SMOL

Cyberspace Liberation - Computer-Network-Takeover-Operator delivers Net Neutrality
Triankolos Edition

Conclusion

12.07.2021

Mehmet Dereli

Felix Kosian

Julius Krüger

Louis Hötzl

In Cyberspace Liberation Triankolos has used his army of minions to take control of Cyberspace. The player takes on the role of the Computer-Network-Takeover-Operator and is tasked with the liberation of Cyberspace.

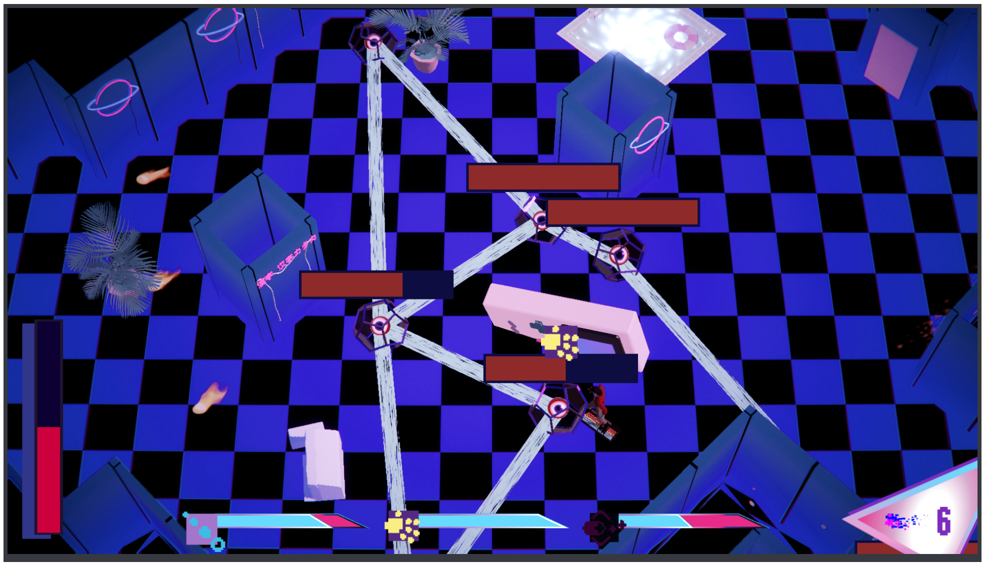

Fig. 1: In-Game Screenshot

To accomplish that the player has to fight through a series of procedurally generated levels to reach the boss. In classic rogue-like fashion, the player has to start over should they die at any point during their run to the boss. To give them an edge, they have a variety of abilities at their disposal, such as a dash, a hack, that allows them to take over an enemy for a couple of seconds, or even an ability that allows them to come back to life once, should they die.

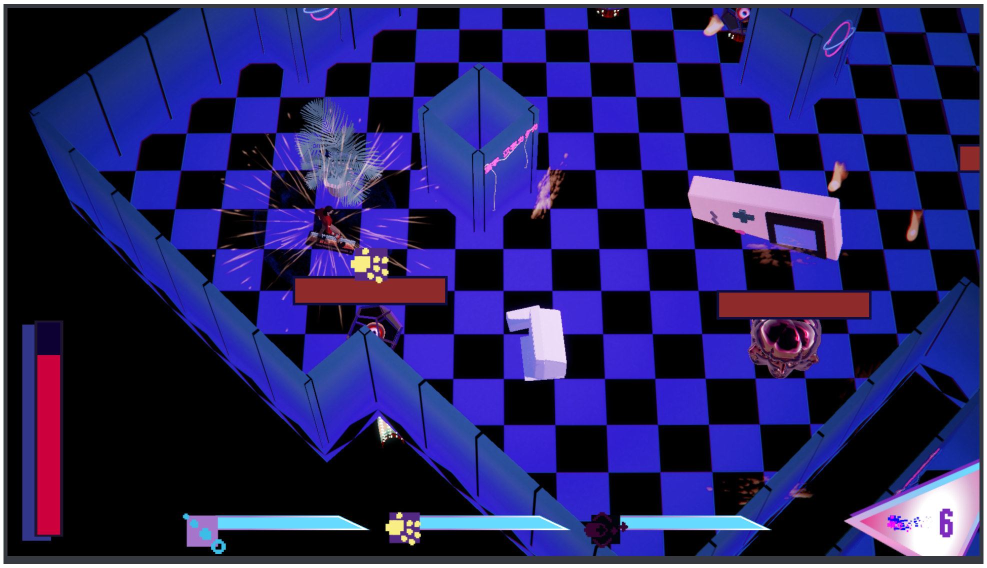

Fig. 2: In-Game Screenshot

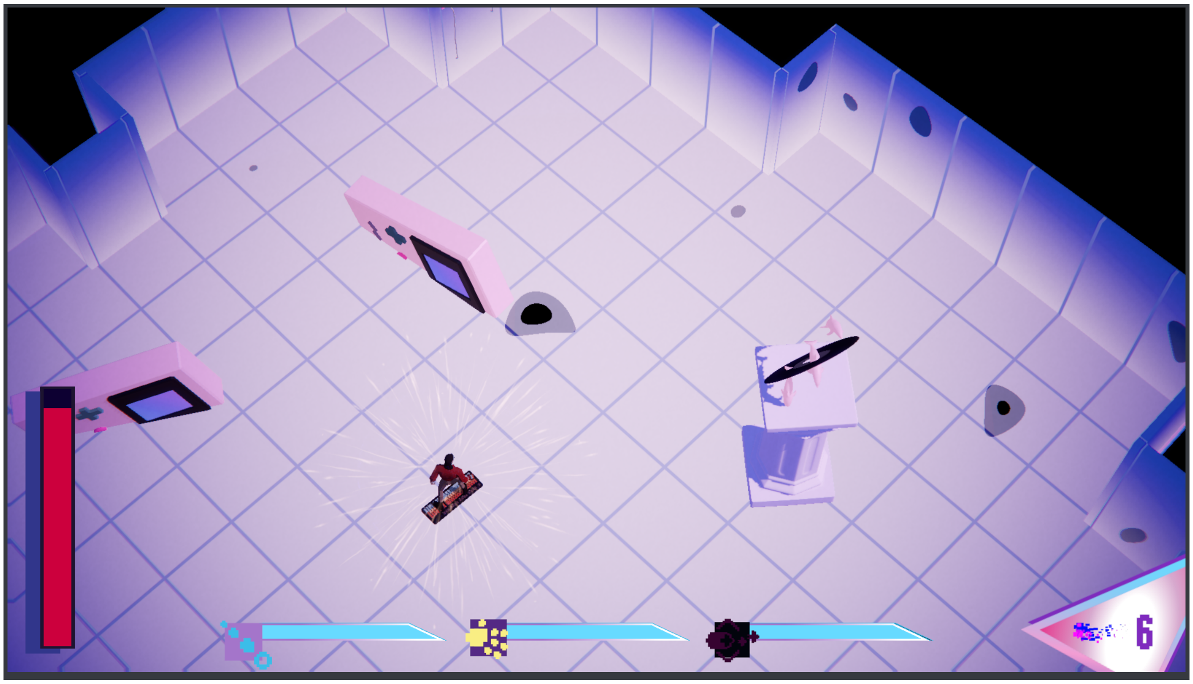

Fig. 3: In-Game Screenshot

Final Changes

For our final release, we have completely overhauled our tutorial. We now introduce the player gradually to the mechanics instead of throwing them into a level and overwhelming them with trying to understand abilities while being shot at. The player learns moving, then the abilities, and, finally, they fight two enemies before they are guided back into the main menu.

We have also redesigned the final boss fight based on player feedback. Simply spamming the shoot button until the boss dies does not offer much of a challenge. Instead, the boss battle now has several stages in some of which the player needs to attack certain minions so the boss becomes vulnerable to the player's projectiles.

We also lengthened the game by adding a couple of levels in order to create a better emotional experience for the player. Instead of having relatively low tension and then spiking at the end with the boss fight, we now gradually ramp up the difficulty curve. On that note, we now also provide the player with a choice between an 'easy' and a 'hard' mode to make the game available to more players.

Finally, we have overhauled the UI to increase its readability and provide the player more easily with the feedback they need from the game, such as their health, or which of their abilities are currently available.

We also fixed a lot of bugs, reworked minor components of the game, and added some more content, but these were less significant than those mentioned in the previous paragraphs.

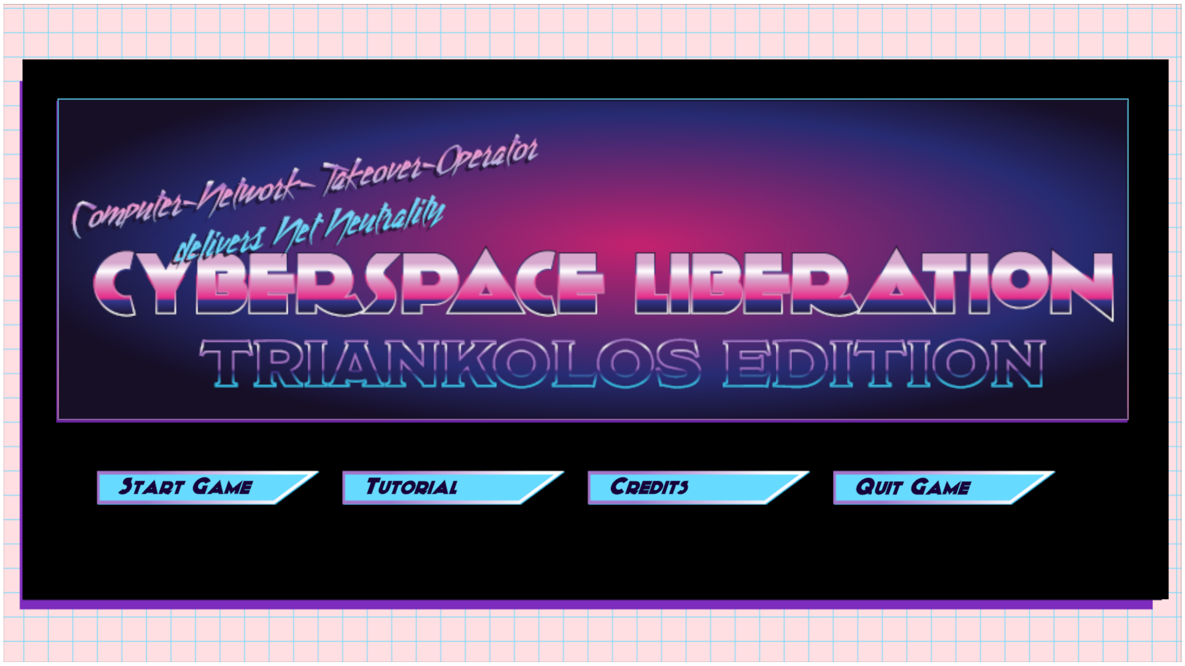

Fig. 4: Main Menu UI Before

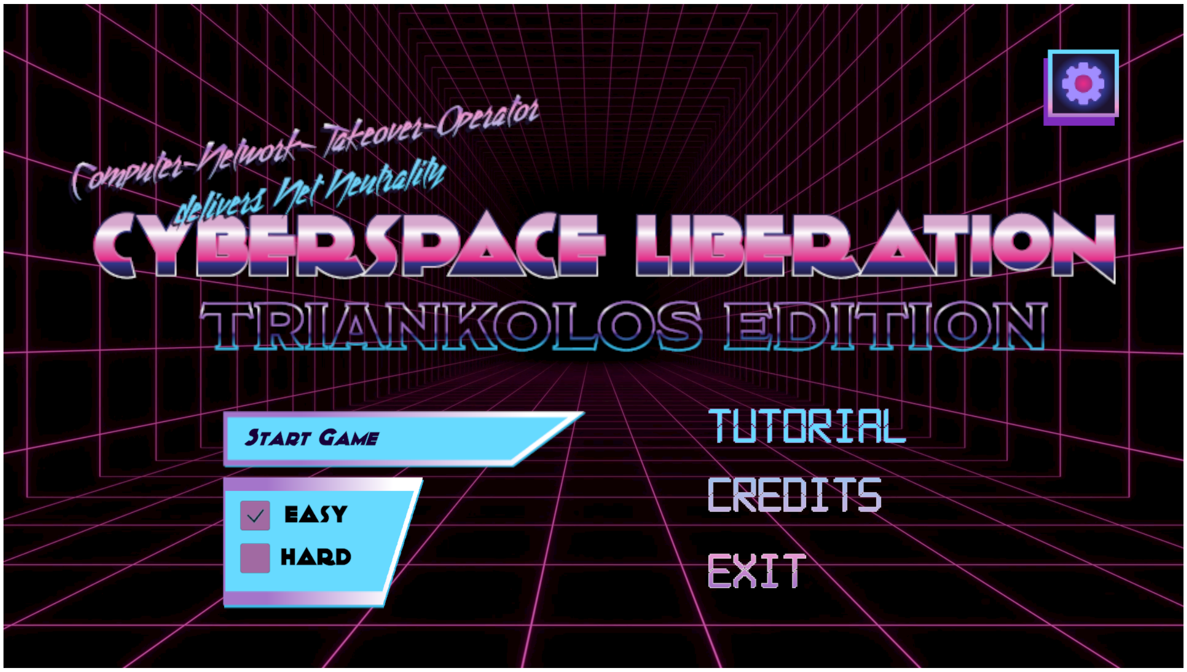

Fig. 5: Main Menu UI After

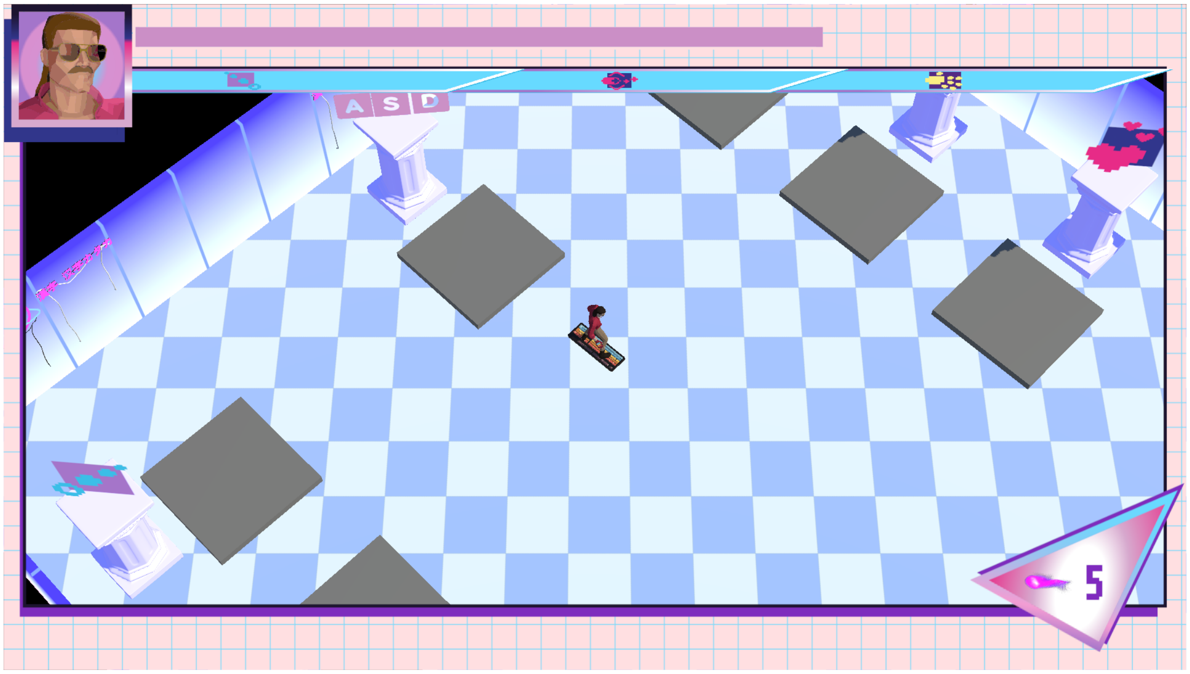

Fig. 6: In-Game UI Before

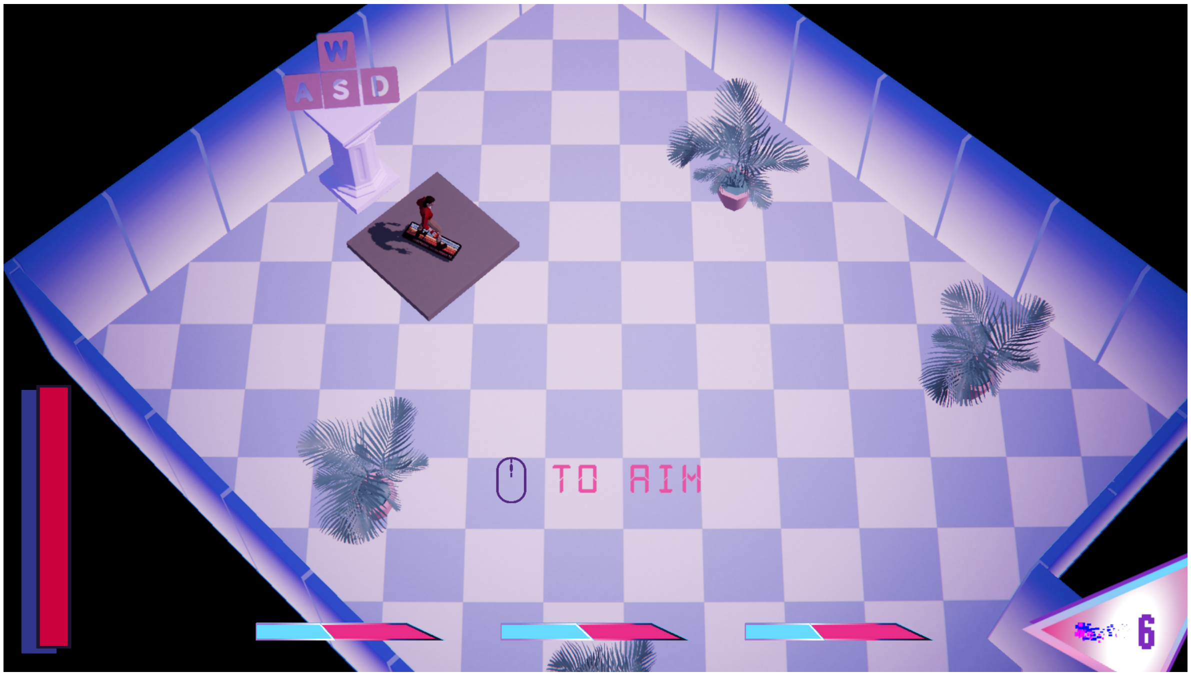

Fig. 7: In-Game UI After

Our Experience

While some of our initial design decisions made it into the actual game we did make many changes as we progressed along. At its core, our game is still a bullet hell/rogue-like with abilities. Our levels are being procedurally generated and we managed to implement two types of enemies as well as a boss. Our initial design also contained different weapons between which the player could choose. This feature got ultimately scratched. Based on feedback we added a progression system to the design, which we ultimately didn't have time to integrate.

We followed a development schedule, but not necessarily the one we developed during the first or second week of the course. We created the initial plan in Excel, but we felt that Excel is not particularly great as a project management tool, so we quickly switched to Trello. We then used our initial schedule together with the milestone deadlines to create a new, more in-depth, but short-term schedule, on a weekly or bi-weekly basis.

Our initial schedule was based on many assumptions about the game design, which features we thought were important, and how long they would take. We had no idea how long those features would actually take and many of our initial assumptions were revised anyway so our requirements changed frequently. We did manage to follow those shorter plans most of the time, however.

One issue with the project structure was that it doesn't leave too much room to iterate. Our initial design disregarded the theme to some degree and also was mechanically heavily derivative of similar games in the genre. Even now, with the final release coming up, we still weren't able to fix that problem with the only thing maybe setting the game apart being its visuals. With a less rigid project structure we maybe could have more easily gone back and fixed issues with the design, but due to the fixed structure we also had to prepare the deliverables for the next milestone so the problems we had just solidified.

Generally, working with a theme can be quite nice as it can provide a lot of guidance, especially if you are supposed to come up with a game design and don't have any ideas in which direction to take it. It also sort of unifies all the games that are developed for the course which is nice. However, we felt that it wasn't communicated clearly enough how important the theme actually is. (could have asked tho) Initially, we were under the impression that the theme was meant to be merely guiding in finding an initial design. It should be used, but wouldn't be super important which is partly why we went for a bullet hell type of game. It would sort of check the theme, as bullet hell games tend to be Kind of chaotic, but would give us a lot of freedom in other regards. However, in the next three or so later presentations we received feedback that boiled down to: Theme should be more present and more important in your game.

However, everything we came up with to properly, deeply integrate the theme into our core mechanics would have required fundamental changes to the game. Time went by and we had deadlines for which we had to deliver and so the game progressed as well. With that it became less and less feasible to implement those changes as they would require huge amounts of reworks for which we simply did not have the time.

Towards the end of the project, we had a huge problem with motivation. We think in part this was because we felt we had to crunch to make the alpha release deadline. After that motivation was pretty low, but we still had to keep going as there were still four weeks until the end of the project.

Another issue was that we worked with a bunch of artists who had no experience working on games, so we had to manage that as well. We probably should have introduced them better as some of the issues we later experienced can be traced back to that.

The greatest success during the project was probably when we gave the game to players during playtesting. Our players all enjoyed the game and seeing people appreciate something you created is just a great feeling.

Overall, we are quite happy with the final result of our game, since players enjoyed it, which ultimately is the goal. However, we would not say that the game fully succeeded from our perspective. Personally, we all learned a lot during the development process. It was the first time we attempted procedural level generation and we are quite happy with the result. It was also the first time that we used behavior trees to implement an AI. So from a technical viewpoint, we would argue that we were successful. What we failed at, however, was finding a clever design twist, in a time frame that would have allowed us to integrate it, to set us apart from similar games in the genre. As a result, our work is quite derivative and, at most, sets itself apart through its visuals. We are split on this last point, however. Some of us would have liked the design to be more unique. Some are fine with it, as we created a fairly polished game that fits well into a programmer's portfolio.

Overall, we enjoyed the course. We had a lot of fun making the game and we learned a lot doing it. We do have a few thoughts on how to improve the course, anyway. For one, we think that it might be good to iterate faster with shorter deadlines, but also shorter presentations. For example, it would have been nice to have less time for playtesting, then have a week or so to improve and fix the game and then have another week of playtesting to get feedback again and see if the game actually improved or whether new issues arose from the changes.

It would also be great if the course could adapt more to the needs of the individual games/groups. In our case, this could have meant more time to take a step back and fix issues with the initial design rather than having to somehow fix the initial design and at the same time press on, because the next deadline is breathing down our neck. We only have so much time for a course as this is not our only course after all.

We also think that it would be nice if some aspects of the course were more clearly communicated and less of a mess. For example, in some cases the deliverables that were required according to moodle differed from those listed in the project structure document. The project structure document felt kind of outdated in general which wasn't great. Another thing, that in hindsight would have been interesting to know, would be what we are being graded on. For example, we are still unsure how important the theme or a unique design actually are in the context of this course. We do need to admit that these communication failures are partly on us, however, as we always could have asked, but ultimately didn’t.

As a final point, on which we are split, we would add that it could be worth thinking about how important design and how important tech should be in this course. While focussing on both can be immensely valuable, some students may be put off by this as this is a computer science degree after all, and some want to code rather than worry about design issues.