat dilakukan pemain dalam *sandbox mode* yaitu aksi-aksi dalam *puzzle mode* ditambah beberapa aksi lainnya, antara lain sebagai berikut.

1. Menentukan dimensi *terrain* yaitu panjang, lebar, dan tinggi maksimal *terrain*.
2. Menentukan besar gaya gravitasi pada *terrain*.
3. Menentukan berat bola.

d. *Rules*

Peraturan yang terdapat di dalam *puzzle mode* antara lain sebagai berikut.

1. Pemain tidak dapat keluar dari *terrain* yang telah di-*generate*.
2. Pemain tidak dapat menggerakkan bola secara langsung, kecuali ditembakkan.
3. Pemain tidak dapat mengubah posisi area *goal*.

4. Pemain tidak dapat mengubah besar gaya gravitasi, berat bola, lebar *terrain*, panjang *terrain*, dan tinggi maksimal *terrain*.

Peraturan yang terdapat di dalam *sandbox mode* antara lain sebagai berikut.

1. Pemain tidak dapat keluar dari *terrain* yang telah di-*generate*.
2. Pemain tidak dapat menggerakkan bola secara langsung, kecuali ditembakkan.
3. Pemain tidak dapat mengubah posisi area *goal*.

f. *Resources*

Sumber daya yang dimiliki pemain dalam *game* adalah jumlah tembakan yang dilakukan pada bola. Jumlah tembakan yang dapat dilakukan terbatas dan semakin sedikit tembakan yang dipakai, semakin tinggi nilai pemain.

g. *Conflict*

Konflik yang terjadi dalam *game* yaitu pemain berusaha menggerakkan bola ke area *goal*, tetapi tidak bisa melakukannya secara langsung. Bola harus ditembakkan dengan gaya yang arah dan besarannya ditentukan pemain, dan bola dapat menggelinding sesuai dengan *terrain* yang ada, yang di-*generate* secara acak. Oleh karena itu, pemain perlu melakukan *trial and error* dengan mencoba menembakkan bola dan mengukur pengaruh gaya pada pergerakan bola.

h. *Boundaries*

Game memiliki batasan berupa *terrain* yang diciptakan. *Game* hanya berlangsung di atas *terrain* yang diciptakan dan tidak dapat keluar ataupun ke bagian bawah *terrain*.

Selain itu, *game* tidak sepenuhnya menirukan fisika pada dunia nyata. Fisika dalam dunia *game* dipermudah untuk memfokuskan pada pembelajaran mekanika

klasik. Sebagai contoh, pada beberapa *level* gaya gesek bola dan udara dianggap tidak ada, gravitasi selalu mengarah ke bawah, dan lain-lain.

i. *Outcome*

Hasil akhir dari *puzzle mode* yaitu *player* dapat memasukkan bola ke area *goal* dan diberi nilai tertentu. Semakin sedikit jumlah tembakan yang dilakukan *player*, semakin tinggi nilai yang didapat.

Adapun *dramatic elements* dalam *game* simulasi antara lain sebagai berikut.

a. *Challenge*

Tantangan dalam *game* ini yaitu bagaimana cara *player* memasukkan bola ke area *goal* dengan jumlah tembakan yang paling sedikit. *Player* harus menghitung besar dan arah gaya tembakan untuk menyelesaikan rintangan ini.

Selain itu, pada *puzzle mode* terdapat beberapa *level* dengan tingkat kesulitan yang berbeda-beda. *Level* yang lebih sulit akan memiliki tinggi maksimal *terrain* yang lebih besar, jarak antara bola dan area *goal* akan lebih jauh, dan gaya gravitasi serta berat bola akan berubah.

b. *Play*

Play yang terdapat dalam *game* ini yaitu kebebasan yang dimiliki *player* dalam menembakkan bola. *Player* tidak diwajibkan menembakkan bola ke arah area *goal*, selama kondisi menang (bola berada dalam area *goal*) terpenuhi. *Player* juga memiliki kebebasan bergerak ke mana saja dalam area *game*, dan bebas menggunakan *time freeze* dan *physics vision*.

c. *Premise*

Game simulasi ini tidak memiliki *premise*.

d. *Character*

Character dalam *game* simulasi ini hanya ada satu buah, yaitu karakter yang dikendalikan oleh pemain dalam *game*, dengan sudut pandang *first person*.

e. *Story*

Game simulasi ini tidak memiliki *story*.

f. *World building*

World building dalam game simulasi ini hanya terbatas pada pembuatan dunia fisik berupa *terrain*. Jenis *terrain* yang digunakan yaitu *terrain* pada bumi berupa tanah berumput.

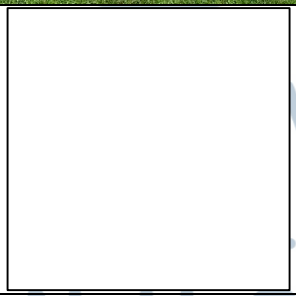
g. *The dramatic arc*

Game simulasi ini tidak memiliki *dramatic arc* karena tidak ada *story*.

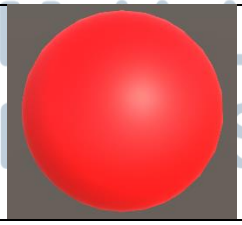
3.3 Penggunaan Aset

Aset yang digunakan dalam *game* simulasi dijabarkan pada Tabel 3.1.

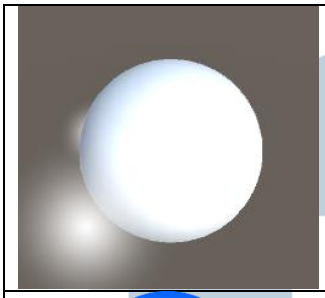
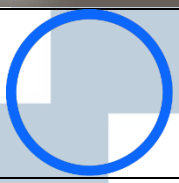
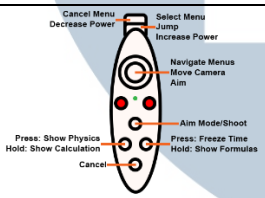
Tabel 3.1 Daftar Aset

Gambar	Penjelasan	Sumber atau Pemilik
	Tekstur rumput untuk *terrain*	Simon Murray (https://www.brusheezy.com/textures/20185-seamless-green-grass-textures)
	Tekstur salju untuk *terrain*	Aset pribadi
	Logo *game*, dipakai pada *main menu*	Aset pribadi

Tabel 3.1 Daftar Aset (Lanjutan)

	Menu Start Game, sebagai tombol pada *main menu*	Aset pribadi
	Menu Options, sebagai tombol pada *main menu*	Aset pribadi
	Menu Exit, sebagai tombol pada *main menu*	Aset pribadi
	Menu Puzzle Mode, sebagai tombol pada *main menu*	Aset pribadi
	Menu Sandbox Mode, sebagai tombol pada *main menu*	Aset pribadi
	Tulisan Level, ditampilkan pada saat memilih level *puzzle mode*	Aset pribadi
	Angka-angka untuk menampilkan level yang dipilih pada *puzzle mode*	Aset pribadi
	Logo Credits, dipakai pada *main menu*	Aset pribadi
	Logo How to Play, dipakai pada *main menu*	Aset pribadi
	Logo Controls, dipakai pada *main menu*	Aset pribadi
	Tombol Apply pada *sandbox settings*	Aset pribadi
	Tombol Back pada *sandbox settings*	Aset pribadi
	Bola, sebagai objek yang ditembakkan *player* dalam *game*	Aset standar Unity3D

Tabel 3.1 Daftar Aset (Lanjutan)

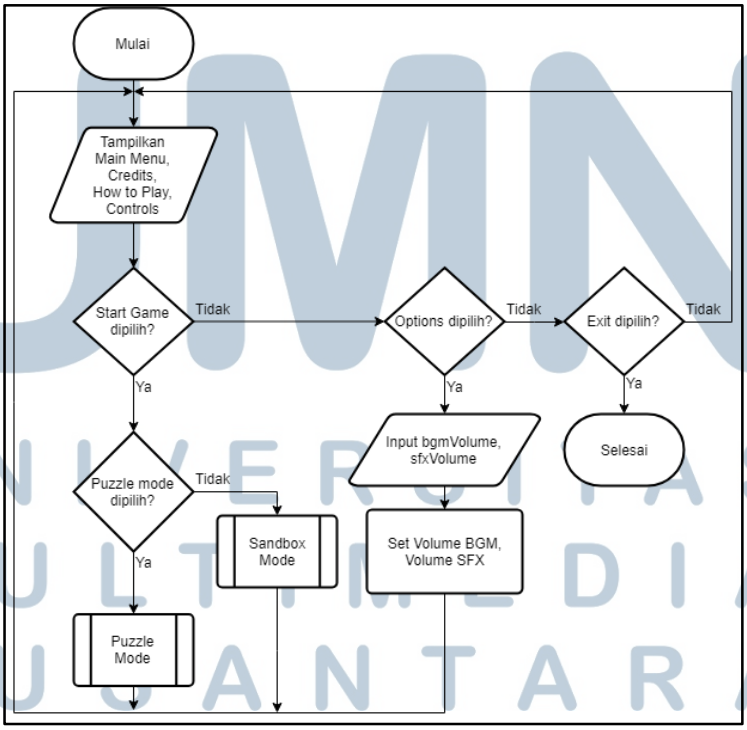
	Objek gol, sebagai target tembakan *player* dalam game	Aset standar Unity3D
	*Sprite* cincin biru, untuk indikator visual besar gaya tembakan	Aset pribadi
	*Sprite* untuk mempermudah pemain menentukan lokasi gol	Aset pribadi
	Diagram penjelasan *controls* dalam *game*	Aset pribadi
(Aset audio) Game-Menu_Looping.mp3	*Background music* yang dipakai dalam permainan	Eric Matyas (https://soundimage.org/)
(Aset audio) Magic-Clock-Shop_Looping.mp3	*Background music* yang dipakai pada *main menu*	Eric Matyas (https://soundimage.org/)
(Aset audio) 131660_bertrof_game-sound-correct.wav	*Sound effect* yang dipakai untuk suara konfirmasi pilihan menu	Bertrof (https://freesound.org/people/Bertrof/)
(Aset audio) 351563_bertrof_game-sound-incorrect-with-delay.wav	*Sound effect* yang dipakai untuk suara pembatalan pilihan menu	Bertrof (https://freesound.org/people/Bertrof/)
(Aset audio) 351564_bertrof_game-sound-correct-with-delay.wav	*Sound effect* yang dipakai saat pemain memenangkan sebuah permainan	Bertrof (https://freesound.org/people/Bertrof/)
(Aset audio) 232358_richerlandtv_heavy-impacts.mp3	*Sound effect* yang dimainkan saat *player* menembakkan bola	RICHERlandTV (https://freesound.org/people/RICHERlandTV/)

3.4 Perancangan Sistem

Perancangan sistem dilakukan dengan pembuatan *flowchart* untuk menunjukkan alur program. *Flowchart* dari *game* simulasi ini antara lain sebagai berikut.

3.4.1 Flowchart Game

Flowchart game simulasi ini ditunjukkan pada Gambar 3.1. Pada saat *game* dimulai, akan ditampilkan *Main Menu, Credits, How to Play*, dan *Controls*. *Main Menu* terdiri dari tiga buah pilihan menu, yaitu *Start Game, Options*, dan *Exit*. Bila *Start Game* dipilih, maka akan diberikan pilihan *Puzzle Mode* atau *Sandbox Mode*. Bila *Puzzle Mode* dipilih maka modul *Puzzle Mode* akan dimulai, sedangkan bila *Sandbox Mode* dipilih maka modul *Sandbox Mode* akan dimulai. Bila menu *Options* dipilih maka modul Tampilkan *Options* akan dijalankan. Bila menu *Exit* dipilih maka eksekusi program akan selesai. Tampilan *main menu* akan terus diulang oleh program hingga *player* memilih menu tertentu.



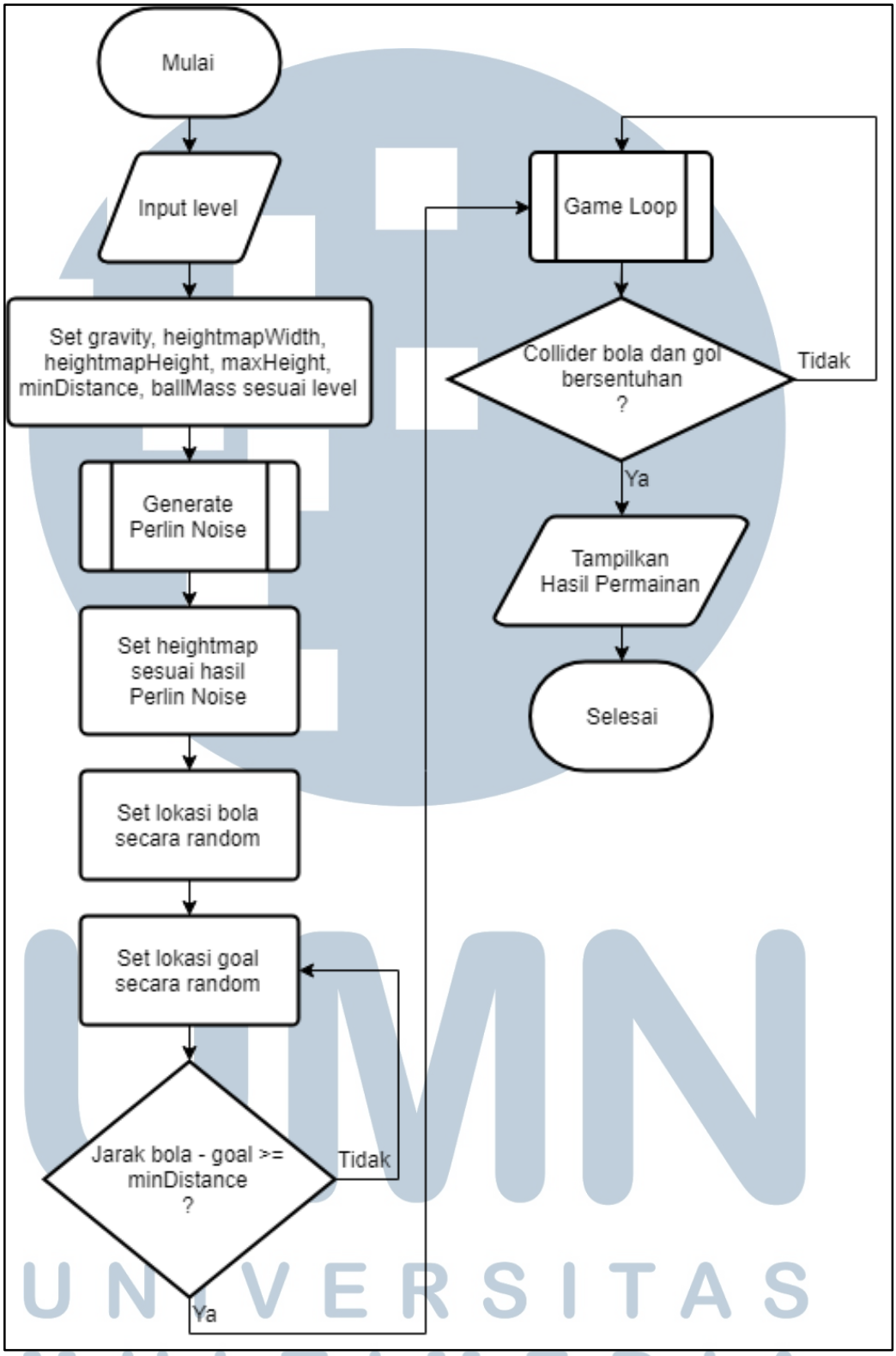
Gambar 3.1 *Flowchart* Game

3.4.2 Flowchart Puzzle Mode

Flowchart untuk modul *puzzle mode* ditunjukkan pada Gambar 3.2. Pertama *level* akan dipilih oleh *player*. Kemudian berdasarkan *level* yang dipilih, variabel *gravity, heightmapWidth, heightmapHeight, maxHeight, minDistance,* dan *ballMass* akan di-*set*. Gravitasi di dunia *game* diatur oleh variabel *gravity*, sedangkan ukuran panjang dan lebar *terrain* diatur oleh *heightmapWidth* dan *heightmapHeight*. Tinggi gunung tertinggi yang dapat di-*generate* diatur oleh *maxHeight,* dan jarak minimal antara bola dan *goal* diatur oleh *minDistance*. Semakin sulit *level* yang dipilih, maka *maxHeight* dan *minDistance* akan semakin besar, sehingga tinggi gunung semakin tinggi dan jarak bola ke gol juga semakin jauh. Massa bola diatur oleh *ballMass*.

Setelah itu, modul *Generate Perlin Noise* akan dijalankan. *Heightmap Terrain* akan di-*set* sesuai nilai-nilai hasil keluaran modul tersebut. Lokasi bola di-*set* secara *random* di *terrain* yang telah dibuat. Kemudian lokasi *goal* juga di-*set* secara *random* di *terrain*, tetapi apabila jarak antara bola dan *goal* lebih kecil daripada *minDistance*, maka lokasi *goal* akan di-*set* ulang. Bila lokasi gol sudah ditentukan, maka diambil tinggi *terrain* pada lokasi tersebut dan ditambah 5 untuk digunakan sebagai tinggi posisi gol, sehingga gol selalu berada di atas *terrain*.

Kemudian, modul *Game Loop* akan dijalankan. Setiap modul tersebut selesai dijalankan akan dicek apakah *collider* bola dan gol bersentuhan. Apabila *collider* bola dan gol bersentuhan, maka *game* akan selesai, hasil permainan akan ditampilkan, dan eksekusi modul dihentikan dan kembali ke eksekusi *game*. Bila *collider* bola dan gol tidak bersentuhan, maka eksekusi modul *Game Loop* akan diulang kembali.



Gambar 3.2 *Flowchart* Puzzle Mode

3.4.3 Flowchart Sandbox Mode

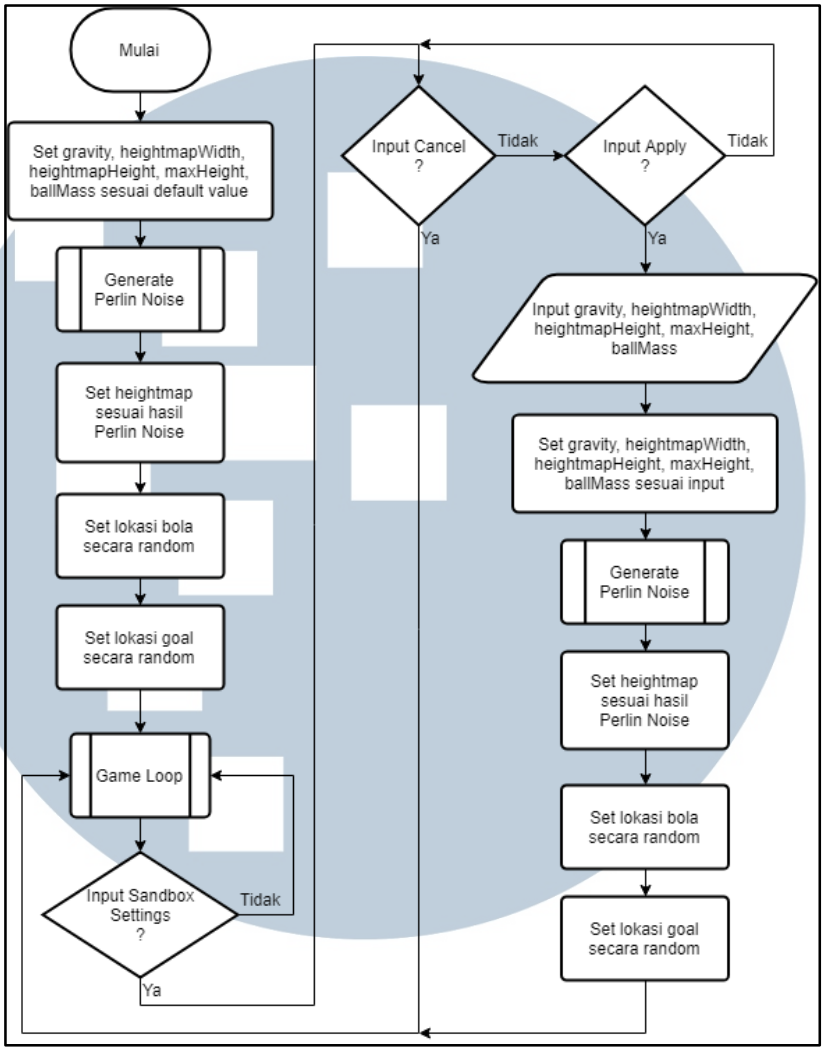
Flowchart untuk modul *sandbox mode* ditunjukkan pada Gambar 3.3. Pertama besar gaya gravitasi, panjang *terrain*, lebar *terrain*, tinggi maksimal *terrain*, dan massa bola di-*set* sesuai *default value*. Kemudian dijalankan modul *Perlin Noise* untuk menentukan *heightmap* awal. Lokasi bola dan area *goal* ditentukan secara acak.

Kemudian, modul *Game Loop* akan dijalankan. Setelah modul tersebut selesai dijalankan akan diperiksa apakah *player* menekan tombol *Sandbox Settings*. Bila tidak, maka eksekusi modul *Game Loop* akan diulang kembali.

Apabila tombol *Sandbox Settings* ditekan, maka *player* akan diberikan lima *slider* untuk menentukan *gravity*, *heightmapWidth*, *heightmapHeight*, *maxHeight*, dan *ballMass*. Kemudian *game* akan mengecek apakah pemain menekan tombol *cancel*. Apabila pemain menekan tombol *cancel* maka eksekusi modul akan kembali ke *Game Loop*.

Apabila pemain tidak menekan tombol *cancel*, maka *game* akan mengecek apakah pemain menekan tombol *apply*. Apabila pemain menekan tombol *apply*, maka nilai *gravity*, *heightmapWidth*, *heightmapHeight*, *maxHeight*, dan *ballMass* akan diambil dari *slider*, kemudian dijalankan modul *Perlin Noise* sesuai nilai baru kelima variabel tersebut. Lokasi bola dan goal kemudian ditentukan secara acak.

Bila pemain tidak menekan tombol *apply*, maka eksekusi modul diulang dari pengecekan input *cancel*.



Gambar 3.3 *Flowchart* Sandbox Mode

3.4.4 Flowchart Game Loop

Flowchart untuk *game loop* ditunjukkan pada Gambar 3.4. Pada saat g*ame loop* dimulai, pertama akan diperiksa apakah *game* sedang dalam *mode shoot*. Apabila *game* sedang dalam *mode shoot*, maka input *movement* akan diambil untuk menggerakkan arah dan kekuatan tembakan, sedangkan input *camera* akan diambil untuk menggerakkan kamera. Setelah itu akan diperiksa apakah input tembak dilakukan. Bila input tembak dilakukan, maka gaya sekarang akan diaplikasikan ke bola. Kemudian diperiksa apakah input *cancel* dilakukan. Bila input *cancel* dilakukan maka *game* akan keluar dari *mode shoot*.

Sebaliknya apabila *game* tidak dalam *mode shoot*, maka input *movement* akan diambil untuk menggerakkan karakter, dan input *camera* akan diambil untuk menggerakkan kamera. Setelah itu, akan diperiksa apakah input tembak dilakukan. Bila input tembak dilakukan, maka *game* akan masuk ke *mode shoot*.

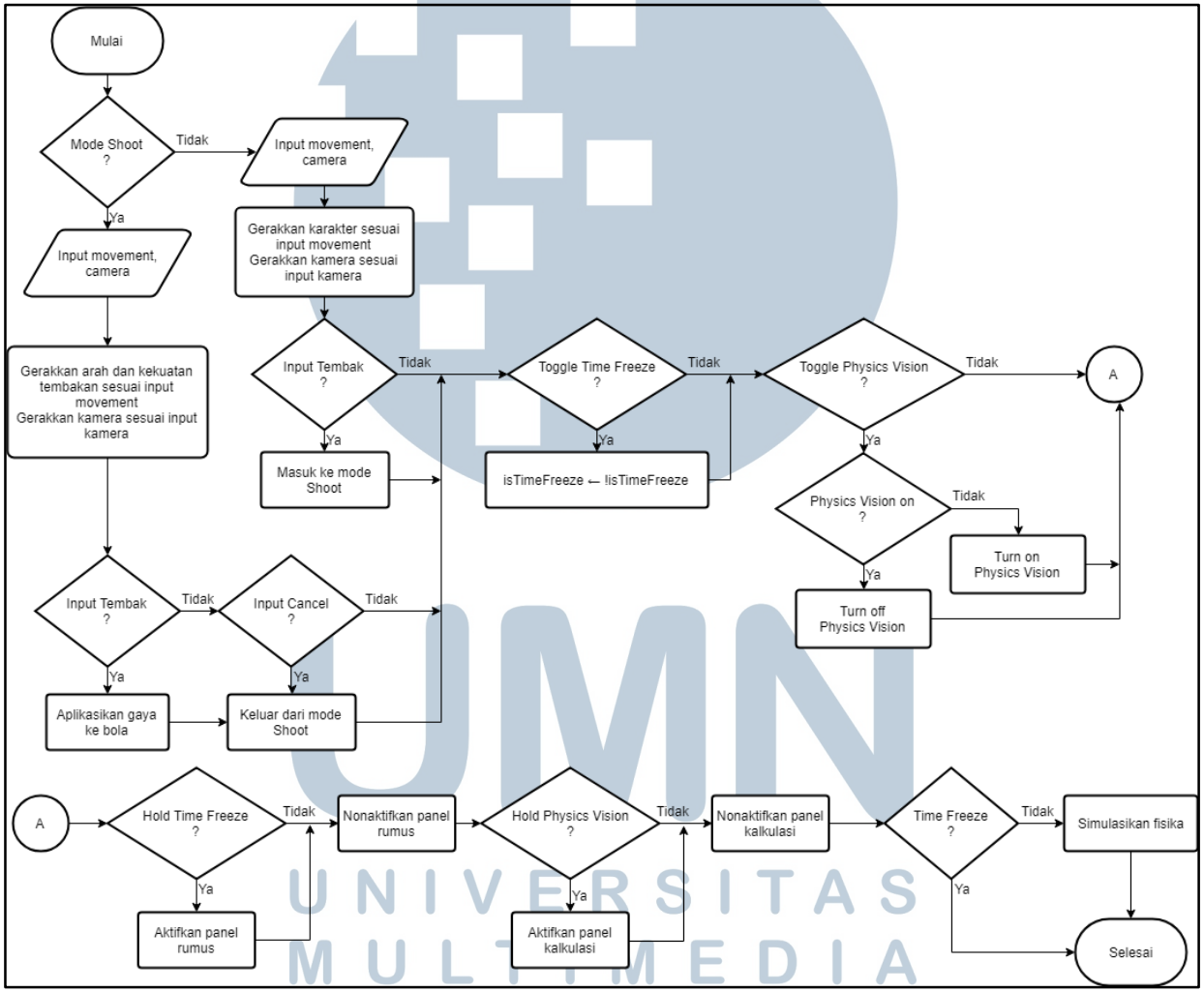
Kemudian, *game* akan memeriksa apakah *player* melakukan *toggle time freeze*. Bila *player* melakukan *toggle time freeze*, maka isi variabel *isTimeFreeze* akan dibalik (benar menjadi salah atau sebaliknya).

Kemudian *game* memeriksa apakah *player* melakukan *toggle physics vision*. Bila *toggle physics vision* dilakukan dan *physics vision* dalam keadaan on, maka *game* akan menonaktifkan *physics vision*. Sebaliknya bila *physics vision* dalam keadaan off, maka *game* akan mengaktifkan *physics vision*.

Selanjutnya *game* memeriksa apakah *player* menahan tombol *time freeze*. Apabila *player* menahan tombol *time freeze*, maka panel rumus akan diaktifkan. Sebaliknya bila *player* tidak menahan tombol *time freeze*, maka panel rumus akan dinonaktifkan.

Kemudian *game* memeriksa apakah *player* menahan tombol *physics vision*. Apabila *player* menahan tombol *physics vision*, maka panel kalkulasi akan diaktifkan. Sebaliknya bila *player* tidak menahan tombol *physics vision*, maka panel kalkulasi akan dinonaktifkan.

Terakhir, *game* akan mengecek apakah *Time Freeze* bernilai true. Bila tidak, maka *game* melakukan simulasi fisika seperti biasa. Sebaliknya bila *Time Freeze* bernilai true, maka simulasi fisika tidak dilakukan sehingga seolah waktu berhenti.



Gambar 3.4 Flowchart Game Loop

3.4.5 Flowchart Generate Perlin Noise

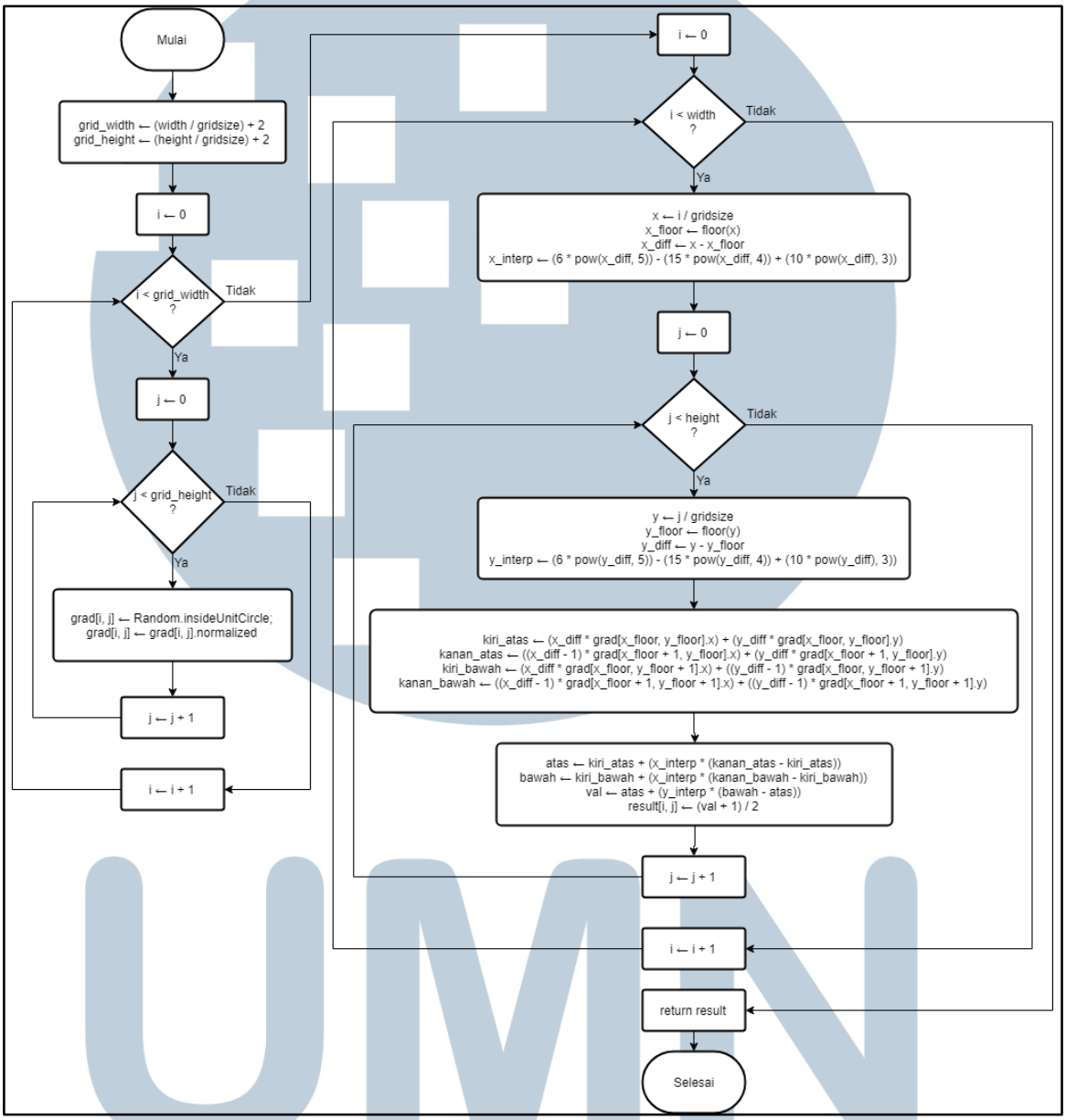
Flowchart untuk modul *generate Perlin Noise* ditunjukkan pada Gambar 3.5. Pertama *grid_width* dan *grid_height* ditentukan berdasarkan *height*, *width*, dan *gridsize* yang telah di-*set* pada modul *Puzzle Mode* atau *Sandbox Mode*. Kemudian untuk setiap titik dalam *grid* yang koordinatnya berupa bilangan bulat (integer lattice), ditentukan sebuah gradien secara acak.

Setelah itu, bagi tiap *grid* berdasarkan *gridsize* dan hitung nilai *perlin noise* di tiap titik. Langkah ini dilakukan untuk menentukan *integer lattice*. Tentukan *x_floor* sebagai sumbu x *integer lattice* terdekat dan *x_diff* sebagai jarak sumbu x titik ke *x_floor*. Kemudian hitung *x_interp* dengan rumus interpolasi *s(t)* dan parameter *x_diff*. Tentukan *y_floor* sebagai sumbu y *integer lattice* terdekat dan *y_diff* sebagai jarak sumbu y titik ke *y_floor*. Kemudian hitung *y_interp* dengan rumus interpolasi *s(t)* dan parameter *y_diff*.

Setelah *x_diff*, *x_interp*, *y_diff* dan *y_interp* dihitung, maka nilai *influence* dari keempat *integer lattice* (kiri_atas, kanan_atas, kiri_bawah, dan kanan_bawah) dapat dihitung dengan mengalikan *x_diff* dengan x dari gradien *integer lattice*, mengalikan *y_diff* dengan y dari gradien *integer lattice*, kemudian menjumlahkan keduanya.

Kemudian dari keempat nilai tersebut dicari nilai variabel atas, bawah, dan val. Variabel atas didapat dari hasil interpolasi linear sebesar *x_interp* antara kiri_atas dan kanan_atas, variabel bawah didapat dari hasil interpolasi sebesar *x_interp* antara kiri_bawah dan kanan_bawah, dan variabel val didapat dari hasil interpolasi sebesar *y_interp* antara atas dan bawah. Terakhir, karena nilai *Perlin Noise* yang dibutuhkan di antara 0 hingga 1, sedangkan hasil val berada di antara

-1 hingga 1, maka hasil *Perlin Noise* diambil dari (val + 1) / 2. Hasil *Perlin Noise* kemudian digunakan sebagai *heightmap* pada *terrain*.



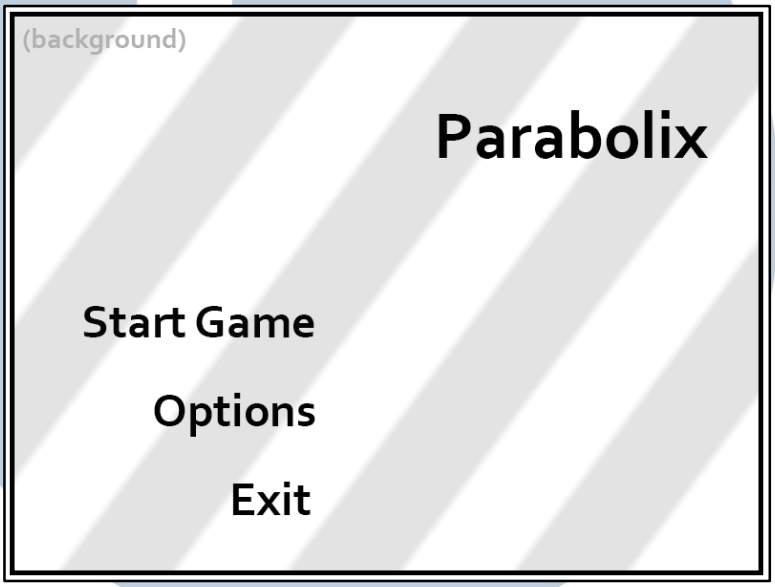
Gambar 3.5 *Flowchart* Generate Perlin Noise

3.5 Perancangan Tampilan Antarmuka

Rancangan tampilan antarmuka dilakukan dengan pembuatan *mockup interface*. *Mockup* untuk *game* simulasi ini antara lain sebagai berikut.

3.5.1 Antarmuka Main Menu

Rancangan antarmuka *main menu game* digambarkan pada Gambar 3.6. Terdapat judul *game* pada layar disertai tiga buah pilihan menu, yaitu *Start Game*, *Options*, dan *Exit*.



Gambar 3.6 *Mockup* Antarmuka *Main Menu*

Rancangan antarmuka pemilihan mode *game* digambarkan pada Gambar 3.7. Latar belakang dan judul *game* tidak berubah dari tampilan *main menu*. Terdapat dua buah pilihan mode *game* yaitu *Puzzle Mode* dan *Sandbox Mode*.

Gambar 3.7 *Mockup* Antarmuka Pemilihan Mode *Game*

Rancangan antarmuka pemilihan *level puzzle mode* ditunjukkan pada Gambar 3.8. Terdapat 10 *level* yang dapat dipilih pemain.

Gambar 3.8 *Mockup* Antarmuka Pemilihan *Level*

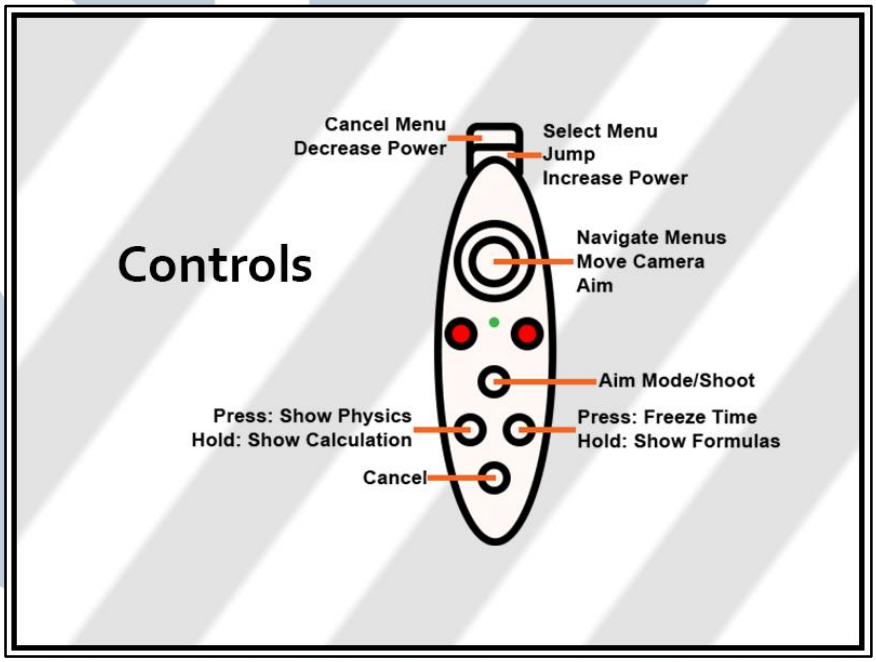
Rancangan antarmuka *credits* digambarkan oleh Gambar 3.9.

Gambar 3.9 *Mockup* Antarmuka *Credits*

Rancangan antarmuka *how to play* digambarkan pada Gambar 3.10.

Gambar 3.10 *Mockup* Antarmuka *How to Play*

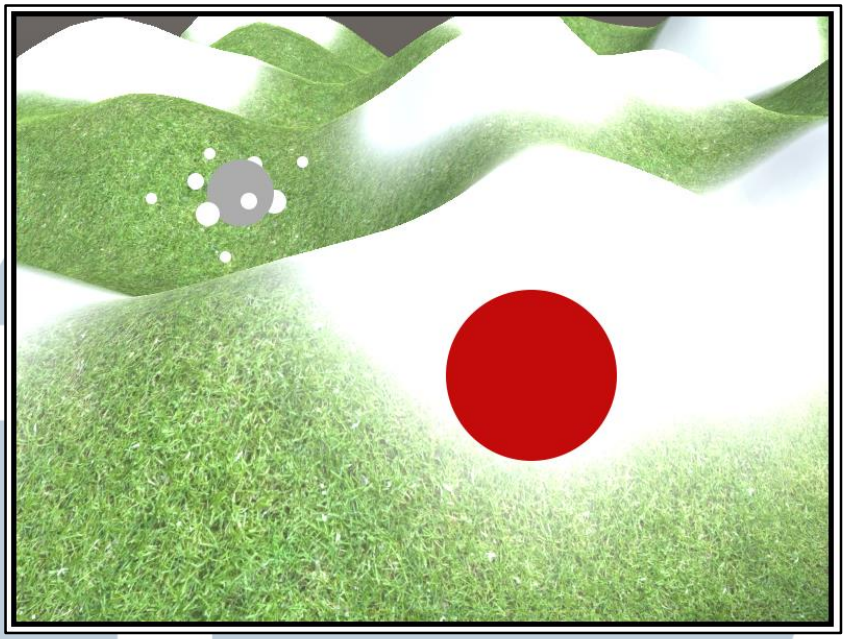
Rancangan antarmuka *controls* digambarkan pada Gambar 3.11.



Gambar 3.11 *Mockup* Antarmuka *Controls*

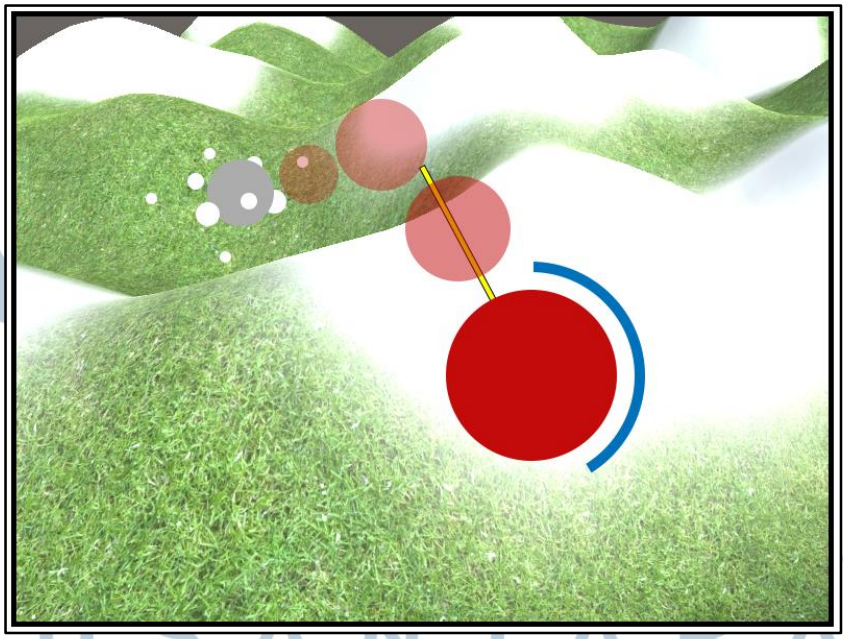
3.5.2 Antarmuka Tampilan Game

Rancangan antarmuka tampilan *game* digambarkan pada Gambar 3.12. Bola, area *goal*, dan *terrain* dapat terlihat di antarmuka *game*.



Gambar 3.12 *Mockup* Antarmuka Tampilan *Game*

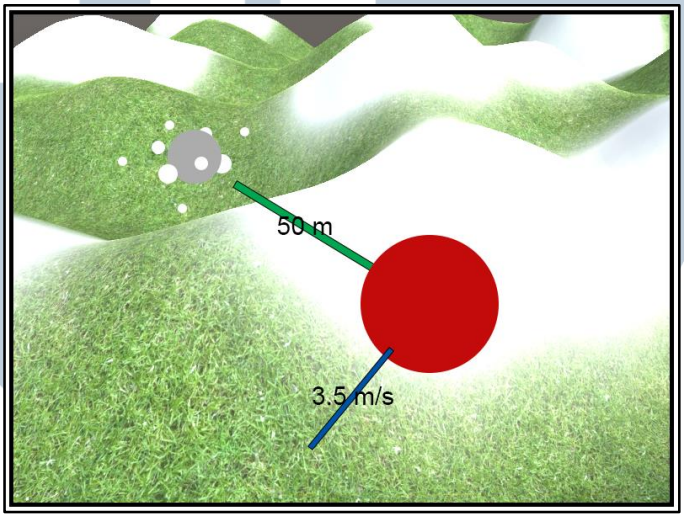
Gambar 3.13 menunjukkan rancangan antarmuka tampilan *game* saat *player* mempersiapkan sebuah tembakan. Terdapat sebuah panah kuning dari bola untuk menunjukkan arah tembakan, serta cincin biru di sekitar bola untuk memvisualisasikan besar gaya tembakan. Terdapat pula beberapa bola-bola merah transparan untuk menunjukkan *trail* dari tembakan.



Gambar 3.13 *Mockup* Antarmuka Tampilan Persiapan Tembakan

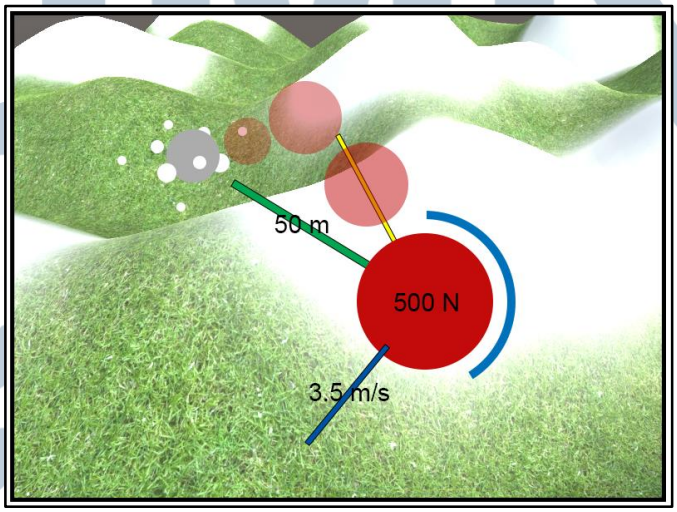
3.5.3 Antarmuka Tampilan Physics Vision

Rancangan antarmuka tampilan *physics vision* digambarkan pada Gambar 3.14. Terdapat panah biru yang menunjukkan arah gerakan bola saat ini, beserta sebuah *textbox* pada panah biru tersebut yang menunjukkan kecepatan bola saat ini. Terdapat juga sebuah panah hijau yang menunjukkan arah gol dari bola, beserta sebuah *textbox* pada panah hijau tersebut untuk menunjukkan jarak bola ke gol.



Gambar 3.14 *Mockup* Antarmuka Tampilan *Physics Vision*

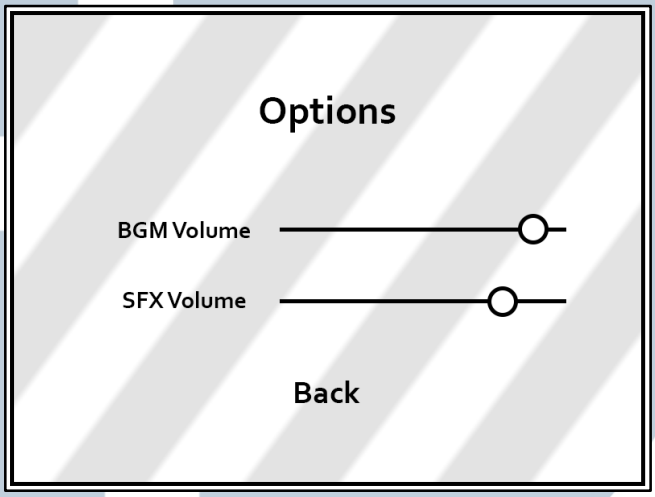
Gambar 3.15 menunjukkan rancangan antarmuka tampilan *physics vision* saat *player* mempersiapkan tembakan. Terdapat tambahan sebuah *textbox* pada bola untuk menunjukkan besar gaya tembakan.



Gambar 3.15 *Mockup* Antarmuka Tampilan *Physics Vision* Persiapan Tembakan

3.5.4 Antarmuka Options

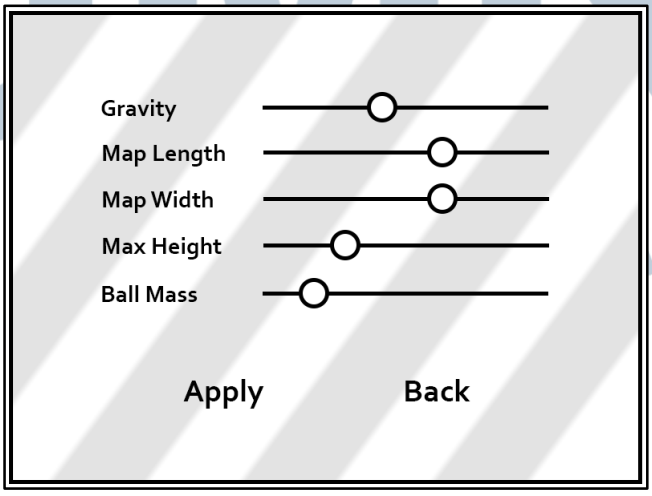
Rancangan antarmuka tampilan *options* digambarkan pada Gambar 3.16. Terdapat dua buah slider untuk mengatur besar volume BGM dan SFX. Pada bagian bawah panel terdapat tombol Back.



Gambar 3.16 *Mockup* Antarmuka *Options*

3.5.5 Antarmuka Sandbox Settings

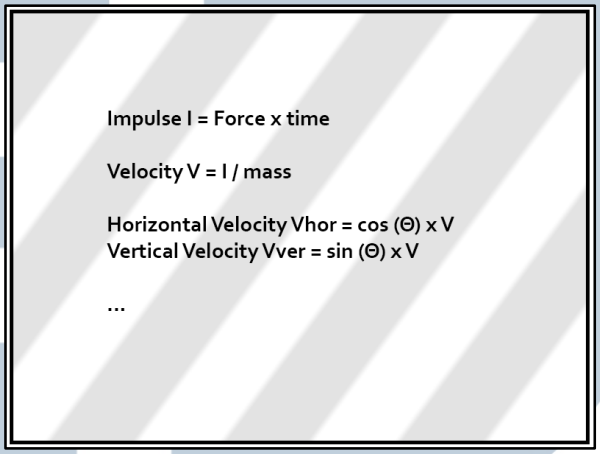
Rancangan antarmuka *Sandbox settings* ditunjukkan pada Gambar 3.17. Terdapat lima buah *slider* untuk mengatur gaya gravitasi, panjang *terrain*, lebar *terrain*, tinggi maksimal *terrain*, dan massa bola. Terdapat dua buah tombol di bagian bawah panel, yaitu tombol Apply dan Back.



Gambar 3.17 *Mockup* Antarmuka *Sandbox Settings*

3.5.6 Antarmuka Panel Rumus

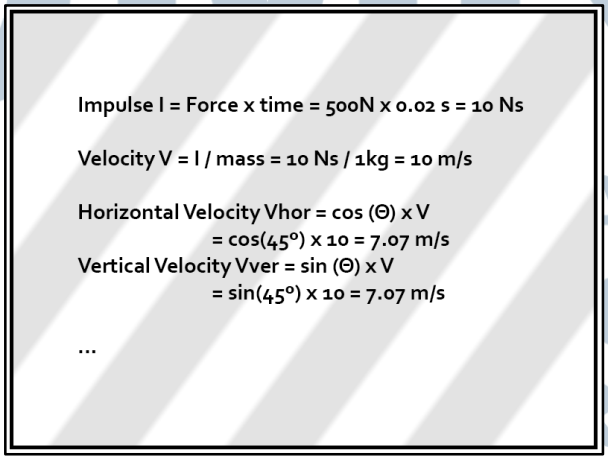
Rancangan antarmuka panel rumus digambarkan pada Gambar 3.18. Terdapat teks yang menampilkan rumus-rumus fisika yang dapat digunakan *player* untuk menghitung gaya dan arah tembakan. Panel ini akan ditampilkan di depan posisi *player* dalam dunia *game*.



Gambar 3.18 *Mockup* Antarmuka Panel Rumus

3.5.7 Antarmuka Panel Kalkulasi

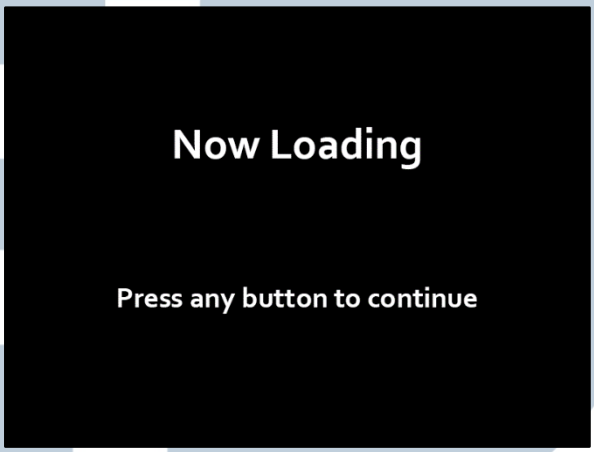
Rancangan antarmuka tampilan panel kalkulasi digambarkan pada gambar 3.19. Terdapat kalkulasi dari besaran-besaran fisika untuk membantu *player* menghitung besar gaya yang diperlukan. Panel ini akan ditampilkan di depan posisi *player* dalam dunia *game*.



Gambar 3.19 *Mockup* Antarmuka Panel Kalkulasi

3.5.8 Antarmuka Loading Screen

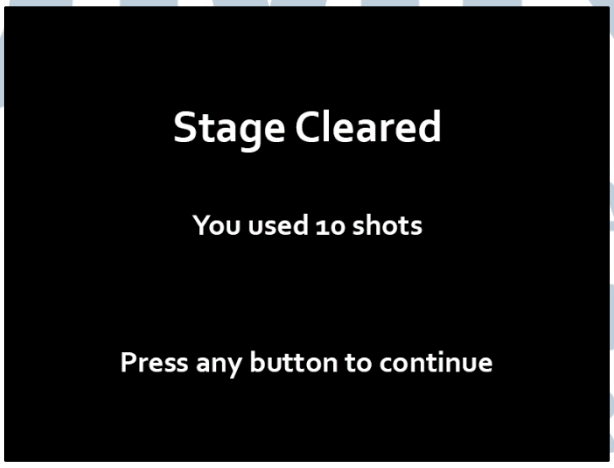
Rancangan antarmuka *loading screen* ditampilkan pada Gambar 3.20. Seluruh tampilan layar akan menjadi hitam dan di depan *player* akan ditampilkan tulisan "Now Loading". Setelah *loading* selesai maka akan ditampilkan tulisan "Press any button to continue".



Gambar 3.20 *Mockup* Antarmuka *Loading Screen*

3.5.9 Antarmuka Win Screen

Rancangan antarmuka *win screen* ditampilkan pada Gambar 3.21. Seluruh tampilan layar akan menjadi hitam dan di depan *player* akan ditampilkan tulisan "Stage Cleared" dan jumlah tembakan yang telah digunakan *player*. Setelah *loading main menu* selesai maka akan ditampilkan tulisan "Press any button to continue".



Gambar 3.21 *Mockup* Antarmuka *Win Screen*