Team 2 – Escher’s 11

Game Programming Laboratory

That Failed Bank Robbery

„Should Have Planned It Better “

Simone Guggiari	– Producer + Game Designer + Gameplay and Engine Programmer
Nicolas Huart	– Game Designer + Level Designer + Menu Programmer
Alexander Lexus	– Game Designer + Physics Programmer + Procedural Content
Andreas Emch	– Game Designer + Physics Programmer + Effects Programmer
Xingze Tian	– Game Designer + Audio Engineer + Graphics Programmer

Chapter 1. Formal Project Proposal

1.1. Game Description

1.1.1. Logline

Two teams of clumsy robbers inside unlikely vehicles must collect money and valuables scattered all over Credit Suisse and disrupt their opponents before the police arrives after a badly timed explosion sabotaged their master plan.

1.1.2. Overview

'That Failed Bank Robbery' is a competitive local multiplayer game for 2 or 4 players in which two teams of thieves try to rob the same bank at the same time. The goal of the game is to collect the most money before the round ends or the getaway vehicle is destroyed. Each player in the game controls a vehicle and can collect money by opening vaults, cracking crates or collecting valuables present on the map. Players can also steal their opponent's loot by attacking them or their base. The game is set inside Credit Suisse and features a wide array of power-ups and different valuables and riches. The more money a player collects before bringing it to its base, the riskier the play becomes, as vehicles move slower and have impaired attack when loaded. This provides a fun layer of quick and dirty action on top of a more strategic game in which several tactics can bring to victory.

The game is developed by Simone Guggiari, Nicolas Huart, Alexander Lexus, Andreas Emch and Xingze Tian as a project for the Game Programming Laboratory offered at ETH Zürich in the Spring Semester of 2018 under the supervision of Prof. Robert Sumner.

1.1.3. Guiding Principles

We have three principles we want to base our game around. These are that our game should be:

- FUN
- SIMPLE
- BEAUTIFUL

We believe that to have a compelling game that can be enjoyed by players as soon as they jump in, we need to provide a simple game with an intuitive control scheme as well as clean interface. We also need a strong fun component, that provides a layer of strategy underneath the frenetic action-packed gameplay. The game rules should be easy to learn yet provide emergent gameplay to keep the game fresh. All of this should be wrapped in a game and a user interface that is both beautiful and attractive, as well as polished.

Therefore, our game should be nice to look at, easy enough to let players jump right in and fun enough to keep them coming back, and have some layers of depth and strategy to keep the player engaged even after playing a few rounds.

1.1.4. Background Story

The story begins in Escher's time, in 1856 as he was founding Credit Suisse. We see him building the bank from scratch with nothing but hard, honest work. Fast forward 163 years, the year is 2019 and Credit Suisse is now a giant in Swiss economy and attracts all kind of people and business. Two teams of robbers, after learning of Escher's history, decide to pay him homage by robbing his bank on the 200th anniversary of his birth. Escher's got rich with honest work, and now it's time to get rich with honest *dishonest* work! They plan a grand entry in the main building to blow up the vault, which is fully packed on this special day. However, due to bad planning, the explosives go off too early and money is scattered all over the bank. The police are on their way and the robbers have limited time to gather all the valuables and get out of there before it's too late. They jump into the first vehicle they can find, a forklift used to move stuff around, and duel to death to be the ones that get away with the most cash.

1.1.5. Design Decisions

1.1.5.1. Uniqueness

Our game strives to be unique by providing an innovative type of gameplay which is not found in most titles. We decided to stay away from platformers and shooters, and instead combine elements of action, racing and strategy games together, as well as some elements typical of party games. The final experience we try to accomplish is a fast-paced action racing game with combat elements, as well as different strategies and tactics that allow to reach the goal of the game.

1.1.5.2. mechanics

The main game mechanic allows players to control their vehicles around the map, to bump into other players and obstacles like a bumper car and destroy crates and stunning the opponents by performing a dash. By going over a valuable, it is collected and added to the inventory, and in a similar manner powerups can be picked up and used.

1.1.5.3. Settings

The game is set inside of Credit Suisse, in the aftermath of an explosion. Money is scattered everywhere, and more valuable items are found on the upper part of the map, near to the vault, jewelry and safe boxes. The map is rectangular, with the two bases (getaway vehicle) on the lower side.

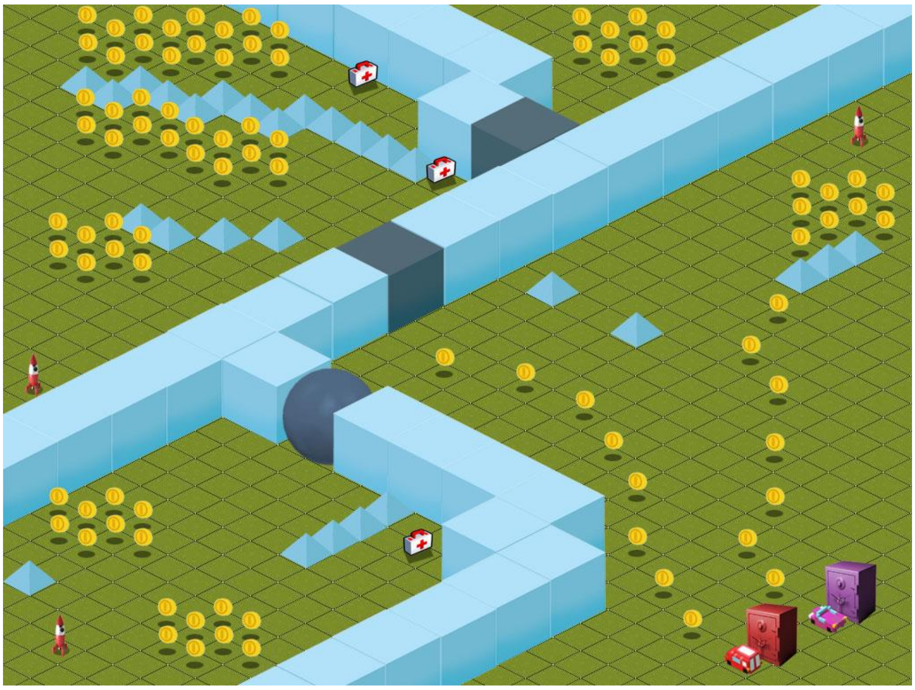
1.1.5.4. Look and feel

We plan to have a cartoonish, colorful and polished look to our game. The models will be low-poly, and this will allow us to have more detail in our scene. The models will fit the theme.

The objects and model are 3D, however most of the action will happen on a flat plane for simplicity and realism.

The screen is split into the number of players, each part showing a player's view (one camera for each player). There is also a small global map, where each player can see where the coins, the obstacles and the power ups are.

The camera looks at the map from an angle, smoothly follows the player and provides an isometric look. Here is a mockup of our view.



1.1.6. Audience, Platform and Marketing

The main audience for the game are casual player, such as those that will play the game at our booth at the end of the semester. The game is also geared towards players with more experience by providing additional challenges.

The game will be developed on windows and deployed on the Xbox One. It will be possible to play with the controllers on both windows and Xbox.

We plan to publish the game for both platforms at the end of the semester as our extra target.

We will be marketing the game mainly via a website that one of our members will setup, as well as other channels once the Lab is finished.

1.1.7. Game Elements

Here we describe some of the main elements that will be found in our game

1.1.7.1. Winning Conditions

The goal of each player is to maximize his profit by bringing money back to his base. The game ends when the time limit is reached or when one of the player manages to destroy his opponent's base. Players can combine different winning strategies for a common goal: collecting the most coins.

1.1.7.2. Characters

Different vehicles will be available in our game. The first one will be a forklift, which is well suited to move around objects and attack other vehicles with his claws. We will also implement two other vehicles for a total of 3, with different stats such as capacity, speed and attack. They will either be two other variations of forklifts, or more 'exotic' types of vehicles such as street sweepers, excavators or something similar as they can both carry material as well as attack.

Each player in the game controls one of these vehicles with specific life, speed and capacity. Each player starts at his own base.

1.1.7.3. Dash Attacks

Players can attack each other by performing a dash. When one player is hit by the other, with a certain probability he will become stunned and lose some of his money and life. During this time the other player can either collect the money and run away, or stick around for more action. When players have their whole life depleted, they lose all the coins they were carrying and respawn in their base after a small delay. Dash attacks can also be used to open crates but cannot damage bases or vaults.

1.1.7.4. Valuables

Our game will feature different types of valuables that can be collected and brought back to the base. Each will have a weight and a value (like the knapsack problem). Items found in the beginning of the map will have less value (such as bills or coins), while some found later (jewelry, gold) will have a higher ratio of value per weight, making them more interesting to bring back to base albeit riskier. Whenever a player goes through a place with a valuable, he collects if it is possible (still has inventory space). A player cannot collect more coins than the capacity of his vehicle allows. If he manages to go back to his base, his total score gets increased by the total value he just collected. If on the way, he loses all his life, he has to restart from his base and all his collected coins are lost, which will be scattered around the place where he died. The coins are initially randomly placed over the map and new coins will appear each time a player brings a coin back to his base or after some time, according to our procedural generation algorithm.

1.1.7.5. Power ups

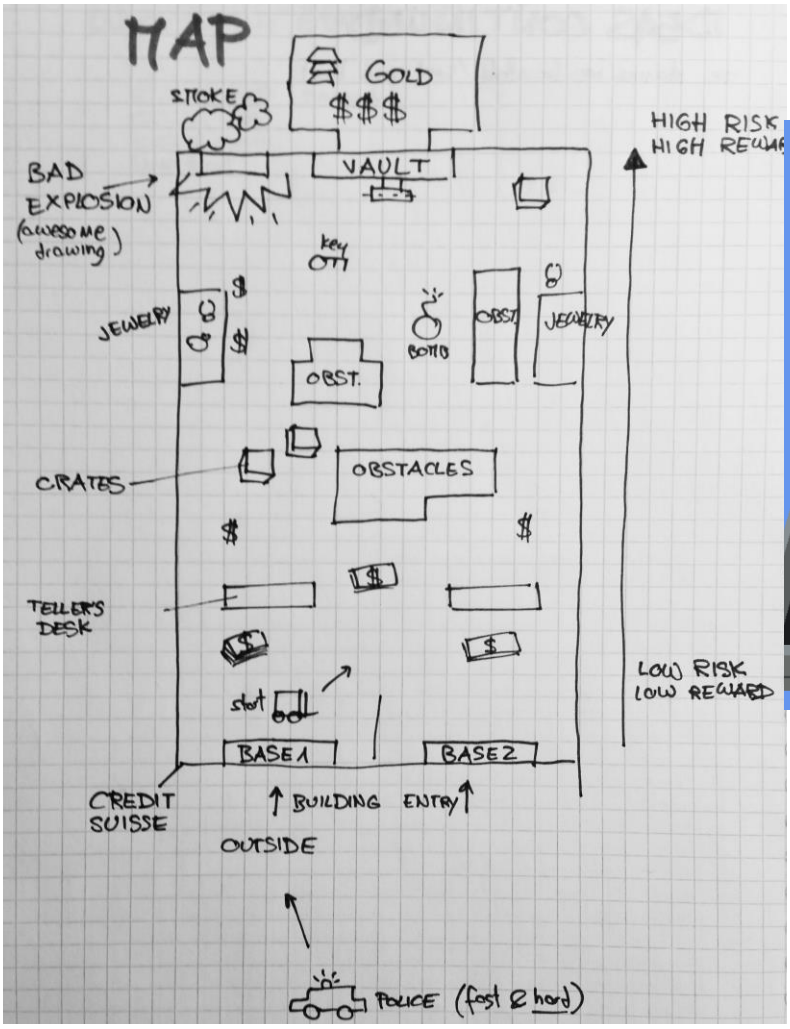
We plan to implement a few power-ups in our game. Power-ups change the property of each character and can be either temporary (e.g. speed boost) or permanent (e.g. capacity boost, medic kit). There is also a special power-up (bombs) that can be used to damage obstacles, open vaults or damage the enemy player's base. Keys will also allow players to get inside the vault. Other power ups might be used to improve the defense level of the base.

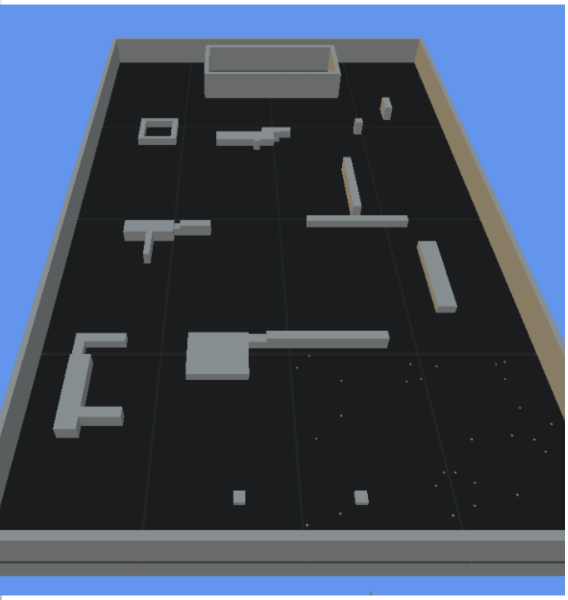
1.1.7.6. Crates and Vaults

Scattered in the map, we will have special crates that can be cracked by performing a dash towards them. Once cracked, the crates will reveal valuables and possible powerups that they had inside.
Vaults on the other hand will be found at the top of the map, in the riskiest place, and it will be possible to open them either with a key or a bomb. After this the player will be able to break in. Once opened, they will reveal a lot of valuables to be collected. This will be a very risky action that a player can choose to undertake, but the potential payoff will also be great as vaults will be filled with money and gold.

1.1.8. Map

Our map will be rectangular, with the entry of the bank on the bottom together with the team bases, and will contain everything that could be found inside a bank, from teller's desks to jewelry, safety boxes, obstacles and vaults. The drawing attached should give an idea of the layout. The further up the player decides to go, the riskier the play becomes.





1.2. Big Idea Bullseye

- *Collect procedural coins on the map or fight your opponent*
- *Strategic layer on top of fast-paced fun action*

BIG
IDEA
BULLSEYE
COLLECT PROCEDURAL LOOT AND FIGHT
STRATEGIC FAST-PACED ACTION

1.3. TECHNICAL ACHIEVEMENTS

1.3.1. RIGIDBODY SIMULATION

All the players will control a vehicle, which will be modeled as a rigidbody simulation. This includes a model for forces, torques, and velocities, making sure that no vehicle can go inside objects (collision detection and resolution). We will also implement friction and restitution coefficients to have cars bounce away when colliding with something, such as bumper cars do. The dash attack will also be modeled as a rigidbody simulation (imagine air hockey), and will strive to keep an arcade feel to our game controls overall by tweaking all the physics parameters.

1.3.2. PROCEDURAL GENERATION

We plan on implementing a procedural generation algorithm that will take care of placing coins, power-ups and obstacles. Doing so the map will always be different and thus it should be more fun and variated to play. Things to be procedurally generated include coins, powerups, crates and obstacles. Our procedural generation algorithm will be smart enough to try to balance the game, meaning that will spawn more valuables where the risk associated with them is proportional to the reward, will try to spawn and favor the player which is currently losing, and will decide when is the best time to spawn one of the most powerful powerups. We will also procedurally generate obstacles inside our map making sure every area is still accessible.

1.4. TEAM

In this section, we present the responsibilities that each member of our team will take upon himself.

1.4.1. SIMONE

Simone will take care of the game engine, as well as the organization of our software structure. He will mainly be involved in programming the gameplay features as well as simple modeling and graphic tasks. He will also make sure that the team is following the project schedule.

1.4.2. NICOLAS

Nicolas will be in charge of the design of the static part of the map as level designer. He will work on the menu design and implementation. He will also be involved in the character modelling and will be managing the library of assets.

1.4.3. ALEXANDER

Alexander is building with Andy the rigidbody simulation to handle the physics correctly and testing the game to make sure it’s fun to play. He is also in charge of the dynamic part of the map as placing special power ups, coins and other dynamic obstacles procedurally. In addition, he is helping out with the visual appearance of the level and the total project. In terms of side tasks, he is working on the slides and the Html pages for marketing the game.

1.4.4. ANDREAS

Andreas will implement the rigid body simulation as well as the collision detection part together with Alexander. This includes physical simulation, collision-handling, spinning wheels and friction, as well to integrate it into the game-play. Additionally, Andreas will be controlling visual effects such as shaders, lightning, particle effects, etc.

1.4.5. XINGZE

Xingze will be in charge of the sound effects (background music, sounds triggered by actions, start and end of game sound effects) and sound library. She will also take care of all the models, textures, light maps and other needed assets. In addition, she will implement lighting effects. In the final steps, she will be creating the trailer of the game and preparing slides with Alexander for the presentation.

1.5. Development Schedule

We will be following an agile schedule, that consists in small sprints of one week in which each team member has one well defined task to complete (or multiple smaller ones). We will have weekly meeting to discuss the current achievements and decide the tasks for the following sprint, as well as test the game, discuss new ideas and make sure we are on schedule.

1.5.1. Layered Task Breakdown

Our high-level view for the layered task breakdown is as follows: in the functional minimum, we plan to work mostly on tools for the engine and gameplay. This should give us a basic playable game. In the following phase, we extend gameplay functionality and add most of the features we want to have in our finished game. At the end of this stage we plan to have a fully working game that although is very rough, allows us to play. In the next phase (desired goal) we plan to focus mostly on graphics and menu, making the game something pleasant to look at, as well as including all the graphic assets and required menus. We include all the polishing in the last phase, such as audio, game balancing, and effects.

1.5.1.1. Functional Minimum

Our goal for the functional minimum is to have a basic game in which the player can control his avatar, move around in the level while picking up money and bring it to his base, with an isometric camera smoothly following the player. We want to have a simple HUD showing statistics such as time remaining in the round and money collected so far. No physical simulation will be present yet, and the level will just be a simple plane. The goal of this phase is to get everybody accustomed to working in MonoGame and have something we can start experimenting with. We also plan to start experimenting with technical stuff such as physics and rendering, producing a simple 2D rigidbody controller. We plan to be able to deploy this build to the Xbox already to make sure we don't run into technical issues later.

- Game engine:
    - Implemented game objects with 3d transforms and components
    - Implemented classes for camera, audio, input, scene, prefabs
    - Implemented classes for utility, time, coroutines, basic physics
    - Drawing 3d models as well as pipeline loading
    - Game running on Xbox
- Gameplay
    - Simple controller to move vehicles around
    - Camera following player smoothly
    - Pickup money, bring to base
    - Round time, winning condition
    - Simple HUD
    - Basic primitive level

1.5.1.2. Low Target

Our low target includes extending the game to allowing a second player to compete. This includes the addition of split screen functionality, ability to perform attacks and cause damage and money loss, respawn. We plan to implement a basic primitive level in which collisions work and to improve our player control to work with rigidbodies. We plan to expand the gameplay with almost all of the features, as well as expanding the HUD and start balancing the game to increase the fun factor.

- Game engine:
  - Physics fully implemented
  - Control implemented with rigidbodies
  - Procedural spawning
- Gameplay
  - 2 and 4 player split screen
  - Attacks
  - Damage/death/respawn
  - Stun/money loss
  - Level with primitives and collision

1.5.1.3. Desired Target

In this phase we start focusing on graphics and menus. We plan to have a working game already, now it's time to have a level with all graphics assets, menu that allows to select a starting avatar, add advanced gameplay functionalities such as power-ups, crates and vaults. We start distinguishing players by vehicle type with different stats such as speed and capacity. We will have some basic procedural money generation that makes the gameplay more unpredictable and thus fun. We will start having a small library of sounds we want to add as well as implementing most of them in the game.

- Gameplay
  - Vaults
  - Crates
  - Base damage
  - Power-ups
- Interface
  - Embellish and power-ups usage
  - Mini-map
- Menu
  - Menu windows implemented (main/play/join/options/…)
  - Menu transitions and effects
- Graphics
  - Added graphic assets
  - 1st level finished
  - 3 vehicles modeled
  - 2d art/title/tutorial
- Audio
  - Basic audio and music

1.5.1.4. High Target

For the high target we will mainly focusing in improving the existing game. The game's graphics will be enhanced with additional work on the visual effects, such as shaders and particle effects (e.g. for dust). To improve the replayability of the game, other levels will be added.

- UI and Menu
  - Additional polish
- Audio
  - All sounds and music

- Effects
    - Particles (dust/sparkles/explosion)
    - Post-processing
- Game
    - 2nd level finished
    - Balance and polish
    - Final trailer and presentation

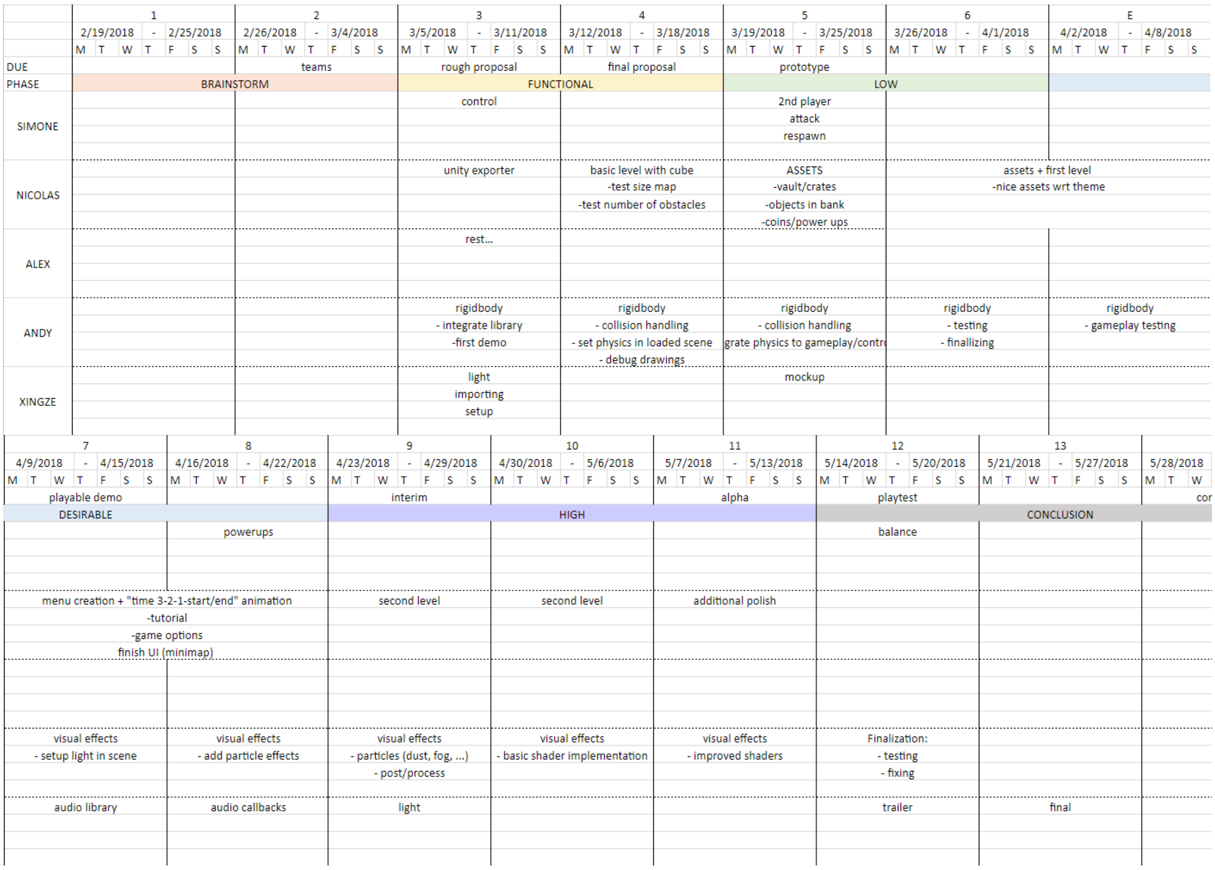
1.5.1.5. EXTRAS

Things we would like to have in our game but know that will not be able to implement in the limited timespan of one semester are a single player mode, in which the computer controls one team of robbers. This includes AI, navigation, decision making and strategic planning. Implementing a more advanced progression mode that is persistent between rounds of the game would also be nice. We would also love to be able to publish our game on different stores, start marketing it with a website and have one article written about our game.

- Single player
    - Enemy AI (decision and strategic planning)
    - Navigation
- Gameplay
    - Other powerups
    - Buyable from store between rounds
    - Persistent between levels
- Publish
    - Store
    - Website / article

1.5.2. TIMELINE

This is a timeline showing the whole semester divided into the 4 phases we described. It is a high-level overview of which tasks will need to be done when.



	1	2	3	4	5	6	E
	2/19/2018 - 2/25/2018	2/26/2018 - 3/4/2018	3/5/2018 - 3/11/2018	3/12/2018 - 3/18/2018	3/19/2018 - 3/25/2018	3/26/2018 - 4/1/2018	4/2/2018 - 4/8/2018
	M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S
DUE		teams	rough proposal	final proposal	prototype		
PHASE	BRAINSTORM		FUNCTIONAL		LOW		
SIMONE			control		2nd player attack respawn		
NICOLAS			unity exporter	basic level with cube -test size map -test number of obstacles	ASSETS -vault/crates -objects in bank -coins/power ups	assets + first level -nice assets wrt theme	
ALEX			rest...				
ANDY			rigidbody - integrate library -first demo	rigidbody - collision handling - set physics in loaded scene - debug drawings	rigidbody - collision handling grate physics to gameplay/contr	rigidbody - testing - finallizing	rigidbody - gameplay testing
XINGZE			light importing setup		mockup		

7	8	9	10	11	12	13	
4/9/2018 - 4/15/2018	4/16/2018 - 4/22/2018	4/23/2018 - 4/29/2018	4/30/2018 - 5/6/2018	5/7/2018 - 5/13/2018	5/14/2018 - 5/20/2018	5/21/2018 - 5/27/2018	5/28/2018
M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S	M T W T F S S	M T W
playable demo		interim		alpha	playtest		cor
DESIRABLE		HIGH				CONCLUSION	
	powerups				balance		
menu creation + "time 3-2-1-start/end" animation -tutorial -game options finish UI (minimap)		second level	second level	additional polish			
visual effects - setup light in scene	visual effects - add particle effects	visual effects - particles (dust, fog, ...) - post/process	visual effects - basic shader implementation	visual effects - improved shaders	Finalization: - testing - fixing		
audio library	audio callbacks	light			trailer	final	

1.5.3. Task List

This is a more in depth and accurate list in which tasks are subdivided by category (gameplay, engine, menu, interface, graphics, …) and by phase, as well as who will be making them. Each category has one or two responsible. As this excel table is quite big, please look at it on the next page. We didn't write an expected number of hours per task, but balanced them in such a way that we will need to follow the weeks marked above. In our estimate, each member of the group will be working around 25 hours weekly to accomplish all of the marked tasks.

1.6. Assessment

We will consider our game a success if we manage to get a fun, simple and polished experience out of it. We believe that the most interesting part of our game is the possibility of competition that will spark challenges to arise, as well as the possibility to coordinate and communicate to reach the common goal. If we also manage to have some emergent gameplay dynamic arise from our simple set of rules, it will be another victory for us, as well as having a beautiful to look at videogame that people have fun playing and that will make them keep the controller for "just another round".

			FUNCTIONAL	LOW	DESIRABLE	HIGH	EXTRA
			An ugly functioning 1player prototype in which a vehicle can be controlled around the level, pick up money and bring them to the base, showing a timer for time left and money accumulated	Add 2-player functionality, including attacks/lose coins and respawn, and more elaborate camera control. Implement collisions/rigidbodies and unity level exporter	Add powerups as well as base damage and vault. Put graphics and sound in place, start working on a second level and nice UI/MENU. Improve feel and balance	Polish game, add menu selection, balance, make fun,	TODO: add stuff
WEEKS			W3 - W4	W5-W6	WE-W8	W9-W11	
GAMEPLAY	control	S	simple joystick to move character		improve feel		
			2nd player	4 players	integrate physics		
			attack	leave cash			
	camera	S	one screen following player	camera rotation	camera fly through		
			split screen		implement splines		
	interaction	N+S	pickup money	base damage		broken effects?	bots
			bring to base	vault			navigation
			life/death/respawn	crates			
			lose coins				
	abilities	N+S	slower the more money	powerup usage		buy powerups	
				opening vault (timer/explosion)		different player stats	
TOOLS	engine	S	basic structure	be able to show video			
			gameobjects+components	scene loading?			
			transforms				
			drawing/loading				
	physics	D+A	unity working of our goal	collision detection			
			basic oncollisionenter callback	rigidbody 2d			
				basic primitives for collision			
	deploy	D+S	test working on xbox	continue deploying			
			make sure everyone compatible	merge branches			
			git working				
			windows developer account				
ASSETS	models	N	primitives exported	purchased assets & imported	modeled 3 vehicles		
			look at assets	1st vehicle modeled	modeled other unique assets		
	level	N	simple primitives 3D	more advanced primitives	integrated models		
			plane	scene made in unity	1st full level	2nd full level	
			unity export tool	improve procedural spawning	procedural generation implemented	procedural generation improved	
	images		basic images for report	rough drawing of what we need		implement 2d art	
					create tutorial image		
	artist		decide budget	start contacting artist	get 2d art		
				come up with art that we need			
INTERFACE	UI	N+S	timer for round	basic minimap	Embellish ui		
			money in base and carried		powerup UI		
			health / attack				
	menu	N	plan look and storyboard	basic gameover/restart	menu windows implemented	additional menus added	
					basic character selection	menu transitions	
POLISH	sound	X	get familiar with how it works	list of sounds	added sound	more sounds	
			play in 3d	library of sounds			
				selected music			
	light	X	lightmaps in unity baked	works in monogame	polish lighting		
			familiar with lights in monogame				
			pointlight/spotlight				
	particles	D	basic engine working 2d	working 3d	create needed particle effects		
				list needed particle effects			
	shaders/materials	X+D	how to use materials		have post processing working	implement post process stack	custom shaders
			mockup of final desired graphics				
DELIVERABLES	docs	-	rough proposal	create paper prototype			
			formal proposal	improve it			
	slides	X	create slides for presentation				
			create pictures/drawings				
	video	X	think of story	trailer script	basic trailer cut	polished trailer	
				trailer storyboard			
OTHER	fun	-	brainstorm	fun factor	balance		
			meetings/logistics				
	debug/test	-	read code and get familiar	always	always	do a lot	
	relations		talk with previous groups	talk with studio gobo			
			get code				
	marketing	A	think about webpage	setup basic webpage/blog	added graphics/text	polished/published	sell game on store
						webpage finished	get some website to talk about