Final thesis

Effects of design and feedback in a motion-based game

by

Rasmus Cronstrand

LIU-IDA/LITH-EX-G--15/061--SE

2015-07-07
Final Thesis

Effects of design and feedback in a motion-based game

by

Rasmus Cronstrand

LIU-IDA/LITH-EX-G--15/061--SE

2015-07-07

Supervisor: Erik Berglund
Examiner: Anders Fröberg
Effects of design and feedback in a motion-based game

Rasmus Cronstrand
LiTH
Linkoping
rascr534@student.liu.se

ABSTRACT
Movement-based games are becoming more frequent in everyday lives. With easier access to webcams built into laptops and web-based games, a game which utilises both concepts can become a good option for everyday gaming. This study evaluates a movement-based game written in JavaScript using phaser.io which uses a webcam for control and HTML5 technique for capture. The main questions in the study are what observations can be done and are there any problems when developing a game with webcam-based motion-detection in regards to factors like flow, feedback and positioning. This study tries to answer these questions by building upon an existing game and implementing new functions and feedback and then observing how these effect the game-play. The results of the study showed that it is important to consider good feedback and how to position oneself when playing such a game and that more testing should be done to gain further knowledge about these two. But also that many of the implementations done made the person achieve flow while playing. Further work should be beneficial also to make the game even better while keeping it easy to play.

Author Keywords
GameFlow; Movement based; gaming; phaser; HTML5; WebGL; Positioning; Feedback; webcam; web-based; JavaScript

INTRODUCTION
Creating a game for all audiences can be a difficult task. Casual gaming is becoming increasingly more popular these days[5]. The group of players which plays these games are also more heterogeneous[5] meaning that it’s not only men or women who play these games, but both. The problem is to create something new which has not been done before while still making it interesting. If you develop a game which uses motion-detection in the form of a webcam, more problems arises. The game should not be too hard to play, it should be intuitive without any instructions as well as being fun and interesting.

Other problems is the use of medium, a computer with a webcam is the obvious choice. But can it be used with a mobile device and it’s build in camera as well? One problem with using a mobile device is with the current technology for the implementation of the motion-tracking in the game is that it needs to be stable to be able to use it. This is because it filters out things that do not move against things that really do move. Too much vibration makes the filter think that everything is moving. So to be able to use such a device means that you need to have it on a stable surface while still being able to see the screen so you can play the game.

There is also considerations when designing a game based on motion-detection, how can you design the game so particular movement does not become awkward? There are also aspects of cheating. How can you design it so the player cannot cheat by positioning him or herself in favourable ways infront of the camera? Also when playing a game you should always be aware of what is happening and what you are doing. So are there ways to give certain feedback to make the game more intuitive for the player?

Research approach and objective
This experimental case study will implement a game in phaser.io with the use of a webcam for motion-tracking. The reason for the case study is to answer this question:

1. What observations can be done and are there any problems when developing a game with webcam-based motion-detection in regards to factors like flow, feedback and positioning?

Answering this question by testing a game with implementations will yield results about which implementations and considerations are valuable when creating a game.

Phaser.io
Phaser.io is a desktop and mobile HTML5 framework written in javascript. It is freely available and works on most browsers using WebGL or Canvas rendering.

Limitations
The game will only focus on use on a Computer and will therefore not take mobile devices as mobilephones in consideration. Extensive testing with several test-groups was not done because of time-limitations. The focus was on the game-development where testing and evaluation were done gradually with one larger test at the end.

THEORY

This work is licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc/3.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.
Game idea
The original main idea of the game (See figure 1) is to make a tree grow by giving it water-droplets. There are multiple paths on the sides of the tree where water droplets and bugs spawn at the top and continue their way down to base/root of the tree. At the root bugs eat away one part of the base and drops make the tree grow one part. By making motions that get picked up by the webcam, objects such as bugs can be "smashed" as the webcams motion is check for collisions against objects by Phaser. The webcammotion is rendered as an object aswell.

When you "smash" for example a bug, the bug-object and motion-object collide and function is called to handle the collision. The function for handling the collision removes the object when a collision is detected and in the case of a bug, an animation is rendered before it is removed. If one fails to remove a bug, the tree shrinks as opposed to the water-droplet which makes the tree grow. The features for expanding the base game-play was concluded in this study to make the game more enjoyable to play and to achieve better flow.

Background
Using movement in games has become more common in the 21st century with the appearence of the Nintendo Wii, Sony Eye-toy and the Microsoft Kinect. Before the ability to play motion-games in your livingroom on a tv, games like Dance Dance Revolution and Virtual Cop were only available in arcades. Nowadays the cost of a console with similar functions is small compared to the arcade-machines. You can now play many titles at home where you use your body to control the games, with or without a controller attached.

Related work
Playing a game with the webcam interface used is similar to playing the Wii in regards to that the physical play space becomes more important.[9] When you play Wii with the controller or with the webcam you can say that the physical space infront of the screen becomes an "interface" as the space can be interfered with. The interfacing is similar though the movements dont exactly match the game on the screen with the Wii motion sensor as with the webcam-interface where you can see a sort of pixelated rendering of your movement.

Though this can also vary as the resolution of the webcam and the placement of the player makes the rendering larger or smaller. This makes the skill of the game inconsistent and as for using the wiimote, players need to learn a new form of body awareness[9] which is important to take into consideration when designing the game.

In regards to natural movement a "interface used by a game is natural if the actions used to play it have a natural interpretation based on the context of the game"[8]. Which means that if it is not a "natural" movement for the action, for example a boxing-game that uses kicks for punches, it could cause an interruption in flow.

Considering this, the game implemented in this study could have problem achieving flow, where "Flow denotes the holistic sensation present when we act with total involve-
ment.”[2] This is as squashing bugs is not really naturally done with your hands but if you consider doing it, you probably would use your feet with shoes on or maybe a rolled up paper.

Earlier studies have shown that movement while playing a game might create more fun[4] and by having different kinds of immersion in movement you can increase the energy[11] of the player. This means that if you can increase the energy of the player you can have more fun, although the movement in a game should not be too exhaustive it should still be tricky enough to challenge the player. Which is important when considering the positioning in the game as if the player is too far away he might have to use too much movement making the game less enjoyable.

The precision which the user can use when playing a game is also important[7], while the studies have been on controllers like the WiiMote[7]; increased precision could have effect on control-less webcam-based immersion as well. But it should not be too hard as too much focus on winning the game can take away the fun in some situations[1].

If taking in regard that exaggerated movement has also been shown to be rewarding[3] for players. There should be means to that a precise skillful action should have a larger outcome for the player in the game.

Feedback from a game is also important. One study finds that feedback and sensors might not make a game more engaging by making it more advanced[6] this because the social aspect of a game. A “broken” game can make the player make the rules[6] which might be of little use in this study but can be important when looking into the aspect of a person nearby telling the player to do something. In that aspect it can be important to not make the feedback too great, but enough so that a player playing alone can understand what to do, but not instantly.

Webcam
The webcam integration in the game uses the getUserMedia/Stream API to access the webcam. As this is not supported on all browsers, the game will not be playable on every platform. The current browsers not capable of using this method is Internet Explorer, Safari and some mobile-browsers[5]. The main browser which is used for the development is Google Chrome, this is because of a currently odd behaviour in Firefox where the update-framerate of the camera makes the rendering of the player silhouette flicker. Another candidate could have been Opera browser but was never considered as Chrome worked well. The original game used an old version of the webcam-implementation which rendered on a canvas, this forces Phaser to use canvas as well and as performance with WebGL is much greater[2] a WebGL-version of the webcam-implementation was developed in parallel with the game.

Implementation
There are several frameworks for developing games that run inside webbrowsers[3]. As this study will be using such framework, there are some considerations when choosing an appropriate one. When choosing the main points were:

- It should be widely available
- Frequent updates
- WebGL and Canvas rendering
- Compatibility with motion-detection implementation

Phaser.io is a choice that works on most web-browsers with two modes, canvas and webgl[4]. So considering availability phaser.io is a very good choice. Also the functionality for motion-tracking were already tested and implemented in phaser.io so to ensure the compatibility phaser.io will be used for the game. The base of the game is also implemented in it so choosing another would mean more time for development. It is also developed under MIT-licensing which makes it freely available.

Tracking movement
As mentioned earlier a webcam-implementation is used that tracks the user in a 2D-space with the help of a webcam. As you move it compares against the previous sequence and creates objects on a canvas in Phaser where it differentiates. The game can implement both small and big gestures when playing the game. The player playing the game can “cheat” by waving the hand in-front of the camera but implementations should strive to reduce this.

The implementations in the game can interact with certain kind of movements. One example of this is the case of clearing all enemies when a specific water-droplet is acquired, a bigger gesture can be used over the screen. Other features could use movements that needs a more precise/accurate movement to work.

Development Environment
When developing the game a customized environment was used. The editor which was used is called Emac[5] and is freely available. In Emac you can use plugins which aids in the development process. By installing plugins in Emac you get certain “Modes” with files for example .js-files which would be modes for JavaScript. As Emac is highly customizable you can specify which files uses certain modes.

Development process
When implementation of new feature in the game were made the following steps were taken:

- Discussion about feature
- Evaluate if it would add value to the game
- Evaluate if it is implementable

---

• Scrap the idea or Implement it
• Post-implementation – re-evaluate

As shown in figure 2. By following these steps timwasting on implementing "bad" ideas were cut although if the post-implementation evaluation would mean the scrapping of an idea some time were wasted. The discussion was mainly done with the customer, but if it was a small feature a nearby colleague was asked.

Development evaluation
When doing the subjective evaluation of the feature the value was defined by:
• Would it be fun?
• Would it be too hard to master?
• Would it be an awkward movement?

And the implementable criteria:
• Does it work with the webcam-implementation?
• Would it make a current feature obsolete?
• Would it take too long time?

By answering the questions a simple evaluation in the figure 2 made it easier for the development of the game. All these questions were answered informally during the discussion about a feature.

Agile Development
A modified type of SCRUM was used in the development process. As the development was carried out by one person and not a team, a diary was used instead of stand-ups. By using a diary where problems that arose during the development and solutions to earlier problems were written down after each session of development, the process of evaluating the result was made easier. The main logging in the diary was based on commits made into the git repository and with the commit-message as base a short description/comment was made under each thing added/removed.

A sort of SCRUM-board was also used where tasks were set up after they came up under discussion. The platform used was Asana which helped in the development-process by having the tasks organised by categories in the game.

Each week a meeting with the product owner was conducted to present the progress in the development as well as discuss the next step.

Internal testing
When developing the game in-house testing were used when needed for discussion about a feature. The testing used a very informal post-implementation evaluation(see figure 2) and were done with developers which were available nearby as well as the product owners to some extent. The testing sessions were documented and later used for the results.

Post-implementation Evaluation
The evaluation that was used in the final testing was a form made by Owen Schaffer (See APPENDIX) based on the GameFlow[10] model. The form makes it possible to measure if the player achieves flow. This is done by taking the following subjects into account:
• What to do
• How to do it
• How well doing
• Where to go
• Challenge
• Skill
• Freedom from distractions

The form used had a simple scoring method ranging from 1-5 where 1 is "Never", 3 is "About half of the time" and 5 is "Always". As the testing was done with the persons who had seen the game and had little time for the test, a too extensive form would be cumbersome considering time and effort.

6Http://asana.com
Participants
The test was conducted with a group of 8 persons in LiU Active Lab at Linköping University. The participants was all male students at Linköping University with various technical expertise as they studied some form of technical program. The age ranged from 20-27. If more time had been used to plan, a group with mixed participants would have been preferred to get more diversity. Because change of plans and little time left a group of friends and colleagues were chosen to test the game. Some of the participants had seen the game and had followed some of the development, this could have biased some of the result.

The setup
The test was conducted infront of a Philips large screen TV. On top of the TV was a Microsoft Kinect mounted. The Kinect was set to only use the webcam-sensor. The Kinect and TV was connected to a High-end Computer with Intel i7 and a unknown graphics card. The computer was tested and had a much higher FPS than a Intel i7 laptop without a graphics card.

When performing the test, the browser used was Google Chrome v.42 which used the feed of the kinect as input for the game. The participant choose the distance to the screen themselves and was encouraged to try to cheat if they wanted to and play as they liked.

Some of the participants watched as other participants played. To remove some bias from participants waiting to play, two other games were tested in rotation before playing the game.

Observation
The observational part in the testing did not use any standardised method. As the participants play observations was made by looking at patterns that were repeated by the players. What was looked at was:

- Positioning
- Enjoyment
- Awareness of objects
- Movement
- Cheating
- Understanding Feedback
- General exhaustion

If two or more of the observed participants showed a similar ‘pattern’ the ‘pattern’ was considered to of value to the results.

EARLY OBSERVATIONS
Before testing the almost finished game a couple of early observations were made of players playing the game. The observations were made when time were given with people playing a early version of the game in various situations. This made room for some early changes in feedback of the game.

In one case a child were observed playing the game in front of a webcam and were very energetic. The player loudly wondered what the golden-drop were supposed to do and did not know how to use it. As you were not supposed to take the drops, taking the golden-drop to activate the power-up was not a natural thing. Considering this the way to activate the drop were changed to activate on hitting the tree.

Another thing noticed early in the development were the “risk” of cheating by jumping in front of the camera or swiping over it with the hand. As the player did not understand the value of the water drops, he could sacrifice these easily by doing just that, swiping.

As score were introduced to the game considerations were made if droplets should give minus-points when hit. At first this made the game somewhat better for not being able to camswipe as you got points deducted from your score. Considering the negative feedback of losing points the multiplier was introduced that gave less multiplier when a drop was touched and more multiplier when a drop hit the tree.

This allowed the player which were only out for scoring points not being able to camswipe. Also if five consecutive drops were touch the multiplier would go down to zero which rendered camswiping useless. You might get to the next level by letting the tree grow, but you don’t get any points by doing it.

The sensitivity of the camera was noticed to be too great at first and picking up too much static which interfered with the game. By increasing the threshold the camera picked up less static and the player could disappear faster if this was sought for. Optimisation in phaser was also tested by rendering the image to a bitmap instead of a texture which made little difference but a slow drop in performance was noticed and was later reverted.

The performance of the game was a key point to optimise. The level design were made in Tiled which caused a draw on the canvas for each layer. As the game had several pathways in which the objects moved through, each path had its own layer plus the background. This was optimised by removing the Tiled-map altogether and replace it with one exported image from the tiled editor.

IMPLEMENTATIONS

Bugs
Bugs (see figure 3) were easy to hit from the beginning where only one hit was enough to kill the bug. Therefore a simple hitpoint system was made where the player had to touch the bug multiple times to kill it. As the webcam-overlay had high
framerate the collision-detection were made several times in a short amount of time. This did not allow the hitpoint system to use a simple way of giving the bug a low hp of for example 4 as when the user touched the bug one time, the hits counted up as fast as the current framerate.

To solve this the bugs had to have a higher hp which in theory accounted for the amount of time you had to hover over the bug where the internal hp-variable decremented until it reached zero before killing the bug. Another solution to this could have been to set a variable to each bug telling phaser to ignore the bug if it already had been hit although this solution would be less natural for the player as the player would have to retract the movement from the bug in order to hit it again.

The hitpoint-system made the player confused as they touched the bug with higher hp and “nothing happened” and on the bugs with lower hp were killed instantly. Aware of this problem a need for feedback emerged. The solution to this were discussed and in the first implementation the bugs just froze in place. After some testing and more discussion this solution made the bugs too easy to hit, so the freezing was left out and instead an animation was made. The animation made the bug “shield” itself by extending the wings. After some small testing this seemed to solve the problem.

Making the bugs have hitpoints made it harder to take the bug right before reaching the tree, so instead of using the old freeze implementation a tween was added so that the bug bounced back in its path. This feedback made the player able to the last second smashing of the bug before the tree. Discussion was also made here if they really should bounce back because of the player becoming too static in the movements when playing because you need to shake at the bugs position.

A speed-variable was also implemented into the game to make it more dynamic. The speed variable was implemented so that the bugs had different speeds when entering their path where the speed determined how many steps down the path a bug would make on every update. This allowed the game to have very fast bugs.

**Speed Bug**

The Speed Bug (See figure 6) was implemented to be a very fast bug. When “smashed” the Speed Bug emits a yellow-blue flash which kills all the bugs currently on the screen.

**Ice Bug**

The Ice Bug (See figure 6) was similar to the Speed Bug but when smashed, it emits a white flash which slows all the bugs and also bugs that spawns on the screen.

**Boss Bug**

The Boss Bug (See figure 6) is a bug with very high HP and very high score. The score was set to 100 points in which if a multiplier of 7x was active at the time, would give 700 points. Other than high points the Boss bug would not give player any game-advantage as the other special bugs. But if the player would not kill the Boss Bug it would take away 2 Tree trunks.

**Global Speed**

As the speed variable was introduced to the bugs a simple way of level progression was discussed as the bugs would get faster on every new level. The global speed-variable was multiplied with the bugs speed on spawning. On every new level the global speed-variable was incremented. This variable also made room for using the Ice Bug power up.

**Score and Points**

To make the player more emerged into the game a simple point-system was implemented (See figure 7). The bugs have points according to the difficulty of killing it. For example if it has higher HP, the bug will have higher points. To give the player feedback of the points, the point of the bug would appear on top of the killed bug and then tween to the location of the Score at the bottom. Also a highscore was implemented where the score on losing was set if it was higher, this because the game was up and running on box under development. As
the computer was restarted every night this highscore was the "highscore of today".

**Multiplier**

To stop the player from "screenswiping" a multiplier was introduced (See left side of figure 7) which was controlled by the drops in the game. The appearance of the multiplier was made with both text and visual representation. The multiplier-bar was created to resemble a charge-symbol similar to a battery for easy recognition for the player. Visual feedback was presented to the player by making the adjacent bar flash green when a multiplier was added and red when a multiplier was deducted. When a player touched a drop a multiplier was deducted and when a drop hit the tree, a multiplier was added. To distinguish the multiplier from the score, a second color was used for the multiplier.

**Water-tree**

For better feedback when reaching the 0x multiplier an icon where a small glowing tree would get water on itself with an animation was added (See figure 8. This feedback was added so when a player would notice that the score he was receiving from each bug was +0 he might look around to notice the tree which only appeared when the multiplier was down to 0x.

**Tree**

The tree (See figure 5 and 7 ) was introduced to be slowly growing all the time to make the game easier and playable even if you accidently touched the drops. Visual feedback was also added to represent the the multiplier by making the tree brown when the multiplier was low and greener when the multiplier was high, see figure5. When the multiplier is maxed out a sparkle and glow is added to the tree.

**Drop**

The drops did not change much from the original design. A "golden drop" was introduced to be able to shield the drops for a period of time. At first the design of the golden drop was made out to be a shield, for “shielding the drops” but after some discussion it was changed as the design was hard to understand.

The golden drop was still hard to understand what it did but no better way to describe the shielding of the drops came up. At first you touched the drop to make the other drops shielded. This was hard to understand as you were not supposed to touch the drops, which made it counterintuitive. The function later changed to letting the drops be shielded as the golden drop reached the tree. When activating the golden drop the other drops becomes more transparent by changing the alpha and when the timer starts to run out, they start to flash before being restored to their original alpha. A puddle appearing after a drop were touched was implemented as a feedback to represent that you had touched a drop.
Cloud
In order to minimize the way of screenswiping a line of moving clouds, see figure 9, was created. When a cloud was touched, points were deducted from the score. This implementation did not give any value to the game as the multiplier was introduced and the clouds were removed.

RESULT
The result of the flow questionnaire that each participant filled out after the test was analysed by using a median score as well as a sum of all the participants questionnaire score. As seen in the figure 10 the median score of the testing resulted in a score that was high, but not perfect as in highest score possible: All fives. The players always knew what to do by looking at the score but at the other parts it showed to be lacking a bit but was still a positive score as they knew “How to do it”, “How well doing” and “Where to go” more than half of the time. The “Challenge” and “Skill” showed that it takes skill to play the game but is does not demand all skill and it is not fully challenging either. “Freedom from distractions” showed that you do not get distracted while playing although the distractions could have been from people around while conducting the test.

To look into it further we can look at the combined sum(see figure 11) of each participant and see that the score here is also fairly high as not one participant rated below 25 of 35 possible.

Observations
The observations made while the participants tested the game resulted in some reconsiderations when designing the movement-based game. Following are comments about the participants that reoccurred more than once.

• Hard to understand golden drop
• Moves around a lot
• Ignores multiplier
• Finds the height/positioning matters
• Uses only arms
• Feeling sweaty
• Reaches to touch Bug
• Shakes arm/hand on bug
• Uses other bodyparts
• Rests with Ice Bug
• Misses Ice Bug
**DISCUSSION**

**Result**
The result should mean that the game could be set up and played at a location. Although the result could have been biased by using people who had seen the game before, the persons who had not seen the game before, had similar indications when they filled out the questionnaire. One participant kept on playing for a longer time than the rest of the group which revealed some flaws in the game design. One major flaw was as the bugs kept on going faster and faster it became easier for the player to smash them as the player could position itself to be static at the tree and catch them around it as when they spawned each bug came almost one and one except for the speedbugs. A solution to this could be to spawn more bugs as the level increases which would make it harder and more rewarding to take them early.

**Observations**
Watching the participants play the game made room for improvements to the game where most of them repositioned themselves while playing to make it easier to reach bugs. The main observation was that they only used their hands and arms to play, while sometimes it happened that some players used the silhouette of their head to smash the bugs. Should the camera be repositioned so that the whole body was rendered, as for now only the upper body was used, a change in method of playing would probably be seen. Only two players used the whole body while playing and seemed to have more fun as they completed the levels, but changed the way of playing to be more static as levels increased.

The feedback from the score on top of the bug(see 7) seemed to be working well as they noticed when the score was dropping. One player got the multiplier to drop to 0x but did not notice it on the side-bar but that the bugs only gave 0+ and then later noticed the water-tree.

One thing that was noticed was that the IceBug, figure 6, was often missed in comparison to the BossBug, figure 6, that was always hit. The IceBug seemed to be hard to see when it spawned and asking participants why, they said that it looked too similar to the drops and missed it because of this. So coloring of objects should be important so that they are not too difficult to see, a more blue-green ice-color should probably work better instead of a light one that is too similar looking to the water drop. Other observations of the IceBug was that it did not work as intended, when smashed the player would rest instead of taking the slowed-down bugs, but it did still work out well as when a higher level was reached a need for rest now and then was welcomed.

**Method**
The method used for testing the game could have been greatly improved. The group of people playing should have been more diversified spanning in different age-groups and genders. So choosing all male participants in the ages of 20-27 could mean that the data collected might be misleading. This is because for example in early tests where younger participants was observed showed a different way of playing as they like to move around more and jump to hit the bugs. The participants in the test figured out that they could be static in one position to become invisible and then shake their hands/arms as the bug arrived at the location of their static arms. This move could be a bit too advanced for a younger group of players meaning that there would be a different way of playing.

The participants also understood quickly that the waterdrops should be saved to increase the score and only one person did not understand what purpose the water had which meant that he got the 0x multiplier. Also here the children could have a hard time understanding the waterdrops as the observation made was to just jump and smash everything. Should another group have been used for the testing a more accurate reading on the feedback from the water-tree. In the test only one participant smashed enough drops for it to appear but seemed to understand its purpose although according to him he never looked at the multiplier he did noticed the tree. This could mean that when the score suddenly starts to drop to +0 a person would look around.

**Future work**
The first thing that comes to mind in ways to develop the game further is more levels. When discussing the level progression early in the development an idea of the tree growing further up into the air, where different kind of flyingbugs appeared. After the sky the tree would grow up into space where alien bugs would appear and so on. This would probably make the game more fun short term but as the game now is in a sort of kiosk mode where levels are infinite and making the bugs faster until you can’t catch them, it would demand an extensive amount of levels to reach the same playability.

Another thing discussed was to add achievements to the game so that a player would be more rewarded for playing the game. One idea was to collect trees when it had grown and by doing that a level could be infinite and the amount of water it had gotten would represent its quality as when collected. If this would be implemented a lot of rework would have to be done. But other achievements could be implemented for example smashing a certain amount of bugs in a certain amount of time or another example is to never hit a drop for a level.

Some implementations in the bugs was also discussed, that they would fly away in the direction which the webcam-rendering would hit them instead of smashing the bug. Special bugs which could only be hit in a certain direction and a Bug that when hit, changed path by jumping when hit, so the player would have to reach in other directions to hit it.

**ETHICAL IMPLICATIONS**
Setting up a game at a public location where people come in groups and play could have some impact regarding social ethics. One example of this is if someone who don’t want to play because of for example anxiety or general social fobia could become pressured to play even though they don’t want to. This problem could either be solved by using a confined space or playing together with someone who they are comfortable with. The same can be thought of when it comes to people with disabilities that cannot play the game at all or needs help.
Another implication could be the ethical aspect of using a webcam where the camera-feed is generally available by the computer. If someone were to access the computer that is used for the game without permission of the owner they could theoretically watch the camera-feed without knowledge of the player. Even though someone who do have permission to use the computer, can also theoretically watch the feed without the player’s knowledge which makes this a big ethical implication. Restricting access to such a computer would be very important considering this.

CONCLUSION
To conclude this study the following question, mentioned in the beginning will be answered with each category and its problems:

- What observations can be done and are there any problems when developing a game with webcam-based motion-detection in regards to factors like flow, feedback and positioning?

Flow
Looking at the result of the flow-analysis, the game can be considered to be a fully playable game and could be set up at a location for usage. The flow of the game was not considered to be perfect, meaning that more tweaking and implementations should be done to improve the playability and flow. Also more testing should be done by testing with groups of mixed age and gender to see what kind of different techniques they use when playing compared to the group chosen in this study.

Feedback
Looking at the observation done at the testing most feedback worked as intended. One of the problems here should be that coloring of the objects should not be similar to differentiate them from each other. Choosing a color for a Bug that is similar to the color of the Drop was considered to be a bad choice.

Another thing could be a problem is that there is too much feedback in the game. Some of the feedback might not be necessary to ensure understanding for the player as for example the multiplier that showed to be redundant, as one could look at the bug-score to see that the points are getting higher or lower. Also as stated earlier, more testing should be done to see how the different game-mechanics like the score and multiplier work with other gender and age-groups.

Positioning
Positioning in the game seems to be important for the playability of the game. When playing in front of the computer this is less of a problem as you can adjust the webcam yourself. But if playing in front of a setup TV-screen as in the test, where one can only adjust by repositioning themselves by going closer or further away this becomes a bit more problematic. A shorter person playing the game in front of a TV does not gain much by moving forward if the TV and webcam is positioned high. Moving towards the webcam to cheat was not seen as a problem as this caused the player to move and with the implementations to avoid cheating, the cheating was rendered useless unless a powerup was active. Further testing should be beneficial to see if the game would be more fun to play if the whole body was seen on the screen as not only the upper body.

REFERENCES
Flow Condition Questionnaire (FCQ)
Owen Schaffer

Please indicate how much of the time you knew each of the following while you were doing the activity by marking one circle for each question.

**How much of the time did you know…?**

<table>
<thead>
<tr>
<th></th>
<th>Never</th>
<th>About half of the time</th>
<th>Always</th>
</tr>
</thead>
<tbody>
<tr>
<td>what to do next</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>how to do what you were doing</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>how well you were doing</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>where to go next</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

Please answer the following questions about how you felt while you were doing the activity by marking one circle for each question.

<table>
<thead>
<tr>
<th></th>
<th>Not at all</th>
<th>Very much</th>
</tr>
</thead>
<tbody>
<tr>
<td>How challenging did this activity feel?</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>How much did you feel able to overcome the challenges you faced?</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>How distracted were you from what you were doing?</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

**Scoring**
Reverse the score of the last question to get Freedom from Distractions. The items above are in the following order: Clear What to Do, Clear How to Do it, Clear How Well Doing, Clear Where to Go, Challenge, Skill, and Freedom from Distractions.
In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Rasmus Cronstrand