The Indie Developer’s guide to immersive tweens and animation

What you need to know as a programmer to animate and increase immersion

Fredrik Öhrström

Handledare/Tutor, Dr. Erik Berglund
Examinator, Dr. Anders Fröberg
Upphovsrätt

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/.

Copyright

The publishers will keep this document online on the Internet – or its possible replacement – for a period of 25 years starting from the date of publication barring exceptional circumstances.

The online availability of the document implies permanent permission for anyone to read, to download, or to print out single copies for his/hers own use and to use it unchanged for non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional upon 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/.

© Fredrik Öhrström
ABSTRACT
Some games are grabbing your attention more than others. Some even do it so well people even lose track of time and their surroundings.

Why does this happen? And how can the effect be harnessed for your own game?

This report studies what immersion is and subjects related to it, such as richness and flow, and then how and what kind of easy animations and effects that build on these concepts that you can create in a 2D puzzle game. Most of the effects, animations and ideas can probably be carried over to other game types without much difficulty if you want a more immersive product.

In the end, the player experience is tested by two surveys to see if players were immersed, PANAS and IEQ. The results go over what kind of effects were implemented and the surveys showed that most players were a bit immersed and that they enjoyed the graphics of the game.

KEYWORDS
Immersion, flow, animation, tweens, game development, Phaser, indie development

INTRODUCTION
What makes a game immersive?

For any indie developer, creating a game is a huge task. Obviously, you need to have mastered tools in all elements surrounding a video game: programming, sound and visual effects. This work will go in depth in the latter. Even though you may be good at all these things, you are likely not an expert in all. Not every programmer has dabbled with animations before, and therefore creating an immersive and polished-looking product or game can be hard. Players do not get attached to games that are not immersive. They want something to look at that stimulates or entertains them all the time. If you cannot achieve this, then they will simply play other games that do.

What kind of small, simple animations can be implemented and added to make a game become more immersive instead of stale and boring, then? This work will go in-depth into tweens, and immersive animations. It should be “everything you need to know” as an aspiring indie developer to create your own animations, and what to think about when you do so that your players are immersed.

The game in which this work will try to answer this question is a HTML5-based math puzzle game for children ages 8-12, called ‘Entangled’, which is in severe need of interaction animations to keep children interested. This article will also explore other titles and game types and summarize what they have done right.

Purpose and objective
It is important for developers who might be more into programming to know that it is not hard to do these kind of ‘easy’ effects and animations using tweens. A menu can become much more alive if it has effects or animations that happen when you hover over options, for example. All you need is imagination and patience. And of course, your game and game mechanics may be a lot better if they contain the right animations, because they are literally what makes a game look and feel entertaining to play.

The objective here is to make it easy for a new indie developer to get into tweens and animations fast.

This work will also summarize some of what we know today about immersion – its different types, effects, and how to achieve them in order to make better and more entertaining video games. Furthermore, it will contain a study about if I, the author, has achieved immersion in “Entangled” with my animations or not, and if its animations are suitable for the target audience.

RESEARCH QUESTION
Formally, the research question is formulated like this to become wide enough to discover general guidelines and hints on what is learned here. In the authors eyes, “what worked well and what did not, and if it even works at all” when creating your own animations and micro details.

- What kind of micro-details and animations can be added to games to increase immersion?

Something that could be of potential interest for the reader is how much time the effects and animations took to implement. This is reflected upon in the results and discussion.

LIMITATIONS
“Entangled”, the base game for this study, is still a work in progress. Ideally this “polishing” part of the production would be carried out in the later stages, if there is time. Beauty is also in the eye of the beholder, and what is fun and immersive is, of course, very subjective, but this work will argue that a game that actually is a “good”, polished and immersive game will be perceived as a “good”, polished and immersive game by a large majority.
THEORY

Immersion

Immersion is something that is not easily defined, despite being a term that is used a lot [7]. But what can be agreed on is that it is a word for the sensation of being ‘drawn in’ or absorbed by something. You feel as if you are part of the game.

The feeling of immersion exists in several forms. It is important to know the difference between these. The most common one is simply called “immersion”, it has no other, better name, but it definitely should. Another immersion type is called flow. Flow is a bit different.

This first type of immersion, “immersion” in video games is basically defined by two steps by Wirth and his colleagues [2], summarized (very briefly):

First, players form a representation of the world in their mind as they play. After that players begin to favor their new media space (in this case, the game world) as their point of reference for “where they are” right now. When your character is chased by monsters, “you” feel the need to run away, or dodge the incoming attack, instead of running away because your character might die and it will reset your progress.

Obviously, achieving this type of immersion a puzzle game is a bit harder, if not impossible, but not in other, big game types, such as role playing games or platformers. This important to know because this type of immersion is different from flow.

Flow

Flow [5] is an important concept in interaction. Much research has been done on this phenomenon. Flow is a kind of immersion, but it is different from immersion that is described earlier. In summary, flow is the feeling or trance, the “balance” when something is not too hard (this generates anxiety) and too easy (this generates boredom).

Flow is immersive in the sense that you lose track of who you are, time and your ego and you become completely absorbed with what you are doing. This phenomenon is called a lot of things, but is most often referred to as “being in the zone”. In video games, flow is essential to keep a player interested in playing the game, or “hooking” them.

Flow is not just limited to video games; it can happen when you do just about anything. Programming is an obvious example.

Richness

In order for a player to become immersed in non-puzzle games, it seems like ‘richer’ game worlds are better [4] than simple ones. This is not really that surprising. A richer game world is a world that leaves less for the mind of the player to fill in the gaps. Ideally, you want to use as many senses as you can. To put it like Jamie Madigan, “Multiple channels of sensory information means simply that the more senses you assault and the more those senses work in tandem, the better. A bird flying overhead is good. Hearing it screech as it does so is better.” [5]

There are also other parts to consider when talking about richness. Interactivity with the world – if you interact with some sort of item or object then you expect some sort of realistic response, or at least some sort of visual or audial cue that something has happened with this object. It would not really make sense if you dropped a coffee mug on the ground and it did not break or make a sound. This is something that would ‘disconnect’ or break your immersion, because you have certain expectations of what should happen in the game world, but sometimes they don’t.

![Fig 1: Geralt (Main character of The Witcher 3: The Wild Hunt) looking down at a forest from a hill on his horse Roach.](image)

The image above in Figure 1 shows many effects that provide overall immersion from richness. A sky with clouds works much better than just a blue sky. Moving trees, the sound of the wind, and with proper lighting and fog throughout the canyon/river provide for a very convincing illusion.

Richness done right

“Amnesia – The Dark Descent”, by Frictional Games, a very famous 3D horror game, is a phenomenal example of this ‘richness’ described earlier. You can interact with almost any object – boxes, doors, handles, valves… you can even light candles and torches as to get rid of the dark. Many of the puzzles in the game force you to interact with objects. Some puzzles are physics based, for example stack 3 boxes and then jump on them to reach a ladder, or throw enough rocks on a bridge hanging from a broken pulley system so that the weight of them pulls it down. Meanwhile you are constantly fed with ‘strange’ things happening as you run through the haunted medieval castle of the game world; eerie sounds, doors closing, noises, strange flashbacks, very unsettling environments and so on. It is all very unnatural, but it is a horror game, so it is not out of context and therefore does not break the illusion.
Good animation
Another key concept of immersion is also obvious; good animation. It needs to look good and be convincing. Getting back to the bird example suggested before by Jamie Madigan, if these birds would appear unrealistic, for example the animation of flapping its wings is unrealistic, or better yet, its missing so the bird is not animated at all, this would only break your immersion.

Immersion breaking and disconnection

The figure above shows an example of an absurd scenario that would probably disconnect your immersion from the game world. It is not easy to transfer yourself and your consciousness into Geralt’s point of reference here because he is inside a pillar.

When reading this paper, it is difficult to show you an example of a poor animation through this medium. With Figure 2 the author wants to illustrate that you need to be very careful when not implementing something well, because the results can be devastating for immersion. Especially with animated objects, because these tend to catch your attention a lot more. If something does not appear to ‘fit’ the theme of the world, players will notice it very fast, and at that point two things can happen. Either the situation is ‘forgiven’ by the player because the scenario provides some sort of comic relief or the player just did not care, or the absolute opposite happens, the player becomes annoyed and with confirmation bias only sees things that turn out bad.

It is also very important to distinguish this extreme example above, which is actually a glitch – something in the game, with something that is just a “badly implemented” feature of the game. There is a big distinction between “broken” and just “done badly”. The example above is obviously broken, but illustrates the point of immersion breaking in the case that the situation is not realistic and will throw you out of the virtual world perspective change.

Uncanny valley
This is an interesting phenomenon in regards to immersion and human-like appearance. The uncanny valley happens when a robot’s human appearance or movement is almost perfect, but not completely perfect[9]. This could potentially happen with non-player characters, or, as it is commonly abbreviated in the video game world, NPCs. This creates an eerie perceived effect in the observer, the feeling of ‘something is just not right’ and promotes disgust, because our brains are really good at determining what is human and what is not. The scope of this work is a pixel art 2D game, so it is extremely unlikely to happen, but might be a future problem in virtual-reality, or VR, environments in regards to immersion.

Interface
The user interface, or UI, is important for immersion[6]. Although this work will not focus on it much, an annoying UI can cause large amounts of frustration. They are a necessity to be able to interact with the game. Simply put, a UI that is in the way, or hard to understand or navigate, is not good for immersion.

Sometimes next to no UI/HUD is the best UI for immersion, as is the case with Amnesia: The Dark Descent. The game’s only UI when moving around (and not looking at your inventory) is changing your crosshair/dot to a hand if you are in range and looking at an interactive object. This provides a very atmospheric and immersive experience.

Character movement
In order to immerse yourself, according to Wirth[3], and as described in the “More about immersion chapter”, you need to transfer your point of reference to where you are. This is a lot easier if the character movement is working with you, and not against you. When creating video games, good character movement is essential for this. There is nothing more frustrating than controls that do not do what you want them to do. Sometimes this causes you to die in the game, and revert your progress. This disturbs both flow and immersion, and is really bad. The span of this work does not cover this, but it is important to know. Character movement is also something that requires much ‘polishing’.

Performance and fps
Frames per second, usually referred to as FPS, is the amount of frames (pictures) the device where the game is played on can send to the monitor each second. A device in this context is usually a computer, which can be a major video game console, like today’s Playstation 4 or Xbox One, but can also be a handheld gaming device, like a Nintendo 3DS or a phone, et cetera.

Most of the time when talking about performance, people measure it in FPS. When talking about FPS, there are two bottlenecks to consider. Either your device is “too slow” to calculate what is happening in the video game world, and cannot deliver frames at a good enough pace to produce a believable illusion.

The second bottleneck can also be your monitor. Your device might be capable of producing frames at a very fast pace but if your monitor cannot display them at that pace it does not matter.
**Performance - sweet spot**
The video game “sweet spot” for FPS is therefore disputed. With too many features or a poorly optimized game, only the better, more powerful systems can run them at an acceptable FPS. With too little content, any system can run them, but you run the risk of low richness, and you are not using the hardware of the device at its full potential.

**Performance goals**

<table>
<thead>
<tr>
<th>Device</th>
<th>“Common” Target framerate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phone or handheld</td>
<td>30+</td>
</tr>
<tr>
<td>Gaming console</td>
<td>30-60</td>
</tr>
<tr>
<td>Mid-tier PC</td>
<td>30-60+, 60+ preferred</td>
</tr>
<tr>
<td>High-Tier PC</td>
<td>60-144</td>
</tr>
</tbody>
</table>

Table 1: Framerates to try to achieve when developing a game.

Most modern monitors today (2016) have refresh rates of 60 Hertz – capable of displaying 60 pictures per second, so that is what should be the best “target”, there is not as much value for most devices to go above. However, on handheld devices, even though most monitors are capable of 60 Hz, there are other limitations, other than just “poor” hardware, such as battery life.

Some high-grade devices, like a powerful desktop PC, can deliver FPS in the hundreds no matter what game is played. A powerful system combined with a monitor that can deliver frames, at for example, 144 Hz, the difference is very noticeable and a very different experience from 60 FPS, much as how 60 FPS is a different experience from 30 FPS.

**FPS drops**

A consistent framerate is better than a wildly varying one. FPS drops, where you have a high framerate but something causes the framerate to fluctuate a lot, is worse than having a consistent ‘lower’ framerate, even if the average frame count is higher with the fluctuations.

**Easing functions and tween animation**

When creating animations in most game engines, you usually use a technique called ‘tweens’, which is a shortened word for ‘betweens’. A tween is a very simple concept; you start with something, maybe a sprite scale or a set of coordinates, and you will end with something, for example a new sprite scale or new coordinates. Then you use something called an *easing function* and specify a duration for how long it will take to run the easing function that controls everything that happens ‘in between’. Please observe Fig. 3, 4 and 5.

In more technical terms, an easing function is used to interpolate one or a set of values to another value or set of values over a period of time.

There are lots of easing functions that are commonly used, such as *linear, exponential, cubic, quintic* and so on. You would obviously have to look at whatever game engine you are using to see which are available, but these can of course be implemented by yourself too if you want.

Another good part of easing functions is that they usually come with their reverse function, where the function is applied at the end of the ease instead of at the start, and often bundled with a ‘both’, or ‘InOut’ mode which applies it at both the end and the start.

A good place to get familiar with these easing functions and interact with them is on the GameMechanicExplorer[^10] website. Everyone who wants to get started with tween animations should definitely visit this page, but here are a few recreated examples of how they look and function.

---

[^10]: [GameMechanicExplorer](#)
It is good to have an idea of how all easing functions work for your game engine, and then know that they can all be reversed for the opposite effect, or their “InOut” mode.

**Tween focused games**

There are some games that are animated almost completely with tweens, such as card games. An immensely popular card game called *Hearthstone*, released by Blizzard Entertainment, uses tween-like animations for a big portion of their game, most specifically the board state where you move cards and characters around.

**Other usage areas for easing functions and tweens**

Tweens are common everywhere, not just limited to game engines. For example, jQueryUI has integrated easing function/tween support to animate things on web pages and is used widely on the web.

**SURVEY THEORY**

**Positive and negative affect schedule (PANAS)**

PANAS, the The Positive and Negative Affect Schedule\(^7\) is a useful way to measure a positive or negative experience. PANAS measures about 20 different emotions where you rate them from “Very little” to “Very much”. This is an important survey because it is essential to know if the animations are affecting players negatively or not, which PANAS will show because irritation, anger or frustration will be apparent. If they are upset or affected negatively, we can assume that they are detrimental to the immersion effect and disrupt flow, which is not good.

**Immersion experience questionnaire (IEQ)**

The IEQ survey\(^8\) evaluates immersion in games. With this survey it is possible to see if players were immersed in the game or achieved flow, which is the goal. If not, we might get hints as to what it was that disrupted it.

**METHOD**

In this chapter, the method for answering the research question will be described: how, and why, and which animations or interactions were implemented and how they affected the game. The animations and interactions were evaluated by letting players test the game and feedback was collected in surveys (PANAS\(^7\) and IEQ\(^8\)) to see if they improved the game or not. The aim was to see if there would be a clear correlation between what people enjoy and how much time that took to implement, to learn what is ‘time well spent in polishing’ which is what this paper is about.

**Phaser as engine**

Since Phaser is a JavaScript based engine, it can run on almost all common systems\(^9\). Special care is taken to not make too rich effects, or implemented in a poor way; this is very important because this can impact the performance of the game. Heavy focus will be on implementing the animations, interactions and other things in the “correct” way in regards to programming.

**Study the game**

The game must also be thoroughly understood, because it is still a work in progress. Ideally all content created by me, the author of this paper, has a proper place in the game and should be good enough to be kept in the final implementation. All animations and interactions have to be motivated enough to fit the theme of the game.

**Add effects and animations**

Something that can do a lot for immersion is simple particle effects. Phaser\(^1\) has very good particle effect support, so effects should absolutely be implemented and iteratively improved. It is important these are subtle and not in the way.

Special care was here implantation-wise because particle effects can affect performance if done wrong and cause FPS drops - many objects are created simultaneously, often in a single frame. Duration and difficulty to implement is also important to quantify. As implementation techniques will be learned, the authors animation skill will be increased, and some effects can be reused or modified, this will of course be factored into the results and be thoroughly documented with some sort of normalization.

**Surveys**

Two surveys were conducted and evaluated after letting a few testers play the game a bit. The age group of the testers were ages 20-25 and we had 9 test subjects. The reason for the relative old age of the subjects being that children require special permissions for the surveys to be legal and there simply was not enough time to organize.

**RESULTS**

**What was implemented**

Entangled required a lot of polishing and animations that made the user interactions more lively and interactive. Since the game is meant for young children, animations had to be adapted to be simple and over-exaggerated in their features, “to look childish” or at least appeal to them. A lot of game assets (pictures and paintings) were provided by the customer of Entangled, because the game is accompanied by a book with hand-drawn pictures that are also digitalized.

Therefore, interface animations were implemented that improve the flow and “obviousness” of what is happening.
Some of the implemented animations use particle effects, but most of the implemented animations are tweens changing the alpha channel (transparency), scale and position of objects. With just these, along with the right easing function, you can animate almost any asset and make it look good.

For the work behind this project, the following kinds of animations below were focused on, because I thought they would be the most useful. ‘How well it worked’ is followed up later with the survey and observation evaluation, and of course discussed in the Discussion-section.

- **Interface animations** that improve the understanding of what you are currently doing, such as selecting items or sliding in submenus or interface screens.
  
  *Reason:* It is reasonable to assume it is necessary for it to be ‘obvious’ what you are doing at all times, because after all, the game is meant for children. Attention span and concentration is limited. It is also important to get feedback that you interacted with something in the game.

- **Improved the board with a more “realistic” card flipping animation.** An animation for the gameboard where the number is actually only visible once the coin is turned.
  
  *Reason:* Together with a colleague I revisited and reworked how the game board was animated. It required refactoring and in turn we could improve the animation.

- **Transition animations**, and items or objects following through different scenes
  
  *Reason:* It probably is less confusing for the player if objects do not just appear out of anywhere but follow from the other stages or scenes in the game. Useful when achieving unlocks. In Entangled it is used for unlocking the picture pieces.

- **Removing “instant-appearing” items** on the screen
  
  *Reason:* Sometimes you can direct your attention to things by timing where and when they appear, and in sequence. This has been useful when changing scenes, or when making objects appear.

- **More varied look on objects on the gameboard**
  
  *Reason:* It was a lot easier to play the game when the cards had distinct shapes when you had to flip a lot of them.

- **Progression for flow and motivation with a “code” and unlocking parts of images**
  
  *Reason:* It would increase flow to progress towards a goal, which the game really needed – before this it did not really have any objective.

---

**Animation/effect implementation time**

This part of the research question is easy to answer. In truth, none of these effects took very long to implement code-wise, once you had an idea of what you wanted to make. If they did, it was because existing code made what you wanted to do difficult and it was not possible to implement it without refactoring.

My best tip is to start with something simple and go from there. Such as moving something and changing the alpha,
instead of doing what I did which was start with the most
difficult animations first that used a lot of effects and chain
animations in tandem.

The two surveys – PANAS and IEQ
To see what players thought of Entangled in its current state,
two surveys were conducted, PANAS and IEQ, after letting
the players test the game. The age group of the testers were
ages 20-25 and we had 9 test subjects under observation.

Survey results
The PANAS evaluation showed that most people had a small
or greater positive experience playing the game.
The IEQ showed that people did indeed lose track of time
and were immersed, and also that they slightly enjoyed the
task they were given. Our IEQ score (which has a scale of -17
to 169) the result was 83.6. When normalized to 0-100 the
result was 45.

When asked if they wanted to play the game again, most
people said no.

Below are some sample questions and interpretations from
the IEQ, which is the most important survey for immersion.

To what extent did you lose track of time?

Fig. 7: The results above tell us that most people lost track
of time, which is a sign of immersion. The subject who
provided a result was stuck in the tutorial for almost 30
minutes – this was described later as a very frustrating and
horrible experience.

To what extent did you feel consciously aware
of being in the real world whilst playing?

Fig. 8: The results above tell us that although people lost
track of time, a lot of subjects were still very aware of the
real world around them. But some were not. This tells us that
there is a tie towards a little bit of absorbing immersion, but
not extremely strong.

Fig. 9: The results above show that even with a poor start in
regards to immersion, all subjects felt they improved by the
end of the test. This is a sign of flow, and could be why most
people were happy playing the game.

To what extent did you enjoy the graphics and the imagery?

Fig. 10: The subjects answered how much they felt absorbed.
The median value is 8 and average is a 6. Note that the scale
is 1-10 here instead of 1-7 like the other charts.

Fig. 11: The above results shows that most subjects enjoyed
the graphics and imagery of the game. This would have
looked very different if the added animations and provided/modified assets were made poorly.

Personal observations during the survey
People had a hard time to understand the mechanics of the game – the tutorial currently in place is not good enough. It is not really the fault of the underlying work behind this thesis, but this did prevent flow at the start because most subjects were frustrated. It was hard for players to let go of the frustration that they had worked up during the tutorial for the rest of the testing period.

Once they did grasp the mechanic the test subjects were quite absorbed with what they were doing, and it seemed like flow was present – judging from both observations and survey results.

After they were done they were asked to fill out the surveys and provide feedback on what they thought. Most of them said the tutorial “was garbage”. Some of them were asked specifically if they thought the animations contributed to the game, and most agreed. This is consistent with Figure 11.

DISCUSSION
What was hard
There was some challenge in working with the existing graphical assets, especially hand-painted pictures. If you wanted a specific object you were forced to crop it out. It is also difficult to modify hand-painted pictures of this kind (digitally hand painted). Unless you are an artist good at digital painting it is very hard to add or remove shadows and fit the aesthetic. Obviously, this will not be covered by this work, this is a skill you have to learn somewhere else. It is, however, important for the discussion.

What worked well
Changing the position of objects with a tween instead of just ‘teleporting it’ to the right place really does a lot for the overall impression and polished feeling, and in this case, hopefully stimulates the children playing the game. With Phaser as the game engine, this was really easy.

Instead of

```
some_sprite.x = 400;
// Move this sprite 200 px left
some_sprite.x = 200;
```

You would just create a tween like this,

```
some_sprite.x = 400;
// Move this sprite 200 px left with a 700ms duration
```

You do not need to add much code, and the code itself is very self-explanatory. It takes about 30 seconds to implement, or two minutes if you really want to think about what easing function to use. This worked really well implementation-wise, obviously. Creating lots of these animations, or hover effects and similar effects (by changing scale) is very easy. However hover effects are not particularly useful in a mobile game like Entangled, because you do not control a cursor and cannot really hover over most mobile phones touchscreen. Instead focus was on animations where you select a coin.

Implementation only starts to get messy when you use tween chains in Phaser, but even then, not much changes. In Phaser you can chain multiple .to() – function calls that are called sequentially after the tween before executes its onComplete()-function call.

Creating a fade-in effect is almost as easy,

```
some_sprite.alpha = 0;
// Make the sprite fade in with a 700ms duration
```

When implementation is this simple, it is important to think about not overdoing it. Use animations where you want attention to be directed. And of course, your animations need to “fit” their purpose or they will cause a negative effect on the immersion level of the player. Irritating animations will also reduce flow.

---

Fig. 12: Asset example: a hand-painted picture illustrating a magical/advanced technological box in the accompanying children’s book where the original graphical assets come from. This was used (after modifications) in Entangled for an animated random number generator.
Hiding loading screens
Another set of animations that worked really well were masking the loading of items and scene swapping, because they serve two purposes; it allows loading and setup in the background, and serve as an obvious hint to the user that “now you did something important because the scene changed” and that explains why everything looks completely different when the animation completes.

What did not work so well performance-wise
The sprites and assets were very large images. If you are working with very large images you should resize them beforehand – do not do it in Phaser, but this concept should apply in all engines.

Not only does it cause longer loading times but it increases the size of your application and the amount of computation that is required to use them.

The first animated version of the random generator described beforehand used very large image assets, scaled them down in-engine, animated them with tweens and created a lot of smoke particles which were also very large, downscaled in-engine sprites. In addition to that the smoke particles changed their transparency depending on the duration they have lived.

This caused the framerate to dip below target – especially on mobile, but was easily remedied by resizing all the image assets beforehand.

It is also better to do it beforehand because that way you have more control of the downscaling and can preserve more detail in the sprites. You can achieve a much better result by using better but more computationally heavy algorithms, such as Bicubic and Bicubic Sharper. The normal Phaser downscaling does not look as pretty because it is done in-engine and in real-time.

Conclusion
Judging from the survey results, we can conclude that the test subjects felt somewhat immersed and seemed to enjoy the game at least a little. Since the game looks almost completely different now than it did when development was started, it might be hard to tell if it is because of the animation or because big gameplay improvements. In any case, the
majority of the subjects proved that in Figure 11 the animations and the assets in the game look good.

A lot of subjects gave feedback during and after the test what could be improved with the tutorial.

LIMITATIONS

Survey discussion
The surveys could have been done better, using children in the target age group, and more people. Feedback from the target age group is a lot better than from people who are a little older.

Whether the difficulty of the game is too easy for "grownups" compared to children is not evaluated and could possibly skew the result for a couple of reasons, namely the kind of simple addition present in the game is not hard for adults compared to children. This affects the rate of which players gain flow – maybe the survey results came out this way because the difficulty was 'just right for flow for adults' and not because 'players could achieve flow because the environment allowed them to'.

The tutorial also took more than 30 minutes for one subject to understand, and at least 5 minutes to understand for most subjects. This individual was very frustrated and had a very negative experience according to the surveys.

Additionally, almost every test subject mentioned that it was hard to understand what to do after the tutorial. With this in mind, there is serious concern that children will definitely not understand the game.

In the survey, most subjects did not want to play the game again. Because of the age group and because most were students it is not necessarily because they did not enjoy the game enough but could also be because they had to spend their time doing other things.

Both surveys (PANAS and IEQ) were provided in Swedish. However, they were translated and provided by previous students who did similar work and not translated by an official source. Some of the translated questions, especially in PANAS, were a bit strange and confused the test subjects.

FUTURE WORK
Players have achieved at least a little bit immersion, and enjoyed the graphics (Fig 11) but that is no reason to stop adding good, high-quality animations. In this project there were tons of animations that could still be implemented.

Due to time constraints and multiple restructures of the menu flow on the project a lot of animations had to be discarded.

This work only focused on animation, which only stimulates the vision sense, sight. The game has no audio at all. It would be interesting to see how much immersion would increase in tandem with sound – I believe this would increase the richness by quite a lot, but I have no idea how much. This is also really important for the absorption measured in Figure 8 and 9. I am positive would be easy to launch the same survey again after implementing sound effects for the game.

REFERENCES