Mechanica

“For Mation”

Simon Dubé
Andrey Titov
Simon Jacques
Tri-Luong Steven Dien

Last Update
2018-04-14

Table of Contents

Executive Summary

Mechanica is set in a post-apocalyptic future where machines are now so advanced that they can even feel emotions. After a violent revolution, machines finally took control over mankind turning human survivors into slaves. For their own amusement, machines are now using humans as gladiators in violent death battles taking place inside giant coliseums. Unwilling to fight on their own, humans can be mind-controlled by drones, forcing them to fight ones against the others.

Contrary to most death match, RTS or battle royale games where players control either a single character or send commands to a group of characters, *Mechanica* brings a new twist to the genre in the sense that players directly control a single drone which then controls characters inside of its “zone of influence”. This offers an interesting feel for the players who are used to have full control over their character as they now have a more indirect control over their minions, requiring them to be very careful with their movements as if they get to far from them, they will behave on their own once more.

The drone and formation control offers a unique and different way to play action games. This presents an additional and interesting challenge for the players. This also opens the door for interesting formation behaviours. The inclusion of rogue humans which can be incorporated to the formation mid-game to increase a player’s power can also really disrupt standard multiplayer play and represent an additional and original threat for the players.

Overview

Mechanica is a multiplayer original top-down shooter taking place in a futuristic setting. After numerous breakthroughs in artificial intelligence, machines have become self-aware and have undertaken a violent uprising against mankind. After a terrible defeat, human survivors were captured by the machines and turned into gladiators. This game gives players control over one of the drones that are used to orchestrate these gladiators fights. As humans don't want to fight against each others on their own, machines implanted humans a mind-control system giving drones total control over a human's mind given that he/she is within a small radius around it.

During these fights, it's actually the drones that are fighting against each others trying to be the last one standing. However, being weapon less, drones must take control over the minds of one or many human gladiators thrown inside the arena and order them to pick up weapons and shoot down the enemy drones for them (different weapons have been carefully dispersed inside the arena). However, "free" humans (humans not controlled by any drone) can also perform actions on their own, like picking up weapons and shooting down drones. This forces players to pay close attention not only to the enemy drones, but also to the free humans running around and forming group of their own.

In standard gameplay, a player (drone) first try to recruit some humans, then run around to find them weapons (if they don't already have ones) and then try to destroy the other drones while making sure to stay alive.

In order to help drones recruit humans and to add a special twist to the fights, drones are also equipped with a stun grenade. This grenade only affects humans and serves as a great gap closer for machines when trying to recruit new humans. It can also be used as a tool to stun another drone's humans to significantly reduce its defenses.

An important aspect of this game is how players can manipulate the humans formation their drone have control over. As such, within a drone radius of influence, a

drone can determine which formation (which is what inspired our tagline) it wants it's humans to take. Players can also customize their formation in many ways by spreading the formation more or less and by locking the formation rotation (humans will still always look at the cursor position). This makes the game very interesting and offers a lot of possible ways to approach each different situations a player can encounter during play. It also gives players the opportunity to use formations in creative ways.

Another critical feature of *Mechanica* is the free humans AI. Not only do the players have to worry about other players, but they also have to pay attention to the double edged sword that represents the free humans. These characters present an interesting behaviour which requires players to really keep an eye out an make sure not to be caught in a situation where he/she has to fight a lot of humans with a smaller group under its control. They also create a surprising win condition for the other players in the sense that they could win without ever encountering an enemy drone if his/her opponents start taking to much risks.

Contrary to our previous projects (COMP 376), a core part of this game in the inclusion of network features to allow two to four players to compete against each other over the network and also allows an unlimited amount of viewers (non-player which have a camera showing the entire map). This reinforces the competitive aspect of the game. At first, we were not sure if we would have time to scale our game up to four players as it requires a very good data management, but we managed to pull it off. This enables a more interactive and fun experience for the players. However, it mostly creates a much more complex implementation for us, the programmers, as we have to take into account the network communication and synchronization of all the aspects of the game, which is far from being a simple task. This was actually the most challenging part of the project for us and led to a lot of headaches.

Throughout the map, humans are able to acquire different types of weapons including pistols, shotguns, submachine guns and assault rifles to diversify their tactics and behaviour. "Free" humans behave differently depending on their weapon at hand. Also, within a drone's zone of influence, an AI was implemented so that humans, based on whether or not a certain weapon will increase the formation

power, automatically leave their slot to go and pick up the different weapons. This enhances the human's behaviour and brings an interesting twist to the game. It also make the situation a lot more manageable for the players as if one of the characters within the formation dies and had a very strong weapon, another human will automatically go and pick it up instead.

Moreover, in a similar way to the currently popular "Battle Royale" games, *Mechanica* includes a "storm" mechanic. The storm is a game mechanic used to force players in a given region of a map by dealing continuous damage to them given that they do not stand outside of the storm (or in other words, inside the game area). This is a simple yet very efficient way to force players to fight at the later stages of the game.

Finally, *Mechanica* is an original multiplayer top-down battle royale shooter featuring twitch gameplay. It offers players the opportunity to use strategy with formations management, risk and reward evaluation for their engagements and decisions throughout the game. In addition of the other players, it also presents the players a new challenge in the form of an intelligent AI that players must be very careful about and play around.

Related Games

Hotline Miami

Publisher: Devolver Digital
Genre: Top-down shooter
Year: 2012-2015

Developer: Dennaton Games
Platforms: Microsoft Windows, OS X, Linux, PlayStation 3, PlayStation 4, PlayStation Vita, Android

We want both games to offer the same frenetic gameplay experience. Hotline Miami is a very punishing top-down shooter with very-fast paced action. As we plan our game to be multiplayer, we want to tone down this pace a little by having 2 main gameplay sequences. The first one where players are carefully exploring the map to gain as many humans as possible, and the second one, closer to *Hotline Miami*, where players fight head-on. We believe that having this first phase offers a richer experience than *Hotline Miami* by adding more of a strategic aspect to the game.

Agar.io

Publisher: Miniclip
Genre: Top-down action
Year: 2015

Developer: Matheus Valadares
Platforms: Browser, Android, iOS

In both Agar.io and Mechanica, players try to gain strategic advantage by collecting wandering NPCs. In Agar.io, players have to take quick decisions with how they move around by dividing their core and altering their formation. In our game, players can change their NPCs formations in order to alter their style of play from offensive to defensive. These changes in formation affect the risk and reward aspect of the game, just like in Agar.io. The aspect of collecting humans in Mechanica reflects the importance of collecting cells in Agar.io, where losing them allows other

players to collect them in their turn. As opposed to Agar.io, our game only features a limited number of players, but allows for long distance combat, which means enemy encounters are much more frequent. This adds another strategic aspect to the game, forcing players to always be aware of their surroundings.

PlayerUnknown's Battlegrounds

Publisher: PUBG Corporation, Microsoft Studios, Tencent
Developer: PUBG Corporation
Year: 2008 (PSP), 2017 (PS4)
Genre: Battle Royale
Platforms: Microsoft Windows, Xbox One, Mobile Phones

PUBG and *Mechanica* both have parts of the online battle royale game genre. The competing players start the game in different parts of the map and they have some time to explore the map and prepare for the fight (unless they find each other early on) before the actual combat begins. Both games would allow the players to loot the map and the combat would mainly be 1v1. However, PUBG places more emphasis on the survival part, while Mechanica places more emphasis on the gathering and combat mechanic.

Game Characters

Drones

Dev name: Drone

Created specifically for the coliseum battles, drones are the most important characters in the battle arena. Because humans are unwilling to fight each others on their own, drones can generate a short range force field allowing them to take control of the humans minds and to force them into fighting each others. However, the drones' main goal being to destroy all the other drones, it will mainly order humans under its influence to attack the enemy drones. With their mind-control ability, drones have complete control over the humans' minds. They can force the humans to move with them and even create specific formations when controlling multiple humans at a time. It also carefully distributes the available guns between them to insure it always remains as strong as possible. However, the force field being only active at the proximity of the drone, if the drone moves too far away from a human, the human will regain its senses and start to act on its own again.

The main goal of the drone in the battle arena is to destroy all the other drones by recruiting as many humans as possible throughout the battle arena and providing them powerful weapons. The destruction of the drone results in a loss in the game for the player controlling the destroyed drone.

The drones are equipped with small boosters allowing them to levitate over the humans and non solid obstacles (mostly water) throughout the arena. It moves faster than humans, but does not carry any lethal weapons. However, it is equipped with a regenerating stun grenade which can be used as a disruptor for the other drones' humans, but also as a gap closer to take control over the free humans more easily. Depending on the player, drones will have a different design.

Humans

Dev name: NPC

Humans, who used to rule the modern world, have been taken over by the machines. Now, they are forced to fight against each other to the death for the mere amusement of highly intelligent artificial systems. To insure their well behaviour, machines implanted chip cards into the humans brains to be able to mind-control them at any moment.

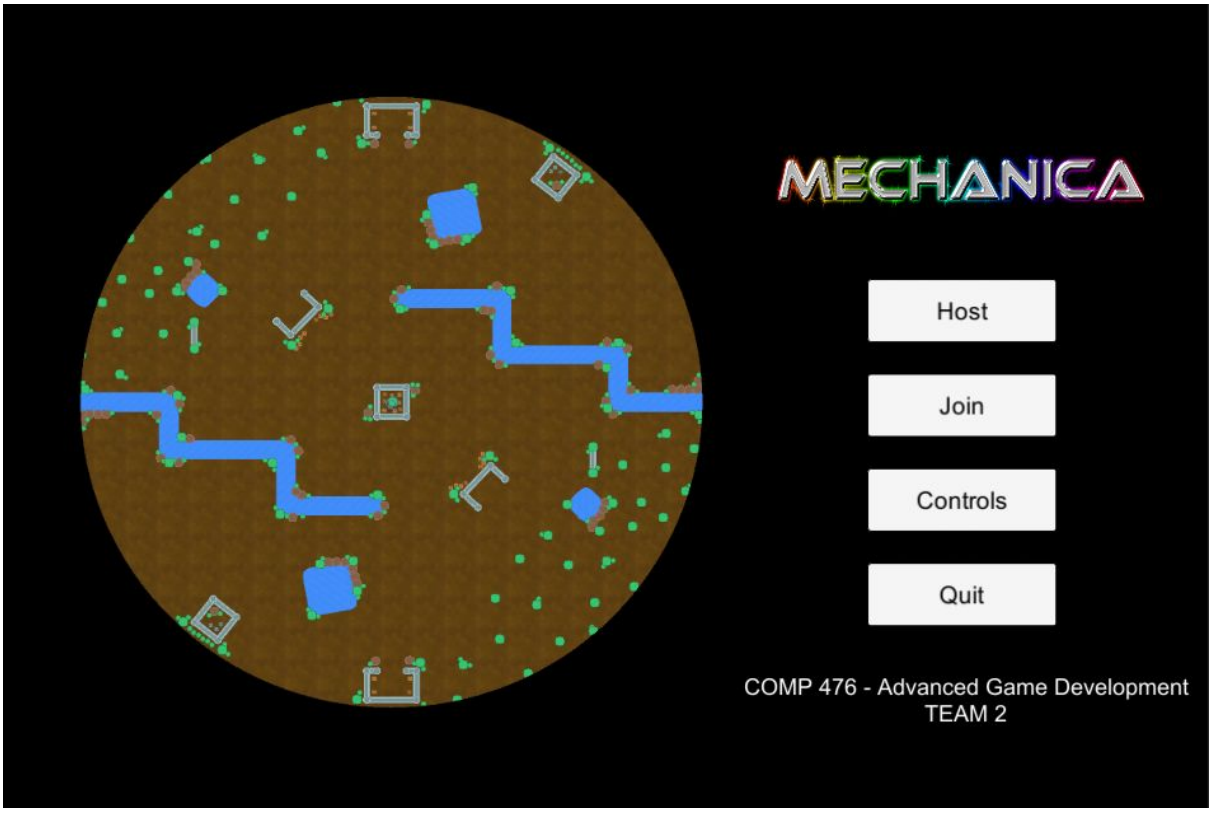
On their own, humans will do anything they can to survive against the drones. That said, if they meet other humans on the battlefield, they will join forces and try to survive for as long as they can, escaping their certain doom. Being on foot, they run more slowly than the drones can levitate. They also *cannot* cross any of the obstacles present on the battlefield. If they ever come across a drone and are confident enough to be able to take it out (based on the guns they or their group has), they will try to do so. However, they are much less resistant to bullets than drones are.

Under the control of a drone, they immediately fall in line and follow exactly the drone's orders. If they are asked to shoot, they shoot. If they are asked to run, they run. If they are asked to go and pick up a certain weapon, they do.

Contrary to the drones, humans can pick up weapons found on the battlefield and use them to their advantage. They can use them to kill other humans (which they will only do under the control of a drone) or to kill a drone.

User Interface Storyboards

Main menu

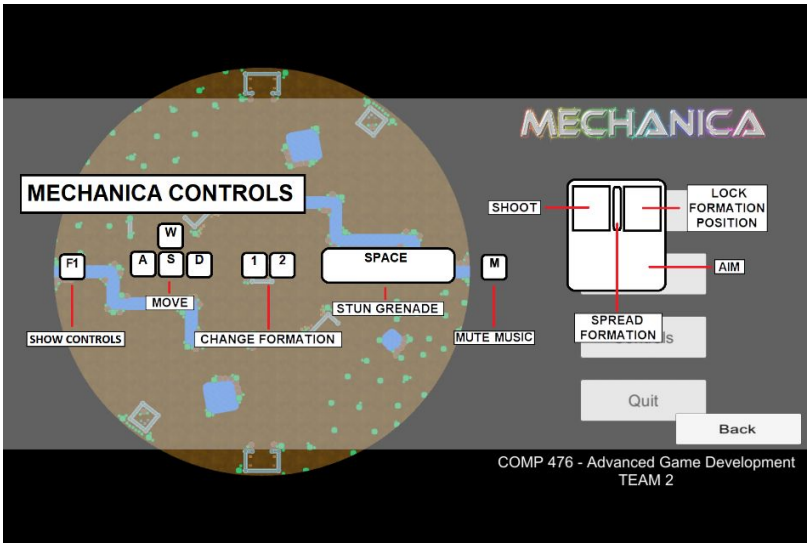


From the main menu, a player can either “Host” a game, “Join” a game, look at the controls (which can be seen at anytime in game by holding F1) or quit the game. By clicking “Host”, the player is transported to the “Lobby” screen whereas by clicking “Join”, given that another player is already hosting a game, the player is shown the following screen.

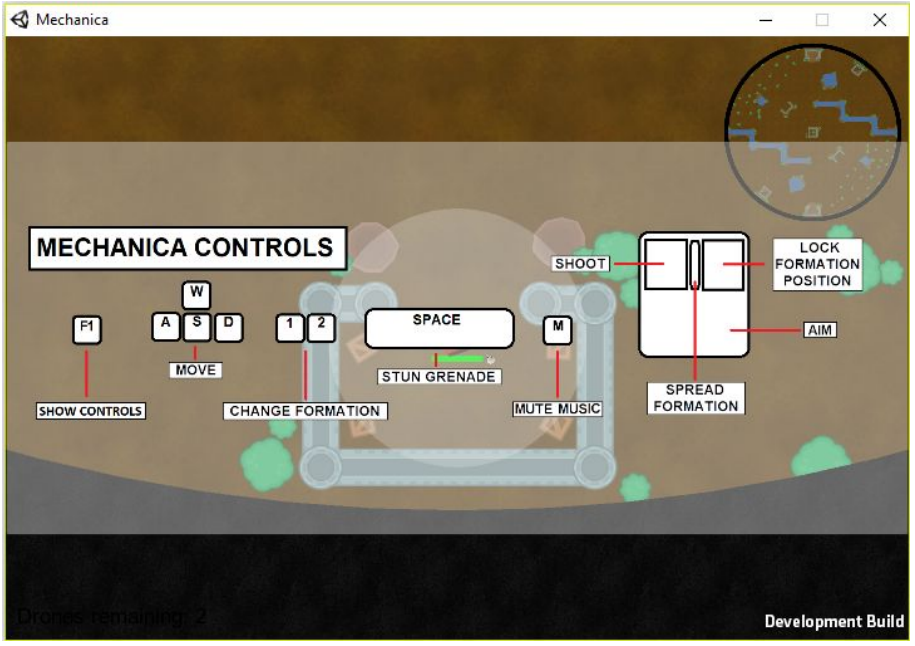
Join menu

Given that another player is already hosting a game, a player clicking “Join” will be presented with this screen from which he/she can join a game based on the host IP address, look at the controls or go back to the main menu. If no other player is hosting a game, the player will simply remain on the main menu screen.

Controls



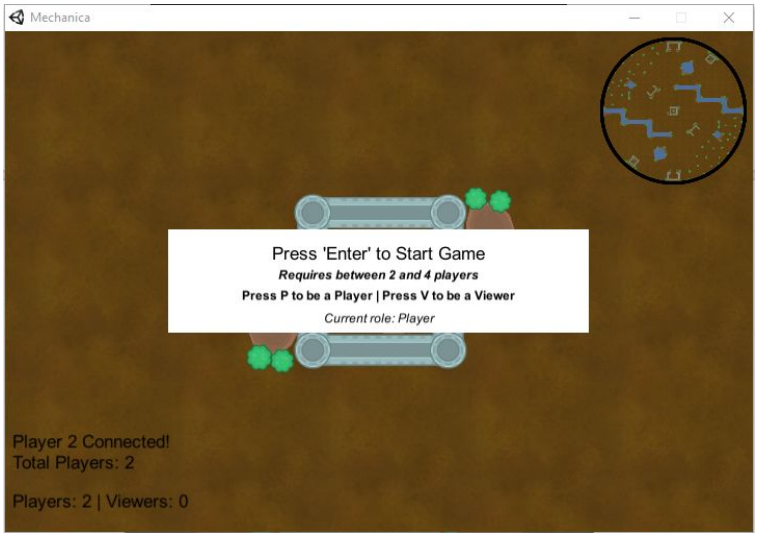
Controls from main menu



In-game controls (displayed by holding F1)

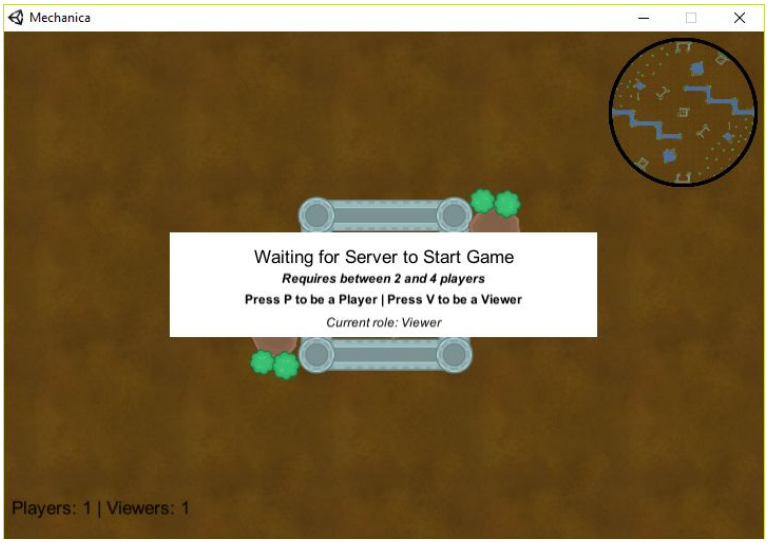
Controls screen serves as a simple way for new players to get a refresher on the game controls.

Lobby Screen



Upon hosting a game, the host is presented with the following screen.in the middle are displayed key pieces of information including how to start the game, the requirements to start the game, how to switch roles (between Player to Viewer) and the user's current role (player by default). In the bottom left corner, the host can also seen when players connect and disconnect from the game with clear messages. Under this is displayed the current amount of players and viewers in the game. The number of players must be between 2 and 4 for the host to be able to start the game. Pressing "Enter" when it is not the case will not begin the game.

The non-host players have a very similar screen displayed when joining the game, except they don't have the player connection messages and their main message reflects their non-host role.



Game UI

This screenshot displays all the possible information a player can have it his/her HUD during standard gameplay.

Health bars

Each elements that can be killed (drone and humans) have their own health bars. This provides fast information for the player in order to know how healthy are the different characters.

Stun Grenade indicator

The stun grenade indicator is a small sprite placed to the right of the drone healthbar. It allows the player to easily know when he/she has access to it and when it is on cooldown (transparent).

Informational Texts

At the beginning of the game and at some other key moments, a big red text appears in the upper half of the screen. It is used to tell the player in which phase they are (gathering or deathmatch), to inform them in how long will the EMP storm close in and to inform them when it actually starts closing in.

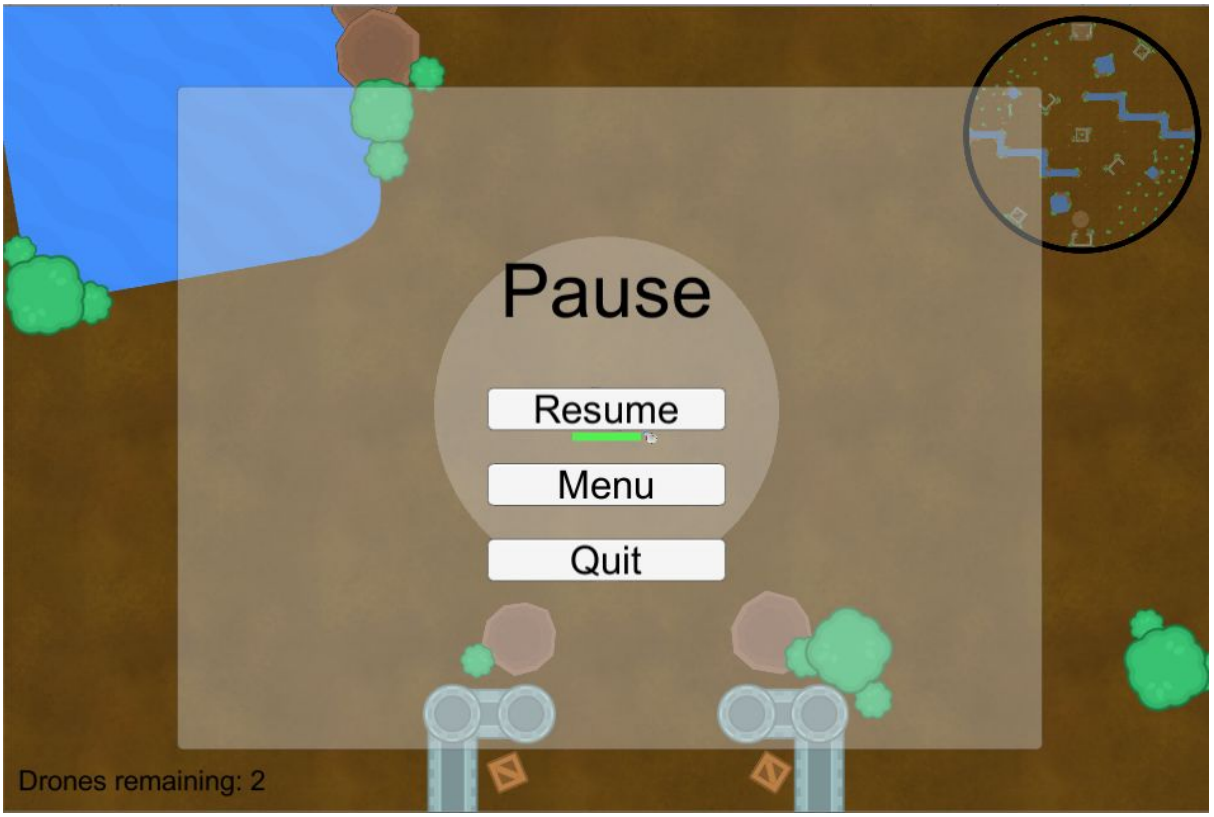
Minimap

The minimap is used to represent the player's position in real-time. This allows players to know where they are at a quick glance. On the minimap, the location of all players' drone and their respective zone of influence are also displayed to balance camping strategies. The minimap also rescale to only show the playable area when the EMP storm closes in.

Player count

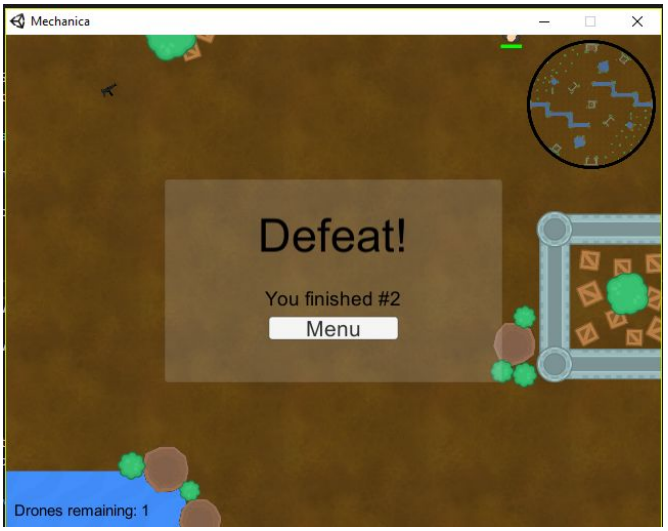
In the bottom left of the screen, a player can see how many players (drones) are still alive.

Pause menu



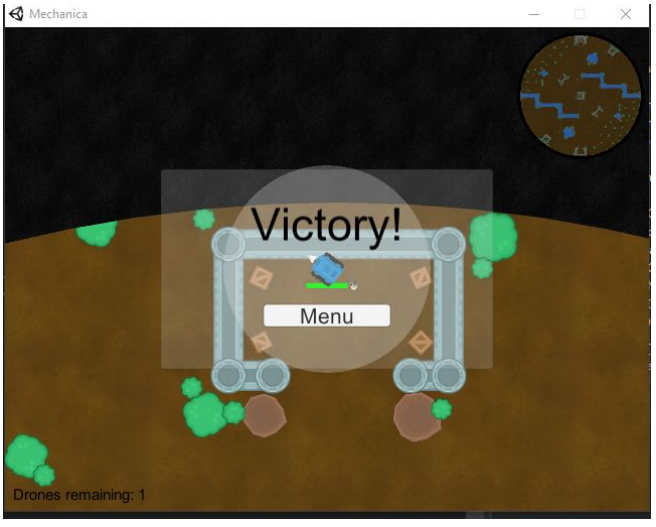
By pressing the Escape key, a player is presented with this screen allowing hier/her to resume the game, leave to the main menu or quit the game altogether. However, the game is still continuing in the background. This does not actually pause the game, just like in any other online game.

Defeat screen



Upon dying, a player is presented the defeat screen indicating his/her rank and from which he/she can return to the main menu.

Victory screen



Upon winning, a player is presented the victory screen from which he/she can return to the main menu.

Technology plan

Technologies used

Here is a list of the different technologies used in the creation process of our game.

Game Engine: Unity

Programming Environments: Visual Studio

Scripting Language: C#

Documents Storage: Google Drive

Documentation: Google Docs

Art: Kenney Assets, PngImg, OpenGameArt

Music: Undertale, The Prodigy

Sounds: Freesound (we used CC-0 licensed sounds)

Version Control System: GitHub

Networking: Unity Networking

Communication: Slack, Messenger

Hardware requirements

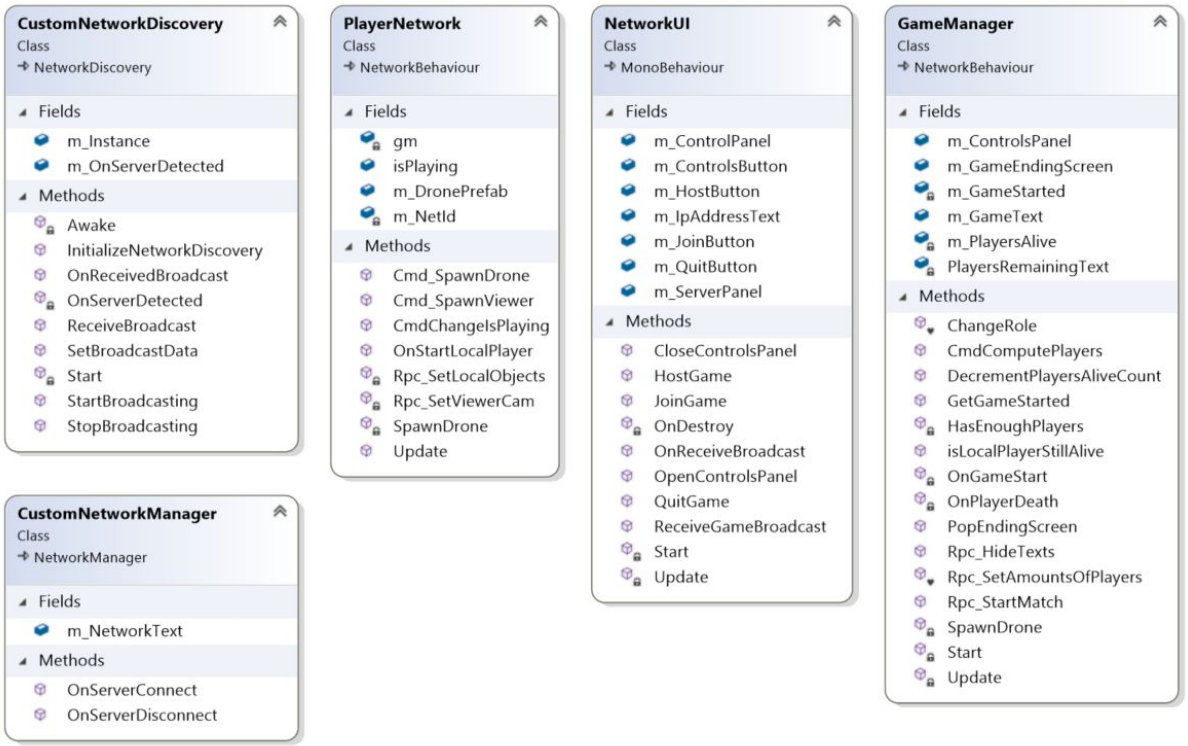
In order to play the game, one must own a computer with a mouse and a keyboard, which is common practice. Also, as this is a multiplayer game, one must also have a router, or be able to establish a peer-to-peer connection, in order to connect multiple clients together (at least 2) and play the game.

Software architecture

The software architecture will present each category of our classes that are closely related to each other and it will briefly describe what each of these classes does.

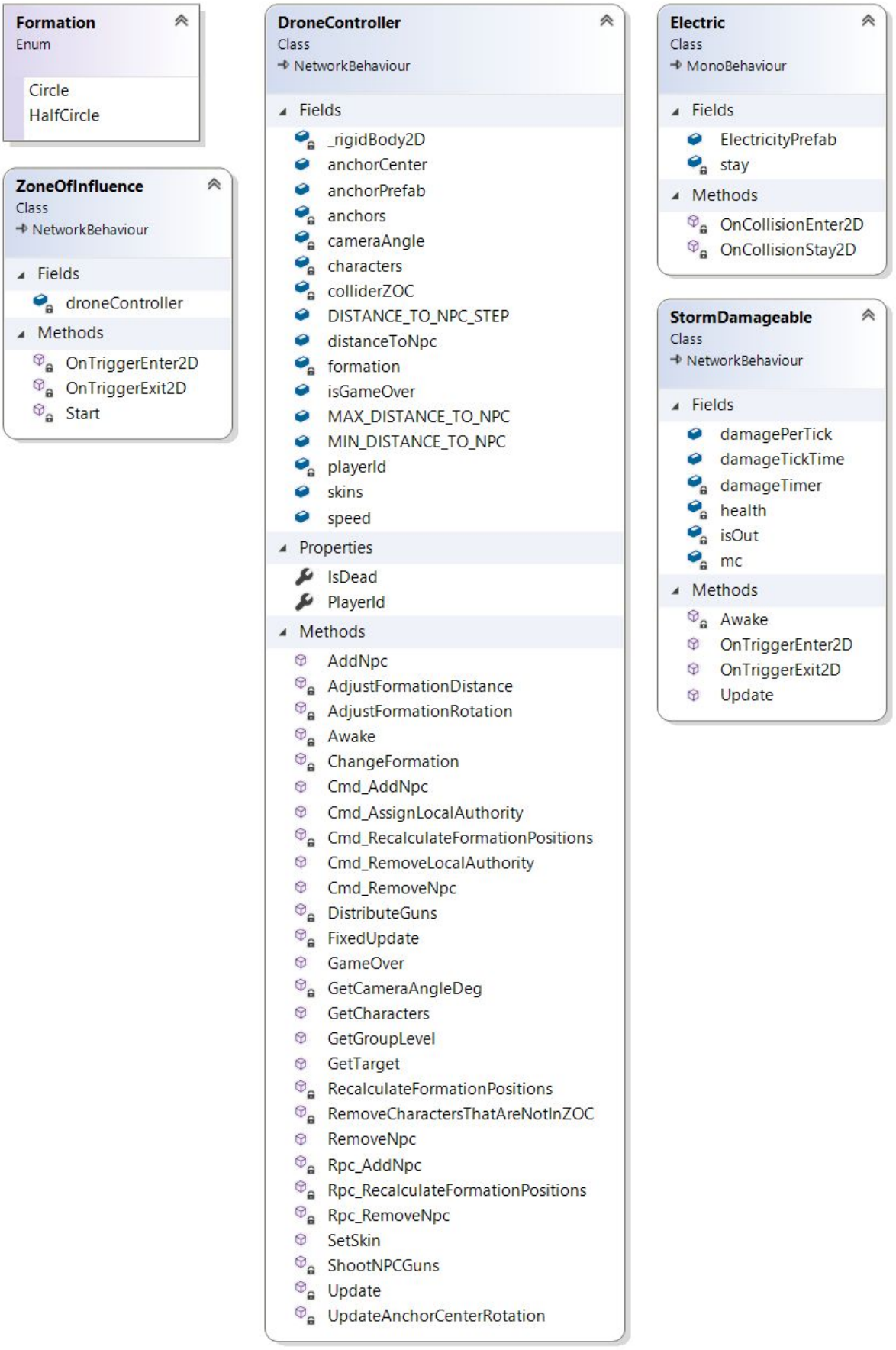
Networking

Here are the classes that allow to manage networking from a high-level perspective. CustomNetworkManager and CustomNetworkDiscovery allow for the host and the client to broadcast their presence on the network, which allows to manage a lobby. PlayerNetwork is the class that appears on the object that is spawned when a client connects to the server. NetworkUI is responsible for the UI that allows to manage the multiplayer aspect of the game. Finally, GameManager manages the initial state of the game (the lobby), as well as the connection or disconnection of the players in the middle of the game.



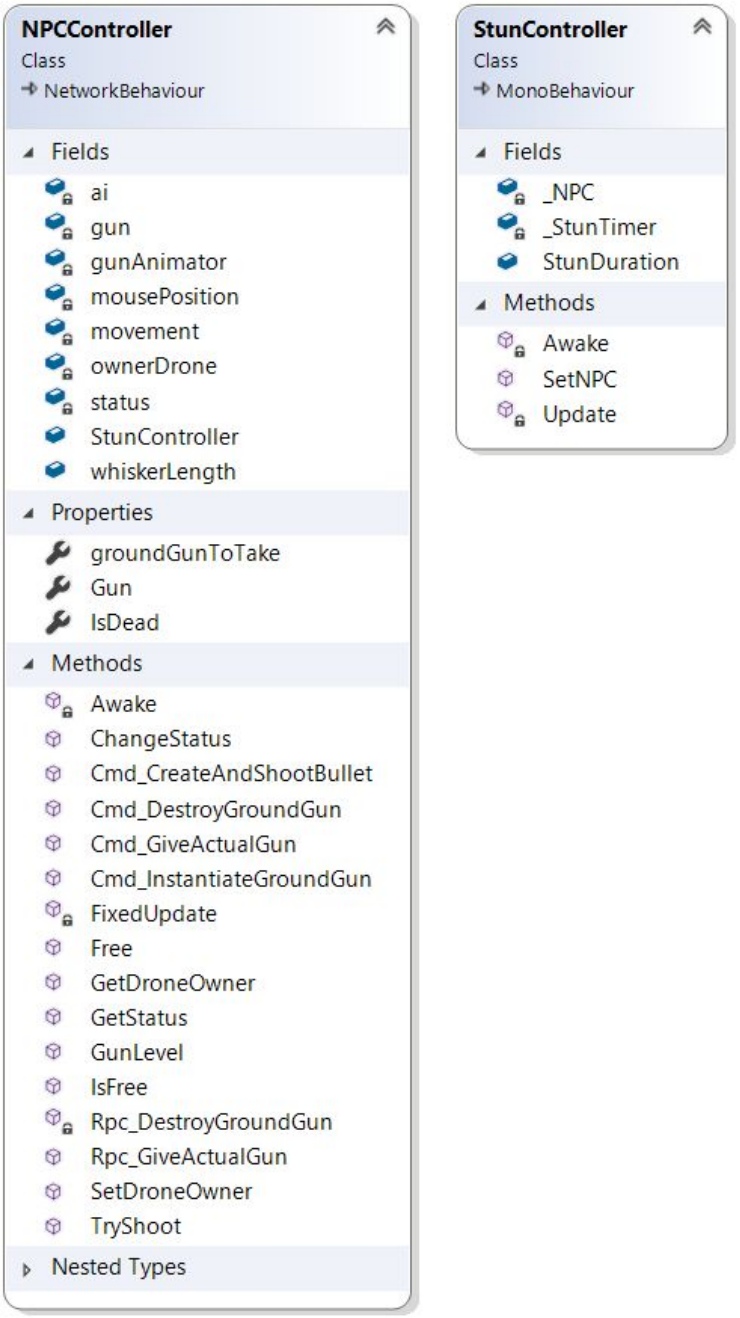
Drone

The following classes manage everything related to the drone, the character that the player is controlling. DroneController is the class that manages the drone itself, which includes managing its movement, the anchor points for the NPCs, the formation, and the AI for the NPCs to get the guns of a higher level. Formation is an Enum that indicates what types of formation the drone can have. ZoneOfInfluence manages the entering and the existing of the NPC inside or outside the ZOC (zone of control) of the drone. Electric allows to create sparks when the ZOCs of two different drones touch each other. StormDamageable allows the drone to slowly lose health when it is inside the EMP storm.


Formation
Enum
Circle
HalfCircle
ZoneOfInfluence
Class
NetworkBehaviour
Fields
droneController
Methods
OnTriggerEnter2D
OnTriggerExit2D
Start
DroneController
Class
NetworkBehaviour
Fields
_rigidBody2D
anchorCenter
anchorPrefab
anchors
cameraAngle
characters
colliderZOC
DISTANCE_TO_NPC_STEP
distanceToNpc
formation
isGameOver
MAX_DISTANCE_TO_NPC
MIN_DISTANCE_TO_NPC
playerId
skins
speed
Properties
IsDead
PlayerId
Methods
AddNpc
AdjustFormationDistance
AdjustFormationRotation
Awake
ChangeFormation
Cmd_AddNpc
Cmd_AssignLocalAuthority
Cmd_RecalculateFormationPositions
Cmd_RemoveLocalAuthority
Cmd_RemoveNpc
DistributeGuns
FixedUpdate
GameOver
GetCameraAngleDeg
GetCharacters
GetGroupLevel
GetTarget
RecalculateFormationPositions
RemoveCharactersThatAreNotInZOC
RemoveNpc
Rpc_AddNpc
Rpc_RecalculateFormationPositions
Rpc_RemoveNpc
SetSkin
ShootNPCGuns
Update
UpdateAnchorCenterRotation
Electric
Class
MonoBehaviour
Fields
ElectricityPrefab
stay
Methods
OnCollisionEnter2D
OnCollisionStay2D
StormDamageable
Class
NetworkBehaviour
Fields
damagePerTick
damageTickTime
damageTimer
health
isOut
mc
Methods
Awake
OnTriggerEnter2D
OnTriggerExit2D
Update

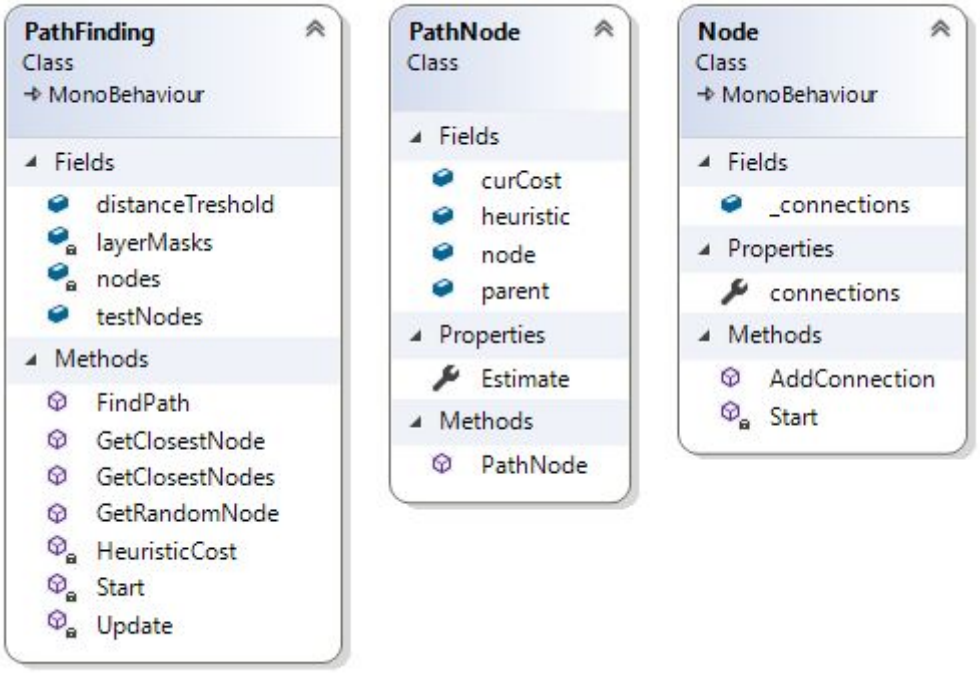
NPC

These two classes manage all the behaviour of the NPC (excluding the AI part). NPCController manages the movement of the NPC, the gun pick-up, the animation of the NPC (the current sprite) and many other things. StunController is responsible to make the NPC stunned when it is hit by a stun grenade.



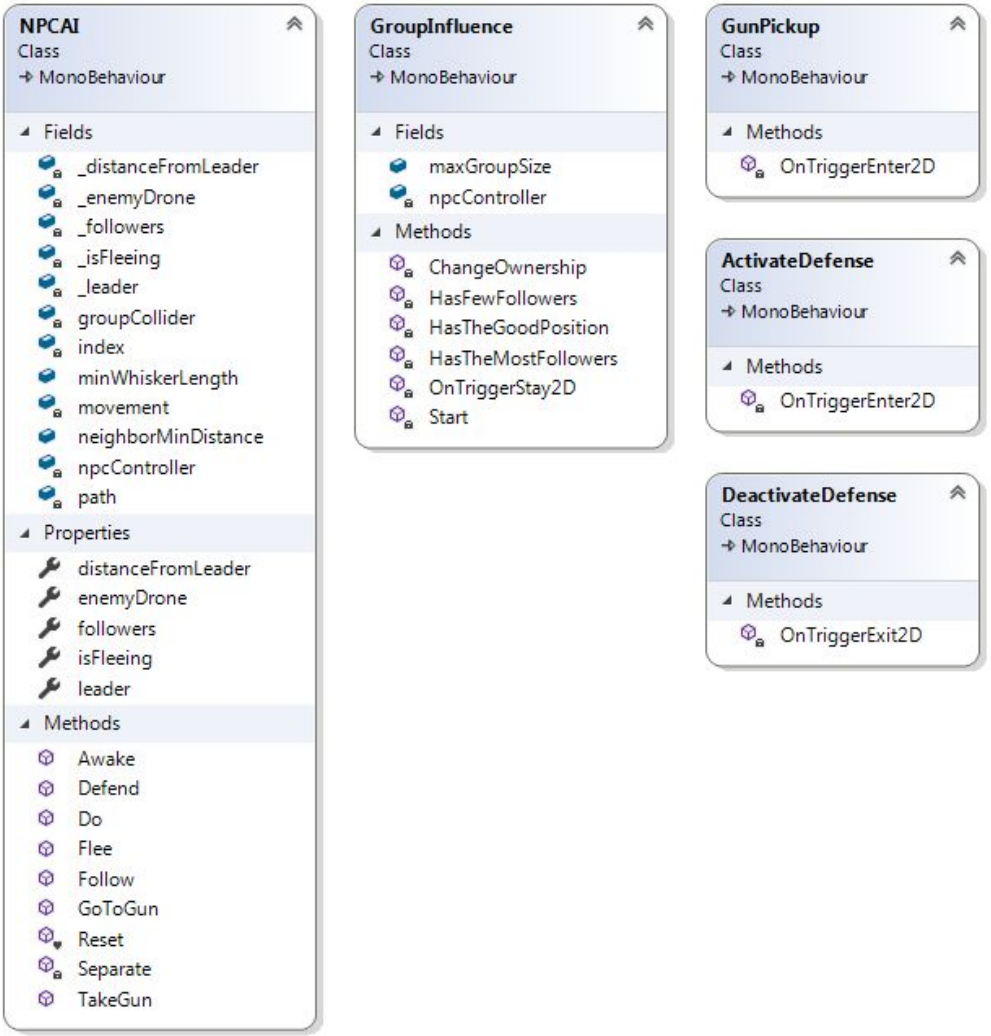
Pathfinding

These three classes manage the pathfinding and A* algorithm. The PathFinding class creates a pathfinding graph by linking all nodes within a given distance threshold, on which we can find the shortest path with an Euclidean distance heuristic. Node contains all connected nodes to a given node and the cost to get to them. PathNode is used during the execution of our A* algorithm, and stores the information needed to run the algorithm.



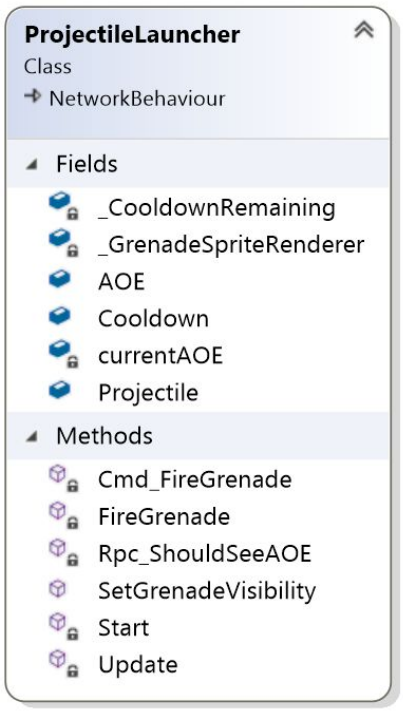
Free NPC AI

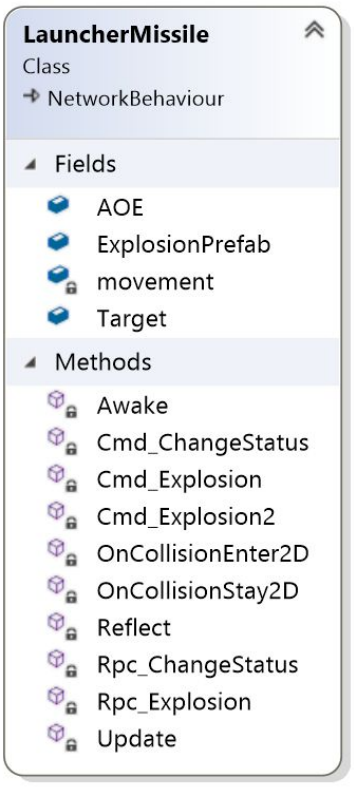
These 3 classes are responsible for the decision making of the free NPCs. GroupInfluence triggers when different NPC groups collide with each other and decides how to join them, create a leader, or change ownership. GunPickup, ActivateDefense, and DeactivateDefense are responsible for the other actions that the NPC might take. NPCAI is where all pathfinding calculations are made and all actions are executed, depending on the state of the NPC and the decision making tree which can be found later in this document.

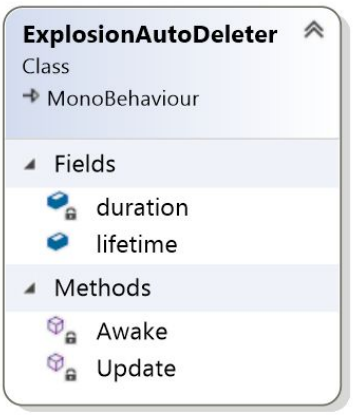


Drone Stun Grenade

The following 3 classes are responsible for the management of the stun grenade that the drone can throw at NPCs to stun them for a short moment. ProjectileLauncher is the class that is put on the drone and that allows it to throw grenades. LauncherMissile is the class that is put on the stun grenade itself when it is throws. Finally, ExplosionAutoDeleter allows to play the explosion animation and it then deletes itself.

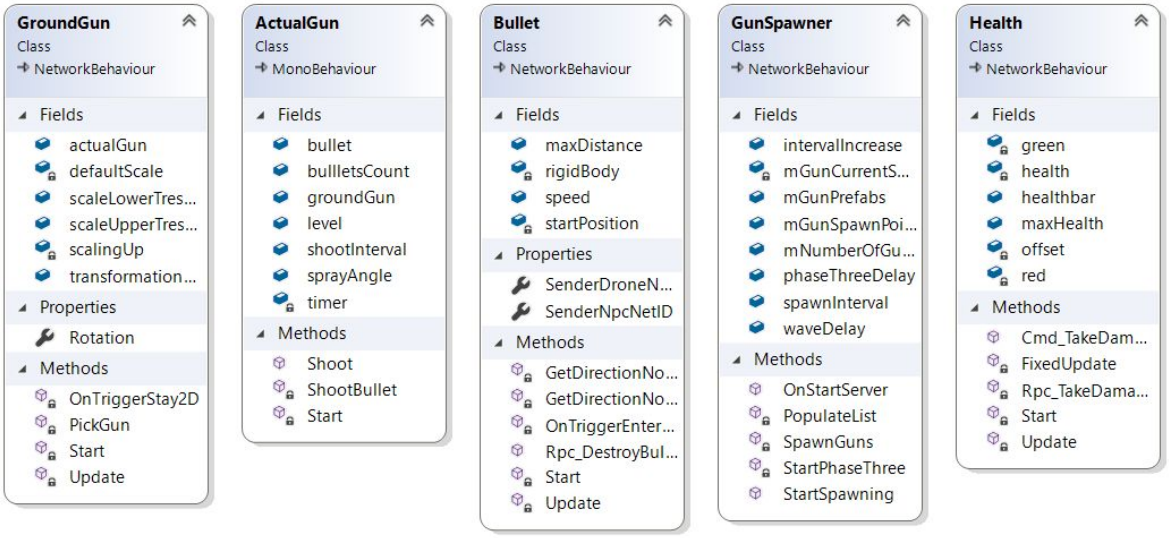






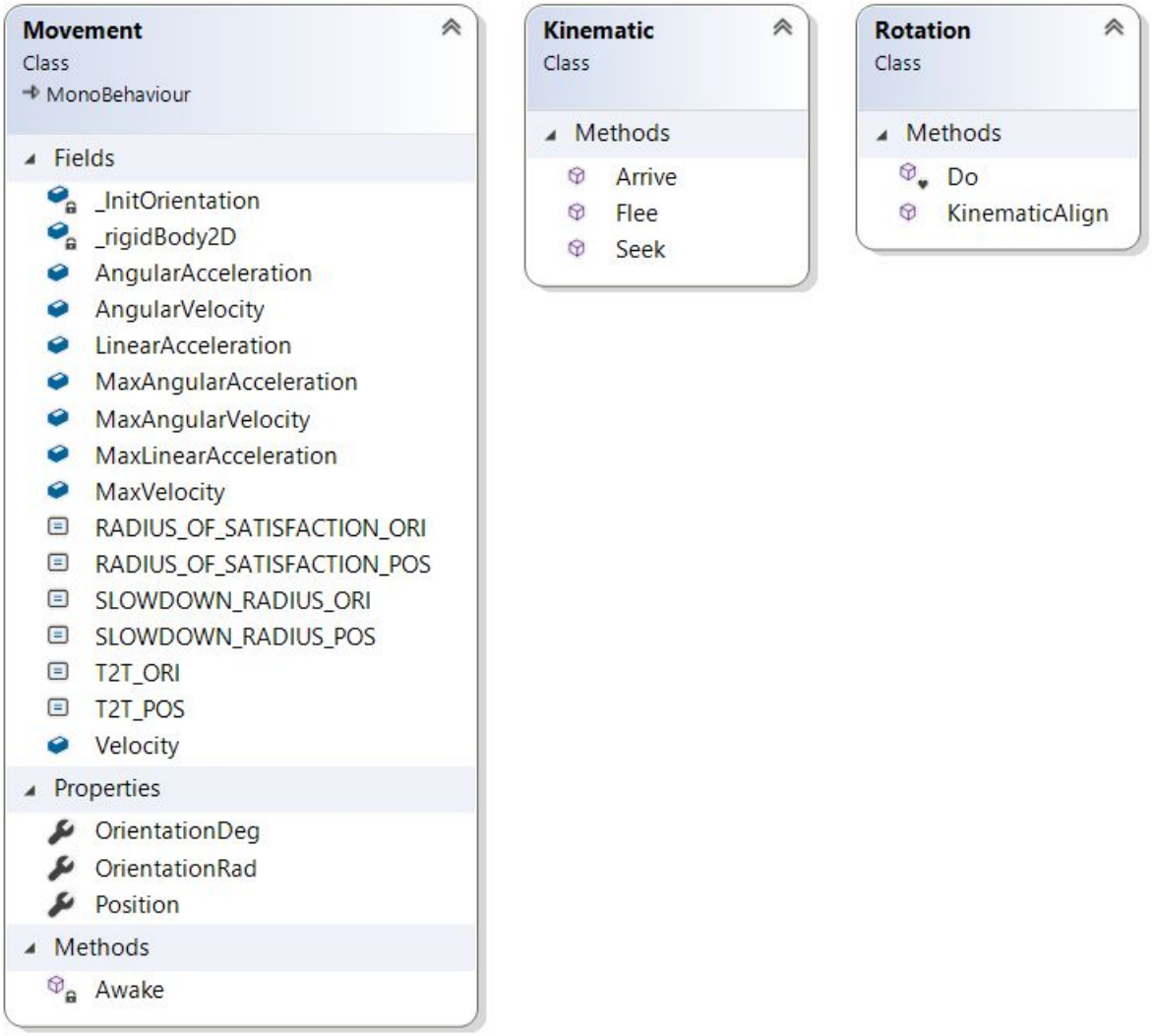
Gun

The following classes allow to manage guns that can be either on the map or taken by NPCs. GroundGun is the class that is put on the gun that is dropped on the map, while ActualGun is the class that is put on the gun that is part of an NPC object (taken by an NPC). Bullet is used on any bullet that is shot from any gun. Health manages the health bar of the drone and the NPCs and it responsible in destroying them if their health reaches 0. Finally, GunSpawner represents a point on which the GroundGun could spawn.



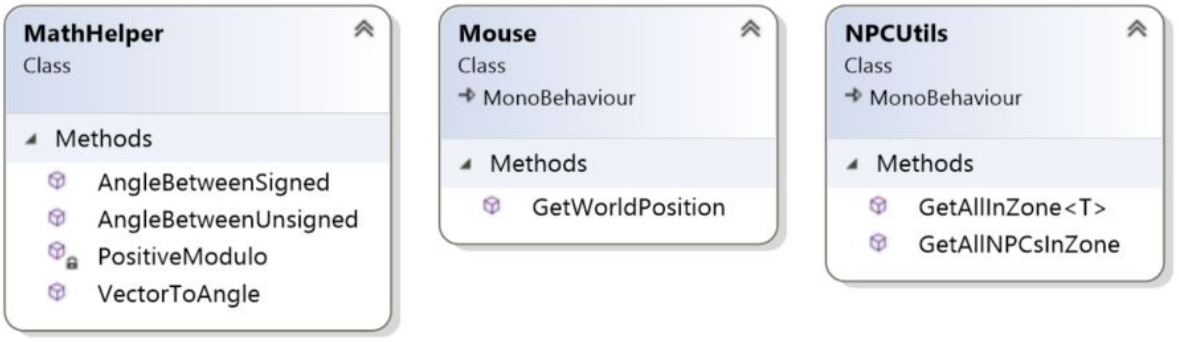
Movement

We only use kinematic behaviours in our application and they are implemented using the following the classes. Movement is a class that stores all the data (all variables) related to kinematic movement for a single object in our world space. Kinematic and Rotation are helper classes that contain static methods which take a Movement object as a parameter and allow to do different kinematic actions on it.



Helper Classes

These classes contains static methods that may be used anywhere in the application. MathHelper contains methods that help to manage rotations. Mouse allows to obtain the world position of the mouse and NPCUtils allows to obtain NPCs and other types of objects in a certain collider.



Controls

The game requires having a keyboard and a mouse to play. Here are the commands:

Input	Action
Mouse movement	*Drone* Rotate the drone (which also causes the **formation** to rotate) *Controlled Humans* Aiming
Mouse left (click/hold)	*Controlled Humans* Shoot (if armed)
Mouse right (hold)	*Drone* Locks the rotation in place
Mouse wheel	*Drone* Scrolling up increases the formation slots distance from the drone. Scrolling down reduces the distance.
1, 2	*Drone* Alternate between the drone's formations
Space bar	*Drone* Sends out a stun grenade By holding down the space bar, the **area of effect** of the stun grenade is

	displayed to the local player. The grenade is sent upon release.
W, A, S, D (hold)	*Drone* Movement *(Controlled Humans)* By moving the drone, this causes the formation slots positions to move, thus causing the controlled humans to move accordingly
F1	Displays the game controls (when in game)
Esc (press)	Displays Pause menu (The game still continues) Allows the player to leave to the main menu or leave the game
Mouse Movements/ Mouse Left (in Menus)	Select and confirm menu selection
P	Changes role to “Player” when in game **lobby**
V	Changes role to “Viewer” when in game **lobby**
M	Mutes/Unmutes the game music

Algorithms mapping analog inputs to actions

The algorithms mapping inputs to actions are used for the different humans movements when they are under the control of a drone. The drones themselves are

simply using a kinematic seek to follow the movement dictated by the player. The drone also rotates to always remain exactly aligned with the mouse cursor. However, the humans' movements are much more complex.

When under the control of a player drone, each human is added to the drone formation. In this formation, the drone serves as the anchor and determines where the slots should be positioned based of its position and the formation selected by the player. Based on this position, each one of the humans perform a **kinematic arrive** to his/her own slot position (similar to a **two-level formation**). We gave this arrive a very short **time-to-target** to insure that the humans get to their slot position quickly, but that they will end their movement in a very smooth manner. It is really important that the humans start their movement and get to their slot quickly as this will otherwise create a very frustrating situation for the player. Regarding the humans rotation, as humans can't rotate on themselves instantly, we decided that to make them use **kinematic align** towards the mouse position with a **high angular velocity**. This makes sure that players will be punished if they are attacked from behind in the sense that they won't be able to counter attack instantly. Also, this provides a much more smooth transition between the characters directions as you can see them rotate. Although they use align, we still gave them somewhat of a high angular velocity to insure that players can adjust their aim rapidly for smaller rotations.

Within a certain formation, we decided to give the player the possibility to bring the humans **closer** or to **distance** them as this significantly increases a player's options. For example, a player can move the humans further from the drone in order to provide them with a line of sight of the enemy, but to keep the drone hidden at the same time. This also allow a player to bring the humans close to the drone to serve as a **human shield** in case of a surprise attack. We also decided to provide the player multiple formations to use so that he/she can feel empowered in the different gameplay situation he/she faces. This also provides the players with more ways to approach the enemies and makes the **formations** more **complex** and **flexible**.

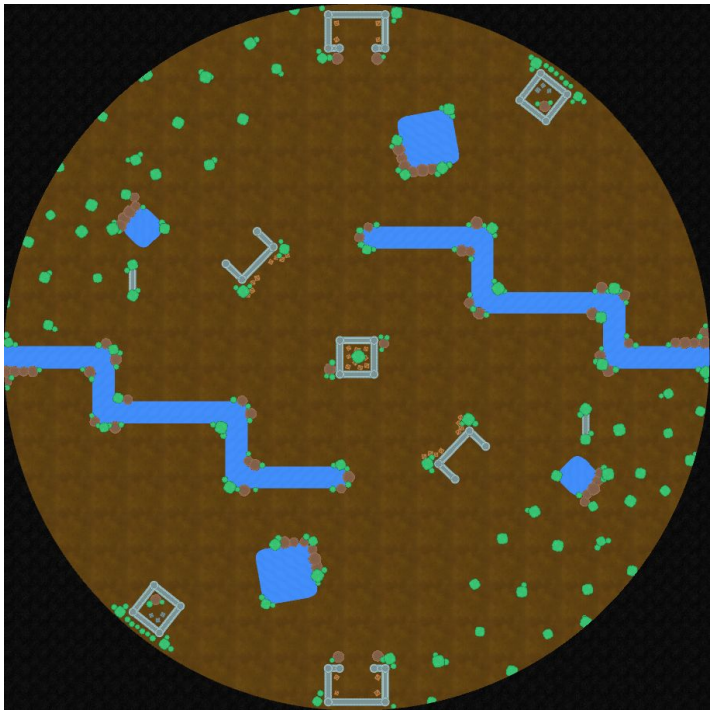
Finally, although the characters stick to their positions in the formation most of the time, if a **weapon** enters a drone's **zone of influence** (area within which it controls humans' minds), the character with the **current worst weapon** in the

formation (in case multiple characters have the same worst weapon level, the **closest** character from the weapon is be chosen) will leave his/her slot to go and get the weapon. The formation selects which human should go and get the weapon in a way that **maximizes** the formation power. This computation is based on the weapon properties and the current weapons of the rest of the formation.

Level design

When designing the level, we tried to include multiple choke points, where players can lead their drones and use tactical advantages to overpower their opponent. While keeping this in mind, we also added symmetry in order to give a fair chance to both players, no matter which side they start on.

The Arena



Players' drones start in their respective bases. One is located to the north, south, east and west of the arena. (East and west bases are simpler, with only rocks and trees cover.) As players start exploring the map, they will find different areas, where they can pursue humans they wish to control, look for guns in order to increase their firing power, or hunt the enemy.

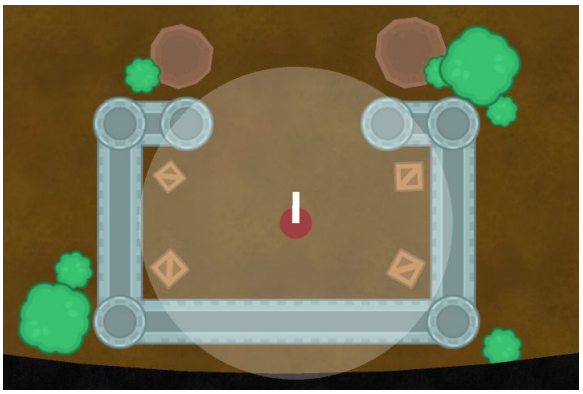
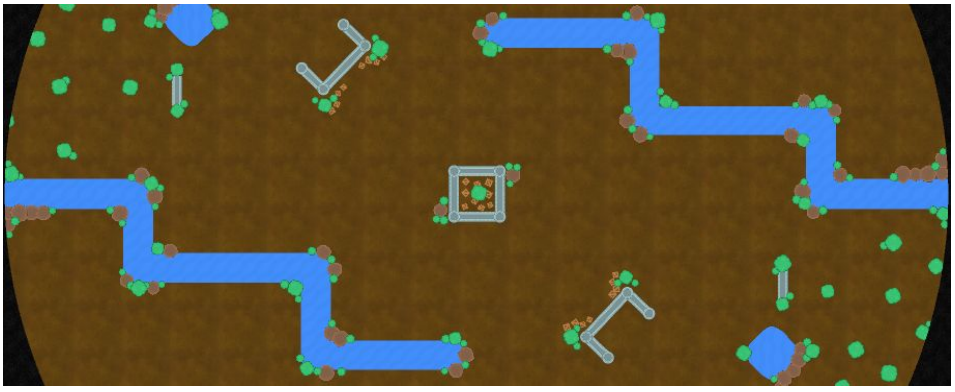


Image of a development drone inside its base

There is a forest located to the right of each players' base, where more humans tend to wander around. There, it is easier for drones to hunt for humans and increase the size of their army.

To the left of the starting base, there is a watchtower next to a pond. Over there, more weapons are lying around to be acquired and boost a drone's army firepower. Players have to be careful while entering this zone as there are dangerous humans who have already acquired strong weapons ready to defend themselves. It is advised that they build a small army before roaming there.

In the center of the arena, there are two rivers separating the bases with a main watchtower. This is the most dangerous area of the map as it forces the two players to face each other. Although the area offers a lot of structures providing defensive tactical points, the chance of being cornered by wandering humans is increased as they focus their strengths on taking the other player down.



Every moving human and drone is affected by the objects, structures and natural geometry around the level. While humans and drones cannot cross solid objects standing in their way, drones and bullets can still fly through bodies of water. This allows players to move behind impassable terrain, take cover behind objects, rocks or trees, and fire at their opponent from the other side of the body of water.

Mechanics analysis

Next is a description of the multiple gameplay mechanics that were included in the game. After this, you will find some game balance changes we made after some testing.

Shooting

In this game, shooting is our main gameplay mechanic. Due to the top-down view, this offers very frenetic action. In a similar way to what is presented in games like *Hotline Miami*, players must very quickly react to their opponents movement and always stay alert for what's coming their way. Mixed with the fact that the game presents AI and players threats, players must always be on their guard and quick to react to what they see. In our case, we really wanted to provide twitch online gameplay and a top-down shooter is definitely a very good way to convey this experience to the players. The human control mechanic also adds a nice twist to this mechanic as players cannot dodge bullets as freely as they want.

Movement

Combined with shooting, moving is also a very important mechanic of our game. As stated previously, we were really motivated to provided twitch gameplay to the players. That said, in the case of a top-down shooter, the second most important mechanic is always movement. As we wanted the controls to feel very reactive to the player inputs, we decided to make the drone's movement kinematic seek. We felt like this was one of the only way for it to feel right, mostly in the context where players must be able to navigate the map efficiently, but also dodge enemies bullets.

Human Control

Human control is one of the demarking features of *Mechanica*. Early on in development, we were looking for an interesting and innovative way to incorporate **group AI** and **formations** to a top-down shooter game. We ended up finding it in the human control mechanic. As we wanted the gameplay to remain very fast paced and the controls to feel very natural, we came up with the idea of the player controlling a

drone, which serves as an anchor for a group, from which slots positions are assigned to the different NPC characters within a certain range of the drone. This range is used to determine whether or not a drone has control over a NPC. This type of formation resembles in a certain way a two-level formation where the player must adapt the anchor speed and position to the NPCs' movements and obstacles.

However, we didn't want the humans movements to be too mechanical. We wanted them to feel natural and to be very smooth at the same time. This is why we decided that the NPC would use **kinematic arrive** to get to their anchor points. We decided to give it a **very small time-to-target** to insure that although their movement ends smoothly when they come into place, they still get there at maximum speed for most of the distance they have to cover.

Moreover, usually, in most top down shooters, players have a direct control of the positioning of their character, which is still the case in our game as the player controls the drone directly. However, the human's control mechanic brings in a very interesting dynamic. Because the drones on their own do not have any lethal weapon, the player must use the humans in the best way possible to take out his/her opponents. On the other hand, humans are more limited than the drone in the sense that they cannot cross obstacles and they do not **move** and **rotate** as fast as the drone. This forces the player to really play around these limitations and to try and get the most possible out of the humans controlled while still staying alive. Also, the player must be more careful with his/her movements as moving the drone too far to dodge a bullet may result in a lost of the control of many NPCs which may then start to attack the drone leading to a possibly even worse situation.

In summary, the human control mechanic really serves as a demarking factor for our game. It creates a second layer of control where although the player controls the drone perfectly, he must remain attentive to the humans it indirectly controls, which is really innovative and interesting.

Zone of Influence

In order to balance this human control mechanic, each drone has a certain "zone of influence" around it determining where humans must stand for the drone to have control over them. Upon entering a zone of influence, a human becomes

controlled by this drone. As drones emit incompatible force fields for their own zones of influence, their zones of influence cannot cross each others (they collide). This also insures that it remains clear at any given moment which drone has control over which human. Upon leaving a zone of influence, a human becomes free again and starts behaving on its own once more.

Formation

With this human control and zone of influence game mechanics, another very important mechanic naturally came along: the formation mechanic. By pressing different numbers on the keyboard, namely 1 and 2, players can alternate between different formations. Each of these formations has its own advantages and disadvantages. For example, the circle formation offers a better all around drone protection and angle of attacks than the umbrella formation. The umbrella formation on the other hand offers a great protection against attacks from a certain angle plus it provides move offensive options, but it also leaves the drone fully opened to attacks from behind. Also, the formations are fully evolutive. This means that as players (drones) take control over more and more humans, the formation will automatically evolve to naturally incorporate these new recruits. The transition between the different formations also feel very natural and each formation fades really well the ones into the others. The formation mechanic really provides the players a customizable way to approach the different gameplay situations they dive into and really gives more richness to the other gameplay mechanics.

On top of that, we decided to give players the opportunity to set the “sparse factor” of their formations to provide even more ways to play. This gives players greats opportunities like to hide the drone behind cover and to move the humans around to cover to attack. The player, by rotating the drone can also align the formation differently. On top of that, we also added the possibility for the player to lock the formation rotation in place so that the formation slots will remain in the same position and only the humans will rotate in the direction of the mouse cursor. Overall, this really enables players to truly customize the way they want to play and opens up a multitude of possibilities for the player to really engage the opponents in any way they want. It also makes sure that although players do not control the humans NPCs

directly, they still have a lot of control on the different actions these characters can perform as a group.

Stun Grenade

The stun grenade serves as a useful gap closer to help the drones take control over humans. As stated multiple times already, for a drone to take control over some humans, the drone must first get close to those humans. However, if armed, these humans might try to destroy the drones. Thus, especially during the early game phase, this makes it very hard for the drones to close the gap between them and the humans without taking damage. In order to balance this, we decided to equip the drones with a stun grenade.

By holding down the space bar, a player can see which area would be impacted by the stun grenade. Then, when releasing the space bar, the grenade is launched and perform a kinematic seek to the target position. Upon reaching it, the stun grenade explodes and causes all the humans in its venicity to be stunned for a short duration. This gives players the opportunity to get close to these humans and take control of them in a safer manner.

In a battle situation, this also gives a player a good counter measure if an opponent regroups his/her humans as part of his/her strategy. With a well placed stun grenade, a player can disable the other players humans leaving them very vulnerable. Also, if the other player decides to move too far while his/her humans are stunned, the player who sent out the grenade can event take control of those humans. Overall, the stun grenades brings a very good balance in the human recruitment process, and also adds a very useful tactical tool in combat. We also used it to implement a special collision resolution which will be explained later in this document.

Weapons Pickup

Spawning around the map are multiple types of guns, each one of them offering a distinct advantage. For example, the shotgun is better at short range whereas the submachine gun and the assault rifles shoot a lot faster and have more

range. The weapon pick up mechanic was implemented to provide more variety and to add a third gameplay phase in each match, the other ones being the recruitment and the fighting phases. Players are then encouraged to explore the map to try to discover the best weapons possible for the humans they recruited. Players can also decide to take on different situation differently based on the weapons they found allowing them to diversify how they approach their enemies each game. For example, when encountering a free human with an assault rifle (strongest weapon in the game), if a player already has a lot of humans, but they all have low level weapons, he/she might prefer to simply kill off that human and only get his/her weapon instead of trying to recruit him/her and risk to take more damage.

This leads to another aspect of the weapon pickup mechanic which is very simple: when dying, humans drop their current weapon where they died. This allows a player to have his/her humans to pick up stronger weapon when humans within or outside the formation die.

Weapons Upgrades

At all time, the goal of a drone regarding its humans weapons is to maximize their power. That said, when a gun enters the zone of influence, the drone will force the humans with the lowest weapon level, and which is the closest from it (given that multiple humans have the same lowest level of gun), to go and get it. This means that if no humans have a weapon, the closest one will simply run to go and get it. Otherwise, if all humans already have a weapon in hand, the human with the worse gun will go and swap his/her weapon for the better weapon. This is a simple and intuitive way for the player to have his/her humans **automatically** maximizing the strength of the group without having to manually perform micro actions to make sure the right character picks up the gun and that a human with a strong weapon will not exchange it for a bad weapon simply because he/she walked over it. This is a quality of life improvement for the player and helps to make sure the fast pace of the game is not slowed down by gun swapping operations.

EMP storm

In a similar manner to the “storm” from *Fortnite* and “The Playzone” from *PUBG*, *Mechanica* has a game mechanic called *EMP storm*. The *EMP storm* is basically a zone restraining the play area in order to force players engagements as the game advances. In our case, this is mostly to force players who would like to keep on recruiting humans to go and fight the other drones. In *Mechanica*, the storm being an EMP storm, only **drones are affected** by it causing them damage over time for as long as they remain in the EMP storm. This mechanic insures a short duration for each round as it forces players to run into one another. This also makes sure that one player cannot simply “camp” and wait for the opponents to come as he/she will eventually be forced out of that position by the storm. Also, the bigger the storm is, the more damage it will deal per tick making it much more punishing later on in the game.

Player vs Player

From the start, we really wanted to create a fun player vs player experience. Nowadays, almost every game includes a player vs player mode, and more and more, games are built as competitive multiplayer experiences instead of casual single player adventures. We also wanted the players to be able to play with their friends and to really enjoy their experience in group. In a player vs player context, players are always sure to find an interesting diversity of strategies amongst different types of players. Some will play more offensively, some others will focus more on recruiting as many humans as possible. Having this duality where some characters are controlled by AI, but the real opponents are controlled by other players, really creates an interesting environment for the players to enjoy.

Viewer role

Finally, mostly for the demo purposes, we decided to add a new role for users joining a game. When in the lobby, a user can decide a role between **player** and **viewer**. By default, every person joining the game is a player. However, as there can

only be between two to four players (we decided that limit), we wanted players to also be able to watch the game without participating. To do so, when selecting **viewer** in the lobby, the user, when the game starts, will be given a point of view of the entire map. This is a good way to take a look at all the interactions taking place in the game.

Balance changes

Over the course of the development of the game, many elements were fine tuned to insure a fun experience for the players. Here are some of the decisions that were taken.

Drone speed vs human speed

In order for the drones to be able to take control over the humans, it was important for them to be able to move faster than the humans. However, if they were to go too fast, it would then make it too easy for a player to capture humans. That said, we spent some time testing different speeds for the drone and the humans. Another important things was that under the control of a drone, the humans would move and rotate at a speed that it not too fast, but not too slow as well. We wanted the humans to take a few frames to get to their slots positions, but if they were to take too long this would really make the experience more tedious than fun.

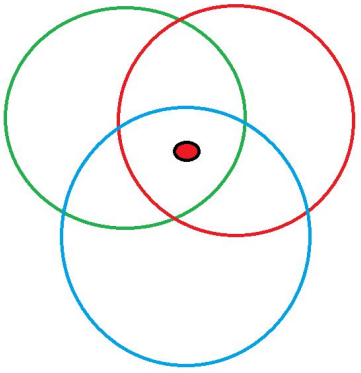
We decided to make the drone rotation instantaneous so that during gameplay, players can quickly see where their humans will end up once done moving (shown by the green dots representing the slots). However, we decided to make the humans rotations slower so that players being surprised from behind cannot immediately answer back. We really wanted to make sure that in such situations, the player that took the right offensive decision would get properly rewarded.

We also made the speed of the free humans and the controlled humans different from one another. Under the control of a drone, humans move faster than when they are free. By being slower when free, it makes them easier to catch, and by making them faster when captured restricts less the drone movement making it

possible to capture other humans without necessarily losing the previously captured ones.

Colliding zones of influence

After some testing, we realized that allowing the zones of influence to cross raised some unclear situations for the players. At first, we thought of giving the drones multiple owners when multiple drones would have this one human under their zone of influence. However, the drone that took control over the human first would keep it for as long as the human did not leave the zone. If the drone would leave the human, he/she would then be captured by the drone that took control over him/her in second. Unfortunately, this mechanic became very unclear with more than 2 players or when the human was leaving and entering different drones' zones of influence fairly quickly. In a situation like the following:



Let's say that the human first entered the red zone of influence. If the red drone leaves the human there, given that the green drone had been the second one to have its zone of influence including the human, it would have become the owner of the human. However, let's say that the human had briefly left the green zone of influence and has only remained in the red and blue ones before coming back into the green zone of influence once more, the blue drone would become the human owner in case of the red drone's departure. Given the speed of the game, we felt like this could cause frustration amongst players as the situation could quickly become unclear. It's for this reason that we decided to make it so that the zone of influence cannot cross and thus, humans always remain in at most one zone of influence at all time.

This actually created a new possible strategy for players, especially for players without humans. With the EMP storm closing in, a player can decide to **block** another drone inside the EMP storm to kill it. This brings in a new and fun mechanic that we really hadn't planned at first. To make this mechanic more clear, we added a spark animation at the contact point when two zones of influence collide displaying the electromagnetic fields interactions.

Stun grenade

At first, the stun grenade was not part of our initial plans. However, thinking more and more about our capture mechanic, it became evident that taking such risks (these risks being to capture a human while this one is attacking a drone's humans or a drone itself) for players would feel quite unfair. This is why we wanted to implement a skill based safer way to capture humans in the form of the stun grenade. It was important for us that this feature would not be too easy to use, making captures "free", but we also wanted it to be quite accessible for players.

We later realized the offensive potential of such a tool and felt like it was really bringing an interesting "game changing" aspect to the fights. However, to insure this tool did not become too strong, we spent some time fine tuning its area of effect so that it cannot hit too many enemies at one time. We also fine tuned the stun duration so that, especially when stunning an enemy drone's humans, it does not end up feeling too strong. We also fined tuned its cooldown to make sure that using it actually represents a risk.

Zone of influence size

Additionally, we also fine tuned the zone of influence size. We realized that if it was too small, it would be more frustrating for the players as humans would leave it very easily. However, if it was too big, characters would then be too easy to keep under control hence negating this interesting mechanic.

Two gameplay phases

At first, everything was spawning all at once: drones, humans and guns. However, we quickly realized that this was really not balanced and could lead to situations where players just spawned in and within a few second, they now have several NPCs coming at them already armed. In order to balance this, we introduced the two phases systems. A message appears on screen at the beginning of each one of them to indicate players what they are supposed to do and when the phase actually transitions.

The first phase, labeled “Phase 1: Gather human” only includes drones and humans. This provides players with an good opportunity to gather many humans without being worried of getting killed and without having to worry about getting weapons. In the second phase, labeled “Phase 2: Fight to DEATH”, guns now start spawning in. Players can then arm the humans they accumulated to kill the other drones. Guns have corresponding spawning rates based on their power (stronger weapons spawn less often) and humans keep spawning in to compensate for the humans getting killed. Finally, as the second phase begins, so begins the EMP storm phases. That said, 30 seconds after the second phase started, the storm will first expand and so on.

This two phases system really improved the general feeling of the game and made it much more balanced and fun to play, removing some problematic situations.

Artificial Intelligence

One of the key aspects of this project was the implementation of the AI. For our game to feel right, the AI had to be able to behave properly in some quite different situations. Let's explore these different situations:

Humans under the control of a drone

Under the control of a drone, the drone (player) really becomes what provides the direction to this **group AI**. As stated earlier, the formations are established based on the drone, which is directly controlled by the player. Players can choose between a few different formations, and based on the formation selected, the orientation of the drone and the "sparse factor" (also managed by the player), the different humans in the formation will be affected a slot position. The standard behaviour of these characters is to simply **kinematic arrive** at their anchor position and to **align** with the player's mouse cursor.

When a gun enters the zone of influence of a drone, there is a group AI that is used by the drone's NPCs. At that point, the goal of the drone is to **maximize** its offensive power. To do so, it will try to provide its humans with the **lowest weapon power** a **better** weapon. This will have for effect to **increase** the group potential and improve the player's chances to win.

When a new character joins/leaves the formation or when a character dies, the formation is **immediately** adjusted. As such, the slots for the remaining characters are moved around in order to adapt the formation to the new amount of characters included. These formations are highly adaptive meaning that they support any number of characters properly (we tested with a very large number of them).

Free humans

The free humans have a much more intelligent and complex behaviour than the mind-controlled humans.

On their own, free humans first act as **leaders of their own group of one**. At that time, their main goal is to regroup with other humans. They also focus heavily on finding a **good weapon** to defend themselves. To do so, they use our custom **pathfinding graph** alongside an **A*** pathfinding algorithm (**Euclidean distance**).

Upon seeing another human, the two will get near one another and one of them will be chosen as the **group leader**. They will then explore the map together, while still keeping a certain distance between themselves. To do so, we used **priority-based blending** so that although the **followers** try to get to the **group leader** (**cohesion**)**,** they still keep **a certain distance** between each others (**separation**). To insure a clean navigation throughout the map, the **group leader** will be using **pathfinding** and the rest of the group will follow along. This will make sure to have groups not only walking on the pathfinding graph edges.

Original plan for weapon pickup

Upon seeing a weapon, if this weapon has a higher level than the weapon of **one person** from the group, the **leader** will guide the group next to it so that one of them can pick up the weapon. If the leader is not the closest to this weapon and the group decides to get to the weapon, the **closest follower** to the weapon will become the **new group leader**. This will speed up the process of getting to the weapon as it will make sure that the group will be directly oriented towards it.

Final Result

Upon seeing a weapon, if this weapon has a higher level than the weapon of **one person** from the group, that person will go and get the weapon at the expense of maybe **getting separated** from the group. If that person drops a gun that is higher level than another person in the group, then they will relay in switching weapons.

When **two groups meet**, the leader of the **smallest group** (or at random if both groups have the same amount of humans) will delegate his leader role to the **other group's leader**. That way **groups** can unite and form bigger and stronger groups. The exact same behaviour is used for solitary humans as they act as **groups of one.**

When fighting against an enemy, each humans part of the group will take some level of decisions on their own. For example, if the drone gets to close from one human, it will then start to flee, while still shooting, to make sure not to be captured by the drone. Also, if another drone ends up showing up, the human will decide on his/her own which drone to fight. Finally, instead of remaining immobile, the humans will move a little to make them harder to hit and make their position less predictable.

Throughout their map navigation, non leaders will also use a **wall avoidance** for their movements in order to follow their group leader as closely as possible and not to get stuck behind a wall.

IMPORTANT NOTE ON UNITY AND BUILD DIFFERENCES

For the longest time, we actually believed that the following was actually an error in our code. Everytime we would try out a different build of the game with some changes to the AI, for some odd reason, some NPCs would spawn and simply be locked in place. After a lot of investigation we realized the following: this **only occurs when the host is a built version of the game**. In other words, when testing in Unity and using standalone builds as **clients**, everything would run smoothly and no NPC would spawn and simply remain immobile. However, when the **host** is any built version of the game, no matter if it is a development build or not, for some reason, some NPCs may get stuck. We looked at many possible solutions like playing with the build architecture, but nothing really seemed to fix this. As you may know, Unity Networking is quite a new system and it is possible that there may be many bugs still in the code. We honestly apologize for this as we really tried to fix this issue, but it really appeared like nothing, other than having a Unity Editor being the host of the game, would fix this really odd issue. Thank you for your understanding.

Edit: We added some new fallback conditions that can help the standalone hosts to get out of this situation, but the NPCs still get stuck for a second or two before moving again. With the time at hand, this appears to be the most appropriate solution as it actually unstucks the NPCs. However, if you end up seeing an NPC blocking using a standalone build host, please keep this in mind. Thank you.

Scrapped ideas

The leaders were supposed expose their positions to each other so that they can use **pathfinding** to meet even if they lose sight of each others at some point. We dropped this idea because it added too many levels of behavior calculations.

By picking up weapons, the groups originally would slowly but surely increases their **confidence level**. This confidence level was mostly influenced by the **total power of the weapons** owned by the group, but also in part by the **amount of humans** part of the group. Based on this confidence level, upon seeing a drone or humans controlled by a drone, the group would have been able to **evaluate** its potential to win the fight and then decided whether to **fight**, **fight back**, **or to flee**.

During the fight, the group confidence level might have been impacted by humans **death** or **status changes** (eg.:stunned). Based on that the **group leader** would have decided whether to keep fighting or to retreat.

Finally, as the EMP storm would close onto one point of the map, the pathfinding graph nodes located outside the playable area would get cut off the graph so that humans could also migrate closer to the players to encourage more face offs. This idea was removed because it made the game too hectic in the center and navigation became difficult.

Animations for Non-Player Characters

In our case, all the animations of the non-player characters, namely the humans, are the same for when a human is under a drone's control and when a human is behaving on his own (or in a group). This is normal as both are *exactly* the same.

An important thing regarding our humans animation was to make sure that each status was clear for the player, especially since these statuses are quite limited. The first status in the normal status. In that situation the character walks around normally, not holding any gun. Upon picking up a gun, an animation is used to display the human holding the gun. As this is a very quick action, it is performed in

a single frame. Also, upon switching weapons, a player can easily see which gun was dropped as it appears in the exact same place the other gun was picked up.

A very important animation is the stun animation. This human status is very punishing as it leaves the human defenseless for a short amount of time. In order for the player to recognize this status easily, a standard symbol (stars rotating around the character's head) was used. The added movement to this animation makes it even more visible.

Scrapped ideas due to time constraints

When a human was not controlled by a drone, additional animations would have been used to communicate better its state to the player. For example, upon realization of an enemies presence and before attacking, the free humans would have performed a short jump in place to notify the player of their new intentions. These animations would have been put in place for the player to be more easily informed of the different state changes of the enemy and to be able to react more appropriately to it.

Physics

Regarding the collision resolution used in this project, for most of it, it is very standard. Bullets get destroyed upon hitting a wall or an enemy and drones and characters cannot cross impassable terrain.

However, a slightly different collision resolution was used for the stun grenade. When it hits a wall, instead on simply exploding on the spot, it instead bounces off the wall while losing some of its velocity. This allows players to come up with some very creative bank shots when using the stun grenade.

Also, a different collision resolution was implemented to deal with drones' zones of influence collision. Upon entering in contact, as small spark is displayed at the contact position and strays there for as long as the contact lasts. This is a creative and simple way to represent this idea that the drones zones of influence are actually electromagnetic fields and that they cannot mix.

Advanced Features

Next, you will find a description of the different advanced features that were implemented in our game, *Mechanica*.

Networking (Local Multiplayer)

The main advanced feature we decided to implement is **local multiplayer**. At first, none of us had experience with Unity's Networking features or any other game networking technology. However, we really felt that the model that fit our game idea the best was a multiplayer game. As stated already earlier, having the possibility to face-off against other human players truly gives a richness to a gameplay experience that cannot be reached otherwise. Also, although we already had in mind the formation idea and the free humans vs mind-controlled humans concept early on, we really felt like using this idea as a simple survival game would waste most of its potential and would also present too many situations in which the game could end up feeling simply repetitive or boring. The local multiplayer feature really adds to the game with a competitive aspect and provides players with more possible gameplay experiences.

In order for the experience to be intuitive for the players, a simple lobby system was put in place. When a player starts the game, he/she can then decide to *host* or *join* an existing game. When deciding to host, the player is presented with a UI displaying the different players in the lobby. When more than one player are in the lobby, the host can then decide to launch the game. When a player wants to join a lobby, the IP addresses of the hosting computers are presented to the player which can easily decide which lobby to join. This process really helps making this online integration fun for the player to use.

Pretty much every cases of players joining a game has been handle. The host will only be allowed to start a game when there is at least 2 players and a maximum of 4 players. However, the game allows, in theory, an infinite amount of

viewers/spectators. As soon as the host start a game. It will then stop broadcasting its game lobby and reject any new client that wants to join the lobby.

The whole process of hosting and joining was done by using a custom Network Discovery where a player can decide to host a game. When hosting, the host will broadcast its local IP address. The client players will join the game by listening to the host message over the local network.

During a game session, we decided to deal with the information in the following manner. Upon a game start, a player is given authority over a drone, meaning that its movement will happen locally before being sent over the network. Additionally, when an NPC is captured by a drone, this NPC becomes under the authority of the local player. We made this mostly to easier all the zone of influence entries and exits interactions and to make them feel more fair to the players. As we first had the NPCs controlled entirely by the server, this led to weird interactions where an NPC would leave the zone of influence locally, but not on the server so it would still keep following the formation. The inverse is also true. Additionally, this caused the NPC movements to locally lag a little bit more due to the additional communication overhead leaving their movement computation to the server. As the NPCs become under the **authority** of a drone when entering the zone of influence, this makes it so that only when an NPC **locally** leaves to the zone, it is returned to its original AI behaviour for everyone. This insures that players don't feel cheated when they lose control over their NPCs as their local vision was used as our source of truth. Additionally, when an NPC becomes under the **authority** of a drone, all of its movements are also computed **locally** which makes them move and rotate **very smoothly** for the player actually controlling them.

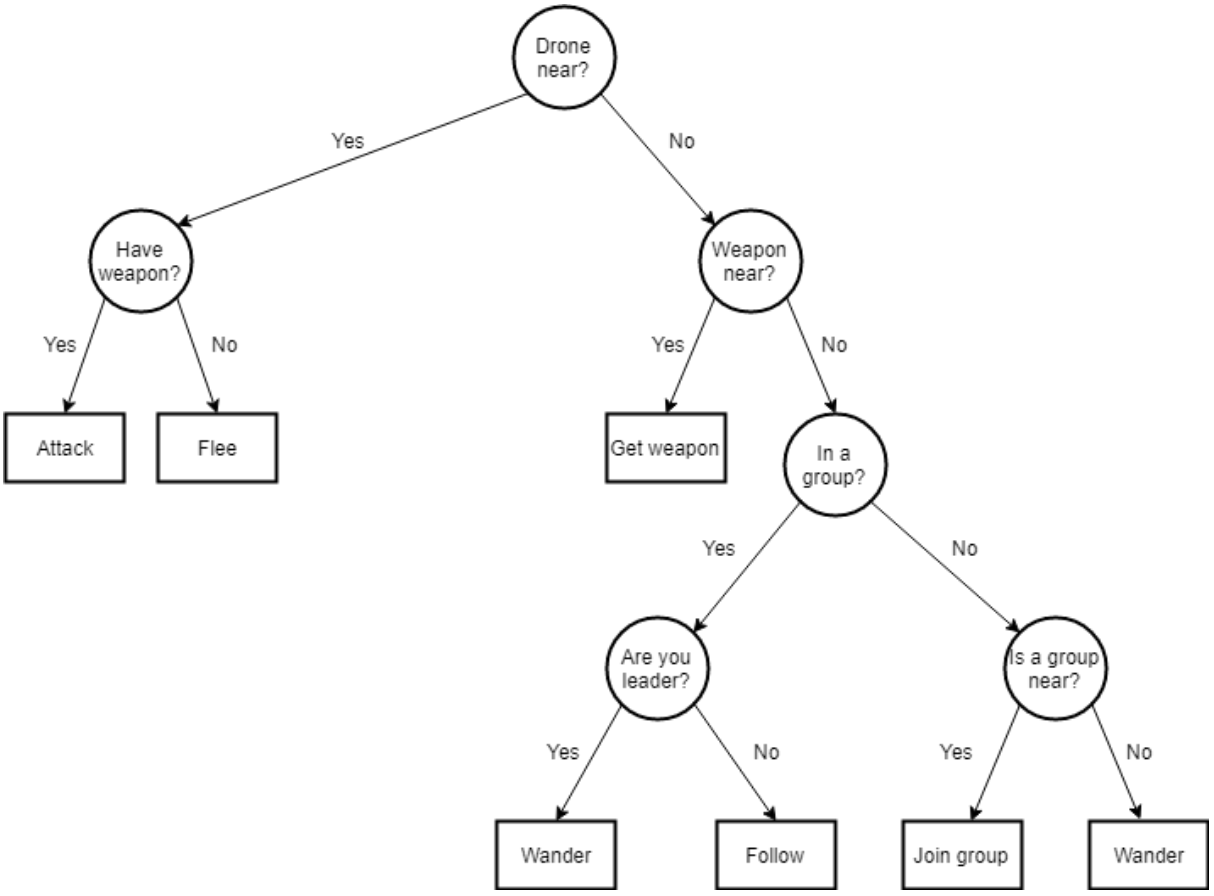
Regarding the NPCs standard AI behaviour, when an NPC is free, all their movement and strategy AI computations are performed on the **server**. In other words, the server is responsible for all the free AI characters. That way, it provides a simple and clear source of truth as to who decides what actions the free NPCs should perform.

Overall, realizing this game over the network using Unity Networking features really proved to be an interesting challenge as this forces programmers to rethinks

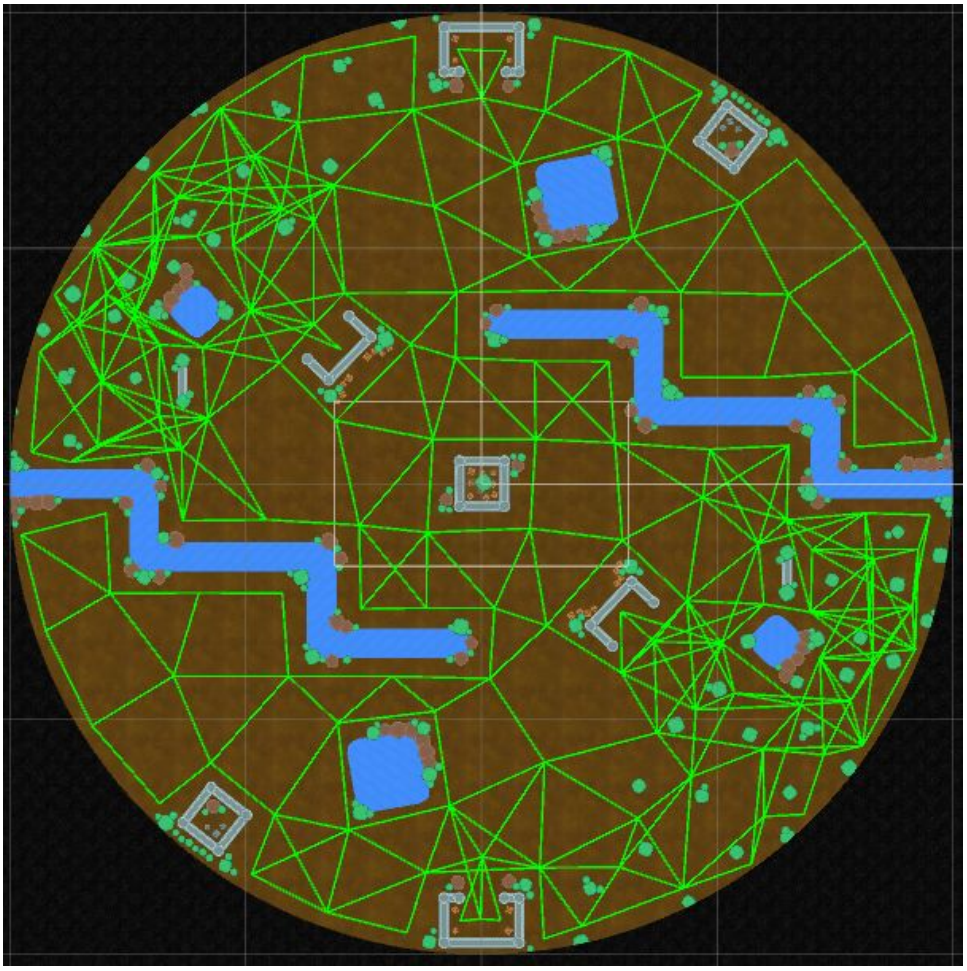
how any simple offline action should be performed over the network in a smooth and efficient manner.

Artificial Intelligence

The artificial intelligence in *Mechanica* uses a decision tree to make its decisions when free (not under drone control). In order of actions, NPCs will first and foremost, try to avoid contact with enemy drones. This will result in them fleeing whenever they come within a certain distance of enemy drones. If those NPCs happen to be armed, they will rotate towards their target and start shooting. The second action that NPCs try to take is to fetch weapons. When NPCs have no weapons to fetch or enemies to flee from, they will try to group up together in order to increase their chance at beating the drones.



The NPCs follow a path finding graph, which is built once at run-time to save computational power. Nodes were placed at corners of the map, between obstacles, and in strategic places, in order to ease the navigation of NPCs around the arena. Only neighboring nodes within a given distance are selected, which results in the following pathfinding graph.



The leading NPCs move along the paths using a seeking behavior, while their followers follow them using an arrive behavior, while simultaneously having some separating factor with the other NPCs and wall avoidance. This makes it so the group follows a random formation and can navigate through obstacles without too much hassle.

Results

In this section, you will find an explanation of the methods we used to ensure the proper functioning of our game as well as well as a demonstration of the different steps of the game.

Quality insurance

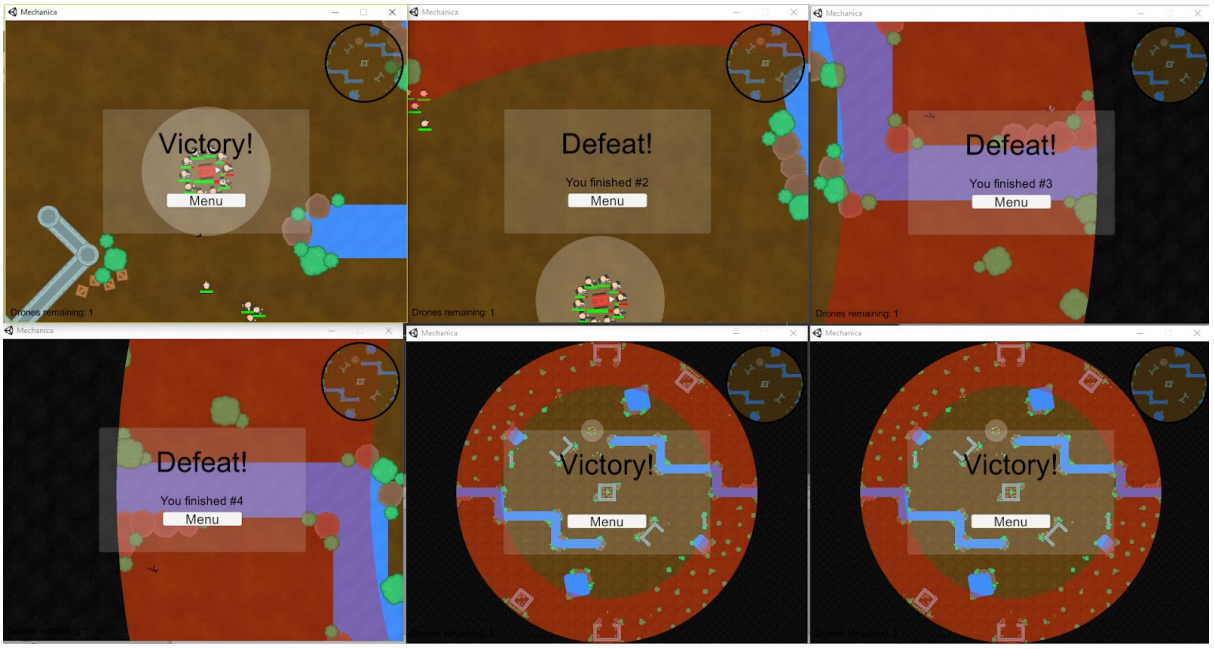
In order to insure the proper functioning of our game, we performed careful testing of every feature over the network. Basically, each time a new feature was implemented, this feature was thoroughly tested on a **client** and a **server** version of the game to make that not only did this feature work properly on the client, but that the impacts were also properly communicated over the network. We also spent some time in order to optimize the different communications made over the network in order to reduce lag as much as possible, while still using some of Unity Networking basic features like NetworkTransform.

Once all features were fully implemented and tested, we went on our final test phase and balancing. To do so, we met at the University labs to play some games against each others. During these games, we took notes of 2 different elements: any glitches, network communication errors or AI issues and guns balancing. This led to us balancing the shotgun and also slowing down the EMP storm progression. After these two testing and balancing phases, we were finally ready to wrap up the project and produce our final build.

Also, please note that in order to help us see the different AI behaviours in action, we added a new user role being a game **viewer**. Instead of being given a drone, a game observer is simply given a point of view of the game from which it can see all of the action. This really helped us debugging as well as demonstrating our project.

Screen shot taken during normal play of the game with 6 clients on a single machine

The lower right clients are viewers and the rest are players



Screen shot taken at the end of a game

User Manual

How to launch and play the game

In order to play a game, one must be connected to a local network and must also have a built version of the project. No other external libraries or tools are required. From there, the user can simply launch the game built version and will be presented with the main menu screen. From there, he/she can either host a match, join an existing match or close the game.

Host a match

When hosting a game, this first creates a joinable lobby. The host is presented with a UI displaying all the players who also joined the game. From this UI, when more than one player are present in the lobby, the host can decide to launch the game.

Joining a match

From the same main menu, a user can also decide to join an existing game. By selecting join, the user is presented a UI where the available hosts are displayed. These hosts are computers also running the game that are broadcasting their connection for other players to join. From there, the user simply has to wait for the host to launch the game.

Game

Once in a game, as described in the controls section of this document, each player can navigate the map with their own drone by using the WASD keys. Also, please note that **all of the game's controls** can be seen at all time by a player by holding the 'F1" key. To orient its drone and formation, a player must aim using the computer's mouse pointer. The mouse wheel and the 1 and 2 keys can be used to

manipulate the drone's formation. The right-click can also be used to lock the formation rotation in place.

When a drone's human holds a weapon, the left-click button will cause this character to shoot. These guns can be picked up simply by including the gun inside the zone of influence of the drone. Shooting can be used to kill other free humans or drones. The drone can also launch the already discussed stun grenade as a gap closing or offensive tool by pressing space.

When destroyed, a player is presented a "YOU FINISHED #X" screen from which he/she can leave the game (or will eventually be kicked from once the server stops). Upon winning, the winner is presented with a winning screen from which he/she can also leave back to the main menu. In case of the host, this return to main menu button will only become available once the game is over (only one drone is standing). This will make sure that the host does not end the game prematurely (before it is actually completed). When a player quits, he/she is sent back to the main menu from he/she can host or join another game.

Note from the team

Our main personal challenge with this game was to provide a fun online experience mixing intelligent AI and player versus player features. We honestly hope this can be perceived in our final result and wish your will have fun playing our game. Thank you!

Changes since the demo

Since the demo, we reworked our AI in order to fix some of the issues where free NPCs would get stuck in certain situations. You can find more details on how we fixed this issue and most importantly the cause of this issue in the section “IMPORTANT NOTE ON UNITY AND BUILD DIFFERENCES”.

Additionally, we also added more checks and some fixes to how we managed some edge cases with Unity Networking regarding players death and mid-game disconnection.

References

A. Bhatt, “Local Network Discovery Using Unet.” *Akash Bhatt*. Updated July 3, 2017. [Blog]. Available: http://bhattakash.com/local-network-discovery-using-unet/, Accessed on: March 10, 2018.

Duion, “Leather Black”, *Open Game Art*. Updated February 17, 2013. [Website] Available:https://opengameart.org/content/leather-black, Accessed on: March 15, 2018.

Kenney, “Topdown Shooter.” *Kenney*. [Website]. Available: http://kenney.nl/assets/topdown-shooter, Accessed on: March 3, 2018.

LuminousDragonGames, “Simple Seamless Tiles of Dirt and Sand”, *Open Game Art*. Updated January 15, 2017. [Website] Available: https://opengameart.org/content/simple-seamless-tiles-of-dirt-and-sand, Accessed on: March 15, 2018.

PngImg, “Bomb PNG image with transparent background.” *PngImg*. [Website]. Available: http://pngimg.com/download/24048, Accessed on: March 3, 2018.

The Prodigy, "The Prodigy - Invaders Must Die (Official Video)." *Youtube*. Updated November 28, 2008. [Website]. Available: https://www.youtube.com/watch?v=EiqFcc_I_Kk, Accessed on: March 14, 2018.

Toby Fox, "ASGORE." *Bandcamp*. Updated September 15, 2015. [Website]. Available: https://tobyfox.bandcamp.com/album/undertale-soundtrack, Accessed on: March 14, 2018.

Toby Fox, "Bergentrückung." *Bandcamp*. Updated September 15, 2015. [Website]. Available: https://tobyfox.bandcamp.com/album/undertale-soundtrack, Accessed on: March 14, 2018.

Unity, “Multiplayer and Networking.” *Unity*. Updated 2017. [Website]. Available: https://docs.unity3d.com/Manual/UNet.html, Accessed on: March 9, 2018.