## Summary of project progress

### Schedule compliance

Overall the team followed the schedule that was planned out at the beginning of each new delivery, we always made sure to check the time chart before starting a new task.

### Resources spent

### Production metrics

Keep on rocking in the free world! We will evaluate our production metrics based on the following criteria:

* Time spent versus time planned.
* Number of implemented features versus number of planned features.

We had planned on spending 341,5 hours on L2 and L3 but we only spent 254,6 hours, but we still managed to succeed in every endeavour that we have undertaken. Which means we had a lot of time where we could play out in the sun, which improved group morale and health, without feeling guilty because of the lack of time spent. In total we spent approximately 75 % of the planned time which means we are at least 25 % better than expected.

There were a total of 17 mentioned features in the description, where 5 of these are so called extras, out of these we have implemented 9 features. This may sound as a low number but the code is written in a way that implementing the rest of the features will be quite possible. We have implemented the following features:

* A 3D labyrinth
* Freely moving ghosts.
* Candy!
* Special candy that allows you to hunt ghosts.
* Two audio sources that plays background sound.
* An audio source for candy.
* Audio sources carried around by the ghosts.
* Pacman has three lives, and he loses his life in a quite spectacular manner.
* We can change the view port to see the whole world, i.e. the map.

## Programmeringsproblem

Referencedriver

At first we ran the project with the reference driver for instead of on the hardware in the computer which gave us severe performance issues. This was solved with the aid of a teacher.

Billboarding

We could not manage to get the code example on the billboarding technique to work in our code even though it worked flawlessly in the example. This was solved with some good ol' fashioned beating of the code.

Camera

There were a lot of small implementation faults in our original camera. In the end the easiest solution was to reuse a different camera from a previous project with the aid class for taking raw mouse input.

## OOD-problem

### Singelton implementation in observer patten:

At first we wanted to have our "screens" as singletons in our observer pattern. This, however, did not work since we needed pointers to every screen. In the end we came to the conclusion that having singletons did not contribute in a positive way to our. Thus we discarded the entire idea of having them as singletons.

## Resultat i Kod

### Camera

The first person camera was finished and is working as intended with no current bugs.

### Rendering components and Screen handler

The renderer for setting up a directX device and viewport is finished and can send these down to all parts that need them.

### Sound

We implemented 3d-sound from all objects as well as the two corners of the world that were specified in the assignment.

### Collision with objects

Collision with all game objects is working as intended without known bugs. The wall collision is not yet working because of the lack of a implemented quad-tree.

### Enemy AI

The enemy AI is done and uses a waypoint system described by a state machine diagram further down in this document. It can be used to implement an limitless number of waypoints and movement patterns for the ghosts.

### Resource Handler

The resources handler is finished and can be used to load in the different textures used in the game.

### HUD

The games HUD was finished and displays the number of lives and the player score correctly.

### User Inputs

The user input class is finished and can handle input from the keyboard which then moves Pacman around the level.

### Score system

Score is given from eating candy and ghosts. The result is saved correctly and can be used in the different parts of the game.

### Simple world

A simple hard coded world has been finished and can be rendered properly without fault.

## Resultat lärdomar

What we have learned is this: OOD is good at the start of a project but will never hold all the way through. You have to be willing to make compromises along the way when writing code, otherwise project will become too time consuming or not cost efficient, in worst case it will end up not working at all.

We also learned that estimating work time is really REALLY hard.

And that anecdotes from the world of software development make for really good seminars.