Figure 1: Screenshots of the original *Hedgewars* interface.

HedgewarsSGC: A Competitive Shared Game Control Setting

Pascal Lessel[1], Maximilian Altmeyer[1], Matthias Hennemann[2], Antonio Krüger[1]
[1]German Research Center for Artificial Intelligence (DFKI), [2]Saarland University
Saarland Informatics Campus, Germany
firstName.lastName@dfki.de
hennemann-matthias@web.de

ABSTRACT

Sharing game control (SGC) is a multiplayer context that is considered within games user research. With the popular "*Twitch Plays Pokémon*", settings of this type have also received broad media attention. In this paper, we introduce and describe *HedgewarsSGC*, our modifications to the open-source game *Hedgewars* to investigate different player roles in this shared game control context: besides considering competing groups who share control over their units via input aggregators, it also provides options for spectators that do not want to give up individual control. Thus, *HedgewarsSGC* is an approach to investigate SGC in such a scenario and additionally, allows further reasoning about input aggregation.

KEYWORDS

Shared character control; input aggregation; game development

INTRODUCTION & RELATED WORK

Typical multiplayer games provide every player with control over a primary actor (for example an in-game avatar in roleplaying games). Within shared game control (SGC) settings this differs and

CHI'19 Extended Abstracts, May 4–9, 2019, Glasgow, Scotland UK

ACM ISBN 978-1-4503-5971-9/19/05.
https://doi.org/10.1145/3290607.3313024

Hedgewars (see https://hedgewars.org) is a strategy artillery game in which teams consisting of hedgehog avatars try to eliminate all other teams. Above every hedgehog the team name and the hedgehog's name is shown (both are customizable). Interface elements show how healthy the remaining hedgehogs are, which team's turn it currently is, the turn time, which hedgehog is active and the wind direction and strength (affecting projectiles). A player can move the camera to receive an overview over the game area. The active hedgehog can act as long as turn time remains, a weapon is fired or it receives damage. The latter is possible through falling damage, via its own weapon or by activating one of the obstacles on the map (e.g., exploding mines).

Every team has a set of weapons and tools (providing special effects) that the team shares (visible on the right border of the middle picture in Figure 1) and which can be used by the active hedgehog. Some weapons require the player to simply click on a position on the map. Other weapons require more complex interactions, in which the player needs to aim by moving a crosshair in a 180 degree radius in front of the active hedgehog, followed by deciding how much power a shot should have. Some weapons and tools have even more complex interactions (see the supplementary video for an overview).

The game can be played by taking turns on one computer or over networks. Additionally, teams can be controlled by an AI. The game is usually played with the mouse and keyboard.

Sidebar 1: Hedgewars in a nutshell.

opens up a whole range of new questions: Sykownik et al. [8] states that "*shared control can intuitively be understood as a game control mode, in which players collectively control one single game character*".

SGC has been considered within different settings from a games user research perspective, but also to investigate crowd dynamics: Gutwin et al. [1] considered a chess game in which up to 16 players per side are able to play the match simultaneously. While the chess pieces can be moved based on the normal rules, every piece could be moved at any time for both sides. No further means were implemented to support SGC. It was found that players started to develop strategies and team coordination occurred, showing the usefulness of investigating these settings to explore crowd dynamics. Sykownik et al. [8] created a game in which a sphere needed to be navigated though an environment. In their approach they tested different methods for SGC. In a user study with four co-located players sharing control, they found that the game was enjoyable and that losing control compared to a typical multiplayer game is not associated with a negative experience per se. Rozendaal et al. [7] investigated shared game control in an *Asteroids*-like game. In groups of three players, they compared conditions in which the control options varied and found that while control sharing impacts the feeling of autonomy negatively, it affects the experience of sociality positively. Overall, these works show that it is beneficial to investigate SGC as a novel way of playing multiplayer games.

With "*Twitch Plays Pokémon*" (TPP) SGC also received attention in the context of game live-streaming [2]. At its peak, 121,000 people played the game simultaneously (see https://goo.gl/P6Rd2Z). Work such as [3, 6] focused on TPP and investigated what kind of communication happened and how the offered input aggregation modes were used. We also contributed to this topic, as we investigated an extended TPP version [4], where we provided means to raise the engagement and self-administration options of the playing crowd. With *CrowdChess* [5] we were able to analyze how effectively a crowd makes decisions in SGC against an artificial intelligence (AI). So far, to our knowledge, only this type of SGC (one SGC group against an AI) has been considered scientifically. With *HedgewarsSGC*, we present a test environment in which SGC groups play separately against each other, allowing us to investigate how dynamics change in a digital game offering a broader range of interactive options than chess. In this paper, we present the design and first experiences we have gained with it.

GAME SELECTION: HEDGEWARS

We decided to use an **existing open-source game** so as to be able to modify the game engine itself (to, for example, add features for the SGC context) and to have access to an **active community**. The latter is useful if questions arise during the development and for access to an existing player base that might also want to play our modified game, making in-the-wild studies easier later on. Furthermore, the **game mechanics should be easy to understand** to ease the onboarding of new players. The game should have a **competitive setting**. To compare the *HedgewarsSGC* results with our previous work [4, 5], we also aimed for a game that can be used through **live-streaming platforms**, e.g.,

Figure 2: User interface adaptions for the different phases in *HedgewarsSGC*.

controlling the game should be feasible with text input (as they offer chat as a main input method [2]). We decided to use the game *Hedgewars* (see Figure 1 and Sidebar 1): it is an open-source game which is under development since 2006 and still has an active community; it is turn-based (thus making it easier to use in a live-streaming context), offers a competitive setting with multiple characters and uses the game principles of the popular commercial game *Worms*. The first *Worms* game was released in 1995 and every one to two years a new game was released (the last release was in 2016). Considering the complete franchise, until 2015, 70 million units were sold (see https://goo.gl/mfA64X). If a player knows how to play *Worms* he/she is directly familiar with *Hedgewars* as well. Finally, the basic controls (e.g., moving, aiming, shooting) are quite easy to manage, even through textual input.

SHARED GAME CONTROL MODIFICATIONS TO HEDGEWARS

In this section, we present our modifications to *Hedgewars*. They were tested in a one-hour session on the live-streaming platform *Twitch* with six participants, who identified themselves as video game players and were already familiar with *Hedgewars*. The participants played several matches as spectator and players in different teams. Through observation, semi-structured interviews and a questionnaire (details were omitted due to space reasons), we recognized parts that were problematic such as reduced game state awareness (e.g., reduced game overview because of missing camera control option), slow game phases (e.g., turn phase are taking too long) or the need to add further feedback and assistance options (e.g., unclear how far a hedgehog can jump). We implemented improvements to mitigate these issues and highlight them in italic and denote them with *(UI)* subsequently.

Finely-structured turns and interface changes

In *Hedgewars*, the active player can do several actions in any order (e.g., choose a weapon, move, target and fire). This seems problematic for SGC, as for example, a fired weapon will terminate the turn for this team. As every individual might have different game goals (as was seen in TPP [3, 4]), we structured a turn to also allow more options for input aggregations (see below) and to mitigate lag issues of the live-streaming context [9]. Every turn is structured into the following phases (every phase has its own time limit). See Figure 2 for the interface changes we describe for every phase:

(1) **Move**: The active hedgehog can be moved by entering a sequence of commands (e.g., *fj right 5*: it would jump forward and move five steps to the right). *This phase ends after one sequence is executed to fasten the gameplay (UI).* To help players to play the game with text input only, we show a indication of how far a hedgehog can move/*jump* (UI).
(2) **Weapon selection**: The weapons/tools are shown with their names. The name can be entered into the chat to select it (e.g., entering Bazooka). For this, we enlarged the weapon menu, added the weapon names and made *the menu 50% transparent to allow players to see through it (UI).*

Anarchy: Only the first valid command entered is executed.
Plurality: Commands are collected in a time interval and the command most often selected in this interval is carried out.
Mean/Median: These aggregators work similarly to *Plurality*, with the difference that for numerical values the mean/median value is calculated before carrying out the command: Assuming *Mean* is active, if player A enters *right 10* and B *right 20*, *right 15* will be executed.
Conformity: A weighted plurality vote, with weights based on the player's conformity to the audience. The weight is continuously adapted, i.e., increased (decreased) if the choice is (not) congruent with the most popular vote.
Expertise: A weighted plurality vote with weights based on the player's expertise. We use an *Elo* rating system (see https://goo.gl/m98XBf). After a match, the average *team Elo* ratings are compared and updated accordingly for every player (increased/decreased in relation to whether the player belonged to the winning/losing team).
Proletarian: A weighted plurality vote with weights based on the player's expertise: the lower the expertise, the higher the weight.
Active: One player is randomly selected and takes control over the game, i.e., only this player's commands are carried out. As long as this player provides inputs (and this aggregator remains active), he or she remains in control.
Leader: Leader combines *Active* and *Conformity*; instead of selecting a player randomly, the one with the highest conformity is selected.

Sidebar 2: Aggregators used in *HedgewarsSGC*. These are based on [4, 5].

(3) **Target**: If a weapon was selected that requires a map target, this phase is activated. In *Hedgewars*, a target can be placed with a mouse click on the map. As we judged chat controls for controlling the camera too cumbersome, we require players to enter the coordinates of the target, which will be shown on the map in a grid. Periodically, the camera moves to all enemies and *remains there for two seconds to allow players (UI)* to select their target appropriately.
(4) **Fire**: In this phase commands are specific to the weapon selected. For many, this means to define the fire angle and power. The phase ends as soon as the weapon has no ammo anymore (some weapons allow more than one shot). Players are able to add moves after fire commands (e.g., to drop dynamite and to move away). As the unmodified game allows a couple of seconds after firing before the turn ends, we have decided not to create a separate move phase after this phase, but allow chaining: For example, *f 95 80 left 4*; fires the weapon in a 95 degree angle with 80% power and afterward the hedgehog moves four steps to the left. We added an interface indication around the hedgehog displaying the degrees. *After each turn, the camera zooms out to give the players an overview over the complete map, and the names of hedgehogs that are still alive are shown at the bottom of the interface to ensure that tactics can be more properly evaluated (UI).*

Aggregators

Similar to our SGC systems [4, 5], players can decide how individual inputs are aggregated (see Sidebar 2). *HedgewarsSGC* allows aggregations per phase: players can vote for their aggregator preference by entering chat commands denoting the phase and the desired aggregator. This is an ongoing plurality vote, i.e., the aggregator with the most votes is the active one in the phase. In phases such as *Weapon selection* it is easy to aggregate the players' opinions, as there is a finite set of options (i.e., the different weapon names). In contrast, *Move* is more difficult because of the unrestricted sequence length. Here, the aggregation works on a per-command level: For example, suppose *Plurality* is active and player A enters *right 5 fj right 10*, B *left 3 fj*, C *right 5* and D *right 5*. The first command of every sequence is considered ({*right 5*, *left 3*, *right 5*, *right 5*}); the winning command is thus *right 5*. The next command is now considered. B, as a player who did not want *right 5*, will not be considered anymore, leading to {*fj*, *empty command*, *empty command*}. C and D do not want to move the active hedgehog further, which resulted in an "empty command" statement for their sequence. As this is most often "selected", the aggregation stops here and only *right 5* is carried out.

Roles and information dissemination

Users that are registered within the streaming platform (and thus are able to use the chat) can actively influence the game: a user can decide to either join a competing team (each is controlled by the corresponding group via SGC) or remain as a spectator. If a user joins a team, he or she can play the

Cheer: The buyer is able to enter a text in the buying process, which is then shown as an in-game text message.

Raise Water: The water level of the game rises, and thus hedgehogs might drown and parts of the map disappear.

Light Weapon Crate: The buyer enters the name of the team while buying. When bought, the weapon crate spawns above one hedgehog of this team, is directly collected and provides access to ammo or a new weapon. Crates are a default game mechanic of *Hedgewars*, but these usually spawn randomly on the map.

Heavy Weapon Crate: Same as the *Light Weapon Crate*, but gives access to stronger weapons that are not available otherwise.

First Aid Crate: The buyer enters the name of a hedgehog while buying. A first aid crate spawns, similar to the *Light Weapon Crate*, above this hedgehog and is thus collected and heals hit points. These crates also belong to the standard game mechanic, spawning randomly on the map.

Poison Hedgehog: The buyer enters the name of a hedgehog while buying. This hedgehog becomes sick and loses a constant amount of health every turn.

Kill Hedgehog: The buyer enters a name of a hedgehog while buying; this hedgehog dies instantly.

Sidebar 3: Available spectator items and their effect when bought. Most of the items directly influence the game.

game (i.e., provide input aggregator preferences and enter commands in the different phases that are aggregated accordingly). Spectators also have means to affect the game, even when not part of a team. These interactions (see below) happen on an individual basis, i.e., without SGC, and thus are an option for those that do not want to give up control. Every new user receives an amount of a virtual currency ("coins") and with it, a spectator is able (again via chat commands) to buy items (see Sidebar 3). The items have different costs (depending on how big their impact is) and the more items per type are in stock (which resets periodically), the cheaper they get. Bought items are used automatically between turns and the buyer is announced via in-game messages. Users can earn coins when they are part of a team and finish matches (they receive more coins if they belong to the winning team), or if they remain spectators and bet (which is another participation option) on which team will win (betting odds are constantly updated in relation to the teams' health levels).

The game and its changes for SGC are explained via a textual description in the channel overview page. In addition, we implemented a chat bot that always announces the current phase *and how valid chat commands in this phase look, adapted to the current game state, e.g., only providing weapon names of available weapons in the weapon phase (UI)*. This bot also informs viewers about hedgehog deaths, the active hedgehog and whether the match is over. The goal was to make it easy for visitors to directly take part in the game. The bot provides relevant information on the viewer's state (available coins, expertise and conformity values, which team the viewer belongs to, etc.) or *issues (e.g., too little money to buy a certain item) (UI)* via direct (whisper) messages.

A sidebar (see Figure 3) alongside the streamed game window depicts information about the state of the aggregators, the recent command inputs, the betting odds and the shop items (i.e., availability, costs and how to buy them). After a match, we present a statistics screen, and here players can switch teams (or become a spectator). The stats screen shows the rewards players on the teams received, which bets were placed, how many coins were spent for different item categories and how high the average expertise per team was. If no items were bought/no bets were placed/no aggregators were used, these areas are used to "advertise" these elements, briefly explaining how these work by rotating through items/aggregators. The screen also shows historical information, i.e., the win distribution of both teams, how many coins were spent on shop items and the betting loss/win ratio.

CURRENT STATE AND CONCLUSION

In this paper, we presented the current state of *HedgewarsSGC*, a test environment to investigate shared game control in scenarios in which multiple people share the control on different game parts and play against each other. With the presented system (a video game with only one player goal), we complement our systems presented in [4] (a video game with various player goals) and [5] (an analogue game with only one player goal). In particular, it allows investigation of two competing SGC groups as well as the role of spectators, which has, to our knowledge, not been considered so far. With

Figure 3: The information sidebar shown next to the main game.

the current state of *HedgewarsSGC*, we are not bound to a particular live-streaming platform: as long as the platform offers an API to retrieve chat messages, *HedgewarsSGC* can be easily integrated.

After the usability changes, we created *HedgewarsSGC* channels on the platforms *Twitch* and *Mixer* and we received support of the *Hedgewars* developers through announcements on their *Facebook* (~3000 subscribers at that time) and *Twitter* (~370 followers) pages. Nonetheless, we also experienced the long-tail issue associated with live-streaming channels [4]: although users visited the *Twitch* channel (231 in 56 days; 6 in 42 days on *Mixer*), it was rarely the case that two users were online in parallel (91% of the matches were played alone). As a consequence, we re-used the *Hedgewars* AI early on (after 8 days on *Twitch*) and activated it whenever only one team had players to at least allow users to experience the modified game with an opponent. From a scientific point of view, this shows that aiming for "in-the-wild" studies in the context of SGC and/or live-streaming platforms is not trivial: independent of the actual quality of the stream, it cannot be assumed that new channels directly attract a large enough user base to conduct user studies.

Therefore, the following next steps are: 1) holding "events" (i.e., announcing dates in which the game is available to raise the likelihood of multiple parallel players) instead of being available 24/7; 2) asking established streamers to "host" the game (and thus making it potentially attractive for their user base) and 3) developing platform-specific features (e.g., allowing users on *Twitch* to control the game through their extensions concept; see https://goo.gl/Kc9igd), to further ease the interaction and explore the options the live-streaming platforms offer for SGC. Overall, these aspects will help to understand the SGC phenomenon further and thus are relevant for the area of games user research.

REFERENCES

[1] Carl Gutwin, Mutasem Barjawi, and David Pinelle. 2016. The Emergence of High-Speed Interaction and Coordination in a (Formerly) Turn-Based Groupware Game. In *Proc. GROUP '16*. 277–286.
[2] William A. Hamilton, Oliver Garretson, and Andruid Kerne. 2014. Streaming on Twitch: Fostering Participatory Communities of Play Within Live Mixed Media. In *Proc. CHI 2014*. 1315–1324.
[3] Harris Kyriakou. 2015. Twitch Plays Pokémon: An Exploratory Analysis of Crowd Collaboration.
[4] Pascal Lessel, Michael Mauderer, Christian Wolff, and Antonio Krüger. 2017. Let's Play My Way: Investigating Audience Influence in User-Generated Gaming Live-Streams. In *Proc. TVX '17*. 51–63.
[5] Pascal Lessel, Alexander Vielhauer, and Antonio Krüger. 2017. CrowdChess: A System to Investigate Shared Game Control in Live-Streams. In *Proc. CHI PLAY '17*. 389–400.
[6] Dennis Ramirez, Jenny Saucerman, and Jeremy Dietmeier. 2014. Twitch Plays Pokemon: A Case Study in Big G Games. In *Proc. DiGRA '14*. 1–10.
[7] Marco C. Rozendaal, Bram A. L. Braat, and Stephan A. G. Wensveen. 2010. Exploring Sociality and Engagement in Play Through Game-Control Distribution. *AI & Society* 25, 2 (2010), 193–201.
[8] Philipp Sykownik, Katharina Emmerich, and Maic Masuch. 2017. Exploring Patterns of Shared Control in Digital Multiplayer Games. In *Proc. ACE '17*. 847–867.
[9] Cong Zhang and Jiangchuan Liu. 2015. On Crowdsourced Interactive Live Streaming: A Twitch.tv-Based Measurement Study. In *Proc. NOSSDAV '15*. 55–60.