Project: Chaos Coaster

SS2023 **Computer Games Laboratory**

Anil Celik Maral

Erick Lorenzen Meneses

Lorenzo Russo da Costa Auer

Matija Jajcinovic

Supervisor: Prof. Rüdiger Westermann

—

Contents

1. Game Description

We decided early on that our goal for the game would be to create a similar player experience to that of riding a rollercoaster: A thrilling and chaotic experience where it feels like we must simply do our best to survive until the end. Something that is tense and hectic during the ride, but afterwards feels like a fun time. How best to emulate such a feeling was our main focus and in the end we arrived at an idea for a game that, while straightforward in some ways, will feel unique, thrilling and, most importantly, fun.

Chaos Coaster is a first-person shooter with a focus on physics simulations for the environment, enemies and, naturally, the player as well. In particular, the goal is that each level becomes a physics sandbox where objects and enemies can be knocked over, sent flying, used as projectiles or platforms, etc. Players should use objects and weapons creatively to defeat enemies and navigate the environment in a swift manner. The big twist of the game comes then from its setting and how it affects the moment-to-moment gameplay. [The genre that we are targeting is the platformer genre but with additions such as physics based gameplay and potentially puzzles.]

Setting and Roller Coaster Connection

In order to make full use of the theme and introduce a whole other dimension to the chaos of every encounter, the game takes place inside a giant train that is driving through space. Every level in the game takes place inside one of the sections of the trains, with each having different environments where an enemy encounter takes place. The train is constantly moving and will, during gameplay, turn in different directions and even do more extreme movements like one would expect from a Roller Coaster, such as loops or turning upside down. These movements should be reflected in the environment and affect the gameplay in a tangible way. The main ways in which train movement can affect gameplay are as follows:

- The train makes a sharp turn left, causing the momentum to push all objects, player included, inside to its right side.
- The train flips to its side or fully upside-down, essentially causing the entire map to flip and objects to fall to the ceiling.
- The train begins a sharp descent, reducing the effects of gravity on the arena and making everything more floaty and slanting the map forward.
- The train begins a sharp ascent, increasing gravity and making everything feel heavier.
- A sudden change in the speed of the train can push objects either forward or backwards.

More complex movements like loops or rotations should be possible as well, though these may end up being highly disorienting during gameplay. One may argue that ending up nauseous is part of the experience of riding a roller coaster, but it may not result in the best experience for the player so these more extreme movements should be used in moderation.

Game Levels

While the game takes place in a “train”, it is not meant to be a realistic depiction of one and each individual level should not be constrained by what a train should look like (seat rows, narrow corridors, etc), instead the focus should lie in creating an arena for the player to fight enemies, use the objects with ample space and generally have room for expression in gameplay. In this sense, each level will more end up resembling an arena from Doom than a train wagon, but we believe here fun should take precedence over realism.

This does not mean, however, that realism should be entirely discarded. Keeping a general theme and making it look like a place that people may foreseeably live in and have used, like:

- A dining hall with many tables and chairs, separated into different floors
- An energy reactor that is a big, circular structure in the middle of the room
- A living quarter with rooms and balconies on the walls and some form of park in the center

Each level should be designed with specific train motions in mind, for example if the level can be flipped upside-down then consideration should be had about how it can be played on both the floor and ceiling, and how the transition will take place (slowly over time vs. a sudden turn).

As for the actual structure of the gameplay, the goal of each level is to defeat one or more waves of enemies, with waves spawning when all the enemies of the previous one are defeated. Then the player will receive a score based on how long it took to clear a level and how they killed the enemies (consecutive kills, headshots, environmental, etc.). Each level should have its own predetermined set of roller coaster movements, which will then loop until the player clears the level. Having the movements be random is also an option and it may result in more chaotic combat encounters, but testing may be needed to see which approach is better. The player should have advance notice of which movement is coming and how long until it arrives in the form of a UI notice.

Weapons and combat

The weapons are the main way in which the player can interact with the environment and, of course, the enemies. As such, they should allow the player to express their creativity and draw out the potential of our physics sandboxes. Traditional weapons, such as guns that simply shoot bullets and deal damage to enemies, are therefore not ideal, instead weapons that use the physics of the game to create even more mayhem on the scene are preferable.

Our main weapon is known as the Fabricator Rifle and it can be used to create smaller objects and launch them as projectiles. The weapon can be charged, causing the force and damage of the projectile to increase. If shot quickly, it behaves similarly to a normal semi automatic rifle, but holding the projectile for longer can be used to push enemies back, deal more damage to limbs or even push the player away from the shooting direction, which may be used for retreating or for super jumps. The player can continue to move while charging the gun and it does not require ammo to be used, so it can function as a backup when all the other weapons are empty.

Another base weapon will be the Gravity Gauntlet, when hitting an enemy it will apply an effect that reduces gravity for that enemy and pushes them away. This way, even though the impact is not super high, the enemy is sent flying away slowly in 0G. Another option is for the player to punch the ground and apply the reduced gravity to themselves for easier mobility. Depending on how strong this ability is, it may be necessary to balance it by adding a cooldown to the gravity effect, making the attack a regular melee if it is off cooldown.

In later levels Gravity shoes may be used to give the player the option to climb walls.

Other weapons can be added which can be collected throughout the arena but have limited ammunition, thus requiring the player to move around to collect them. Some ideas for these weapons are as follows:

- Ballista: A powerful precision tool; in peace times used to fix broken pipes. Shoots devastating heavy bolts which can stick enemies to walls. Alternatively, it may be used as a grappling hook in an alternate fire way.
- Lifter Field: A grenade that activates an artificial gravity field in a small area. Used to lift heavy loads.Can be used to create negative gravity, launching foes upwards, or positive gravity, slamming foes into the ground and slowing them down.
- Force Blaster: A weapon that does little damage but can send enemies and players alike flying. It can blast the area in a cone in front of the player, applying a strong force that pushes enemies away while also applying a strong force to the player in the opposite direction. An alternate fire mode is also available which pushes the player and the enemies towards each other. Use with caution.

- Klex: Nobody SAW it coming. A malfunctioning, limb-separating flex-saw which propels its disc forward is a horror for every company's insurance contribution. However, it is a useful tool in case of an invasion.
- ARD gun: it is not the kind of ARD that burns your tax money, but releases a cluster of small flying autonomous repair drones equipped with electric welders. With their new update they should make a short process with everything hostile.
- Glue Gun: shots a fluid toxic glue, which hardens and sticks to surfaces with time

Not all of these weapons may make it into the final game, some entries in the list are more serious than others.

Enemies

Different enemy types should be added to add variety to gameplay. The main priority when coming up with enemy types is how they can challenge the player in a different way, by forcing them to move, aim difficult shots or approach the game in an entirely unique way.

The current list of enemy archetypes is as follows:

- Grunt: Melee focused enemy that will attempt to rush the player and hit them, dealing damage. This is the most basic enemy and common enemy. A zombie-like creature could work well here.
- Armored Grunt: Grunt upgraded with very robust armor plates and equipped with a shield with which they try to block the players bullets.
- Ranger: This enemy has access to a ranged attack and will try to position itself to shoot the player. The projectile from the enemy should be telegraphed and possible to dodge on reaction. An example could be a Spitter of some kind that throws acid from its mouth.
- Drones: Flying machines that shoot projectiles and are difficult to shoot. Vulnerable to being disrupted by external forces.
- Nightmares: creatures with several limbs, able to crawl on all surfaces. An arachnophobic's worst nightmare.

Procedures and Player Mechanics

In this particular project, the primary emphasis will not be placed on novel movement and aiming systems of the player. Consequently, we will implement rudimentary first-person shooter mechanics, utilizing basic WASD keys for locomotion, spacebar for jumping, and a designated key for executing close-quarters combat. Due to the limited spatial dimensions of the environments, telescopic aiming abilities will be omitted. Nonetheless, supplementary mobility features, such as double-jump mechanics, will be subjected to iterative play-testing and evaluation for potential integration. The player will have a health bar, the enemies will not: their health will be signaled with other visual cues (like another texture, slower movement speed etc.).

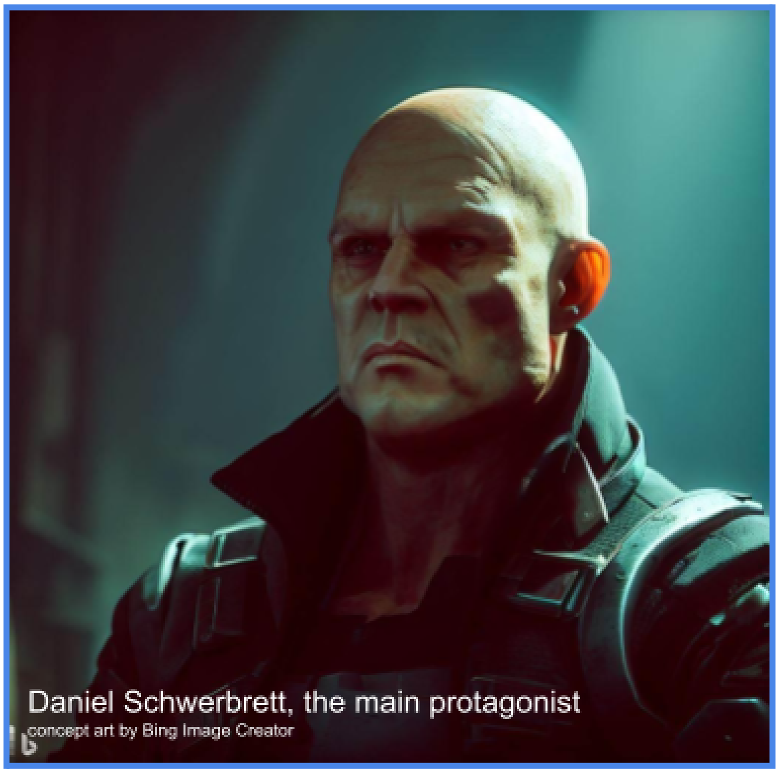
Story

You are Daniel Schwerbrett, a technically talented space janitor working at a dystopian, giant Roller Coaster driving in space; It is reserved for the wealthiest only, built by the famous billionaire Mrs Melon Busque. It is a luxury line with entertainment functions; a drive takes several weeks.

You went to sleep in your servant wagon when suddenly the alarm siren for radioactivity started ringing… a strange storm hit your ship.

While you got protected by your windowless wagon the other passengers had less luck: Deformed creatures start to attack you; defend yourself with everyday weapons until you get to the security room.

Your only hope is to get alive to the command center positioned at the end of the roller coaster in order to stop it and drive safely home.



Daniel Schwerbrett, the main protagonist

concept art by Bing Image Creator

2. Technical Achievement

Introduction

One of our main features is going to be the incorporation of the erratic movement of the roller coaster. However, this introduces a major technical hurdle: It is about designing a sound, intuitive and interactable system that responds to the change of the environment and the player's actions in a (physically) plausible way. The rigid-body class of objects like cups, popcorn, etc. is easy to simulate thanks to the built-in Unity's/Nvidia's PhysX. But the movement of the player and enemies needs to be handled more sophisticatedly.

Our solution is to use Unity's ML-Agents, a framework that utilizes modern Reinforcement Learning algorithms and provides a powerful interface to Unity. Using PPO it gives us the ability to train the character rag dolls of the various enemies to move, react, attack and die in a physical way that introduces an almost infinite amount of intractability with the player compared to hard-scripted reactions.

This supports the paradigm of the game designer as planner of interactive elements; but the player produces the game play by themself by interacting with them.
Training rag dolls with ML-agents to solve the movement of the agent is already built into the example project provided by Unity; what we will work on is more complex behavior (enemies keeping balance despite the train movement, etc.) through appropriate training solutions for the agents and potentially an incorporation of classical algorithms to the agent (like A*, PID-Controllers).
The player character controller will be only partly made with machine learning, since exact movements are required for a fast-paced action game.

The game industry has been using this concept of "Virtual Robotics" for years, but in combination with more sophisticated systems such as using motion captured data with matching learned balancing etc. (which is out of scope of this project)

Our secondary technical achievement will be the use of CFDs (most likely implementing XPBD/ FLIP) to convey artistic effects like blood, beverages etc. but also to enable certain puzzles or other Interactions. Best case will be to optimize it with parallel programming either with Unity's DOTS or Compute Shaders.

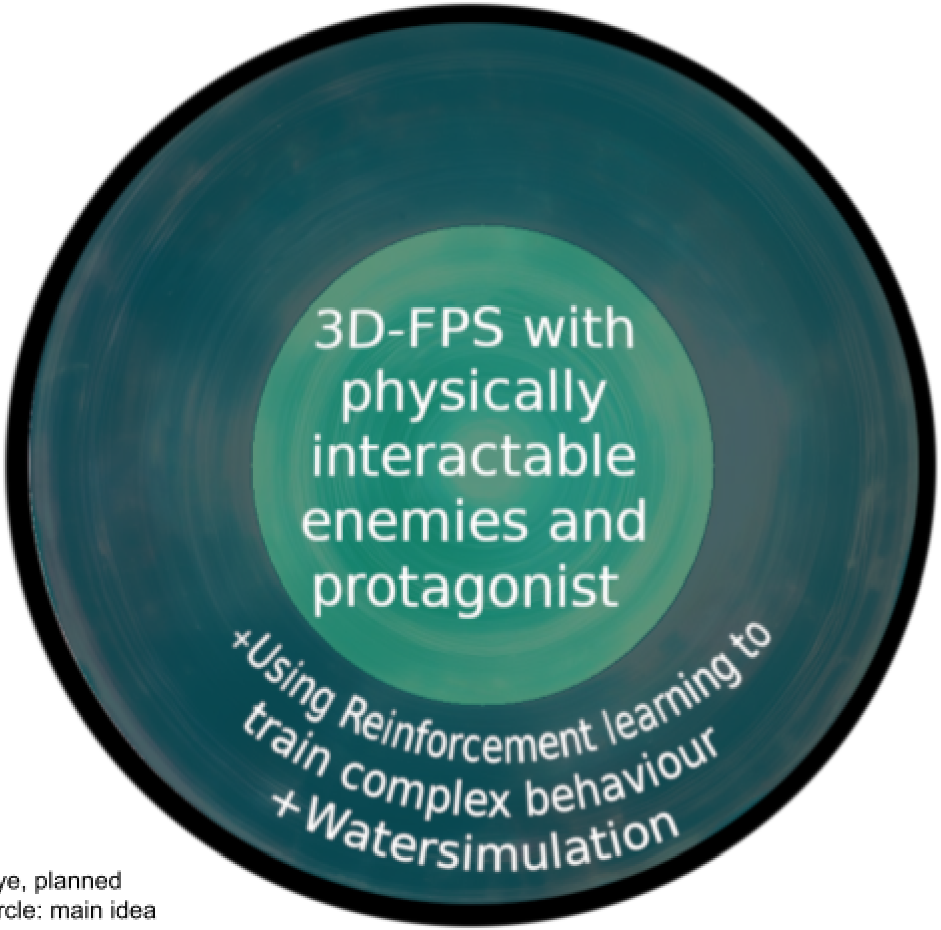
DOTS is a new framework that enables writing extremely performant code by default. It relies on three pillars:

- Use of the ECS programming pattern and data-oriented systems to optimize memory access and minimize cache misses.
- Burst compiler translates IL/.NET bytecode into highly optimized native code using LLVM
- Jobs system makes multithreaded code extremely easy and automated

According to Unity, DOTS is faster than their vectorized/highly optimized C++ code.

The downside is the almost total incompatibility with standard Game Objects and the relative lack of support and features (basically only basic physics and rendering are built in).

3. “Big Idea” Bullseye


3D-FPS with physically interactable enemies and protagonist
+Using Reinforcement learning to train complex behaviour
+Watersimulation

Bull's eye, planned
Inner circle: main idea
Outer circle: main technical achievements

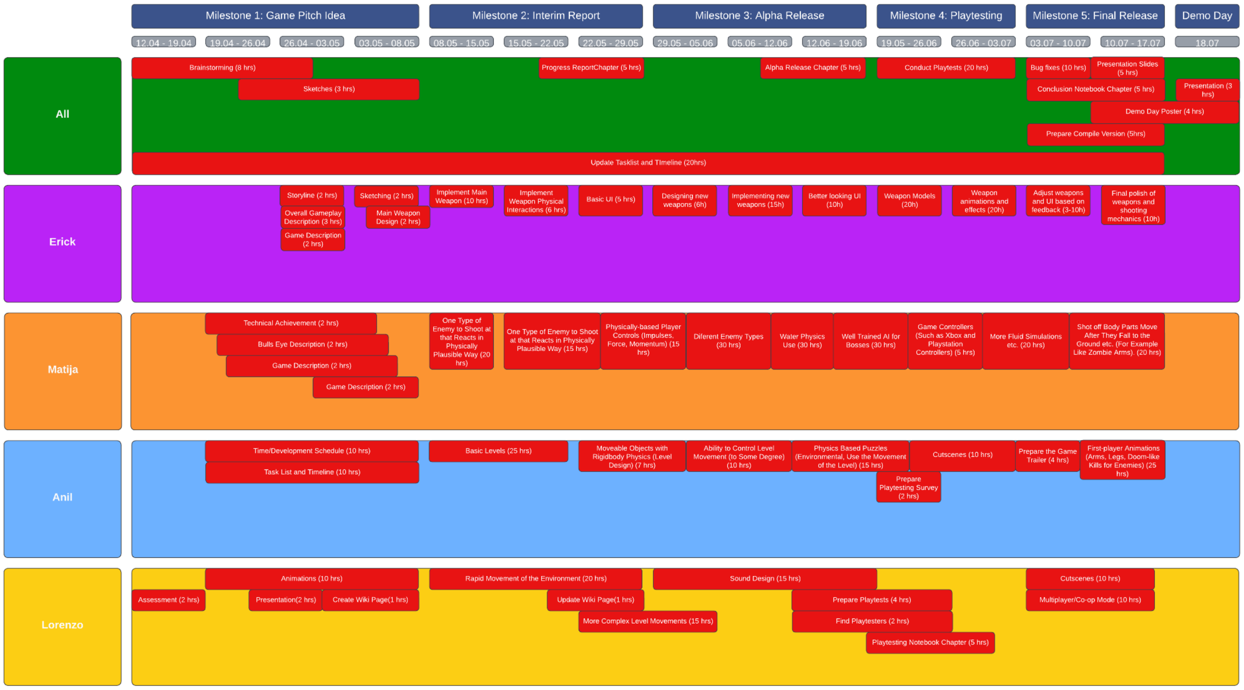
4. Development Schedule

Layered Development Description

- Functional Minimum
  - One type of enemy to shoot at that reacts in physically plausible way
  - Basic gun play

  - Basic levels
  - Rapid movement of the environment (only a few like e.g. rotation)
  - Rigidbody physics
  - Keyboard and mouse control
- Low Target
  - More complex level movements (g-force, free fall, sudden turns, acceleration)
  - Weapons have accurate knockbacks and recoils (such as rocket jumps)
  - Physically-based player controls (impulses, force, momentum)
  - Ability to control level movement (to some degree)
  - Basic UI
- Desirable Target
  - Sound design (movement of the roller coaster, music, weapon sounds etc.)
  - Different Projectile weapons (push enemies, break things, cannon balls, spikes etc.)
  - Different enemy types
  - Water physics
  - Breakable objects (glass, boxes, destructible environment)
  - Nice looking UI
- High Target
  - Story and characters
  - Physics based puzzles (environmental, use the movement of the level)
  - Good weapon effects and animations
  - Nice art and assets
  - Well trained AI for bosses
  - Game controllers (such as Xbox and Playstation controllers)
- Extras
  - Cutscenes
  - Multiplayer / co-op modes
  - First-player animations (arms, legs, doom-like kills for enemies)
  - More fluid simulations etc.
  - Shot off body parts move after they fall to the ground etc (for example like zombie arms).
  - Weapons that use fluid dynamics like a water jet gun etc.
  - Weapon progression such as adding mobility skills to weapons (crossbow provides a hook etc.)

Task List and Timeline



Tasks	Developer	Time
Brainstorming	All	10
Game Description	Erick	6
Storyline	Erick	2
Technical Achievement	Matija	4
Bullseye Description	Matija	6
Overall Gameplay Description	Anil	2
Time/Development Schedule	Anil	10
Task List and Timeline	Anil	4
Assessment	Lorenzo	2
Updating the Wiki Page	Lorenzo	2
Presentation Slides	Lorenzo	2
One Type of Enemy to Shoot at that Reacts in Physically Plausible Way	Matija	20
Basic Gun Play	Erick	10
Basic Levels	Anil	25

Rapid Movement of the Environment (Only a Few Like e.g. Rotation)	Lorenzo	15
Moveable Objects with Rigidbody Physics (Level Design)	Anil	7
Keyboard and Mouse Control	Matija	15
More Complex Level Movements (G-force, Free fall, Sudden Turns, Acceleration)	Lorenzo	10
Projectile Weapons (Push Enemies, Break Things, Cannon Balls, Spikes etc.)	Erick	20
Physically-based Player Controls (Impulses, Force, Momentum)	Matija	15
Ability to Control Level Movement (to Some Degree)	Anil	10
Mutual Critiques	All	1
Determining the Art Style	All	10
Determining the Gameplay Demonstration Content	All	2
Prototype Preparation	All	4
Feedback Consideration	All	1
Iterating over the Game Idea	All	2
Prototype Notebook Chapter	All	8
Updating Time/Development Schedule	All	4
Updating the Wiki Page	Lorenzo	1
Gameplay Demonstration	All	1
Presentation Slides	All	1
Conduct Playtests	All	20
Prepare Playtests	Lorenzo	4
Sound Design (Movement of the Roller Coaster, Music, Weapon	Lorenzo	20

Sounds etc.)		
Weapons Have Accurate Knockbacks and Recoils (Such as Rocket Jumps)	Erick	15
Different Enemy Types	Matija	30
Water Physics	Matija	30
Breakable Objects (Glass, Boxes, Destructible Environment)	Lorenzo	10
Progress Report Chapter	All	5
Updating Time/Development Schedule	All	
Updating the Task List and Timeline	All	20
Updating the Wiki Page	All	6
Gameplay Demonstration	All	10
Presentation Slides	All	5
Story and Characters	Anil	10
Physics Based Puzzles (Environmental, Use the Movement of the Level)	Anil	15
More Unique Weapons (Gravity Gun, Saw Blade, Bombs, etc)	Erick	20
Nice Art and Assets	Erick	10
Well Trained AI for Bosses	Matija	30
Game Controllers (Such as Xbox and Playstation Controllers)	Matija	5
Alpha Release Chapter	All	5
Updating the Task List and Timeline	All	20
Updating the Wiki Page	All	
Gameplay Demonstration	All	1
Cutscenes	Anil	10
Multiplayer / Co-op Modes	Lorenzo	10

First-player Animations (Arms, Legs, Doom-like Kills for Enemies)	Anil	25
More Fluid Simulations etc.	Matija	20
Shot off Body Parts Move After They Fall to the Ground etc. (For Example Like Zombie Arms).	Matija	20
Weapons that Use Fluid Dynamics Like a Water Jet Gun etc.	Erick	20
Weapon Progression Such as Adding Mobility Skills to Weapons (Crossbow Provides a Hook etc.)	Erick	40
Playtesting Notebook Chapter	Lorenzo	5
Email to the Supervisors the Contributions	All	
Prepare Playtesting Survey	Anil	2
Find Playtesters (Minimum of Five Participants for Playtesting)	Lorenzo	2
Updating the Wiki Page	All	
Presentation Slides	All	
Conclusion Notebook Chapter	All	10
Prepare the Demo Day Poster	All	4
Prepare the Compiled Version of the Game	All	5
Prepare the Game Trailer	Anil	4
Updating the Task List and Timeline	All	20
Updating the Wiki Page	All	1
Demo Day Presentation	All	12
Presentation Slides (Final Release Slides)	All	6

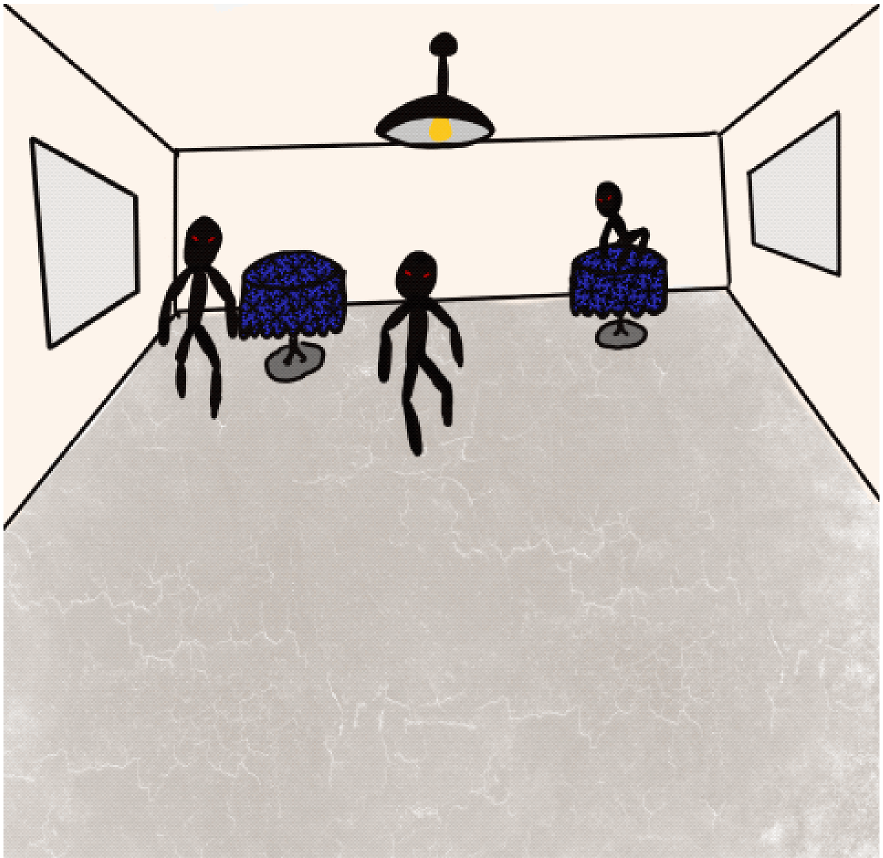
5. Assessment

The game offers a unique and thrilling gaming experience that combines fast-paced shooter action with physics-based mechanics. Players will be thrilled to press the trigger, since the weapons feel strong and impactful and change the environment and the enemies. The main selling point though, is the enemy movement, which should adapt to outside influence and look as realistic as possible. Furthermore the change of the game world and the constant battle against gravity should keep it challenging and fun. While shooter veterans should have an easier time picking the game up, the plausible physics make it so everybody feels comfortable and immersed in the world very quickly. The icing on the cake will be the nice fluid visuals together with the cinematic scenes that automatically come from those disruptive movements of the environment. Which makes you feel like you're in the middle of a Hollywood blockbuster. The main points making the game a good game are the guns, the enemy movement and the constant change of the environment which should lead to great moments.

6. Game Art

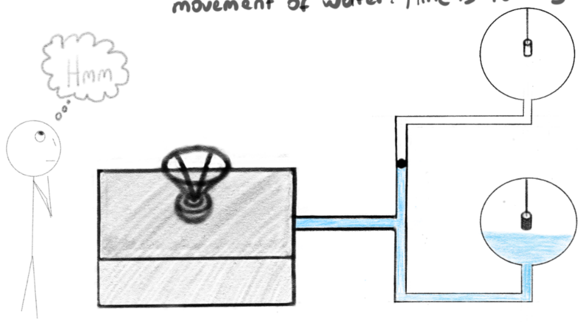
6.1 Concept Art / Inspiration (AI)

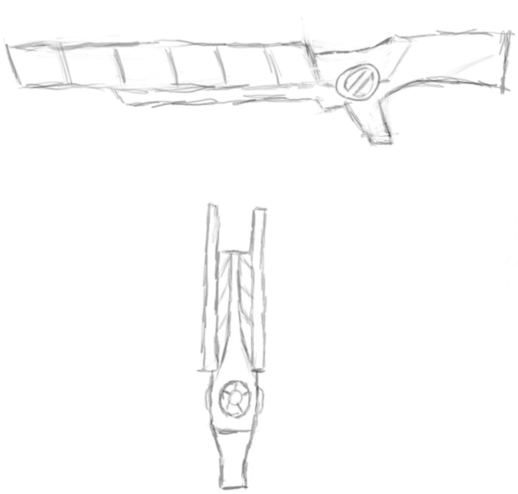
6.2 Sketches



our Animation showing the idea of the erratic movement of the “Grunt” and the anticipated separability of limbs.

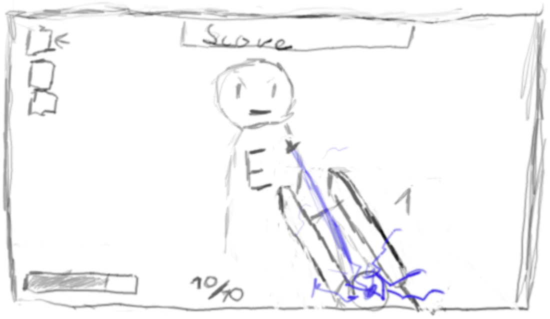
Task: Limited amount of water (water particles) must be filled in to the chamber while the coaster tilts left or right which affects the movement of water. Time is running short!





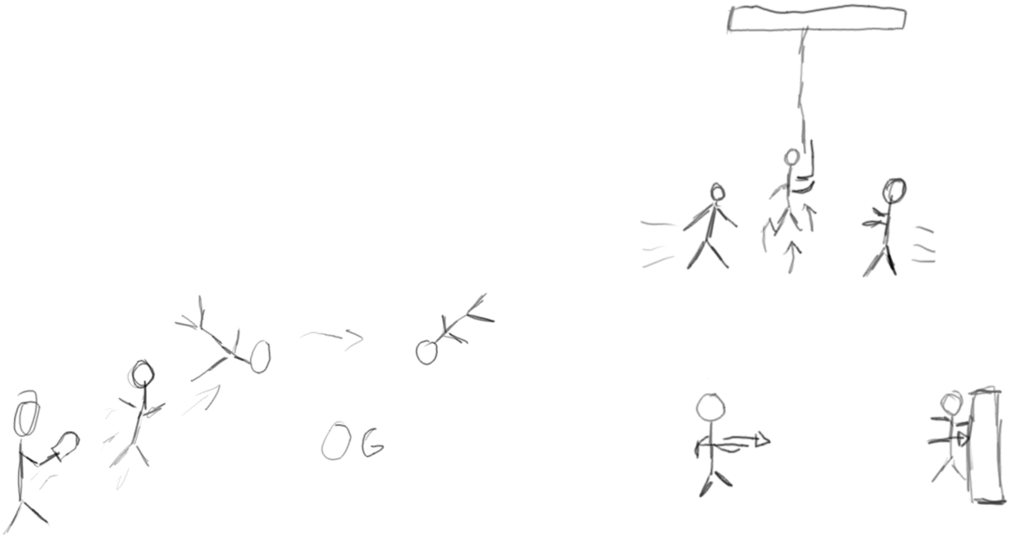
Fabricator Rifle rough design.

Shooting Fabricator Rifle with different levels of charge.

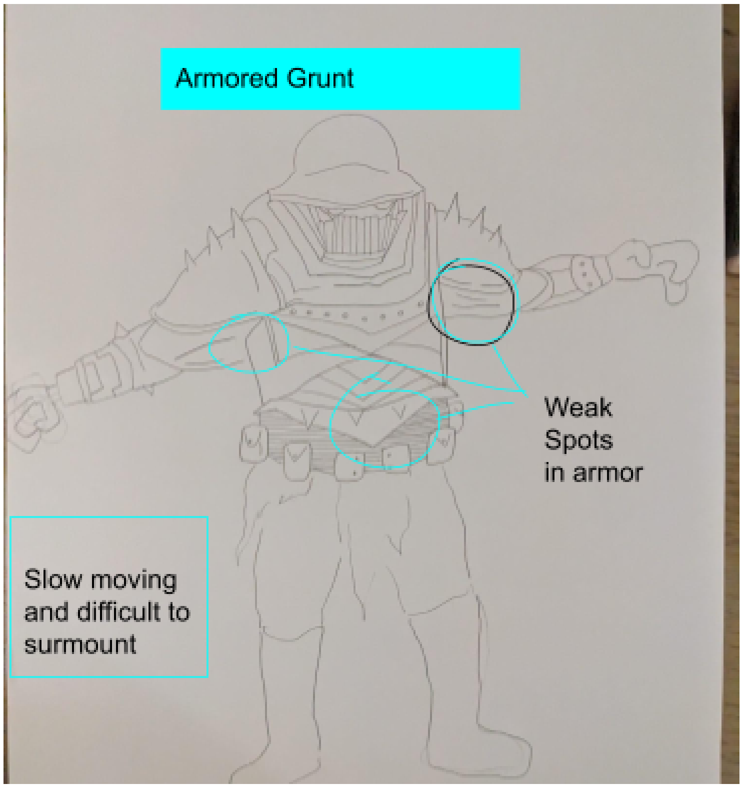


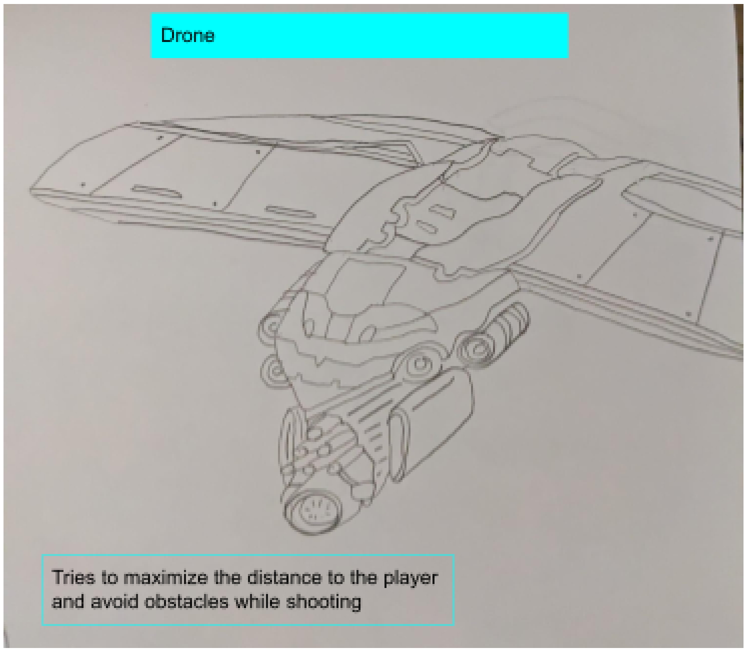
UI and first-person view.

Force Blaster with the standard and alternate fire modes.



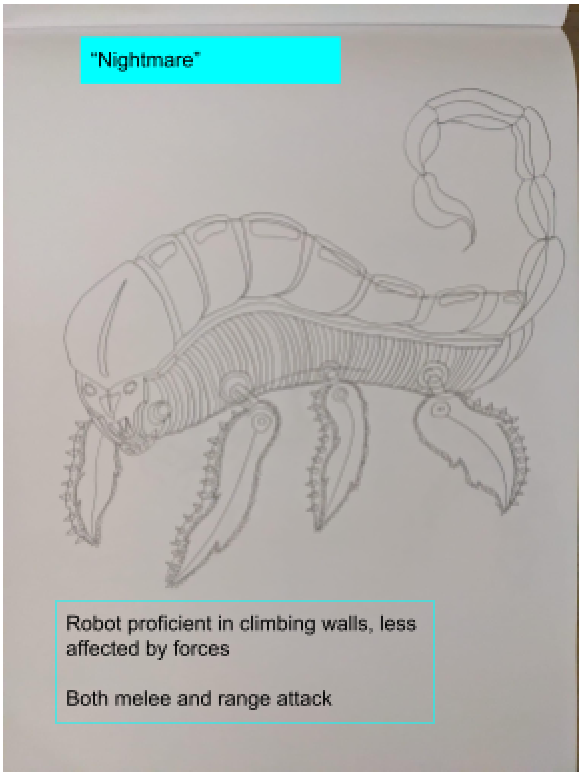
Gravity Gauntlet and Ballista weapon ideas.





Drone

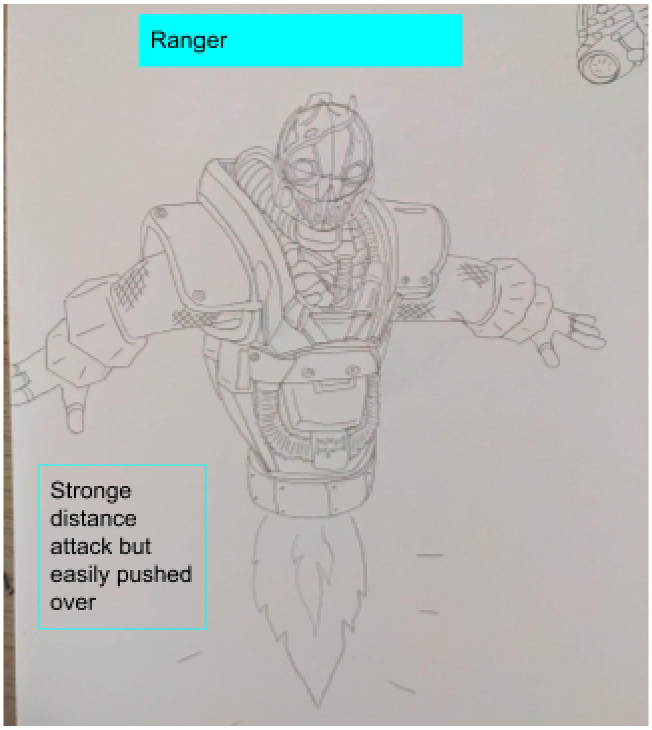
Tries to maximize the distance to the player and avoid obstacles while shooting

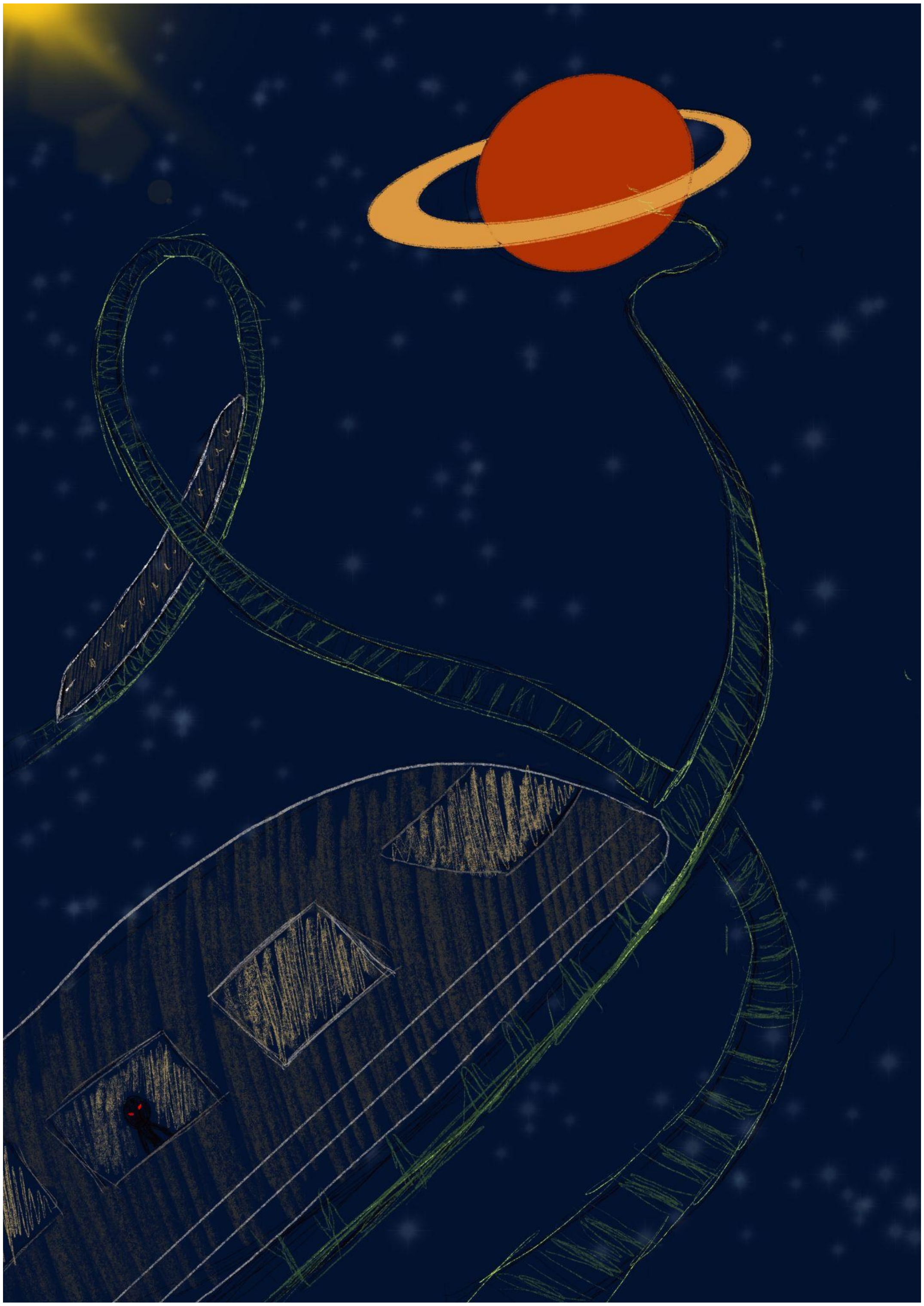


"Nightmare"

Robot proficient in climbing walls, less affected by forces

Both melee and range attack


Ranger
Stronge distance attack but easily pushed over



7. Game Idea Pitch Presentation