The background is a deep blue gradient. It features four large, concentric circles in the corners, each with a dark blue outer ring and a lighter blue inner circle. A series of small, bright white and light blue dots are arranged in a path that starts from the top right and moves towards the bottom left, creating a sense of movement or a trail.

Steering User Behaviour through Enticing Elements

*Applied to jump-based
electronic playgrounds*

Richard Kok

Supervisor: dr.ir. R.W. van Delden (Robby)

Co-supervisor: dr.ir. D. Reidsma (Dennis)

UNIVERSITY OF TWENTE.

Abstract

In this paper two games are created, both were projected on a life size playground that we could use where four Kinects track user positions. The aim was to influence play behaviour through enticing visualizations. The visualizations are enticing as they only act as embellishment and do not interact with the core game play, as such reacting on those elements did not influence the players' score. For the first game it was tested whether it is possible to steer positions of players by seducing players to walk towards a colourful spot in the playfield, this was not possible as the players did not observe the enticed parts of the background. Another goal was to let the playground detect jumps, but the playground was unstable in detecting jumps as the system confused the movement of the players with jumps. The feedback of the players during and after the playtest however did lead to new requirements which were applied to a second game. In this new game the players stood on the same position during the game for better jump detection. Whenever multiple players jumped simultaneously neon-like elements formed a connection between them to entice synchronized jumps. The players did not notice the enticing elements. However jump detection worked properly and the applied algorithm can be implemented in future electronic jump-based games. Limitations were that students from either primary or tertiary education were used for playtesting which makes it hard to foresee how a broader public will play the game. Further work could try to let players run over the playground for maximal physical exertion, the game could entice players to stand still before jumping so that jumps will be properly detected.

Summary

Influencing behaviour can be done in various ways, to place importance on autonomy we opt to choose the subtle way of enticing behaviour instead of requiring it. The context upon which the enticing elements will be applied are interactive jump-based games, driven by an existing system of four Kinect track positions and one beamer projects the game on a lifesize playground. Jump-based games are chosen since they require substantial active effort, not only physically but also cognitively due to the coordination it requires. Finally it benefits players on the social front as players urge others to join and when playing they need to attune their behaviour to each other by means of either verbal or physical communication.

An initial play concept was derived by means of combining eight playground games with eight jumping rope variations, resulting in 64 newly derived games. From these games, criteria have been imposed by means of a three-point Likert scale (simplicity, scalability, variability and originality) to trim the games down to eight games with the highest rankings. From these, one winner is chosen by also taking the requirements with respect to enticing elements into account.

The main requirement was that the game should have both passive and active players such that passive players can be seduced to action through the enticing elements.

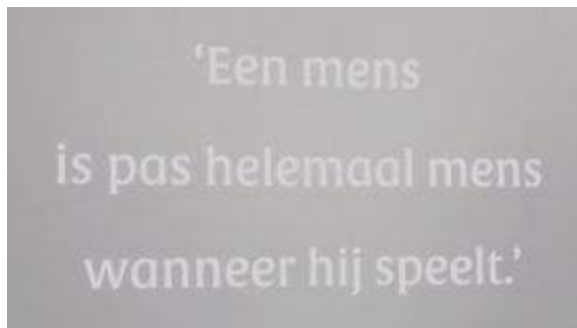
The resulting game was tested on a primary school class with children aged between seven and eight. Their play was curious-natured as well as explorative which resulted in them making connections between the game elements that were not intended by the designer, nonetheless this creativity did lead to the children having fun whilst playing. However the game rules and the enticing elements were generally not observed by the young players and as a consequence the outcomes lead to new requirements but no statements could be made about the unobserved enticing elements. Their goal to aspire players towards active behaviour by guiding them towards the location where the action peaked was not obtained.

The main new requirement resulting from the playtest was that players should stand on the same location for the whole game such that the jump recognition was optimized. Whereas the former game used the standard deviation to track jumps, the new game should set its jump threshold according to the average height. The new concept used an enticing effect in the form of neon-like elements that connected the platforms upon which players stood whenever both players jumped simultaneously. During the playtest, done with students of the University of Twente, the jumps were tracked with fairly few errors and although the accuracy would not be enough for scientific measurements it was good enough for proper gameplay. However, the enticing elements were not visible enough to adapt play behaviour and as such the goal to increase the ratio of simultaneous jumps versus normal jumps was not obtained. Summarized, the games developed in this project showed engaged gameplay but the enticing elements were not present enough to alter play behaviour in the context of jump-based games. Jump detection was sufficient for proper play of jump-based games. Future work could focus on a game where

people can walk around whilst they are urged by enticing elements to stand still when jumping for optimal recognition of jumps while they are not bound to a specific position. The enticing elements should be created so that they are clearly visible by amongst other reasons having a good contrast against the background.

Acknowledgements

Many thanks for Robby Van Delden for being the backbone of this project and for applying an enticing, instead of forcing, method to steer my behaviour with respect to this project. At times however, Robby shifted from enticing towards requiring so that the paper as a whole would benefit from it. One of the most underrated characteristics of a teacher is to be someone who inspires and I happen to think Robby has done just that. Also thanks to Dennis Reidsma for being the project leader for a project in year two of my study Creative Technology, where our team created a game that was used on the same playground setup as this project was played on. The feedback for that project was always direct and clear and I am sure that seeds of it have been continued in this paper.



Friedrich von Schiller

Page	1	3	14	17	30	35	40	45	46	49
	1	2	3	4	5	6	7	8		
	INTRODUCTION	STATE OF THE ART 2.1 Enticing Strategies 2.2 Traditional Games 2.3 Electronic Games	IDEATION 3.1 Ranked Combination Matrix	SPECIFICATION 4.1 Concept Development 4.2 Jump Recognition 4.3 Initial Concept 4.4 Playtesting 4.5 Conclusions	REALIZATION	EVALUATION 6.1 Participants 6.2 Methods 6.3 Measurements	DISCUSSION 7.1 Development Process 7.2 Expected Results 7.3 Boundaries of Enticing Elements 7.4 Project Limitations 7.5 Future Work	CONCLUSION	BIBLIOGRAPHY	APPENDIX

	A	B	C	D	E	F	G	H	I	J
	Combination Matrix		Car and Turtle Concept		Results Playtest 1		Consent Forms		Jump Simulation Code	
		Selection Matrix		Jump Simulation		Jump Graphs	Synchronized Jumping Code		Data Mangling Code	



1 INTRODUCTION

This study seeks to investigate whether aesthetic elements can prove sufficient to steer behaviour. With respect to the importance of autonomy, this steering will be done by means of nudging, that is to say that the exerted influence is non-forcing and the subject has the freedom to ignore the elements that entice him to change course. For this to be of practical use, there has to be an incentive to change behaviour.

The behaviour we want to change is to make it more likely for passive children to play. This is especially important for games where less skilled players are out of the game quicker due to rules that punish mistakes, where on the other hand those who are skilled make fewer mistakes and therefore get more playtime on average. According to Schell (2015), one of the reasons to be aware of varying playtime is that players might or might not benefit from increased confidence of being able to beat the game by having incremental successes. As such, the skill gap can become bigger as more games are being played, where an escalation of confidence and success is contrasted with players who are battling with discouragement and losses. Schell indicates how this positive feedback loop (positive as changes are amplified, whereas a negative feedback loop would lead to an equilibrium) is enforced by the observation of psychological studies that punishments are less effective for reinforcement learning than rewards. As a result, the loss of confidence and low motivational effectiveness of punishments makes it difficult for less skilled players to keep up with their more skilled co-players who get more and more confident, motivated and skilled as time pass by.

To make players change from sedentary to active behaviour as a way to bridge the gap between skill levels, the game can invite them to play by nudging elements that act independent of the game's core functionality. Another option could be to create a more aesthetically pleasing environment as more people play in an active manner, this reward tied to activity might impose group pressure directed from active towards passive children as more players means more visual responses. With respect to these options for using either nudging elements or rewards for cooperative play, the research initially focuses around the question if enticing elements can make it appealing enough for sedentary players to actively participate in jump-based games. The last part resolves around the appearance of visuals, whose occurrences is linked to the cooperation of players, and accordingly it should elicit more active play through group pressure.

By using either of the strategies, more players will reap the benefit of active play as in both the nudging and visual reward the passive players are encouraged to join the active players. With respect to the target-group, not only does it involve individuals who are inclined to show passive behaviour unless participation is stimulated but also potential players in general. To have more players in general participate in active play is beneficial since unenergetic activities as watching television and playing behind game consoles are on the rise (Singer and Singer 2005). When people endorse in active play, research shows that it can lead to improvements of (a) physical skills, such as coordination, (b) cognitive skills, for example problem solving (c) psychological attitudes, in particular self-esteem and self-efficacy and finally (d) social factors, for instance peer support and interactions (Lieberman et al. 2011) .

The goal of this study is twofold. One objective is short termed, namely the creation of a game where children are more inclined to reap the diverse benefits of active play. The second objective is to provide guidelines for the use and development of enticing elements to subtly steer player behaviour. This study starts from the premises that behaviour can be influenced to the extent that the targeted early-out-of-the-game child did have an internal motivation to play, such as having fun, but lacked the external incentive such as an open and inviting attitude from those already playing. The following chapter, the State of The Art, shows (a) what has already been done in the field of steering, and more specifically enticing, physical play behaviour; along with argumentations for the relevance of applying these methods in this project (b) a structural analysis of traditional jump-based playground games; including viewpoints on the positive/escalating feedback loop which influences play negatively (c) examples how traditional jump-based playground games are currently augmented in the domain of digital playgrounds.

To build further on these preliminary findings, a mix-and-match method is applied on the traditional games and as a result the initially limited set of play methods diverges. The resulting larger set is in turn filtered by an assessment on how fit the individual games are for application of enticing methods. Finally, a single concept came out that proved, with respect to the requirements, to be the optimal choice; to be further formatively evaluated on children by means of a playtest to extract issues encompassing either the effectiveness of steering elements, game mechanics, the visibility of elements or the ease at which the concept is understood. From the observations and other results of the playtest, a sharpened concept came out that was in turn evaluated, this time with more focus on the enticing elements. This report concludes with a view on possibilities to continue where this project left out, obtained guidelines to guide the future work as well as what I have learned and what other interested parties could take home from the established work.



2 STATE OF THE ART

2.1 Enticing Strategies

Steering by means of enticing elements forms a central part of the research, in order to get a good grasp on the possibilities that can be implemented, it is useful to first take a closer look on existing steering strategies. One of these strategies was skill-balancing, an adaptation of gameplay for an interactive tag playground (Moreno et al., 2016), based on cues sensed from each player, which resulted in different experiences for each game round. The players were individually followed by a projected circle that would lead to a change of role when a tagger's circle touched the circle of a runner. The skill-balancing meditation adjusted the size of the circles such that those having a hard time were helped additionally by adjusting the circle size, dependent on the context of the player being a tagger or a runner. A year later, the circles also had the possibility build in to become increasingly more aesthetic as players collected particles that were projected on the playground (van Delden et al., 2017).

These interventions to normal play did not insist nor require change of behaviour, furthermore the enticing strategies were detached from core mechanics. This decoupling from core mechanics can be compared to Thaler and Sunstein's (2012) more political-minded description of nudging, "Nudges are a way to change behavior in a way that is not (significantly) related to the users' economic incentives and does not obscure options". Consequently, enticing and nudging strategies show overlap in the sense that they are not overpowering while having an underlying motivation to gently change behaviour. Following this line of thought, Verbeek (2015) places them under the umbrella term of being a weak influence due to their persuasive, instead of coercive, nature. The author places value on subtle forces that neither require nor insist on change of action, since it respects autonomy, which is not put at stake with a nudging approach. Consequently it results in a distinction between the strong forces of requiring and insisting, that contrast against the weaker force of enticing, combinations of these forces are also exemplified in the playgrounds of Moreno et al. (2016) and Van Delden et al. (2017).

The argument for autonomy by placing the final decision power at the individual level is opposed by Berlin (2016) who mentions that some parties believe that the interest of the whole community is more important than that of the individual. As a result authority should steer the behaviour of the individuals. Monitoring behaviour and using strong steering forces from a centralized monitoring is argued by Berlin to eventually benefit the collective of individuals more than if they all were to decide their choices separately. This phenomenon is comparable to the workings of game theory where the best overall result is obtained when each decision-maker anticipates the reaction of those affected by decisions, due to the benefits of having a central system do the anticipation and steering, a centralized monitoring and steering of choices can help to steer actions and in doing so guide towards an optimal outcome.

This might seem an argumentation that is distant from playground games, but in essence even the difference on the macro level such as communism versus capitalism can be related back to technological implementations. For example a game computer that processes data from multiple sources and in turn adapts the playfield accordingly uses a data processing strategy that can be compared to centralized powers observed in a communist state, at the other end of the spectrum setting your alarm clock is a more distributed form of data processing mimicking capitalism on the micro level. More directly related to game mechanics, Schell (2015) adds to the argumentation that providing too much decision power to the players tends to fail in practise. As players benefit from winning a challenge but the value that they place on recognition leads them to change the game such that they can win it easily and thus avoid the challenge that might benefit them.

Verbeek steps in between the boundaries of society as a whole and the playground, by mentioning that designing technology in a sense equals designing humanity. For the reason that any technology shapes human actions and experiences which have an impact that should be understood in ethical terms. With respect to ambient environments such as the interactive tag playground as well as fridges since they adjust its light and temperature, they are not just a background for our existence but they are immersive as well, providing an interactive context. In doing so our actions and perceptions are changed by the technology, with rules for desired behaviour and perspectives can be embedded into them. From the different perspectives on mediating technologies it can be concluded that the game mechanics of the playground should enhance social and physical play dynamics whilst making sure that negative perceptions and actions are minimized.

2.2 Traditional Games

Common playground games

Now we know that changing behaviour can be a beneficial addition to games, to be considered along with its implications for the flow of the game, it is useful to take a step back and learn more about rudimentary games that are not yet externally influenced. Specifically, this section takes a closer look at a selection of popular jump-based playground games that are being played without the interference of technology, for some examples the date of origin extends far back. For example the popular game Simon Says was already played by the romans (Glover, n.d.). Considering the influence of the roman empire, it is safe to assume that this game has been played across a variety of countries for an extended period of time. Another game with Roman origin is Hopscotch, where either the soldiers or roman children were the first players of the now ubiquitous game, albeit in western society. Whereas some games have been around for long that does not imply that the manner in which they are being played is rigid.

It can be speculated how the environment of upbringing leads children to give their own twist to the skeleton of games such as tag, to enhance it to for example TV-tag.

A preference for play methods, based upon the living area is confirmed by Singer et al. (Singer, et al., 2009). The researchers monitored differences in play for 16 different countries on five different continents, between children living in industrialized and rural areas. Their study concluded that pretend play occurred more often in urban environments whereas painting, drawing and toy play, which can be categorized as creative activities, were found to be more popular for children growing up on the countryside. How romans played Hopscotch is also likely to differ from how it is currently played. It is safe to assume that time, demographics and geographics are all factors exerting influence on the attitude towards games.

For this project, the play of western children in modern times is dissected further. Williams (2009), a copywriter of the popular technology inspired magazine WIRED, reminisced the playground games she used to play as a child. Table 1 shows a selection of the games she used to play, which shows more rigid and ubiquitous games such as Simon Says, Hopscotch, tag and jumping. Additionally it shows activities that require an awareness of modern tools such as TV tag, where children need to stand still when tagged until they shout the name of a TV show, however a child growing up in areas where watching television is not the norm could easily replace a TV show with for example an animal name. Likewise freeze dance which is originally played by having people run around until the radio stops playing, can be played equally well by having someone sing a song.

Name	Description	Tools	Players
Hopscotch	A pattern of numbered shapes is outlined on the ground. An object is tossed into one of those shapes whilst the jumper has to collect the object while jumping on the shapes in between. Optional shapes can be hopped through without any penalty.	Chalk or permanently marked hopscotch areas.	1+
Marbles	A circle is drawn with marbles in it, each one belonging to a specific player. The current players gives one marble an initial acceleration such that it knocks an opponent's marble out of the playing field.	Marbles, chalk or any other method to lay out a circle.	2+
Simon says	One person is Simon and starts by saying "Simon says", followed up by a command that the others must perform	None	3+
Tag	One person is the tagger chasing others around with the goal to tag them, upon which they become the tagger.	None	3+
Freeze tag	If a person is tagged, he stands still until he is tagged by another person besides the tagger.	None	3+
TV tag	A variation of freeze tag where the unfreezing happens by calling out a TV show, the name of a show can only be called out once.	None	3+
Crack the whip	All players hold hands in a line, a person at one end runs around changing directions upon which the movement propagates as a wave through the chain, where the goal might be to let people on the other side loose grip on the chain	None	6+
Freeze dance	When the music starts everyone dances, when it stops everyone freezes their position, otherwise they must step out of the game.	Music player with speaker	3+
Jumping rope	2 people spin a rope whilst one or more people jump over it with their feet close together. A player could also spin the rope by himself whilst jumping.	1 rope	1+

Table 1. Traditional playground games.

Jumping rope alternatives

With respect to one of the objectives to have more children reap benefits of exergaming by means of jump-based games, jumping rope seems an ideal candidate due to its simplicity in concept and potential for physical effort. The popularity of skipping, a jumping rope variation, amongst the fitness industry confirms its applicability for the argument of exertion. Matt Hopkins, a former teacher, competitive speed jumper and jump rope coach is well aware of the variety of games that children can play with merely a rope, a selection of these adaptations of jumping rope ("12 Fun Jump Rope Games for Kids," n.d.) are described in Table 2.

The games show differences concerning the minimum amount of required players, as well as their competitive versus noncompetitive nature, which has not been quantified further since the conversion to digital play can still alter these characteristics.

Name	Description	Tools	Players
Skipping	The difference with normal jumping rope is that a shuffle-like movement is performed whilst hopping, where one legs stays beneath you whilst the other leg moves forward. The feet stay close to the ground.	1 rope	1+
Action jumping	The jumper has to perform the commands that the rope-twisters give him while mid-air	1 rope	3+
Double dutch	Two long jump ropes spin in opposite directions.	2 Ropes	3+
Chinese jump rope	A rope is stretched out by the feet of two people, with the jumper(s) jumping over the rope with a predetermined pattern	1 rope	3+
Cat and mouse	Two jumpers enter at one end of the rope, perform three jumps, and start over. One of the player is the cat and he can tag the mouse if he or she is preparing to jump or if the mouse makes a mistake whilst jumping.	1 rope	4+
Snake in the grass	Two people hold the ends of a rope, keeping it flat on the ground while shaking it such that it performs wave-like motions. Other kids jump over it	1 rope	3+
Helicopter	One person rotates a rope over the ground with his own position as pivot point. The jumpers jump over it	1 rope	2+

Table 2. Jumping rope variations.

2.3 Electronic Games

General

A defining feature of playground games are that they are played within a social setting and children can easily join or leave the game. Furthermore they are relatively simple, in the sense that a minimum amount of tools is required, if any, and explanation of the rules and methods can be done quickly. This creates a low barrier for play which has been picked up by institutions that saw opportunities in the digitalization of games with these typicalities. The intuitive aspects of the games allow for absence of an instructor, while the fun is not diminished.

Companies that use digital projected games, responding to motion, are amongst others EyeClick ("BEAM" n.d.), TouchMagix ("MotionMagix™", n.d.), Vertigo ("living floor" n.d.) and LumoPlay ("Lumo Play" n.d.). These projected games have applications for fitness, waiting rooms, low-barrier entertainment and physiotherapy. Due to the simple requirements of having a projector and additionally a sensor to measure player position in the case of an interactive environment (used for example with Vertigo's entertaining living floor as shown in Figure 1), the products allow a large amount of playable games since the hardware does not need to change. Despite the variety of games, the elements and game mechanics of each game are fixed and often the sole variable is the position of the players.



Figure 1. Vertigo's projected games are projected in amongst others shopping centers ("living floor - interaktive Bodenprojektion," n.d.).

Beyond commercial games, there are also more research-oriented translations of traditional playground games to the digital domain. The game "Interactive tag" (van Delden et al., 2017), investigated behaviour in regards to required aspects (for example power ups such as shields) and enticing elements. The enticing aspect was created by adding particles that can be collected, upon which their collection resulted in positive, implicit feedback by means of the circles that follow the player show more visual complexity (Figure 2). The researchers hoped to bring the tagger and runners closer together, as the tagger and runners normally avoid each other by staying either close to the center or borders of the playfield respectively. This enticing strategy influences game dynamics, social factors and activity levels as the increased tension and decreased proxemics between runners and taggers increases the risk of being tagged and demands faster and more accurate adjustment to other players. As the strategy uses weak forces, players can choose to either ignore or act upon them, dependent on their confidence of collecting the particles without being tagged whilst finding enough satisfaction in the visual feedback. Due to the available option, players can tune the difficulty of the game such that they can end up in a flow state where the degree of anxiety and boredom are attuned to each other (Figure 3). The takeaway is that alluring methods can be applied to let players cocreate the play environment whilst not allowing interference with the carefully considered core game mechanics.

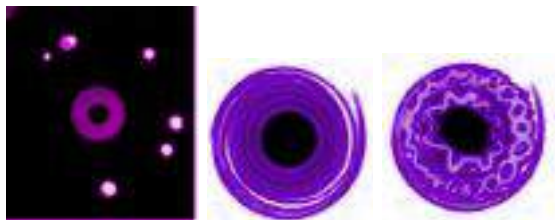


Figure 2. Upon collecting particles (left), the players circle get more complex (van Delden et al., 2017) .

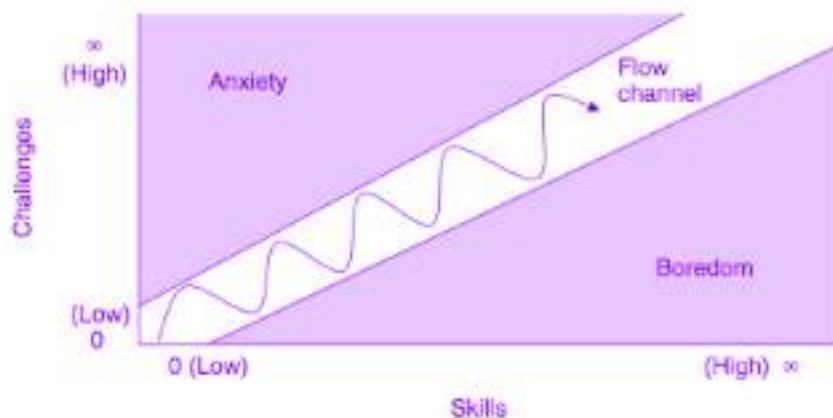


Figure 3. Flow is obtained when challenge is tuned to skills.

Jump-based

To switch from electronic games to jump-based games, additional difficulties arise as not only changes in horizontal position but also adaptations of player height need to be measured. Moreover a general increase in required equipment can be the case as will be highlighted further on this section. Furthermore, commercial games often provide a large plethora of playable undertakings per platform due to their customizable nature and a larger pool of games to base the concept around, for example whack-a-mole, ice-hockey and soccer. In the case of jump-based games, they seem to be inspired by either regular jumping rope or hopscotch. Despite these complications, jump-based interactive playground's show potential for contributions to a variety of research domains.

Examples of jump-based games include Magic Hopscotch (Figure 4, left), with hopscotch played in a room where the room itself is projected on the wall whilst it slowly turns into a forest. Experiences and reactions of the blend between reality and virtuality lie in the interest of Human Computer Interaction studies. Another example is Hopscotch for Exer-learning (Figure 4, right), where the pads contain letters of the alphabet for increasing the vocabulary of children in a playful manner, which can stimulate the modernisation of education.



Figure 4. On the left, the hopscotch markings is present in both the room as well as the projection ("Sky Blue" 2017), the right image shows a child who jumps on pads to activate letters or numbers to complete english sentences (Lucht & Heidig, 2013).

Another project is named GlowSteps (Figure 5, left), with embedded pressure sensors and LED's that stimulate movement through enticing color combinations that invite to action.

Potential domains include the fitness industry, education and HCI.

Intelligent trampolines (Figure 5, right) has 'satellites' spread around the protective foil that track the jumpers and give feedback through LEDs dependent on the played game. It is created with the philosophy that play equipment should work even when the technology is shut down.

Games are aimed either at energising, encouraging tactic play and reacting physically to the rhythm of music. For the trampoline, multiple jumpers stimulate activity and social dynamics whilst also increasing risk of injury, most notably for the smaller kids. Another project that uses various sensing equipment is Multi-Jump (Figure 9, left), where the social interaction is observed whilst jump-roping over a distance. Equipment includes amongst others a projector, camera, acceleration sensor, pressure sensors and led-strips. Sociology and telemedicine might benefit from the communication over distance whilst doing physical exercises.

The games mentioned in this section are useful for different research domains (in particular HCI) as well as industrial fields (for example the fitness industry). However despite the variety in measured variables such as vertical position of the player and exerted force on the pressure sensor, the elements remain fixed and do not mediate dependent on the skills of the player nor is the player's data used beyond immediate feedback. With the exception of learning english with hopscotch although the feedback for the game is only visible on a screen and no other players can join.



Figure 5. The jump patterns of children is motivated by the colour patterns of the tangible tiles (left) (Rijnbout, Graaf, & Schouten, 2015). The center image shows sensors on the trampoline foil that track jumper positions and adjust their LED's accordingly to signal the current game rules (Karoff, Elbæk, & Hansen, 2012). On the right Different equipments working together to allow jumping rope over distance (Denic & Charalambous, 2007).

In other projects the Kinect is involved to monitor jumping, thereby increasing the correspondence to this project. The game shown in Figure 6 on the left displays the jumping rope on the screen and the timing of the jump is tracked by Kinect's and processed by the computer. It aims to enhance cooperation between children by means of full-body multiplayer interaction. Several strategies are used to let the players coordinate their jumping dependent on each other. For example, if one player jumps, his or her co-player should too. Likewise the kinect is involved in a projected jump-based playground, namely Clemens Grunewald's variation of the game 'the floor is lava' with 'capture the flag' mechanics" (Figure 6). In the game, players were assigned a time and had to jump on a stone, stand on it until and changed color and thereby the players had collected the stone. Standing too long on the stone would deactivate the stone upon which the games rules insisted that they return to a specific position for a recharge. The examples use jump-based playgrounds whilst using the Kinect as monitor device however neither of them use enticing elements to steer behaviour.

Jumping in video games as a result of jumping in real life, has encountered commercial availability by means of the Microsoft Kinect. Bolte et al. (2011) mention how jumping through games allows for faster travel of distances without the use of teleportation. However they also note that walking is a more basic and intuitive way of moving and therefore jumping should be an addition instead of a replacement for walking. The authors conducted a preliminary user study with 11 participants to evaluate jumping (JM) on a digital display while being triggered by physical jumps, with real walking (RW) and teleportation (TP). According to the results shown in Figure 6 on ease of learning, ease of use, satisfaction and effectiveness, jumping scores higher than teleportation, with effectiveness being the only factor to rank higher than walking. Nonetheless, considering the importance that jumping can have with respect to game mechanics, the differences with real walking are rather minimal and jumping can be a good addition with positive effects on gameplay.

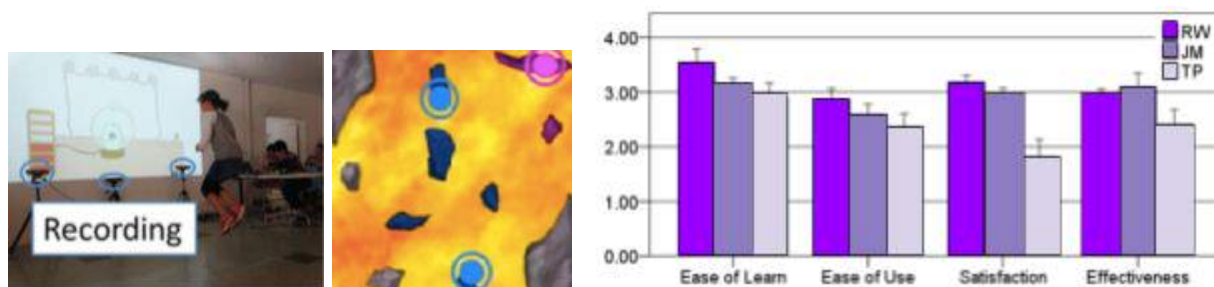


Figure 6. The left image shows a girl who reacts to the displayed rope, upon which a set of Kinects measure the jump.(Sakai et al., 2016). Clemens Grunewald's digital adaptation of the game 'the floor is lava' is shown on the right (van Delden, n.d.). The right image shows Human factors and effectiveness of jumping through games (Bolte et al., 2011)

Besides looking at forms of active play with physical jumps as a component, it is also good to look at how jumping can be detected by the Kinect to provide feedback to such interactive systems. To start of with, the Kinect has a RGB and depth sensor, to extract jumps from the data the Kinect body tracking software can be used. To determine whether a person jumps, 20 vertices that resemble skeleton-based positions are provided by the Kinect for motion monitoring. To acquire these data points the Kinect uses the coordinate system (Figure 7) with the line in the direction to the horizon as the Z axis, the width of the horizon as the X axis and the y axis as an extraction of a line from the bottom to the top of the Kinect.

Huang et al. (2016) use an Unity plugin to bind the skeleton into a running model on the computer. Upon which they determine the action that the model makes by means of the relative position of these sample points. Huang et al. (2016) use the algorithm shown in Figure 7 to determine jumping. Whenever the head, left and right knee articulation points exceed a threshold on the vertical exas, the action of jumping can be identified.

Researchers have also used the skeleton points provided by the kinect to identify the preparation for the jump (peak flexion) and the landing (initial contact) (Gray et al.,2017). For the preparation, the researchers measure the distance between the knees as well as the distance between the ankles. This accurate measurement was obtained by using a controlled setting and providing the participant instructions for the ideal proper jump beforehand. A more rudimentary measurement was the initial contact with the floor, which was calculated by finding the first frame where the velocity of the ankle and foot joint decreases. Based on this, the knee position is not possible to track with respect to the particular hardware setup that is intended for this project (Figure 8). However tracking the head position might suffice and should be possible.

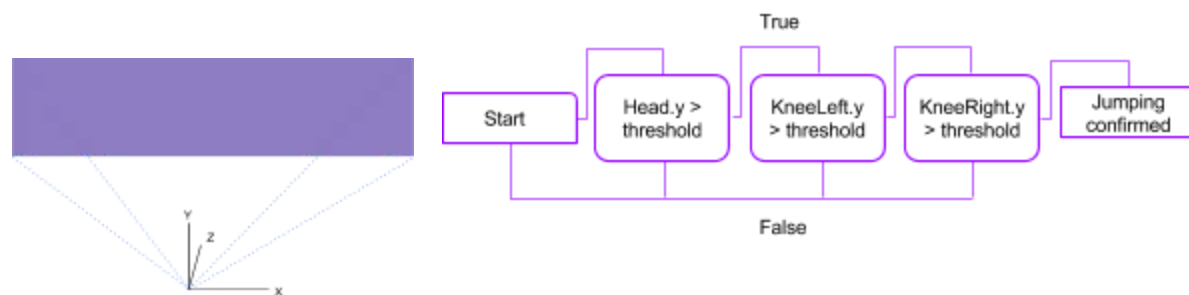


Figure 7. Kinects coordinate system (left) and the process for confirming whether an observed person jumps (Huang, Yu, Si, & Lv, 2016).

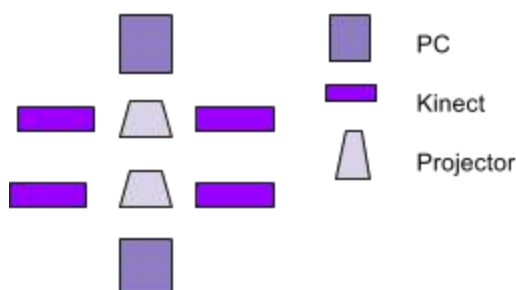


Figure 8, four Kinect's and two PC's and projectors are attached to the ceiling and work together to monitor, process and display data.

3 IDEATION

3.1 Ranked Combination Matrix

Table 1 of the State of The Art showed seven jumping rope variations whereas Table 2 described nine common playground games. Combining these traditional playground games with the jumping rope variations lead to 63 new games that have been described in Appendix A. To narrow down to a game, that fitted well within the objective of the project to investigate the effect of enticing elements, it is useful to have a selection based upon common criteria based upon the state of the art. A Three-Point-Likert Scale is used since it is applied to general concepts, a ranking of 1 is a low score, 2 an average score and 3 is a good score.

These criteria are (a) simplicity, how easy is it to grasp the concept such that children can play without requiring instructions, (b) scalability, can children join and leave the game without affecting the flow of the game, (c) variation, for more enjoyable play and (d) originality, to rank the creativity in concept and in how far the concept differs from existing games. The ranking of the 63 combinations is shown in Appendix B, From these rankings, the combinations with more than 10 points are shown in Table 4, the exact score has been left out since a concept with 10 points might seem better fit after further inspection than one with a score of 11 or 12. For correct interpretation of the used symbols see the legend of Table 3.








 A player	 Moving objects	 Objects that can not be touched	 Platforms upon which the player can stand.
 A pivot point for angular rotation.	 An event has occurred that triggers an action.	 The player should jump over the border without actually touching it.	

Table 3. Legend for the visualization of play concepts.









Jumping rope alternative	Playground game	Visualization	Explanation
<i>Chinese jump rope</i>	<i>Hopscotch</i>		Moving platforms make it hard to jump over the border to collect points.
	<i>Freeze dance</i>		The platforms can change state suddenly. If so, the player can not touch the platform nor jump over its border.
<i>Snake in the grass</i>	<i>Marbles</i>		Both the spinners and the marbles that are propelled by the spinners should be avoided.
	<i>Hopscotch</i>		Standing on platforms provides points, however the platforms that have been touched by a spinner will subtract points until the platforms reset after a given time. The spinners also act as independent objects that should be avoided.
<i>Helicopter</i>	<i>Hopscotch</i>		When a child jumps on the platform, the helicopter- spinner becomes smaller. The goal is to remove all platforms upon which the game will be reset.
	<i>Marbles</i>		Not only should the helicopter spinner be avoided, periodically the spinner lets go of one of his marbles that become game objects to be avoided consequently.
<i>Action jumping</i>	<i>Crack the whip</i>		The rope hovers around the playground while changing color. Whenever a player jumps on a platform with the same color as the rope, the rope breaks into smaller pieces. When the rope is small enough the players win and the game resets itself.
<i>Cat and mouse</i>	<i>Marbles</i>		The player should try to jump over the border to collect points while avoiding the marbles that are spinning around.

Table 4. Visualization of combinations scoring more than ten points.

By taking the average of the cumulative score of jumping rope alternatives for each playground game, rounded to one digit as shown in Table 5, patterns emerge per set of jumping rope variations. An example is the low score on simplicity for cat and mouse score, since it involves rules for tagging as well as jumping, whereas regular jumping rope uses much simpler methods. As for the total score, snake in the grass scores highest, most notably due to its ease of joining in since the length of a 'snake' as well as the amount of 'snakes' is independent of the amount of active players.

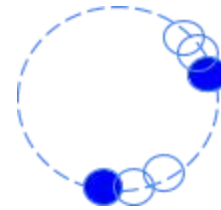
Playground game	<i>Simplicity</i>	<i>Scalability</i>	<i>Variation</i>	<i>Originality</i>	<i>Total</i>
<i>Cat and mouse</i>	1.8	2.3	2.3	2.4	8.8
<i>Regular</i>	2.3	2.3	2	1.6	8.1
<i>Action jumping</i>	1.6	2.3	2	2.4	8.3
<i>Double under</i>	1.7	2.6	2.1	2.3	8.7
<i>Chinese jump rope</i>	1.9	2.4	2.3	2.6	9.1
<i>Helicopter</i>	1.7	2.6	2.4	2.6	9.3
<i>Snake in the grass</i>	2.1	2.8	2	2.5	9.4

Table 5. Ranking of jumping rope variations.

To make a final selection of combinations, it is useful to lay them against the concepts of insisting, requiring or enticing behaviour, as mentioned in the State of The Art. With regards to the scope of this project, insisting will be left out of the equation to keep the game relatively simple and to minimise the pressure for children to perform actions. This leaves requiring, the explicit steering of actions while not having it deemed necessary for proper play, as well as enticing strategies which involve subtle stimulation of desired behaviour. In doing so, the game is allowed to have a reward/punishment system as well as the possibility to apply alluring elements. To entice behaviour, there should be possibility for player to both be passive as well as active without consequences for gameplay. If we were to zoom in on the combination action jumping and crack the whip, this would not be possible since the sweeping of the rope over the complete playfield will lead to negative feedback when children choose to stand still whilst the rope approaches. Likewise negative feedback is possible with Chinese Jump Rope - Hopscotch due to the always present possibility of collision with moving platforms. Composite games that allow for passive behaviour without feedback are freeze dance and marbles. Due to similarity with the Floor is Lava game, as shown in Figure 6 of the state of the art, the game Cat and Mouse - Marbles remains, which will be the game further develop in the specification. It is important to note that the combination scored a 3 on simplicity, which is the most optimal ranking of the options 1-3, and therefore it does not follow the trend of Cat and Mouse games to have rather complicated constructs as shown by the low score of Cat and Mouse games in Table 5.

4 SPECIFICATION

4.1 Concept development



Brainstorm

Figure 9, the winning combination.

From the previous Chapter, the combination Cat and Mouse - Marbles, shown in Figure 9, seemed to be the optimal choice with regards to the possibility for enticing elements and a proper distribution of active and passive gameplay. Note that there can still be a winning and losing condition, but the fact that the whole team wins or loses simultaneously makes it so that all players have equal playtime and can play continuously if desired.

The game motivates players to jump over the borders whilst avoiding the spinning elements, which circulate around the border. Thus we can decompose the game into a circular track that should be jumped over as well as elements that are speeding and can not be touched. As is made clear by Schell (2015), the separate elements should be based on a unifying and resonant theme as a way to create powerful experiences. Therefore we will have a short look at the right theme and metaphor and how the theme can be reinforced in all the elements of the game.

As a starting point, the required actions 'jump over the line' and 'avoid the moving elements' can be extracted into analogies. Such as a train rails for the circular border, and spiked marbles for the elements to be avoided. Other possible options for the border include amongst others a water stream, NASCAR track, aqueduct, bridge, hyperloop, canyon, lava stream, mid-atlantic ridge or a canyon. For the dangerous elements crocodiles, bumper cars, ninja knives, biting turtles, sharks, stealth planes, fire spitting phoenixes or objects on fire or with rose thorns can be used. Combining tracks with elements provides possibilities such as a NASCAR track with bumper cars, where the jumps of the players can lead to earthquake like vibrations that will spin the bumper cars out of control. Another mix is that of a water stream with biting turtles, where the turtles swim gently and are even vulnerable for jumps on their shield, but they will attack a person that lands in the water. Third, a canyon with stealth planes would evoke a sense of avoidance due to the sharp and impenetrable exterior of the planes, as well as their ammunition

which can be aimed at those trying to ‘jump’ over the canyon. A fun effect could be to try to make the planes shoot each other out of the sky by careful timing of the jumps. Finally, a mid atlantic ridge/shark environment can be created such that when one jumps from the European to the American tectonic plate or in the reverse direction, it will lead to vibrations of the plate that will bring them closer together. If this is done often enough, the players can bring the plates so close together that the sharks can't swim anymore. From these ideas, the NASCAR track is considered the best option since children are drawn to the idea of NASCAR races wrapped in a more child friendly theme, which can make use of the popularity of Pixar's movie Cars (Appendix C provides more arguments and a comparison with the turtle game).

Game mechanics

To simulate jumping there should be a cause and effect relationship, for example jumping might lead to cars spinning out of control, cars ending up in the playfield, the collapsing of the surrounding stage or the occurrence of cracks in the track. For simplicity, the concept can be such that children might jump over a NASCAR track. Since the cars spinning out of control is an effect, the cause of these events should be clear to the players. To simulate jumping, an image of a trampoline can be used, where the protection foil can function as the track upon which the cars race. However, this would still not suggest that the players should jump over the track, instead of merely on the trampoline itself. Nonetheless, how people jump is not of strict importance thus for simplification the game mechanics can be such that jumping on the trampoline, without jumping over the tracks, would suffice.

With respect to the curious nature of children to try out all possible options, the rules should be clear to minimise chaos. Furthermore children might be inclined to form their own opinion on whether feedback is negative or positive, for example boys can be motivated by cars crashing whilst girls might be more tempted to keep the cars undamaged. Schell (2015) confirms this by stating that males like destroying things, where he makes an example of young boys building a tower but receiving the greatest enjoyment of knocking the tower over when built. The virtual possibilities for amplified destruction only add up to the boys excitement. Therefore negative and positive feedback should not be applied simultaneously since that might be hard to trace back whether an action is desired or not. According to this reasoning, it is advantageous to have only one effect that is related to jumping, which in this case will be the cars spinning out of control. But as described, this is not per definition negative or positive. A logical option would be to add collectables to the game from which collecting them is almost always perceived as positive.

These collectables can then be laid out just beyond the reach of the cars in their normal driving route (Figure 10, left), such that they can only be collected by the cars when the cars are ‘wiggled’ (Figure 10, center) out of the track as a response to a jump, the implementation will use a spring-mass-damper effect to recreate spring dynamics which pull the car back into its equilibrium position on the trampoline foil. To emphasize the effect sparks are visualized. If a car is already wiggling and meanwhile a player jumps, then the car's motion will be amplified which might allow to collect items that are laid out further from the track.

To stimulate competition, there will be two cars of different colours as shown in Figure 10 (right). The players are followed by a circle when they enter the playfield, and the colour of their circle will be arbitrarily matched to the colour of one of the two cars so that teams are automatically created. As for the collectables, they will appear in one of the two colours as well, where each team can only control the cars that matches their circle colour, and these cars can collect only the collectables that appear in the matching colour as well.

The collectables consist of fuel and nitro. The nitro gives the cars a temporary speed boost, which may lead to more collection of fuel collectables. The fuel is of higher importance since it determines how fast the cars drive, whenever a car stands still, it has lost the game. Losing the game provides negative feedback. However the instant reset of the game which leads to a refuel of the cars does not undermine continuous gameplay, consequently there will not be a moment where a child is not able to play. A GUI shows the amount of fuel left with the red bar, whereas the blue bar shows the amount of nitro. Whenever the red bar, resembling the fuel, is empty, one tank station icon as shown in figure 11 will disappear until none of them are left. When the icons are spawned on the playfield, they will appear in either a grey or red colour.

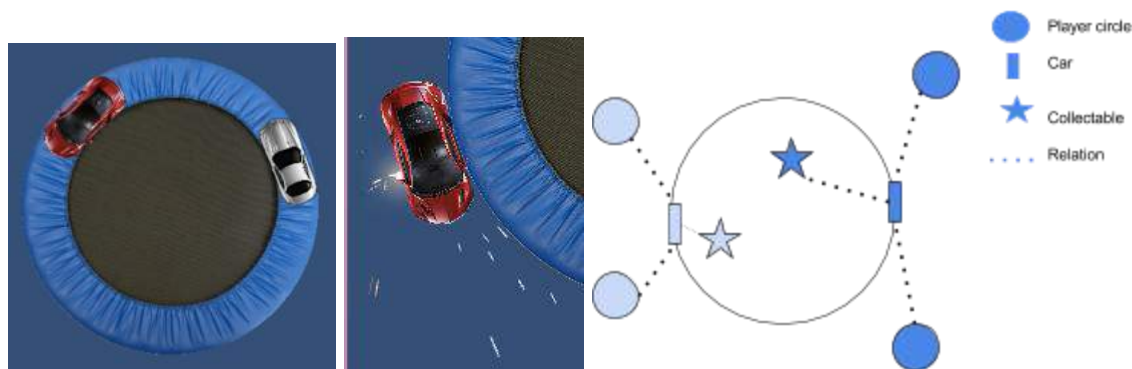


Figure 10, implemented view on the game elements (left) as well as the symbolic connections.

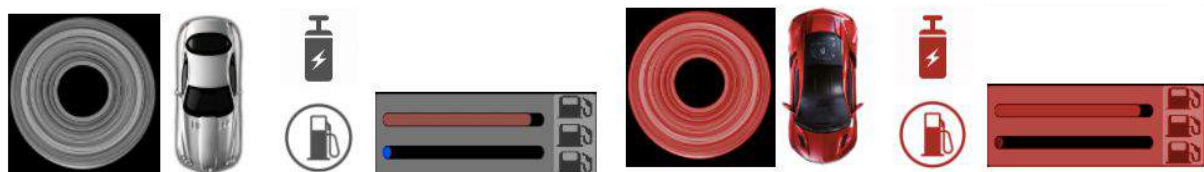


Figure 11, player circle with associated car, collectables and GUI for team grey and red.

Enticing elements

As for the enticing elements, the players should be seduced to perform an action whilst they are not required to do so, thus the elements will be independent of the core game mechanics. The elements can influence for example proxemics and the timing of jumps, but they can also be applied to simulate a more even division of gameplay between passive and active players. For the initial playtesting, proxemics are more applicable since it can be recognized more easily than jumps and proper timing thereof. Furthermore, currently proper timing of jumps requires manual checking and therefore we opt to do let the system report behavioural data by using proxemics.

To influence proxemics with enticing elements, the players can be seduced by a more stimulating or attractive environment when they navigate towards the desired position. Hence, there should be a central position where the aesthetic aspects are optimal, with colours becoming more grim as they are located farther away from that specific spot. Another option would be to have an inviting trail from the players position to the central point, however due to the inclination of children to run in different directions, this can be computationally expensive as the trail needs to be recalculated for every new position. The interactive tag game as described in the State of The Art, makes use of circles following the players, when the circle corresponding to the tagger touches a circle of a runner, the runner is tagged. In his research the author desired a strategy that would get the runners closer to the tagger. Van Delden accomplished this by releasing collectables around the tagger (Figure 2) which can be collected by the runners which in turn will beautify the runner's circle. Thereby, the runner is motivated to move towards the tagger because of aesthetics, despite the higher risk of being tagged. The car game does not have a tagger, but it could lay out collectables around the desired position. However, this might obscure the original concept of the game and lead to confusion, amplified by a higher risk of enticing collectables covering up the fuel and nitro collectables which are tightly related to the game rules. From the different options, changing the colours of the environment depending upon proxemics is the most suitable option with regard to the workings of the game. The implementation works relatable to someone walking through a garden at night with a torch, the environment is pretty but he can only see a small, proxemic, part of the environment. But the intensity of this light is dependent on his position, such that if the person desires to see pretty colours he will need to move towards a specific position. As such, he can be allured towards a position merely through aesthetics.

To translate the torch analogue into a game, we can use a background divided in rectangular sections with alluring colours, where the transparency of the sections has a minimum point and gets more transparent as it moves away from this point. To do so, I implemented in Processing, a software sketchbook, initially a field of squares that decay as they move from the center with a white background (Figure 12, left) and afterwards I applied the algorithm in Unity, this time with a black background (Figure 12, center). Now we have a resemblance of a garden, where the wanderer might be drawn towards a specific position because that is where the colours are the brightest. However, now there is no reason to walk towards the most brightly coloured position, since it is colourful despite the position of the person. Now to complete the analogy we can give the wanderers a spotlight, which allows them to see the space around them, but it only gets interesting as they move towards the hotspot. In The players position is visualised in Figure 12 (right), where the right image shows eight players with the four lowest positioned players walking around the bright area which they can admire with help of their 'torch' and the other players walking around the dim area. Figure 13 shows the complete concept of the car game, where the image on the left lures players towards the center, whereas the right image allures them to the bottom of the playfield.

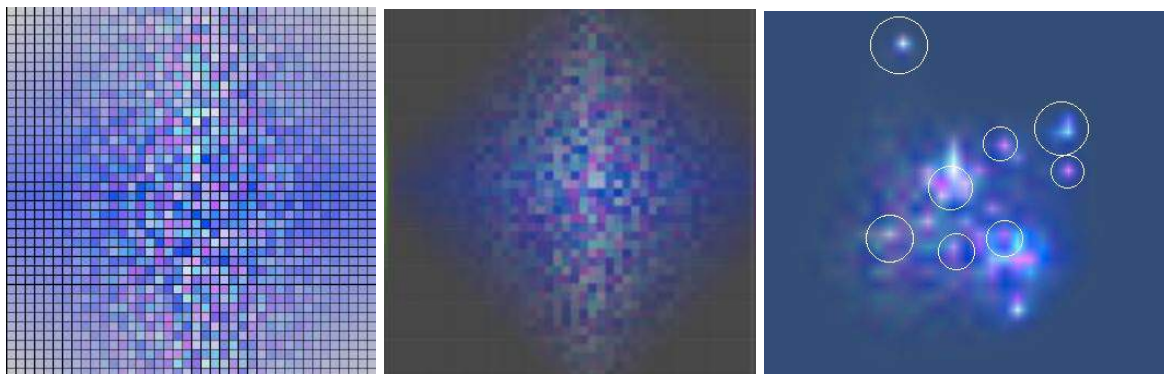


Figure 12, backgrounds where the brightness of the rectangles peaks around the center, with the left image being the most general whereas the outer right image adapts to positions.

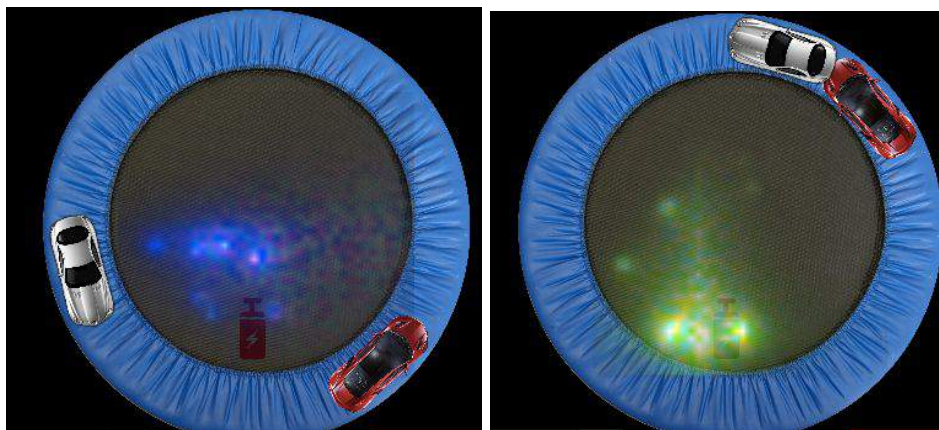


Figure 13, enticing elements applied to a jump-based game.

4.2 Jump recognition

The playground sends pre-processed data from four Kinects over the internet to a central computer, included in this data are the coordinates of the players. Figure 14 shows a 3D representation I made of a coordination test round where the observed person jumped five times at six different locations. Each dot is a recorded position from a given time frame. There are six visible positions with a high density of dots, which means that the person has stood still for a longer duration at those positions, which one can expect when one jumps at those positions. From these dense positions, a hunch (lower dots) followed by a jump (higher dots) can be extracted, additionally the lower dots have a higher density which implies that more time is spent in the 'preparing to jump' phase than jumping itself.

To extract jumps, the average depth from the Kinect to the player of each player is stored as well as the standard deviation of the depth. Whenever the measured depth of multiple following frames (to prevent outliers triggering jumps) is lower than the average height minus the standard deviation, the system recognizes this a jump. For increased control, this standard deviation can be multiplied by a factor to tune the threshold for reporting jumps. Appendix D shows an test simulation in Processing to test the algorithm in a simplified manner, where the applet tracks the vertical position of a finger to see if a change of height, similar to that of a jump, is indeed confirmed by the system. This intermediate test gave more insight in the workings of the Kinect such as the different ways to visualize the data, it also provided an increased awareness of boundary cases of minimal and maximum changes in height that trigger jumps. Additionally, it became clear that the SD needs to be calibrated a bit, since when no change in height has occurred yet, the SD is still close to zero and thus the threshold for jumps is easily exceeded, resulting in too many false positives. Another option would be to recreate the playgrounds default tracking system by hanging up a Kinect on the walls and directly streaming the data to Unity or processing, in case the default system would fail.

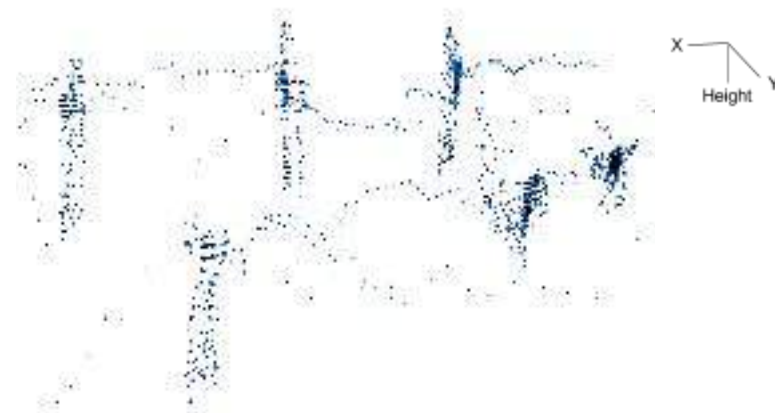


Figure 14, a player jumping five times at six different locations.

4.3 Initial concept

The concept used for the playtest is shown in figure 15, it shows a nitro collectable for the grey car as well as fuel for the red car. Note that the grey team is lucky since the grey collectable is positioned on the track of the cars and will thus be collected automatically. To collect the red fuel, the red team should time their jumps carefully such that the car is swinging at the moment that it passes the item positioned outside the track so it can be collected. If the item were to be positioned at a greater distance from the track, one person can jump first to give the car initial motion, with a second player jumping soon after to increase the swing effect to allow the team to catch items that are dropped further away from the track. As for the circle, it is currently green, which is an indication that the corresponding player has jumped. The jump can also be observed by monitoring the size of the circle since scales proportionally to the players vertical position. From the image we can also observe that the circle has grey as its inactive colour, since the grey car shows sparks and is pushed outside of the track, which means that a player of the grey team has jumped.



Figure 15, the game concept used in the first playtest.

4.4 Playtesting

Participants

To playtest the concept, children from a primary school who did a field trip to the university around the age of seven and eight, were asked in groups of circa 4 players to participate in the game.

Methods

Beforehand, consent was asked to make observations when they were playing. The rules were initially not explained such that the understanding with regards to the game mechanics could be analyzed. Once it turned out that they did not grasp the game or concepts, explanation was given by another facilitator of the university and myself. Non-anonymous data as video recordings were not used for privacy reasons.

Observed actions, verbal statements and responses to informally asked questions were noted and categorized into understanding of the concept, enjoyment, exertion and improvisation as shown in Appendix E. Human observations were supported by those of the system, were logged data resulted in a separately recorded data set of the movements of the children from which horizontal and vertical position could be extracted. The questions were relevant to the direct observations, for example children jumping on cars lead to the question how they could make the cars wiggle. If this resulted in an answer incoherent with how the game worked (“you should jump on the cars”, “when you hit the icons”, “if you don’t stand still”), further instruction could be given to how it should be played since the observation was made that the steps were too complicated. For example, asking the children to stand still, observe the cars and then observe the cars once more after jumping, resulted in the children making the relation between jumping and swinging cars.

Results

Game mechanics

As the playtesting progressed, questions shifted from game mechanics to uncover more detailed aspects such as whether a trampoline is a good fit to encourage jumps (which resulted in general answers such as jumping, but also a more specific, less applicable, memory of falling of the trampoline and breaking an arm). On the question if children preferred other icons such as a car, responses mainly fell into the category of an animal from which pets such as a dog and cat and bunny were named fairly often, this can be caused by the fact that their relation with those animals is more close than that of an object as a car, which can also not be their property as opposed to a pet.

However, the most prominent problems or misunderstandings were tied to game mechanics, the collection of items was not always evident as the mere reposition of an item did not give enough confidence that this was positive feedback as a result of collection.

Secondly, no associations were made between circle colour and the ability to only control the car with the corresponding colour, which in turn can in fact only pick up collectables of the matching color. The effect of jumping was not linked to the wiggling of the cars either, and the function of the wiggling of the cars was unclear as well. Instead a large variety of children taught that collectables could be collected by jumping on them.

Furthermore, one child did point out that picking up fuel did not lead to a change in the corresponding GUI slider. This was not a technical error but rather conceptual.

As figure 17 shows, the grey and red car have their own separate GUI. But the size of the playground made it hard to keep track of both images. Combined with the fact that Nitro collection was hardly noticeable, the child only saw the grey GUI and figured that the upper slider represented the fuel for the red car, and the bottom, blue, slider the fuel for the blue car. However, it was observed that when the fuel was empty, one tank station icon would disappear. On the other hand, the game was created such that when one team has lost all of its tank station icons, the other team has won. This was not observed by the children due to the unawareness of a division in teams. Beyond that, the winning text did not scale properly and merged too much with the background colors as opposed to being clearly visible with help of a big contrast between colours. The following circles were also scaled too big, which could be prevented by setting up the resolution beforehand.

Enticing elements

As for the enticing elements, they were not observed by the children due to lack of visibility. The lack of visual presence lead to the fact that enticing elements did not influence proxemics, furthermore no child mentioned them. By that, further details with regards to its effectiveness were not tested since visibility and awareness were prerequisites for researching their effects.

Additionally, the children were moving fast over the playground. This would not exclude the possibility that players can be guided towards a central location, but their position would be more likely to deviate from this point. Thereby would it be overall more challenging to allure players when they are moving swiftly over the playfield. Figure 17 shows the horizontal movements of three players that were playing simultaneously, the results do show that the children do have a preference to wander around a given area of the playfield. For example one child walks more on the west side while the others stay on the east side of the playground. This shows that it could be possible for these three players to guide them as no individual trail covers the entire playground.

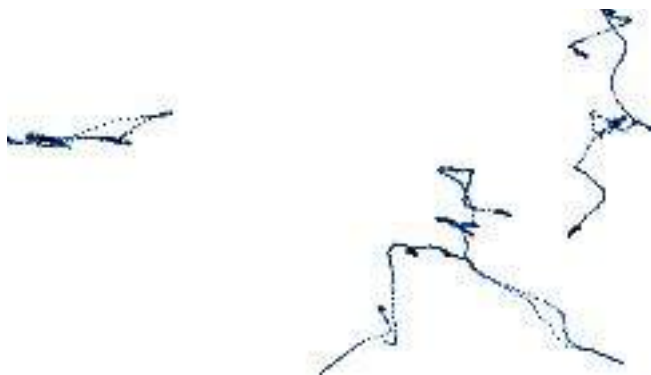


Figure 16, visualized positions of three children walking around the playground.

Jump recognition

With respect to jump recognition, the system did not recognise them properly. The left image of figure 17 shows the depth data of a bin that is pushed around the playground by a facilitator of the university, one spike is circled in red but apart from this incongruence in height the approximately the same depth is measured over the playfield. This would suggest that the position on the playground did have a relatively small influence on the measured depth. This leaves the problem with the algorithm of setting a depth threshold for jump recognition based on standard deviation.

The right image of Figure 18 shows the measured depth for two playing children. The blue graph follows, with the exception of the start and end of the game, stability in the measured depth whereas the red graph seems to consist of plateaus rather than a stable line. The higher plateaus could be linked to the observation that players often tried to sit on a car or collectable which would increase the depth from their head to the Kinect. The plateaus indicating a lower than average depth would suggest jumps but for that their duration is too long and spikes would be more logical than plateaus in the case of jumping. The low probability that the above average height for an extended time period would be caused by jumping, might leave us with the guess that some plateaus are caused by peaks such as the circled one in the left image of Figure 18. To minimise the effect of distortions on the depth measurements related to the position in the playground instead of actual player height, the players could remain on a single position to increase the probability that deviations in depth data are caused by actual higher or lower positions of the player. Setting the threshold for jump recognition according to the standard deviation is also error prone since the SD will initially be close to zero until children have moved up and down to get data points that deviate from the averaged depth. Beyond that, children that tend to crouch, sit on the floor or jump very often will have a relatively large standard deviation whilst in real life you would not let such factors influence whether someone jumps high enough to classify it as a jump. Conclusively, determining the jump threshold based on the average height times a tuning factor (f.e. $\text{jump threshold} = \text{average height} * \text{factor}$) would be a more stable solution.

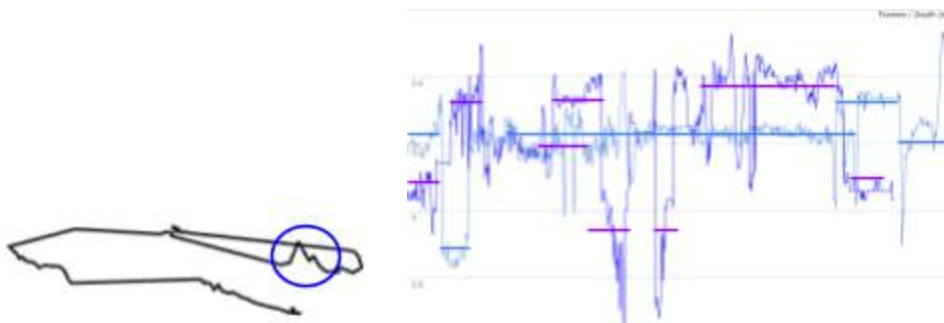


Figure 17, 3D coordinates of a bin pushed around on the playground (left) and the measured depths of playing children during a playtest.

4.5 Conclusions

Jump recognition is hard to do when players change their positions swiftly, since it affects the measured depth. Moreover, the distinction of two teams, as indicated by the colours grey and red, was not observed and the children played as one team. The elements of the GUI were not linked properly to their function and required that the children were aware of their team color (grey or red). Links were also not made whenever a collectable was collected (a car bumping against a collectable of the corresponding colour), what the effect of jumping was (cars wiggling), what wiggling cars could lead to (picking up collectables that were positioned outside the race track) and the function of the collectable (fuel or nitro). Human explanation was also necessary for a proper understanding of game mechanics. Although the playtest uncovered many points of improvements, if the goal were solely to provide a fun and active experience that stimulates the creative mind, a selection children's statements from Appendix E are shown in Figure 19 and show some examples that this goal has been accomplished. On a more technical note, for further prototypes the concept should adhere to the conditions as described in the simplified MoSCoW method of table 6.



Figure 18, Expressions of children playing the game (photo by [Robert Collins](#) on [Unsplash](#)).

Must	Should	Could
Everyone plays cooperatively in a single team. <i>Since no link is made between the color of the circle, the wiggling of the cars and jumping.</i>	The state distinction between jumping and standing should be evident. <i>Children responded that the circles should wiggle.</i>	The necessity of a GUI could be minimized as it lead to confusion between the colours of the cars and the colour of Nitro and fuel as indicated by the GUI.
The game mechanics must be such that children stay as much as possible in one place when jumping. <i>As horizontal movements affect the measured depth. Thus standing in one place will maximize changes that variations in height are due to actual crouching or jumps.</i>	Jumps should be triggered by a threshold directly derived from the average height, <i>instead of the standard deviation. Due to the standard deviation needing to be calibrated properly which leads to unnecessary steps before gameplay can take place.</i>	
Players are nudged with enticing elements to perform actions that don't require horizontal movement. <i>This is necessary, as a result of players standing primarily on the same spot.</i>	It should be clear when collectables are collected. <i>Children indicated that they did not know why the icons disappeared after a car with matching colour touched them.</i>	
Items must be clearly visible on the playground. <i>The enticing elements blended too much with the background colours and that being so they were not observed.</i>	The game should build up tension as playtime progresses. <i>No tension was built up as there was no win-lose state, which could increase the level of engagement and peer pressure for active play.</i>	

Table 6, prioritization of further game concepts by the MoSCoW technique.

5 REALIZATION

No distinction between teams.

The colours of the player's circles are the same and they all influence the same object. The relatedness between grey or red elements was not recognized in the playtest, and competition is not a necessary part to play a game such that it can be left out without consequences, such that with respect to simplicity it is a better option to play with one team.

The game mechanics must be such that children stay in one place as much as possible when jumping.

By having jump platforms as shown on the right, where jumps will only be triggered as long as the players touch the platform, the depth data will be more regular since changes in depth will be mostly from vertical changes in position instead of horizontal ones. Four platforms are chosen because it allows for proper supervision of gameplay whilst still having a large enough group for dynamic social interaction. With the playtest concept, the enticing aspect consisted of a background with increasing aesthetics around a specific position as players moved towards that center point. By having the players jump on the same location, the enticing element of position has become the required element.

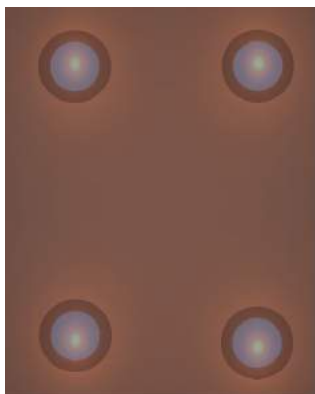


Figure 19, four jump platforms.

Players are nudged with enticing elements to perform actions that don't require horizontal movement.

To simulate actions whilst staying on the same position, the participants can be not be allured to change proxemics, as a results the undertakings should be tied to jumping. This can be further extracted into variations in time or vertical distance. As for the latter, the system will not be precise enough as of now to extract detailed information concerning the height of the jumps or depth of the crouches. As for jumping with respect to the time domain, the time at which players jump can be either measured dependently or independently. For the first, specific patterns in jumping can be encouraged. However, division in teams lead to confusion whilst playtesting, moreover one single team might positively affect a feeling of cooperation and as a result positively affect the group dynamics. The initial game required that players jump together simultaneously or soon after each other to increase the swing of the cars and thereby to collect items positioned further from the track. The current setup does not require those synchronized jumps, but rather aims to allure them through enticing elements. To simulate synchronized jumping, there should be an effect whenever multiple person jump at the same time. To enhance the feeling of cooperation, the effects can be such that they connect the players by means of connecting elements. Figure 20 shows the early prototypes of the elements and their application in the final game. The elements are triggered whenever multiple players jump simultaneously or at least within a given time zone from each other. The elements also create the illusion that they move, whilst in fact this effect is obtained by timed disappearance of one single element at a time. For example the first frame would show elements 1, 2, 4 and 5, while after a few frames 1, 2, 3 and 5 would be visible. In doing so, the elements give the appearance of all shifting one spot.

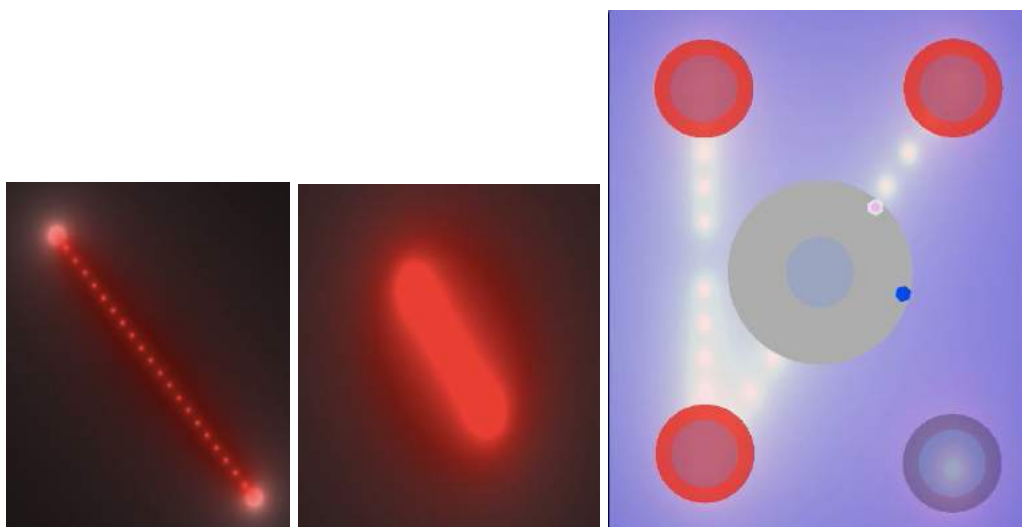


Figure 20, early prototypes of the neon-like elements (left) and their final application to connect three activated platforms.

Items should be clearly visible on the playground.

The resolution is set beforehand to 1920 * 2400 pixels, which equals that required for running it on the playground. As for the colours, blue is used for the platforms in their standard state whilst they turn red when a player jumps on them. The neon elements are blue and thereby contrast the complementary coloured orange background. The track is grey with a white/purple collectable and a blue car. When I was trying to recreate the new game based on the objects of the original concept, I used intermediate objects such as bare circles as a stepping stone for the final concept. However I found that the 'placeholders' showed an interesting futuristic effect and its simplicity might lead players to put place more focus on the game mechanics and the neon-like elements. As a result, I decided to keep the placeholders and with respect to Schell's (2015) notion of unification of theme all components were created in the same style using basic forms of hexagons and circles. Schell also notes how a designer can lever on the imagination of players, and thereby objects might benefit from staying minimal as it invokes more of the imagination which often is more creative than realistic objects.

The state distinction between jumping and standing should be evident.

The player has a following circle with an enclosing hexagon which opacity has a linear relation with the vertical position of the associated player. Whenever the player jumps, the hexagon will rotate around as to indicate that a jump has occurred as shown in Figure 21. Figure 20 shows how the platforms additionally turn red to further emphasize jumps.

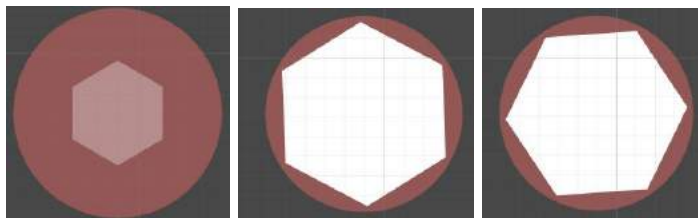


Figure 21, on the left the matching circle when the player is in normal position, the right two images show a change in rotation, which is put in effect when the player jumps.

Jumps should be triggered by a threshold directly derived from the average height, instead of the standard deviation.

By scaling the average height with a factor to set the threshold, the resulting threshold will be less dependent on change of movement and therefore be more reliable. Listening for jump is done when a player stands on a platform, only then does the program monitor jumps. When he or she exits the platform, the data concerning the player height is reset and will be updated whenever a player reaches the platform again. If a player is standing on a platform and his or her height exceeds the threshold, the game controller is notified of the jump, along with the platform upon which the player jumped so that the enticing elements can be connected to the proper platform.

It should be clear when items are collected.

When the driver touches a collectable, the collectable will spin around and get increasingly smaller. At its smallest point, as seen in the middle picture of figure 22, it will relocate itself to a new position whilst maintaining its rotation as it returns to its original size. As for now, sound is not included as to avoid annoyance often caused by repeated sounds. This can be avoided by choosing sounds randomly from a 'pool' of sounds but as the location of the design lab where the project will be run has many working students who might be distracted by it, it is avoided as of now.

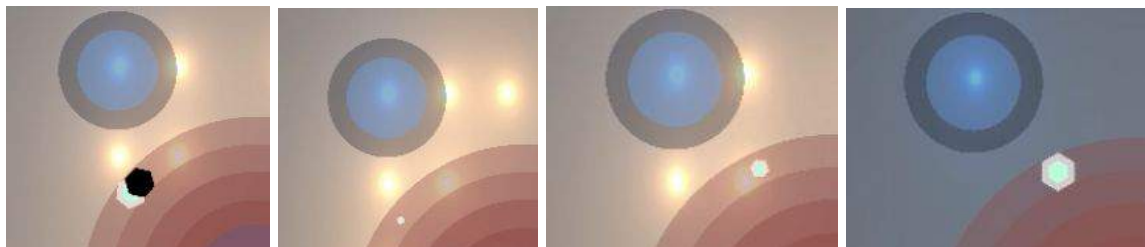


Figure 22, the collection of a pick-up item.

The game should build up tension as playtime progresses.

Initially, the requirement is to collect one collectable. When this objective is completed the players go to the next round, at which they have to collect one collectable more than in the previous round to proceed. For example, round 2 (visualized through the GUI as shown in Figure 23) requires the collection of two items, round 3 three etc. The caveat is that the car is also slowing down over time, when it has halted the game is over but players can reset the cars speed by making it to the next round. When the car stops and the game over text shows up, the game resets soon after such that players can continue playing. Nonetheless it can induce some tension to increase the engagement of players and to have them striving for a shared goal to increase a feeling of cooperation.

The GUI should be minimized.

The function of the GUI was to show how much fuel and nitro the car has. As the nitro was mainly serving for an increased potential in collecting fuel, it was not a necessary functionality and therefore it has been left out. In the former concept, the fuel collection was also not always directly tied to an increase in speed by the players therefore in the new concept the change in speed is more abrupt. Either collecting the item result in no change in speed if not enough have been collected or it results in full speed reset and an indication that the next round has started when enough have been collected. Figure 23 shows the functions of the GUI, namely to inform of the round number and the game over state.

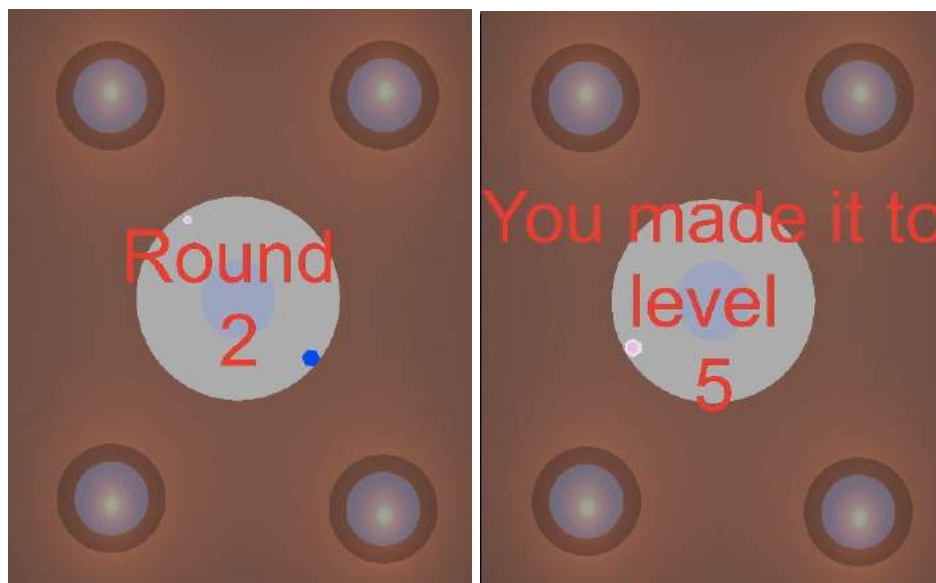


Figure 23, GUI text indicates either the round number (left) or the game over text.



6 EVALUATION

6.1 Participants

Four groups of three students have played the game, as well as four groups of two students. All participants were students from the University of Twente, and consequently they belong to a well educated group with an age ranging from around 18 to 26 years old. As for the gender, 10 women and 13 man participated.

6.2 Methods

Each group initially played the game with enticing elements turned off, after which they played the game with enticing elements turned on. This made it a within-subject test as the difference between the games with and without enticing elements were both tested on each group.

To analyze the percentage of the jumps that were executed as a synchronized jump (SJ) with respect to the total amount of jumps (TJ). Three methods were used to report SJ and TJ. First, reports were automatically provided by the game's code itself, secondly logged data was extracted from a graph and finally video recording were used as explained below.

The game itself adapted its environment according to synchronized and normal jumps, as such a simple increment whenever such a (synchronized) jump occurred would report the necessary statistics. Beyond this, the game also logged the coordinates of the players, this data has been visualized in a graph from which the synchronized jumps could roughly be estimated. Unfortunately it was not possible to count normal jumps from it as the data was too rough for such detailed extraction. Finally, the playtests were video recorded, which was the most accurate monitoring method as the video could be played back at a lower speed for more accurate logging. Furthermore I could adjust the jump threshold such that minimal jumps that the system might have missed could still be taken into account. Due to the logged video recordings having the highest reliability, the video recording have been used to compare the final results with and without enticing elements to determine whether the elements had an

effect. The null hypothesis is as following: “applying enticing elements to jump-based game will not stimulate synchronized jumps”.

For the video recordings, I counted the total amount of jumps as well as the jumps that were involved in a synchronized jumps. For example when a pair jumped simultaneous, I incremented the count with two as two people jumped at the same time. It would not be valid to count by pairs, as one paired jump and two people has a different SJ to TJ ratio ($\frac{1}{2}$) then two paired jumps and three people ($\frac{2}{3}$).

To determine whether enticing elements affect the percentage of synchronized jumps per total amount of jumps, we need to perform a paired t-test as the setup was within-subject. The alpha will be 0.05 as it is small enough to indicate a difference whilst not being so critical that the difference between both playtest has to be very large as would be the case with an alpha of 0.01.

The extracted data from the video recordings are also compared with the system reports as well as the graphed data to make a statement about whether the game has been able to accurately extract (synchronized) jump data. To make a statement about the validity of research methods, it will suffice to make a rough observation by comparing the relatedness of the different methods' (system, video and graph data) outcomes. Due to the system counting SJ's by pairs instead of people, it will by definition not be valid to measure the percentage of SJ/TJ, however we can still use the data for rough comparison. The question is whether the game's inbuilt jump determination extracted jumps in a reliable manner.

6.3 Measurements

Enticing elements

Table 7 shows the logged jumps from the video recordings, for the recorded SJ I counted the amount of jumps that were involved in synchronized jumps (when two people jump at the same time, I added two to the amount of synchronized jumps), the total jumps and the percentage of synchronized to normal jumps. One of the participants of playtest seven did not give permission to video record the gameplay. Applying a one-tailed, paired t-test to the percentage of SJ with respect to TJ for both default and enticed games, the result was a p-value of 0.16. Which is larger than the formerly established alpha value of 0.05, therefore the null hypothesis can not be rejected. This means that we have not been able to confirm that enticing elements can lead to more synchronized jumps in the context of a jump based game.

Playtest	Participants	Default SJ	Default TJ	Default SJ / TJ * 100%	Enticed SJ	Enticed TJ	Enticed SJ / TJ * 100%
1	3	32	66	50%	23	38	60%
2	3	14	30	50%	38	50	80%
3	3	39	72	50%	36	70	50%
4	2	24	69	30%	16	69	20%
5	2	16	39	40%	10	55	20%
6	3	11	101	10%	10	40	30%
8*	2	12	54	20%	32	78	40%
Average		21.1	61.6	31.3%	23.6	57.1	37.5%

Table 7, Comparing SJ and TJ and their relationship for default and enticed games.

*Group 8 follows 6, as group 7 did not give permission to video record the playtest.

Jump extraction

Table 8 shows the SJ and TJ extracted from the system, video recordings or graphs (video data from group 7 is missing and the individual jumps from the graph could not be extracted as described in the Method section). The difference with table 7, besides the measurement methods, is that the SJ's are counted by pairs (two or three people jumping at the same time counts as one synchronized jump). By analyzing the data we can see that the video recordings give the highest amount of TJ and SJ. To compare it with the lower averaged values of the system reports, the difference is most likely due to the system counting jumps according to the average height times a factor, this might exclude the small jumps that did not exceed the threshold to be counted as a jump. These small jumps could however still be extracted from the video recordings as humans do not measure according to a set value but rather according to an indication that the person was attempting a jump, albeit a small one. The graph reports the lowest value of SJ, which is most likely related to the fact that it was hard to properly extract jumps from the graph, as such not all synchronized jumps were counted as such. Nonetheless, the difference between the average of the SJ as measured by the system and video recording is $(9.4 - 7.6) 1.8$, for TJ it is $(61.6 - 55.9) 5.7$. The difference between video recordings and the graph extractions is $(9.4 - 6.9) 2.5$. From this we can conclude that the logged coordinates visualized in a graph lead to a rough estimation of the actual amount of jumps, with the games inbuilt code being more precise and reporting jumps accurate enough for enjoyable gameplay but not so precise that it can be used for scientific research.

Playtest	Participants	System SJ's	System TJ	Video recorded SJ's	Video Recorded TJ	Graph extracted SJ's
1	3	14	60	14	66	7
2	3	3	35	6	30	5
3	3	14	79	17	72	11
4	2	15	64	12	69	7
5	2	11	46	8	39	6
6	3	0	94	3	101	5
7	2	0	17	N.P.*	N.P.	6
8	2	4	52	6	54	8
Average		7.6	55.9	9.4	61.6	6.9

Table 8, jumps determined by the system, video recordings and graphs.

*No permission was given to film.

7.1 Playability

The synchronized jumping game was more simplistic than the initial game concept but it evoked more engagement and people understood the game quite fast and no questions about the workings of the game were asked (perhaps this is a biased opinion as university student instead of primary school children played the game). A big change was that level ups were included, as such the player had a better grasp on how well they did and besides negative feedback of a game over state the game now also had positive feedback by winning rounds. This might have lead to more internal motivation. Schell (2015) shows how a good game shows a build up in tension/interest with intermediate 'reset' points, as visualized in Figure 24. The increased difficulty of avoiding a game over state as the synchronized jump-based game progressed mimicked the buildup in tension, and winning the current round would lead to a small drop as people could take a small break and be glad about the fact that they made it to the next level. The tension in the NASCAR game did not follow the same trajectory as people were not aware of how far away they were from a game over state, nor was collecting fuel a concrete sign of 'leveling up', instead it was more a slight suggestion that you were on the right track which might be too subtle to cause a temporarily break of celebration.

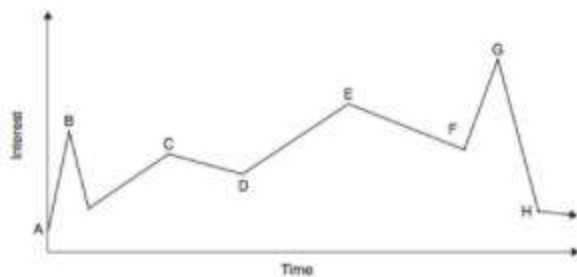


Figure 24, the interest curve shows how tension in a game should progress for optimal engagement.



7 DISCUSSION

7.2 Expected results

The results of this project are such that no considerable difference has been reported between play behaviour with regards to the ratio of synchronized to normal jumps, when enticing elements were or were not applied. This means that we have not been able to reject the null hypothesis “No significant difference can be observed for the amount of synchronized jumps in a jump-based games when enticing elements are applied”.

So why has no considerable difference been observed between the playtest rounds with and without enticing elements that were meant to allure people jumping together?

When I showed the effect of the elements after the game to same players, they said that they did not see the effect during gameplay. From observations during the playtest it was evident that people were focussing hard on the objective to make it to the next round, even though the enticing elements were rather small they were made such that they had a light emitting effect which in fact coloured the entire field blue instead of orange. This can be seen by comparing Figure 19 with enticing elements turned off with Figure 20 where they are activated. However, neither the change in background colour nor the neon-like elements were observed. Possibly, because they were only activated for a rather small amount of time and the elements did not stand out enough to be clearly visible but it might be that there is another reason that influenced the lack of awareness even more.

In research by Simons ("The Invisible Gorilla" n.d.) viewers are asked to count the amount of times the players wearing white pass the ball. The answer turns out to be 15 which most people from the research answered correctly, however since they were so busy counting the amount of times that the ball was tossed over most of the observers did not see the gorilla walking through the screen. This shows that when we selectively focus our attention we are inclined to glance over events that in retrospect are obviously present. Whether the enticing neon-elements were strongly present can be discussed but the fact that people were really engaged in the game and carefully timing their jumps whilst watching the position of the car made for very selective attentional focus.

This might have lead the players to oversee the enticing effect and thus the toggling between the elements state of activated versus deactivated had no effect. As the players minds were too occupied with making it to the next round to observe the present of changes in the background that were not directly related to the core game mechanics. There might be no definitive answer with respect to steering behaviour, but the research does tempt towards a more psychological hypothesis that the presence of changes in the game environment, that are decoupled from the game's core rules and methods, are unlikely to be observed whenever the players exhibit enough engagement in playing the game as was the case during playtesting (Figure 26). And for both concepts in this research, initially a game with enticing elements applied to steering proxemics which was later on replaced with a more simplistic version where that steered jump behavior, it has indeed been accomplished to provide an engaging and fun experience for the players, albeit it being a secondary goal. Further research could elaborate more on this hypothesis so that a better understanding can be obtained with respect to usefulness of decoupled game elements during engaged play experiences.

Another possibility why the enticing elements did not influence gameplay might be related to the fact that the playtest of the final concept was performed with students of the university. As a result, the characteristics of this group form a stark contrast with the initial playtest where the players came from a primary education school. Both did not observe the enticing elements very properly, but I suspect that for the young children, if they were to observe it, the enticing elements would have a greater effect since it is generally well known that young children are more curious. Consequences of a more educated and older group is that they might be more rational and less explanatory, whilst having a greater strive to achieve results that they often achieve by filtering out distracting elements as this would normally hinder them in their education and career. Directly related to the game, it showed itself in players focussing on sheering each other on and focussing on their target to make it to the next round while paying minimal attention to the environment.

Other results turned out to be more positive, for example the system reported the jumps quite reliably. Good enough for proper play of a jump-based game. The algorithm of applying a factor to the average height to set the jump threshold can be used by other developers for jump extraction. I would also urge them to first visualize how the system extracts coordinates as was done in this project, it gave away that more time is spent in the crouching phase than in the jumping phase such that one might consider to not let the crouches influence the average height. In the code for this project, the jumps did not influence the average height but it might be an option to also filter out the lower positions. By applying slight adjustments and additions, further work on the jump detection algorithm could increase the effectiveness of filtering out the jumps.

7.3 Boundaries of Enticing Elements

Enticing elements try to influence behaviour by means of a mild, non-cognitive demanding nudging approach. This subtle set of characteristics makes this approach limited by its definition. You can not force, only seduce. As such, the barrier for the action needs to be low if the effects were to have any effects at all. In our game players should carefully time your jumps to each other, while one person could also play the game perfectly well by itself. So the attuned timing might not be an action everyone is willing to undertake.

In our project, the reward also passes by rather shiftly. the Android game Ketchapp (Figure 24) uses different basketballs that the players can win when he or she has obtained enough points. The player might play a long time with the same basketball, so the enticing effect of playing with a more (or less) interesting ball has a longer duration and therefore the player might be more inclined to obtain this new and enticed ball.



Figure 24, different basketballs as a reward for collecting enough points.

7.4 Project Limitations

Transcribing jumps from the video recordings is still error prone since it is hard to make an accurate distinction between normal and synchronized jumps when their timing is close but still not close enough to exclude doubt as in how to categorize it. Therefore even the most reliable method of extracting jumps from the video recordings might be error-prone.

Furthermore, the low visibility and awareness regards enticing elements are such that they might as well have had an effect on play behaviour if people had observed them. Therefore it is not clear whether to suggest a complete different approach or if the colours and brightness should be tuned more accurately for optimal visibility.

As for the participants, both games were only tested with either children from primary school or from university. Therefore no claims can be made in general with respect to playability and understanding of the rules and methods. Also, the fact that the games were observed could have made people less likely to exit the game so that the appeal of the game might be overestimated as people were biased to show positive attitudes as this is what is normally expected by societal norms.

7.5 Future work

Others that tend to work with jump-based game should take great care to really consider the targets group age and education to derive their cognitive functioning which will influence how complex the game will be. However, the final game concept was rather simplistic in its nature and applied not to young children but to students and even for students the simplicity was beneficial. Therefore simplicity might be a rule of thumb for most jump-based playground games that are played without (extensive) explanation.

Engagement was higher with the concept were players jumped on one location, most likely this is related to the fact that the jump recognition worked better and that the concept was understood. Also, the tension build up as the game progressed by making it harder to pass each consecutive round. Flow and simplicity might therefore be stronger causes of fun than the complexity of interactions between game elements.

The enticing elements might also be used more indirectly, for example in the form of collectables that upon collecting them will give enticing feedback. This can lead to a trail of collectables that people can choose to collect if they prefer a more beautiful surrounding. Figure 2 shows the work of Van Delden et al. (2017) where collectables will lead to enticed circles, however their goal was to bring taggers and runners closer together. My suggestion would be to let players discover a path by following and consequently picking up collectables so that more complex positional changes can be formed. Figure 27 shows autonomous characters that either follow a path or randomly move around a field, these driving characters could also act as collectables such that children will have to trace them over the playground and will thereby have to show active behaviour.

Further work with enticing elements could still allow horizontal movement as was the case with the car concept, a possibility might be to let people stand still whenever they are jumping such that the jump recognition will still work properly. An option could be that whenever a child touches a collectable after tracing it, it stops moving and the player needs to jump on the collectable to evoke the enticing element.

Enticing elements could also be applied to more tangible products. Exergame ("Fit Interactive JumpQ," n.d.) gamifies fitness, one example is the Fit Interactive JumpQ (Figure 25). It shows platforms of different heights, each platform has a light embedded in it. As a platform lights up, children jump on top of a platform as such the scoreboard will be updated. However a score board might not be enough motivation for children that are less competitive. To increase the fun for the children who do not feel internal motivation to beat the scores, enticing elements might be a better alternative. An implementation example could be to light up the pad upon which the child jumps if he jumped on time, for this LEDs can be used that light up in a dynamic manner that will continue to surprise the jumper whilst not being so luminous that it will distract the jumper. Vitruvius ("Vivirius," n.d.) goes one step further with jump detection by visualizing an avatar (Figure 25) performing the same motion, it can provide a source of inspiration for new jump-based projects. The program not only mimics the motion but also reports about the state of jumping versus standing on the ground and it counts for how many frames the player is jumping, these details have also been used in our project. However it was not necessary to show a complete avatar, this would also make the game more heavyweight.

Vitruvius also shows additional capabilities that our system lacked by its more precise reporting of coordinates, if scientist desire to take measurements that have more strict requirements they could opt for this program.

Beyond this, Vitruvius was also able to report the height of the players, possibly due to the fact that the Kinect was not attached to a ceiling but should be placed on a platform. Nonetheless, being able to detect players height could be useful when the system could for example adjust its game difficulty to the players. It could then make a rough distinction between children and older players based on the players height. Our system could however make this distinction based on the speed at which players move over the playground. Once again it is necessary to make an assumption for this to work, namely that younger people move more shiftily and make more abrupt and sharp turns.

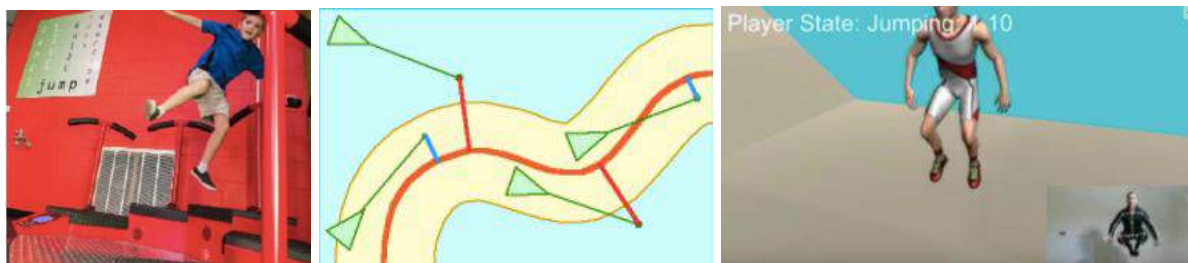


Figure 25, children jump on platforms ("Fit Interactive JumpQ," n.d.) of different heights to boost their fitness level (left). The center image shows autonomous agents following a trail (["Steering Behaviors For Autonomous Characters," n.d.](#)). On the right a fragment is shown of Vivirius' jump mimicking software ("Vivirius," n.d.).



8 CONCLUSION

No significant difference has been observed for the ratio of synchronized jumps to normal jumps when enticing elements were applied in a jump-based playground. Furthermore enticing elements were also not able to change proxemics.

This does not mean that enticing elements are not able to change behaviour, but rather that the enticing elements of this project did not manage to do so. This was largely caused by factors such as lack of visibility for the enticing elements, as the projection blurred the neon-like elements so that players did not observe them and as a result their behaviour was not altered. The enticed reward also appeared briefly in the case of the platform jump game, if it were to last longer people might appreciate the effect more and therefore undertake more action to obtain the desired visual feedback.

With respect to engagement and other test results, for the NASCAR game, the primary school children showed a curious and inventive spirit and had fun playing the game. This led to minimal attention given to errors and bugs of the system, although explanation was necessary for the players to understand the concept. Even after explanation the game mechanics were not always clear due to the system not always responding correctly to behaviour, and as a result links between for example jumping and wiggling cars were not always made.

The jump platform game which resulted from applying the MoSCoW prioritization method led to a substantially simpler game which was understood without questions, the target group were university students so it is not clear in how far this played a role or whether the game was just easier to understand. The players of the platform game were also more engaged, although less curious and inventive possibly due to their age. This engagement was caused amongst other reasons by proper balance between increased tension mixed with short rest periods to keep the game dynamic. As for recognizing jumps, shifting from a jump threshold to one set according to the average height and letting people jump on the same spot led to large improvements for the effectiveness of tracking jumps. The jump-tracking system was good enough in the final game for enjoyable gameplay without irritation due to a lacking jump recognition, however the algorithm is not accurate for scientific measurements.

Future work could use enticing effects that are longer lasting, and that could be applied such that they would urge people to stop and jump for collecting them such that players can still run over the playground if they desire to do so.

Bibliography

- 12 Fun Jump Rope Games for Kids. (n.d.). Retrieved January 13, 2018, from <https://www.buyjumppropes.net/resources/jump-rope-games/>
- BEAM By EyeClick - Interactive Gaming Projector System. (n.d.). Retrieved November 9, 2017, from <http://joinbeam.com/>
- Bolte, B., Bruder, G., & Steinicke, F. (2011). Jumping through immersive video games. In *SIGGRAPH Asia 2011 Posters on - SA '11*. <https://doi.org/10.1145/2073304.2073367>
- Cars 3. (n.d.). Retrieved from <http://films.disney.nl/cars-3>
- Denic, S. Z., & Charalambous, C. D. (2007). Control of Jump Linear Systems Over Jump Communication Channels - Source-Channel Matching Approach. In *2007 IEEE International Symposium on Information Theory*. <https://doi.org/10.1109/isit.2007.4557593>
- Fit Interactive JumpQ. (n.d.). Retrieved January 30, 2018, from <https://www.exergamefitness.com/products/active-floor-games/fit-interactive-jumpq/>
- Glover, J. (n.d.). Simon Says origin – Julie Glover, Young Adult Author. Retrieved November 7, 2017, from <https://julieglover.com/tag/simon-says-origin/>
- Gray, A. D., Willis, B. W., Skubic, M., Huo, Z., Razu, S., Sherman, S. L., ... Siesener, N. J. (2017). Development and Validation of a Portable and Inexpensive Tool to Measure the Drop Vertical Jump Using the Microsoft Kinect V2. *Sports Health*, 9(6), 537–544.
- Huang, J., Yu, W., Si, M., & Lv, W. (2016). Research and application of running action sequence recognition algorithms based on kinect. In *Proceedings of the 3rd Asia-Europe Symposium on Simulation & Serious Gaming - VRCAI '16*. <https://doi.org/10.1145/3014033.3014034>
- Karoff, H. S., Elbæk, L., & Hansen, S. R. (2012). Development of intelligent play practice for

- trampolines. In *Proceedings of the 11th International Conference on Interaction Design and Children - IDC '12*. <https://doi.org/10.1145/2307096.2307127>
- living floor - interaktive Bodenprojektion. (n.d.). Retrieved November 9, 2017, from <https://www.vertigo-systems.de/produkte/living-floor/>
- Lucht, M., & Heidig, S. (2013). Applying HOPSCOTCH as an exer-learning game in English lessons: two exploratory studies. *Educational Technology Research and Development: ETR & D*, 61(5), 767–792.
- Lumo Play Interactive Projection Games and Effects. (n.d.). Retrieved November 9, 2017, from <https://www.lumoplay.com/>
- Meckl, M. (2016). Isaiah Berlin and the Politics of Freedom: “Two Concepts of Liberty” 50 Years Later. *The European Legacy, toward New Paradigms: Journal of the International Society for the Study of European Ideas / Sponsored by the European Cultural Foundation*, 21(4), 437–438.
- Moreno, A., van Delden, R., Poppe, R., Reidsma, D., & Heylen, D. (2016). Augmenting playspaces to enhance the game experience: A tag game case study. *Entertainment Computing*, 16, 67–79.
- MotionMagix™: Interactive Playground Equipment | Interactive Floor & Wall For Kids Play Areas, Schools, Indoor Play Centers -. (n.d.). Retrieved November 9, 2017, from <http://www.motionmagix.com>
- Rijnbout, P., Graaf, M., & Schouten, B. (2015). The Richness of Open-ended Play - Rules, feedback and adaptation mechanisms in intelligent play environments. In *Proceedings of the 7th International Conference on Intelligent Technologies for Interactive Entertainment*. <https://doi.org/10.4108/icst.intetain.2015.259742>
- Sakai, T., Mizoguchi, H., Tamaki, H., Ota, Y., Egusa, R., Yamaguchi, E., ... Sugimoto, M.

- (2016). Multiple-Player Full-Body Interaction Game to Enhance Young Children's Cooperation. In *Proceedings of the The 15th International Conference on Interaction Design and Children - IDC '16*. <https://doi.org/10.1145/2930674.2935993>
- Schell, J. (2015). *The Art of Game Design: A Book of Lenses, Second Edition*. CRC Press.
- Singer, D. G., Singer, J. L., D'Agnostino, H., & DeLong, R. (2009). Children's Pastimes and Play in Sixteen Nations: Is Free-Play Declining? *American Journal of Play*, 1(3), 283–312.
- Sky Blue (ba). (n.d.). Retrieved January 14, 2018, from <http://viveka.id.au/skyblue/>
- Steering Behaviors For Autonomous Characters. (n.d.). Retrieved January 30, 2018, from <https://www.red3d.com/cwr/steer/gdc99/>
- Thaler, R. H., & Sunstein, C. R. (2012). *Nudge: Improving Decisions About Health, Wealth and Happiness*. Penguin UK.
- The Invisible Gorilla: And Other Ways Our Intuitions Deceive Us. (n.d.). Retrieved January 29, 2018, from http://www.theinvisiblegorilla.com/gorilla_experiment.html
- van Delden, R. (n.d.). (Steering) Interactive Play Behavior. <https://doi.org/10.3990/1.9789036543040>
- van Delden, R., Moreno, A., Poppe, R., Reidsma, D., & Heylen, D. (2017). A Thing of Beauty. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*. <https://doi.org/10.1145/3025453.3025816>
- Verbeek, P.-P. (2015). COVER STORY Beyond interaction. *Interactions*, 22(3), 26–31.
- Website. (n.d.). Retrieved January 30, 2018, from <https://vitruviuskinect.com/>
- Williams, J. (2009, August 20). 30 Classic Outdoor Games for Kids. Retrieved January 13, 2018, from <https://www.wired.com/2009/08/simpleoutdoorplay/>



Combination Matrix

A

Criteria ranked according to a three-point likert scale (low, average and good score):

- 1) Simplicity, how easy is it to grasp the concept?
- 2) Scalability, can the idea be easily react to changes in the amount of players?
- 3) Variability, is there an element of variability and perhaps surprise to keep the game fun?
- 4) Originality, does the game distinguish itself from the traditional games named in Table 1 and 2?

Skipping rope alternative	Playground game	Result of mix	1	2	3	4
Regular	Hopscotch	The hopscotch platforms all have their own spinning ropes which players should avoid collision with while they are jumping over these platforms.	3	1	3	2
	Marbles	Not only should players jump over ropes, they also should jump over marbles.	3	3	1	1
	Simon says	One player can jump on the “master” platform which means that its pattern is recorded. Other players have to jump afterwards in this same pattern.	2	1	2	2
	Tag	The players have to jump in order to be able to move their platform (likewise dribbling for basketball). These	1	3	3	3

		platforms are used to tag each other.				
	Freeze tag	Tagged participants need to jump the rope a specific amount of time before they can re-enter the game.	3	3	1	1
	Crack the whip	The skipping rope has adjustable speed, if it goes faster it becomes harder to jump but due to the force on the rope it might also loose parts and become shorter.	2	2	2	2
	Freeze dance	When the music starts or stops, the participants should switch from avoiding the rope to jumping on the rope.	2	3	2	2
Action jumping	Hopscotch	Passive players have to solve puzzles in order for a new hopscotch platform to appear on which the active players can jump.	1	2	2	3
	Marbles	There is a division of two teams, one team should jump over marbles while the other team can solve puzzles in order to spawn more marbles in the competitors playfield.	1	2	3	3
	Simon says	If a player's jumps on a simon play button, he can jump in a specific pattern which the other players have to mimic on the other platforms spread around the playing field.	2	1	2	3
	Tag	The tagger can get additional skills by temporarily skipping rope. The other players can take this effect away from him by solving puzzles.	1	3	2	2
	Freeze tag	If a player is tagged he can unfreeze himself by solving arithmetic puzzles which can be answered by jumping x amount of times.	1	2	1	2
	Crack the whip	One big skipping rope is affecting all players, it is changing colors and players can shorten it by jumping on a circle somewhere in the playing field with the same color.	3	3	3	2
	Freeze dance	Those who are tagged get a specific color, and can momentarily not participate in the game other players can free them by jumping over the skipping ropes with the corresponding color.	2	3	1	2
Double under	Hopscotch	Players jump over platforms while they have to avoid two skipping ropes that are swinging over the platforms.	2	1	3	3
	Marbles	Two ropes are being swung over the playfield. In between these two ropes, marbles are moving and	3	3	2	3

		simultaneously being propelled by them. The players should collect the marbles while avoiding the ropes.				
	Simon says	There are several colored balls moving through the playing field. These balls can be collected if someone touches them. One person can control which color should be collected. However they should simultaneously avoid the two ropes spinning underneath them.	1	3	2	3
	Tag	Tag is being played but at the same time the ropes spinning underneath them should be jumped over.	3	3	2	1
	Freeze tag	If players beside the tagger touch the rope, their 'taggable circle' freezes and the tagger can touch them. They can unfreeze it by jumping over the rope.	1	2	2	3
	Crack the whip	There is one big rope moving around the playfield. However with time the ropes get recursively divided in two. Thus at the end there are many small ropes spinning around the playfield which should be jumped over.	2	3	2	1
	Freeze dance	If the rope stops spinning, participants can not move otherwise they lose points.	2	3	2	2
Chinese jump rope	Hopscotch	The platforms are rotating and moving slightly, making it harder for those who jump over them.	3	2	3	2
	Marbles	There are rectangular shapes moving throughout the entire space, the players can gain extra points by jumping in and out of their center while avoiding their walls.	2	2	3	2
	Simon says	There is one big rectangular shape and one small one. The small one is the leading one and if someone jumps over its edges in a given pattern, all the others should repeat that pattern while jumping over the edges of the big shape.	1	2	2	3
	Tag	The players can play tag within the bounds of the shape, but if people jump over the edges of the shape, it will get bigger leaving more room to play tag thereby making it harder for the tagger. When someone else is tagged the size of the playing field (thus shape) resets itself.	1	3	2	3
	Freeze tag	Tagged players can unfreeze themselves by jumping over the outer edges of the playing field, so they should be cautious to be in range of the fields edges.	2	3	2	2

	Crack the whip	One large rope is spinning within a rectangular shape. If players jump repeatedly over the edges of this shape, the shape and thus the rope shrinks. They should be cautious however to avoid the rope.	2	2	2	3
	Freeze dance	The edges of a shape are changing in color. If the color turns black however it is forbidden to jump over the edge. All the players that are outside the shape lose a point if the color is black.	2	3	2	3
Snake in the grass	Hopscotch	Players can only stand on platforms. But beware, if a 'snake' touches a platform this platform gets poisoned and is no longer a safe spot so they should jump on a neighboring platform.	3	2	3	3
	Marbles	Snakes propel marbles forward, so the players should not only avoid the snakes but also the marbles.	2	3	2	3
	Simon says	Snakes can be deactivated by solving puzzles.	2	2	2	3
	Tag	If a snake hits a player, he becomes an additional tagger.	3	3	1	2
	Freeze tag	Tagged persons are unfrozen if they are touched by a snake. However when they are not frozen they should avoid the snakes.	2	3	2	2
	Crack the whip	At first there is one snake that should be avoided. This snake moves in the direction of the average of the players, if it has too much unexpected movements, it will get shorter.	1	3	3	3
	Freeze dance	Once in awhile, the snake gets extra speed and players should be extra cautious to avoid it.	2	3	2	2
Helicopter	Hopscotch	One big "propellor" consisting of multiple ropes is spinning from a central pivot point, at the outer edges of this rope there are platforms attached. If one jumps on this platform, the propellor loses the corresponding rope.	2	2	3	3
	Marbles	The ropes of the propeller consist of multiple marbles being held together by a spring. On a given interval, the outer marbles detach from the rope and become ingame objects that should be avoided by the players.	2	3	2	3
	Simon says	While the players are avoiding the propeller's ropes, they have to collect other moving objects as well. Which objects they should collect is chosen by the players that are not actively jumping.	1	2	3	3

	Tag	If the ropes touch a player, he becomes a tagger.	2	3	2	2
	Freeze tag	A tagged person gets his own propellor, moving along with him. He is unfrozen if he can hit another player with his propellor.	2	2	3	3
	Crack the whip	The ropes of the propellor get shorter if a player jumps on the outer edge of one of his ropes.	2	3	2	2
	Freeze dance	Like normal tag but If the music stops, propellers show up that should be avoided, when the music plays they disappear again.	2	3	2	2
Cat and mouse	Hopscotch	Same as regular skipping/hopscotch, this time there is also a tagger who can tag other people that are jumping over the platform. There are also collectables that unlock bonuses such as randomly disappearing platforms (useful for the tagger) or 'fly ability' (useful for the runners).	1	2	3	3
	Marbles	There is a big ring, over which players can jump to collect points. But, beware because the ring also acts as a trajectory for marbles (they follow the ring as a train follows the rails) that should not be touched.	3	3	2	3
	Simon says	Before one enters the rope, they should perform a command, thus while they are doing this they are vulnerable to be tagged.	1	1	3	2
	Freeze tag	If one is tagged, the rope spins with extra speed for a given amount of seconds. Thus creating a bit of chaos affecting all players.	3	2	2	2
	Crack the whip	The more activity there is, the more the rope starts to wiggle. The rope has a breakpoint at which parts of it start to break down thus creating a smaller rope.	1	3	2	2
	Freeze dance	As soon as the music stops, everyone should stand still, otherwise they lose points.	2	3	2	1

Table 9, combination matrix from traditional playground games mixed with tag variations.

Selection Matrix

B

<i>Jumping rope alternative</i>	<i>Playground game</i>	<i>Simplicity</i>	<i>Scalability</i>	<i>Variation</i>	<i>Originality</i>	<i>Total</i>
<i>Regular</i>	<i>Hopscotch</i>	3	1	3	2	9
	<i>Marbles</i>	3	3	1	1	8
	<i>Simon says</i>	2	1	2	2	7
	<i>Tag</i>	1	3	3	1	8
	<i>Freeze tag</i>	3	3	1	1	8
	<i>Crack the whip</i>	2	2	2	2	8
	<i>Freeze dance</i>	2	3	2	2	9
<i>Action jumping</i>	<i>Hopscotch</i>	1	2	2	3	8
	<i>Marbles</i>	1	2	3	3	9
	<i>Simon says</i>	2	1	2	3	8
	<i>Tag</i>	1	3	2	2	8
	<i>Freeze tag</i>	1	2	1	2	6
	<i>Crack the whip</i>	3	3	3	2	11
	<i>Freeze dance</i>	2	3	1	2	8
<i>Double under</i>	<i>Hopscotch</i>	2	1	3	3	9
	<i>Marbles</i>	1	3	2	3	9
	<i>Simon says</i>	1	3	2	3	9
	<i>Tag</i>	3	3	2	1	9
	<i>Freeze tag</i>	1	2	2	3	8
	<i>Crack the whip</i>	2	3	2	1	8
	<i>Freeze dance</i>	2	3	2	2	9

<i>Chinese jump rope</i>	<i>Hopscotch</i>	3	2	3	2	10
	<i>Marbles</i>	2	2	3	2	9
	<i>Simon says</i>	1	2	2	3	8
	<i>Tag</i>	1	3	2	3	9
	<i>Freeze tag</i>	2	3	2	2	9
	<i>Crack the whip</i>	2	2	2	3	9
	<i>Freeze dance</i>	2	3	2	3	10
<i>Snake in the grass</i>	<i>Hopscotch</i>	3	2	3	3	11
	<i>Marbles</i>	2	3	2	3	10
	<i>Simon says</i>	2	2	2	3	9
	<i>Tag</i>	3	3	1	2	9
	<i>Freeze tag</i>	2	3	2	2	9
	<i>Crack the whip</i>	1	3	2	3	9
	<i>Freeze dance</i>	2	3	2	2	9
<i>Helicopter</i>	<i>Hopscotch</i>	2	2	3	3	10
	<i>Marbles</i>	2	3	2	3	10
	<i>Simon says</i>	1	2	3	3	9
	<i>Tag</i>	2	3	2	2	9
	<i>Freeze tag</i>	1	2	3	3	9
	<i>Crack the whip</i>	2	3	2	2	9
	<i>Freeze dance</i>	2	3	2	2	9
<i>Cat and mouse</i>	<i>Hopscotch</i>	1	2	3	3	9
	<i>Marbles</i>	3	3	2	3	11
	<i>Simon says</i>	1	1	3	2	7
	<i>Freeze tag</i>	3	2	2	2	9
	<i>Crack the whip</i>	1	3	2	3	9
	<i>Freeze dance</i>	2	3	2	1	8

Table 10. Decision matrix for playground combinations.

Car and Turtle Concept

C

The turtle game was a result of the ideation phase with the intention to increase the happiness of the turtles by jumping over the water, as a result branches with leafs will be dropped in the water. Turtles then eat these branches and become more happy. But players should be cautious to not touch the water since that leads to a small water swirl, which worsens the environmental conditions. Also be aware to not jump on a turtle since that will kill the fragile animal. All in all, it is the task of the jumpers to create an optimal environment without dangers for the animals. The contentment of the turtles can be visualized by means of a smiling/sad emoticon or with a health bar. If the turtles are happy enough, one of them might breed a new turtle. Just be sure that there is a good balance between the amount of turtles. Conclusively, the following elements play a role in increasing the happiness of turtles: Dropping branches in the water and having a balanced amount of turtles. Water swirls and too many or too few turtles have a negative effect on the turtle's overall happiness with jumping on a turtle being deadly for the turtle. The left side of Figure 27 shows the different elements of the game, with a representation of the turtles contentedness on the right side.

Table 11 shows a comparison between the car and turtle game, according to different criteria. Overall, the cars score more points, this is mostly due to the complexity of the turtle game which can be partly observed from the table below. The turtle game requires teamwork and a deeper understanding of the game mechanics, which can be a positive aspect however it can distract from the focus of this project.

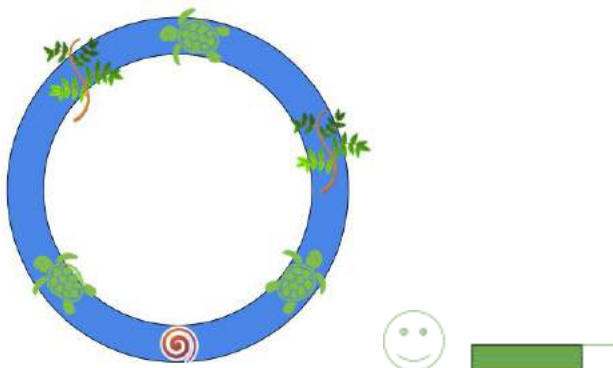


Figure 27, turtles, branches and water swirls affect the overall happiness of the animals.

Criteria	Cars		Turtles	
Fun for both genders	Mostly boys due to visual effects, girls might like the trampoline	1	Mostly girls due to nurturing, with boys more enjoying the sabotaging or the possibility to be good at the game (Schell, 2015).	2
Worst scenario effects are minimized	Players do not jump or jump on either the cars or the protection layer. It can be possible to not react to those scenarios.	3	More children walk through the water or over the turtles which will decrease the amount of turtles and thus inhibit the goodwill of these who try to create an optimal environment	1
Low dependence of elements	Cars can bump against each other and visual effects and position of cars depend on the amount of jumpers.	3	Amount of turtles is dependent on leaves, other turtles, water swirls, people walking through the water or on turtles.	1
Many people can play	If there is always someone jumping, there will be a constant appearance of visual effects which will lead to no variation.	2	With a large amount of players there can be a lot of change in the dependent factors making it hard to track down cause and effect..	1
Ease of comprehension	Easy.	3	Normal to hard.	1
Understanding of the functionality of elements.	Recognizing the function of a trampoline should be easy, although it can be harder to link its protection plastic to the stage around a NASCAR circuit.	2	Walking through water creating swirls seem obvious but it might be harder to convey that jumping is the most optimal result since one can also walk instead of jump over the water.	1
Feasibility	Jumping slightly changes the course of cars with mass damper effect which is moderately difficult likewise the visual effects.	3	The dependent elements will need tuning of the variables which can be simple for a few players but hard for many.	1
Maximal effects are still enjoyable.	The maximum amount of visual effects as well as the outer and inner trajectory of the cars might still be enjoyable.	2	Maximum amount of turtles, leaves and swirls can create chaos.	1
Total points	19		9	

Table 11, comparing the car game with the turtle game.

To a certain extent, it can be possible to predict statements made by players as well as how they might behave whilst playing the games. Table 12 shows such predictions where the sabotaging versus nurturing preferences of play for both genders mainly are contrasted against each other. With respect to teamwork, the turtles require more complex forms of communication. Thus, the table shows a somewhat generalized prediction partly formed by Schell's (Schell, 2015) emphasize on the difference in play between boys and girls.

Observation	Cars	Turtles
Stimulation of teamwork	Synchronized jumping for increased damage.	A lot of teamwork and communication is required for a prosperous fauna.
Expected sentences during play	"Did you see those effects?" "look the cars go inward", "This car is mine", "I hope they will collide", "due to the collision yours goes slower now".	"The turtles are happy", "Oh no I killed one", "I created a water swirl", "No don't step on the water, you'll make my turtle unhappy", "we should all work together to make the turtles happy".
Expected behaviour during play	Children jumping as well as standing still or walking over the trampoline, children jumping in unison to test the maximum effect, children trying to kick at cars.	Boys walking over the water to see what happens due to the water swirls it creates. Girls jumping over the water to feed the turtles. Testing outer bounds such a no turtles or the maximum amount.

Table 12, expected observations during a potential playtest.

Jump Simulation

D

Figure 28 shows a Processing sketch where the jump determination system is recreated by using the vertical position of the finger.

The Kinect has a depth sensor, the data of this sensor can be monitored with Processing. From all the incoming data, the sketch filters the positions within a given range, as for the example this range was such that it was tuned to easily show the hands. The next step is to take the highest point of the filtered data, which can be used as a tracking point. Then, the program took the average height of this tracking point and calculated the standard deviation. As a last step, whenever the current height of the finger exceeded its average plus standard deviation, the system reported a jump.



Figure 28, simulating jumps through finger movements, tracked by a Kinect.

Results Playtest 1

C

Observed actions

Understanding the concept

No link is made between the color of the circle, the wiggling of the cars and jumping.

The observer now asks the children to stand still and wait, until they all are asked to jump again. Now the children make the observation that jumping is tied to the changing of the circle's colors and the movement.

The observer asks whether the players prefer an active game or a computer game where you can sit. All children go and sit on the playground and seem to wait until something special will happen.

A child mentions that a car needs fuel, as observed from the GUI.

A child is concerned about the fact that the grey car hits the icon but the fuel is not replaced. It is not clear that the color of the car should match the collected icon. And the distinction between nitro and fuel is also not clear to him, since nitro has a blue color and fuel has a reddish color which with the grey or red car.

Enjoyment

Some girls arrive and say that they want to join the game.

Another group arrives to come and play, they are told to wait till the current group is ready.

Someone walks out of the game

The game is over but a child wants to continue playing

A boy would like to join the game

Exertion

One child does not want to play anymore since he is tired and leaves

Improvisation

Children keep chasing the cars and think that they should either destroy the cars by jumping on them or they do it for fun "Look I am sitting in the car", "I am faster than the car".

Statements

Understanding the concept

- "Why are the icons there"
- "The circles get a different color when you jump"
- "How does it work"
- "Jump there" (Child points to trampoline)
- "You need to jump to move the cars"
- "Wait, do I control only this car."
- "I don't get it"
- "What is the goal"
- "If you jump on the icons the cars go faster"

Enjoyment

- "The car goes super fast"

Exertion

- "I am tired".
- "I get nauseous from jumping"
- "This is like running hard"

Improvisation

- "I caught up the car"
- "I want to grab the cars"

Questions

Understanding the concept

Can you tell me what the idea of the game is?

- "The cars drive and when you jump you get more fuel."
- "It is difficult to explain."
- "Cars move when you stand on them".
- "You need to get the cars of the trampoline."
- "You need to jump so that the cars jiggle."
- "The tankstations go away when the cars jiggle."
- " You need to jump on the cars and the images, then they go faster."
- "If you jump on the cars, they wiggle."

Who knows when the cars jiggle?

- "When you jump on them."

"When you hit the icons."
"If you hit the cars, they will wiggle."
"Because they hit these icons."
"When you jump."
"If you don't stand still."
"If the circle's color changes."

When does your circle change color?

"If you touch the cars."
"Because of the cars."
"If you stand on the icons."

"Can you explain why the tank station went away?

"I don't know why."

What do you think of when I say trampoline?

"A broken arm, since I fell off the trampoline."
"Jumping." x2

What can we do to make sure you know that you are jumping?

"The circles should wiggle."
"Use a mega trampoline."

Did you see something special about the circles?

"They were blue, green and red."
"They become yellow when you stand in the middle and jump."
"When you jump they become red."

What if you all stand still for a while and then all jump?

"When you jump, the circles become green."

Enjoyment

What did you like about the game?

"I like the cars."
"The wiggling of the cars."
"That you can make them drive faster."
"That the circles follow you."
"That you need to run."
"That the circles chase you."
"Chasing cars and jumping."
"I don't like the game."

What did you not like about the game

"The cars went too slow". x2

"I don't like cars". x2

"The tankstations are there for nothing."

"The following circles."

"That you need to constantly search for fuel."

"It is not really a trampoline since you can't jump high in the air."

"It doesn't really work."

"It is not very easy to break the cars."

Exertion

What takes more effort, running or this game?

"Jumping"

What takes more effort, swimming or this game?

"Swimming"

Improvisation

What would you like to see in place of cars?

"a dog" x4

"At least no princesses." x2 (a boy and two girls)

"a parrot" x2

"A turtle" x2

"A rabbit" x2

"Butterflies." x2

"a cat" x2

"A motorcycle"

"Birds"

"I like cars"

"frogs or lizards"

"a cheetah"

"rhinos"

"a peregrine falcon, they are super fast".

"a fox"

"I would rather defeat bad guys."

"a cow"

"a BMW"

"A dinosaur"

Why would you prefer a dog or cat?

"Because I have them as pets"

Points of attention

Gameplay

For children of age seven to eight, there are too many steps to achieve a goal without explanation.

jumps -> wiggling cars -> -> collection of fuel or nitro> extra distance covered or a speed burst
-> winning when the other cars stand still. Also, the colors of the circles should be linked to the cars and elements with the same color.

Game elements

Circles are too big, such that they often cover the entire field and the icons.

Icons fall outside the projection, due to a change in resolution.

The GUI is too small to be noticed

Environment

Enticing colors are nearly visible and the trampoline does not leave enough room to observe positional changes due to the elements, which do not even occur since no remark nor change or behaviour in relation to it has occurred.

The effects of sparks that occur when the cars jiggle are hardly noticeable.

Tracking

Jumping recognition does not always work when children are running. Yet when they are asked to stand in the middle of the trampoline, the player circles indicate no sign of jumping, whilst when all the children then jump around 3 out of 5 circles become green which means that the jump has been recognized.

Suggested improvement by facilitators of the university

Sound that is triggered when icons have been collected would increase the association between cars hitting them and the collection of the items.

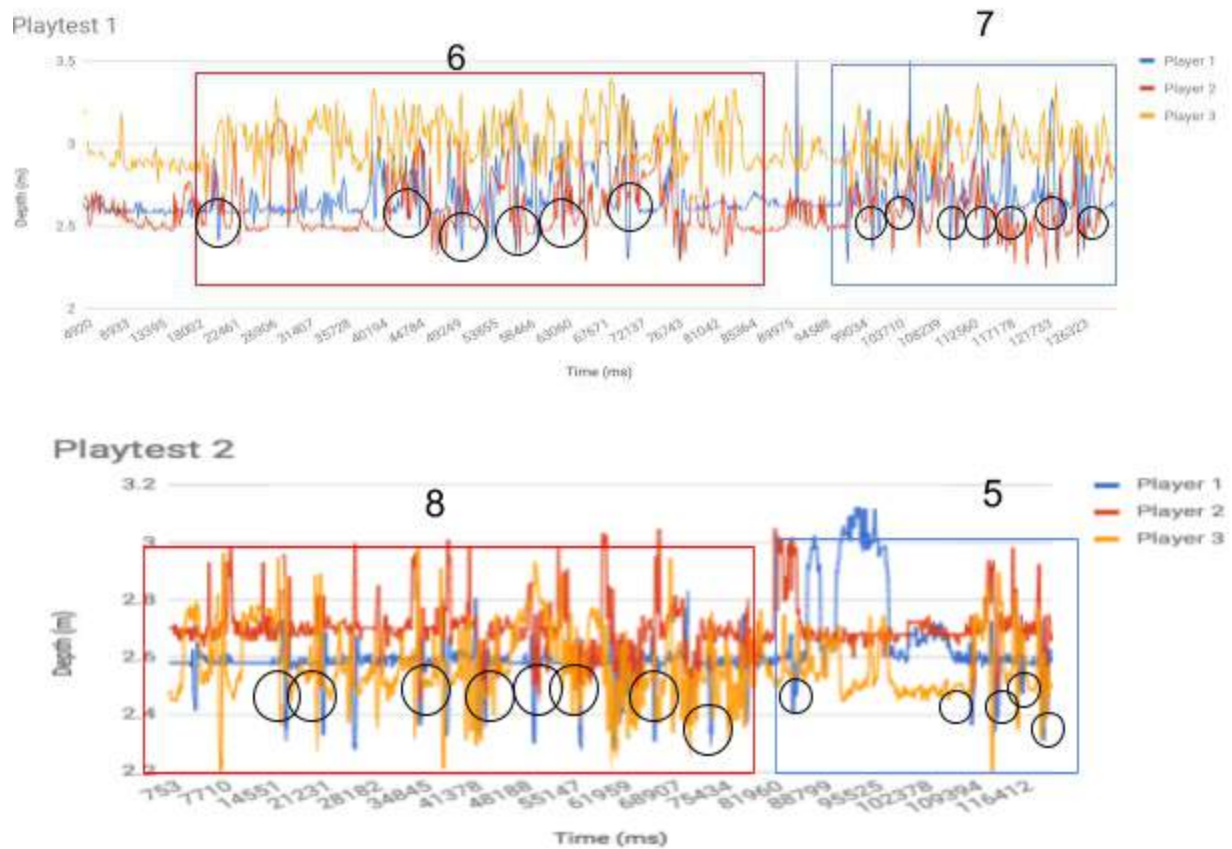
A pause option should be implemented as not to distract children when they are not playing.

Jump Graphs

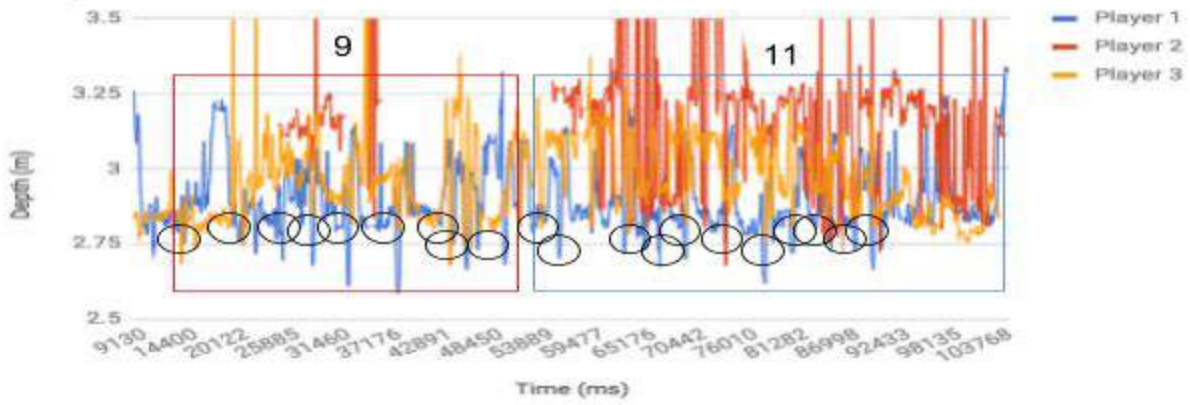
F

- Playtest without enticing elements
- Playtest with enticing elements
- Synchronized jump

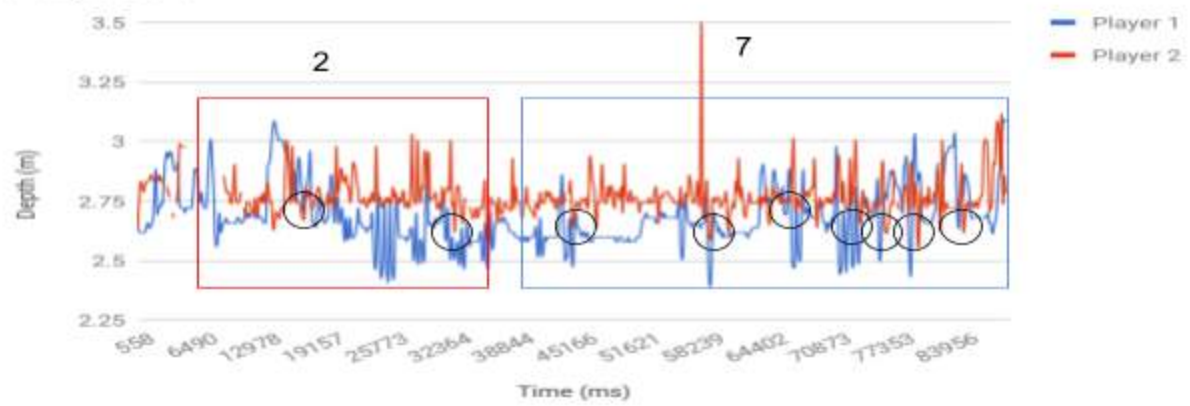
Nr active jumpers over all synchronized jumps



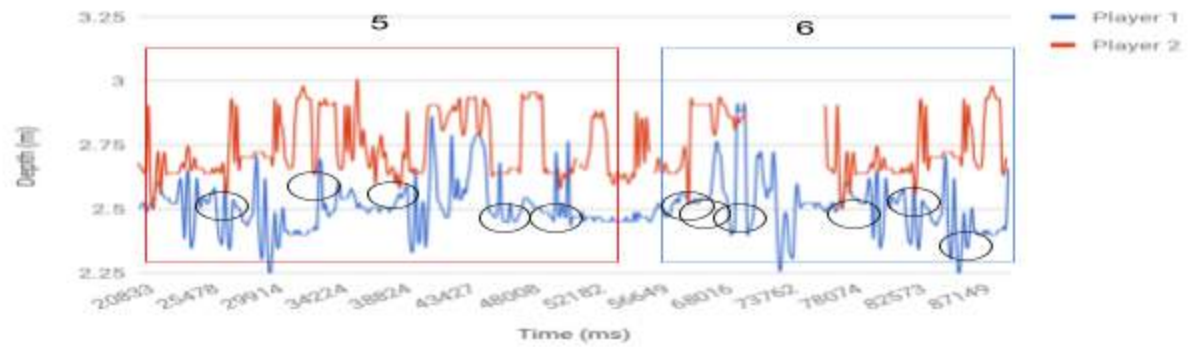
Playtest 3



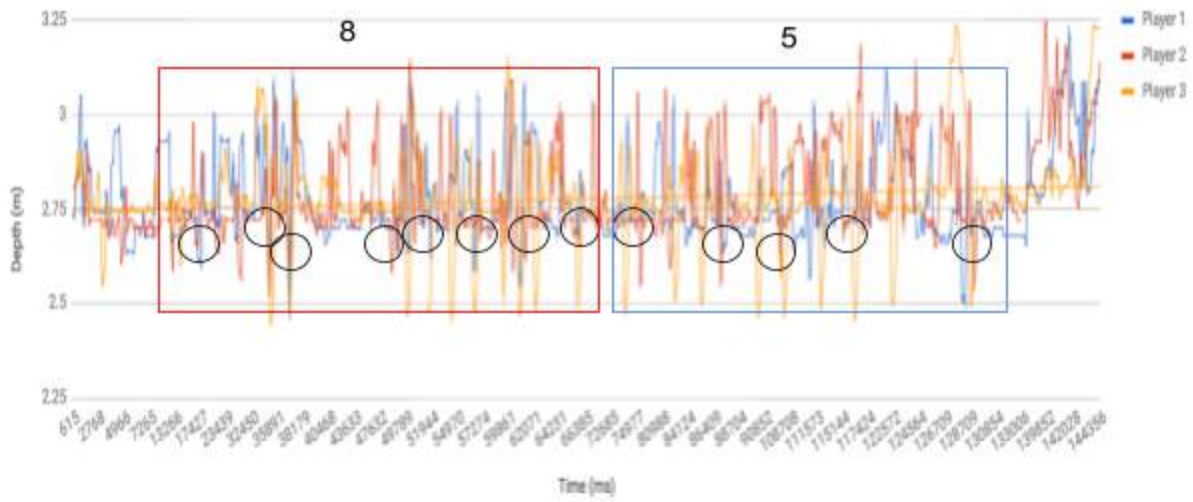
Playtest 4



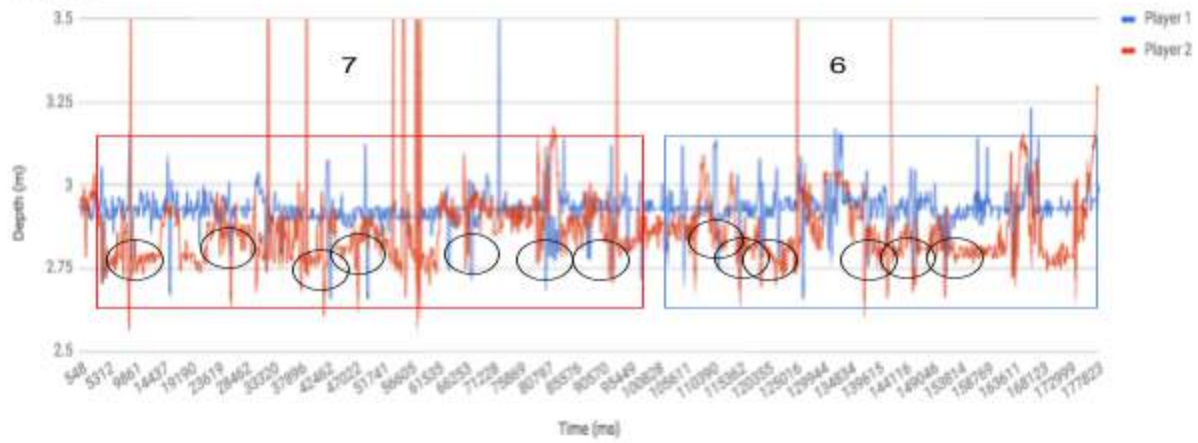
Playtest 5



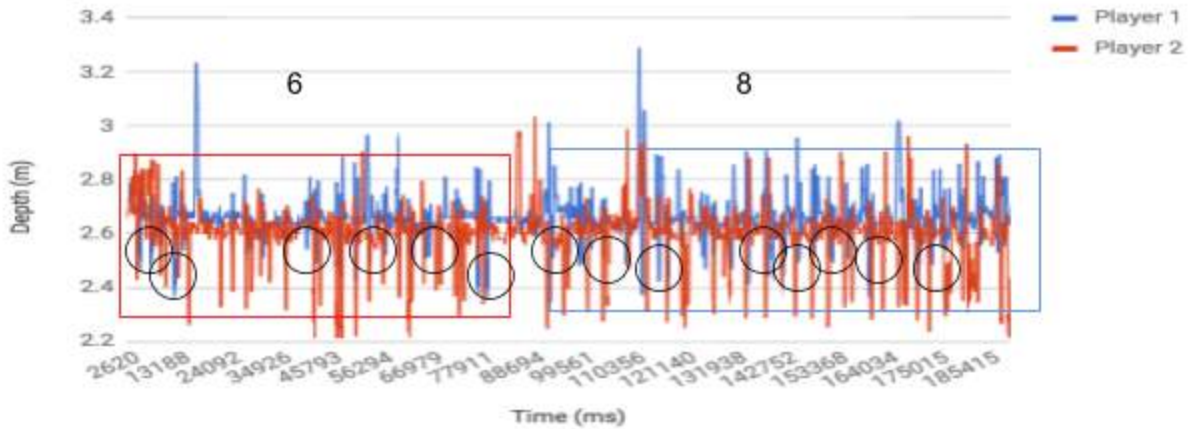
Playtest 6



Playtest 7



Playtest 8



Consent Forms

G

I am working for the University of Twente and doing research on visuals and gameplay and invite you to participate in my research through playtesting a game with the following game mechanics:

1. An abstract version of a car drives around a track, this car is slowing down over time.
2. If the car has halted, you lose the game. The only way to reset its speed is to win the round.
3. You win the round if you collect the same amount of collectables as the round number. F.e. to pass round 1 you will need to pick up one collectable, for round 2 two collectables etc.
4. To pick up the collectables, the car needs to touch them. Since the collectables are positioned more outwards than the car, the car needs to move further away from the center as well.
5. The car shifts to a more outwards position if a player jumps. If any player jumps soon after it will shift again.
6. The system will recognize your jumps if you position yourself above one of the four platforms.
- 7.

If you choose to participate in the game, you give permission to let the system extract information with regards to your vertical and horizontal position on the playground.*

If you do not understand the concept, please wait until after the demonstration to see if things are still unclear, if so feel free to ask your question(s).

- ☐ I give permission to be recorded for analyzing purposes.
- ☐ I give permission to let anonymized images of the playtest be used in a report.

Name participant

Autograph

- ☐ I give permission to be recorded for analyzing purposes.
- ☐ I give permission to let anonymized images of the playtest be used in a report.
-

Name participant

Autograph

- ☐ I give permission to be recorded for analyzing purposes.
- ☐ I give permission to let anonymized images of the playtest be used in a report.

Name participant

Autograph

- ☐ I give permission to be recorded for analyzing purposes.
- ☐ I give permission to let anonymized images of the playtest be used in a report.

Name participant

Autograph

Date:

Time:

Playtest nr:

Name researcher

Autograph

**During playtest participants were informed about the following sentence: You can exit the game at any time and you are not required to provide reasons for doing so, you will not be recognizable in any footage that is used for the report or other means besides analyzing purposes solely. Future consent forms should state these rights explicit in the consent form as well.*

Synchronized Jumping Code

```

public class DriverRotator : MonoBehaviour{
    void FixedUpdate() {
        transform.Rotate(Vector3.forward * curSpeed);
        curSpeed -= (deceleration * Time.deltaTime);
        if (roundText.enabled && !gameOver) {
            textTimer++;
            if (textTimer > textFrames) {
                textTimer = 0;
                roundText.enabled = false;
            }
        }
        if (curSpeed < 0 && !gameOver) {
            gameOver = true;
            ShowLevelInText();
        }
        if (Input.GetKeyDown(KeyCode.R)) {
            curSpeed = speed;
            roundText.enabled = false;
            collItems = 0;
            itemsToCollectBeforeSpeedReset = 1;
            deceleration = defDeceleration;
            interactionContr.jumps = 0;
            interactionContr.synchronizedJumps = 0;
        }
        if (Input.GetKeyDown(KeyCode.LeftArrow) && deceleration > 0f) {
            deceleration -= 0.001f;
        }
        if (Input.GetKeyDown(KeyCode.RightArrow)) {
            deceleration += 0.001f;
        }
    }
    public void ResetSpeed() {
        curSpeed = speed;
        collItems++;
        if (collItems >= itemsToCollectBeforeSpeedReset) {
            itemsToCollectBeforeSpeedReset++;
            collItems = 0;
            rendColor = rendColor.Equals(Color.black) ? Color.blue : Color.black;
            rend.color = rendColor;
            roundText.enabled = true;
            roundText.text = "Round \n" + itemsToCollectBeforeSpeedReset;
        }
    }
    void ShowLevelInText() {
        roundText.enabled = true;
        roundText.text = "You made it to level \n" + itemsToCollectBeforeSpeedReset;
    }
}

public class Collectable : MonoBehaviour{

```

```

void OnTriggerEnter2D(Collider2D collision) {
    if (collision.CompareTag(ColliderTag) && canBeCollected)
    {
        StartCoroutine(Timer());
        OnCollision();
    }
}

IEnumerator Timer() {
    canBeCollected = false;
    timer = impactResetDuration;
    while (timer > 0)
    {
        timer -= Time.deltaTime;
        yield return null;
    }
    canBeCollected = true;
}

}

public class GameController : MonoBehaviour{
    void Update() {
        UpdateDriverAlphaToSpeed();
        //Toggle gameplay pause
        if (Input.GetKeyUp(KeyCode.P))
            Time.timeScale = Time.timeScale < float.Epsilon ? 1f : 0f;
    }

    void UpdateDriverAlphaToSpeed() {
        Color driverCol = driverRend.color;
        driverCol.a = driverRotator.curSpeed / driverRotator.speed;
        driverRend.color = driverCol;
    }

    void HandleCollectableCollision() {
        driverRotator.ResetSpeed();
        StartCoroutine(UpdateCollectableSizeRotationAndPos());
    }

    IEnumerator UpdateCollectableSizeRotationAndPos() {
        for (int i = 0; i < collChangelter; i++) {
            ScaleCollectable(collectableScale.x, collectableScale.x * collSizeIncrease, (float)i/(float)collChangelter);
            collectable.transform.Rotate(Vector3.forward * Time.time * collRotationSpeed);
            if (i == collChangelter / 2)
                ResetCollectablePos();
            yield return null;
        }
        collectable.transform.localScale = collectableScale;
        collectable.transform.rotation = collectableRotation;
    }

    void ScaleCollectable(float a, float b, float t)
    {
        collectable.transform.localScale = new Vector3(Mathf.Lerp(a, b, t), Mathf.Lerp(a, b, t), driver.transform.localScale.z);
    }

    void HandleJumperChange(int numJumpers) {
        if (numJumpers == synchr_jumping_off)
        {
            trackReleaseDuration = InteractionController.instance.singleJumpDuration * jumpVsReleaseTime;
            CondMoveTrackUp();
        }
        else {
            trackReleaseDuration = InteractionController.instance.synchJumpDuration;
        }
    }
}

```

```

        ChangeDriversTrack(numJumpers - 1);
    }
}
void CondMoveTrackUp() {
    if (curTrack < trackPositions.Length - 1) {
        ChangeDriversTrack(curTrack + 1);
        StopCoroutine(TrackReleaseTimer());
        StartCoroutine(TrackReleaseTimer());
    }
}
IEnumerator TrackReleaseTimer() {
    jumpTimer = trackReleaseDuration;
    while (jumpTimer > 0f)
    {
        jumpTimer -= Time.deltaTime;
        yield return null;
    }
    if (curTrack > 0)
        ChangeDriversTrack(curTrack-1);
}
void ChangeDriversTrack(int trackNr) {
    Vector3 driverPos = driver.transform.localPosition;
    driverPos.x = trackPositions[trackNr];
    driver.transform.localPosition = driverPos;
    curTrack = trackNr;
}
void ResetCollectablePos() {
    int randomAngle = Random.Range(0, 360);
    collectableRotator.transform.Rotate(Vector3.forward, randomAngle);
    int newTrackForCollectable = Random.Range(1, trackPositions.Length);
    float distanceFromCenter = trackPositions[newTrackForCollectable];
    Vector3 collectablePos = collectable.transform.localPosition;
    collectablePos.x = distanceFromCenter;
    collectable.transform.localPosition = collectablePos;
}
}
public class HeightTracker : MonoBehaviour {
    void FixedUpdate() {
        height = (int)Util.Map(transform.position.z, minHeight, maxHeight, 0f, 100f);
        if (Input.GetKeyDown(KeyCode.N))
            tresholdScale -= 0.01f;
        if (Input.GetKeyDown(KeyCode.M))
            tresholdScale += 0.01f;
    }
    void OnTriggerEnter2D(Collider2D collision) {
        if (collidingObject.CompareTag(collision.tag)) {
            activeCollidingObject = collision.gameObject;
            StartCoroutine(coroutine);
        }
    }
    void OnTriggerExit2D(Collider2D collision) {
        if (collidingObject.CompareTag(collision.tag)) {
            StopCoroutine(coroutine);
        }
    }
}
IEnumerator ListenForJump()
{

```

```

jumpTreshold = height * tresholdScale;
avgHeight = height;
measuredHeights.Clear();
while (true) {
    measuredHeights.Add(height);
    UpdateAVGHeight();
    jumpTreshold = avgHeight * tresholdScale;
    if (height > jumpTreshold)
    {
        InteractionController.instance.ActivatePlatform(activeCollidingObject);
        visualiser.OnJump();
        jumpCount++;
    }
    yield return null;
}
}
void UpdateAVGHeight() {
    float sum = .0f;
    foreach (float h in measuredHeights)
        sum += h;
    avgHeight = sum / measuredHeights.Count;
}
}
public class InteractionController : MonoBehaviour {
    void Update()
    {
        if (Input.GetKeyUp(KeyCode.E)){
            enticingElements = !enticingElements;
            jumps = 0;
            synchronizedJumps = 0;
            ResetConnections();
        }
        if (Input.GetKeyDown(KeyCode.S))
            SaveData();
        if (Input.GetKeyUp(KeyCode.Alpha6))
            ActivatePlatform(platforms[0]);
        if (Input.GetKeyUp(KeyCode.Alpha7))
            ActivatePlatform(platforms[1]);
        if (Input.GetKeyUp(KeyCode.Alpha8))
            ActivatePlatform(platforms[2]);
        if (Input.GetKeyUp(KeyCode.Alpha9))
            ActivatePlatform(platforms[3]);
    }
    private void SaveData() {
        string path = writePath + DateTime.Now.GetHashCode().ToString() + " " + enticingElements + " " + Time.frameCount + ".txt";
        writer = new StreamWriter(path, true);
        writer.WriteLine("Jumps " + jumps + ", Synced jumps: " + synchronizedJumps + "\n enticing elements: " + enticingElements);
        writer.Close();
        UIText.enabled = true;
        UIText.text = "Saved file";
    }
    void ResetConnections() {
        for (int i = 0; i < platformConnections.Length; i++)
        {
            platformConnections[i] = i;
            UpdateNeonElements(i);
        }
    }
}

```



```

}
public void ActivatePlatform(GameObject platform) {
    int index = PlatformIndex(platform);
    Color platformCol = Color.red;
    platformCol.a = 0.8f;
    platforms[index].GetComponent<SpriteRenderer>().color = platformCol;
    //Do nothing if already activated
    if (activePlatforms[index] == 1)
        return;
    activePlatforms[index] = 1;
    if (!synchronizedJumping)
        OnJumperChange(SYNCHR_JUMPING_OFF);
    MultiJump(index);
    StartCoroutine(PlatformReleaseTimer(index));
    jumps++;
}
int PlatformIndex(GameObject pf) {
    for (int i = 0; i < platforms.Length; i++)
        if (platforms[i].Equals(pf))
            return i;
    return int.MinValue;
}
void MultiJump(int index) {
    activeJumpers++;
    for (int i = 0; i < activePlatforms.Length; i++) {
        if (CanAttachToPlatform(index, i))
        {
            platformConnections[index] = i;
            synchronizedJumps++;
            ConditionalUpdateElementsAndJumpers(index);
            break;
        }
    }
}
bool CanAttachToPlatform(int thisPlatform, int otherPlatform) {
    return thisPlatform != otherPlatform && activePlatforms[otherPlatform] == 1
        && platformConnections[otherPlatform] == otherPlatform;
}
IEnumerator PlatformReleaseTimer(int index) {
    float jumpTimer = synchronizedJumping ? synchJumpDuration : singleJumpDuration;
    while (activePlatforms[index] == 1) {
        jumpTimer -= Time.deltaTime;
        if (jumpTimer < 0f) {
            platformConnections[index] = index;
            ConditionalUpdateElementsAndJumpers(index);
            activeJumpers--;
            activePlatforms[index] = 0;
            jumpTimer = synchJumpDuration;
            platforms[index].GetComponent<SpriteRenderer>().color = defPlatformColor;
        }
        yield return null;
    }
}
void ConditionalUpdateElementsAndJumpers(int platformIndex) {
    if (enticingElements)
        UpdateNeonElements(platformIndex);
    if (synchronizedJumping)

```

```

        OnJumperChange(activeJumpers);
    }
    void UpdateNeonElements(int platformIndex) {
        GameObject attachP = platforms[platformIndex];
        GameObject reachP = platforms[platformConnections[platformIndex]];
        Vector2 attachedPosition = attachP.transform.position;
        Vector2 reachPosition = reachP.transform.position;
        float lerpNum = 1 / (numNeonElements + 1f);
        float lerpAdd = lerpNum;
        Vector3 lerpPos = Vector3.zero;
        for (int j = 0; j < numNeonElements; j++)
        {
            lerpPos.x = Mathf.Lerp(attachedPosition.x, reachPosition.x, lerpNum);
            lerpPos.y = Mathf.Lerp(attachedPosition.y, reachPosition.y, lerpNum);
            //activate element if connected
            lerpPos.z = (Vector3.Distance(attachedPosition, reachPosition) > .1f) ? 1f : 0f;
            lerpNum += lerpAdd;
            neonElementsPos[platformIndex * numNeonElements + j] = lerpPos;
        }
        textureController.NeonElementsPos = neonElementsPos.ToArray();
    }
}

public class PlayerVisuals : MonoBehaviour{
    void FixedUpdate() {
        height = GetComponentInParent<HeightTracker>().height;
        MapColor();
        MapScale();
    }
    void MapColor() {
        if (invisible)
            return;
        Color mappedColor = new Color(ScaledComp(lowColor.r, hightColor.r),
            ScaledComp(lowColor.g, hightColor.g),
            ScaledComp(lowColor.b, hightColor.b),
            ScaledComp(lowColor.a, hightColor.a));
        rend.color = mappedColor;
    }
    public void MakeVisible() {
        invisible = false;
    }
    public void MakeInvisible() {
        rend.color = new Color(0, 0, 0, 0);
        invisible = true;
    }
    public void OnJump() {
        transform.Rotate(Vector3.forward * Time.time * 10);
    }
    void MapScale() {
        float mappedVal = ScaledComp(minScale, maxScale);
        transform.localScale = new Vector3(mappedVal, mappedVal, 1f);
    }
    float ScaledComp(float comp1, float compo) {
        return Util.Map(height, 60, 100, comp1, compo);
    }
}

public class TextureController : MonoBehaviour{
    IEnumerator UpdateCounter() {

```

```

while (true) {
    if (NeonElementsPos != null) {
        count %=( numNeonElements / 2);
        count++;
    }
    yield return null;
}
}
IEnumerator UpdateTexture() {
    while (true) {
        if (PlatformsPos != null)
        {
            UpdatePixels();
            texture.SetPixels(pixels);
            texture.Apply();
        }
        yield return wait;
    }
}
void UpdatePixels() {
    for (int x = 0; x < textureWidth; x++)
        for (int y = 0; y < textureHeight; y++)
            pixels[x + y * textureWidth] = PlatformPosBasedColor(textureWidth - x, y);
}
Color PlatformPosBasedColor(int x, int y) {
    float h = 0f, s = 0f, v = 0f;
    foreach (Vector3 platformPos in PlatformsPos) {
        float colorScale = (bgLightning /
            Vector2.Distance(new Vector2(x, y), platformPos));
        h += colorScale * effectHSV.r;
        s += colorScale * effectHSV.g;
        v += colorScale * effectHSV.b;
    }
    for (int i = 0; i < neonElementsPos.Length; i++) {
        if (neonElementsPos[i].z > float.Epsilon && i % numNeonElements != count) //If elements are connected
        {
            float colorScale = ((bgLightning * neonLightning) /
                Vector2.Distance(new Vector2(x, y), neonElementsPos[i]));
            h += colorScale * interEffectHSV.r;
            s += colorScale * interEffectHSV.g;
            v += colorScale * interEffectHSV.b;
        }
    }
    return new Color(h, s, v);
}
Vector2[] WorldToTexturePos(Vector2[] worldPos) {
    Vector2[] texturePos = new Vector2[worldPos.Length];
    for (int i = 0; i < worldPos.Length; i++)
        texturePos[i] = WorldToTexturePos(worldPos[i]);
    return texturePos;
}
Vector3[] WorldToTexturePos(Vector3[] worldPos) {
    Vector3[] texturePos = new Vector3[worldPos.Length];
    for (int i = 0; i < worldPos.Length; i++)
        texturePos[i] = WorldToTexturePos(worldPos[i]);
    return texturePos;
}
}

```

```

Vector3 WorldToTexturePos(Vector3 worldPos) {
    float xTexture = Util.Map(worldPos.x, upperLeft.x, lowerRight.x, 0, textureWidth);
    float yTexture = Util.Map(worldPos.y, upperLeft.y, lowerRight.y, 0, textureHeight);
    return new Vector3(xTexture, yTexture, worldPos.z);
}
}

public class DataFeeder : MonoBehaviour{
    void ReadCoordInput() {
        positions.Clear();
        StreamReader reader = new StreamReader(path + "/" + fileName);
        string line = string.Empty;
        while ((line = reader.ReadLine()) != null)
        {
            string[] coord = line.Split(' ');
            Vector3 c = new Vector3(float.Parse(coord[0]), float.Parse(coord[2]), float.Parse(coord[1]));
            positions.Add(c);
        }
        reader.Close();
    }

    void ReadEdgeCoordInput() {
        List<float> edges = new List<float>();
        StreamReader reader = new StreamReader(path + "/" + edgesFileName);
        string line = string.Empty;
        while ((line = reader.ReadLine()) != null)
        {
            edges.Add(float.Parse(line));
        }
        reader.Close();
        inputUpperLeftLow = new Vector3(edges[0], edges[4], edges[2]); //0 xmin, 1 xmax, 2 ymin, 3 ymax, 4 zmin, 5 zmax
        inputBottomRightHigh = new Vector3(edges[1], edges[5], edges[3]); , game axes y is for jumping
    }

    Vector3 ScreenToWorld(float x, float y, float z) {
        return Camera.main.ScreenToWorldPoint(new Vector3(x, y, z));
    }

    void Update() {
        if (Input.GetKeyDown("space"))
            StartCoroutine(MovePlayer());
    }

    IEnumerator MovePlayer() {
        player = GameObject.FindWithTag("Player").GetComponent<Player>();
        heightTracker = player.GetComponent<HeightTracker>();
        foreach (Vector3 coord in positions) {
            Vector3 mappedCoord = InputToWorldPoint(coord);
            player.moveTo(mappedCoord.x, mappedCoord.y, mappedCoord.z);
            yield return wait;
        }
    }

    Vector3 InputToWorldPoint(Vector3 input) {
        float lerpedX, lerpedY, lerpedZ;
        lerpedX = Util.Map(input.x, inputUpperLeftLow.x, inputBottomRightHigh.x, worldUpperLeftLow.x, worldBottomRightHigh.x);
        lerpedY = Util.Map(input.y, inputUpperLeftLow.y, inputBottomRightHigh.y, worldUpperLeftLow.y, worldBottomRightHigh.y);
        lerpedZ = Util.Map(input.z, inputUpperLeftLow.z, inputBottomRightHigh.z, worldUpperLeftLow.z, worldBottomRightHigh.z);
        return new Vector3(lerpedX, lerpedY, lerpedZ);
    }
}

```

Jump Simulation Code

I

The code of the program as explained in Appendix D.

```
void draw()
{
    kinectTracker.run();
    jumpCanvas.run();
}
class JumpCanvas
{
    void run()
    {
        updatePlayerPosToHighestPoint();
        outputJumpersToText();
        drawTrackingLines();
        drawFadingCanvas();
    }
    private void updatePlayerPosToHighestPoint()
    {
        PVector highestPoint = kinectTracker.getHighestPoint();
        if (highestPoint.y > yTop && highestPoint.y < yBottom)
            player.runOnPosition(highestPoint.x + xOffset, highestPoint.y);
    }
    private void outputJumpersToText()
    {
        fill(0);
        for (int playerId : jumpTracker.jumpingPlayers())
            text("Player " + playerId + " is jumping", xOffset + 100, 100);
    }
    private void drawTrackingLines()
    {
        strokeWeight(trackerLinesWeight);
        line(kinect.width, yTop, width, yTop);
        line(kinect.width, yBottom, width, yBottom);
    }
    private void drawFadingCanvas()
    {
        fill(255, 10);
        rect(xOffset, -rectMargin, width + rectMargin, height + rectMargin);
    }
}
class JumpTracker
{
    ArrayList<Player> players;
    float sdMultiplier;
    JumpTracker()
    {
        this(1);
    }
}
```

```

JumpTracker(float sdMultiplier)
{
    players = new ArrayList();
    this.sdMultiplier = sdMultiplier;
}
void addPlayer(Player player)
{
    players.add(player);
}
void removePlayer(Player player)
{
    players.remove(player);
}
ArrayList<Integer> jumpingPlayers()
{
    ArrayList<Integer> jumpers = new ArrayList();
    for (Player player : players)
        if (isJumping(player))
            jumpers.add(player.id);

    return jumpers;
}
private boolean isJumping(Player player)
{
    return (player.getCurrentHeight() < (player.avgHeight() - player.standardDeviation(sdMultiplier)));
}
}
class KinectTracker
{
    PVector getHighestPoint()
    {
        return highestPoint;
    }
    void run()
    {
        setDepthToMouse();
        update();
        display();
    }
    private void update()
    {
        updateTresholdImage();
        updatePixelsInRange();
        updateHighestPoint();
    }
    private void setDepthToMouse()
    {
        kinectTracker.minDepth = mouseX;
        kinectTracker.maxDepth = mouseY * 2;
    }

    private void updateTresholdImage()
    {
        int[] rawDepth = kinect.getRawDepth();
        for (int i = 0; i < rawDepth.length; i++)
            thresholdImg.pixels[i] = isInRange(rawDepth[i]) ?
                inTresholdColor : defaultColor;
    }
}

```



```

    thresholdImg.updatePixels();
}
private boolean isInRange(float currentDepth)
{
    return currentDepth > minDepth && currentDepth < maxDepth;
}
private void updatePixelsInRange()
{
    pixelsInRange.clear();
    for (int i = 0; i < thresholdImg.width * thresholdImg.height; i++)
        if (thresholdImg.pixels[i] == inThresholdColor &&
            i > thresholdImg.width * 4 && i < (thresholdImg.width * thresholdImg.height) - thresholdImg.width * 4) //avoid errors at the margins
            pixelsInRange.add(i);
}
private void updateHighestPoint()
{
    PVector record = new PVector(0, Float.POSITIVE_INFINITY);
    PVector point;
    for (int i : pixelsInRange)
    {
        point = indexToPVector(thresholdImg.width, i);
        if (point.y < record.y)
            record = point;
    }
    highestPoint = record;
}
private PVector indexToPVector(float imgWidth, int index)
{
    return new PVector((int)index % imgWidth, (int)index / imgWidth);
}
private void display()
{
    image(thresholdImg, 0, 0);

    fill(highestPointFill);
    ellipse(highestPoint.x, highestPoint.y, 40, 40);
}
}
class Player
{
    int getId()
    {
        return id;
    }
    void runOnPosition(float x, float y)
    {
        setPosition(x, y);

        updateHeights();

        display();
    }
    private void setPosition(float x, float y)
    {
        position.x = x;
        position.y = y;
    }
}

```

```

}
private void display()
{
    strokeWeight(playerDotWeight);
    stroke(playerDotColor);
    point(position.x, position.y);
}
private void updateHeights()
{
    measuredHeights.add(position.y);
}
float getCurrentHeight()
{
    float h = 0;
    if (measuredHeights.size() > 0)
        h = measuredHeights.get(measuredHeights.size() - 1);
    return h;
}
float avgHeight()
{
    float sum = .0;
    for (float h : measuredHeights)
        sum += h;

    return sum / measuredHeights.size();
}
float standardDeviaton(float mult)
{
    return sqrt(variance()) * mult;
}
private float variance()
{
    float avgHeight = avgHeight();
    float sum = 0;
    for (float h : measuredHeights)
    {
        float deviaton = h - avgHeight;
        float sqDeviation = pow(deviaton, 2);
        sum += sqDeviation;
    }

    return sum / measuredHeights.size();
}
void reset()
{
    measuredHeights.clear();
}
}}

```

Data Mangling Code

J

As used for the 2D and 3D representation of logged coordinates from the playground.

```
void draw()
{
    background(255);
    translate(width / 2, height / 2, -50);
    if ((mouseX / 100) > 1)
        rotateY(PI * width / (mouseX));
    drawPoints();
}
void keyPressed()
{
    int k = key - 48;
    if (keyCode == RIGHT && fileNr < fileNames.length - 1)
    {
        fileNr++;
        updateMangler();
    }
    else if (keyCode == LEFT && fileNr > 0)
    {
        fileNr--;
        updateMangler();
    }
    else if (k >= 0 && k < 6)
    {
        updateManglerWithFilter(k);
        println("player " + k);
    }
    if (key == 's')
    {
        String[] coordString = mangler.coordToString();
        println("Coord");
        printStrings(coordString);
        saveStrings(path, coordString);
        String[] endings = endCoord();
        println("Endings");
        printStrings(endings);
        saveStrings(writePath + edgesFileName, endings);
    }
}
void updateMangler()
{
    updateManglerWithFilter(0);
}
```

```

void updateManglerWithFilter(int nr)
{
    mangler.setInput(loadStrings(fileNames[fileNr] + ".txt"));
    mangler.filterPlayer(nr);
    mangler.updateCoord();
    coord = mangler.getCoord();
    println("\n\n\n\n\n\n\n" + fileNames[fileNr]);
}
void printStrings(String[] strings)
{
    for (String s : strings)
        println(s);
}
//xmin, xmax, zmin, zmax, ymax, ymin, for convergence with clean playground world coordinates
String[] endCoord()
{
    String[] endings = new String[]{
        minAndMaxPos(1)[0], minAndMaxPos(1)[1], minAndMaxPos(2)[0],
        minAndMaxPos(2)[1], minAndMaxPos(3)[1], minAndMaxPos(3)[0]
    };
    return endings;
}
String[] minAndMaxPos(int axes)
{
    String[] pos = mangler.floatToString(
        mangler.orderedFloats(axes));
    return new String[] { pos[0], pos[pos.length - 1] };
}
void drawPoints()
{
    for (int i = 0; i < coord.length; i++)
    {
        PVector v = coord[i];
        PVector point = new PVector(v.x * factor, v.z * factor - mouseY, v.y * factor);
        pushMatrix();
        stroke(0);
        strokeWeight(2);
        //translate(point.x,point.y,point.z); //y and z interchanged
        point(point.x, point.y, point.z);
        if (i > 0)
            line(point.x, point.y, point.z, prevPoint.x, prevPoint.y, prevPoint.z);
        popMatrix();
        prevPoint = point;
    }
}
class DataMangler
{
    PVector[] coord;
    String[] input;
    int lineReadInterval;

    DataMangler(String[] input, int lineReadInterval)
    {
        this.input = input;
        this.lineReadInterval = lineReadInterval;
    }
    PVector[] getCoord()

```

```

{
    return coord;
}
void setInput(String[] input)
{
    this.input = input;
}
void filterPlayer(int nr)
{
    int counter = 0;
    ArrayList<String> linesPlayer = new ArrayList();
    for (int i = 0; i < input.length; i++)
    {
        String line = input[i];
        println(nr + " " + input[i].split(",")[1]);
        if (input[i].split(",")[1].equals(Integer.toString(nr)) && counter % lineReadInterval == 0)
            linesPlayer.add(line);
        counter++;
    }
    input = linesPlayer.toArray(new String[0]);
}
void updateCoord()
{
    coord = new PVector[input.length];
    String[] splits;
    for (int i = 0; i < input.length; i++)
    {
        splits = input[i].split(",");
        coord[i] = new PVector(Float.parseFloat(splits[2]), Float.parseFloat(splits[3]), Float.parseFloat(splits[4]));
    }
}
String[] coordToString()
{
    String[] ret = new String[coord.length];
    for (int i = 0; i < coord.length; i++)
    {
        String cS = "";
        PVector cPV = coord[i];
        cS += cPV.x + " " + cPV.y + " " + cPV.z;
        ret[i] = cS;
    }
    return ret;
}
float[] orderedFloats(int axes)
{
    float[] ret = axisValues(axes);
    for (int j = 1; j < ret.length; j++)
        for (int i = 1; i < ret.length; i++)
            if (ret[i - 1] > ret[i])
            {
                float temp = ret[i - 1];
                ret[i - 1] = ret[i];
                ret[i] = temp;
            }

    return ret;
}

```

```

String[] floatToString(float[] floats)
{
    String[] ret = new String[floats.length];
    for (int i = 0; i < floats.length; i++)
    {
        ret[i] = "" + floats[i];
    }
    return ret;
}

float[] axisValues(int axis)
{
    float[] axes = new float[coord.length];
    for (int i = 0; i < coord.length; i++)
    {
        axes[i] = getValue(coord[i], axis);
    }

    return axes;
}

float getValue(PVector a, int axes)
{
    switch (axes)
    {
        case 1:
            return a.x;
        case 2:
            return a.y;
        case 3:
            return a.z;
    }

    return 0.0f;
}
}

```