Dev Update 1: Interims

Progress update #1.1 - On the way to a functional minimum - Nov 25, 2017

In the past two weeks, we got almost all of our functional minimum features done. As can be seen in the (already outdated again...) video above, Hikari no to so far has a working lobby system (based on the Unity Technologies lobby asset) that is set up for both local area play and online matchmaking using Unity's UNET services. A lobby can hold up to 5 players (1 master, 4 crawlers), with the host having "admin" capabilities like kicking players. For now, we assume that the host is also always the VR player, on one hand because the master is the unique core feature, as well as because it is likely that the VR machine inherently has enough processing power to handle the additional server work.

Next to the networked lobby, we of course also find networked gameplay. This too is powered by Unity's built in networking framework, albeit not based on a particular sample asset. As of this stage, the server handles AI computation, hit detection, and a handful of broadcast calls, while the clients mostly just synchronize entity transforms and only actively communicate with the server to handle player attacks. We try to always handle network communication efficiently so we can keep the traffic and latency requirements as low as possible.

The enemy AI went through two iterations by now. Our initial design was merely based on range checks and line-of-sight raycasts in order to detect the nearest visible and viable player target to attack, without regard for any more complex behavior like patrolling. But exactly that is what we felt was missing, perhaps not for the functional minimum, but for peace of mind. The second iteration now theoretically supports more modular behavioral patterns and in its current form offers detection, patrolling and attacking. There is still the question of how e.g. patrolling will work in tandem with random level generation, however we postpone those issues to a later date.

The crawler players are a rather simple setup for now, they can move for-/backward and strafe, freely rotate the camera - which also rotates the player and attack enemy entities.

Both enemies and crawlers use Unity's handy NavMesh feature set to facilitate pathfinding and level collision.

The VR master player is coming along nicely as well. We prioritize HTC Vive compatibility for now, but have at least verified that Oculus Rift is also compatible with our setup and should only need some tweaks to improve controls and adapt to possible constraints in >270° tracking. We found room-scale and hand tracked virtual reality play to be a very enjoyable and captivating experience, even when devoid of any meaningful gameplay in this early stage. Our Vive controller control setup as of now consists of a selection radial for master abilities and direct hand-pointing at intended targets - with a certain allowed deviation range to account for sensor jitter and player inaccuracy.

We are still heavily tweaking scaling of both the master and the dungeon, as we are not yet set on how gigantic the VR master should ideally be to be able to overlook a significant area yet still retain the ability to lean down to focus on encounters (and which potential locomotion requirements this entails). Additionally, we feel the dungeons need to be far wider and more open than the sample in the video in relation to the ground entities.

Overall, what is missing for us to reach the functional minimum target is to complete basic VR master interaction with ground entities, a more appropriate sample dungeon, a more properly tweaked sense of scale and a basic end condition/handling - instead of just endlessly respawning the crawlers. While working on these tasks, we are of course also starting to work out more detailed specifications for the low target, such as crawler classes, abilities, collectible items, master skills, and a more complete game framework, e.g. offering a more "complete" UI.

Unfortunately the embedded video does not showcase all the latest progress, but due to the daily rate of progress, it is futile to try and get every new change in there.

Progress update #1.2 - The low target? Eeeh not quite - Dec 5, 2017

Dev Update - Jonas

My focus over the last week was to **refactor and polish** the Dungeon Master in VR. In addition i wanted to finalize the functional minimum which includes a buff/debuff ability for the master and a synced appearence.

The master can now select either "**buff**" or "**debuff**" from his radial menu, and apply it to crawlers or enemies as an **effect** (Artem and Inshal have more on that) by pointing at them. In order to implement a soft selection, targets will be chosen based on who is closest to the pointing direction and still in range. The selection is visualised by a bezier curve (blue for buff, orange for debuff) that's being shot from the Master's hand to the target. To give the master a better feeling for his actions there is also force feedback for most of his actions.

In addition the master is now visible to the crawlers (and he's so cute! :3) which should enable primitive communication. The headset and the controllers are now tracked by a **network synced visual representation**, buff and debuff rays are also visible to players.

Next week I'll try and find a fitting hand model for the master, preferably a rigged hand mesh that suits the art style. This will allow us to implement advanced master-crawler communication using different hand postures and some better visuals for abilities in general. Additionally, physics based abilities (heal, fireball, etc.) are to be explored.

Dev Update - Artem and Inshal

We were working on the baseline framework that supports:

- Different AI behaviours for enemies
- Different weapons for enemies/crawlers
- Various effects: buffs, debuffs, abilities
- Character stats, classes
- Simple UI for crawler

While we already gave some comments on AI patterns earlier, now we would like to dive into **the effects system** and **t he abilities and classes** for crawlers. After a fruitful discussion, our team came up with four classes for our future release: soldier, infiltrator, berserker and... tank? Anyways, we don't know the final names for our classes yet, but we outlined the main skills and gamestyle that every class would have. In the interim demo we are ready to present two prototype classes, each of them has two active and one passive ability.

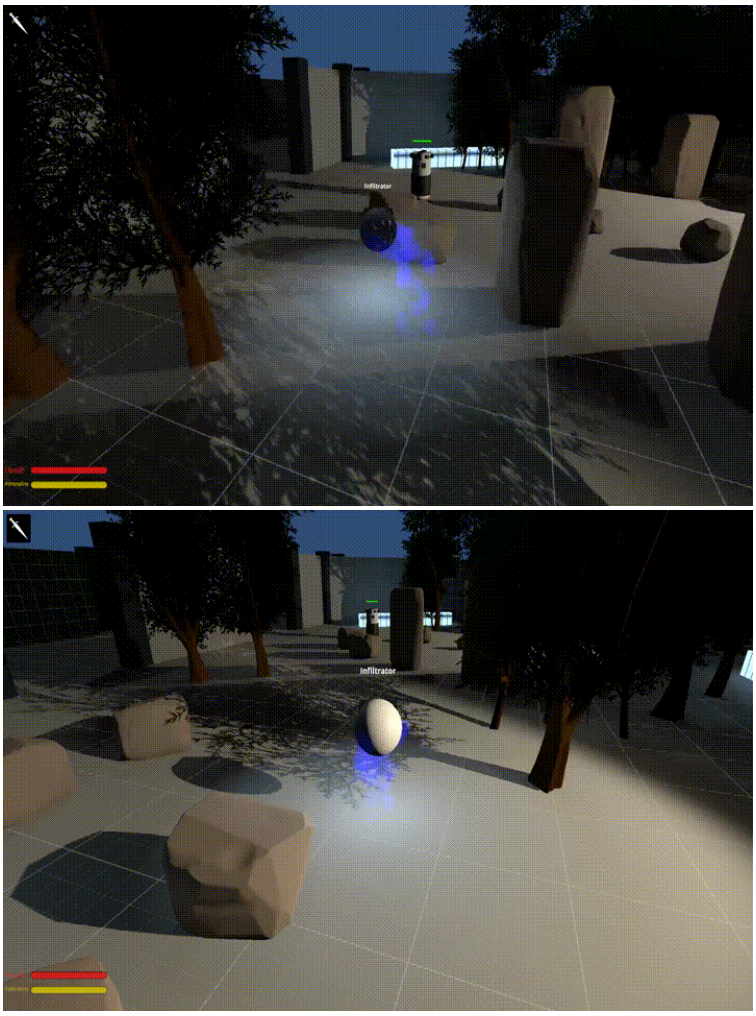
Soldier + Berserker

Soldier has a rifle that shoots. Being ideal for killing enemies at a distant range, she can greatly damage the incoming army before it gets too close. Additionally, she possesses the unique ability to emit destructive laser beam that goes through walls and enemies!

Since we haven't been very certain about how to implement other abilities for the soldier, for the purposes of the interim demo we spiced up the soldier class with some of the berserker's abilities. Berserker can gain adrenaline for killing enemies and when the adrenaline bar is full, she can go RAGE: katana replaces the gun for a while, and instead of fighing from afar the crawler runs and destroys everything that she can reach. Neat!

Infiltrator

He has a short sword and he is not strong at all. But his abilities are superior: firstly, he can go invisible and sneak past the enemies; secondly he can detect enemies hiding behind the walls. It makes this class perfect to complete the game killing as less enemies as possible. However, there is a reason why this class is called "infiltrator" and not, let us say, "pacifist" or "enemies lover": his special passive ability allows to sneak on an enemy and back-stab it with 4x damage!



For the enemy detection, several passes of rendering are used, when some enemies are drawn again with different shaders and the images are blended. At the and, we implemented a nice utility that allows us to highlight labeled parts of the scene in various ways (or apply more complex effects to them) with only a couple of scripts.

Technical notes

Up to this moment, we have a flexible system (with modular approach) that allows to add various types of buffs, debuffs and abilities. From the architectural point of view, everything that can be applied to a crawler or to an enemy (or to any other hypothetical character in the level) is basically one entity: **effect**, which can be **enabled** or **disabled**. So, an active ability is nothing but just an effect that crawler can apply to itself under certain conditions. Buffs and debuffs that dungeon maser gives out to crawlers and enemies are also effects. Passive abilities are effects that are enabled in the beginning of level and never disabled. Traps and some enemies can also, theoretically, apply effects. Every effect, its cost and duration, as well as its icon and name, can be set up independently from the game logic, which makes it easy to add new effects and tweak the game balance. Another good thing is that we made some effort to sync enabling/disabling effects over the network and reduce the technical burden while designing and implementing our game ideas. Effects are stored as the Unity Assets.

For now, every effect applied to a crawler, is automatically shown on the UI. It helps with debugging and looks good. Some stats, like health and adrenaline for the berserker are also exposed in the UI.

See you on interim demo.

Dev Update - Paul

Lobby work + networking

Half my work since the last update pertains to the lobby system and UI. For starters, the HMD checks were broken and out of sync before. Now they properly detect if and if so which type of VR headset is connected and displays as much in the lobby player list. The point of this, theoretically, is for everyone to see who is using a VR headset and for the host to be able to pick who of the eligible players will be the VR master. However, for the foreseeable future this pick will not actually be used, as there are still inconsistencies in the networking for a non-host VR master. Additionally, this might open up avenues of potential VR *crawlers* as well, which are absolutely not planned in the current schedule, but are a possible extension for the future.

Second, we needed some sort of class selection for the crawlers. Now the design decision was to put this selection into the lobby instead of the game level, primarily to simplify development as it didn't need an entire new UI for the selection. Secondly, we feel it is advantageous to figure out a team composition before the match starts, both to save time and to avoid potential dropout delays after people are unhappy with the picks and leave a match. There are still a few troubles related to accurate hiding and showing of other players' selection depending on VR master status etc, but these are basically just relevant if at some point we decouple the VR master from the match host. Plus, the backend implementation is not particularly clean with respect to how the UNET system usually handles player spawning, but it works pretty sleek for our purposes.

Third, a basic recolor of the lobby UI made it a slightly better fit for the theme and setting of our game.

As a side activity, there was a constant lookout and troubleshooting for possible networking bugs and sync issues.

End condition + UI

Another addition to the game was a basic end condition setup. Now if all crawlers die or all enemies are killed, the match is lost or won respectively. In either case a simple message is displayed on the screen and the player gets the option to return to the game lobby. An equivalent setup for the VR master is actually still missing as we did not manage to implement a VR-based UI interaction system on time.

Interim demo map + scaling

The third big task was to tweak scaling of the dungeon with respect to crawler and master size, and once a suitable scale is found, to build a small demo level for the interim. Overall, hallways and rooms are now about 4 to 6 times as large as in the first prototype video. The master was also scaled up by about a third. The demo level is an attempt at creating a believable office floor structure, with a larger cubicle area, a conference room, a recreational section, a large server room and a central court with some nature objects to provide a nicer work atmosphere. The remaining rooms don't have a specific function, but in light of our plan to replace the fixed structure with dynamically generated levels, there is little point in fleshing out a lot of detail in this demo level. One aspect that fell short, however, is giving the level a high-rise feeling with large windows and perhaps a steep view down onto a larger city. As it stands, the "dungeon" gives a more realistic sense of local scale and should be a good starting point to cut modules from for our planned level generator.

Target practice

To recap, we believe we have certainly completed the goals of our Functional Minimum and are on a good way to our Low Target. Our core gameplay system are mostly in place, the majority of Low Target tasks ahead are related to creating more enemy variety, complete four different crawler classes instead of two, more fitting 3D models for our entities, more direct crawler signalling and a first foray into a wider level setup. In short, mostly visual and diversity efforts.