Crowd Game Design

by
Anne Birgitte Jerichau Clausen (abic)
Simon Stålhandske (slad)

Master Thesis
IT University of Copenhagen
31 May 2016

Abstract

Audience participation games used as advertisement in movie theaters are on the rise in our part of the world.

This thesis outlines the design space of the relatively unexplored type of game called a crowd game. Crowd games are here defined as local multiplayer games for 50 players or more. The thesis has identified several design concepts that are specific to crowd games. This was done by analyzing the crowd games that currently exists and by designing and testing 5 prototypes of new crowd games.

The possibilities offered by the design space in terms of input methods and feedback are explored. This includes an analysis of different types of inputs and how they can be used in crowd games, pinpointing the challenges that designers of crowd games can relate to.

In this thesis, the concept of a game host is introduced. The game host is the person that runs the crowd game, and it is elaborated how this role can be used and expanded in the design of crowd games.

It is examined how crowd games can be tested, concluding that not all aspects of the design needs a crowd setting in order to be tested. This is done with basis in the work with testing crowd game prototypes as part of the thesis.

The project explores how to design for physical and social interactions in crowd games, based on Bernard De Koven and Douglas Wilson's theories.

Acknowledgments

This project had not been possible without the support from the community around local multiplayer games. Thanks to all the developers and designers that shared their knowledge with us during this project. This includes: Benjamin and Bryan from Colossi; the man behind HappyFunTimes, Greggman; Tim Garbos; Alexander Birke; John Sear, co-designer of *Renga*; Andy King from PlayWest; Johannes Kristmann, co-designer of *Happy Hockey*; the man behind Localmultiplayer.com, Lorenzo Pilia; and probably many more. We are grateful to all of you for taking the time and effort to share your knowledge with us.

Thanks to Lena Mech from Copenhagen Game Collective for inviting us to design a game for the Nordic Game Jam Pre-Party. Thanks to Andreas Lagerstedt, who made and performed the music for *Colorave*, and helped us immensely with the design. We love your work and could not have done the game without you! Also a big thanks to our thesis supervisor Martin Pichlmair for guidance and faith in us and this project.

Contents

1 Introduction

> At the concert venue Pumpehuset in Copenhagen, a DJ plays for the crowd, which is assembled in front of the stage. 160 glowsticks bounce from the hands of the players to the rhythm of the music. Above the DJ, a projection of the crowd with bouncing glowsticks is displayed. The image is overlain by a grid of triangles with green and red strokes scattered around it, as illuminated trails following the glowsticks. As the strokes fill up a triangle, it flashed with the corresponding color. "Locked" it says on the screen.

The above description is from a test of the crowd game *Colorave*, a crowd game which was made as part of this project. A crowd game is a digital local multiplayer game for a large number of players. This thesis researches crowd games and their design.

1.1 Background and related work

Figure 1: The BioSpil crowd game *Dolce Word* is one in many new crowd games is movie theaters all over Denmark

In October 2013, the first Danish movie theaters started using crowd games as advertisement alongside the usual commercials that are run before movie screenings. Since then, BioSpil, who run these crowd games, have extended their games into even more movie theaters in Denmark [1]. Using games as advertisement is not limited to Denmark, however. Companies like Audience Entertainment [2] and TimePlay [3] are advertisement companies that fuse commercials and crowd games for the movie theaters. This emergence of crowd games over recent years, is currently dominated by commercial actors. With this thesis, we want to explore the possibilities for crowd games as play experiences, not as a commercial instrument. Local multiplayer games have had a rebirth in recent years, and many new and innovative titles have been released[4]. Inspired by the innovation of local multiplayer games, we see crowd games as being able to be more than just tools for advertisement and branding. Among other things, crowd games can enable strangers to

[1] A blog post from the danish advertisement company Dansk Reklame Film that makes BioSpil, titled "Another record for BioSpil" `http://drf.dk/nyheder/endnu-en-rekord-for-biospil/`

[2] https://www.audienceentertainment.com/

[3] http://timeplay.com/

[4] Examples of the recent releases and renewed interest for local multiplayer games can be seen from the article *Videogames That Bring Families Together* in The Wall Street Journal (`http://www.wsj.com/articles/videogames-that-bring-families-together-1455208649`)

play along side one another, a feat that few other games can achieve. We think crowd games have unspent potential and want to see them gain ground as games in their own right.

In 1991, Loren Carpenter had the audience at the SIGGRAPH congress play a game of *Pong*[5], using colored paddles to control the game. As far as we know, this is the first attempt at making a crowd game. The inventors themselves though, do not use the term crowd games. Instead, they call the system "audience participation technology", and it is now commercialized under the name Cinematrix Interactive Entertainment Systems [6] making both games, voting systems and other forms of interactive systems. In the 90s, other interactive audience participation systems appeared. The StarRider Digital Dome system, a video system for planetariums, implemented buttons in the seats of the audience, to interact with the videos (Digital Media Online, 2016). But it was not until the SIGGRAPH congress in 2004, that another research project in crowd games would appear. Here, the game *Squidball* was presented, as part of the NYU research into motion capture (Bregler et al., 2005). The focus of this research is very much of technical character, focusing on using object tracking as input. Another example on research in the crowd games field can be found in the 2004 article *Techniques for Interactive Audience Participation* (Maynes-Aminzade, Pausch, and Seitz, 2002). This research also focuses on input methods for crowd games, including camera for tracking movement, laser pointers, and a ball casting a shadow. While this article also addresses the issues on how well these inputs engage the players, the focus is still on the technology of the input methods. These early attempts on crowd game research and design used motion capture, props and buttons built into chairs as inputs, and some of those techniques are still in use today. The crowd game *Renga* (WallFour, 2012) uses laser pointers as inputs, and the company *Audience Entertainment*[7] uses camera and motion sensing in many of their crowd games. But since the earlier crowd games, the current landscape has also changed. Smartphones have become widely available, and the new wave of crowd games in movie theaters all uses smartphones as inputs.

Common for this previous research, is that none of it is done from a game design perspective. They all use the term "the audience", not "the players". This current research is different from previous research, by taking a game design perspective. It explores the design space of crowd games with the perspective of games, not focusing solely on technology or on crowd behavior. By analyzing crowd games in the context of other games, including literature on games, this thesis can add a new perspective to this field of research. Since the field of crowd games is relatively unexplored, it is our hope, that this work builds a basis, from which research and design of crowd games can develop further.

1.2 Definition of crowd games

Crowd game is not currently broadly recognized as a label for games of this type that is defined as crowd games. This makes it necessary for us to make a definition of what is meant by crowd games in the context of this thesis. We propose that crowd games are digital local multiplayer games that are intended to be played by many players at the same time. Breaking down this definition, a crowd game is required to:

1. Be a game.
2. Be designed to be played by a large number of players at the same tim.
3. Be designed to be played locally.
4. Incorporate interaction with a digital component.

[5] *Pong* was developed in 1972 by Atari Inc.
[6] http://usa.cinematrix.info/the-system/games/
[7] https://www.audienceentertainment.com/

The first criteria, game, relates to the degree of how much the play is structured. Without going into a longer discussion of the definition of games for this project, it is just relevant to point out that the game should not only be a playful artifact or toy. In *The Well-Played Game*(Sicart, 2014) Miguel Sicart argues that games are merely a form of play, a manifestation of play. He states that games tend to be less free and more formal that toys, making players able to surrender to the game and play. Similar to that understanding is Salen & Zimmerman's, that games are a structured form of play (Salen and Zimmerman, 2004). For this thesis, this understanding of games as a structure within which play can be expressed, is used. When the term game is used in this thesis, it is always used in the meaning of a digital game, unless otherwise noted.

The second criteria is that the game supports a large number of players. This criteria is very vague. How many players does is take to have a large number? Traditionally, local multiplayer games support up to 4 players. The common consoles like the PlayStation, the Xbox and the Wii have a limit of 4 controllers. The local multiplayer games for those consoles therefore – with few exceptions – have a maximum limit less than or equal to 4 players. This tradition has largely carried over to local multiplayer games today. A look at the list of local multiplayer 2014 releases (Localmultiplayer.com, 2014) reveals, that a majority of the local multiplayer games released in 2014 support 1-4 or 1-2 players. Few games support up to 8 or 12 players, while even fewer support more players than that. There is no exact number of players needed for a game to meet the criteria of being large, however. Lorenzo Pilia, the administrator of `localmultiplayer.com`, for example defines crowd games as being local multiplayer games for more than 4 players[8]. For the purposes of this thesis, the limit has been set at 50, meaning that a crowd game is required to support at least 50 players at once. This sets crowd games far apart from the traditional local multiplayer games by separating them by an order of magnitude in the number of players. The hope of this separation is that it will illuminate the design challenges that come from crowd games, which might not be illuminated by local multiplayer games for 8 or 16 players.

The third criteria is that the game must be designed to be played locally. This requires the game to be designed to be played with all the players in the same physical location. In practice, any online multiplayer game can be played locally, but this criteria requires the game to be designed specifically to be played locally, excluding the branch of massively multiplayer online games, that exists.

The fourth criteria is that the game must incorporate interaction with a digital component. The digital component ensures that crowd games separate themselves from folk games. Examples of these can be found at Bernie DeKoven's deepfun.com, which outlines multiple games that are both local and support many players but that are purely physical[9]. Digital interaction means that there must be both input and output from a digital device, and that the output somehow relies on the input. It is for example not enough that the game uses a video played from a DVD such as used by games like *Atmosfear: The Gatekeeper* (Clements and Tanner, 2004).

Furthermore, for the sake of this thesis there is an additional delimitation in what parts of crowd games that are explored. When researching, it quickly became clear that crowd games can be put into one of two categories. There are games like *Tonde Iko* (Tavares, 2014) which are set up as an installation at a conference, museum, or convention, where people can then approach the game and join as players. In *Tonde Iko* and other such installed games, players come and go. It is very dynamic and there is no guarantee that a certain number of players are playing at any given time. On the other hand there are games like *Renga* (WallFour, 2012) where players gather specifically to play the game or are already gathered for some other reason. In this setting, the game is presented to the players at the beginning of the game and the player-base is mostly consistent from start to finish. This thesis chooses to focus on the latter type, the presentation game. In presentation games, there are a bigger chance that the number of players will continue to be high throughout the game. The installation games, on the other hand, has people coming

[8]Lorenzo Pilia states this in a comment on the public Local Multiplayer Facebook group at `https://www.facebook.com/groups/localmultiplayer/permalink/1649508351986376/?comment_id=1649549378648940`

[9]Find Bernie DeKoven's list of physical games for many players at `http://www.deepfun.com/othergames`.

and going and would possibly often be played as local multiplayer games. The delimitation of only looking at presentation games was also chosen because these games are currently more common. By limiting the research to this specific setting, the project can touch upon more facets of this particular area. In continuation hereof, we also chose to confine the research to crowd games that use a large common display, which can be seen by all players. This delimitation is based on the findings that most crowd games use this setup in any case, and that many interesting questions regardign crowd games is aimed specifically at using a common display.

For the purpose of this thesis, a crowd game is defined as a local multiplayer game that supports at least 50 players and is played on a large screen in a presentation setting.

1.2.1 The game host

In playing presentation crowd games, the games must sometimes be presented by a person, before they can be played by the crowd. When the company Audience Entertainment had a crowd of 20,000 players playing *Breakout*, they had a host on stage instructing the players and cheering them on (see Figure 2). We use the term **game host** to describe that person. The game host is the person that initiates the game, but as we discovered during this project, the role of the game host can be expanded from that. A game host can both set up the game, execute the game, calibrate the game, talk to the players like a presenter, and so on. Note that the game host role can also be distributed to more people. All of the people that are not players but who ensure that the game is running properly, from start to end, fall into the game host category. The game host role will be described further in this thesis.

Figure 2: The game host of one of Audience Entertainment's crowd game being played in Lyon, France. In the video from the event, he is seen instruct and encouraging the crowd. From the video *20,000 people playing at Lyon Festival of Lights* (https://vimeo.com/8487908)

1.3 Research questions

The aim for this thesis is to analyze the design space of crowd games in order to explore the following research questions:

What design concepts are specific to the crowd game design space? And how can these design concepts be taken into account when designing crowd games?

Throughout the exploration of the field of crowd games, some specific areas of crowd game design emerged, that seemed to require further investigation. These areas are:

- Interaction, with a focus on inputs and feedback
- Game host
- How crowd games can be tested
- Physical play and the play community

1.4 Outline

The first part of the thesis, Section 2, addresses the methodological questions at hand, going through some of the concerns involved in the design of the research framework of this project. After that, in Section 3, theories on interaction design, which are used in the thesis, are presented and discussed according to their relevance for this project. An overview of the design space of crowd games is presented in Section 4 and Section 5. These two sections first give context to this thesis by describing the crowd games that exist today and thereafter go into detail with the current methods of input available for crowd games. Section 6, prototypes, describes each of the prototypes made for this project. In Section 7, the analysis, the design space is explored further in order to answer the research questions. This is done with the prototypes and the content of the previous sections as a starting point. In this part of the thesis, both existing crowd games, the prototypes, the design and observation journals from the prototypes, local multiplayer games, and relevant theories and interviews are used to analyze the design space. The analysis is divided into 4 sections, each one addressing one area of interest as they are outlined in the research question (see Section 1.3). The final part of the thesis is a conclusion where the main points of the thesis are summarized and related to the methodical approach for the thesis.

2 Methodology

In this section, the approach and methods used in this thesis are presented. In doing this, relevant research approaches and methods within the field of design research and game design research in particular is presented and discussed. When the framework for the research has been clarified, the concrete methods used in the project are presented. This is done in order to clarify why this research project is conducted the way it is and what that means for the outcome of the project.

2.1 Methods for game design research

Since this thesis explores a fairly new and little researched topic, a starting point to consider for the approach is explorative research. When doing explorative research, the goal is to gain an understanding of the context in which the research take place. The contexts could be anything from the theories involved in a given field to the understandings among people being studied. While this might very well be consistent with the goal of this research, "exploring the design space of crowd games" (Section 1.3), the exploratory research approach does not offer any specific methods for researching. It is also a very broad term, encompassing studies done in fields such as natural and social sciences. An approach that on the other hand is specifically designed for researching game design, is experimental game design proposed by Waern and Back in the book *Game Research Methods: An Overview* (Waern and Back, 2015, p. 341). This approach is divided into 3 categories; Controlled design experiments, evocative design experiments, and exploring a game genre. Controlled design experiments is concerned with how specific design features or changes affect the game design. When doing this type of research, one would test a design over and over, changing or tweaking small parts to see how that affects the results of the test. This kind of research requires some knowledge about the context of the researched topics in order for the researchers to know what to test. It is also best suited when the intended outcome of the research is specific requirements (Waern and Back, 2015, pp. 341-344). For this current project, the context and the questions at hand are not specific enough for this type of research. This research could however be applied when researching crowd games beyond the scope of this project, to gain in-dept understanding into specific design challenges. Evocative design experiments focus on gaining insight into what certain designs, for example certain mechanics, evoke in the players. The research is centered around the players and their experiences and emotions. Research is done by iterating on designs, refining them to achieve certain experiences or emotions (Waern and Back, 2015, p. 344-348). While this approach might be suitable for some game design research, it does not fit the goal of this project. It is not suitable for exploring a design space or multiple possible solutions, since it instead focuses on refining. The approach *exploring a game genre* (Waern and Back, 2015, p. 348-349) at first glace looks appealing for this research, after all, the goal is to explore the design space of the crowd game genre. But according to Waern & Back, these kinds of studies tend to be rather comprehensive:

> An ambitious objective for experimental game design is to explore a novel game genre. Although such experiments still can be small and focused, this ambitious goal will sometimes require the development of full-scale and sufficiently complex games, and study them extensively. (Waern and Back, 2015, p. 348)

As they describe it further, they write about these kinds of studies as big scale, expensive and comprehensive. They do not give any guide on how this kind of research could be conducted on a "small and focused" (Waern and Back, 2015, p. 348) scale. What characterizes this approach though, is that it involves doing one specific game design, iterating and refining that. The goal of this project, however, is not to make a focused study, but rather to explore the crowd game genre more widely. It is not enough to focus on what is available within the genre as it is right now, it also requires looking at what potential possibilities crowd games have to offer.

However, Waern & Back emphasize what separates designing a full-scale game as part of experimental game design research from designing a game outside of the context of research. The

separation lies in that "An experimental game needs to emphasize the factors that are interesting from a design research perspective." (Waern and Back, 2015, p. 348). Focusing the game design on factors of interest for the researcher is indeed a valuable approach, which obviously is a factor when doing a research project such as this one.

What all of the mentioned methods lack, however, is a framework for exploring possible designs, that incorporate the designing into the method itself. For this, the approach of constructive design research might add some insights. This is a practice that stresses the importance of construction and design practice as part of research. This approach to research does not limit methods to those of design practice, but also extends to methods from other fields.

2.1.1 Constructive Design Research

The book *Design Research Through Practice: From the Lab, field, and showroom* (Koskinen et al., 2011) is an overview of the field of constructive design research. It is an introduction to the history and development of constructive design research that also gives insight in the methods involved in this way of researching. According to this book, the central method of constructive design research is *doing*:

> When researchers actually construct something, they find problems and discover things that would otherwise go unnoticed. These observations unleash wisdom, countering a typical academic tendency to value thinking and discourse over doing. (Koskinen et al., 2011, p. 2)

With constructive design research, rationalistic research thinking is no longer the main focus. Instead, it is replaced by a combination of engineering, social science, and design, with design as the core of the research (Koskinen et al., 2011, p. 28). It is a way of thinking that borrows from all of the above disciplines but is also influenced from the fields of art and culture, where many designers also function (Koskinen et al., 2011, p. 29). Like with exploratory research, constructive design research is not a discipline of analyzing or providing solutions to problems. But whereas exploratory research is concerned with making a basis for further research, constructive design research also concerns itself with imagining and building future realities (Koskinen et al., 2011, p. 42).

A central method in constructive design research is prototyping. When taking this approach to research, prototyping is not done in order to make a design that could necessarily be realizable. Instead, it is done to make suggestions that could explore where a design field could be moving. By pushing the limits and experimenting with the extremes, it opens up the research field and provokes new ideas and understandings (Koskinen et al., 2011, p. 46). This is also part of the reason why user-centered design alone is not viable for research. It focuses very much on what is now, and does not call for the designer to go beyond what currently exists and re-invent the future (Koskinen et al., 2011, p. 22). This is one of the key reasons for this thesis to take a constructive design research approach. The field of crowd games is a relatively unexplored field. If researching with a traditional qualitative approach, the research would be able to uncover how crowd games are currently designed. With constructive design research, the hope is to also shed light on what possibilities the design space of crowd games has to offer, as mentioned in the research questions (see Section 1.3).

Since imagining possible futures is a goal for constructive design research mass-production and commercialization is not the goal for the designed product. This is beyond the scope of the researcher. Instead, imagining and abstracting from the current setting is sufficient (Koskinen et al., 2011, p. 43). The prototypes made as part of this current research are therefore not made to be marketable, or made as a response to a market demand. Instead, they are designed based on curiosity into the possibilities of game design in the field of crowd games. This way, the research for this thesis is not only grounded in theoretical considerations. Curiosity and the available technologies are guiding the research and the design decisions for the prototypes. This approach

is also what characterizes the many universities and labs that with a sandbox culture, which stresses building and experimentation, lead the way for research that builds on construction:

> They [the institutions] showed that it is possible to do research with things at hand without complex justifications and theoretical grounds and just let imagination loose in the workshop. (Koskinen et al., 2011, p. 24)

When working in this constructive way, researchers make ideas tangible in order to be able to discuss and critique them. In this part of the process, a lot of things are at stake that can not necessarily be described rationally. Instead of rationalizing, these design decisions rely on the designer's experience as a design practitioner:

> In this work, analysis and reasoning are important, but equally important is design experience, whether it is based on emotions, feelings, or intuition. This work may start from theories, methods, and fieldwork findings, and just as often it begins with playing with materials, technology, and design precedents. (Koskinen et al., 2011, p. 43)

As stated above, the starting point of making ideas tangible may have very different starting points. One way to start off is by doing research into users, a method known from user centered design. But the problem with user centered design in regards to constructive design research, is that it is best used within the frames of what currently exists. That way, the focus of designing for the future and creating new designs, fade into the background (Koskinen et al., 2011, p. 22). But methods from user-centered design can be used in constructive design research to provide an understanding of the current context in order to inspire researchers (Koskinen et al., 2011, p. 22). For this thesis, methods like playtesting, observations and interviews are used to give a context for the design space.

2.2 Prototyping

A big part of the design method is prototyping, and this is also at the core of the research of this thesis. Much of the literature that has been written about prototypes and prototyping methods focuses on the different types of prototypes, from lo-fi Wizard of Oz prototyping to working artifacts, or experience prototyping. But, since this thesis is not about traditionally designing a product, service, or system, but rather about gaining an understanding of the whole field of crowd games, this traditional view on prototypes might have its shortcomings. We are not using prototyping as a way of optimizing a design or finding solutions to a specific problem. Rather, we want to explore the possibilities and design problems surrounding the designing of crowd games.

Prototyping often takes an approach of filling requirements, an example on this can be seen in Floyd's *A systematic approach to prototyping* (Floyd, 1984). As Floyd describes prototypes, they are very much viewed as a means to evaluation, in terms of outlining requirements, communicating with the users and evolving the design. She mentions *exploration prototyping* and in her understanding these types of prototypes explore the user and the relationship between the user and the system. This view of prototypes as a means to evaluate, is challenged by other design theorists. In *The Reflective Practitioner* (Schön, 1984), Donald A. Schön emphasizes the dialogic relationship between a designer and a prototype; that a prototype "talks back" to the designer. This same argumentation can also be found in Bill Buxton's book *Sketching User Experience. Getting the design right and the right design* (Buxton, 2007).

In the article *The Anatomy of Prototypes: Prototypes as Filters, Prototypes as Manifestations of Design ideas* (Lim, Stolterman, and Tenenberg, 2008), Lim et al. also argue that looking at prototypes only as a means for evaluation gives an incomplete understanding of prototypes and the activity of prototyping:

> They [the prototypes] are design-thinking enablers deeply embedded and immersed in design practice and not just tools for evaluating or proving successes or failures of design outcomes (Lim, Stolterman, and Tenenberg, 2008, p. 2)

Lim et al. point out that prototypes evoke new ideas and reflections on the design space in the designer, that would otherwise not have been discovered (Lim, Stolterman, and Tenenberg, 2008, p. 2). This can be explained by the concept of "the extended brain", which describes how cognitive processes not only take place as internal cognitive activities, but also extend to artifacts in the world. Prototypes are manifestations of design ideas, that participate in the thought processes of the designer. This is why prototyping can be used not only for developing artifacts and evaluating design goals but also for exploration. This point relates to the idea of construction in constructive design research. When constructing a design, it opens up new possibilities and perspectives.

In this context, imagination is not to be understood as the internal mental task of imagining, the understanding here is much broader. Imagination is here understood as an ongoing process between the different actors within the research. From the designer to the prototype and back again. Related to this is another reason for doing constructive design research. When designers do, they come to realizations regarding material and aesthetic choices, that would have gone unnoticed if the designers were just theorizing (Koskinen et al., 2011, p. 43).

In order to use prototyping as method, a framework for understanding prototypes is needed. In the article *The Anatomy of Prototypes: Prototypes as Filters, Prototypes as Manifestations of Design Ideas* (Lim, Stolterman, and Tenenberg, 2008) such a framework is proposed. The article suggest a framework that describes the anatomy of prototypes as a tool for both analyzing prototypes and for designers to use when prototyping (Lim, Stolterman, and Tenenberg, 2008, p. 18). This framework can be used both to give a language to the prototypes, but also to describe how prototypes can be used not only as a means to design solutions but also as a way of exploring a design space:

> In design and development processes, prototypes are used not for providing solutions but for discovering problems or for exploring new solution directions. (Lim, Stolterman, and Tenenberg, 2008, p. 8)

2.3 Prototype profiles using filtering and manifestation

When describing prototypes, Lim et al. use the concept of a prototype profile that describes what the designers consider when creating the prototype at hand. Using the model of the anatomy of prototypes gives designers a language to communicate about prototypes and the prototype profiles can be used by designers to plan prototyping in advance (Lim, Stolterman, and Tenenberg, 2008, p. 28). The article suggests two types of dimensions that prototypes can be created around. The filtering dimensions are focused on *what* to prototype, while the material dimensions describe *how* to prototype.

The manifestation dimensions are; material, resolution and scope. The material dimension is concerned with the material the prototype is made of. The resolution is concerned with how high or low the fidelity of the prototype is. Scope is concerned with the number of aspects of the design the prototype includes, how complete it is (Lim, Stolterman, and Tenenberg, 2008, p. 15). For this thesis, the designs will mostly be digital, since the field is digital games. One might not at first glance think of digital games as artifacts, mainly because they are not tangible. But digital artifacts have a materiality and aesthetic qualities as well. Visual feedback, controllers, and game feel[10] are all examples of concepts that lie inside the field of crowd game design, and that have material and aesthetic qualities that are relevant to explore in order to research crowd games. When designing for a specific controller, the designers might experience qualities of the controller, that were not apparent when only analyzing how other games uses those controllers.

[10] These concepts will be introduced later in the thesis.

When deciding on the parameters for the manifestation dimensions, the guiding principle is the economic principle of prototyping: "the best prototype is one that, in the simplest and most efficient way, makes the possibilities and limitations of a design idea visible and measurable." (Lim, Stolterman, and Tenenberg, 2008, p. 3). When prototyping, the designer must find the most effective way to make the manifestations of the prototypes. The higher the fidelity of a prototype, the more costly and time consuming it is, so it is important for the designer to only make the prototype just as high fidelity as it needs to be in order to explore the selected aspect of the design.

The filtering dimensions, *what* to prototype, are concerned with which parts of the design that is being prototyped. When designing the prototypes for this project, the focus was to gain insights in the whole spectrum of what it takes to design crowd games. Instead of limiting the prototyping to only filtering some dimensions of the prototypes, the scopes of the prototypes for this project are rater big, filtering for as many dimensions of the prototypes as possible. This is done in order to be open to the questions and insights the prototypes reveal, and not focus the research area too much beforehand. As described above, this is of course a trade-off. The consequence of this approach is, that the prototypes have a rather low fidelity, none of the filtering dimensions get completely finished or polished. That means that the insights from the prototypes will be hard to generalize. What could have been done instead was to choose some areas of specific interest, and make more in-dept analyses of how those aspects could be designed; as one would do in for example experimental game design research. Instead, this thesis aims to provide a basis for further research, that could very well be conducted as experimental game design research.

2.4 Method of this thesis

The research process for this thesis is explorative, exploring the broad field that is the crowd game design space. The first part of the research was done by making an overview of what crowd games that exist. This was done by compiling a list of crowd games with help from the local multiplayer community[11]. The list produced with their help was made in an online Google spreadsheet, which can be publicly accessed at `http://localmultiplayer.com`[12]. Only we have the access to edit the list, but anyone can comment or email us with additions or corrections. It should be noted that the publicly available list includes many games that do not fall under the definition of crowd games used for this thesis. Among other, the list includes games that support a maximum of players lower than 50. Therefore, a separate private list has been kept, which includes only games that fall under the definitions set for this project. This list of crowd games for the thesis was the basis for the further process, and can be found in Section 4 Figure 5. References to the list of crowd games throughout this thesis, refer to that limited list, unless otherwise noted.

Based on the list of crowd games, an overview of inputs for crowd games was collected. The different technical options for those inputs were listed, together with some possibilities for how to use them. Prototypes were then made. Some of the prototypes took starting point in the research that had preceded, others came out of other ideas, but each of them explored something new that the games in our lists did not have. Starting with the prototype *Jump on heads*, different prototypes were made in order to examine the concepts described in the research questions (see Section 1.3). While designing the prototypes, design journals were written to collect our considerations and design decisions. The design journals can be found in the appendix. All prototypes were also tested with players, both to gain insight into their designs but also to research crowd game testing. These observations are collected as observation journals in the appendix. In few cases, observations on others crowd games were included, as they offer an opportunity to compare with a broader array of crowd games.

[11]The local multiplayer community in reference, mainly consists of the members of the Facebook group *Local Multiplayer / Couch / Co-Op Gaming Hangout* found at `https://www.facebook.com/groups/localmultiplayer/`, but also includes followers of the Twitter account @localmulti and visitors of `http://localmultiplayer.com`.

[12]The direct link to the crowd games list is `https://docs.google.com/spreadsheets/d/1zYwGcMiEAt0OstCqmPz8f5OUUWoyCEoQbPRdQVt4nSA`

The thesis also includes 2 interviews with other designers that work with crowd games. One interview is a correspondence interview with game developer Andy King. This interview is included because King has expertise in designing crowd games played with traditional controllers. This is something very few have done and something that was not possible to do for ourselves within the scope of this thesis. The other interview is with the game designers from Colossi, an American company that makes crowd games for large venues, typically stadiums. They have experience designing for big crowds of hundreds of players. The interview is concerned with their experiences on both general crowd game design, testing, and scaling up for a bigger crowd.

The approach for this thesis is explorative in its starting point and goal, researching into a rather unexplored field with the goal of understanding what the design space consists of. But the concrete methods used are based on the idea of constructive design research, constructing prototypes to gain insight into this novel field of research. This research project therefore aims to illuminate the design space of crowd games, focusing on gaining a broad insight and illuminating as many aspects as possible. This mean that most aspects will be touch upon lightly, researched deep enough to develop an understanding of what they consist of, but not a fully, in-depth understanding of their nature. The aim is to make a basis for what to consider when designing crowd games and sparking curiosity for further research and game development.

3 Theories of interaction

One of the areas of interest for this thesis is the interaction (see Section 1.3). In exploring this question, a better understanding of inputs is needed. In this section, the concepts from Donald Norman's 7 design principles from the book *The Design of Everyday Things* (Norman, 2013) are incorporated along side and in relation to the model of interaction from Steve Swink's book *Game Feel* (Swink, 2009). By including Norman and Swink, a language to describe and analyze interaction with crowd games is provided. In this section the theories from each author will be presented, broken down, and put into relation with their relevance to crowd game design. The section does not seek to give a comprehensive theory of interaction, but instead focuses on the parts which have pertinence to this thesis.

3.1 Norman's design principles

Donald Norman's book *The Design of Everyday Things* is a self proclaimed "starter kit of good design" (Norman, 2013, p. xi). It focuses on the design of ordinary objects that people interact with in their everyday lives. In this section, Norman's 7 principles of design is introduced, and discussed in relation to their relevance to games and to crowd games in particular.

In his book, Norman describes a model of action, from setting a goal, to making an action towards that goal, and to finally comparing the outcome of that action with the original goal (Norman, 2013, p. 41). Each step in the model poses a question from the user, which the designer effectively has to answer through her design. This leads Norman to 7 fundamental principles of design. These principles are: discoverability, feedback, conceptual model, affordances, signifiers, mappings, and constraints. Of the 7 principles proposed, only some of them have relevance for the design discussion of this thesis. The relevant principles are feedback, conceptual model, affordances, and mappings. Following is a short introduction to each, how each applies to digital games, and why each has relevance to the design of crowd games specifically.

In a game like *Super Mario Bros.*, the character Mario jumps when the player presses the A button on their controller. This results in a jump animation, along with a jump sound, and a movement of the Mario character, all of which is part of the **feedback** from the game. Much of the work in this thesis explores new types of ways to interact with digital games, that are much different from the way players interact with a game like *Super Mario Bros.* Providing players with clear feedback therefore stands out as one of the main concerns in this undertaking. On feedback, Norman summarizes "There is a full and continuous information about the results of actions and the current state of the product or service. After an action has been executed, it is easy to determine the new state." (Norman, 2013, p. 72). Discussing how these new interaction methods used in crowd games can provide players with continuous information will be central in the analysis of crowd games in this thesis.

In the game *Canabalt*, the player helps a character escape his death by tapping the touchscreen to make him jump over the gaping abyss. The world and story that *Canabalt* constructs around the relatively abstract act of tapping a screen, gives meaning to the action for the player. An essential part of *Canabalt* as with most digital games is that it has a **conceptual model** that takes the abstract mechanics of the game and fits them into a metaphor under which they are suited and that the player understands. Norman summarizes the conceptual model: "The design projects all the information needed to create a good conceptual model of the system, leading to understanding and a feeling of control." (Norman, 2013, p. 72). In games, the conceptual model and the metaphor of the game is often given very explicitly. But digital games also contain other components such as menus and other graphical elements, that are more abstract, that need a different kind of conceptual model to be understood.

An analog stick on the face of a controller allows the player to provide free directional input in two dimensions, which for example can control the movement of a character in a game. This is an **affordance** of the analog stick, which is not shared by an input such as the D-pad, which only provides discrete directional input. Affordances are the ways humans are able to interact with

physical objects. Norman's design principle on affordances states: "The proper affordances exist to make the desired actions possible." (Norman, 2013, p. 72). Since this thesis is exploring new methods of input, one question that arises is what affordances those input devices have.

Most games use a standard **mapping** from the controller to the reaction in the game. In first person shooters, the mouse is used to look around with and the *w*, *a*, *s*, and *d* keys are used for walking or running. This is a standard mapping and has in many ways become a **natural mapping** for this genre of games. A natural mapping is a mapping that the player finds intuitive. A natural mapping is often based on culture, such as the example above, where the natural mapping only applies to the player who is used to playing first person games. Norman writes about mappings: "The relationship between controls and their actions follows the principles of good mapping, enhanced as much as possible through spatial layout and temporal contiguity." (Norman, 2013, p. 72). With the nontraditional input devices that this thesis introduces, it is an interesting point to look at the mapping between those and the response of the game.

Norman's theory is about the interaction with everyday things. Although crowd games are not everyday things, the language that Norman uses, can also be applied to the different aspects of crowd games. Feedback, conceptual model, affordances, and mappings are still important for crowd games because they are things that players interact with and it is relevant to discuss them in effect of the nontraditional ways that players interact with crowd games, as will be seen in this thesis.

In digital games, players are constantly interacting with the game, for example making a character jump or making a car steer to the left. Players constantly provide input to the computer and the computer constantly provides output to the player. In order to be able to discuss this process, it is helpful to have a framework within which to talk about it. Donald Norman falls short in this matter, in that his discussion revolves around the design of non-digital objects, and it is in many ways one step too low-level to be able to meaningfully describe the interaction with digital things. Swink, however, sets up a concrete 7-step model of interaction with digital games in his book *Game Feel*. Swink's model is specifically focused on digital games, which makes it relevant to work with his model within the context of this thesis.

3.2 Swink's interaction model

Figure 3 shows Swinks interaction model (Swink, 2009, p. 62). It consists of the following steps: step 1, the human processor; step 2, muscles; step 3, input device; step 4, the computer; step 5, the game world; step 6, the output device; and step 7, the senses. Arrows show how signals flow from step 1 through all the steps and back to step 1 in a constant cycle.

Seeing as the figure is taken out of the context of Swink's book, it also includes some information that will not be relevant to this thesis. The perceptual field, for example, which is drawn as a dashed line around the edge of the figure, is a model for how prior experiences affect the way humans perceive the world. The arrow labeled "intent" is meant to describe how the player has a certain goal for what her actions are meant to achieve in the game. These parts of the model will not be relevant to our discussion.

In *Game Feel*, Swink analyses the 7 steps of his interaction model in great detail for traditional singleplayer games controlled with traditional controllers or other input devices. Crowd games are, however, significantly different in several parts of the interaction model in Figure 3. Following is an outline of the steps in Swink's model and what is and what is not relevant for crowd game design.

The 7-step model starts and ends with step 1, **the human processor**. The human processor is a model of a human and is in essence just another way of saying *the player*. Swink uses this model to build up a theory of game feel. Most of this theory will not be used in this thesis. But one take away Swink has, is that of **real-time control** (Swink, 2009, p. 35-37). Through the human processor model and tests with games, Swink concludes that feedback needs to be presented to the player within 100 ms in order to optimally retain the feeling of real-time control. Any feedback

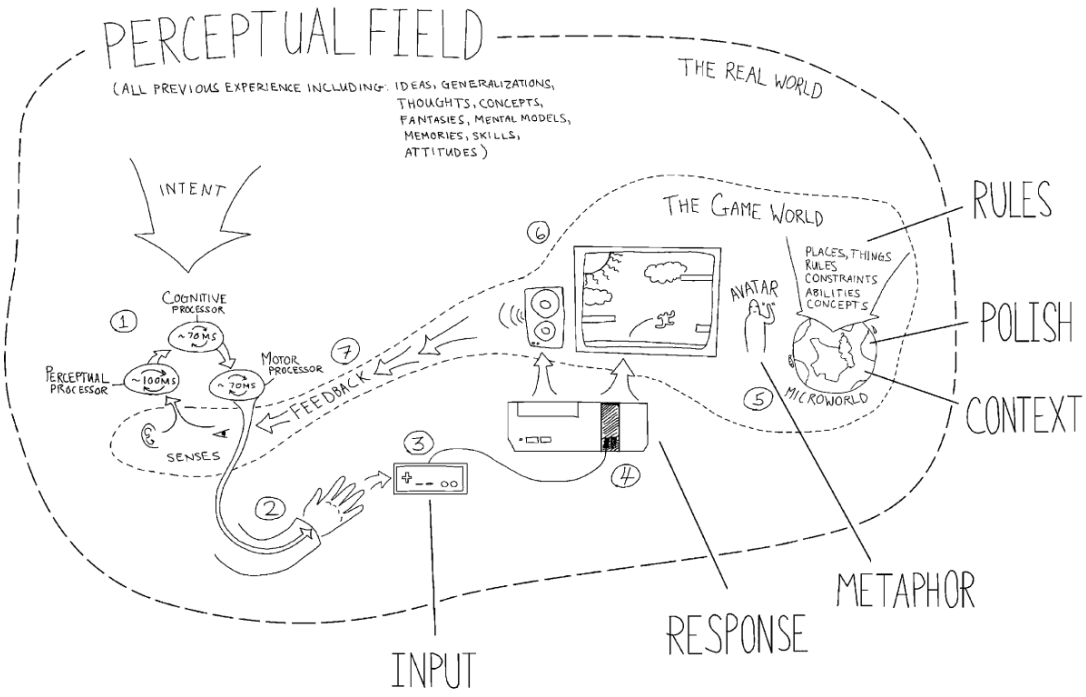

Figure 3: The game interaction model as shown in Swink (Swink, 2009, p. 62). Among other things, the diagram shows the 7 steps of the model; each step is indicated by a circle with a numeral inside. The different steps in the interaction go from the human processor (1), through muscles (2), input device (3), the computer (4), the game world (5), the output device (6), and finally the senses (7). The diagram also shows other pieces of information which are irrelevant to this thesis. These are the perceptual field, intent, and the model for the human processor.

slower than about 150 ms will make the response feel sluggish, whereas feedback withing 100 ms feels tight. Looking at the model of interaction, this means that the signal traveling from step 2, muscle, to step 7, feedback, should take less than 100 ms in order for the player to get the best possible feedback. This will later become relevant when discussing some of the more processor heavy input methods.

Step 2 of Swink's interaction model is called **muscle** and describes how "the impulses from the human processor flow out into the real world." (Swink, 2009, p. 63-65). The player can manifest these impulses in the real world by the push of an analog thumb stick on a controller or through the movement of a mouse over a table. These manifestations do not only work as an output from the player, but in most cases they also provide tactile feedback, like the feel of a button press. **Input device** is step 3 of the interaction model and is "the player's organ of expression to the computer." (Swink, 2009, p. 63-65). Examples of input devices are the traditional video game console controllers, the mouse, and the touchscreen. The job of the input device is to translate the muscle movement of the player into electrical signals. It can be seen as a **filter** between two forms of signal. For crowd games, we will see that other types of input devices than those traditionally used in step 3. These are devices that are not necessarily controlled by the hands of the player.

Step 4 in Swink's model is **the computer** and should be understood as the computer in a broad sense of the term. It can be a smartphone, a video game console or any other digital device that takes input and provides output for the game. In the computer, the signal from the input device is filtered again before it then affects the computer's internal model of **the game world**, step 5. For example, the binary input from the press of a button, becomes a running motion of the character in a platformer. As Swink points out, signals from an input device are rarely used raw and unfiltered when translated into mechanics in the game (Swink, 2009, p. 131-134). Inputs

are usually filtered down to the relevant parameters, then mapped to a response. The press of a trigger on an Xbox 360 controller, which gives a raw input on a continuous scale from 0 to 1, may for example be filtered by being multiplied by a constant or by being broken down into discrete steps. Afterwards it can be effectively mapped to the acceleration of a car or the firing of a gun. This filtering from raw electrical input signals to response in the game world, is also relevant for the further discussion and analysis of interaction methods for crowd games.

As shown in Figure 3, there are a number of aspects like rules, polish, and context that go into a game world. These details are not necessarily as relevant here as they are for Swink. The game world serves two main functions in relation to this thesis. First, it serves as a metaphor for the player to understand the game. Comparing to Norman's design principles, the game world corresponds to what Norman refers to as a conceptual model. The game world — and thereby the conceptual model — exist to give meaning to the player's input and the mechanics of the game. It enhances understanding and coherence between input and feedback. Second, there is a digital representation of the game world in the state of the computer. At any given time the game world exists as abstract data inside the computer. As Swink describes it

> For our purposes, the game world exists primarily in the player's mind. There is also an internal representation of the game world that exists in the computer, one which is more precise and mathematical than the rich, expressive world experienced by the player. (Swink, 2009, p. 64)

The game world is thereby twofold, the representation in the computer and the player's representation.

Step 6 is **the output device**. "The output devices, which may include a monitor, speakers, controller's ruble motors, haptic feedback device and so on, are the player's window into the game world." (Swink, 2009, p. 65). From the computer and the digital representation of the game world, feedback flows to the output devices, such as screens and speakers, which make a representation of the current state of the game world that finally, in step 7, **the senses**, flows back to the player. She gets feedback through her senses as described by Swink: "The eyes, ears, and hands (both tactile and proprioceptive senses) perceive the new, changed state of the game's reality [...]" (Swink, 2009, p. 65). Feedback comes from multiple channels, not only the output devices.

Swink's model of interaction is specifically made for digital games, but is written within a certain context of games. The inputs Swink is concerned with is primarily traditional controllers, keyboard, and mouse. But a common input to use for games now, 7 years after his book was published, is the touchscreen which resides on all smartphones and tablets. In *Game Feel* (Swink, 2009), touch input is included in a separate section in the end, under the headline "The Future of Game Feel" (Swink, 2009, p. 321). His model of interaction is not based on a context where touch inputs are wide spread, and might therefore have its shortcomings when it comes to those. His examples are also from games that are different from crowd games. His examples come from singleplayer games, first-person shooters, action games and more. Common for those types of games is that they are singleplayer, and that they are therefore distantly removed from crowd games in some respects. Still, the model provides a language which to a large degree helps describe the interaction in crowd games. However, this thesis drills into the shortcomings of the model, and discusses the parts of the model that do not apply well to crowd games. Specifically, the input devices used by crowd games and the feedback from crowd games are radically different for crowd games than for traditional singleplayer games.

4 Overview of Crowd Games

This section gives an overview of the different crowd games discovered as a part of the research for this thesis. It will present all of the crowd games that were found and give a more detailed description of those that are repeatedly used as examples throughout the thesis.

As described in Section 2.4, a list of crowd games was devised as part of the research for this thesis. The list itself is shown in Figure 5 and is a table containing information about all the crowd games collected for this thesis. The list is sorted by year, and starts with games made all the way back in 1991. Entries are divided into two types: either single game entries or companies that have made a large number of games. Single game entries specify the year of release, the name of the game, input devices used to play the game, the developer of the game, and a link to a website describing the game or the developer. For company entries, the list includes the year the company started making games, a list with the names of the games made by that company, the input devices used by those games, the name of the company, and a link to the company website[13].

The crowd games list is presented here in order to easily refer to these games throughout the thesis. It also gives an overview of the time-line of crowd game creation and the input devices used historically. Furthermore, this section describes a number of games in detail. These games are chosen because they are used as examples in numerous cases throughout the thesis.

4.1 THE WUUUUUUUUUUUU

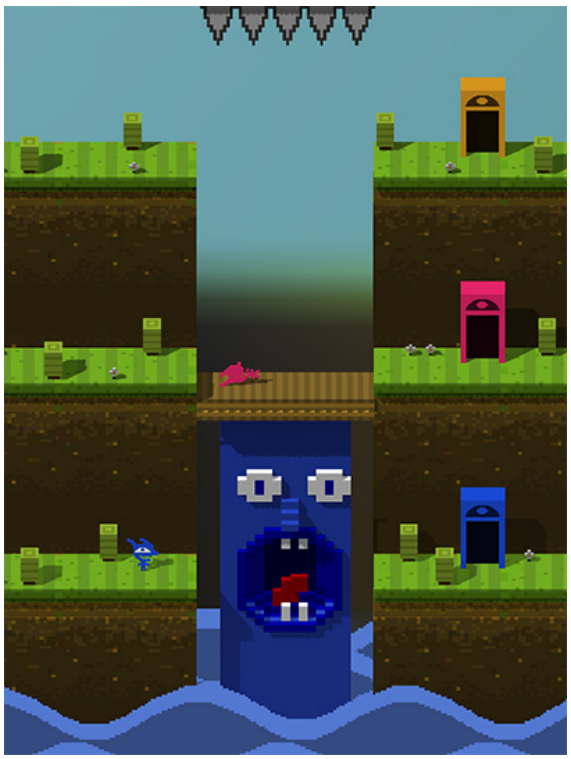

Figure 4: A screenshot from *THE WUUUUUUUUUUUU* which is a game controlled only using a microphone.

THE WUUUUUUUUUUUU (Headspider, 2015), which will be called *THE WUU* from this point,

[13]Many of the links go through `http://www.archive.org` which gives access to a cached version of websites that no longer exist.

Year	Game	Input device	Developer	Website
Since 1991	Seating Chart Exercise Light Sabre Dog and Cat The Flying Game Basketball [... and more]	Colored reflectors & camera	Cinematrix Inc.	http://usa.cinematrix.info/the-system/games/
Since 2004	[Various games]	Smartphone	TimePlay Entertainment Inc.	http://timeplay.com/
2004	Squidball	Inflated balls & camera	NYU Movement Lab	http://manhattanmocap.com/squidball/
Since 2007	Flappy Crowd Fruit Basket Pigskin Pursuit Dodger [... and more]	Camera & microphone	CrowdControlGames	http://crowdcontrolgames.com/
Since 2007	NewsBreaker Live Volvo - Live Gaming [... and more]	Camera & microphone	Audience Entertainment	https://www.audienceentertainment.com
2008	1942 Falldown Frograce Crack on Hook BomberMatch [... and more]	Controller	Projector Games	https://web.archive.org/web/20150320065219/http://projectorgames.net/node/3
2011	Renga	Laser pointer	wallFour	http://dorkshelf.com/2012/10/05/massive-multiplayer-offline-wallfours-cinematic-experiment/
2012	Olympic Crowd-Controlled Game	Camera & microphone	Inition	https://www.inition.co.uk/olympic-crowd-gaming-gadget-show-live/
Since 2013	Tuborg Musik Quiz SB-12 spil LiniCin BioQuiz SAS travel BioSpil [... and more]	Smartphone	BioSpil	http://biospil.dk/
2013	WOOORRK!	Microphone	Tim Garbos	http://kanako.dk/clients/fun/WOOORRK!/
Since 2014	[Various games]	Smartphone	Ibidex	http://www.ibidex.com/
2014	Atlantis	Smartphone	NERDLab	http://claytical.itch.io/atlantis
2014	BoomBoom	Smartphone	Gregg Tavares	https://github.com/greggman/hft-boomboom
2014	Lie Swatter (part of Jack Box Party)	Smartphone	Jackbox Games	http://jackboxgames.com/project/jbpp/
2014	Screamy Bird	Microphone	Tobias Wehrum	http://blog.dragonlab.de/2014/06/screamy-bird/
2014	Seat Leon SC interactive experience Skemmi Shake Experience [... and more]	Camera	Skemmi	http://www.skemmi.com/
2014	Top Gear Mass Audience Game	Camera	Inition	https://www.inition.co.uk/case_study/live-mass-audience-games-top-gear/
Since 2015	Astro Beams Bee Racing Cube Ball Feudal Football [... and more]	Smartphone	Eddie's Social Committee	http://eddiessocial.com/
Since 2015	Quiz Cards Against Humanity AirShields PolyRacer Valhalla [... and more]	Smartphone	AirConsole	http://www.airconsole.com/
Since 2015	[Various games]	Smartphone	Colossi	http://www.thecolossi.co/
2015	Happy Hockey	Smartphone	Alexander Pieper Johannes Kristmann	n/a
2015	How We Make Demos	Smartphone	Jonathan Giroux	https://github.com/Bloutiouf/How-we-make-demos
2015	Stock Market Simulation Game	Smartphone	Christiaan Janssen	https://github.com/ChrisJan00/stock-option
2015	THE WUUUUUUUUUUUUU	Microphone	Headspider	http://headspider.itch.io/the-wuuuuuuuuuuuuu
2015	Touch me Together	Smartphone	Jonathan Giroux	https://github.com/tmp-demo/touch-me-together
2016	FaceClap	Smartphone	Gregg Tavares Wong Chee Hui	https://github.com/greggman/hft-faceclap
TBA	Kachiku 64	Controller	PlayWest	http://playwe.st/games/kachiku-64/
TBA	Sentree	Mediator	Glitchnap	http://www.sentree-game.com/

Figure 5: A list of crowd games that fall under the definition given in this thesis. Sorted by release year.

is a microphone controlled game made at Nordic Game Jam 2015[14]. It can be played by any number of players as long as they can see the screen and can be heard by the microphone. A screenshot from the game is shown in Figure 4. The game is cooperative and the goal of the game is to safely transport the yellow, pink, and blue monsters from the left to the right side by raising and lowering the platform. The platform moves up when the volume of sound is over a certain threshold. The platform moves down if the volume is below that threshold. The game has no end condition, so it is not possible to fully win or lose the game.

4.2 WOOORRK!

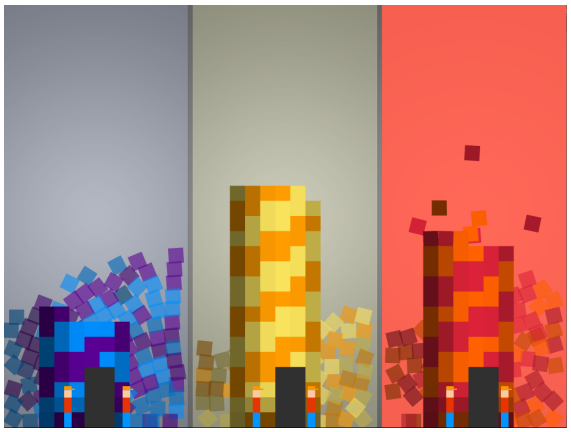

Figure 6: A screenshot from *WOOORRK!*, a competitive game in three teams that is controlled using just a single microphone.

WOOORRK! (Garbos, 2013) is a game for three to any number of players. The players are split into three teams – blue, yellow, and red – and need to make as much sound as possible when their color lights up on the screen, which makes their workers build a tower (see Figure 6)[15]. The louder a team is the faster their workers work. The team to finish their tower the fastest, wins the round.

4.3 Enurendo

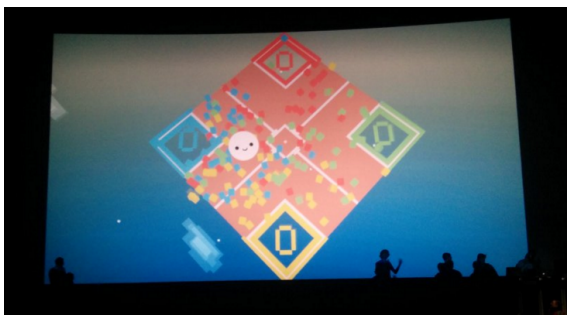

Figure 7: A screenshot from *Enurendo*, a competitive game in 4 teams. Each player can join from the browser of their smartphone, where a virtual controller lets the player control a character that appears on the screen.

[14]THE WUU was in fact the the winner at Nordic Game Jam 2015. The final presentation was recorded and can be watched at `https://youtu.be/yr80d3Th2Ak?t=1928`

[15]Gameplay footage from *WOOORRK!* can be found at `https://vine.co/v/hXhmu1vTU16`

Enurendo[16] (Half Past Yellow and friends, 2016) is a game for 4 players and up. A screenshot of the game can be seen in Figure 7, which shows the only screen there is in the game. Players join through the browser of their smartphone and get assigned a character and a team. The character is controlled with a virtual controller that is displayed on the touchscreen of the player's smartphone. The goal for each team is to score a goal on one of the other teams by pushing a big ball to one of the corners.

4.4 Renga

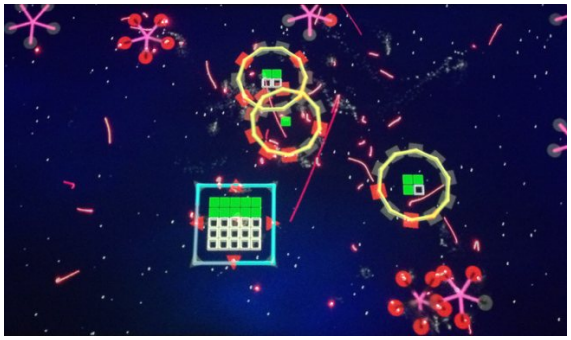

Figure 8: A screenshot from the combat phase of *Renga*.

Renga (WallFour, 2012) is a game for about 100 players, and takes about an hour to play. It is a cooperative strategy game, in which players are given the task of building a base and defending it from enemies. Each player controls the game using a laser point, that they are given before the game begins. The game is split into different phases. In the building phase, players point laser beams at the screen where they want to build in order to expand their base. In the combat phase, players point their laser beams at the approaching enemies in order to kill them and protect their base[17]. These are just two examples of ways the players can interact using the laser pointers. A screenshot from the combat phase is shown in Figure 8.

4.5 Sentree

Figure 9: A screenshot from the game *Sentree*. The goal of the game is to shoot down the enemies that approach from the sides of the screen.

Sentree (Glitchnap, n.d.) is a game played on a smartphone and a tablet which are wirelessly

[16] After the initial version, *Enurendo* has been developed further and now goes by the name *Bash and Block*.

[17] Gameplay footage from *Renga* can be found on YouTube at `https://www.youtube.com/watch?v=8VaxYnMqxcI`

connected[18]. The game is collaborative between one person who has the smartphone and one or more players who are looking at the screen of the tablet (see Figure 9 for a screenshot from the tablet). The player with the smartphone has to turn around and tap the touchscreen to shoot enemies. The players with the tablet can see the game world and have full information about what is going on, while the player with the smartphone is blindfolded. The players with the tablet have to communicate to the blindfolded player how and when to turn, reload and fire the gun.

4.6 Happy Hockey

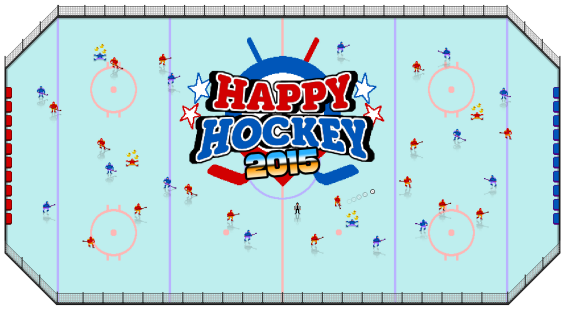

Figure 10: A screenshot from the game *Happy Hockey*. This hockey game is controlled by swiping on the screen of a smartphone. Each player controls a character.

In *Happy Hockey* (Pieper and Kristmann, 2015) players can join through the browser of their smartphone if they are connected to a specific wireless network. Each player gets a hockey player character, which is controlled by swiping in the direction of movement on the touchscreen. The goal is to hit the puck and score on the opposing team[19].

4.7 Colossi games

Figure 11: A screenshot from one of the games by Colossi. This particular game is a racing game, where players have to collaborate to get their character to the end of the level.

Colossi have made a number of different games that are meant to be played by huge crowds. The games all share the same mechanic, which is swiping in order to move characters around in a level. Players connect to the game through the browser on their smartphone and get assigned to one of the two characters on the screen. Each player can then influence the movement of their character by swiping on their screen. The more a player swipes, the more influence they have on

[18]Gameplay footage from *Sentree* can be found on YouTube at `https://www.youtube.com/watch?v=u3SLQdi8Lmo`

[19]A video of the gameplay of *Happy Hockey* can be found at `https://www.youtube.com/watch?v=OjFLA6qAQrk`

the game. The screenshot in Figure 11 shows a racing game, where the two teams compete to get to the goal as fast as possible.

4.8 Summary

The crowd games presented in this section form a basis of examples for the understanding of what a crowd game is. The games in the list in Figure 5 gives the broad overview, while the few games that are described in more detail, give an insight into what a crowd game looks like. The games that are described are chosen because they work as examples in the thesis. The reason why exactly these games are used as examples is either because they separate themselves from the other games in some fashion and thereby answer a question that the other games do not; because they could be researched, because there was information on them online; or because we ourselves have played those games and therefore have more insightful knowledge of the games.

5 Input Methods

This section examines the different types of input methods that are used in the crowd games from the crowd games list (see Figure 5 in Section 4). Each input method is exemplified and analyzed in the light of the theories of Norman and Swink presented in Section 3. For each input method, its former use and some possible future uses as an input for crowd games will be explored and discussed. The input devices included in this section are controllers, microphones, cameras, smartphones, laptops and props.

5.1 Filtering & modes

One way that the input devices are examined, is by looking at which parts of the signal sent from the player to the computer that is actually registered and used, and looking at how this selection happens. A game never uses every single motion, sound, smell, and drop of sweat from the player, but rather **filters** out everything that is not relevant to the game itself. As described in Section 3.2, there are two steps along Swink's model of interaction where inputs are filtered: with the input device, step 3 and in the computer, step 4.

Super Mario Bros. (Nintendo, 1985) serves as a great example of filtering by the input device. The player's input to the game world is constrained by the D-pad and the two buttons on the NES controller, and thereby the complex motions of the player's fingers is filtered to the binary signals that the controller sends to the NES. With any game, as with *Super Mario Bros.*, an amount of filtering is done on the physical side, by the input device. Depending on the input device, the inputs are filtered more or less. Conventional game input devices, such as controllers, only provide very few means of input and filter away everything else.

In contrast to using a controller, which physically filters out most of the input from the player, using a sensor such as a microphone or a camera as an input device for a game, however, collects much more data than is necessarily used in the response of the game world. In the game *THE WUU* for instance, the complex input from the microphone is filtered by the computer to control the vertical movement of a platform. Depending on the way one chooses to filter input in the computer, the microphone and other input devices can have multiple different **modes**. A microphone, for example, has a mode for recognizing pitch, a mode for recognizing volume, a mode for speech recognition, and so on. Modes can be seen as giving the input device different affordances based on how the computer filters the signal. While examining controllers in this section, it will be explored what options there are for filtering the data, leading to an exploration of the different modes of devices, the different ways they can be used as controllers for games.

5.2 Controller

The Xbox 360 controller, the PlayStation DualShock, the built in controls on the Nintendo DS, and the PlayStation Move motion controller are all examples of input devices specifically designed for gaming, and which are heavily used for both singleplayer and local multiplayer games. Controllers are made and designed to be used to play video games with. At the same time, controllers are very generic, only providing few components for interaction such as buttons, accelerometers and analog sticks. They need to be generic, because they need to serve the purpose of being able to control a huge number of games. Still, some conventions rule. Analog sticks, for example, are often used for moving around a character in the game world or for aiming at a target.

Games that are played with controller, of course require that players own or have access to these designated input devices solely for the purpose of playing those games. Attempting to make a crowd game that is controlled by traditional controllers is presents two major challenges. First off, getting access to enough controller to support 50 or more players can be difficult. Second, the hardware and the drivers that controls it are not designed to have a large number of controllers

Figure 12: A traditional Xbox controller. (Source: Wikimedia Commons.)

plugged into a single computer. However, it has been done. The British game studio PlayWest[20] are currently developing the game *Kachiku 64* (PlayWest, 2016), which promotes itself as being a local multiplayer game playable by up to 64 players at once. "*Kachiku 64* is an attempt to break a record for the most number of locally connected players on a single screen! That's right – sixty-four simultanious players all connected to the same screen. No networks, no bluetooth; just good old fasioned physical gamepads!" (PlayWest, 2016). In an interview with the creative director Andy King, he explains how they met the difficulties of making crowd games with controllers,

> Most laptops couldnt get above 8-12 controllers, so we moved from 360 pads (IRQ heavy) to generic cheap 2$ snes pads (v. light - but v. unreliable!) and shifted to sampling raw data from the pads through a wrapper we called 'control 64' which in essense bypassed all of unity and windows low-level. We found one machine that got to 92 connected devices (Appendix M)

Here King explains how they circumvented the difficulties of getting hold of a large number of controllers by buying very cheap hardware. A search on Ebay shows that low quality controllers can be bought for about $4 including international shipping[21]. According to King, the cheap controllers also have a technical benefit over the more expensive Xbox 360 controllers. Although they are very unreliable, they are also very simple, so it was possible for King and his team to write custom software in order to filter the raw input from the cheap controllers. Surprisingly, it is possible to connect 64, and even 92 controllers to the same machine.

However, in spite of the achievement, King and his team still decided to go another way, "We've cheated recently and moved to a pad-splitting method a-la micro machines. Not because it was not possible, but that PC USB standards are so poor" (Appendix M). The "pad-splitting" King refers to here, is a technique for multiplayer which is used in the local multiplayer game *Micro Machines 2: Turbo Tournament* (Supersonic Software and Codemasters, 1994). It lets two players share one controller, and thereby let a total of 8 players play the game together. This means that Kachiku 64 only needs 32 controllers to let 64 people play, easing up the hardware difficulties.

PlayWest found a way to use many controllers for their game *Kachiku 64*, but had to deal with many difficulties in doing so. These traditional controllers are not designed to be connected to a computer in dozens, thereby the difficulties in doing so.

[20] PlayWest is a indie studio that collaborates with University of West England Bristol

[21] This was checked at `http://www.ebay.com/sch/i.html?_from=R40&_sacat=0&LH_BIN=1&_nkw=nes+usb&_sop=15`, visited 2016/05/01.

5.3 Microphone

Figure 13: A computer compatible microphone. (Source: Wikimedia Commons.)

In one of the many levels in the game *Progress* (Ludosity, 2015), the player must speak or sing or hum loudly into the microphone in order to solve the puzzle. The game secretly records the action and later plays the recording back, to the surprise of the player. The microphone is quite a simple sensor that converts acoustic into electrical signals. Through a analog-to-digital converter, the analog electrical signal from the microphone is converted to a digital signal and fed to the computer. As opposed to the controller, which only infrequently changes its state, a microphone changes state continuously and the analog-to-digital converter sends tens of thousands of values a second to the computer. Unfiltered, the signal from the microphone can be recorded and played back to the player such as it is done in *Progress*, but there are also other modes available for the microphone.

The simplest mode of the microphone is possibly that of using the **amplitude** of the audio signal. The crowd games *THE WUU* (Headspider, 2015) and *WOOORRK!* (Garbos, 2013) already make good use of this mode in order to get simple input from entire crowds of players. Getting the amplitude of an audio signal is a process in multiple steps. Since sound is a wave, the signal constantly fluctuates over time. To get the amplitude of the signal, one must therefore average the signal over time, giving the root mean square amplitude. Converting this amplitude into decibels gives a better scale, which more closely correspond to the volume that the human ear perceives. Therefore, using the decibel value of the amplitude for responses in the game world, gives the players the largest experience of coherence between their input and the feedback.

The game *Singstar* (London Studio, 2014) uses microphones to detect the **pitches** of the players and match it to the pitches of the songs they are singing. Using the fast Fourier transform algorithm, it is possible to convert an audio input from a microphone into its representation in the frequency domain. This outputs the amplitude of the signal for each frequency in a specified range. In other words, it is possible to get the pitch or rather the amplitudes of the different pitches in the audio.

Finally, more complex algorithms can be used to filter the signal from a microphone to obtain modes that recognize inputs such as speech, percussion, or the tempo of music. Maybe these will be used to control games in the future.

5.4 Camera

Figure 14: A webcam used to stream video data to the computer. (Source: Wikimedia Commons.)

In *That Selfie Game* (Triband, n.d.), the players use the camera of a smartphone to take photographs of themselves posing as other people. The camera in the smartphone is used as a camera in the traditional sense of the device. It records a picture and later presents it, incorporating the act of photography into the gameplay itself.

A different approach, however, is to regard the camera as a controller for a game. A camera, such as a webcam connected to a computer, provides a discrete stream of 2D images often used for broadcasting or recording and playback. As opposed to the high frequency of data sent by the microphone, most cameras only send a new image about 30 times a second. Each image consists of an array of pixels, each with a color. The color data is most often given as an RGB value, corresponding to the red, green, and blue lights used to reproduce all colors on a display.

There are many different types of cameras available to use. A standard webcam is already compatible with sending data to the computer. Cameras of a similar quality are already built into most modern smartphones, tablets, and laptops. A DSLR camera can provide a high quality video stream, but is more difficult to connect to a computer. The Kinect sensor from Microsoft is built as a controller, supposed to be used for games. It also comes with an API that can help the developer both filter the input from the Kinect and control the mechanics of the hardware. The choice of camera depends on the use case.

Since the camera provides a large amount of data, it can be filtered in many different ways, making the range of possible modes for the camera as a controller rather broad. Following are a number of suggested modes for using a camera as an input device for a crowd game.

In the game *Progress*, one puzzle has the player use the built in camera of their device to find a color in their surroundings. As mentioned above, the images received from the camera are given in arrays of RGB values. This means that the input has already been filtered to provide a spatial layout of **colors**. Especially detecting red, green and blue colors is easy, but it is also possible to detect other colors by converting the RGB values to HSV or HSL color space. Apart from giving a measurement of color (referred to as hue), these color spaces can also be used to detect **saturation** and **intensity** of the light.

By comparing sequential frames, it is possible to detect **change** over time. Instead of viewing each image in isolation, the changes can be used to detect motion in the picture, visual changes of the

scene, or motion of the camera. A use case would be to detect total motion of players and use that in a game, similar to how sound is used in the game *WOOORRK!*. It is also possible to compare images that are far from each other in time. For example, in the article *Techniques for Interactive Audience Participation* Maynes-Aminzade et al. use images of a crowd leaning to one side and the other to train a computer to recognize these movements of the crowd (Maynes-Aminzade, Pausch, and Seitz, 2002).

A more complex mode is **object tracking** or motion capture, which lets objects be tracked by the video captured from the camera. The Microsoft Kinect is a device made for tracking the motion of people (Microsoft, 2016). Using the Kinect SDK, it is possible to track each body part of the player, which is used by many games for the Xbox systems. Maynes-Aminzade et al. propose how to track laser pointer beams and the shadow of a beach ball (Maynes-Aminzade, Pausch, and Seitz, 2002). Whether one wants to track people or objects, methods exist to provide this sort of data. With object tracking, one is able to identify and track an object in real-time using a regular computer webcam.

The company Skemmi made a game for Seat which is documented in a video online[22]. As shown in the video, the image of the cinema is divided into zones. On the screen a visualization of each zone shows how much movement there is in that zone. Because the pixels of an image have a positional relation to each other, it makes sense to make this **division** the image. The Skemmi games takes advantage of this by grouping pixels that are close together. This mode of dividing the image stream can be combined with other modes, by for example getting the total movement or total intensity of light in each zone.

5.5 Smartphone

Figure 15: A smartphone. (Source: Joydeep / Wikimedia Commons / CC-BY-SA-3.0)

In Denmark the advertisement games by BioSpil are played in many cinemas as part of the commercials before the movies are screened[23]. Players connect over the internet through an app on their smartphone, which is then used as the controller. In Canada, USA, UK, Ireland, and Russia, cinema visitors are able to play TimePlay games on the big screen, also by using their smartphones[24]. On top of that, game concepts such as ESC[25] and The Colossi[26] tour events with their smartphone controlled games. For game developers there are also multiple frameworks

[22]Find Skemmi's video of the Seat game at `https://vimeo.com/68690103`
[23]BioSpil can be found at `http://biospil.com/`
[24]See `http://timeplay.com/`
[25]See `http://eddiessocial.com/`
[26]See `http://www.thecolossi.co/`

to develop with, counting HappyFunTimes[27], AirConsole[28], and Ibidex[29]. The possibilities are plentiful for these types of games. However, the smartphone is not designed to be a controller such as the traditional video game input devices are, and such is as well the case for many of the other types of inputs used for crowd games. In the following section, a range of possible input devices that are available to use for crowd games, will be explored. It will be outlined what their properties are and what affordances they offer as input devices for crowd games. This will be done by looking at input technologies and then how they are currently used in traditional games and crowd games alike.

As outlined above, smartphones are the most commonplace device used for crowd games. The smartphones can connect to a server either locally, over WiFi, or remotely over the internet. The smartphone can then send and receive input and output from the game.

The most common operating systems for smartphones are iOS and Android as stated in an analysis of the smartphone market share (IDC, 2016). According to IDC, in 2015, 82.8% of all smartphones ran Android, while 13.9% ran iOS. Both iOS and Android have a myriad of different versions and are installed on many different pieces of hardware. Therefore, not all of the possibilities mentioned in this section apply to all software and hardware out there.

However, a few things are consistent across all devices, the touchscreen being one of them. The touchscreen is the most commonly used interaction method for games on a smartphone. Interaction with games through a touchscreen can take on various forms as for example tapping, double tapping, swiping in a certain direction, or swiping a complex gesture. In the game *Downwell* (Futomo, 2015) the player presses certain areas of the screen that look like buttons in order to get their character to move left, move right, and jump. It is a commonly used input for smartphone games, where virtual buttons are placed on the display. The virtual buttons emulate real buttons by being surfaces that react to being pressed. An example is the crowd game *Enurendo*, which uses virtual buttons for the player to control their character up on a big common display. *868-HACK* (Brough, 2014) is another smartphone game. But this one uses swiping instead of tapping to move the character around the screen. The player can swipe anywhere on the screen in order to move the character either up, down, left, or right. Similarly, in the crowd game *Happy Hockey* (Pieper and Kristmann, 2015), players swipe on their smartphones to control the movement of their hockey player on a large screen where the game world is displayed.

Smartphones also have other input devices, that are less commonly used in games. Among them are the accelerometer, the gyroscope, the camera(s), the GPS, and the microphone[30]. Depending on how and which smartphone is used, more or less of the inputs are available. If the smartphone application is run in a browser, touch inputs are always available, but many other input methods are not, and some are only available on certain devices. Browsers are also becoming more strict as to the security requirements for being able to use many of the smartphone inputs[31]. As of now, Android has access to both camera, microphone, accelerometer, and gyroscope from the browser, while iOS only has access to accelerometer and gyroscope.

Another issue with access to these inputs of smartphones is security. As summarized in a Facebook update by the developer of HappyFunTimes, applications run in the browser of a smartphone are limited in their access, if not using a secure HTTP connection.

> Google, Mozilla, Apple, Microsoft, are all in the process of banning several features from web pages that are served from HTTP instead of HTTPS (secure http). Those include going fullscreen (which HFT [HappyFunTimes] uses on Android if possible) as well as reading device orientation and motion (used on some games). Also reading

[27] See `http://docs.happyfuntimes.net/docs/`

[28] See `https://www.airconsole.com/`

[29] See `http://www.ibidex.com/`

[30] The article *A Study of Mobile Sensing Using Smartphones* (Liu, 2013) gives an outline of the different sensors in a smartphone.

[31] The site caniuse.com lists which inputs of the smartphone are supported by each browser and operating system. Camera and microphone at `http://caniuse.com/#feat=stream`. Accelerometer and gyroscope at `http://caniuse.com/#feat=deviceorientation`.

> the camera or the mic which a couple of games have used on Android. Responding to touch input still works fine. (Tavares, 2016c)

As stated by Tavares, smartphone browser applications have access to touch, orientation, motion, and in some cases microphone and camera, but these features are only available through HTTPS, making it more difficult for developers to use these features[32].

Developing for a browser, there are at least two frameworks available to develop with: HappyFunTimes and AirConsole. In a Facebook comment by Gregg Tavares, he outlines his views of the advantages and disadvantages of each platform. A couple of the more objective points are given here.

> * Airconsole, at least for iOS 9, data from players goes all the way to Airconsole's servers then back to the game. Advantage, players don't have to be on the same network as the game. Disadvantage, longer latency. [...]
> * Airconsole runs on the internet - players must be able to connect to airconsole.com. Happyfuntimes doesn't need internet - in installation mode players can connect without internet access. (Tavares, 2016b)

As Tavares states, the main difference between the two frameworks is that AirConsole's servers are online, whereas with HappyFunTimes the server is local on the computer where the game is running. This gives a longer input latency for AirConsole, and means that all smartphones need to be connected to the internet. However, they do not need to be connected to the same WiFi. A disadvantage of using local WiFi is that wireless routers always are limited in the number of connected devices they can handle. In the HappyFunTimes documentation, Tavares outlines his experiences with connecting multiple devices to the same WiFi and has had the best success with an Apple Airport Extreme. "One time I had 92 people playing *Bombbomb*. I'm not sure if that was a bug or real since the specs say 50 people." (Tavares, 2016a). Tavares had the experience that he could have more smartphones connected than the specifications of the router specified, and goes on to write that he is frustrated by the confusion between specs and practice, and that he doesn't know the cheapest or best way to be able to connect as many smartphones as possible.

The alternative to sending inputs through a browser, is to make a native application. Native applications give access to most or all of the input methods of the smartphone. A singleplayer game that makes use of many of the input devices is *Progress*, which uses both touchscreen, gyroscope, accelerometer, camera, microphone, and physical buttons. Using an application, however, means that players must install the application on their smartphones. Even distributing the application to players can be problematic, because many smartphones only allow applications to be installed from certain sources. Getting an application posted on the app store for iOS for example, can take multiple weeks.

5.6 Laptop

The laptop, or notebook, is a portable computer, which most often has a keyboard, a touchpad, a webcam, and a microphone built in. Laptops, as opposed to desktop computers, have the advantage of being portable, which means that many people might be in possession of a laptop in contexts where crowd games could be played. That could for example be at a lecture in a university, at a conference, or in work related environments.

Laptops, like smartphones, can be networked together in order to play games locally. To our knowledge, there are currently no crowd games that use laptops as input devices. There are however local multiplayer games that do. The cooperative game *Artemis Spaceship Bridge Simulator* (Incandescent Workshop LLC, 2013) uses one computer as the server, which is also connected to

[32] A longer discussion of the difficulties of developing local networking applications using HTTPS can be found on Gregg Taveres' blog at `http://games.greggman.com/game/cas-now-get-to-decide-whos-on-the-internet/`.

Figure 16: A laptop. (Source: Lenovo / Wikimedia Commons)

a projector or large screen. Players then join through other computers to play the game together. Each computer has to have the game installed, which leads to the next point. The main draw of the laptop as an input device is the possibility to easily install new software on it. In contrast to smartphones, installing new software on laptops is quite easy, meaning that each laptop that is used as input device, can have software tailored for that specific game, instead of using a browser. This also means that the software can have access to all of the built in input devices of the laptops.

Some games on the Localmultiplayer.com list of crowd games (Clausen and Stålhandske, 2016) such as *Bombe à Rebours*[33] and *Elbow Room*[34] uses keyboard as input, and would therefore often be played on a laptop. In these games, the players each control a key on the keyboard. The player limit is therefore set to the number of keys that the keyboard has or to the number of players that can reach the keyboard at the same time[35]. Games like these are not included in the crowd games list for this project, since the reality is, that it is close to impossible to gather 50 players around one keyboard. Instead, a scenario could be to make use of keyboard sharing by combining it with networking. Considering that not every person in the crowd might have a laptop available, and that not every player could join at one laptop, one could allow multiple players to join the crowd game from a single machine.

5.7 Props

In the game *Renga*, players are each given a laser pointer to use to control the game. The laser pointer is not the direct input to the game but rather acts as a prop between the player and the input device. Looking at Swink's model for interaction, props are inserted into the model between step 2, muscles and step 3, the input device. In *Renga*, the player's muscles control the laser pointer, which then in turn is detected by the input device, which is a camera. The use of props in crowd games is very diverse, since it covers all uses of objects which are used between step 2 and 3 in Swink's model, be it a huge inflated ball or a laser pointer. This section outlines the few examples that were uncovered from the research done.

Cinematrix have made a version of the game *Pong* to be played by a crowd. In all of their interactive experiences, Cinematrix use a camera to track small reflectors given to each participant. Their system then detects which side of each reflector that is turned towards the camera, and uses that as input to the game world. One side of the reflector is green, while the other is red. In

[33]Find at *Bome à Rebours* at `https://dunin.itch.io/bombe-rebours`

[34]Find *Elbow Room* at `http://www.deepdarkhole.com/elbowroom/`

[35]The number of keys on the keyboard that can be pushed at the same time also depends on the keyboards rollover. On a keyboard with n-key rollover, it is possible to have one player pr. key.

their clone of *Pong*, the ratio of red to green reflectors decides how high the paddle (the one in the game) goes up[36].

Squidball is a game by NYU Movement Lab, which uses an infrared camera and two infrared lights in order to track a large ball in physical space. The audience interacts with multiple balls in order to control the game, which is shown on a large screen[37].

The company Audience Entertainment make games that use the sideways motion of the crowd in order to move a character on the screen from side to side. As an advertisement campaign for the company Orange Mobile, Audience Entertainment made a clone of *Breakout* controlled by 20,000 people holding glowsticks[38].

These are just examples of how props are used in crowd games. One could imagine many other ways that props can be used in the future of crowd gaming.

5.8 Summary

This section has provided an overview of some of the different input devices available for use with crowd games. Following is a summary of the findings for each input device.

Controller Controllers are difficult to use for crowd games. This is partially because of the difficulties in obtaining enough controllers for that many players. Second, it is because controllers and the computers they are plugged into aren't designed such that one computer can have a large number of controllers connected at once. PlayWest has found a way to overcome these challenges by investing in extremely cheap USB controllers, writing custom software to handle these controllers, and letting players share controllers two and two. This proves that it is possible to make crowd games for controllers.

Microphone The microphone is not designed as an input device to be used for games, but the signal from a microphone can be filtered in the computer in order to obtain modes that make sense to use in games. This section outlined two different modes for filtering audio input.

- The first mode measures the total **amplitude** of the sound. This gives a single number that changes over time.
- The second mode measures the **pitch** of the sound. This mode gives a spectrum of amplitudes distributed by frequency.

Camera The camera is another input device, that does not have a tradition of being used in games. Through filtering, the camera has several different modes that can be used separately or in combination.

- For each pixel of the image, **color**, **saturation**, and **intensity** can be measured.
- **Change** can be detected by comparing different images from the camera to each other.
- Complex algorithms can be used in order to **track objects** or people.
- The image can be **divided**, to have the above modes measured separately for different parts of the video stream.

[36] A video of Cinematrix's crowd game version of *Pong* being played can be found on YouTube at `https://www.youtube.com/watch?v=-9eVz4wBBgU`.

[37] A video of *Squidball* being played can be found on YouTube at `https://www.youtube.com/watch?v=kN9mbaR_fgk`

[38] A video of the happening can be found at `https://vimeo.com/8487908`

Smartphone Because of high availability and the possibility for wireless network connection, smartphones are the most popular inputs for crowd games. The smartphone has a number of inputs that can be used but the availability of each input depends on the device, its operating system, and the application which makes the connection to the game.

- The **touchscreen** is always available and can be used by the browser as well as by native apps on the smartphone.
- **Accelerometer** and **gyroscope** data are available through both browser and native apps, but is limited to HTTPS connections.
- **Microphone** and **camera** inputs are available in native apps. Through a browser they are only available for Android at this time and they require an HTTPS connection.

Seeing that the connection from a smartphone is wireless, inputs from the smartphone will have a higher degree of latency than wired controllers.

Laptop As of yet, no crowd games exist that use laptops as input devices, even though they are readily available in some contexts. Most laptops have three input devices, a keyboard, a touchpad, and a webcam, which are to different extents used as input devices for games. Local multiplayer games are sometimes supported just with the built in inputs of the laptop. The possibility for exploring the use of laptops as input devices for crowd games therefore exists.

Prop Introducing props as a coupling between the players and the input device opens up a well of new possibilities for interacting with games. This section has given some examples such as *Pong* played with physical reflectors and the game *Squidball* controlled by a huge inflated ball. These games show how props fit into a slot in Swink's interaction model between step 2, the muscles and step 3, the input device.

6 Prototypes

As outlined in Section 2.4 a core method of this research is the construction of prototypes. In doing this research, a number of prototypes were constructed in order to explore the design space of crowd games. This section outlines the prototypes made as part of this research. The prototypes are listed in chronological order according to when the design of them were initiated. With each of the prototypes follows a description and a reference to the relevant design and observation journals that can be found in the appendix.

6.1 Jump on Heads

Figure 17: A screenshot from a test of *Jump on Heads*. The characters on the right side are stacking in order to reach the cake on the platform.

The prototype *Jump on Heads* is a 2D platformer game built on the HappyFunTimes framework, which lets players join from the browser on their smartphone after connecting to a specific wireless network. Players are then presented with virtual controllers on the touchscreens of their smartphones, which let them control their characters in the game. Players are split into two teams, and the goal for each team is to reach a cake on a platform high above ground. The only way to get to the platform is for the players' avatars to stack on top of each other in order to get up high enough to reach. Once a character reaches the cake, their team wins and a new round begins.

The design journals of *Jump On Heads* can be found in Appendix A.1 and the observation journals can be found in Appendix A.2 and Appendix F.3.

6.2 Put a Smile On

Put a Smile On is a crowd game that uses a camera as input for the game. Players in a crowd access a website from their smartphones, which gives them an image of a smile on their phone. The smiles can have three different colors, and players are divided into teams depending on the color of smile they get on their phone. The players are instructed to hold their smartphones in front of their faces such that the image covers their mouths. The game itself is similar to the physical folk game *Freeze Dance*. The players need to dance while the music is playing, and freeze when the music stops. A camera detects movement of the mouths and adds points to teams that dance while the music is playing, and subtracts point from teams that are moving while the music is stopped. The first team to reach a certain number of points wins the game.

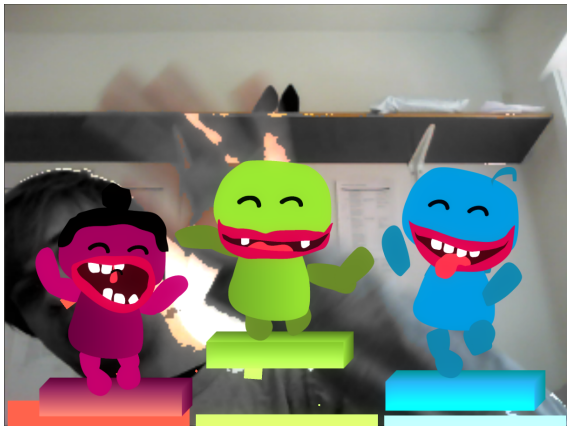

Figure 18: A screenshot from an internal test of the game *Put a Smile On*. In the background is the image from the webcam. Bright parts of the image indicate movement of either red, green, or blue color. In the foreground are three avatars, each representing a team.

The design journals of *Put a Smile On* can be found in Appendix B.1 and the observation journals can be found in Appendix B.2.

6.3 BlowIT

Figure 19: A screenshot from a technical test of the prototype *BlowIT*. In the background is a video feed from the camera. All the brightly colored parts of the image indicate movement. In the foreground, the 4 teams each have an avatar at the bottom of their section of the screen. A ball floats along, through the 4 sections, and any movement triggers an updraft which counteracts the gravity on the ball.

The prototype *BlowIT* is another game that uses the camera as input device. The video stream from the camera is shown on the screen and is divided into 4 vertical columns. Each column represents a team and each player within a column is part of that team. In the game, a 2D ball floats on the screen and movement from players will create an updraft in their column of the screen. The ball continually drifts around on the screen, and it is the goal for the teams not to have the ball hit the top or the bottom of the screen within their vertical column. This causes a loss of one life for that team. Each team has three lives and the game ends once there is only one team left standing.

The design journals of *BlowIT* can be found in Appendix D.1 and the observation journals can be found in Appendix D.2 and Appendix F.3.

6.4 S.T.A.C.K.

Figure 20: A screenshot from a test of *S.T.A.C.K.* Players are divided into 4 teams and need to follow the instructions on the screen. The named blocks floating at the top of the screen are the players' characters, while the 4 rectangles at the bottom are the scores of each team.

S.T.A.C.K. is a crowd game version of the local multiplayer game *B.U.T.T.O.N.* (KnapNok Games, 2011). Each player joins through the browser in their smartphone after connecting to a specific wireless network. A single virtual button appears on the touchscreen. This is the player's only input to the game. On the common display, players are instructed to put down their smartphone and take a number of steps to the side. Then players are instructed to perform some playful task. This could be "Do the boogie woogie", "Epic lightsaber battle, NOW! May the force be with you", or "Poke everybody around you". Finally players are instructed to push their button a certain number of times, hold the button for a certain number of seconds, or refrain from holding or pushing their button. Players win by pushing or holding their button as instructed, but lose if they fail to do so. Players are split into 4 different teams, and the team with the highest percentage of winners, wins the round overall. After this, a new round begins, and the process repeats itself.

The design journals of *S.T.A.C.K* can be found in Appendix F.1 and the observation journals can be found in Appendix F.7.

6.5 Colorave

Colorave is a mashup of a live music performance and a crowd game. The game features glowsticks for all players, a large screen or projected display, a camera, a DJ, and an MC. The camera records the players from above and project the image on the screen, so each player can see their glowstick on the screen. There are two teams, a red team with red glowsticks and a green team with green glowsticks. The goal for each player is to color as large an area of the screen as possible. The screen is split into fields and if a whole field is colored, it is locked and the team who colored it gets a point. At the end of each round, the total score for each team is counted and a winning team is found. Each round, the game has an added rule that the players have to follow. Some rounds ask players to put the glowstick in their mouths, others that players dance like robots or go down low. Figure 21 shows players playing the game[39].

The design journals of *Colorave* can be found in Appendix G.1 and Appendix G.4 and the observation journals of *Colorave* can be found in Appendix G.3 and Appendix G.5.

6.6 Overview of the prototypes

Since one of the areas explored by this thesis is how to test crowd games, each prototype was developed to the stage where it could be tested in a larger playtest setting than self-testing. In

[39]For a video of *Colorave* and more information, visit `https://simonstalhandske.itch.io/colorave`.

Figure 21: A picture from *Colorave* being played at the concert venue Pumpehuset in Copenhagen. In the foreground are all the players with their glowsticks. On the screen it can be seen how players are coloring and capturing the fields.

few cases, the prototypes were developed further after being tested on a large scale. This is the case for the prototype *Colorave* and to some extend also *BlowIT*. With *Colorave* we got the opportunity to test the game with a full scale crowd, in a setting that was not perceived as a test situation for the players. Other prototypes were developed in a very short period of time, tested on playtesters once or twice and not iterated further. This is the case for *Jump On Heads*, *Put A Smile On*, and *S.T.A.C.K.*.

The prototypes made for this project vary in the inputs they uses, from smartphones as controllers in *Jump On Heads* and *S.T.A.C.K* to camera inputs in *Put a Smile On*, *BlowIT* and *Colorave*. The prototypes do not cover all of of the different input methods. Instead, the focus was to cover both one of the main inputs used in current crowd games, smartphones, and to explore one of the less used, the camera. The reason for choosing to focus on camera, was that using a camera as input also makes it possible to design games for physical interaction and props, both of which are little explored in crowd games. The choice could also have been on another input such as microphone. But for games with this input we had some quite good examples to explore first hand. An example of this is *THE WUU* (Headspider, 2015) that we had the opportunity to present and play with a crowd at the AMAZE festival in Berlin, April 2016. Instead, we focused on getting one technology to work, and then explore the opportunities of that particular technology.

7 Analysis

In this part of the thesis all of the findings from this project are laid out. It is described and discussed what challenges the design of crowd games involve, from how to use the different kinds of inputs to how testing of crowd games can be done. Furthermore, suggestions are made on what design decisions in crowd game design can be based upon. All of this is done on the basis of the research of existing crowd games (as listed in Figure 5 in Section 4), the exploration of input methods (see Section 5), and the designed prototypes (see Section 6).

7.1 Interaction

Figure 22: A screenshot from *TowerFall*. Players share the entire screen, but have a character each.

This section presents the design challenges connected to the nontraditional input devices used by crowd games. With basis in the design and testing of prototypes made for this thesis, along with the research done on other games, the design challenges are treated and analyzed. The theories of Norman and Swink described in Section 3 are used as a framework to describe and discuss the challenges. The goal of this section is not to present specific solutions to any of the challenges. Rather the goal is to explore how the challenges are handled by the games that exist, and compare that to the observations we have from our design and testing.

In most traditional local multiplayer games, each player controls a different part of the game world. Often, each player controls a character. This is also true for some crowd games, such as the game *Enurendo* (Half Past Yellow and friends, 2016), where each player controls a separate character. The opposite is true if multiple players are controlling the same part of the game world, such as they do in a game like *THE WUU*, where the entire crowd is controlling the same platform. To distinguish those two types of input, we use the terms **heterogeneous** and **homogeneous** inputs. When players control separate parts of the game world as in *Enurendo*, the inputs are said to be heterogeneous. If multiple players are controlling the same part of the game word, as in *THE WUU*, the input is homogeneous. In this section, it will be explored what challenges arise from designing for each of those types of input.

7.1.1 Heterogeneous inputs & characters

With crowd games, as with any other types of games, one of the most important things to be aware of as a game designer is the feedback that players receive when playing the game. As covered in Section 3, feedback is also one of Norman's seven design principles. When designing

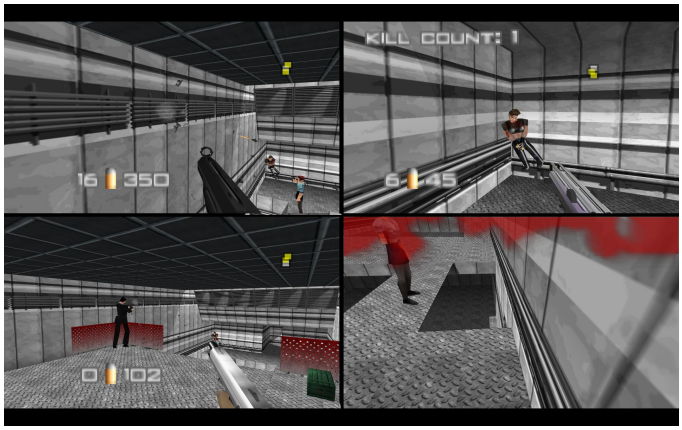

Figure 23: A screenshot from *Goldeneye*, the classic Nintendo 64 game. The screen is divided into 4 parts, one for each player.

a game, the designer wants to make sure that players have continuous clear information about the results of their actions. The basis for providing feedback to the player, is that the player can identify what parts of the screen that belong to her, and thus observe the feedback meant for her. Since crowd games need to provide feedback for so many players at once, traditional ways of designing feedback might be challenged. Here, it will be examined how one can design crowd games so that the player can identify and track her character throughout the game, in order to be provided feedback.

Figure 24: 4 player *Mario Kart 64* with the characters racing, here the players have a part of the screen assigned to each of them

Traditionally, local multiplayer games have different approaches of how to let each player identify and track her character. In existing local multiplayer games, there are several ways of doing this. In games like *Super Smash Bros.* (Nintendo, 1985) and *TowerFall* (Thorson, 2013), the screen real estate is shared by all players at once. When creating these types of games, the designer will provide a distinct character, color, or other visual mapping to indicate where each player should find the feedback that belongs to them. In *TowerFall*, the characters move around on the screen, trying to hit each other with arrows. The characters are relatively small, only taking up a very little part of the screen, and are meant to be in rapid motion (see Figure 22). This means that the player constantly needs to shift her attention to different parts of the screen in order to follow her character and have feedback. Another approach is to have a static division of the screen real estate

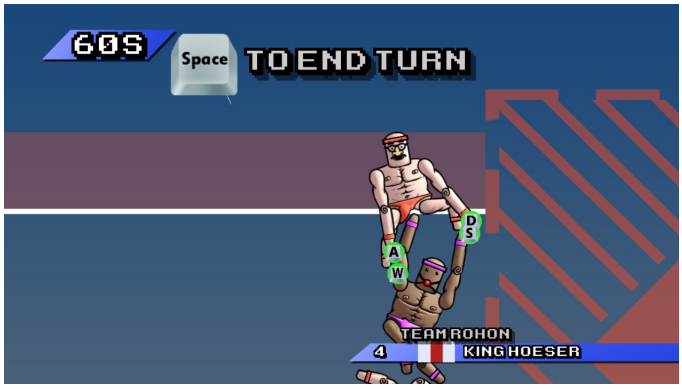

Figure 25: A screenshot from the local multiplayer game *Mount Your Friends*. Here, each player has a limited time to climb as high up on the other characters as possible.

among the players. This is done in games like *B.U.T.T.O.N.* (KnapNok Games, 2011), *Mario Kart 64* (Nintendo, 1996) or the Nintendo 64 first person shooters *Goldeneye* (Rare, 1997) and *Perfect Dark* (Rare, 2000). Players can play the game without having to look outside of the screen real estate allocated to them. *Goldeneye* and *Perfect Dark* have rather traditional first person shooter layouts on each player's part of the screen, with only the weapons of the player's character visible (see Figure 23). In *B.U.T.T.O.N.* and *Mario Kart 64*, on the other hand, the players also have a visible character as a signifier within their part of the screen. Another way to divide up local multiplayer games is shown in games such as *Heroes of Might and Magic III* (New World Computing, 1999) and *Mount Your Friends* (Stegersaurus Software Inc., 2014). These games are turn based, giving each player a temporal window of interaction. In *Mount Your Friends* players take turns playing out their move within a specific time, before handing over the PC or controller to the next player (see Figure 25). In this section, we look to traditional local multiplayer games to compare and discuss how they solve problems that apply to crowd games as well. This is done on the premise that the designs of local multiplayer games and crowd games have some common characteristics and can be compared. The commonality is that they are both local, often with one common screen. The subject of interest for this section is the common screen output, and we therefore find it relevant to involve local multiplayer games, since this genre is much more mature and has a wider spectrum of games on which to base analysis.

In many existing crowd games, the screen is shared as in *TowerFall*, where players control a small character each. This is also the case for *Jump on Heads* (see Appendix A), the first prototype made in connection to this thesis. It is a 2D platformer game that has two teams of players competing to reach a platform way above the ground. The goal for each player is to jump on the heads of her teammates or opponents in order to reach the platform. To be able to support controls for a large number of players, the framework *HappyFunTimes* (Tavares, 2016d) was used. Already during the design of this first prototype, problems arose around how to make players identify and track their character. Observations from a playtest of *Jump on Heads* with about ten players show that the players had a hard time finding their character on the screen:

> Comments while playing were that the game was too fast and that people had a hard time finding their character on the screen, especially since the teams changed with every restart of the game. (Appendix A.1)

Another place where a similar problem became apparent, was when playing the game *Enurendo*. At the initial presentation of the game, in the Chromecast room at Nordic Game Jam (see Appendix H), some players also had trouble identifying their character:

> I sat besides from one of the players, and she got really frustrated by playing the game as she had no idea what character were hers and if she was even doing anything.

Figure 26: In *Jump on Heads* the characters on a team have a name displayed and have a slightly different hue from each other. In this image, the names are all the same, because the screenshot was taken during a technical self-test of the game.

> She also mentioned that she figured it might lack [lag] a lot, and she ended up with give up on playing. (Appendix H.1)

The same thing happened at the Nordic Game Jam award show (See Appendix I), when the game was played again, this time by an even larger crowd:

> I ended up joining the game along with about 50 other players at the award show and never found my player character. (Appendix I.1)

The prototype *Colorave* also has heterogeneous inputs, and showed similar problems. When playing *Colorave* at the Nordic Game Jam pre-party, players proved to have a hard time finding themselves. As it is stated in the observation journals from the evening, some players "[...] had trouble finding themselves during the game, and some even doubted if the game was actually working, and thought it was just faked." (Appendix G.5). All in all, there seems to be a general issue in these games using heterogeneous inputs, concerning getting the players to identify and keep track of their character.

When designing *Jump on Heads*, there were however some design considerations taken, regarding how the players should identify their characters:

> For the players to easily recognize their character on the screen, we split the two teams into red and blue and then gave each player a slightly different shade of that color, so they would be able to tell the difference. (Appendix A.2)

Figure 27: *Happy Hockey* being played at JOIN Local Multiplayer Summit 2015. The characters all have names above them, that the players have chosen themselves.

Figure 28: The character in *Jump on Heads*, a person riding a seahorse.

Figure 29: The character in *Happy Hockey*, a rather simple hockey player.

All characters in the game have the same base sprite, but are colored according to their team, and are clearly labeled with the name of each player's choice. Furthermore, each character has a slightly different hue than the rest. So even though one team is primarily red, each character has a different nuance. The nuance of the character is also reflected on the controller to better communicate to the player what color to look for on the screen. All of these mappings between the player and their character are designed to make the player able to find her own character on the screen. But apparently, this was not enough to make players easily discover and track their character. A closer look to the characters in *Jump on Heads* reveals, that they are visually quite complex. The characters are a person with a cap on, riding a seahorse. It is a rather detail pixel art sprite, with a lot of pixels pr. sprite. Furthermore, it has more than 10 different colors, also

Figure 30: In *Towerfall*, the characters are very detailed, but with different colors, and also signifiers like the arrow count over each character. The blue character on the left also has a visual overlay, showing that a special ability is currently in use.

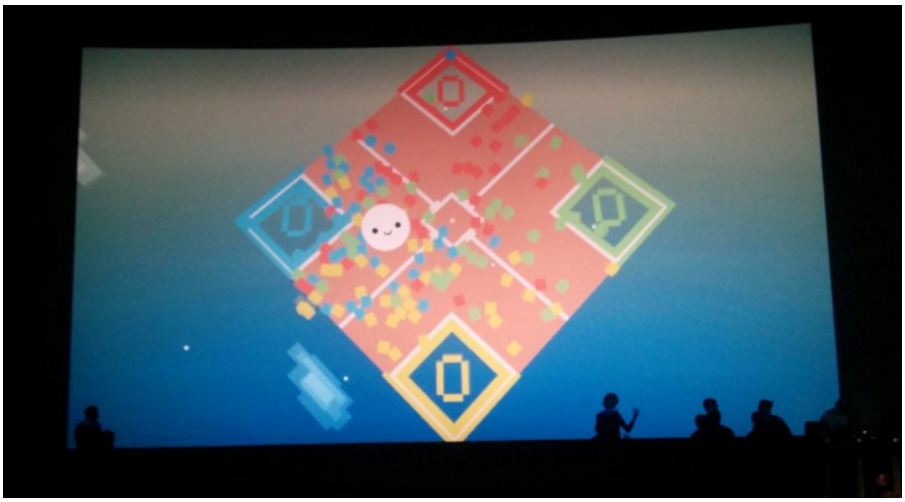

Figure 31: *Enurendo* as it looked at the big screen in Imperial during the Nordic Game Jam Award Show.

counting the different hues of the blue or orange (See Figure 28).

Another game that uses heterogeneous inputs and a shared screen is the *HappyFunTimes* game *Happy Hockey*[40] (See Figure 27). In this game, the hockey player character is more simple than those of *Jump on Heads*, with less different colors and elements. It is also a pixel sprite, but it is just the silhouette of a hockey player, with only one color per team and per character. This could point to the understanding, that characters in crowd games should be simple for the players to identify and keep track of them. This is different from traditional local multiplayer games, where characters can be even quite complex, and still recognizable. In *Towerfall*, the characters are quite complex (See Figure 30). Also here, they are each assigned a different color, and the style is pixel art. But the characters are very detailed, with clothes, crowns, hair and bows. But in comparison to *Jump on Heads*, the characters here are made up by rather large pixels, making them a little more simplistic to look at. Besides that, they also have some UI elements attached to them. The arrow count are signified by arrows over their heads, and even though it changes when the player uses an arrow, it is still one more visual element to look for when identifying and tracking the characters. When a player uses a special ability, as for instance a shield, it is also visualized on the character, and again makes the character stand out from the others. But *TowerFall* is a local multiplayer game, not a crowd game, with a limit of 4 players to keep track of. This kind of character design might not be suitable for crowd games. Still, the idea that the character sprite alone is not the only representation for the player to look for when identifying

[40]The game *Happy Hockey* is developed by Alexander Pieper and Johannes Kristmann. Unfortunately, the game is not published anywhere, but we have played it at JOIN Local Multiplayer Summit in 2015. While writing this thesis, we got in contact with the developers and got some pictures from them to use for the thesis. For a peak on the game play, a short Vine video can be found at https://vine.co/v/irA1hIiqjLq

her character, might be useful. One of the ways the characters in *Jump on Heads* are design to be recognizable for the players, is by having a name for each character, that the players themselves choose. In *Happy Hockey* there is no difference in the hue of each player's colors, so the only thing for a player to visually identify her character from the rest, is by the name. Instead the name is relatively large in comparison to the characters (See Figure 29). In *Happy Hockey*, the name is a prominent part of the character, whereas in *Jump on heads*, the character sprite is the most visually significant. In *Enurendo*, like in *Happy Hockey*, all characters are the same except they are colored according to their team. The sprites are just a block in the color of the team, and thereby even more visually simple than the characters in *Happy Hockey*. But whereas *Happy Hockey* has the name above the characters, there are no graphic elements telling the players apart in *Enurendo*.

What also distinguishes the characters in *Towerfall* from each other, is the fact that they are different sprites. The blue character is hooded, while the green one wears a crown and has long red hair. This is very common in traditional local multiplayer games, and is also the case in games like *Mario Kart 64* and *Super Smash Bros.* For instance, a game like *Super Smash Bros.* has 12 characters to choose from. Still, 12 characters is a limited amount of characters that have to be designed for the game. For a crowd game for 50 players to have a unique character for each player, it would require 50 different characters. One crowd game does this. In *Kachiku 64* (PlayWest, n.d.) each player gets an individual character. This technique might be able to also help players distinguishing the characters from one another. One could argue that this would be quite a hassle though, designing unique characters for every player when the game is made to support 64 players such as *Kachiku 64* is. However, this could possibly be done with tools for generating sprites, either programmed into the games system or by using an external program.[41]

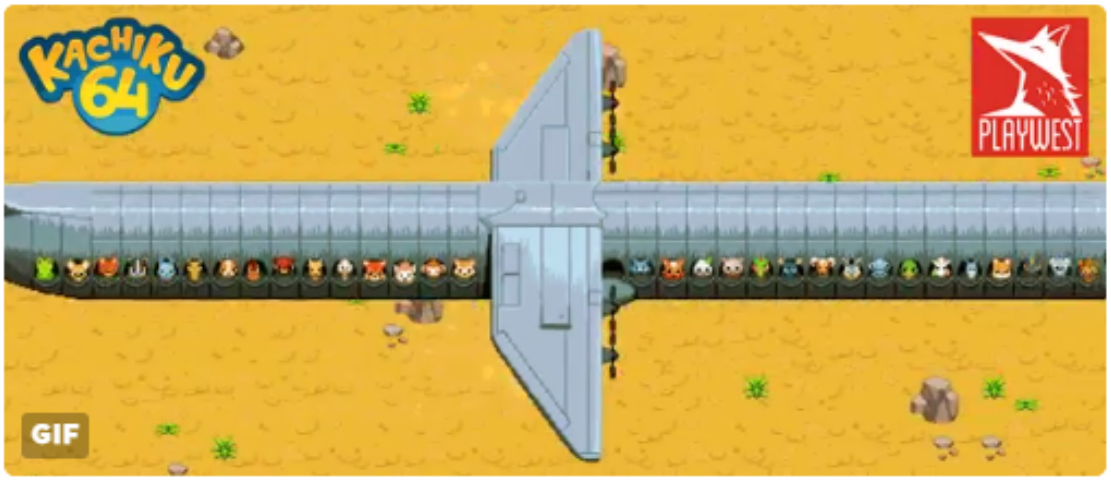

Figure 32: Screenshot from the game *Kachiku 64*. Here some of the 64 characters are showed sitting inside a flight.

But the visual side of the design might not be able to do it alone. Andy King from the studio PlayWest, that is currently working on the game *Kachiku 64* (PlayWest, 2016), has some insights on the ability of players to find their character in the game. *Kachiku 64* is a local multiplayer game for up to 64 players, using an individual controller for every 2 players as input. Andy King got interviewed for this thesis, and in the interview he shares some of his insight on the topic. He explains, that the actions and feedback on actions that players can take in the game, are closely related to the issue of locating one's character:

> Everyone thinks will be impossible to find, but finds quickly as we use special highlight system based on some research I did using a game called theif town, which

[41] Sprite generators can be found online to use for this purpose, for example the `http://img.uninhabitant.com/spritegen.html`

> actually plays on the fact you look the same as everyone else. [...] Typically once you realise what the face buttons do, you find yourself in a couple of seconds, even with random drop on [...] Gameplay isn't so affected because the lobby is diagetic (Appendix M)

In King's experience, and based on his research, players find themselves quickly once they know what the buttons on the controller do. He also states, that the players know these controls before the game starts, because they are already able to control their characters in what he refers to as "the lobby", the sign up screen showed before the game begins. The game *Thief Town* (Glass Knuckle Games, 2014) that King refers to, shares some of the same mechanics as a game called *Hidden in Plain Sight* (Spragg, 2014).

Figure 33: A screenshot from the local multiplayer game *Hidden in Plain Sight*. Many of the characters look the same in order for players to be able to hide their character in the crowd of NPCs.

The local multiplayer game *Hidden in Plain Sight* is a game that intentionally hides the visual mapping between player and character. The game is a top-down 2D game with rather small characters. In one of the mini games (shown in Figure 33), one player is selected to hide among a swarm of NPCs, while the other 1-3 players have to spot the first player. As the hidden player, the first challenge is to find oneself in the midst of about 30 NPCs on the screen. The character of the hidden player looks identical to many of the NPCs, and only from comparing her input with the movement of the characters on the screen, is the player able to find the character she controls. *Hidden in Plain Sight* has real-time control, which according to Swink gives the player a strong link between input and feedback (see Section 3.2). Opposite *Jump on Heads*, *Hidden in Plain Sight* has slowly moving characters, and uses the entire space of the screen to display them. The combination of real-time control, slowly moving characters, and a large spacing between characters lets this game use feedback as the only way for the player to find her character on the screen.

Since the players of *Enurendo* are divided into 4 teams, it is only 1 in 4 characters that matches a player's team color. Even with only about 50 players that were playing, observation still shows that identifying a character among the circa 12 that match one's color, was incredibly difficult (see Appendix I.1). *Enurendo* had some other problems entirely than those of *Jump on Heads*, though. As the observation journal says, the input of *Enurendo* had a lot of latency (see Appendix I.1). Both *Enurendo* and *Jump on Heads* use smartphones as controllers. But whereas *Jump on Heads* uses local WiFi to send inputs to the game, *Enurendo* uses an online server. This means that the signal from the players' smartphones travel a lot farther to reach the computer running the game. This was probably what caused the heavy latency in *Enurendo*, that the observed player in the Chromecast room also suspected. In addition to that, *Enurendo* was also lagging, resulting in an even greater response latency. As the observation shows, it was difficult for the player to identify the character on the screen by matching his input with the motion of the characters, when he did not know how long the latency from button press to response was. A similar reason might have been the problem for *Colorave*, which has a known latency of about 200 ms because of the

processing of data from the camera. 200 ms of delay is enough delay to break the feel of real-time control according to Swink, which means that the player feels a disconnection between their actions and the result on the screen (Swink, 2009). As opposed to *Hidden in Plain Sight*, which has real-time control, players of *Enurendo* and *Colorave*, might have had a hard time finding their character or glowstick on the screen because of the lack of real-time control.

The game *Tonde Iko* (Tavares, 2014) has an interesting approach of solving the problem of fitting many characters on a single screen. Instead of having characters move slowly as in *Hidden in Plain Sight*, *Tonde Iko* spreads the game over 6 different screens, giving players the possibility to travel between screens. This gives each player a much larger amount of screen real estate. Both *Jump on Heads* and *Enurendo* have the problem that they only use a small portion of the screen for characters to move around. *Enurendo* uses about half of the screen and in *Jump on Heads* the characters clump together at the bottom of the screen, giving them very little spacing between the characters. Spreading out characters more on the screen, giving each character more room around it, making it less likely that the player would confuse it with one of the other characters, might also solve this problem. Examples of a games that do that, are *Happy Hockey* (see Figure 27) and *Hidden in Plain Sight*.

The prototype *Colorave* has some similarities with *Enurendo* in the visual mapping between player and character on the screen. In *Colorave* each player has a glowstick in their hand and can see that same glowstick on the screen. The glowstick on the screen comes from a live stream from a webcam connected to the computer. This way, the glowsticks in the hand of a player maps 1-to-1 with a glowstick on the screen. Similar to *Enurendo*, the visual mapping of *Colorave* is ambiguous. Each player's glowstick has the same color as half of the glowsticks in the room, and thereby on the screen. There is no way for the player to distinguish her glow stick from the rest of her team's glowsticks just by its visual representation.

> We discussed and tested various possibilities, like setting the camera in the same position as it is on a laptop for example, but we ultimately ended up with a solution where the camera is above the crowd. This ensures that everyone in the crowd is seen equally, except perhaps people who are very tall. (Appendix G.4)

In *Colorave* it was the intention that player's should understand the clear mapping that existed between their physical position and their position on the screen. While testing, the camera orientation was thoroughly tested in order to create the most intuitive spatial mapping between physical and game world space. However, the game had no signifiers telling the player what mapping that was. Had the mapping been clearly indicated, the players might more easily have been able to find themselves on the screen, from knowing their positions in the physical space.

Problems displayed in this section, reveal the difficulties for players to get feedback from a game with heterogeneous input. Games with heterogeneous input solve this difficulty by dividing the game such that each player controls a different part. The different parts can be different parts of the screen, different characters on the screen, or different moments in time. Most of the games studied use characters to distinguish players from each other and there are several ways in which the games try to ensure that players are able to discover and track their characters. It is done through the player's real-time control of their character, the visual distinction of each character, the amount of screen space given to each character, movement speed of the characters, and for some games as *Colorave*, the mapping from something in the physical world to something in the digital. This reveals an aspect of crowd games that could benefit from further research into how character identification and tracking can be improved in crowd games that use heterogeneous inputs. By taking a more systematic approach into researching this specific area of crowd games, it might be possible to uncover the relationships between the different solutions and guide designers more thoroughly on how to use them.

7.1.2 Homogeneous inputs

In the problems displayed from *Jump on Heads*, *Colorave*, and *Enurendo*, it might be suggested that designing games that use homogeneous inputs with fewer characters would ease the ability for players to track a character and thereby enhance the clarity of feedback for each player. Examples of games that do this are *THE WUU* (Headspider, 2015), the games designed by Colossi, and our own prototype *BlowIT* (see Appendix D). In *THE WUU*, the players in the crowd together control a huge platform, moving it up and down using their voices. In games by Colossi, the crowd is split into two huge teams, each of them in control of a character. Each player influences the character's movement by swiping on her smartphone in the direction she wants the character to move. In *BlowIT*, the crowd is split into 4 teams. Each team controls the updraft in their section of the screen by using the movement of their bodies. These games have in common that they might as well have been single player, 2 player or 4 player games respectively. But through their use of homogeneous inputs to control the platform, the characters, and the updrafts, the games are instead controlled by a whole crowd. In this section, ways of combining inputs from a whole crowd into homogeneous inputs are explored.

A central question regarding homogeneous inputs is in what part of the interaction between the players and the game world, the inputs get filtered to form the combined input. To explore this, we look to Swink's 7-step interaction model discussed in Section 3. Somewhere between step 2, the muscles, and step 5, the game world, the signals sent from the muscles of players, are combined and put into the game world as a single signal. This combination of inputs is achieved through filtering (see Section 5.1) and is therefore called **combination by filtering**.

Filtering can mean many things, but applies any time one signal is modified or changes form. This could for example be the press of a button on a controller. The button filters the movement of the finger, into electrical signals. In a crowd game like *THE WUU* the microphone acts like a filter. Each person has their own voice, but the microphone does not distinguish between who in the crowd made each sound, it simply takes input in as **anonymous** and passes it on to the computer. The input being anonymous just means that the computer cannot differentiate between different players in the input it receives. If we look to Swinks model, the combination of inputs in this case happens at step 3, input device. We call this **combination by input device filtering**.

The crowd games *THE WUU* and *WOOORRK!* (Garbos, 2013) both use the microphone as an input device and thereby a device for filtering the inputs. Both games only use the total amplitude of the signal from the microphone as input after it has been processed by the computer. Even more simply, *THE WUU* uses a threshold to switch between on or off, so the signal from the microphone is essentially turned into a binary signal in the end. This means that the crowd only needs to learn how to switch between two different states, making sound, or not making sound.

The opposite of having an anonymous input, where the computer can't distinguish between different players in the input, is having an **identified** input where the computer receives one or more signals, in which each player can be identified. But even if a game uses identified inputs, such as a smartphone controller, the inputs can still be combined to a homogeneous input. In the games designed by Colossi, players all have their individual smartphones as controllers. Only inside the computer, does an input signal get combined with all the inputs from other players of the same team. Here, it is step 4 in Swinks model, the computer, that does the combination of inputs. We call this **combination by computer filtering**.

An overview of the different possible scenarios relating to combination by filtering is shown in Figure 34.

In the realm of combination by computer filtering, the most clean examples are the games by Colossi (see Appendix L for an interview with the developers from Colossi). In these games, input from each player are gathered by the computer over the local network. The computer then takes that input and combines it with all the other input from players on the same team, making one player's input only count as a fraction of the total input of how the character in the game world should move. The interview with Jonas Ahm who has played Colossi's games (Appendix K.1), surprisingly indicates that Ahm felt like he had an amount of control over the character

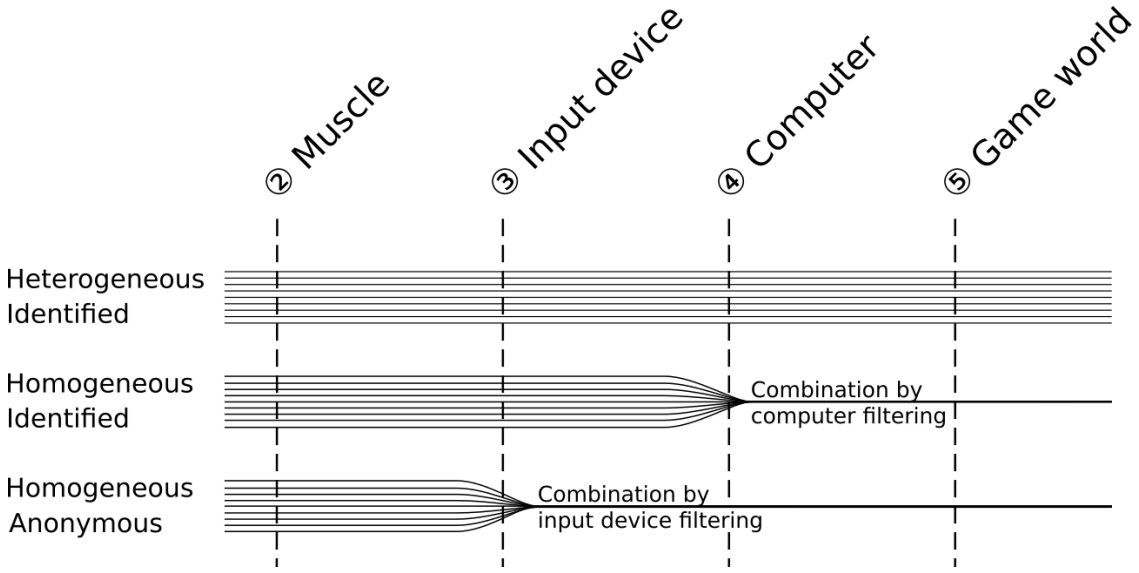

Figure 34: This diagram shows the correspondence between Swink's interaction model, different types of input, and types of filtering. There are three different scenarios shown. From top to bottom there is first a scenario with no filtering, which means that the game in this scenario has heterogeneous input and identified inputs. Second from the top is a game that uses combination by computer filtering, which means that it has homogeneous inputs that are still identified. The last scenario uses combination by input device filtering, meaning that it has homogeneous and anonymous input.

even though he knew that he was only one player out of around 400 players playing on his team. From the interview with the developers, it was gathered that single inputs are visualized more clearly than their actual impact on the movement of the character.

> if everyone is swiping up, and one person swipes down, the down person can still see their input. So, it's. It's kind of like, 'hey, I want to see, no one is swiping to the left, I'll swipe to the left, oh hey there is my thing, I did something.' (Appendix L)

Colossi's visualization of the players' swiping even gives feedback to players who swipe a different direction than the majority of the crowd. This quote shows how Colossi made clear feedback to players, even though each player has a very small influence on the outcome of the game.

Games by Colossi, while using homogeneous inputs, take advantage of the fact that the inputs are actually identified by the computer. This means that they can have the computer send customized feedback back to each smartphone. They use this to send small achievements to the players at the end of each round, to summarize how that player did in the round.

> So, now we have these achievements where we can kind of keep track of what direction you are swiping in, compared to everybody else. So.. If you, if everyone is swiping up, trying to get to the goal, and it's right above them, and you are swiping down, we can track that and say 'well, you swiped against your team, like, this percentage of the time' (Appendix L)

The achievements let the players understand that they actually had an impact on how the game played out. It shows them that their inputs were recorded and used, by sending them small messages between rounds.

Combination by computer filtering also appears in the prototypes *Put a Smile On* and *BlowIT* that use camera as input. In *Put a Smile On* players are randomly assigned a color – either red, green or blue – which appears on the screen of their smartphone. The game needs to be played

in a dark room, so only the light from the smartphones light up. In theory, the signal from the camera to the computer then clearly shows the positions and colors of all the smartphones in the room. This raw input is then combined by computer filtering, in order to track movement of the colored smartphone screens. The more movement a color has, the more points are added to that team's score. So in the end, the complex input from a camera is filtered into a single number for each color, which increments the scores. This gives the player clear feedback on how well her team is doing in the game but it does not tell her much about her own role in that score. Just like games by Colossi, *Put a Smile On* uses the raw input from the device, in which each player can be identified, to give direct feedback to each player. This is very simply done by mirroring the video stream from the camera and displaying it in the game. This means that all players can find themselves on the screen and compare their own movement to the movement of the other players. In addition, the game clearly indicates how much the player moved their smartphone each frame. As stated in the design journal for *Put a Smile On* (see Appendix B.1) it was found that mirroring the image from the camera presented more intuitive feedback to the developers when testing. This relates to the mapping between player movement and the response they see on the screen just like for *Colorave* in Section 7.1.1. If the webcam image is mirrored, the player gets the same mapping between their movement and the image, as they would when looking into a mirror. If the player moves her hand up and down, it is also moves up and down on the screen, and if she moves her hand from left to right, it also moves left to right on the screen. This is the natural mapping because people are used to seeing themselves as reflections in a mirror, and worked a lot better than displaying the raw video input.

The prototype *BlowIT* uses a very similar method to provide direct feedback to each player. As stated in the design journal for *BlowIT* (Appendix D.1), one simple improvement the prototype has over *Put a Smile On* is that the direct feedback provided to each player uses the color of the team to indicate where motion happens. It does this by coloring all the pixels from the webcam feed that have a large change in intensity from one frame to the next. The combination by computer filtering in *BlowIT* is based on a division of the screen into 4 vertical columns. The game takes the sum of all movement in a column and uses that as the input for the corresponding team. As opposed to *Put a Smile On*, this means that the lights can be on when playing *BlowIT*, so it is more easy for the players to find themselves in what is essentially a large mirror. It is even required of the players that they find themselves in order to know what team they are on.

Interestingly, looking at the games that use homogeneous inputs, almost all of them use one homogeneous input for each team. In *THE WUU* there is one homogeneous input and everyone is cooperating. In *Put a Smile On* there are three homogeneous inputs – three colors – and there are three teams. *WOOORRK!* is the only game that sets itself apart in that it has a single homogeneous input, the microphone, but three teams. *WOOORRK!* solves this problem by giving teams set time frames where they can control their character.

This section outlines some of the ways homogeneous inputs are used by games, and how different games solve the issue of providing feedback to players when their inputs are combined with the inputs of other players. In games that use combination by computer filtering, the fact that the inputs are identified, means that players can be given individual feedback. In games using combination by input device filtering, the inputs are anonymous and all feedback must therefore be given to the crowd as a whole. In the next sections, an alternative method for combination by filtering will be presented, and a way of providing feedback for games that use combination by input device filtering will be introduced.

7.1.3 Combination by mediator filtering

There is also another rather unexplored way of combining heterogeneous inputs. In the local multiplayer game *Sentree* (Glitchnap, n.d.), a single blindfolded player controls a character using the gyroscope and touchscreen of a smartphone. All the other players can see the game world on a separate screen, i.e. a tablet, and have to communicate to the player with the smartphone what to do. This means that most of the players actually have no direct input to the game, they only

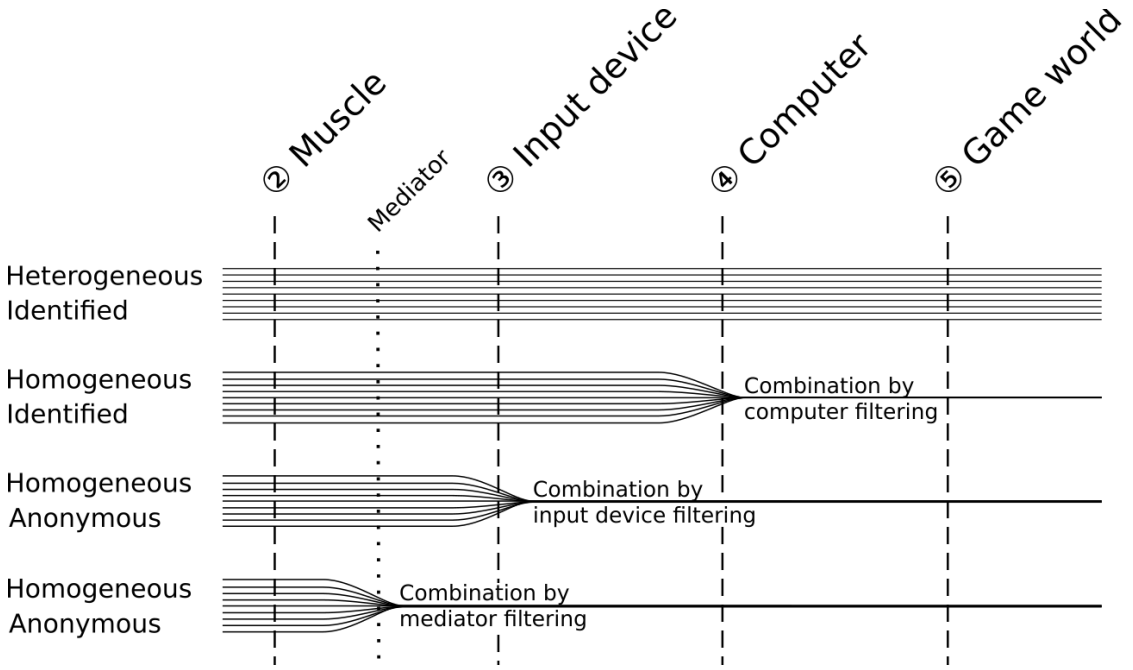

Figure 35: This diagram shows how combination by mediator filtering fits into Swink's interaction model. The diagram is similar to that in Figure 34, but includes a fourth example at the bottom of the diagram, where inputs are combined by mediator filtering. This combination method leads to homogeneous and anonymous input.

have their communication with the player with the smartphone. The player with the smartphone becomes a filter which translates the other player's input into a touch gesture or other types of inputs to the game. We call this player a **mediator**.

Figure 35 shows how the mediator fits into Swink's interaction model. The mediator acts as a layer between step 2, muscles and step 3, input device. The mediator is used as a filter to combine the inputs from the crowd and we therefore call this **combination by mediator filtering**.

The idea of using a mediator as it is used in a game like *Sentree* (Glitchnap, n.d.) was combined with the idea of having multiple players vote for the inputs for a game, as is done in the internet phenomenon *Twitch Plays Pokémon*[42] by testing the use of mediators for crowd games:

> Knowing these two examples, it seemed obvious to use these games as inspiration on how to make a crowd play. Combining the two ideas, you get a singleplayer game, hacked using a mediater to translate input from a crowd to a conventional input device. (Appendix E)

To try out how combination by mediator filtering can be used for crowd games, we tried hacking single player and local multiplayer games into crowd games by using mediators. For example, we took the local multiplayer game *Soccer Physics* (Ojala, 2014) and let two mediators control the button for each team (see the observation journal in Appendix E). The crowd was then split into two teams, each team signaling to their respective mediator when to push the button. The general idea is to take games which use conventional input methods, like controllers, mouse, or keyboard, and hack them into crowd games by inserting a mediator between the input device and the crowd. The mediator is disallowed from looking at the screen, and is instructed by the crowd on what inputs to give. We call these modified games **crowd hacked games**. The instructions

[42]Twitch is an online streaming service for games. In *Twitch Plays Pokémon*, the original game of *Pokémon* is controlled through an automated system while people watch over the internet. The system makes inputs to the game based on the comments from players in the Twitch chat.

to mediators can be given in many ways and in the prototypes shown at the LET'S PLAY event F, both audio and visuals were used, using both clapping, voices, and gestures.

One lesson learned from designing and running the LET'S PLAY event, is how easy crowd hacked games are to set up. They can be done on the fly, in any setting with a computer, a projector, and a crowd. Observation from the LET'S PLAY event also shows that a mediator can take relatively scattered input, interpret it and convert it into somewhat precise button press input. Both for single presses of a button and for holding and releasing a button. However, mediators had a hard time converting two rapid consecutive inputs into button pushes. An interesting observation was that players came up with their own gesture for the game Mole Hammers. At some points in the game, the two mediators have to smash their button rapidly in order to get a lot of points. Players in the crowd came up with various inventive ways of gesturing this. They used common body language to communicate to the mediators. This shows that using mediators can have the advantage of having the players invent new strategies of communication – and thereby input – through playing the game. Observations from the LET'S PLAY event also suggest that using mediators make the rules of the games more flexible. At one point, a player started clapping as communication to the mediator instead of doing the gesture that had been chosen as communication: "Astrid started clapping in the throwing game. She says 'I think this is too hard to see' while making a throwing motion with her hand." (Appendix F.8). She thereby broke one of the rules that had been set for the game by the host, but the mediator can adapt to that easily and interpret her intend.

While a blindfolded mediator and a hardware input such as the microphone both react to sound, they are rather different types of inputs. The microphone is of course much more precise in its measurements and much more detailed in its signal to the computer. On the other hand, the microphone does not interpret the signal in any way, it only stupidly passes it on to the computer, which is then left with the task of processing the signal. Looking at the games *THE WUU* and *WOOORRK!*, which use microphones, it is seen that they in the end only use the total amplitude of the signal from the microphone as input after it has been processed by the computer. Both games use only a very simple mode of the microphone signal. The games interpret the signal very precisely, which would not have been possible using a mediator. The mediator, however, can understand language, differentiate voices, and react to unforeseen circumstances, in a way that the computer is not able to.

In *Sentree*, the mediator controls the character on screen using the gyroscope and touchscreen of a smartphone. What is interesting about this input, is that the mediator actually has a continuous directional input in 360 degrees of movement. On top of that, the mediator also has two binary inputs, that are shooting and reloading the gun. These inputs are already much more complex than the ones for the microphone controlled games. On top of that, the players of *Sentree* are not given a protocol for communicating with the mediator. They need to make up the communication on the fly, which would be impossible to do with a computer. In the end, one could say that deciding between the mediator and microphone comes down to the question whether one wants the precise raw computational power of the computer to process the input or the slower but more adaptive mediator with a more complex interpretation of the signal from the crowd.

But it is not only that, instructing a mediator to interpret input, will often be much quicker and simpler than instructing a computer to do the same. It therefore also comes down to the time needed for development of the game.

In conclusion, the use of humans as mediators of input into crowd games opens up a number of different possibilities. Mediators are flexible in a play setting and can interpret the rather complex signal from a crowd and turn it into a simple game input, without having to be programmed such as a computer. Therefore, a mediator can be used for hacking any game into a crowd game without the need for a huge setup, and for testing purposes, mediators can be used as stand-ins for technology. However, mediators are not as precise in their handling of input as a computer is and they react more slowly than computers do.

7.1.4 Observable inputs

In a traditional local multiplayer game such as *TowerFall*, players use controllers as input devices. As Swink describes in his interaction model, the act of using a controller gives tactile and proprioceptive feedback to the player. However, the action does not give any clear feedback to the other players. The controller is situated far from the screen, where the player has her main focus in these types of games. This is a good thing for a game like *Hidden in Plain Sight* where players are encouraged to conceal their inputs as much as possible. But some crowd games, as will be explored in this section, make explicit use of inputs that are not at all hidden.

A player playing a game such as *THE WUU* or *WOOORRK!* does not only have the screen and the host available to give them feedback, the player is also able to hear the other players around her. E.g. in *Renga* players are able to see the red laser dots from the other players in the crowd. This form of feedback we choose to call **observable inputs**. In Swink's interaction model, feedback streams from step 6, output devices, to step 7, the senses and from step 2, muscles, to step 7, the senses, as tactile and proprioceptive feedback. Observable inputs also flow from muscle to senses, but flow from one player's muscles to another player's senses. In *THE WUU*, when a player hears that the voices of the crowd go from quiet to loud, she will simultaneously see the platform in the game rise. The observable input gives her a sense of coherence between the inputs and what is happening in the game world. The observable input means that players can adjust how loud they are, based on the how loud the players around them are. In *WOOORRK!* players can hear how well the other teams are doing just based on the observable inputs. In *Renga* players can adjust their strategy based on the information they get from seeing how other players are moving their laser dots. Interestingly, the observable inputs in *Renga* are not much different from normal visual feedback from a game, in that the laser dots move around on top of the game world, that they are in a way part of. The laser dots are in a way the characters of the players.

THE WUU and *WOOORRK!* are able to take the homogeneous input from the crowd and expose it directly to each player, which is possible because they use a microphone as input device. However, in *Put a Smile On* the input from each player is given through a smartphone that they hold in front of their mouth. During the presentation of the game, Simon realized that players were not seeing each others' smartphones, but rather staring at the screen and into the backs of those in front of them.

> Watching the crowd play, I realized how it must look from the players' perspectives. They don't see a sea of players with smiles on their mouthes, they see the back heads of other players and a screen, where they can barely recognize themselves. (Appendix B.2)

As Simon was looking at the crowd, he saw everyone with their smiling smartphones in front of their mouths, but he understood that the players must be seeing something else. While players were not able to see each other's smartphones, they were able to see the movement of those in front of them, making the inclusion of observable inputs partially successful. The same was true of the crowd hack of *Soccer Physics* where players used their movement of their bodies to signal the mediator. A comment from the playtest of the crowd hack was that one of the players in the front row was lacking feedback and that she could not discern how the rest of her team was doing. From the observation journal of the *Soccer Physics* crowd hack:

> She pointed out the problem that she could not feel/read the crowd very well because she was sitting in the front row. Therefore she felt less significant and could not tell exactly what the volunteer [mediator] for her team was reacting on. (Appendix E.2)

The fact that the player was sitting all the way in front, meant that she could not observe the inputs from her team mates, that were sitting behind her. This illuminates a flaw of observable inputs in games using a visual as input, whether it be detected by camera or by a mediator as

in this case. Players are only able to see the input from players in front of them, meaning that only the players at the very back might be able to observe the entire crowd. Also, if players need to keep their sight on the screen, they can only use their peripheral vision in order to sense the inputs from the crowd around them, as opposed to sound based games where players can keep their eyes on the screen while hearing everything at the same time.

Put a Smile On, *BlowIT*, and *Colorave* all used a camera as input device, and for all three games, the image feed from the camera is shown on the screen, in a more or less filtered form. In the design journal for *Put a Smile On*, Simon explains why this was done, "We liked having the webcam image of people show in the background of the game, so people could actually see themselves play the game." (Appendix B.1). Having people see themselves and others while playing the game attempts to solve the problem of the observable input being hard to see for the player. Players can instead see what they and their team are doing directly on the screen that they are looking at anyway.

As it can be seen from this section, observable inputs give additional information to the players. This information acts as feedback that gives the player a coherence between the game world and the input. It also acts as information which players can use to adjust their strategy. Camera based games have the inherent flaw that players cannot see the movement from players behind them, which is tried solved in some games by projecting the video feed to the screen, and in *Renga* by giving players an extension of themselves in form of a laser pointer. Games based on auditory inputs use an input that can easily be observed, even when not looking at a screen.

7.1.5 Additional insights on using cameras as input devices

The use of cameras as input has already been discussed in relation to placement thereof and how they function with observable inputs. Since both the prototypes *Put A Smile On*, *BlowIT* and *Colorave* uses camera as inputs, this specific input has been explored further than other types of input for this project. This section outlines some of the other insights found when working with cameras.

When working on the prototype *Put a Smile On*, one of the main challenges of making the game, was getting the camera input to detect colors of the smartphone screens. From the observation journal, we explain the difficulties of working with a webcam borrowed from the university, which automatically adjusts exposure and white balance.

> One technical problem we hit very early on was that the webcams we had available automatically adjusted exposure and white balance. This meant that colors could easily look different from setting to setting and it was impossible for us to compensate for the settings that the camera automatically set. The solution was to be as generous as possible when detecting colors, but ultimately this was not a very good solution because we were still unable to detect very bright and very dim screens. (Appendix B.1)

Not having a consistent rule for the measurement of brightness and hue of the colors from the camera, meant that a green smartphone screen would appear very differently depending on the other lighting in the environment. It would even appear different in the same environment, depending on how the webcam decided to adjust exposure and white balance. For the prototype *BlowIT*, this problem could be ignored because the game did not have to distinguish between colors but only spatial placement instead. For *Colorave* it once again became relevant to distinguish between colors, but this time we took multiple precautions in order to be able to detect colors. In the designing of *Colorave* three colors were reduced to just two colors; smartphones were substituted for glowsticks, which have much more consistent lighting; and a new webcam was acquired, which had manual adjustment of both exposure and white balance (see Appendix G).

Another difficulty that arises for cameras is that they produce a 2D projection of a 3D environment, meaning that objects or players closer to the camera will have a larger size than objects or

players farther away. In *Put a Smile On* players that were far away from the camera had much less influence over the game, than players that were all the way up close. This meant that some teams might have an advantage if they randomly had many players in the front row, close to the camera. In *BlowIT* the game was divided into vertical columns to prevent this distortion and enable all teams to have a fair distribution of players in front and in the back. This, however, still meant that players in the back of the room had less influence than players in the front. Finally, for *Colorave*, the camera was positioned such that it recorded the players from above. This almost completely eliminated the problem, only allowing for taller players to gain a bit more advantage than shorter ones.

7.1.6 Conclusion on designing interaction

This section has provided an overview over what possibilities for interaction and feedback different input methods provide. With heterogeneous inputs, the main issue discovered during this project is how to provide clear feedback for the individual player. Possible areas to focus on to solve this are suggested. This includes solutions based on the visual aspect of the game, the spacing between elements of the game, but also real-time controls and a good mapping between a player and their part of the game are suggested as important. For homogeneous inputs, it is showed that they can be divided into 3 categories according to how they combine the input: combination by input device filtering, combination by computer filtering, and combination by mediator filtering. Combination by input device filtering provides the opportunity for the input to be observable, which can then give the player feedback directly from the other players in the crowd. Combination by computer filtering provides the game with inputs that can be identified, meaning that individual feedback can be given to the players. This can for example be done directly to a player's input device. The third kind of combination, combination by mediator filtering, is placed in Swink's model, and shows that the original model has its shortcoming when it comes to crowd games. Still, it is possible to use his framework to understand crowd games and the mediator as input. Mediator filtering is demonstrated to be easy to do and versatile. On the other hand, it is also imprecise and rather slow compared to a computer or device as filtering. Finally two insights on the use of camera as input are given. First, that the input from a camera is difficult to filter if exposure and white balance are automatically adjusted. Second, when filtering camera input, it is important to keep in mind that cameras project the 3D world onto a 2D image, making things that are farther away smaller.

7.2 Testing

One of the challenges when designing a crowd game is how to test it. This section explores the area of interest, testing, as defined in the research question (see Section 1.3). In general when designing games, testing is at the core of the iterative design process. Testing is also one of the reasons to prototype. Prototypes present a way of testing assumptions and design decisions.

In crowd games, it is hard to test in conditions similar to those the game is designed to be played in, since getting a crowd to test a game is not easily done. During this project, there were a few opportunities to test with crowds. But common for those experiences was that there was only one shot at testing.

To explore how crowd game prototypes can be tested, this project included the prototyping and testing of 5 crowd games. For the purpose of describing the tests, test sessions are divided into 3 categories: **Self-testing**, **small scale playtesting**, and **crowd playtesting**. Self-testing is the informal testing conducted by developers and maybe the few people around them in their development environment. Small scale testing refers to a more organized playtest. The small scale playtest has more players but still not enough that they can be said to form a crowd. Finally, the crowd playtesting refers to the organized playtesting with an entire crowd of people. In the following it will be described, what the different types of tests can be used for when designing crowd games, with a focus on when it is necessary to scale up or down the testing.

Since none of the games made for this project ended up fully polished and refined, this section is mainly focused on the early stages of playtesting. However, thanks to the interview with Colossi, this section also includes some insights on the part of the testing that is more related to the later states of design. This section focuses on what challenges can arise when testing a crowd game. It also proposes some possible solutions to these challenges. Furthermore, it includes some methodical reflections on applying the findings from small scale testing to crowd games in general.

7.2.1 Playtesting

Since the scope of the prototypes for this project is rather small, there has not been time to test and iterate on all of the designs based on these tests. Instead, a lot of self-testing was done, trying out the game by ourselves. Many features and parts were also tested on a few people, mostly the people in our office and other visitors. According to *Game Design Workshop: A Playcentric Approach to Creating Innovative Games* (Fullerton, 2008), self-testing can be a useful tool in the beginning of the design process, specifically for designing the core of the game. According to Fullerton, the goal of this kind of testing "is to make the game work, even if it is only a rough approximation of the prototype" (Fullerton, 2008, p. 273). All of this is done in order to be able to later test in a bigger setting, closer to the intended setting for the game to be played in.

When the prototyping is taken from the office and out to a more formal test setting, there are several ways of conducting the playtest. Observing playtesters playing is often the core of the playtest[43], and is often combined with different types of interviews (Fullerton, p. 289). For this thesis, testing was done by observation combined with informal talks around the games afterwards.

7.2.2 Testing for interaction

When designing the prototypes, iterations were to some extend based solely on self-testing. The core of *Colorave* is coloring the screen and filling fields with color. The first prototype that was made for *Colorave* isolated that functionality by filtering the camera inputs and making it possible for players to draw on the screen. When trying it out, it became apparent to us that it felt intriguing and engaging to color the screen by using colors in the physical world, captured by the camera. This was easily tested by us and people that happened to be around us at the time. It was not necessary to collect a crowd of people to test that core part of *Colorave*. Same goes for BlowIT, from which the design journal states:

> First thing we made a prototype that were basically just a toy. It was a screen where you should control the air[']s upwards force by moving your arms. By doing this, you can keep the ball in the air. We tried it out a bit with people passing by at the office and with each other. When we thought it felt nice, we iterated on the goal of the game. (Appendix D.1)

A general notion from the prototypes is, that the basic functionality — how one individual player interacts with the system in order to play — was tested quite well using self-testing. Getting the feel of the airflow in *BlowIT* right, designing conditions for how to capture a field in *Colorave*, or trying out how the movement in *Put A Smile On* feels. One explanation for this, is that all of these games can in fact be played just by a couple of players. The core interactions of these games can therefore also be tested by just a few players. It especially might make sense for a game with heterogeneous inputs like *Colorave*, where the fundamental part of the interaction does not change depending on the number of players. Whether there are 2 or 50 players, the core interaction of *Colorave* is still coloring the screen with your glowstick. *Put a Smile On* and *BlowIT* are different in that they have homogeneous inputs. In both games, the inputs are combined by computer filtering. This makes the feedback to the individual player mix with the feedback to

[43] Unless a more quantitative approach gathering data in-game, using metrics to understand how the players interact with the game

other players, which changes the interaction. So even though the interaction of *BlowIT* and *Put a Smile On* can be self-tested, the interaction changes as the game scales, which might render the self-testing insufficient for these purposes.

Another game where the core functionality is not very well tested with self-testing is *Jump on Heads*. When there is only a single or a couple of players playing the game, achieving the goal is a very easy task. The player only has to have their character jump on the head of one other character to reach the platform with the cake. As the number of players scales up, however, the gameplay theoretically scales very differently, and players have to work together in order to reach the platform with the cake. This way, the core functions change when more players play the game. Unfortunately this was never tested with a crowd, and it is therefore left unsolved how the core functionality plays out for a crowd playing the game.

All of the prototypes are played in teams, and to test this aspect of the games, it requires that there are enough players to actually divide teams. Comparing the testing at the LET'S PLAY — a small scale test — with self-testing, the LET'S PLAY event let us have teams play against each other as opposed to just having one player per team or even one player playing for multiple teams as had been done in some of the self-testing. For example the prototype *S.T.A.C.K.*, had only been self-tested up until the LET'S PLAY event. In *S.T.A.C.K.*, players are divided into 4 teams, but in the self-testing the prototype had only been tested by two players. From the observation journal of the LET'S PLAY event (see Appendix F.7), however, nothing indicated that players were concerned with what team they belonged to; players focused more on playing their own game. This highlights that the players did not play as we intended, in team, but instead played as individual players. If the prototype was to be iterated on further, this could have guided the design in a direction that was not taken into account when only self-testing was done.

7.2.3 Iterating & testing

Some features need large scale testing in order to be tested properly. But the bigger the test setting is, the more rigid it gets. It is hard to ask playtesters to sit and wait for a designer to iterate on a design, so iterating during the test session will most likely have to be done seamlessly and quick.

At the LET'S PLAY event (see Section F), the games *Canabalt*, *Mole Hammers* and *One Button Bob* were played in crowd hacked versions, using one or two of the playtesters as mediators pushing the buttons in the game. Multiple rounds were played of each game, and in some rounds the playtesters were asked to use different signaling to the mediators. The games could thereby be changed during testing. *Colorave* also contains several rounds of play, with several different modes of playing the game. Those modes are in some respects similar to small iterations of the same game, informing the designers how players react to different challenges. The key to being able to try out all of these different things, is also that the rounds of these games are so short.

When the game host calibrates the camera sensitivity during a playtest of *BlowIT* one could argue that this is a way of iterating on the design while testing. What is important in this regard is that it is hidden from the playtesters that this is going on. The tools for calibrating the camera are already embedded in the system of the game for the host to use. An aspect of testing crowd games that could benefit from further research is the possibility of extending the tools available for the host to calibrate the games on the go, with possibilities of saving those calibrations in order to generalize based on the tests. This way, design decisions relating to balancing could to some extend be tested more in-depth even though it is tested in a crowd setting.

7.2.4 Testing for understanding

When testing a crowd game, one could think that the closer to the intended context the testing would be, the more the designers can learn from them. But this might not always be the case. During this project, some of the testing was done with a group of playtesters that can not be

described as being a crowd, simply because they were not that many people. At the LET'S PLAY event held at ITU a small group of playtesters were present. While playing the prototype *BlowIT* it became clear that players had a hard time understanding how the interaction with the game worked. What had been tested in a smaller and more informal setting by ourselves and our friends at the office, was still not fully understood by the playtesters:

> People had a problem understanding that movement caused the balls to float. For example: Astrid asked whether it was possible to push down the ball. (Appendix F.4)

This was not that the interaction did not work, but rather that the players did not understand how they were supposed to interact with the game. Issues like that were the general subject for all the tested games at the LET'S PLAY event. The issues did not make the games unplayable, but illuminated a problem in the understanding of the interaction with the games and required some explanation or clarification. This meant that we had the chance to talk to the players about the things that confused them in order to understand what exactly was unclear. Because there were so few playtesters, we could also easier instruct them individually when the prototype itself was inadequate in doing so. If these prototypes were tested with a crowd, chances are that we had not gotten an understanding of the players' problems, since some of them might just have played along not really knowing what was going on. An example of this was the playtest of *BlowIT* at Demodag a few weeks before the LET'S PLAY event. Playtesting at Demodag with about 50 people, there were no questions or comments during the playtest. The context of Demodag and testing with this many players, does not prepare the ground for people asking questions or commenting with their insights or problems playing the game.

From the interview with Colossi, it also becomes apparent, that they had a similar experience with testing their crowd games. According to Benjamin, they use these kinds of tests to "try and just iron out all these, these user issues, user experience issues" (Appendix L). Because, as he explains, these kinds of issues are hard to solve in a crowd setting:

> During our earlier playtests, people would like, hold their phones to us, be like, 'oh, its broken, I don't know what I did. And, it's like. Okay, like, you obviously can't do that in a large, large group setting. (Appendix L)

So this kind of testing, on a scale smaller than a crowd, has its advantages when trying to get a more in-depth understanding of how players interact with the prototypes.

7.2.5 Testing for setting

Figure 36: The countdown timer from *Colorave* when played at ScrollBar.

There are however also aspects of the prototypes that a small scale playtest might be less suited for testing. An example of this is the timer in *Colorave*. For the ScrollBar playtest of *Colorave*, a countdown screen was implemented. The intent was that the countdown screen would let potential players know that a game was about to be played (see Figure 36). In a small scale playtest environment like for example the LET'S PLAY test session (see Appendix F), the purpose of the gathering was to play games. But at ScrollBar, the possible playtesters were at the venue to go to the university bar. Only few of the people there had come with the purpose of playing *Colorave*. Therefore, players were mostly strangers to the game and to the designers, and did not know that a game was about to begin, until the screen told them that it would. Testing whether the countdown screen functioned as intended, needed a crowd like the crowd at ScrollBar in order to be tested. It needed for the players to be engaged with something else in order to test how it communicated to these unsuspecting players. This is an example of an aspect of a prototype that needed a crowd and a crowd game setting in order to be tested. Specifically, we needed the assembly of strangers at an event to see if the countdown worked for that purpose.

When designing the prototypes, a lot of the design was based on intuition and assumptions of how a crowd would play the game, without actually knowing whether these assumptions would be true. When playing the prototypes in the context in which they were supposed to be played, it becomes clear whether players interact with the games in the way the designers actually intend. At the ScrollBar test of *Colorave*, the game was taken out of the comfort of a controlled self-testing setting and into a party setting resembling the one that the game was designed to be played in:

> I was afraid if the drunker part of the crowd would participate, or if they would do something that would ruin the experience for the rest of the players. But they actually all managed to participate. (Appendix G.3)

This was important to test because there could have been a big difference in how players would act depending on their state of mind and the setting they would play in. As it turned out, drunk players at a party were also interested in joining and playing along.

7.2.6 Testing technology

Testing in a crowd setting can also have an importance for testing technology, in that the space in which a crowd testing is conducted is not the same as is used for either self-testing or for small scale testing. For example, setting up the camera and projector for *Colorave* in ScrollBar revealed some of the flaws in the design. Even though there was a preliminary plan for how to set up the game in ScrollBar, the observation journal reveals some of the difficulties of the setup.

> When we got to the party, however, there was no stage and the speakers were blocking the view from setting up the camera close to the screen. We therefore had to place the camera out to one side, covering only most of the dance floor and from an awkward angle. (Appendix G.3)

Setting up in ScrollBar revealed one of the challenges with using a camera as input device for a game. From our own testing, the camera had been mounted immediately above the screen which the game was shown on. However, achieving this setup in ScrollBar was not possible, something that only occurred in the act of trying to set up the camera. With *BlowIT*, the experience was similar when showing the game at Demodag. The observation journal (see Appendix D.2) states that the camera which was initially intended to be used for the game, turned out to have a field of view which was too narrow to capture the entire crowd. Improvising, the built-in camera in the laptop had to be used. The technical difficulties displayed for the playtests of *Colorave* and *BlowIT* might have been foreseen, but testing the games in their intended setting, brought the matters into light and manifested them as problems.

Another advantage of using playtesting is that having many playtesters makes for a more diverse crowd than having only few playtesters. One way this is apparent is in the games using smartphones as controllers. When designing for smartphones, it is necessary to test if the game works differently for different devices, operating systems, or browsers. The devices used by different players can be very diverse. Some might use an iPhone 6 with iOS and a Chrome browser, while others might use a Samsung phone running an old version of the Android OS and Firefox. Benjamin mentioned this aspect of testing in the interview with Colossi. "Just making sure that, 1: that it works on everybody's smartphones, because everyone has a different one." (Appendix L). As they mention later, the technical issues are especially important in the first part of the development process, and a big challenge to overcome (Appendix L). When developing the prototype *Put a Smile On*, testing with a variety of devices was a priority from the start. The website that shows an image of a mouth was first tested through self-testing. It was possible to test it with a few different devices and OS's, by borrowing phones and using the test devices at hand:

> First I did this by testing on my laptops browser. Later, I used my smartphone, a android OnePlus2 which has a rather big screen, and my old iPhone 4s, that has a smaller screen and a safari browser, to try it out and see if it worked. When it worked on those, I got some of the other jammers to test out as well from their phones. (Appendix B.1)

Of course this did not mean that every device, OS and browser, or combination of those, was tested but it meant that the biggest issues were highlighted and solved. When testing at the Global Game Jam Presentation, the website worked on most phones. But other issues related to smartphones, that were not tested while self-testing, came to light:

> After the game had finished, one person told me that they played most of the game without having color on their smartphone. If you don't interact with them, most smartphones will automatically shut off after a little while. (Appendix B.2)

The standby functions on the smartphones varied from device to device. While self-testing, this was not something that was taken into account. Only when players were playing the game for its whole duration with their own smartphones, this problem became clear.

On a technical level, an aspect to test specifically for crowd games, is how the functionality works when many players play at the same time. From a technical angle, this is especially important in regards to games that depend on networking. When observing *Enurendo* at the Nordic Game Jam award show, it was clear that the functions became poorer when a lot of people joined the game. The game started lagging, making the feedback slow, ending out with the functionality suffering:

> Once 150 players had joined, the server crashed, but already before that the game was lagging, apparently because of all of the 'rigidbodies and colliders' according to Max. (see Appendix I.1)

This was done in a crowd setting, but might not have needed a crowd to be tested. There are several ways this could have been foreseen and possibly avoided, had the correct measures of testing been taken. The lag that occurred from "rigidbodies and colliders", is purely happening in the software, and has nothing to do with the networking of the game. This could have been tested by adding 150 characters to the game, but without having to connect players to them over the network. Just dropping them into the game, the developers would be able to check for lag. There is also the possibility of adding a very simple AI to each character, making it possible to test whether the movement of 150 characters would cause any problems. Any problems caused by the networking could also have been tested, to accommodate these problems. Tools like *Tsung* are used to stress test servers by simulating any number of users[44]. Games that players join over a network can therefore be tested using this type of software.

[44] *Tsung* is free software and can be found at http://tsung.erlang-projects.org/

7.2.7 Discussion of testing

One of the challenges regarding this research, is how to interpret the findings that are based on small scale observations of the prototypes. Most of the prototypes have only been played by a crowd of around 10-15 players. When playing *Colorave* at the Nordic Game Jam Pre Party, around 160 people got a glowstick, but not all of them were in the game (see Appendix I.1). Since this thesis defines crowd games as games that can be played by at least 50 players, one could argue that this number of players is not sufficient in researching crowd games. Designers with experience regarding the testing of crowd games, are the developers Bryan and Benjamin from Colossi. When interviewed for this thesis, Benjamin shared his experiences on what makes a significant numbers of players for testing their crowd games:

> It doesn't need to be like a large number of people, we don't need a hundred people, right. We just need enough to where... you know.. maybe you don't know everybody in the room and you know, everyone... you know, you are not like a group of like ten people, right? (Appendix L)

For them, the aspect of not knowing everybody in the room seems important for them, in order to make a testing situation that resembles the intended setting for the game. Benjamin again explains:

> when you know everybody else in the room the, the feeling is very very different. And so, trying to simulate the real world is much more possible where you don't know everybody and, and in turn you can kind of, you can kind of see a little bit better how people will react when they don't know everyone around them, when they are not comfortable. It's... It... Can we still get the same emotions out of, of... You know... Will they still cheer, will they still start yelling? (Appendix L)

For Colossi, the important part of not knowing everybody else seems to be, that they expect the players to behave differently in a crowd setting. But as these statements go, it is not only the size of the crowd that matters, but also the general context.

In this project, most of the prototypes are not taken any further than one formal playtest outside of our office. The argument for this, lies in the focus of this project. In a traditional iterative game design process, such as described in *Game Design Workshop* (Fullerton, 2008), playtesting is done as part of a iterative process. This means that the playtesting is done to inform the designers on issues that will later on be solved or improved based on the playtest. For this project however, the goal for playtesting is to gain insights, not to improve the prototype at hand, but rather to learn about the design of crowd games in general.

More elaborate and structured playtesting, including post-testing surveys or more structured interviews, might have increased the depth of the insights gained from theses playtests. On the other hand, those kinds of interviews take time both to prepare, conduct and evaluate. For this thesis the time was instead spend trying out more different games at the tests. At the LET'S PLAY playtest, the major issues became apparent quite clearly, but of course there might be things that we missed. At the Colorave playtest at the All The Rave party, it was not possible to talk to all of the participants afterwards, since for them, this was perceived as part of the musical acts for the night, and not as a playtest. When observing the games being played by a crowd, there was a lot going on. While observation of a crowd made it possible to see what was going on in the crowd as a whole, it was hard to see how the individual player perceived the game. What could have been done was to conduct post-playtest interviews with a few of the participants, to gather insights about their experiences with the game. Another approach could have been to focus on observing just one or two players, but this would require getting help with the observation, since the game itself also required attention in order to be run.

As presented in this section, there are ways of testing crowd games that do not involve a crowd playtest. Though crowd playtesting might be suitable for testing some aspects of the design, small-scale playtesting and self-testing have other advantages. Smaller playtests make it possible for quicker iterations and make it possible to gain in-depth understandings of the testers' experiences. This might also be possible in crowd setting though, if the testing includes interviews or similar methods to let the designers get more focused feedback. The crowd playtest setting on the other hand is not necessarily as easy to get to test in, but can give designers insights on how the game is being played on a bigger scale, and whether it is still played like they intended it to be played. The setting of the crowd playtesting also makes issues related to the context become apparent, like it was seen with the camera placement. As with any other kind of testing, it is relevant to consider what parts of design that need to be tested in order to determine what kind of test would fit that.

7.3 Game host

One main element that separates crowd games from other types of games, is that crowd games can have a game host (as introduced in Section 1.2.1): a person who helps run the game. In this section it is explored what the role of the game host can be. It is discussed how the game host can be used both as part of testing and designing crowd games, but also how the game host can be part of the game that needs to be designed as well as the rest of the game system.

7.3.1 Calibration and playtesting

A problem with scaling up the games, is that it can be hard to predict how the system and the players will behave. With singleplayer games it is possible to have a controlled test environment, which closely resembles the context in which the game will actually be played, but with a crowd game, you can not test having 100 players playing the game before you actually let it out to 100 real players. A solution to this is giving the game host a way of controlling the game while it is played.

When designing a crowd game, the system reacts to inputs or conditions, and those can either be controlled by the crowd, be embedded in the system itself, or be controlled by a game host. When the events of the system are executed based on the crowd input, the players are in control of the events. An example of that is the air flow in *BlowIT*, where the players need feedback so they feel that they are changing the force of the air stream. The crowd controlling the system can be useful when designing interactions that the crowd needs to feel that they are in control of. Another way of controlling events within the game system, could be to embed the conditions for executing events into the system itself. An example of this is in the prototype *S.T.A.C.K.*, where the round ends after a certain time has elapsed. It also counts the score and executes an animation of winning and losing players' names based on this. The strengths of this kind of control is automation. If the program is written correctly, the system can quickly and accurately count, calculate and execute. This can be done rather easily by coding the system, and would be hard for a human to replicate live. Also, the code only has to be written once, and will then work accurately for how ever many rounds that will be played. The downside of embedding the control within the system itself, is that the system depends on data. Points and scoring are rather simple kinds of data, but figuring out if a crowd is ready to play or if they need to be challenged with a harder level, can be more complex. Also, a computer is rather rigid. It cannot suddenly change if the context changes. To interpret these kind of conditions, a game host might be better suited.

In the first prototype, *Jump On Heads*, the game host only controls when to start the game. All other controls are embedded directly in the system. When the game starts, players can add themselves by joining via the *HappyFunTimes* system.

In *Put A Smile On*, controls for adjusting sensitivity of the camera input were implemented. In that way, it it still the system that handles the input, but the game host has the ability to tweak how the inputs should be interpreted. When starting the game, it takes some time for the players

to find their smartphones, go to the website, turn up the screen brightness etc. It would be rather complex to design a system that could determine if all the players that wanted to and were able to play, were ready for the game to start. The game host therefore controls when to start the game. The game host also controls when to transition from one scene to another in the initial state of the game, from welcome screen, to information on where to find the mouth, and to the game start. This is again because these transitions depend on reading the crowd, answering to the question "are people ready to play"? Still, whenever the actual game is initiated, the system does all the work. In *BlowIT*, the game host again controls the sensitivity and the initiation of the game. In this prototype there is also a mini level before the game starts, that works as a playground or tutorial. The host can then end that level, when he considers the crowd to understand the concepts of how to play.

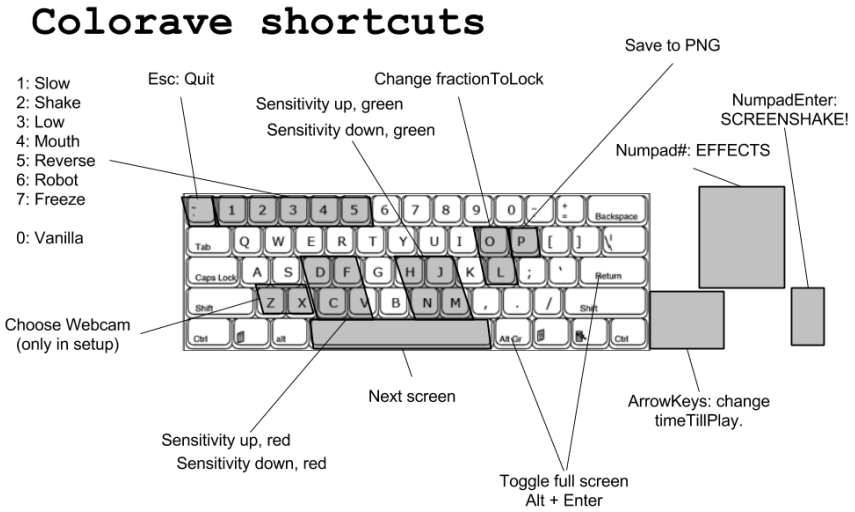

Figure 37: An image of the keyboard layout used by the game host to control *Colorave*.

In *Colorave*, the game host controls an array of events. As in the previous prototypes, he controls the initiation of the game and can calibrate the setting for the inputs so it suits the current setting. But what is different about this prototype, is that he also controls when to initiate levels and has some control over the feedback during the game. As seen in Figure 37 the game host can press the numpad enter key to enable screen shake. There is no screen shake executed by the system except for the one the game host initiates. Another crucial feedback and control of *Colorave* is the music and sound effects. The sound effects in *Colorave* are divided into automated audio and live performed audio. The automated audio consists of sounds for collecting and adding up points. This again depends on accuracy and efficiency, and not on reading the mood of the crowd. All the non-automated sounds are performed by a second game host — a DJ — during the game. This includes the music, which is crucial to the game, since it is a party game where people should be dancing. In a way, the different audio tracks also work as levels. They are designed to fit the rules that are added on top of the game for each round. The DJ will execute the different music tracks to indicate how the players should now behave. Another part of the audio that the DJ controls, is an air horn sound effect and numerous small samples that the DJ can add on top of the music. This is all to give the DJ a possibility to react dynamically to the crowd and to the crowd's behavior and mood.

In the hypertalk at A MAZE., Simon performed as a game host, while Anne was doing the talk. The game host had control over almost everything, from calibrating *THE WUU* (Headspider, 2015) to changing slides and getting items of a list to appear on the screen (see Appendix J). The talk ended out with a clapping meter, visualising the volume of the crowd's clapping, going from good to epic. The game host had control to just get the meter to the highest parameter if needed.

7.3.2 Designing for the game host

Figure 38: Screenshot from the prototype *Put a Smile On*. The screen shown is one of the introductory screens which teach the players how to play the game. The screen has the text "Press [Space]", even though the space bar is only meant to be used by the game host.

When it became apparent that a game host is an important aspect of crowd games, an important part of the design to iterate on, was how to integrate the role of the game host into the designs. As it can be seen in a screenshot from the *Put a Smile On* prototype in Figure 38, the UI is designed to communicate not only to the players but also to the game host. The "Press [Space]" at the start screen communicates directly to the game host and is not relevant to the players in the crowd. In the prototype made after *Put a Smile On*, *BlowIT*, these controls are hidden from the players. Instead, the game host just knows that he needs to press space to start the game. When designing *Colorave*, it became apparent that there were too many keyboard shortcuts for the game host to remember. A keyboard layout with the shortcuts written on (see Figure 37) was then designed to help the host keep track of the keys. In that way, the crowd and the game host had different interfaces. When designing the Hyper Talk, a keyboard layout was also made and printed for the game host. One thing that can be learned from this is, that crowd games have two different kinds of interfaces. Like in other types of games, it has an interface for the player. But what differentiates crowd games, is that they can also have an interface for the game host. For this thesis, the game host interfaces were a printed keyboard layout and a keyboard. But one could also imagine a separate screen with controls for the game host to use, similar to the separate view a presenter can have when doing a *PowerPoint* presentation as shown in Figure 39. With the presenter view in Microsoft *PowerPoint* the person presenting the slideshow has her own interface. This interface allows her to keep track of the time and the order of the slides and also take notes. It also allows for the presenter to draw on the slides live while presenting.

As this section shows, the role of the game host is a whole area of design, in which design decisions can be taken to enhance the role of the game host and aid the goal of the overall game design. Since game hosts are not a integrated part of other kinds of games, this is another area that is unexplored in previous research. Future projects exploring the possibilities of how to use the game host as an active part of the game might be able to expand the understanding of the role even further.

7.4 The play community

Since crowd games are played locally, the players are gathered in physical space. But this physical aspect of crowd games is little explored in most of the existing crowd games. In this section some possibilities for physical and social interactions in crowd games are examined. This is done by involving Bernard De Kovens theory on *the well-played game* and *the play community* from the

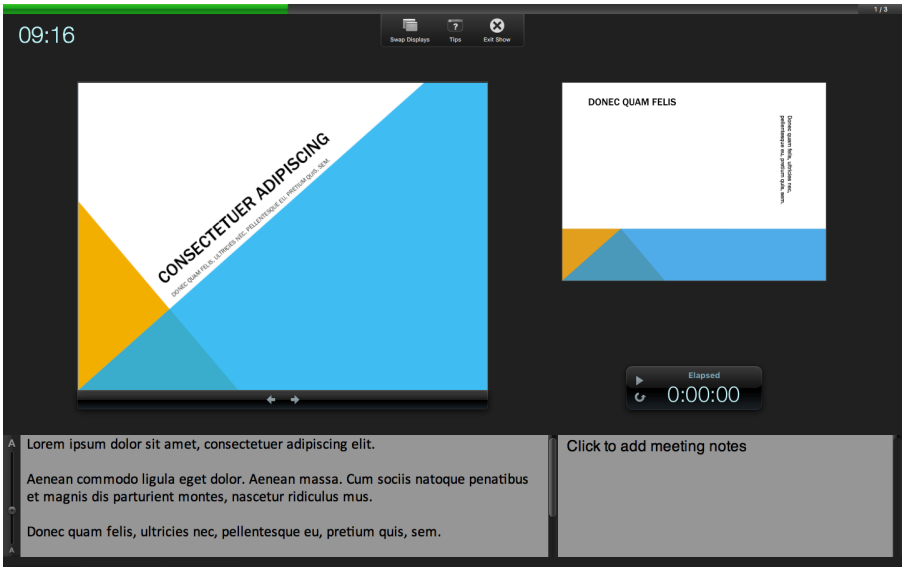

Figure 39: A screenshot from the interface when using the presentation mode in *PowerPoint*.

book *The Well-Played Game* (Koven, 2013). Furthermore, the article *Brutally unfair tactics totally ok now: On self-effacing games and unachievements* (Wilson, 2011) and *Wii are out of Control: Bodies, Game Screens and the Production of Gestural Excess* (Simon, 2009) are used in order to examine the use of physical and social interaction and games as performative acts in crowd game design.

A thing that became apparent when researching the games on the list of crow games (see Figure 5), was that most of the crowd games do not specifically encourage the players to interact with each other. Neither do any of them encourage the players to move around in physical space. Instead, in almost all of the games on the list, the players interact directly with the system and not each other. In *Happy Hockey* (Pieper and Kristmann, 2015) the players control the characters by using their smartphone as controller. In *Renga* (WallFour, 2012), the players point laser pointers at the screen, and in many of the other games, the smartphone is used as a traditional controller would be used. Interaction between the players in the physical space is not required in order to play the games.

An exception to this is the game *Sentree* (Glitchnap, n.d.) for phones and tablet. In *Sentree*, the protagonist hero is shooting enemies in a dark forest. Only, the player of this hero is blindfolded in real life and has to rely on her helpers, holding another device, to guide her. The guiding is meant to happen by talking to the player and therefore requires interaction between players in the physical space. In some of the crowd games, the players interact with each other's characters, like in *Happy Hockey*, where players can stun other players' characters by having their character attack with the hockey stick. But the interaction stays inside the digital part of the game, and again, it is not designed to be taken in to the physical space at any point. But just because a game is not designed to enable a certain type of physical or social contact, it does not mean that it will not happen. In the trailer for *Sentree*, a player is seen physically grabbing the hero player and moving her around instead of just speaking to her[45]. The game does not suggest these kind of behaviors to the players, but it does not restrict them from these behaviors either. The behaviors emerge because the players are free of the system. According to Wilson, this is also the strength of games like *B.U.T.T.O.N.*:

> The game's defining characteristic is the "incompleteness" of its underlying system, in the sense that it is so obviously up to the players themselves to interpret and enforce

[45]The trailer for *Sentree* can be found at `https://youtu.be/_CeKD4AOft8`, with the described moment being at 0:45.

Figure 40: A photo of the Hyper Talk on crowd games at AMAZE 2016. Simon (center) is serving as game host, controlling elements of the game and the graphic elements while Anne (right) is presenting.

> the rules. (Wilson, 2011, p. 5)

It is obvious to the players that the game is not able to detect whether the rules are obeyed. And it is not clear-cut how the game should be played, leaving space for the players to interpret the rules.

But most of the time, the opposite behavior is seen for games. Players tend to play strict to the rules, not challenging the system. If the players were really just in the game for winning it, the hero in *Sentree* would take off her blindfold and look at the screen. In *Happy Hockey*, players could grab the controller out of their opponents' hands, leaving them no way to control their character. The reason players usually do not act this way, can be explained by Bernard De Koven's concept of the well played game.

7.4.1 The well played game

In the book *The Well-Played Game* (Koven, 2013) game designer Bernard De Koven describes games as an activity that provides players with a common goal. He also emphasizes that games exist because of the continuous act of creating the game:

> I think of games as social fictions, performances, like works of art which exist only as long as they are continuously created. (Koven, 2013, p. xxii)

While De Koven describes games as lifelike, he also states that they are not real. The same way, he defines play as an activity that is not real. Even though one might be evil or rich through play, that does not mean that that person is in fact either evil or rich. As De Koven writes, play is "intended to be without consequence." (Koven, 2013, p. xxiv). One can play fight without getting hurt for real.

In *The Well-Played Game* (Koven, 2013), Bernard De Koven explores what it means to play a game well, and how players accomplish that:

> The success of well played game is not determined by who wins but that they can actually play well together. Playing well, the quality, is the real accomplishment. (Koven, 2013, p. 5)

By "well-played", De Koven refers not to the game, but to the way it is played (Koven, 2013, p. 7). This explains why the hero in *Sentree* does not take off her blindfold; even though it would make her win, it would not have been "playing well together". But when another player grabs the hero instead of just talking to her, they still both get to play the game. But whether it is well played, can be examined using De Kovens guidelines. In order to play "the well played game", De Koven states, there are certain guidelines that have to be followed: Players need the intention to play, the willingness to play, safety, trust, familiarity, and conventions (Koven, 2013, p. 7). Establishing the intention to play is something that is done continuously throughout the game, it is not enough to do it when the game starts. Players get caught up in the game and can forget why they were there in the first place. As the game or as the players' needs shift, the intentions to play well together have to be re-established (Koven, 2013, p. 7). The willingness to play is important for the well played game as well. Without the willingness, the players would not play and could not play well (Koven, 2013, p. 8). Connected to willingness, is the need for safety. Players need to be assured that they are not risking too much by playing (Koven, 2013, p. 8). Another aspect of this is trust. Players need to trust each other to be safe to play with (Koven, 2013, p. 8). The basis of this trust is familiarity. If not familiarity to the other players, then at least to the game (Koven, 2013, p. 9). Conventions are the specific conventions for the play community, and will be described further later on in this section. As it can be seen from the above description of the guidelines for the play community, they are all connected and related to each other. For the sake of exploring crowd games, the concepts of familiarity and conventions will be explored further.

When playing crowd games, the familiarity is hard to obtain in regards to the other players. A player of a crowd game rarely knows the majority of the other players. This suggest that the familiarity should be obtained another way, in order for the players to be able to play well. Some crowd games have the familiarity of using inputs that seem familiar to the players. Players using their own smartphones as controllers, with controls that resemble those they know from consoles and computers. An example of this is the controller from *Jump On Heads* that has a button on the right side, and left and right arrows on the left side (see Figure 41). A traditional physical controller, here inspired by the NES controller, which also has buttons on the right side, and arrows on the left (see Figure 42). Though the NES controller has more buttons than the smartphone controller, the similarities are clear.

Figure 41: The controller from *Jump On Heads* imitates a traditional physical controller.

Figure 42: A NES controller for comparison. (Evan-Amos / Wikimedia Commons)

Another way crowd games achieve familiarity, is by having gameplay that resembles games familiar to the player. For example, *Jump on Heads* does this by using the same interaction as a standard 2D platformer. But in crowd games that do not resemble traditional games, one might need other measures to obtain the familiarity for the players. In *Colorave*, the game is not similar to any widely known games, neither is the way of interacting with the game. Instead, the setting of the game is familiar. A DJ playing music, an MC directing the crowd, combined with glowsticks. It

resembles a night on the dancefloor or an evening at a concert, and in this context many players know what to do. Even though people have different starting points for how to act on a dance floor, there is still a common understanding of what is expected.

De Koven introduces the concept of the play community, the community of players and those players conventions for how to act in this community. The play community is a safe space where players can play together, with a mutual understanding of insisting on the play community conventions. The conventions of the play community could be that you play nicer towards a younger or less experienced player or stop the game when someone gets hurt (Koven, 2013, p. 11). What happens with some games is that the game gets prioritized before the play community, and it turns into a game community instead. In a play community, the willingness to play is what binds the players together, while for a game community, winning the game is the focus. In a play community there has to be a common understanding, that some things are more important than playing the game, a limit to the game. De Koven exemplifies this with children playing: "When someone gets hurt, the game stops" (Koven, 2013, p.11). When the players at the LET'S PLAY event played *S.T.A.C.K.*, this limit became apparent:

> And when someone accomplished to knock a phone down from the table and on to the floor, the room just went deadly silent, until the owner revealed that the phone was not broken, and people started laughing about the tension and sudden seriousness. (Appendix F.7)

A broken smartphone was clearly not a sacrifice someone was willing to make in order to play the game, neither the owner of the phone, the other players, or the hosts of the event. While *S.T.A.C.K.* was made as an attempt to get people to play physical and unfair like in *B.U.T.T.O.N.*, this was not what happened. As the example above shows, a big part of this might be found in the choice of controller. A more sturdy controller, designed for a bit of bashing around, might be better suited. Another option is to abandon the idea of using a controller completely and instead make the players fight over the physical space. As for example in *Colorave* where the players have to capture fields on the screen by waving the glowstick at the camera.

7.4.2 Unachivements

But players are willing to play for the sake of playing, and not primarily for the sake of winning. This could also be seen at the LET'S PLAY event. In the crowd hack of the game *One Button Bob*, a rule was added that players should gesture the move that the character was making in the game, even though the mediator only had one option to press a button no matter what. Still, the players in the LET'S PLAY session did the gestures with passion:

> When we asked people to act out what the player should do - for example throw or jump - all of the players acted along. When the character had to walk and then stand still at a certain time to avoid some obstacles, the crowd needed to show really precise when to stop. So when they stopped the walking, the stood completely still, almost like freezing, in stiff positions. It was fun to see them really embodying the character. (Appendix F.8)

To explain this behavior from the players, Bart Simon's article *Wii are out of control: Bodies, Game Screens and the Production of Gestural Excess* (Simon, 2009) provides some insights. He has a similar example, on players playing the game *Wii Tennis* which is part of the *Wii Sports* game (Nintendo, 2006). As Bart Simon observes, many players playing this game realize, that it is limited what kind gestures the Wii registers. To hit the tennis ball with power, a flick of the wrist is enough, but players still use the whole swing of the arm, playing like it was a real, physical match of tennis:

> In this way, the Wii game becomes an excuse for engaging in a different sort of bodily play, a kind of performative gestural excessiveness that the game-as-software neither demands nor acknowledges. (Simon, 2009, p. 12)

In the first test of *Colorave* at ScrollBar, the players quickly figured out where the camera was, and turned against the camera to play the game. Still, there seemed to be a limit to just how much the players would exploit the advantages of getting close to the camera:

> An interesting observation from Anne was that she saw one of the players cheat by going very close to the camera with their glowstick in one of the first rounds. This easily made green team win that round, however the behavior generally stopped after that round and was only repeated in one of the later rounds. (Appendix G.3)

Just as *Wii Tennis* can get boring and tedious when playing just by a flick of the wrist, *Colorave* is also a game that is designed to be played for the sake of bodily performing and playfullness between the players. This also became apparent when playing the crowd hacked *One Button Bob*. The players would be able to tell the mediator when to push the button much easier by just shouting or signaling with the arms, like playing the other crowd hacked games. But when asked to, they willingly perform the rather silly motions in order to play the game (see Appendix F.8). Adding rules like these might be able to expand the games, making them playful and engaging in another way than just the instrumental understanding of the game as a system:

> There is a kind of group-induced karaoke effect where the act of playing, singing or dancing becomes a performance for others that are as important as, or more important than, the gameplay as defined by the software. (Simon, 2009)

Another example on this could be seen in the *Colorave* prototype (see Appendix G). Here, each round had different rules on top of the game. In one round, players should stand completely still when the music stopped, in another round, they should have the glowstick in their mouth the entire round. All of these rules were not checked by the system, so if a player wanted to win, following those rules would not have been the most efficient way to do so. As Wilson mentions when describing his experiences with designing the game *B.U.T.T.O.N.*, there is a certain ambiguity in games with those types of rules that are not enforced by the system:

> The computer has no way of refereeing whether you took exactly six steps back, or if you did indeed spin around five times. That the players are collectively responsible for policing themselves only serves to exacerbate the ambiguity of the rules. How is a "step" measured? How slow do you need to be during the slow-mo round? These physical world actions are not so easy to systematize ad hoc. (Wilson, 2011, p. 5)

This element of a game, the gaps that are left for the players to interpret themselves, Wilson defines as "unachivements" (Wilson, 2011, p. 10). While a game like *B.U.T.T.O.N.* uses unachivements to the utmost extreme, *Colorave* does is more subtly. Freeze dance is for example rather binary, you either dance, or you stand still. But in a round like the robot round, asking players to "do the robot" (see Figure 43), it is not so clear what the task involves, and leaves more room for the players to interpret the rules.

In *Colorave* the DJ plays a big part in communicating this. The music is made up mainly of samples, and has a rather silly tone. A sound sample from an 1993 ad, warning kids not to put things in their mouth because of poisoning or choking hazards, saying "put it in your mouth". This sample is used during the round where the players are supposed to put the glowstick in their mouth[46]. These silly tunes are one of the ways the game communicates to the players, that this game is not to be taken too seriously, suggesting that the rules are not rigid.

[46]The commercial can be found on youtube under the name *Don't Put It In Your Mouth (Full Version, 1993)* https://www.youtube.com/watch?v=5AuLkMBAFZg

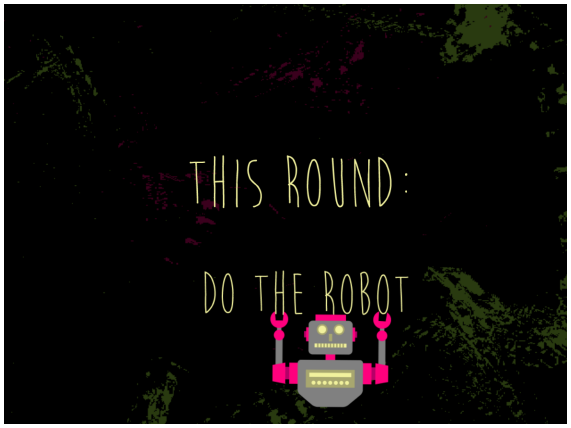

Figure 43: A screenshot from the prototype *Colorave* telling players to dance like a robot in the next round of the game.

7.4.3 Well-played crowd games

Bernard De Kovens starting point is physical games, and it is with these games in mind that *The Well-Played Game* (Koven, 2013) is written. But even though De Koven is not writing specifically about digital games, the concepts of the book are still relevant for digital games. When designing local multiplayer games, the players are assembled in physical space. We therefore find the concept of *The Well-Played Game* relevant to use when exploring the aspect of players playing crowd games together.

As part of experimenting with the possibilities within the space of crowd games, physical play and unachivements were used as parts of the design for the prototypes *Colorave* and *S.T.A.C.K.* This section suggests using unachivements and physical play in crowd game designs, as a way to expand the possibilities within the design space of crowd game designs.

This thesis has not explored how to establish the intention to play, the willingness to play, safety and trust for the players. These areas could be of interest for other research, researching crowd games with a more focused research on the social aspects of playing crowd games. Furthermore, to our knowledge, research with focus on the social aspects using theories like De Kovens is also deficient from the research into local multiplayer games in general, with Wilsons *Brutally unfair tactics totally ok now: On self-effacing games and unachievements* (Wilson, 2011) as the only example.

8 Conclusion

This thesis outlined the design space of crowd games, highlighting what characterizes the design space and distinguishes it from designing other types of games. Crowd games were in this thesis defined as local multiplayer games for 50 or more players. The first part of this thesis collected an overview of games and input technologies that exist in the current design space of crowd games. The overview of crowd games resulted in a list of crowd games (see Figure 5 in Section 4) that was used for the further research. In the overview of the available input technologies, it was described that inputs can have different modes, and that not all of them are used in crowd games as of yet.

The approach for the method of this project was constructive and therefore a number of prototypes were constructed in order to explore the design space of crowd games. The prototypes *Jump on Heads*, *Put A Smile On*, *BlowIT*, *S.T.A.C.K.*, and *Colorave* were made for this thesis, and they all use smartphones or cameras as input device.

In the analysis, design concepts specific to crowd games were identified. When exploring the input methods for crowd games it became apparent that inputs can be divided into heterogeneous and homogeneous inputs, depending on how the crowd inputs to the game system. This distinction is only apparent because the games are designed to be played by a crowd, and is a factor that can be taken into consideration when designing or analyzing crowd games. For homogeneous inputs, the concepts of observable inputs and mediator filtering are identified as design concepts. Neither is the concept of observable inputs relevant the design of single player and traditional local multiplayer games, but as demonstrated in this thesis, observable inputs can be an aspect to consider regarding crowd games. The use of mediators as input method that can be used for crowd games, also as a way of prototyping or testing crowd games. For heterogeneous inputs, one of the main design challenges is to give the players clear feedback. By comparing the prototypes for the thesis with both crowd games and traditional local multiplayer games, the key challenges of providing clear feedback in games with heterogeneous inputs were identified. The challenge is that the players have a hard time finding and tracking their characters, and possible solutions to this challenge were proposed, including real-time control, visual distinctions, spacing, and the movement speed of characters.

The thesis demonstrated how different ways of testing could be used to illuminate different aspects of a designed game. The key finding here is, that a crowd setting is not necessarily the most suitable setting for testing every aspect of crowd games. Some aspect of crowd games can be tested using small scale test settings.

Another feature unique to the crowd games design space that was identified in this thesis is the concept of a game host. As seen in this project, his role can be taken into consideration in many different aspects of the game design, also as part of the prototyping and testing process. The game host is a role that can be designed, just as with other aspects of the crowd game. In addition, an interface to control the game can be designed for the game host to use.

Furthermore, this thesis shows examples of how the concepts of the play community and unachievements can be used in designing crowd games. It demonstrates how this can be done to design for the local aspect of crowd games and highlights the possibility to research this possible expansion of the design space further.

This project was done with the goal of gaining a broad perspective on crowd games, using construction of prototypes as a main method. This has also resulted in a broad array of topics that has been touched upon. However, this means that none of the topics are investigated systematically and in-depth. If this project was done with a method that was more focused on the player's experience, using more focused interviews and observations of players, the thesis could have provided more clear answers to questions about what gives players certain experiences. Similarly, a method focused on systematically exploring technological challenges could have given more in-depth answers to what possibilities technologies and technological methods can afford. By having the research be explorative with construction as a core method, this thesis highlighted some areas of the design space to be developed further, and shed light on some of the game design challenges specifically related to designing crowd games.

9 Bibliography

Bregler, Christoph et al. (2005). "Squidball: An Experiment in Large-Scale Motion Capture and Game Design". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Ed. by Mark Maybury, Oliviero Stock, and Wolfgang Wahlster. Berlin, Heidelberg: Springer Berlin Heidelberg. Chap. Squidball: pp. 23–33.

Buxton, William (2007). *Sketching user experiences : getting the design right and the right design*. Amsterdam [et al.]: Morgan Kaufmann is an imprint of Elsevier.

Clausen, Anne Birgitte Jerichau and Simon. Stålhandske (2016). *Crowd Games list*. URL: https://goo.gl/blFYnQ (visited on 05/13/2016).

Digital Media Online (2016). *Digital Producer Magazine*. URL: http://www.digitalproducer.com/pages/starstruck.htm (visited on 05/19/2016).

Floyd, Christiane (1984). "A systematic look at prototyping". In: *Approaches to prototyping*, pp. 1–18.

Fullerton, Tracy (2008). *Game Design Workshop: A Playcentric Approach to Creating Innovative Games*, p. 496.

IDC (2016). *Smartphone OS Market Share, 2015 Q2*. URL: http://www.idc.com/prodserv/smartphone-os-market-share.jsp (visited on 05/16/2016).

Koskinen, Ilpo. et al. (2011). *Design research through practice: From the lab, field, and showroom*. Elsevier, p. 224.

Koven, Bernard De (2013). *The Well-Played Game: A Player's Philosophy*. London, England: The MIT Press, p. 148.

Lim, Youn-Kyung, Erik Stolterman, and Josh Tenenberg (2008). "The Anatomy of Prototypes: Prototypes as Filters, Prototypes as Manifestations of Design Ideas". In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 15.2, pp. 1–27.

Liu, Ming (2013). "A Study of Mobile Sensing Using Smartphones". In: *International Journal of Distributed Sensor Networks*.

Localmultiplayer.com (2014). *Localmultiplayer.com's big list of local multiplayer games released in 2014*. URL: http://bit.ly/localmulti2014 (visited on 05/13/2016).

Maynes-Aminzade, D., R. Pausch, and S. Seitz (2002). "Techniques for interactive audience participation". In: *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*.

Microsoft (2016). *Kinect*. URL: https://developer.microsoft.com/en-us/windows/kinect (visited on 05/15/2016).

Norman, Donald A. (2013). *The Design of Everyday Things*. The MIT Press.

PlayWest (2016). *Kachiku 64 website*. URL: http://playwe.st/games/kachiku-64/ (visited on 05/12/2016).

Salen, Katie and Eric Zimmerman (2004). *Rules of Play: Game Design Fundamentals*. MIT Press, p. 688.

Schön, Donald A. (1984). *The reflective practitioner : how professionals think in action*. Basic Books, p. 384.

Sicart, Miguel (2014). *Play Matters*. The MIT Press.

Simon, Bart (2009). "Wii are out of Control: Bodies, Game Screens and the Production of Gestural Excess". In: *Game Screens and the Production of Gestural Excess (March 5, 2009)*, pp. 1–27.

Swink, Steve (2009). *Game Feel*. CRC Press Taylor & Francis Group.

Tavares, Gregg (2016a). *HappyFunTimes Network*. URL: http://docs.happyfuntimes.net/docs/network.html (visited on 05/16/2016).

— (2016b). *Facebook AirConsole Comment*. URL: https://www.facebook.com/groups/timesfunhappy/permalink/1751042885128767/?comment%7B%5C_%7Did=1751217858444603%7B%5C&%7Dcomment%7B%5C_%7Dtracking=%7B%%7D7B%7B%%7D22tn%7B%%7D22%7B%%7D3A%7B%%7D22R0%7B%%7D22%7B%%7D7D (visited on 05/16/2016).

— (2016c). *Facebook HTTPS Post*. URL: https://www.facebook.com/groups/timesfunhappy/permalink/1718661358366920/ (visited on 05/16/2016).

Tavares, Gregg (2016d). *HappyFunTimes Docs*. URL: http://docs.happyfuntimes.net/docs/ (visited on 12/15/2015).
Waern, Annika and Jon Back (2015). "Experimental Game Design". In: *Game Research Methods: An Overview*. Ed. by Petri Lankoski and Staffan Björk. Chap. 20, pp. 341–354.
Wilson, D (2011). "Brutally unfair tactics totally ok now: On self-effacing games and unachievements". In: *Game Studies* 11.1.

10 Ludography

AirConsole (2015). *PolyRacer*.
Brough, Michael (2014). *868-HACK*.
Clements, Brett and Phillip Tanner (2004). *Atmosfear: The Gatekeeper*.
Futomo, Ojiro (2015). *Downwell*.
Garbos, Tim (2013). *WOOORRK!*
Glass Knuckle Games (2014). *Thief Town*.
Glitchnap (n.d.). *Sentree*.
Half Past Yellow and friends (2016). *Enurendo*.
Headspider (2015). *THE WUUUUUUUUUUUU*.
Incandescent Workshop LLC (2013). *Artemis Spaceship Bridge Simulator*.
KnapNok Games (2011). *B.U.T.T.O.N.*
London Studio (2014). *SingStar*.
Ludosity (2015). *Progress*.
New World Computing (1999). *Heroes of Might and Magic III*.
Nintendo (1985). *Super Mario Bros.*
— (1996). *Mario Kart 64*.
— (2006). *Wii Sports*.
Ojala, Otto-Ville (2014). *Soccer Physics*.
Pieper, Alexander and Johannes Kristmann (2015). *Happy Hockey*.
PlayWest (n.d.). *Kachiku 64*.
Rare (1997). *GoldenEye 007*.
— (2000). *Perfect Dark*.
Spragg, Adam (2014). *Hidden in Plain Sight*.
Stegersaurus Software Inc. (2014). *Mount Your Friends*.
Supersonic Software and Codemasters (1994). *Micro Machines 2: Turbo Tournament*.
Tavares, Gregg (2014). *Tonde Iko*.
Thorson, Matt (2013). *TowerFall*.
Triband (n.d.). *That Selfie Game*.
WallFour (2012). *Renga*.

A Jump on Heads

A.1 Design journals

Design journal, Simon

At autumn Exile Jam of 2015, we decided to make a crowd game using HappyFunTimes in one of the preliminary days before the main jam. At this point we already knew that we wanted to write about crowd games in our thesis, but we had little idea except for that.

We announced that we would work on a game using HappyFunTimes and collected a small team of people who wanted to co-design and do art. Simon would be doing the coding. After brief discussion on what we could do, we decided on building on top of one of the demo scenes that came with that version of HappyFunTimes. It was a simple platformer example where players would spawn into a world in which they could walk around. There was no gameplay at this point but it was possible to jump onto each others heads in order to reach higher parts of the level and we decided to use this relationship between the player characters as a core part of the design.

In the final game. You spawned into the world as a character on one of two teams. You then had to jump on the heads of the other characters to reach a cake on a high platform. Once a player reached the cake, the team of that player won and the round ended. A very simple game.

We wanted the game to fit any number of players, so we had the height of the cake platform vary dynamically depending on the number of players logged in. For each two more players, the cake would rise the height of a player.

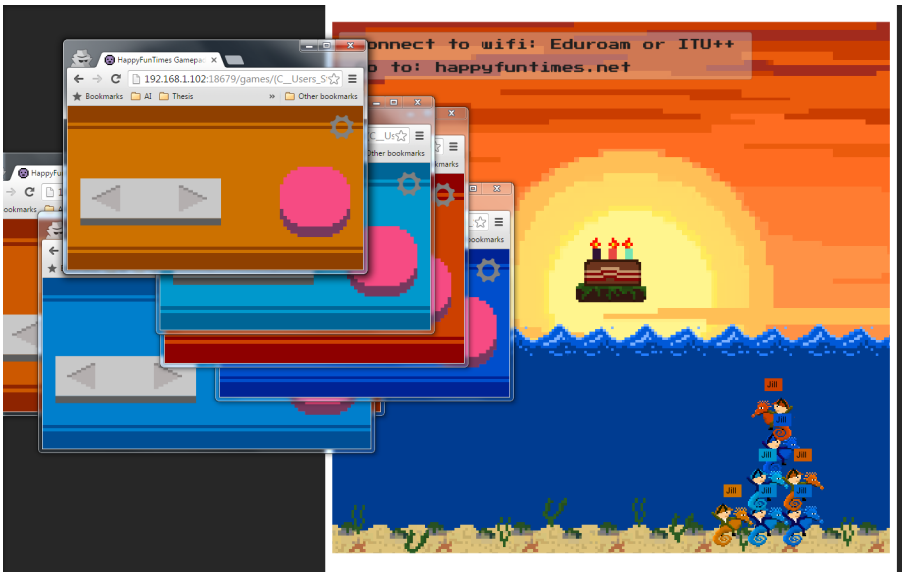

Figure 44: A view of Jump on Heads during testing. Multiple browser windows are open in order to emulate players.

For the players to easily recognize their character on the screen, we split the two teams into red and blue and then gave each player a slightly different shade of that color, so they would be able to tell the difference. The difference in color can be seen in Figure 44

For ease of instruction, we added a small text to the screen where it instructed players on which WiFi to connect to and which website to go to to join the game. This meant that we could set up the game and just leave it on and watch people join and play.

We did a minimum of internal testing, basically only to confirm that the tech was working and to adjust speed, jump height etc. This can also be seen in Figure 44. HappyFunTimes made it easy

to test what it is like to have many players in the game. Unfortunately I could only control one character at a time. From the test project that we used, there was however implemented controls so it was possible to control all of the characters at the same time by using the keyboard.

Design journal, Anne

I was also working on other projects at the game jam, so I were only present for the first initial brainstorming and testing, and then again at the presentation where it was played by the crowd.

When testing the game during the beginning of the design process, we focused on designing the game so that the players would need to cooperate. We wanted them to jump on each other in order to reach the goal, the cake in the top. We wanted the players to experience that they got rewarded for cooperating. When we tried out the first early prototype, playing it ourselves, it seemed like it was a interesting way of playing. It was challenging to work together, but it also felt satisfying when we managed to jump more avatar together on top of each other. But we had no idea how it would feel with more players in the game, since we were only a handful of people designing the game.

From then on, I left the development process to work on other game jam projects, and did not see the game before the final presentation.

A.2 Playtest at Exile Game Jam

Observation journal, Simon

We set up a TV so people could try and play the game in the large gym.

It was rather easy to gather people around the screen. Once the first players had joined, more quickly gathered around. People logged in just fine and were quickly able to join the game and get the idea. Comments while playing were that the game was too fast and that people had a hard time finding their character on the screen, especially since the teams changed with every restart of the game. Because of the intended gameplay of jumping on top of one another, the characters ended up clumping together a lot, meaning that they were standing very close together. A comment from one of the players was that the virtual controller on the smartphone was hard to use while looking away from the smartphone screen. At this point about 8-10 players were playing the game.

Rounds ended quickly and people were not communicating a lot, except for when cheering over a victory.

We later set up the game in the large meeting room where everyone was gathered. Here, so many players logged on that the cake (the goal) floated all the way off the screen and the players were unable to get to it at all.

Observation journal, Anne

I watched the game being played during the presentation at Exile Game Jam. There were working WiFi in the room, and a lot of people managed to login. The game quickly became crowded when the avatars started appearing on the screen. Every time a character appeared on screen, the cake got higher up in the air. Quickly, it ended up outside the screen. So Simon had to stop the game before anybody won, because it seemed quit impossible to reach the cake at that point.

While playing the game, many of the characters where just jumping and running around as what appeared as randomly. Maybe they were trying to figure out which character belonged to them. Or they just did not know what to do. The players did not manage to stack their characters on top of each other as we had hoped for. Once a new player wants to join a stack, she needs help from other players to do so. Either the bottom avatar needs to jump, or other avatars needs to

work as steps. The highest stack I saw were a stack of 3 players, where we had hoped for the players to be able to make high stacks so that they could reach the top.

B Put a Smile On

Put A Smile On is a game we made during Global Game Jam 2016. The players go to a website (http://djscoretrick.com/party/) which gives them a randomly assigned picture of a mouth in either red, green or blue. The colors respond to the 3 avatars on the screen, and what team you are on. The players then put the picture of the mouth up in front of their own mouth. Via a camera, the game uses the movements of the players as input. The game is a kind of freeze dance. The goal is to move as much as you can while the music plays, and freeze when the music stops.

B.1 Design journals

Design journal, Simon

For Global Game Jam, Anne and I had decided to make a crowd game of some sort, without actually having decided on what exactly to do.

Anne came up with the idea of using the smartphone as a add-on to a body part, in this case it would be the mouth, but it might as well have been an ear or a nose. The nice thing about using the mouth was that we could have everyone smiling while playing the game, no matter whether they were actually doing so or not.

It would have been straight forward to then send input from the phones to the computer using HappyFunTimes, but we were unsure whether the technology would work with everyone's phones at ITU, since many people might not be able to log onto the WiFi from their smartphones. We therefore came up with the idea of using a camera to register input. This also seemed fitting, since we were already having the players use their smartphones, and smartphones produce light.

Capturing light from smartphones, however, would be problematic unless all other light sources were turned off. We therefore decided to make it a a game about dancing at a party, which meant that we could justify having all lights turned off.

We liked having the webcam image of people show in the background of the game, so people could actually see themselves play the game. Testing the game on our own, we quickly realized that we had to mirror the screen in order for people to easily find themselves. Otherwise a player standing on the right, would be shown on the left side of the screen, which would be counter intuitive.

We also enhanced any color that the computer recognized as being a smile, and showed that clearly on the screen, graying out all other colors. This was another attempt to let players more easily find themselves and feel like they were actively participating.

For the gameplay itself, we took offset in freeze dance, a game known from children's birthday parties where you are supposed to stop dancing when the music stops. Players who keep dancing are out of the round. We made some slight modifications, making it a more fluid scoring system, where the team loses points if its players keep dancing while the music is off and gain points if they dance while the music is playing. A scoring system that would not have been possible in a folk game. We did not want to eliminate players, but we did want to punish them for not stopping.

Andreas made some nice music for the game. It intensifies as the teams get more points and ends up in a climax when a winner is found. This was made in an attempt to match the sound to the dramatic arc of the game.

One technical problem we hit very early on was that the webcams we had available automatically adjusted exposure and white balance. This meant that colors could easily look different from setting to setting and it was impossible for us to compensate for the settings that the camera

automatically set. The solution was to be as generous as possible when detecting colors, but ultimately this was not a very good solution because we were still unable to detect very bright and very dim screens. We knew this when finishing up the game.

We also knew that we would have to present the game to a crowd and play it with them, so we designed a number of introduction slides inside the game that would serve as visualizations for our introduction of the game to the crowd. This seemed easier than having either the entire explanation rest on our shoulders or on the shoulders of the game.

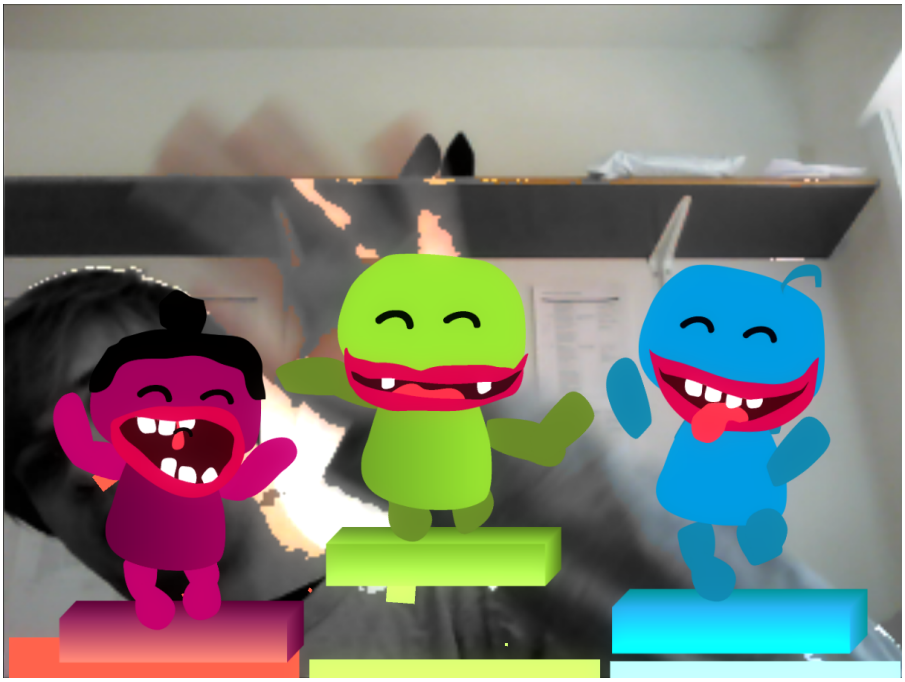

Figure 45: Testing the interaction in *Put a Smile On*. The white colored pixels show where change is detected.

During development, the game was tested several times. Most of the time, it would just be me sitting in front of my computer as seen in Figure 45, using movement of my body or the red backside of my Nexus 5 to emulate the motion of a dancing crowd. This gave us me an idea about whether the technology would work and how it was reacting, but since we would be playing with smartphones in the dark, we also had to do multiple tests in dark rooms we found at the ITU. The first room was completely dark, but very small. Here we could test the game under almost perfect conditions. Afterwards we found an abandoned classroom which looked more like the location for the final presentation. This room was brighter, larger, and had a projector. This meant that we could test distances from the camera and calibrate the game for this lighter environment.

Design journal, Anne

The theme for the game jam was rituals, so we started out by brainstorming about rituals. When brainstorming for ideas for this game, game designer Tim Garbos was hanging out with us. He suggested that we found a gimmick to design the game around - something that would make it stand out from the many game jam games. He also mentioned, that we should try and take the theme and make a twist on it, so we would not make a game about the same thing as all the others. So after brainstorming around tribe rituals, rites of passage and church traditions (like everybody else), we ended up brainstorming about the rituals around going out. Putting on make up and clothes before going to a party or going out. While designing the game, we came up with the idea of using the smartphone as a prop for the crowd to use. We liked the idea of the mouth

for people to look silly with, and toy around with, and thought it could be fun also outside of the game. This was our gimmick. I wrote "Put your worries behind you and put a smile on" on the brainstorm wall. A ideal scenario I imagined while designing, were that people would take selfies while "wearing" the mouth (however they would do that with their phone on their mouth).

When drawing the mouth, I first did it on the computer. I used a portrait picture of Simon in Illustrator, to put the smiles I had made on to see how it looked. When I have made a design I was happy about, I exported it and send it to my phone, where I could open it up and "try it on" myself and other people.

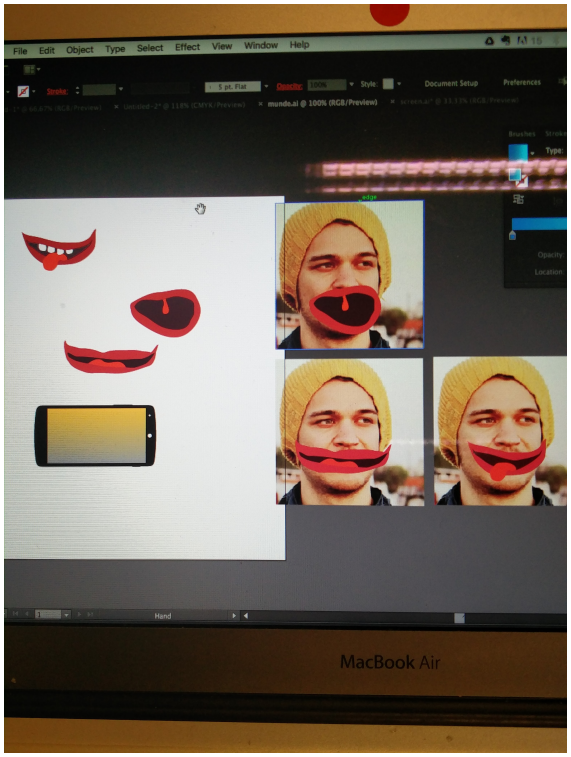

Figure 46: Testing out the mouth design in Illustrator on a photo of Simon

I spend some time making the website work properly with different browsers and just looking nice when you "wore" it. First I did this by testing on my laptops browser. Later, I used my smartphone, a android OnePlus2 which has a rather big screen, and my old iPhone 4s, that has a smaller screen and a safari browser, to try it out and see if it worked. When it worked on those, I got some of the other jammers to test out as well from their phones.

A challenge when designing the graphics were that they had to be not only illustrations but also communicating information to the camera. I therefore needed them to be clearly distinguished from each other by the camera. I choose the 3 colors, red, green, and blue, because they are easily distinguished by the camera. A purple, for example, could easily be confused as either blue or red by the camera. I also wanted the picture on the phone to be mostly just one color, since other colors could also make it hard for the camera to distinguish. I had tried with red mouths on a different colored background, but it was too hard for the camera and it actually ended up interpreting some of the green and blues as red. I therefor made the mouths and background same color, but varied them a little bit so they still had some contrast to each other.

I also made some graphics and animations for the UI to illustrate what to do with the mouth/smartphone when starting the game.

When we tested the game, we went into a windowless room and shut the lights down. Before the website with the mouth was up and running, we tested by opening a picture of a color on the

Figure 47: This is how the mouth ended out looking in the browser of a phone

phones. Later we could try with the actual site open in the browser, but the result were pretty much the same. What we learned by testing with the phones were that the rounds could not be too long, since the phones would go to standby mode. There were also a big difference if the phones light were turned all the way up or not. So we needed to instruct the crowd to turn up the lights on their phones, if they wanted an advantage.

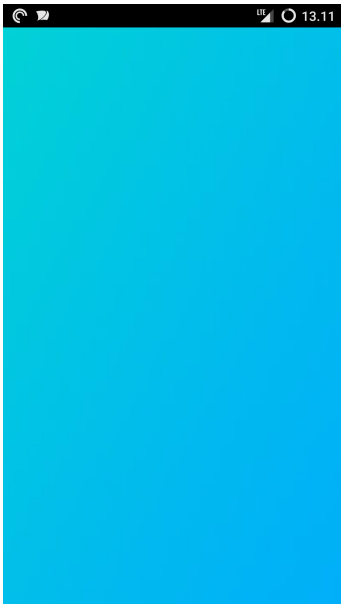

Figure 48: A screenshot from my smartphone for when we tested with a color we just googled in the browser.

Before the presentation of the game at the game jam, we were concerned whether it would be dark enough for the camera to actually track the mouths and not just take everything red, blue and green in as inputs. We made sure to go last for the presentations, and also to pull down the blackout blinds. But since we had no way of knowing how dark it would get at that time a day, we made a decision to make it possible to adjust the camera setting and the sensitivity before the game started.

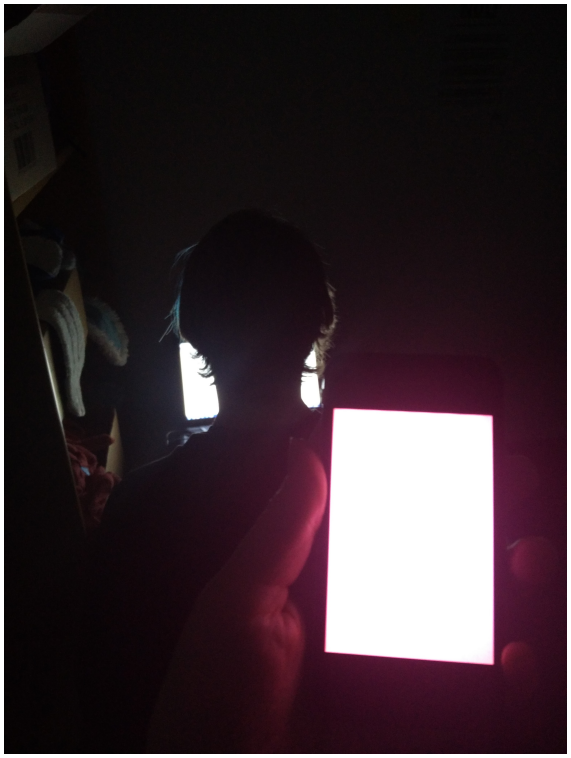

Figure 49: Testing in a dark room at ITU during daytime of the jam. On the picture it is my hand holding a phone and in front Simon is setting up the prototype on the laptop.

B.2 Global Game Jam presentations

Date :	31Jan2016
Location:	A big class room at ITU with people standing and sitting.
Crowd size:	$\sim$ 30 players
Context:	At the Global Game Jam presentations.
General atmosphere:	Eagerness to show off games. Tired after a long game jam.
Equipment & setup:	Webcam on tripod.

Observation journal, Simon

We had a couple of problems getting to set up our game. Misunderstandings between us and Horatiu. We finally got to do it though in front of the entire crowd at Global Game Jam, ITU. Our setup consisted of a webcam on a tripod, both from the ITU reception.

Talking about the webcam: This particular one auto adjust white balance and exposure, making it hard to recognize colors and light intensity. This means that a red colored screen for example can be seen as almost white by the camera, making it impossible for our software to detect color differences.

We placed the webcam up pretty high at the front of the room, for the best overview of the crowd. It was not possible to catch the entire crowd inside the field of view, but most were represented. We turned off the lights and started the game, explaining the rules as we went through the first slides of the game. When most people were ready with their smiles, we started the game. “When the music is playing you dance, when it stops you stop”.

While the game was playing, I helped the crowd with instructions on when to do what, at least for the first couple of times that the music played.

Something that we had already discussed showed its true nature as we played the game. Players in the back of the crowd had a much smaller influence on the game than those at the front. This is of course due to the fact that the game cares about the sizes of the color blobs it tracks and that the camera sees things in perspective.

Watching the crowd play, I realized how it must look from the players' perspectives. They don't see a sea of players with smiles on their mouthes, they see the back heads of other players and a screen, where they can barely recognize themselves.

After the game had finished, one person told me that they played most of the game without having color on their smartphone. If you don't interact with them, most smartphones will automatically shut off after a little while. Some might even quicker dim their screen, making it hard for the camera to detect them. Many of the players' smartphones also had different display brightness, making some smartphones look completely white to the camera and making others be too dark to be detected.

One hope we had for the game was that players would interact physically when playing, for example pushing players from the other teams when they had to stand still.

Talking to one of the players after the game, he mentioned that he had a hard time discerning whether he actually had an impact on the game.

Observation journal, Anne

When presenting the game, Simon was controlling the game while I was presenting. Since it was a game jam presentation, it seemed natural to talk about the game and instruct the players live, instead of using only on-screen instructions. When we played the game, I told the crowd to go to the website showed on the screen, and I demonstrated how to position the phone.

The team that won was green, and it was clear to us that a little too many players had the green mouth. This could have been avoided by making the website less random, so it would show equal number of each color mouth. After the presentation, one of the players told me that he had changed team during the game, because he saw that his team was losing, and wanted to be on the winning team.

Another problem was the lighting. We made sure to turn off the lights and get the blackout curtains down, but the light from the projector was super bright and reflected in the phones, which made the game unbalanced. Whenever a team was in the lead, the UI on the projector screen got more of the green color, leading to more green color being reflected.

People were not entirely sure when they had to dance and not. We saw people dancing when they shouldn't, and it did not look like it was on purpose or just because they were slow to react. People were dancing and laughing, but not a lot, and some even had more like a nervous laugh.

My reflections based on this observation:

The introduction of the game is important for how the players perceive the game. Who are the players? How to get people started.

People have not tried a crowd game before. Make sure to introduce them to how they should play it.

If we want people to be creative when approaching the game (like, BUTTON), how can we introduce the games so that will happen?

More time on UI! Gameplay is super simple, so make sure to spend time on getting people to play and understand the game.

How to make people feel they have influence on the game when in a crowd. With sound feedback you can easily recognize yours and others inputs to the game, makes it feels significant. In Put A Smile On, you can't see your own screen. But seeing the cam feedback also helps.

Gimmicks work! Do they only work in crowd games? Single-use games?

You don't collect a crowd in order to play the game. You have a crowd and therefor you play the game. Purpose of the game. Is it playing the game or to warm up to something else? It's super cool to make a game to make people play that normally wouldn't play.

Shared vs. individual inputs.

Crowd games need a game master! Guidelines for the game masters.

Do's & dont's, guidelines for game masters,

Multiple things made it hard for the players to realize whether they had an impact on the game or not.

- The input from many phones was ignored because they were either too bright, too dim or the camera's exposure and white balance were off.
- The input was not observable as in for example THE WUUUUUUUUUUUU
- The teams were spread everywhere on the floor, so it was hard to recognize who was on your team.

C Playtest, AirConsole

AirConsole is a virtual console on the internet. Setup is straight forward: go to the website on a computer and now players can join in by going to the same website on their smartphone. This gives players access to a slew of local multiplayer games which can be played using their smartphones as controllers. We thought we would test this to see how many players it would support. If we would get it to work, we could then observe players playing the games on AirConsole.

C.1 Observation journals

Date :	04Feb2016
Location:	A normal class room at ITU.
Crowd size:	$\sim$ 20-30 players
Context:	Game students at ITU during game development class.
General atmosphere:	Ready to listen to a lecture, eager to get started with projects.
Equipment & setup:	PC, Projector, and own smartphones. Projector regularly used for lectures in the classroom.

Observation journal, Simon

For game development class on this day we tried to set up some crowd games while the students were waiting for the other TA's to finish creating groups.

At first we just set up the AirConsole on a computer and allowed students to log in by providing them a link. Only one of the students (Andreas) joined on his own without being instructed to do so, and I played a game of PolyRacer with him (AirConsole, 2015).

After restarting the console, I asked the class to join in and about 10-20 people did so and someone accidentally chose a game Silly Run Valley, which only supports 1-8 players. The first 8 players got to choose characters but we weren't able to start the game. We were unsure why this was but suspected that the game had crashed because of too many players or that some players weren't properly connected, and that the game therefore had a condition not allowing it to start.

We reset the console one last time. Funnily enough, only player 1 is allowed to exit a game. In the future I will make sure to be player 1 so I can exit games and choose different ones.

This time we got the correct game chosen (PolyRacer) and got a lot of people connected (20+). The game wouldn't start.

After contacting AirConsole, they inform us that the technical maximum of players is 3000, but that PolyRacer only supports 32 different colors, and that we should try playing Cards Against Humanity.

After failing to get AirConsole to run, we set up HappyFunTimes, but only got one player connected. The computer running it was on eduroam, but other players on eduroam were still having trouble connecting. At this point we got cut off.

It later became apparent that we had problems with HappyFunTimes because TA's and students are not on the same network.

I thought, that we should have tested the technology beforehand, since that is testable.

D BlowIT

In BlowIT, the crowd is divided into 4 teams. Each team controls 1/4 vertically slice of the screen, where their avatar is placed in the top and bottom of the screen. A ball floats through the screen, and if it hits an avatar, the team loses one life. The last team alive wins the game. The players control the force of the vertical airstream within their part of the screen, the more they move, the more powerful the airstream gets.

D.1 Design journals

Design journal, Anne

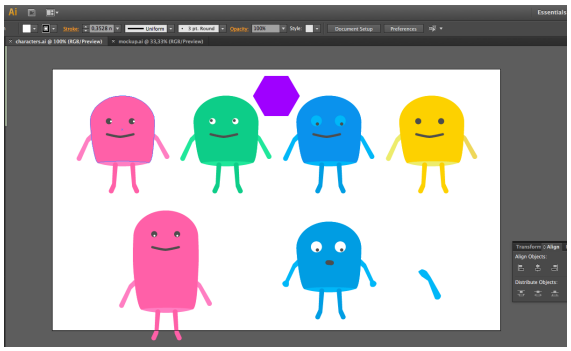

Figure 50: Iterations on the characters. They were made fast, and I choose a clean/minimal 2D style to make them look rather finished. Still, I wanted them to have a cute look, so you got the feeling that you should protect your character - while wanting to hurt the others. I ended up with the version with white eyeballs and a black dot in the middle, so that the dot could rotate according to where the ball was, making the characters look out for the ball. I also ended up making the mouth worried, so that it signaled that the characters were afraid of the ball

First thing we made a prototype that were basically just a toy. It was a screen where you should control the airs upwards force by moving your arms. By doing this, you can keep the ball in the air. We tried it out a bit with people passing by at the office and with each other. When we thought it felt nice, we iterated on the goal of the game. First, we tried so that you would just need the ball to not hit the bottom of the screen. But this way, it was really hard to play against each other. We just played defensive, avoiding getting the ball to hit the bottom. We also needed a way for the ball to move horizontally, since it was rather unfair if the ball kept staying in the

same place. So we introduced a air-stream going from left to right, moving the ball this direction. We also made it so that it was both the top and the bottom of the field that would make team lose points.

For the graphics, I made the characters alike but different colors. I made it so that you could get the feeling that it was the same character in the top and the bottom. But we did not have much time to iterate, since we had a deadline of showing it at the Demodag event the day after we started making the game. We spend a lot of time adjusting the particles so that it felt responsive when playing. The rest of the UI were just simple numbers adding up for every time you got hit, and last team standing was the winner.

Design Journal, Simon

Already having the technology from Put a Smile On, it seemed obvious to reuse that technology for another game. And this time, instead of using differing colors to divide the team, we were inspired by this Seat advertisement game that we had seen on the internet, and thought it'd be a good idea to divide the teams by position in space. We saw from Put a Smile On that players in the back counted less that people in the front. It would have been cool to compensate for that, so that people who were farther away weighted more than people close by. Anyway, we just decided to split the screen vertically instead. This way each team would have an equal number of players in the front and in the back.

So having the technology and having decided on how to divide the screen, we then needed a game to put on top of that. I think it was Anne who had the idea of keeping a ball afloat using the movement of each team. All movement would then cause a updraft which kept the ball afloat. This was pretty easy to implement with the technology we had. To keep the game interesting for all four team, we had the ball move constantly to the right and screen wrap to the other side when it hit the edge of the screen.

To make a scoring system, we gave each team three lives, and they would lose one if the ball hit the top or the bottom of the screen inside their team's vertical slice of the screen. From testing internally, we had discovered that the challenge lie in holding the ball between the top and the bottom of the screen and therefore chose to make this the scoring. We also discovered that it was possible to make the ball hit the top of the other players' screens. We liked that teams could be aggressive towards each other.

Anne made some graphics in the form of characters that would get hurt if the ball hit the top or bottom of the screen. A simple metaphor. We added a particle system to act as the updraft and to serve as feedback.

D.2 Playtest, BlowIT at Demodag

Date :	23Feb2016
Location:	The office of a startup company. Vestergade 20C, 2. About 10 rows of seats with 5 seats in each row. A rather deep room. The screen and wall were close to the frontmost row and the camera could therefore not be setup far enough away to see the entirety of the first row.
Crowd size:	$\sim$ 50 people
Context:	Startup tech people at a demoing event.
General atmosphere:	Some had a beer. Crowd seemed excited about Demodag. We were the last presentation so some people might be ready to leave for the day.
Equipment & setup:	Screen had a small size considering the depth of the room. Camera FOV was too narrow to see entire crowd.

Observation journal, Simon

For Demo Dag in Copenhagen, we signed up to present BlowIT. At this point we had spent one evening and one afternoon on the prototype and it looked a bit like this:

We were the last ones to present with a promise from the host that this would be "more fun" than the previous speakers.

The audience was sitting down, which meant the we didn't need to place the camera very high to be able to record everyone. On top of that, the webcam we had borrowed from ITU had a very narrow field of view, so we pretty quickly concluded that it would be better to use the built in webcam. The setup was therefore quite simple. We had my laptop placed on an upside down trashcan placed on a table. From there, the camera could see most of the crowd, although from a slight angle, since we could not place the computer in front of the projector.

Setting up was smooth except for a couple of glitches with the projector, but it went pretty fast. At the same time we presented ourselves and our thesis and the game and quickly tested that the camera worked as expected by starting the game shortly.

Explaining the game was quite awkward since we decided that the best way to do it was to start the game and then explain the teams and have each team try out the controls by waving their hands. In this way we sort of explained the game as they played it, just like you might do when teaching someone a board game. We had a "prøverunde" as you would say in Danish. Everyone in the crowd seemed to follow instructions and at least try to participate.

After the explanation we restarted the game and instructed that it was for real this time and joked that the winners would get free potato chips (which were free at the event anyway). The game was played, and the crowd was definitely doing a lot better than the first time around. Simon encouraged them as they played "wave more" etc. They were having a bit of a hard time getting the ball to stay in the air. Both Anne and Simon were watching the screen, and therefore paid minimal attention to the crowd. We could of course observe them through the screen, and many people seemed to be playing. Also, a lot of people laughed while playing.

In the end, green team won and we finished off with an applause.

By the end of Demo Dag, a couple of guys sitting behind us complimented our presentation but commented that they needed more feedback. In addition, they commented that it was difficult to sense what other people on their team were doing, since they had their backs turned towards them.

My reflections: Camera FOV too narrow. GoPro next time. The practice round did not go too well. Not enough feedback. Move detection sensitivity too low.

Observation journal, Anne

The talks before us were all more serious projects., everything from apps for healthcare professionals to live customer support plugins for websites. All of them very professionally looking and projects people have been working on for a long time. One developer told that he had been working on his project for the past year. We had only worked on our game since the day before, so we were a little nervous on how our very prototype looking game would be received. There were only one other game showcased, a singleplayer game that was not really playable yet. So when it was our turn, we were presented as the fun bit.

We did not have the opportunity to see the place before we got there, so we had no idea how the setting looked before the event started. The seats were really close to the screen and the area in front of the screen were we had to present were narrow - approximately 1 meter. We had brought a tripod to put the camera on, but the logistics of camera, laptop and cables was still a bit hard to figure out where to best place the webcam. We tried to set it up during the break, but the webcam was not capturing enough of the crowd, so we decided to use the laptop instead, placing it on top of a trashcan on some furniture.

After the trial round, we announced that it was "for real" this time, and some people got confused and asked if the thing they just did wasn't the game. I looked at the screen while the game was playing, and it looked like everybody was participating. People laughed, but even though they were participating, we did not see any super agressive moves or people yelling as their team almost lost.

E Soccer Physics, crowd hacked

Soccer Physics is a two player local multiplayer game by Otto-Ville Ojala(Ojala, 2014). Each player uses one button to control two characters who are players on a soccer team. Both characters on a team make a jumping motion whenever the player presses the button. The goal is to score on the other team. Rounds of play are usually very chaotic, with the characters jumping all over the court.

Anne was not part of the process for this game, and therefore there is only a design journal and an observation journal from Simon.

E.1 Design journal, Simon

The game *Sentree* by danish Glitchnap uses a split up gameplay where one person controls the avatar in the game and another person (or even a crowd of people) give instructions to the first player (Glitchnap, n.d.). In the Twitch phenomenon Twitch Plays Pokémon, thousands of users on the streaming service Twitch collectively play through the game Pokémon. In *Sentree*, the game is designed to be played even if you have a large group of people. In the latter case, the singleplayer game *Pokémon* was hacked in a way that enabled thousands of people to participate in the playthrough.

Knowing these two examples, it seemed obvious to use these games as inspiration on how to make a crowd play. Combining the two ideas, you get a singleplayer game, hacked using a mediater to translate input from a crowd to a conventional input device.

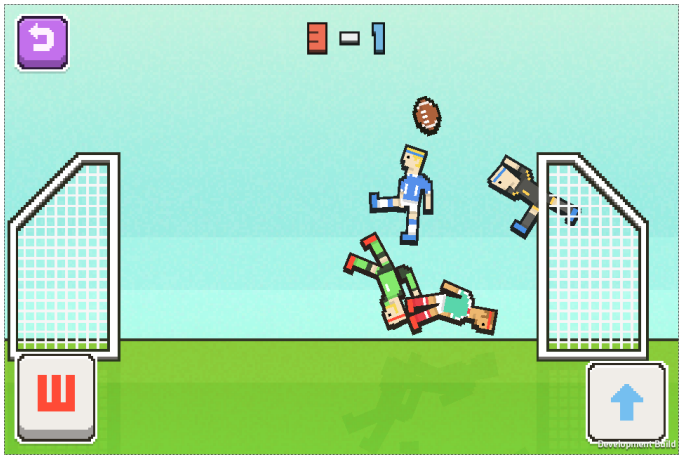

Figure 51: A screenshot of the game *Soccer Physics*

To test out this idea, *Soccer Physics* (see Figure 51) was extended to include a crowd by having the two players face away from the screen, towards the crowd. Only the crowd can see the screen and a clear physical gesture is established as communication between the crowd and the two player volunteers. Each player is teamed up with one half of the crowd. The players are called volunteers or mediators.

Our reason for using Soccer Physics was many-fold

- The input to the game is relatively simple; it only uses one button. This makes the interaction between the crowd and the mediator simple as well, and therefore lesser likely to mess up.
- The game is competitive between two players (or in this case two teams), which is different from what Sentree and Twitch Plays Pokémon do.
- Soccer Physics was the only competitive one-button game we knew at the time.

I found out that I could use an Xbox controller as input device by routing the inputs through Keysticks in the computer. This gives a button on the controller to each of the mediators, having them standing closely together. We thought this was a nice idea since it gives the game a little more physicality. We were eager to find out whether this had an effect on the way the game was played.

E.2 Testing crowd hacks at ITU

Date :	01Mar2016
Location:	Normal classroom at ITU. About 5 rows of seats with 10 seats in each row. Not quite enough room for everyone. The screen should be big enough for everyone to see what is going on.
Crowd size:	$\sim$ 50 people
Context:	Games students in class
General atmosphere:	Crowd a bit tired from listening to presentations for an hour. On the other hand they had just had a break and were ready to sit in for more presentations.
Equipment & setup:	Screen with fine visibility for the whole crowd. A shared Xbox controller for the two volunteers.

Observation journal, Simon

The mediators got the controller and were instructed which button to press. The crowd was instructed to raise their hands up to gesture to the mediator.

About 80%-90% of the crowd was participating in the game, making gestures to the volunteers. You could tell that people were tense and eager to give signal. What exactly

I interviewed Andreas, who was one of the volunteers. He noted that he thought that he probably had more fun than the people in the crowd, something I was relieved to hear since one concern was that being a volunteer would be boring.

I had a short interview with a person in the crowd, Lena, whom commented that she had "a lot of fun". She pointed out the problem that she could not feel/read the crowd very well because she was sitting in the front row. Therefore she felt less significant and could not tell exactly what the volunteer for her team was reacting on. Lena also said that it reminded her of a physical game where the crowd controls two NPC street fighters by using gestures. Street fighters would only react if the entire crowd was doing the gesture. A much more slow paced game.

Reflections: The nature of soccer physics is very chaotic. There is of course a clear win/lose condition, but the path to reaching that condition is not so clear. The crowd, however seemed to often agree on when to push the button and when not to. Especially some situations guided the crowd to be very coherent in their decisions. Interestingly enough, even coherent decisions in this game don't always lead to correct choices being made. It could be interesting to try with a game where correct choices and precision are much more important.

I (Simon) failed to instruct the two volunteers precisely on when to and when not to press the button. It is therefore important to try and ask them exactly what they acted on.

As Lena pointed out, the people at the front of the crowd were unable to sense the crowd behind them. Using an input such as sound might have helped in that regard.

F LET'S PLAY event

F.1 Design journals, before LET'S PLAY

Design journal, Anne

We decided to try out some more crowd hacked games. We wanted to explore the use of a mediator even more, and since it was fast and easy to crowd hack games, we just tried it out. We wanted to make some games that explored the mediator but with different gameplay. We went googleing for small singleplayer and two player local multiplayer games, that could suit this. We tried out a bunch of games, and ended out with One Button Bob, Canabalt and Molehammers. We choose One Button Bob because it was different actions the player should take every time.

We wanted a competitive game, where to players played against each other, so the crowd could be in teams. This was the same for Soccer Physics, bu there Simon had observer that it seemed kind of random who won, and we wanted to then choose a game that was less about chance and more about quick respond. We chose Mole Hammers, because it seemed like it had these qualities, and it was also rather simple for the crowd to understand. An animal shows up, and you have to hammer it in the head. That seems logic, from a game view at least. We also wanted a game where the players had to give a more complex input, something that was not either push or don't push, but could also be holding. We knew that Canabalt had these qualities, on top of also being a game that can be played rater simple. In Canabalt, you can focus on getting the maximal points, but you can also just play it to see how far you reach in the level. This was the way we intended the crowd to play it. We chose One Button Bob because it had a lot of different actions, and that they worked in different ways.

I thought it could be a good idea to make a controller for the mediator that was more visible to the crowd, so everybody could see when the button was pushed. I made two big buttons with MaKey MaKey. One button was an old metal hard disc case, the other some other piece of unidentified hardware trash - it looked like it was also a casing of sorts. I found these among other recycle materials in the Design Lab at ITU. The reason I chose these were that they are first of all super conductive. It is metal all around and I wanted to make sure that they would give an input instantly, and that the players should not hold it super tight or at a specific place for it to be conductive. The other ting was, that I wanted them to be rather big, so they were visible for the crowd players. Both pieces of hardware is big, about the size of the palm of a hand. Also, it is sturdy because of the material. I wanted the players to not be afraid of bashing it hard or quickly. When I was looking through the potential materials for buttons, I tried to bash it to see how it felt and sounded. What I found was also good about the hardware we ended out using, was that it had a nice sound to it. Both casings are a bit broken, so they make some noise when you bash them. I figures that this would make the buttons even more visible to the crowd - that they could actually hear when the button was pressed.

One of the biggest changes we did to BlowIT was that we made a tutorial/playground level before the game started. We wanted them to be able to play around with the game for a while before starting it, so they all understood how the air-stream and controlling it worked. We did this by putting up walls between the 4 teams areas inside the game. Then we made a ball for each team, so they could play around with it. We also adjusted some other stuff. We worked a bit on the feedback, so that the video feed would be colored where it detected motion.

Before designing S.T.A.C.K, we wrote the creators of B.U.T.T.O.N to make sure it was okay by them that we used their game for this. They agreed, commenting that it seemed like a fun project

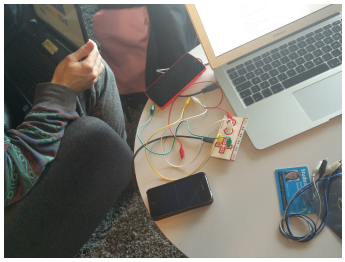

Figure 52: MaKey MaKey ready to be connected to a button

Figure 53: Cut out of a photo taken by the webcam during the LET'S PLAY event. In the bottom of the picture, under the laptop, you can see one of the big costume made buttons for the mediator.

but htat they were kind of concerned about using smartphones as controllers in such a physical game. While we shared their concern, we really wanted to do the experiment, and see exactly what part of it didn't work and possibly why.

When designing S.T.A.C.K, we tried to replicate the different aspects of B.U.T.T.O.N as closely as we could. We first played some rounds of B.U.T.T.O.N and during the game we discussed how we could translate it into a crowd game. I started out designing the algorithm to make commands and how the game should validate if the players succeeded or failed. One thing they do in B.U.T.T.O.N regarding the commands is that it is made so it is not super fast and easy to understand the commands. Instead of it saying "Do" og "Don't", it says commands like "The first player to push the button exactly 3 times win". The player has to decode this - both what the action is and if it is a good or a bad thing to complete the action.

Another design issue in B.U.T.T.O.N is the end of each round. In B.U.T.T.O.N it is made visible who won and lost in each round, illustrated by the characters. Since S.T.A.C.K should be made for a crowd, we instead made every player into a block with their name on it. This way, we could have all of the players represented on the screen at the same time.

Design Journal, Simon

After trying to crowd hack Soccer Physics and getting experience with that, it seemed obvious to try and crowd hack some other games. The design process was actually pretty straight forward. We went online and searched for one button games. We found a bunch and tested them out and decided which ones would suit the playtest well.

Anne decided to make a custom controller for the event. She used Makey Makey to wire up some metal cases as buttons, so the players didn't have to use an Xbox controller like they did in the Soccer Physics crowd hack. The intent was to make it feel different from a just a regular game and to make the inputs from the mediators super visible to the crowd.

Describing the games that we chose in turn:

Canabalt is all about precision. It's interesting to see how precise the input from the crowd can be.

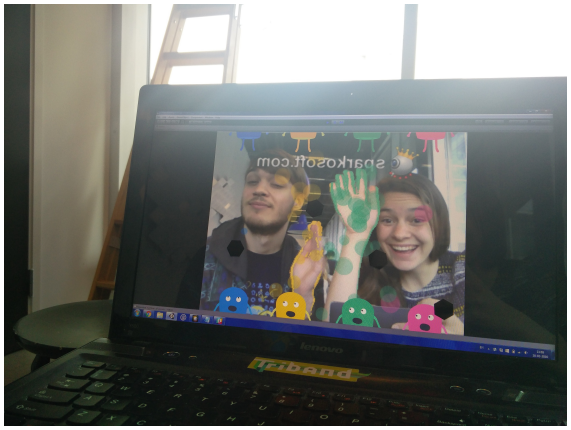

Figure 54: Prototyping coloring the areas that are in motion

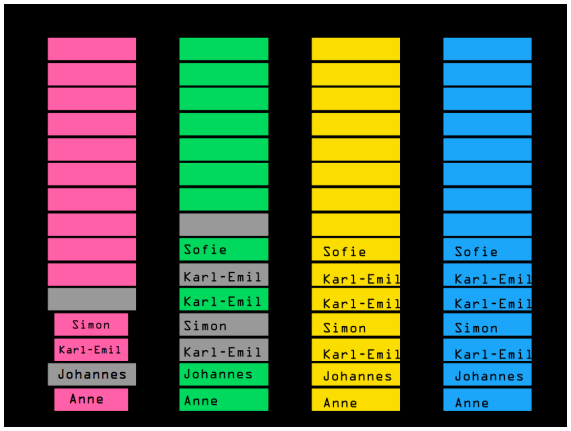

Figure 55: First iteration mockup on how the end of a round could look. Each player has a block with their name on it, and each team is a stack of those blocks. In the end of a round, the blocks of players that have failed turns grey and disapears, making them stacks hight correspond to the number of players who win the round.

Molehammers is all about being faster than the other team and not falling for traps. We liked having two teams play against each other, since that would be more fair than having the crowd play against the system, since the system is usually tweaked to match a single player rather than an entire crowd.

One Button Bob is just pure genius. We could have the players acting out the actions for each level, which would add some fun mimicry into the game. The challenges also start out pretty easy, so we decided that the crowd could probably get through at least a couple of levels. Also it got to test out a bunch of different methods of input, which could potentially be super useful.

We also made some changes to BlowIT based on our observations at Demo Dag. First off we wanted to make a tutorial screen to take a little explaining off of the game hosts shoulders. We made this by putting up walls between the teams in the beginning of the game and giving them a ball each, as can be seen in Figure 57. The idea is that players can get familiar with the mechanics of the game in a safe environment before the game begins. The game host can then use the space bar to move on the real game.

We also added extra feedback to players in the form of colored pixels where the game detects motion. The color depends on what team - aka which vertical slice - the motion is detected in as shown in the screenshot of Figure 58.

Finally we added some more feedback when a team loses a point. We felt like this wasn't clear

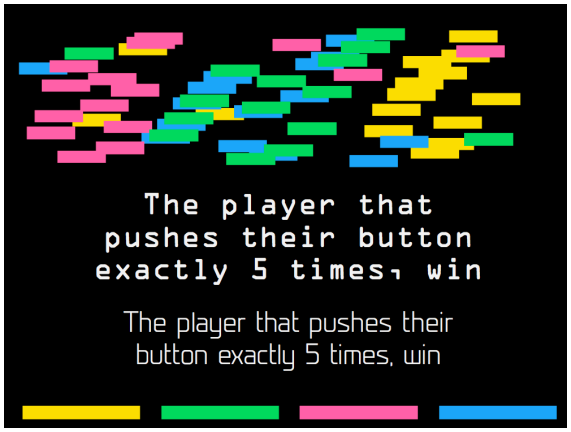

Figure 56: Second iteration mockup of how the screen could look. This way, there coudl be mpre players, and the players could all be showed while the game was playing. When the round ended, the blocks of the failing players would "pop" - signified by a sound - and disappear.The winning once would drop down to their teams platform.

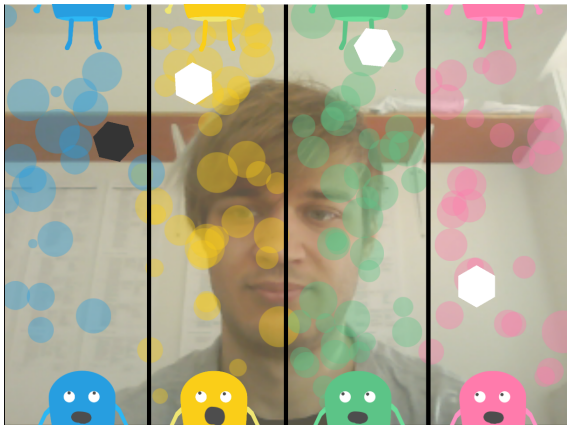

Figure 57: A screenshot from the new tutorial added to *BlowIT*.

enough a Demo Dag.

S.T.A.C.K. is a game we made using HappyFunTimes. It is a clone of the game B.U.T.T.O.N., but made for a crowd to play using their smartphones. We wanted to make more games that had players interacting outside of the screen, and that is exactly what the game B.U.T.T.O.N. does. On top of that, button only uses a single button, which is super easy to set up with HappyFunTimes. It also means that we don't have to display a complex controller on the screen. We can just display a big single button.

Once we had the idea and had asked permission from Die Gute Fabrik, it didn't take much more than a day to put together the game. We were able to clone most things from the original game, but had to come up with our own tasks for the players and also make a gameplay and UI that worked for so many players.

We decided to split players into teams rather than to have them play as individuals. We have an idea that only letting a single player win each round leads to too many losers, which leads to too many unhappy players. We therefore went with four teams, without much of a reason why it should be four rather than any other number. I guess we just defaulted to the maximum number of players in traditional local multiplayer games. I don't quite remember.

We had a lot of fun with the UI. We wanted it to be super minimalistic. At first I actually wanted to give each player a small pixel avatar from an avater generator I have lying around, but Anne

Figure 58: Simon, sitting in front of his computer, testing the new feedback in *BlowIT*.

didn't like that, so we went with giving the players boxes instead. Each player has a colored box with their name written on it. When you press the button, your box squishes together. It felt pretty cute and neat, so we went with that.

When players failed, their box would become grey, and in the end all boxes that had won the round would get sucked down to the bottom, where they would get collected and added to the team's points. I thought this was pretty neat be cause it created a chaotic movement as the boxes zoomed down towards their score. Boxes would collide and fly around the screen frantically. Pretty cool.

Anne designed the tasks for the players. There were three different kinds.

1. tap at least X times to win/lose
2. tap exactly X times to win
3. hold button for X seconds to win/lose

I think she made some pretty neat and interesting tasks.

F.2 The event

Date :	18Mar2016
Location:	Teacher's meeting room at ITU. This is a pretty long room with a large table in the middle, making the space divide in two.
Crowd size:	$\sim$ 5-10
Context:	Event just for playing crowd games and going to the ScrollBar afterwards.
General atmosphere:	careful
Equipment & setup:	A PC, two large buttons, a webcam.

This was a playtest, where we invited players to come and try out our games. It was organized via a Facebook event, inviting our friends and fellow students to come play with us, and with the invitation to join us for the Friday bar afterwards, as we hoped this would attract some players.

The event was video recorded for us to use when writing out observation journals.

F.3 Observation journal, Jump on heads

People playing:	5-6
People not able to play:	2 (no WiFi and no Eduroam account)
Game duration:	$\sim$ 1 minute after everyone had joined

Observation journal, Simon

Players were confused that they got new teams, new colors each round.

People from outside ITU could not join the game Some were running out of battery and could not join One phone did not have functioning WiFi

HappyFunTimes works mostly for installations, not presentations. Maybe HFT would work well for games where two players play on one phone.

It was hard to work together, because it is too easy to help the opposing team. Not much stagnation. Games ended quickly and new ones began.

Observation journal, Anne

We started out by playing this while people were arriving. Simon instructed people in how to play it, while I was playing along. One of the first players asked if they were on teams or not, and if they were on the same team. But Simon told them that they were actually on competing teams. They were not completely sure what to do and how to win. Someone said that she kept forgetting who she were. A surprisingly lot of the people didn't had a phone, I borrowed mine to one of the participants. When a team won, the players on the winning team would congratulate each other.

F.4 Observation journal, BlowIT

People playing:	7-8 (teams of 2 - 2 - 2 - 1). 10 players in second round.
People not able to play:	1 (Anne)
Game duration:	$\sim$ 2 minutes of tutorial. 2 rounds of 1 minute each.

Observation journal, Simon

People had a problem understanding that movement caused the balls to float. For example: Astrid asked whether it was possible to push down the ball.

People didn't understand that the ball hitting the ceiling caused a failure.

For round 2 everyone seemed to understand the game and there was a lot of tenseness (as could be heard from the sounds people made when they had to avert being hit by the ball).

Comment from Yamit in round 2: "Am I even doing anything?" Suggests that she felt like she had little impact on the game.

Teams kept playing even though they were technically out of the game. Good!

One team failed early and didn't understand that they could keep on playing. They found out by themselves eventually though.

Still problem with people being far away or too close to the screen.

"Hvad skal man?" Many people were unsure what the goal of the game was.

Make floor and ceiling approach each other slowly to press the issue.

Observation journal, Anne

I introduced people to move in order to get the ball to float. Unfortunately, there were a table in the middle of the room, nailed to the floor, and since this was the only available room on that day, people had to play like that.

People were confused on how to play. "Can you move it up if you do like this?" someone asked, waving her arms upwards. So we had to explain that no, it did not take into account what direction one were waving, just how much you move. In this test, people were really into it, waving like crazy and yelling "nooo" when the ball was close to theirs. Somebody suggested to their team that they needed to hit one of the other teams character.

F.5 Observation journal, Crowd Mole Hammers

People playing:	7
People not able to play:	1 (Anne)
Game duration:	$\sim$ 2 minutes

Observation journal, Simon

No particular instructions were given how the crowd should communicate. They ended up using both sound and gestures.

Again a lot of tenseness to be ready to give the signal to the mediator quickly.

Players made up their own signal for pushing the button repeatedly rapidly and the mediators seemed to understand what was meant by this signal. What's going on here? Common body language?

Yamit comment: "My cheeks are hurting" suggesting that she laughed very hard playing the game.

Figure 59: The players at LET'S PLAY playing *Mole Hammers*

The button worked well because it gave the mediators more freedom to move around. The setup of the button can be seen in Figure 59.

The game was a bit broken in the sense that the two mediators could see the signals from the other half of the crowd. Simon's half of the crowd talked about a tactic where we could fool the other mediator to press his button.

Observation journal, Anne

As soon as the mediators were presented with the buttons, they tried them out, starting the game. When the game started, the crowd players just started yelling at the mediators when to push.

The mediators were confused, asking if they should hold the button or just push it quickly. They had not seen the game and had no idea what the game was and how to play it. A bit into the game, I explained them that they were controlling hammers and supposed to bask some animals with it.

F.6 Observation journal, Canabalt

People playing:	9
People not able to play:	1 (Anne)
Game duration:	$\sim$ 2 minute

Observation journal, Simon

Canabalt has first played with sounds and afterwards only with gestures.

Mediator (Stefano) reacts to just one person saying "jump". This aligns with what Andreas told me after a similar session in the Game Dev class.

When playing with sounds, people tended to be very quiet, and you can mostly hear Simon shouting in the video. When playing with movement, almost everyone is helping.

Canabalt needs a lot of precision in how long the button should be held. The medium started out with pushing the button for too short but was then told to hold it for longer. This however messes up some of the later jumps where you have to jump short.

Observation journal, Anne

Now everybody understood the concept with a mediator and the buttons, so there was not the same confusion as with the first game. When getting stuck at a jump in the level, Simon told the mediator that he would need to sometimes hold the button, instead of just bashing it quickly.

F.7 Observation journal, S.T.A.C.K.

People playing:	8
People not able to play:	2 (Simon (no Wifi), Miki (no eduroam account))
Game duration:	$\sim$ 5 minutes

Observation journal, Simon

People start out holding back and not doing what the game tells them to (sounds are also missing)

It should be possible for the game host to choose which kind of task to give the players. Either that or the game should cycle through them. We got the same kind of task a couple of times in the beginning.

Nothing tells the players why they lost (BUTTON does this perfectly though)

Astrid was unsure if the game counts taps for a team or for a player. It would be super interesting if it counted for the team as a whole.

Anne made it interesting by pressing the other peoples screen.

Lars' phone fell on the floor and that WAS NOT FUNNY AT ALL! BUTTON is just not made to be played with phones. Praveen: "That escalated quickly".

After a while most people are doing as told by the game.

Praveen and Anne are pretty good at fucking it up for each other :D

The tech worked as intended almost.

Observation journal, Anne

When we played it,people were really careful and just did exactly what the game told them. It as not hectic and physical like when playing B.U.T.T.O.N, and nobody really touched other phones than their own. I was playing along so I started playing "unfair", tapping on the phone of the player next to me, Praveen. At first he was a little confused about this, but quickly he got the idea and played along, trying to grab my phone. But still, I felt like I couldn't really let go and just play rough. The fear of a broken smartphone screen was a little to apparent. And when someone accomplished to knock a phone down from the table and on to the floor, the room just went deadly silent, until the owner revealed that the phone was not broken, and people started laughing about the tension and sudden seriousness.

F.8 Observation journal, One Button Bob

People playing:	9
People not able to play:	1 (Anne as game host)
Game duration:	~ 5

Observation journal, Simon

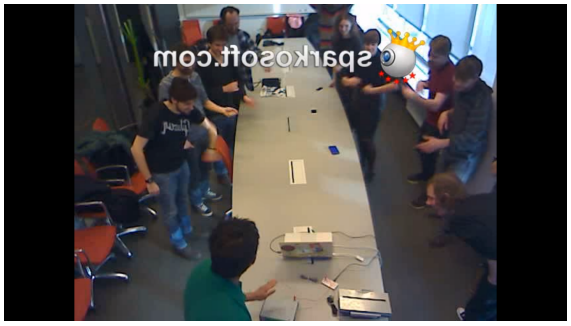

Figure 60: The players at LET'S PLAY playing *One Button Bob*. Here they are doing a throwing action.

Players were asked to make gestures corresponding to the actions Bob was doing in the game. Figure 60 shows how players are pretending to throw a ninja star.

Making two consecutive actions works very badly. Examples are pushing and quickly releasing the button or pressing the button twice quickly.

On the screen where you step onto a moving platform, praveen (pushing the button) steps a bit too far and actually manages to hit the platform and the crowd goes wild. Awesome moment.

On that same screen, you have multiple different strategies. You can either go to the edge and then stop, wait, and step onto the platform when it approaches, or time your walk more carefully from the beginning and not stop but go directly onto the platform. I (Simon) noticed that I wanted to use a different strategy than the majority and therefore suggested to the others that we use my strategy instead.

In the throwing game Praveen says: "Everyone is throwing at different times". This is probably because there are no clear times at which there should be thrown. You have multiple different possible strategies.

Astrid started clapping in the throwing game. She says "I think this is too hard to see" while making a throwing motion with her hand.

Reflections:

Mediator input - both as audio and visual - is bad for quick sequential input, games where there are multiple clearly viable strategies, input that takes great precision (medium precision is OK).

Putting peoples smartphones into jeopardy is a bad idea. 2 minutes seems to be a good max length of time to play the same game. More than that gets repetitive when the games are as simple as these.

Observation journal, Anne

When we asked people to act out what the player should do - for example throw or jump - all of the players acted along. When the character had to walk and then stand still at a certain time to avoid some obstacles, the crowd needed to show really precise when to stop. So when they stopped the walking, the stood completely still, almost like freezing, in stiff positions. It was fun to see them really embodying the character.

In this game, the challenges were so hard that they had to try out some of them many times in order to figure out how to overcome them. And also, even when it seemed like they all had the same idea about what to do, it really was hard for them to coordinate so it would be at the precise time. It created some tension in the room, and the players were paying close attentions. So whenever they did accomplish an obstacle, they were super happy about it, cheering at each other.

G Colorave

G.1 Design journals before first playtest

Design journal, Simon

The design of *Colorave* came out of a request from Lena and Patrick who were planning the Nordic Game Jam preparty at the time. Since they knew that Anne and I were working on prototypes of crowd games, they asked us if we would be up for making a game for the preparty and we of course said yes.

For a long time, we had been wanting to make a game that was controlled by a game master, who would also make the music for the game, and we quickly established a collaboration with the music producer Andreas Lagerstedt, who would be in charge of the sound and music for the game.

The three of us, together worked out the mechanics of the game. We already had the tech from *Put a Smile On* and *BlowIT*, which both used the webcam as an input device. The thing we liked about using the webcam was that it allowed anyone to join the game without requiring any large amount of effort. The microphone would have met this requirement as well, but it wouldn't work with a game where the music and sound was an important part. Looking back at games like *THE WUUUUUUUUUUU* and *WOOORRK!* that use microphones as input, you all of a sudden notice that they don't use any sound other than the voices of the crowd.

Anne and I had also had many discussions about the use of props in crowd games. Props take the weird and quite limited inputs that crowd games have available and add an extra physical dimension to them. With *Put a Smile On* we had also used props, but those were in the form of people's smartphones, which turned out to not be too great for a webcam to recognize. Smartphones have glare and they don't all give off the same kind of light. Also smartphones automatically have their screens turn off. We wanted to avoid all of these problems and use a more homogeneous light source and pretty quickly came up with using glowsticks. The glowstick also has the advantage

that it comes with a certain aesthetic for the game. It has a very rave-y feel too it. It's out there. It's doing all sorts of crazy stuff.

In the beginning we had a lot of discussions on how and what the DJ should control. When Andreas would be triggering a sound from his MIDI controller, should he then also control a certain part of the game. We quickly decided that the tech would be too much of a hassle to set up and that Andreas would rather just press two buttons simultaneously if he had to control both the game and the sound. This would later turn out to be a great decision, since we actually had to split up the role and have both a DJ and a game host (Simon). Andreas would control everything sound-related and Simon would control the game.

Then we had the problem with the camera that we had used for *Put a Smile On*. We quickly discarded the idea of using the same camera, since it automatically did white balance and exposure, which could randomly be detrimental to the game. The camera we needed to get was required to have a fairly wide lens, not automatically adjust anything, and have a fairly large sensor so it would be able to see the glowsticks at a decent framerate. We looked into using a DSLR, but could not find a good setup for that, the framerate was too low on the stream to the computer. We looked into using a GoPro because of the fisheye lens which would have been perfect, but it does not have webcam functionality. In the end, we asked ITU if they would be interested in buying a camera for us, and they got us a nice Ultra Wide Angle HD Webcam from Logitech. It had a 90 degree field of view and fulfilled all the other requirements.

The gameplay itself came along quite naturally from the constraints. We decided that using the glowsticks to color the screen would be quite cool, and from that we came up with the idea of capturing fields on the screen for your team. At this point we had a discussion exactly how many teams we should split people into, and I came up with the idea of having only two teams. This would make it feel more like a presidential election where you have a clear winning team and a clear losing team. Adding a third color would muddle up the two losing teams and would make the winning or losing less interesting. We also decided to use red and green glowsticks, since we had prior experience with these two colors being very bright, which was important for the technical side of things.

On the technical side of things, I very early made a simple test that it would actually be possible to use a webcam to color things on the screen. I was already pretty confident that it would work, and it turned out to be the case when I tested at the office. I cam up with something super smart in order to make any sizes of fields to color. Anne could make the fields in vector and then just make sure to give all fields a different color value. As long as all filed had different colors, and that all pixels within each field had the exact same color, I was able to process the image and assign a field to each pixel on the screen. Once a certain percentage of pixels in a field had been colored, the entire field could be captured by that color.

We also pretty quickly tested in a dark room with glowsticks that we had gotten at Tiger. These would not be the final kinds of glowsticks, but the test gave an impression of how to calibrate the camera in a dark setting and what kind of control was necessary to give to the game host from within the game. When it was just Anne and me playtesting the game, we would run around a lot and try to color as many different fields as possible. Bumping into each other in order to try and color on top of what the other had colored.

At this point we realized that when we shake the glowstick rapidly, the camera didn't have a chance to get enough color to react. We had to move the glowsticks slowly for the computer to be able to discern them. We decided to go with big thick glowsticks for the playtest at ScrollBar.

We also realized that the game was reacting too slowly to the input, so I sat down and tried to do some optimizing. I found a smarter way to mirror the image and could cut some of the processing off of each pixel. In reality all of this processing should have been done on the GPU, but since all of this was prototyping, I had no time to get into shader programming. It would just have to be as good as possible. After optimizing the game as much as possible, the feedback latency was measured to be 200 ms. Unfortunately not within Swink's golden 100 ms for real-time control.

At this point the game was pretty much ready, except for the whole framework around it. I made

sure to implement calibration controls as we have learned is the smartest. We even decided to make an image that is a keyboard layout with everything on there, so I could remember all the keys while being up there game hosting on stage. We didn't show any of those controls on the screen. Only actually the first screen which was calibration, but which wasn't meant to be shown to the crowd.

Andreas had designed music for a number of different levels in the game, so Anne designed animations to fit each of them, and I made sure that I could trigger them from the keyboard. Something that had seemed lame in many of the games before *Colorave* was that the end screen where a winner was declared was either nonexisting of kinda lame. So we decided to make a super cool win screen where the points are counted up fast at first and then more and more slowly. I basically just stole it from *Splatoon* but that works.

We also made a countdown screen, which would be the first thing people would see on the projector. The countdown screen is controllable, so the game host can add and subtract minutes and seconds from the keyboard.

One of the last things I added, was the ability for me to introduce screenshake by the press of a button. I thought this would be a quite nice way to add something to the game at any point. The idea was also that I would use it at the same time as Andreas would use some of his sounds, so that visuals and sound would compliment each other.

Design journal, Anne

We were asked by Copenhagen Game Collective to do a crowd game for the Nordic Game Jam pre-party. All the organizers could tell us about the event at that time, was that it was going to be at the venue Pumpehuset in Copenhagen (http://pumpehuset.dk/), and that it would take place the evening before the game jam were kicked off. They had no idea how many would actually show up, but since the game jam itself had almost 900 participants, they figured it would be a couple of hundred people.

Since it was an party event, we wanted to make a game that could get the party started. We also had an idea about taking the game host role further, making it more of a live performance than a traditional game. It was clear to us, that an important part of this was music, so we asked a friend of ours, Andreas Lagerstedt, to help out. Andreas is both a DJ and electronic musician, but besides that he also studies game design. He was immediately intrigued by the idea of live performing music and sound effects to a crowd game. His part of the work was mainly to share his knowledge about how a musical setup like that could be, what was possible and also aesthetic concerns about the sounds. And then of course, he made and performed all the sounds and music.

In the initial brainstorm for the game, we had these goals in mind: A game that could get the party started, with a live performed element and music as a big part of it. We got the idea of coloring with a glowstick, a little bit like when we traced the movement of players in *BlowIT* or *Put A Smile On*, but just even more clear, since the room could be totally dark and the glowsticks would light up even more. This could fit the party setting and mood that we were going for. We decided to let the glowstick be the core of the design, designing music and graphics around that. We decided on a rave-theme, with 90s rave in mind.

After this initial brainstorm and getting to this idea, we worked on the game without Andreas for the next week or so, while he did some samples for the game by himself.

When doing the graphics, I made it with glowsticks in mind. After trying different colors as inputs, by holding up different colors to the camera in a dark room, we found out that red and green were easiest for the camera to distinguish. So this was the basis for my decision of the colors for the two teams.

I made a kind of in-game storyboard for the game, to visualize the different stages of the game. This worked as a design document for us to discuss around, but also for both of us as an overview of what needed to be made for the game.

Figure 61: The in-game storyboard, showing the progression of the game. Should be read from left to right, starting with the top row.

We mentioned to Andreas, that we have also considered elements from folk games in the game. He then came up with the idea of adding rules to each round, like "this time, do it with the glowstick in your mouth".

We also thought it could be fun to let people just play around with the glowsticks before the game started, while they had to wait for the game to start anyway, but also had got their glowstick already.

G.2 Playtest, ScrollBar

Date :	01Apr2016
Location:	The dance floor in ScrollBar
Crowd size:	$\sim$ 30
Context:	ScrollBar goers and a bunch of drunk nurse students
General atmosphere:	Drinking some beers after a long week. Ready for the weekend.
Equipment & setup:	PC, DJ-stuff, projector from ITU reception, tripod from ITU reception, webcam.

G.3 Observation journals

Observation journal, Simon

This playtest took place in the ScrollBar at ITU on a somewhat regular Friday evening. Regular in the sense that ScrollBar is always open on a Friday, a bit different in the sense that a crowd of quite drunk nurse students was visiting the bar on this particular day.

The setup process already had some difficulties. Andreas was already DJing and we noticed that the one speaker was broken. One thing to note is that in ScrollBar, everything is controlled from the bar, which means that one has to get the attention of the already busy bartenders to get them to help.

Earlier that day, we had been by the ScrollBar to decide on where to set up the webcam. Unfortunately we had been a bit late in planning this, and ended up concluding that the stage + a tripod + a table would get the camera up high enough to make it work. When we got to the party, however, there was no stage and the speakers were blocking the view from setting up the camera close to the screen. We therefore had to place the camera out to one side, covering only most of the dance floor and from an awkward angle. We know from testing ourselves that it can be quite difficult to find ones own glowstick on the screen, even when the camera is positioned optimally, so this definitely didn't help. Furthermore, coordinating movement to match what you see on the screen is pretty difficult as well, especially when the camera is at a different angle than the screen. On top of all that, the players were drunk and didn't take instructions very well, or quickly forgot them, and even though Andreas shouted out multiple times that the camera was on the side, some players tended to forget or not care. This was also due to the fact that some players left and new players joined in during the game.

The speaker was fixed after a short while, waiting. This was in no way detrimental to the game itself, it just caused a delay.

ScrollBar has a built in projector and screen. Unfortunately we could not get it to work. Again this caused a delay but we fortunately were able to borrow a projector from the reception. Although the display ended up distorted and much smaller than we had wanted, it served as an OK substitute.

After sound and visuals were working (almost working. We didn't get sound output from the game, only music and sounds from Andreas computer, which meant less feedback to the audience), we were ready to start the game. We put on the waiting screen, which instructs players to get a glowstick and get ready to play. During this time, the players are also able to color the screen with their glowsticks and we ended up with a myriad of amazingly drawn penises.

The game was started and Andreas and I had planned to work together as a team controlling the game. Andreas would control the music and the MCing, while I controlled the game and the visuals. This meant that we constantly needed to communicate, to make sure the music and game followed one another.

Andreas put on a show, he basically carried the whole experience both with his music and his voice. Imagine one of those really good concerts where the artist communicates well with the audience and where small mini-games are played like "everyone get down and then jump at the same time" or "everyone take their shirt off and just throw it into the air as high as possible". Those concert experiences are often remembered. I remember being at a The Knife concert where the warm-up band was actually an aerobics instructor, making the entire crowd do an actual warm-up before the concert. Experiences like that stand out and are remembered in my experience.

People responded very well to Andreas instructions and mostly followed them. For example when he told the crowd to dance slow or put the glowstick in their mouth, the players followed instructions and played along, mostly.

An interesting observation from Anne was that she saw one of the players cheat by going very close to the camera with their glowstick in one of the first rounds. This easily made green team win that round, however the behavior generally stopped after that round and was only repeated in one of the later rounds.

Because of our setup. Players were more inclined to color the top of the screen than the bottom, except for that one round where they were told to "Go down low".

One piece of feedback that we received was that it was difficult to tell that a captured field could not be recaptured by another color.

Observation journal, Anne

When setting up, we had some trouble with the projector at Scrollbar. We then switched to another one, resulting in a much smaller image. Before the countdown started, there were some party lights on and Andreas was playing his dj set. People were sitting at tables around in the bar and talking and drinking. The atmosphere were more relaxed than partying, and only one person was at the dance floor. In the middle of that, a crowd of students from another university had just arrived, and they were really drunk and in a party mood.

I was giving out glow sticks to people sitting at the tables in the bar, encouraging them to come and play. I also handed out to the drunk students from the other university, thinking that they might not really want to play but just wanting the cool arty gadget glow sticks. Most people asked me, what the game was about, and generally seemed rather confused about me handing them glow sticks. I got a lot of questions like "what is this for" and almost every person asked me "how do you play it". I answered this with that it was a game that involved getting on the dance floor when the countdown ended, and that there would be an explanation on how to play when the game started. When the countdown was ending, people began to enter the dance floor. Some people waited until Andreas announced the game was about to start and the start screen was shown.

During the countdown, we asked for the bar to turn off the party lights. This meant that people could use the glow sticks to interact with the picture on the screen. Some of the people started entering the dance floor, when they saw that this was possible. They then started to draw on the canvas, and quickly the first penis appeared. This seemed super funny to us, since this was something that we joked around with. Than whenever people had an opportunity to draw anything, they would draw a penis.

When the game started, the crowd quickly got the concept of the game. I was afraid if the drunker part of the crowd would participate, or if they would do something that would ruin the experience for the rest of the players. But they actually all managed to participate.

Reflections: Play community - a rather mixed crowd with mixed expectations and boundaries.

The countdown was helping people to prepare, was good for me to explain that the game was not started yet and would start soon

Good to have a game host to initialize the game

People got the idea that they could draw.

G.4 Design journals, before All the Rave

Design journal, Simon

After the playtest at ITU, we mostly had issues with the setup. The crowd had been responsive, Andreas had made them dance and rave and do all the things we wanted them to do, but no one had understood the input. The camera had been sitting there, off to one side and people hadn't really understood. So for the NGJ preparty we wanted people to understand. The first step in getting them to understand was therefore re positioning the camera. We discussed and tested various possibilities, like setting the camera in the same position as it is on a laptop for example, but we ultimately ended up with a solution where the camera is above the crowd. This ensures that everyone in the crowd is seen equally, except perhaps people who are very tall. It would

also let us define a space, a rectangle on the floor within which people should stand in order to participate in the game.

Having decided to place the camera from above, we tested exactly which orientation the camera should have to make the inputs as intuitive as possible. After having both of us sit in a chair in front of the screen with the other person holding the camera from above, we decided to orient the camera such that right and left would be right and left on the screen and forward would be up on the screen and backwards would be down on the screen. In a way, this is also the mapping that a regular controller has to a game on the screen, so there is not much surprise there.

For fun stuff, I added small text things every time a field is captured. They have texts such as "combo", "multiplier", "x2", "hotdog", and other such silly words. Most of them however have the words "locked" or "captured" in order to communicate to the players that they are capturing fields when they color them enough.

I also wanted to be more in control of the visuals, so I added buttons for particles for both teams. The win screen was still a bit dull, so these could help put something more interesting there. Also the particles could potentially be useful elsewhere. The good thing about putting the button as only a possibility, was that I could choose not to use it at all if it didn't become relevant.

I fixed the clock which was broken at the ScrollBar playtest, nothing too interesting there, except that we decided to make i blink as soon as it hit 00:00, and to make it not go into the negative.

Design journal, Anne

Since some players had a hard time understanding the goal of the game and when they actually captured fields (and that those fields could not be captured by someone else afterwards), we added some text to show when fields were captured. I also iterated on the main font for the game, making it look a little more rave.

Figure 62: The new font/logo type for Colorave, made up by overlapping fonts

G.5 NGJ preparty - All the Rave

Date :	07Apr2016
Location:	Pumpehuset. Large concert venue. 8 m stage, 6 m to ceiling. Huge screen on stage.
Crowd size:	$\sim$ 100 players
Context:	Game developers going to Nordic Game Jam and their friends.
General atmosphere:	Excited for the upcoming party and game jam. Wanting to play games.
Equipment & setup:	Simon's PC, Andreas' Mac, projector, huge screen, new wide angle webcam, nice speakers.

Observation journal, Anne

The game was announced from the stage at the welcome talk, so people already had an idea that there would be a game taking place. After the talk there were 10 minutes of people hanging around and drinking drinks.

When the game started, me and two friends handed out glow sticks. We activated all of the glow sticks just before the game started, to make the glow sticks appealing and intriguing so people would want to get one and hopefully join the game. My friends told me, that people was really excited about getting the glow sticks. I have instructed them to go around and hand them out, but it wasn't necessary since people came to them for glow sticks. Almost all of the 160 glow sticks were handed out before the game started, I placed the last few glow sticks at the front of the stage and by the end they were all gone.

A general thing people asked us about afterwards, was how big the play area were. They had trouble finding themselves during the game, and some even doubted if the game was actually working, and thought it was just faked. People not in the game area played anyway. But we also talked to people who was actually in the game area, and they did manage to find themselves and could see that they were coloring.

For the rest of the evening, people were wearing the glow sticks on them. I found this Colorave reminiscence to be a reminder of the collective experience the crowd shared earlier in the evening.

Observation journal, Simon

NGJ preparty was going to be our big moment for our game Colorave, which we had worked hard on up until this point. In the game, we had put all of our experiences from all earlier prototypes + some new ideas to test. The main point of the game was to see what happens in the collaboration between a game, a game host, and a DJ, which in turn also became the MC. A lot of new features were introduced compared to our older games, and the game ended up as a mashup of a concert and a local multiplayer game. Setting up the game Compared to our playtest in ScrollBar, for Pumpehuset we were well prepared for the setup of the game. Improvements included:

- Setting up the webcam above the crowd in direction that represented the most intuitive coordinate system we could come up with.
- Having 160 glowsticks glowing and ready to be handed out by Anne, Mira, and Tanja (as a side note, Mira and Tanja had a really good time handing out glowsticks. People are just happy to receive free stuff)
- A big projector connected with HDMI
- Audio from the game mixed into Andreas' music
- A keyboard layout for the game host
- We set up everything the day before
- Andreas and Simon were wearing costumes to make a more rave-y mood
- We had a schedule and todolist on paper which helped us remember everything Setups went pretty smoothly. Anne and Andreas helped me calibrate the camera. Things we did not prepare:
- We did not mark the area in which you were actually in the game, playing. In the end this meant that people were too spread out and that only 20-30 people were actually inside the camera's view during the game (see Figure 63) even though 160 people actually had the glow sticks.

The beginning of the show was a bit chaotic. Andreas was supposed to be DJ-ing when people were let in, but unfortunately there were a couple of misunderstandings and the crowd poured in before expected, which meant that Andreas wasn't playing and that I (Simon) already had some visuals up for testing purposes. I quickly took down the visuals and Andreas started DJ-ing.

The hosts of the party had a couple of announcements after which we started the 10 minute countdown screen for our game and Andreas went back to DJ-ing to get people in mood for the

Figure 63: A capture from the webcam during the game. The stage is a the top, which is why there are no glowsticks there.

game. During these ten minutes, Anne, Tanja and Mira handed out glowsticks to the crowd and Andreas did occasional announcements to hype the game. On top of that. People were already coloring the screen with their glowsticks (see Figure 64). Letting people color the screen during this waiting time, meant that a lot of them were staying on the dance floor.

Right before the game started, Andreas did and announcement and explained the rules to the crowd. At this point Andreas should probably also have mentioned that the camera was positioned above the crowd and that the area of play was small.

During the game, Andreas and I didn't have to talk much, but we did communicate with our bodies, Andreas signaling when he was about to end a round, so that I could press "next" at the right time. Andreas controlled the flow and progression of the game, while I acted more as a VJ, always keeping the game in the correct state and adding screen shake and particle effects to spice things up.

The crowd was responding nicely to Andreas' MC'ing.

Green won the two first rounds handidly. Thereafter red seemed to take over, winning a total of 6 out of 8 rounds. Many rounds were pretty close, but red did definitely dominate in the end. Reasons for this are uknown. Looking at the screenshots from the game, both teams had about 10-13 players. The teams were equally balanced.

Being a game host, I was quite happy that I had implemented all of the screenshake and particles, because I used them a lot during the game. It was so cool to be able to respond to Andreas' voice and sounds with the visuals. In the end I just spammed the red particles when they won everything. So cool.

Talking to people after the game, many of those I talked to came up with excuses if they were on the losing team, or told me stories of how they contributed if they were on the winning team. Setting the stage as a "presidential election" seemed to work well.

Some people were carrying around multiple glow sticks after the game, collecting ones they found.

One person I talked to found the mapping between their motion and what was shown on the

Figure 64: An image dump of one of the drawings that the crowd made while they were waiting for the game to start.

screen counter intuitive. Would it have helped if Andreas had explained that the camera was positioned above? Would it have helped to show a coordinate system on the screen?

Someone (Johannes?) said that he didn't feel like he had influence on the outcome of the game, but that he had a good time anyway just following the flow of events and dancing in the crowd.

Figure 65: Andreas and Simon on stage with the game about to start. The projector on the right.

H Nordic Game Jam presentations - the Chrome Cast room

Date :	10Apr2016
Location:	Nordic Game Jam, Aalborg University, Campus Sydhavn
Crowd size:	$\sim$ Around 80 people were in the audience
Context:	One of several rooms for presentations at Nordic Game Jam
General atmosphere:	Tired crowd, almost everybody there brought something to present themselves
Equipment & setup:	A Chrome cast and a big projector screen. The players own smartphones.

H.1 Observation journal, Anne

Figure 66: The crowd watching the presentations in the Chromecast presentations room

At the Nordic Game Jam, there were a few crowd games. They were supposed to present in the room for the Google Chromecast games. I was there both to observe the crowd games, and present my own game (that was not a crowd game, by the way).

Figure 67: *Enurendo* presenting at Nordic Game Jam in the Chromecast room

One of the games presented was *Enurendo* by Half Past Yellow. I sat besides from one of the players, and she got really frustrated by playing the game as she had no idea what character were hers and if she was even doing anything. She also mentioned that she figured it might lack a lot, and she ended up with give up on playing. All of the character of one team were identical, and there were nothing in the graphics to indicate what character belonged to which player.

Another game presented was the game *TRACK & FIELD*. It was also made by some of the participants at the jam. In this game, two runners are competing against each other. The interesting part of this game was that when you go to the website, you can choose what team to play for.

Figure 68: Start screen for *Enurendo*, asking players to get their smartphones and join the game

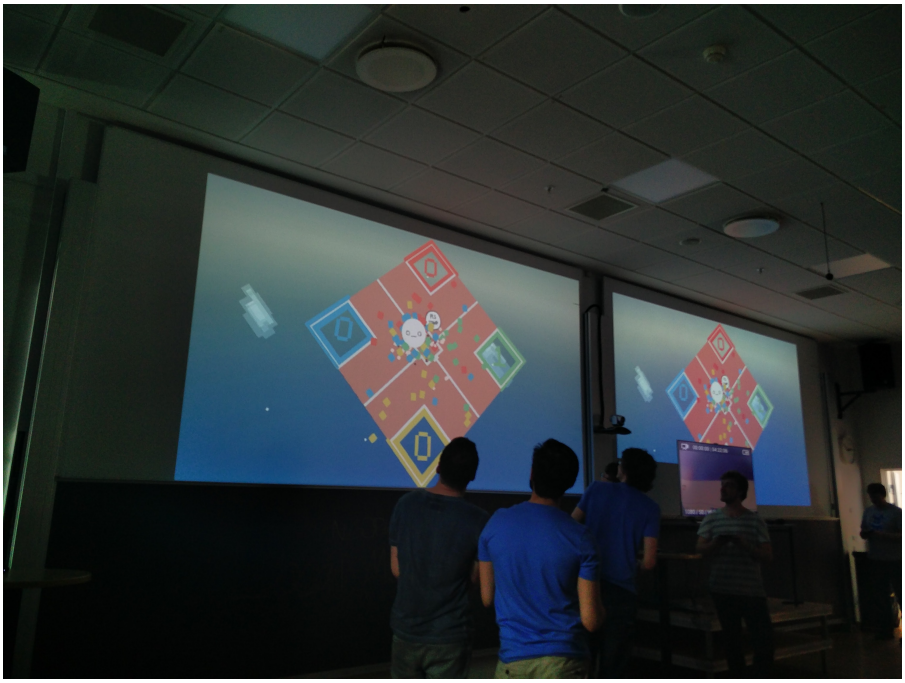

Figure 69: Playing Enurendo. As it can be seen from this picture, there are around 15 players on each team, so about 50-60 players in total, and they all look identical to their team mates.

Figure 70: *TRACK & FIELD* start page

Figure 71: This is how the controls look in the browser

Figure 72: This is how the game looks. The two characters are running, and there is also stats on how many players is currently in the game

I Enurendo - NGJ Award Show

Date :	10Apr2016
Location:	Imperial, largest screen in the north
Crowd size:	$\sim$ 160 players, but a much larger crowd not playing
Context:	The Nordic Game Jam award show. Participants were game developers from the Nordic Game Jam
General atmosphere:	Tiredness after a long jam
Equipment & setup:	Huge screen, webserver, and players' own smartphones.

I.1 Observation journals

Observation journal, Simon

Game by Half Past Yellow. They made it to the finals of Nordic Game Jam and were on stage with their game which is basically Go Nuts in isometric view for a large amount of players. All players look exactly the same so it was really hard to see which one was you, especially because there was a large amount of latency. I talked to Max, one of the designers, afterwards who brushed it off by saying that it just "feels more like controlling a ship" and that they wanted to give the players a feeling of moving a swarm together rather than having control over a single character. Unfortunately that is not necessarily a good thing.

I ended up joining the game along with about 50 other players at the award show and never found my player character. Other than being on one of four teams, the game had no indication of which character I played as, making it only the synchronization between input from my smartphone and movement of a character on the screen that could hint me towards which character I was controlling. Once 150 players had joined, the server crashed, but already before that the game was lagging, apparently because of all of the "rigidbodies and colliders" according to Max. Interacting with the developers on Twitter, I later learned that the game had maxed out at 160 connected players.

Observation journal, Anne

When watching the game at Imperial, I didn't have power on my phone, so I ended up observing the game instead. Simon was lucky enough to get to play the game, and I sat beside him and watched him play. I saw that he had the same problems as I observed earlier, that he couldn't find his character on the screen. When the designers presented the game, they said that they would just want to let everybody into the game and see how much pressure the server could take, and eventually it crashed.

J Hyper Talk - A MAZE.

Date :	21Apr2016
Location:	Haubentaucher, Berlin. Large screen and tribune with seats
Crowd size:	$\sim$ 100
Context:	Hyper Talks at the A MAZE. independent game developers festival in Berlin. Crowd consisting mostly of independent game developers
General atmosphere:	Pretty bored after a slew of Hyper Talks
Equipment & setup:	Macbook Air, built in microphone, projector.

J.1 Observation journals

Observation journal, Simon

At A MAZE. we did a 5 minute talk in which we played the crowd game *THE WUUUUUUUUUUUUU* while explaining some of the things we consider while designing crowd games.

The game itself was rather quickly explained by basically reading aloud what is on the pause screen. On top of that I explained that the goal was to get the monsters from the left side to their homes on the right side.

It should be noted that this was a slightly modified version of THE WUUUUUUUUUUUUU. Since the microphone for the original had not worked, I had to write my own, which was probably inferior. The crowd had to be very silent for THE WUUUUUUUUUUUU not to go upwards. For the first minute or so they therefore had a hard time getting THE WUUUUUUUUUUUUU to go down. Some people began hushing on the crowd to make them silent so THE WUUUUUUUUUUUUU could go down.

Anne noticed that the crowd tried to align THE WUUUUUUUUUUUUU perfectly with the platforms on the right and left hand sides even though this is not necessary to have the monsters cross over. This is hopefully something that the crowd would learn after playing for a while.

For a second round of the game, I asked the crowd to clap instead of using their voice. The crowd did as I instructed and used their hands rather than their mouths. The round went pretty smoothly and ended with someone shouting, making THE WUUUUUUUUUUUUU go up and destroy several monsters, something I felt was a great end to the game, so I stopped the game at that point and we did our roundoff.

I was later told that a player had not understood that the monsters had to go to the same color as their skin. This, while it might have been frustrating, just added an extra layer of learning to the game, which I think is fine.

In general, the feedback we received for our talk was good.

Figure 73: The room where the talk took place. Besides that, there were also an outside area with a lot of people watching from sun chairs

Reflection: The playthrough was an excellent example of how the crowd learns and teaches itself how to play a game. Even though the gameplay of the game is simple, the crowd learned which strategies to use from experience over time.

Observation journal, Anne

When we did the talk, I was doing the most of the talking while Simon was acting as the game host. We have practiced the talk a lot of times beforehand, and when we gave it, I felt pretty

confident in what I was saying and doing. It felt great to know that Simon had the game and slide under control and that I could just concentrate on the content of the talk.

Figure 74: Simon calibrating the game

Figure 75: Me explaining that Simon calibrated the game to fit the current crowd, and that this is a general thing to consider when designing crowd games

K DreamHack

Date :	16Apr2016
Location:	Malmö Arena. Huge space with a capacity up to 15.000 people in concerts.
Crowd size:	$\sim$ 800
Context:	DreamHack Masters, CS:GO tournament. Crowd consisting of CS:GO fans and players
General atmosphere:	Tenseness watching esports
Equipment & setup:	Huge screen

K.1 Interview with Jonas Ahm by Simon

We randomly got to talking to Jonas who has been at DreamHack Masters in Sweden where The Colossi showcased a number of crowd games. He said that it was super fun to play the games, that it was really tense, and that you felt that you had an impact even though 800 other players were also playing. Both games used the same mechanic, two teams, one avatar each, each player goes to a website and can then swipe to move around their avatar around. Each player only has a small influence, but the faster you swipe, the larger influence you have and as a group you move your avatar around over time.

In the one game the two avatars have to find their way through a maze. The team that finishes first wins.

The other game was a soccer game, where the two avatars fight to get a ball into the other's goal. Reflections

Surprised that Jonas felt like he had control over the game even though there were so many participants.

L Interview with Colossi

The following is a transcript of an interview with the designers and developers Benjamin and Bryan of Colossi. The interview was conducted via Skype, since they are based in Los Angeles. The interview was recorded and transcribed.

Anne We have some questions about testing, because, we want to know about your methods to test, like, what kind of leap you take, and like how many people do you need for a test session, basically.

Benjamin Okay, I mean so typically for, I thinks for a test session to be significant, we need you know, it don't... It doesn't need to be like a large number of people, we don't need a hundred people, right. We just need enough to where... you know.. maybe you don't know everybody in the room and you know, everyone... you know, you are not like a group of like ten people, right? So it has to be more than ten less than a hundred, I think we typically, the number that we found is like 50 or 60.

Bryan Yeah

Benjamin Just like a class. Like, we go to, we go to a class and its a small little lecture hall. Actually maybe even like down to like 30, 35 to 50.

Bryan 30 yeah, 30 to maybe to 60. It depends on what class that is and if they wanted to skip that day of class.

Anne [laughs]

Bryan Yeah.

Benjamin But that way it could, I mean, when you know everybody else in the room the, the feeling is very very different. And so, trying to simulate the real world is much more possible where you don't know everybody and, and in turn you can kind of, you can kind of see a little bit better how people will react when they don't know everyone around them, when they are not comfortable. It's... It... Can we still get the same emotions out of, of... You know... Will they still cheer, will they still start yelling? .. Yeah.

Anne Cool. What, what about... When you test. What is it you test. Because we know that your technology is kind of set. What is it you test then?

Bryan A lot of it is like, design, you know. We have a good idea of what we think is fun and what we think is not fun... It all about - have you ever heard the quote it's like "change a number and you change the experience". It's like, you change one small part of it. It'll be like having a bicycle and you make one tire like, slightly smaller than the other one. It's gonna affect like the whole thing. Because you know, maybe you are gonna ride a little bit weirder, you gonna be a bit farther back, or forward, and its not quite exactly the same, its similar, its an analogous. So its, normally testing, is this fun, is this something we need to tweak, maybe this door opens too quickly, you know. Maybe, this level is too hard, you know, maybe it should have more time or less time, or... Yeah, it's a lot of tweaking really small thing, like most game design.. .You know... If Mario jumps one block higher, then all the levels needs to change so that all the parts that he couldn't jump to before, can still be injumpable, or if that change the whole game, you need to change it, in some way. So. yeah.

Benjamin And then, we also do, we also look at, the user experience part. I mean so for us, not so much for you guys, but for us, we have an interface on the phone, and.. You know.. Just making sure that, 1: that it works on everybody's smartphones, because everyone has a different one. And two, when we add a little thing into the experience.. You know.. Just like, like an extra like step or we take away a step, you know. How do people react. Are people confused. You know. During our earlier playtests, people would like, hold their phones to us, be like, 'oh, its broken, I don't know what I did. And, it's like. Okay, like, you obviously can't do that in a large, large group setting. So.. Let's.. Let's try and just iron out all these, these user issues, user experience issues... Yeah. And then. And then like Bryan said, test make content not only that make sure that it works from a design perspective, well, that it works from a design perspective you know, in, in.. Kind of like content itself, but also with the crowd too, so... Yeah. I mean, it's different design to design for a crowd, so.. It's not really something that has been done. I mean, you can do like, you can do for a platformer, you could say, well you need like platforms and you need a little character and you need to be able to jump and move. You can do like Mario, like Bryan said. Bryan was using. But, you know. For a larger group. It's still, that untested kind of frontier, so you gotta like.. You gotta test like everything, because you don't know what will work and what doesn't. I mean with everything else you kind of have a history, with this you don't really so.. You know.

Anne But is... Is there something.. I'm just wondering, if there is something you have tested on a smaller scale and then when you tested like bigger or when you put it out there, you were like, wow. Wauwwauw, what happened? Or... Something that surprised you?

Benjamin You mean good or bad?

Anne Sorry

Bryan [interrupting] It's always been.. It's always been add more people, add more energy. Like, that's just like what it has always been. It's like if it is fun with 30 people, 50 people, whatever, 100 people, it tends to be really fun with a thousand people, or you know, a few thousand people. So. It just kind of extrapolates bigger. Ahm. And just more people, more cheering, more screaming, means you are gonna cheer and scream more, means you are gonna get up and cheer more. You know. It just has that effect. It's like the wave in a stadium, you know? If 10 people are doing the wave, it's not really that cool. I mean. The energy is still there, it is still the same thing, it's just shorter.

Anne Mmmm..

Bryan If there's thousand people, there is more, if there is 10.000 people there is a lot more, if it is a whole stadium it goes all the way around eight times and.. and we are all cheering and screaming. Still largely the same experience from it just being a wave, but the community experience is different.

Anne Mm.

Benjamin I mean there hasn't been, to speak to you question, well there hasn't really been any surprises in terms on like, like we tested something well in a small scale and we took it up bigger and that didn't work. There hasn't really been any of those, just because any, any thing that we get, kind of, there's a little iffy on the small scale, like someone said "oh, I didn't quite understand what was going on", "I didn't" you know, "wasn't having as much fun as the other ones". It's just, it get bigger, and so. When you get more people, then suddenly a lot of people

are not having fun and just kind of like, okay. To really just because we go so small, there are no surprises on the bigger scale.

Bryan Yeah. It's just kind of what end it's leaning towards, like, if it's leaning to being really fun, more people will make it more fun. If its leaning towards not being fun, more people is gonna be more people not having fun. So.. Yeah.

Anne Cool, yes.

Simon That's a cool experience

Anne Yeah.

Benjamin That is awesome, yeah.

Simon Yeah

Anne Yeah

Anne I wanted to, I also wanted to ask you about, about level design. And about, we.. Previously when we talked, we talked about the game host, or like, the designers having control during the game.

Bryan Okay, yeah.

Anne So maybe, if you can tell a little about how you use that in your games? The, the...

Bryan Yeah, I mean level design, so... [pause] Is really trying to go from just that first order type of level design and then extrapolate upwards, continuing to go kind of more difficult. It's something where... You - We were talking about comedy last time, we talked about things being funny, and. A lot of the funny part is, if you are one person playing like a really simple game, like a touch screen game, you will probably be able to get through it really quickly, but that is not really fun. The fun part is kind of struggling through it, with hundreds if not thousands of other people. So, the level design we keep largely simple for just.. Well. Also, because our people are not all game players, you know, they are not all. They don't play League of Legends, they don't play Call of Duty, maybe they play Candy Crush maybe they play Farmville. But this is a little bit different, so we keep it really simple to understand, really just engaging, quick to get in to. You know, we kind of take the Apple approach to minimalist, beautiful, engaging, immediately, you know

Benjamin Yeah, and actually another part too, um. So, so it actually is a little, in, in some respect, a little easier to design for like a funny, kind of, fun-felt type thing for a large group of people, because, um, when you are playing with a hundred other people on your team, and your team starts doing badly, you never feel like it's you, you always feel it's everybody else not doing what they are supposed to be doing, and so you don't feel frustrated, you are just kind of like "argh, come on guys", like you didn't feel frustrated at yourself, like you might during like a, like a Dark Souls, Mario, like, whatever, you know, like, you missed a jump, you missed something, like, "that was my fault".

Bryan When you get killed in Call of Duty, you'll be like "Aaaw, I, that's..."

Benjamin [interrupting] Or it's the games fault, right? The game was being unfair, and this, its. Because it's deliberately very random, like, with some of the obstacles, and then because you are playing with a hundred other people on your team, you don't feel that same sense of like personal responsibility, as in like.. You know, I was the reason I failed, or the game was being unfair to me. It's like, no the game was being unfair to me and a hundred other people. Like, it was just randomness, arrrgh.. You know, "Come on guys, we can still win, we can still get back" or, you know, it's like "we are going the wrong way, why are we all going that way" it's like "come on guys"

Anne [Laughing] Yes

Bryan Yeah.

Benjamin Yeah, so... You know just, I mean, it was not too hard, from a level perspective, other than just making it so that there is no, like, frustrating parts, I mean, anything that gives you remotely fru.. [interrupting himself] I mean, we have parts where the two team can kind of stuck up against each other, and that a little bit more deliberate, because it's like, now they are stuck and now they are kind of fighting it out, and one team will eventually make it through, and, and whatever. You know, and sometimes they'll get pushed backwards and it's all fun and everyone's like, "aaargh" you know and then it makes everyone feel..

Simon yeah

Benjamin Speaking of, from the hosts site. The host's site lets us kind of accentuate that and, and really like, get those feelings going, that kind of close call feeling, right? Where it's like "I almost won, but then we lost at the last second" or, "we almost lost, we pulled it out the last second", both of those feelings, like, are really powerful, so we have the host kind of over here, um, he is able to, to manipulate it to where like, if one team is falling really far behind, for whatever reason, because of the randomness or because of whatever, it can still make sure that it is still fun and it is still close. You know. It doesn't always have to be, but if a majority of the matches, are like, one team is doing really poorly, then that team is gonna feel really bad.

Anne Yeah

Benjamin So like, let's just like, kind of like help them out and then you know, next match they will usually be better, that's fine. But.. Than, I mean. That's why we have all that kind of controls that we have over here. Not so much really as to control the experience, but to guide it if, it need be. Because the crowd is randomness, i mean, you have hundred people, it's all random, right? You have to make sure the experience is still fun, right, despite it being random

Simon Yeah

Bryan I mean in all actuality, we almost never are doing it. Um... Almost never, umm- It always is pretty equal. I mean, when you get two teams of people, they tend to perform similarly. Sometimes, outliers happens, of course. And then just from a game design perspective, it always good to have kind of a backup plan, you know? I'm sure you guys are running into that as well?

Simon Yeah, yeah

Bryan It's like, "uh, wauw, the thing that shoots confetti out, like, didn't go on the trigger it is supposed to, so here is a button that just makes it go", or just, you know...

Anne Yes

Simon Yeah. Exactly

Anne [laughing]

Bryan And.. All the pyro effect that are supposed to go off whenever this guy walks across the stage, for some reason they didn't go off, so I'm just gonna manually make them go off.

Benjamin I mean, one of the things that we have, you know kind of as a debug thing, is actually adding in players. So we have a counter for the numbers of players on both teams, and, I, just so people can visually see, like, "okay, there are this many people on the team" we have a button that will just like, maybe add like 5 or 10 people to it, because we found that if, you know, at the beginning of the experience is the biggest hurdle, and so, if there is only 10 people on both team playing it, and you want like, 500, you know, you have, if you simulate, kind of it up a little bit more, if you pushed it up, and then in the next game, you don't do anything, more people will turn in, because they are like, "oh, more people are playing and so I wanna play too". And gives, you know, that just more fun and just nobody, nobody's able to tell the difference, in terms of experience, it's... it's the same.

Benjamin Mmm

Anne Uhm, is it, is it so that, so that you can only join a game before it starts, and you can't join during the game?

Benjamin Yes.

Bryan Can't join during the game, but you can join in preparation for the next one.

Benjamin [interrupting] Yeah, because, I mean, yeah. I'm sure, as you guys have thought about, having a user join in the middle of the game, or having a bunch of people joining in the middle of the game, might effect the experience So we just kind of said, you know, "no, the game is.. Once you, once you get to a, to a certain point, the game is closed, you cant get in to this one, but you can get into the next one, right? Get ready...

Bryan From a user experience perspective, for them, I mean, they wouldn't have a lot of ownership. I wouldn't really feel like they are a part of it? And it didn't really make that much sense for us, for them to be there, i mean, our games are really really short, with that kind of action packed burst, and every moment of that is really important. You know like, when the game starts, you know, its like starting Mario Kart. Its like, when you are at those stating lines of Mario Kart, and you can do like the extra boost thing at the beginning, and it's like, a strong moment, and just, you would miss that.

Simon Mmm

Anne Yeah. What, how long are your rounds, approximately?

Benjamin The rounds can be anywhere from around, so, explicitly they are about a minute. I mean, the races, we have a timer for the racer that goes about a minute. Just, you know, to kind of cap it off, typically takes you know, about 30, maybe 45 seconds, if... its going really long. We have never had it, ever, go past the minute timer, like the crowd... We are always, we design the levels to like, "oh like, these levels are too hard" initially when we were starting out, we were like, "oh these levels are too hard, let's make really easy levels for", because you know like, "ten people", right, "they are not going to be able to work together", and then they all were like blow through our levels in like, 15-20 seconds, okay, so like...

Anne [laughing] Yeah

Benjamin Lets make them a little harder, because they are a little bit better. And so.. Never. You know, we have 500 people on a team for DreamHack. They, just completed it, 30-45 seconds, same, always the same. Never. It's never gone over the... It's not like us manipulating it to make sure it's - it's finished. Its just like, it just happens this way. Because the crowds ends up being pretty smart

Bryan Yeah, more people, there is a book... I don't know who wrote the book, It's not [??], but the book is called The Wisdom of Crowds, its a really interesting book, just about kind of crowd think, like, what happen when a large group of people are together. Highly recommend it.

Simon Sounds interesting, yeah.

Bryan Yeah. The Wisdom of Crowds

Benjamin It's a... It's a little bit different, because it mostly talks about how.. So, you have a group of people trying to make a prediction typically, they'll be pretty... I mean, if you take the whole crowd on average, all their predictions together, like there's a jar of candies, and how much candy is in that jar. If you take all their predictions, all together, you'll typically get very very close to what it actually is, you know? So then, that's, you know, say, we kind of extrapolate that out a little bit. This is the same principle for us, right? They are all looking at the same bit of information, but they have their own kind of control over it, and so the average of all of their results is that will eventually get to where they are going. It's very, it's very very fascinating, we'll send you the, the link to the book., after, when we find it.

Anne Thank you, thank you. Great.

[Simon & Benjamin talking at the same time - hard to distinguish]

Simon Can I just as you a technical question?

Benjamin Yeah, sure.

Bryan Sure

Simon So, are all votes equal? So let's say there are 10 votes up and 10 votes.... No, not really... Like...

Anne [Interrupting]: No like, not votes...

Simon [Mumbling] No not really.

Simon Like, ten votes up and five... Yeah, inputs, yeah, sorry. Inputs. Ten swipes up and five swipes down. So, the five swipes, would be worth half as much as the ten swipes.

Benjamin Yep

Simon Yeah, okay.

Benjamin Yeah, I mean it's like, everybody's, everybody has the same effect, of course if you swipe more, that's different.

Simon yeah

Benjamin Hmm. Yeah.

Bryan I mean so we, we basically, we have, you know, kind of an algorithm that lets us kind of divide everyone's votes, as more people come in, we kind of do, you know, some fancy kind of math to make sure, the feeling of.. That it's not too overpowered per person, right? If you have one person, the experience is much different than if you had a hundred people, you know? So you are kind of accounting for, you know, one to ten to fifty to a hundred, just kind.. You know, just doing some behind the scene stuff there. But, everyone's vote, always is equal, that's very important to us. Besides the votes of like, the master kind of controller over here, that's the..

Benjamin Input, input, not votes

Bryan Yeah.

Anne [laughing]

Bryan Whatever, [mumbling] vote, I don't vote for anything

Anne Oh

Bryan I don't vote for where to eat, I don't vote for what dog breed is better, I don't.. No voting.

Anne [Laughing] No

Bryan Americans, we don't vote

Anne No voting. Just, just.. No voting, just swiping

Simon [Laughing] Just swiping

Anne [Laughing]

Benjamin [laughing] Just swiping

Bryan yeah, yeah...

Benjamin That's why Tinder is so successful

Simon Right

Anne [laughing]

Anne Yeah, just speaking of swiping, actually. We have some questions about that. Because, you talked about you also tried other kinds of inputs.

Benjamin Yeah

Bryan Yeah

Anne What make, what made you decide on using swiping?

Bryan I think, I can probably take that one..

Benjamin Okay

Bryan Ah... The thing that we do the most probably on this is, swipe around, like, I have my phone right here. Uh, its definitely the most visually impactful. I mean, we talked about accelerometer before, we talked about like, basic tapping on thing, and.. what else was it? Oh it was...

Benjamin Yeah, the thing is...

[Benjamin and Bryan talking at the same time - can't make out the words]

Benjamin ...messed around with sounds, too.

Bryan Yeah. Yeah.

Bryan It's just, its not as immediately interactively good, like it doesn't feel good right around. The game feel so to speak, of this toy, because this is essentially what it becomes, a toy, doesn't feel as great. It's like, if you have two different pairs of dices, and one of them has like a really weird kind of grainy texture, that doesn't really feel good to hold, but like the other one is like slick and smooth and it just.. They do kind if the same things, but, one of them is just better to interact with. I mean, we are from USC. One of the first thing i remember doing in class is like, they hand out a bunch of toys and they are like, random like toys and card games and like things, whatever. And we kind of just examined them. And there is something about playing cards, well if there, if they have like a laminate on them, they feel better just to hold and to like kind of bend with, because you know they are not gonna bend completely, unless you do that, it feels kind of nice. I don't know if you've played like, poker, blackjack, or like... But... There's a reason why those are like that. So , I mean. Of course we tried many different game designs, and, things we think were fun, but I mean... A universal language, is swiping.

Anne Mmmm.

Benjamin I mean, 'cause you can do accelerometer, but like, people. were, when we tried that, people were like, "well, how do I hold my phone", like, what's the default position. And when we say, "hey, just swipe" and they are all like "okay" like, "I know how to that" immediately. Like, that's something, I don't have to hold it upside down or sideways, it's like, "no, nope, just, I know exactly how it's supposed to go"

Bryan Just because it's how you hold it all the time. I hold my phone always like this, I don't hold it really like that when I'm checking my text messages, or like that... You know...

Anne Yeah

Bryan ...I'm holding it like this

Anne [Laughing]

Benjamin So, I mean, yeah, it's largely, not replacing a ritual, uh.. Have you ever... have you ever heard that before?

Simon I have. Yes

Bryan It's basically going with a ritual, i mean we have already a ritual, it's very ingrained in us. I, if I'm right handed i will normally be holding my phone in my right hand, and I can text with one hand. My thumb. It's a ritual, it works. Why not, why not use it to kind of go down one avenue or another. So yeah, accelerometer, you don't do that all the time.

Anne Yeah

Benjamin Tapping, and, tapping just isn't fun, and.. Sound doesn't really work. because, it's really loud everywhere.

Bryan [interrupting] People screaming and cheering, yeah. Sound doesn't work.

Simon [Laughing]

Anne [Laughing]

Bryan Sounds in the game, it's just like, you can't hear anything, it's just like, cheering as if your football team is going for a touchdown. Like. It's just eruption. So yeah, no sound.

Anne So, just.. I guess you guys need to be going soon? But just a last...

Bryan Yeah, in a little bit. We can, happy to go to 11:30 or a couple of minutes after, we did start kind of late.

Anne Oh, it's okay. Its just.. like a last question. Like, because. There is many challenges.. challenges in designing crowd games. But what has for you guys, what has been like, the biggest hurdle in a way, or like?

Benjamin Initially, our biggest hurdle was probably the technology. Just, I mean. That's something that you guys are dealing with too. You know how.. You know, it's like, okay, I wanna play a game with a hundred people in a room that's just what I wanna do, and its just like, okay, well how do you do that, how do you do so that in there and having fun, I mean you could play a game, you could play like one of the bigger kind of just like, physical game, like maybe like tag or something like that...

Bryan Capture the flag

Benjamin ..Capture the flag, uh yeah, but how do you do something that is digital, right, where you need a screen, you need them to kind of like, pay attention to something. And then, just by its nature, you need to all to have some sort of input, otherwise, they'll get bored, they need to be playing, otherwise they'll get bored. So.. Definitely, the technology hasn't.. We're.. we're both of us are, we are exploring it, right. So, you, you, your doing your own way, and you are doing your own was so it's just, it isn't, it's not a thing that, is really kind of popular. like. It's not something that people really done. It does some for movie theaters like, 10-20 years ago, like that, that died out for very good reasons. So.. You know.. How do we, how do we take those lessons of what they did, that didn't work, and move forward and kind of do something that.. that does work. So, the technological hurdles just, you know, kind of how, how do we get everyone inside, how do we get them playing, and then.. You know.. The next biggest hurdle was the content. Just, how do we, you know. Now that we have kind of the capability of all of them to do something together, how do we make it fun, how do we make it, you know, like, like make them feel like they are impactful. Like, a lot of the early critique we got was "I don't feel like I'm doing anything" and its like, well, its not you know, you... We cant have you feel that way because it is not fun, so what can we do, just little, little things in our site, that make you feel like you are doing something. And, yeah. Besides those two...

Anne Yeah, what.. what did... what do you do, like, to make people feel impactful?

Benjamin So we have, there is like, two different things that we kind of tried, and that really really worked well. I mean, so for our actual.. So for.. In our games, we have, you know, the little kind of circular piece that everyone is moving to move around. We have it basically, where. For every.. If you started swiping up, there's a little meter, on the, that kind of faces up. That will start kind of like, kind of, filling in. It starts transparent it kind of goes, it goes darker as more people are swiping that way and then it fades out slowly, and then as more people start swiping that way, it goes darker again.

Bryan Kind of like a heat map

Benjamin Like a heat map where people are swiping. But we made it so, where, if one, if everyone is swiping up, and one person swipes down, the down person can still see their input. So, it's. It's kind of like, "hey, I want to see, no one is swiping to the left, I'll swipe to the left, oh hey there is my thing, I did something."

Anne Mmm

Benjamin The other thing that we start, we tried to do with like, scores, like. We keep track of how many times you swipe, right? Just.. You know. It's like, you swipe, we try if we can keep track, it might give you that score at the end. That didn't really work, because more people just said "well, that just mean that I need to swipe more" and that wasn't, that wasn't that fun... So then we kind for transitioned that into like, achievement. So, now we have these achievements where we can kind of keep track of what direction you are swiping in, compared to everybody else. So.. If you, if everyone is swiping up, trying to get to the goal, and it's right above them, and you are swiping down, we can track that and say "well, you swiped against your team, like, this percentage of the time". And, and. And all that good stuff. And then the.. The last thing, one of the last things we did, was just call people out at the end of the game. So we have kind of a way, if you, if you are able to log in with Google or Facebook, we can bring your name up on the screen. Right?

Anne Mmm

Benjamin It's like a random person, in the whole audience, its like, "Hey, hey, you win a price, you, you know, you guys were good, you won a prize." And people love that! They are just like "hey, that persons name is on the screen" and.. If you are in a small group, chances are that you know them. If you are not, you know, it's like, you are like "oh, somebody in the audience here" like, "got in on the screen" like, "that is great, I wanna try, I wanna be the person that is on the screen". We call it the "I wanna be the guy"-feeling. Have you ever been to like.. I don't know if you guys ever heard of like American Basketball games, but, they typically have. Like, in the middle of the game, they will have a time or like.. one of the fans will come down, do like a shot from half court. Which is supposed to be, which is really hard. It's not easy to do. No, pros don't really do it. Its not a thing you do. So, if you make the shot from half court, you get like, you know, money, or you know, prizes or something. And.. It's, it's, the feeling of being the guy who made the half court shot, who, in front of like, the whole audience, is like a really, really great feeling. And that's the same feeling that we, that we try and capitalize on, by putting your name up there, it's like, you specifically, John Smith, you were the best, you, you got the prize, and your name is up here for everybody to see. You get your 15 seconds of fame. You can go to everyone and say like, "hey, I won" you know, like "here's my t-shirt that I got, because I won that"

Anne [Laughing]

Bryan People will do that, people have done that

Benjamin It's crazy

Anne Awesome idea.

Benjamin Just for the chance of being the guy. Like, people do crazy things for, like the contest, like all the time. They fill out like the contest stuff, and it's like, "Hey, send in a video of you", like, "dumping ice water on your head" or, you know, whatever, right?

Anne Yes.

Benjamin And it's like, "great, I'll do that" and, you know, just for the chance to win a.. a price. So, if.. It's impactful because we hand out those prizes, impactful because we call out those people and people feel like, "I have an impact, because", you know, "I'm, I'm up on there and I won" or "Someone else is, and I could be next".

Simon Yeah

Anne Awesome

M Interview with Andy King

The following is an interview with Creative Director of PlayWest, Andy King. The interview is conducted as a written correspondence between Simon and Andy King.

Andy King Hi Simon. Thanks for the interest in K64. It's a concept very close to my heart actually, and has a storyline not just the epic battles. We have proof of concept build with a few classic arcade games but MMLP (massively multiplayer local party), and in the end there will be 3 "Main battles" each made up of 3 different mini games, culminating with all three games PLAYING AT THE SAME TIME. But ultimately - it's the story of some alien tech which crash landed from a military transport.. and the resulting explosion made all the farm animals sentient! They must escape the farmer, the FBI and more!

I often tweet when I'm doing dev on it, so stay in touch and we will have a sneaky new build and some games with 30-50 people playing it together again. When we've gont one we will fire it over if you like!

Simon Stålhandske That would be lovely. So basically I and DJScoretrick have been researching crowd games for the last half year, but unfortunately didn't stumble upon this project until now. We have a list of crowd games up on http://localmultiplayer.com . I'll make sure to add K64 there. We choose to call them crowd games instead of MML (massively multiplayer local games) or whatever else you would call them, because we don't really like the reference to MMO's which we feel are totally different.

Uhm. but yeah. What kind of tech do you need to connect 30-50 controllers to a PC? I'm guessing it's not that trivial.

Andy King Nice, I'll use that ref as MMLP never sat nicely! Tech wise, it wasn't trivial at first. We've cheated recently and moved to a pad-splitting method a-la micro machines. Not because it wasn't possible, but that PC USB standards are so poor

Most laptops couldnt get above 8-12 controllers, so we moved from 360 pads (IRQ heavy) to generic cheap 2$ snes pads (v. light - but v. unreliable!) and shifted to sampling raw data from the pads through a wrapper we called "control 64" which in essense bypassed all of unity and windows low-level.

We found one machine that got to 92 connected devices

which pad-split would have been 180!!!! But back then had an *k resolution which downsampled back so we have massive play space.

Our recent changes have been to balance gameplay systems from 8-32 players, and accept that 64 player is CARNAGE. But it also contains (hopefully) a lot of retro love and charm, but still doing things like player select in a clever / intuitave way.

Having run 48 player parties, it's INSANE without a master of ceremonies one controller to rule them all - so I made it so everyone can play even in the menus. The plane is the only menu in the game and it's the same plane you steer when you die. It's even called "crowd bombing" inspired by bomberman on the saturn (definitive version)'s "mad bombmer"

everyones inputs steer the plane, and everyone can drop one bomb on those players still living!

Anyway.....

:D

Happy to be involved, as every game we do @playwesthq is multi! And my research is doing big things with games in europe too

or silly things...

Simon Stålhandske Jeez you write quick :D

But thanks for all that info. That's super useful as I thought it was completely impossible to connect that many controllers.

BTW. Do you usually have a master of ceremonies (we call them game hosts) when you playtest the game?

Andy King Sorry, had to dash out and finish a bid document so was a bit turbo-charged there!

So depends on the game status.

If it's just a core loop and we want to see if people understand without written instructions we don't use host, but for big events (like EGX rezzed) we do use a host as it can cover the imperfections.

In the case of "The Breakfast Club" people just think it looked like i am bread as we did bread level first.. so a host was useful to pull people in by saying fun things, and to "tap dance" because game wasn't finished!

Simon Stålhandske Ahaha. Yeah, that makes a lot of sense. We have similar experiences, especially because we are mainly testing unfinished prototypes. Does your game host have any kind of control over the game? Like starting, stopping, changing scenes, calibrating things on the fly?

Also I'd be super interested to hear if the players have any difficulties finding their character on the screen when there are so many players?

Andy King Our first system had every option you can think of, but only one level, we'll one and a half! But as we balanced gameplay for 2-64 players much of this went. However, we still have loads of debug options!

And as for player finding.. I designed a simple way to find yourself

Everyone thinks will be impossible to find, but finds quickly as we use special highlight system based on some research I did using a game called theif town, which actually plays on the fact you look the same as everyone else.

Typically once you realise what the face buttons do, you find yourself in a couple of seconds, even with random drop on

Gameplay isn't so affected because the lobby is diagetic

And you already work out how disco system works

Basically I saw so many ppl saying, this many players can never work..

So k64 is a mainstream game with silly game mechanics of your favorite games all happening at once.

MB1 features tron light bikes, Atari dogem, and bomberman at the same time in the final round, but our bombs are sentient so will cause you many problems! Currently working on planes for all players and farmer npc

Simon Stålhandske Cool, thanks Andy. Thanks for answering all of my questions, that was super kind of you. :)