Game Programming Lab 2016

Dance with the Cooks

Goran Saric, Christian Schmidhalter, Stefan Blumer, Pius von Däniken

Conclusion Chapter:

Changes:

During the last weeks, we completed our rewrite. Although it was last minute, we don't regret our decision at all. The new system we implemented allowed us to overcome the timing issues we previously had. The most important change was to create a system that triggers events at a certain time in the future. With this, we implemented the game logic update pipeline as well as all the visual effects. Thanks to these changes we were able to improve the readability of our code, could implement additional game elements like the cakes and tomatoes as well as add lots of animations like blinking, moving, particles, fading and others.

Hit in any direction:

After removing the need to rotate your character before being able to move in that direction, we now added the possibility to strike in any direction by pressing a direction button and the strike button at the same time. This change makes the game even more fast paced.

Tomatoes:

We changed the behaviour of tomatoes so that they don't land on random spots on the map but land in predefined patterns. Patterns differ in how many fields they affect and are categorised as easy, medium and hard accordingly. As a match progresses the spawned patterns get harder.
The patterns address two issues that were raised previously. The version with random tomatoes was considered too confusing and there was a wish for some kind of area denying game object, forcing players out of certain regions.
We also tried to make the indication of a tomato landing clearer visually. The field where a tomato lands is first indicated in a light red two beats before it lands. In the beat before the tomato lands the field gets increasingly darker until the tomato finally hits.
A player hit by a tomato will get knocked to a neighbouring field and drops one star.

Cakes:

As an additional game object we added cakes. Cakes always spawn on one of five predefined fields. After a few beats it explodes. The explosion radius is indicated in red two beats before the cake goes off. A player hit by the explosion will be knocked back two fields and loses two stars. To avoid cakes a player has to either move out of the explosion radius or they can hit the cake which will knock it back two fields. This opens additional strategic options as they can knock the cake towards opponents.

Falling off the stage:

A major change that affected the game a lot was adding the ability to fall off the stage. Falling off the stage results in the player dropping two stars to the ground and respawning at a random location on the stage. This mechanic becomes very interesting together with the knockback effect of hits of players, tomatoes and cake explosions, as they can knock you off the stage and make you, in combination with the other elements, lose a lot of stars in a short time.

Match Flow:

As mentioned before the tomato patterns are grouped into easy, medium and hard. We use this to implement a very simple match flow that progressively gets harder. We increase difficulties of cake events by spawning more cakes at once. We never spawn cakes and tomatoes at the same time as this is very confusing and sometimes makes it borderline impossible to avoid getting hit. Finally we increase the speed of the music very slightly over the match.

Conclusions:

Our final changes improved the game's visuals, as well as the game mechanics. Although the graphics and the game mechanics feel complete in the current state, we found some important usability problems during the game session during the apéro after the final presentation. The main issues are:

1. The menu was very misleading for many players. This lead to many situations, where people wanted to join a game but didn't know what buttons to push or which character they were

controlling. We will have to rework to menu and add some kind of gating mechanic where players have to execute some movement to show that they grasped the concept.
2. Many players struggle with finding the beat, either not pressing periodically at all or pressing on the off beat. This results in a lot of player frustration at least in the first few matches. This fact of course conflicts with our plan to create a party game, everybody can play. We will try to address this problem by adding an easy mode to the game, where input is never rejected. Also adding a calibration menu, to adjust audio and video delays would probably help.

Although our game has very simple mechanics, we found it to have an unexpected high level of depth, making it much very skill based.

Comments on the Course:

Overall we are very happy with our end product. The time invested in the final weeks to add additional game objects definitely payed off.
We think the fact that we decided on a fairly simplistic game play from the start helped a lot during the whole process since we could focus on improving and iterating on them. The major changes from initial concept are the improved movement rules and the concrete implementation of additional game objects. We found that the movement rules we decided on during the prototyping phase were not adequate for the dynamic gameplay we wanted to achieve. Player feedback was very helpful to figure this out.

Implementing our own procedural music showed to be a hindrance since it tied down a lot of resources that could not be invested to improve on the game. Additionally from a player's perspective there is almost no difference between pre composed and procedural music, apart from the fact that procedural music can be perceived as qualitatively inferior. The effect of having adaptive and responsive sound effects could probably also have been achieved by having annotated pre composed sound files.

The theme first felt very restricting. But we soon came up with some cool ideas, although most of them would also have worked with another theme.

Handling the timing issues from the beginning would have saved us a lot of time. This is a mistake we definitely won't make again.

We had a very good division of work within the team which help a lot with the schedule. It was hard to work within the schedule we had to hand in because at the time of creating the schedule we didn't have a very concrete idea of how to break down different tasks. Additionally some of the requirements changed over time, for example we had to scale back our ambitions for the procedural music. In the end we followed the schedule more as a guideline and could probably have done without one.

Having to work with Monogame was probably the worst part about the whole course. While XNA was a great framework, with great documentation and few bugs, Monogame turned out to be a very unstable and incomplete remake of XNA. We also think, that having to reinvent the wheel again and having to write your own engine takes a lot of time, that could otherwise be used to improve the game mechanics and visuals. Writing your own engine is a fun project, but does take the focus from creating a fun game.