**Project Plan**

DV1478 Kandidatarbete i datavetenskap

2017-04-03

|  |  |  |
| --- | --- | --- |
| Thesis | Tentative title | Procedural city generation viable in games |
| Classification | Procedural content generation, Games, Rendering, |
| Student 1 | Name | Elias Frank |
| Email | Elias.frank@gmail.com |
| Social security number | 19940327–4179 |
| Student 2 | Name | Niclas Olsson |
| Email | ohlsson.niclas@gmail.com |
| Social security number | 19930421- 3797 |
| Supervisor | Name | NA |
| Email | NA |

**1: Introduction**

Exploring a huge open world environment is a desirable feature in a game. But creating a big open city such as in the *Grand Theft Auto* series, *Assassins Creed* and *Batman: Arkham City* involves years of work for a lot of people. Making big open cities in games is simply not feasible for most game companies. These games all have massive success with their big open worlds making a feasible generated city an attractive technique for smaller companies to be able to compete with big open worlds of their own.

To create massive amounts of content without a big workforce there exists algorithmically based solutions, *Procedural Content Generation* (PCG). PCG was in the past used as a way to minimize the disk space required for games, *.kkrieger* is an excellent example of a game using PCG to minimize disk space. But has evolved in to a method to minimize workforce required for content, *No man’s sky* is an excellent example of a game using PCG to minimize workforce while maximizing content.

In this work we want to explore the possibility of procedurally generate a city feasible to use within a gaming environment. This include many requirements as a game is both a real-time application as well as a interactive medium for enjoyment. Geometrical complexity must be kept low for it to be rendered in real time. The use of graphical memory (VRAM), Random Access Memory (RAM) and disk space must also be within reasonable limits expected of real time applications. The city does not need to be believable but it need to have variety for the players to not lose interest, this is a requirement for the city to be viable in games. Loading times must also be constrained. Specified constraints will be defined in the thesis after more research has been done for realistic and viable constraints for games.

There are many techniques to achieve PCG such as ray marching, squarified treemaps, perlin noise, fractals and many others. Part of this work will be to research what all these techniques are good for and which to use for what purposes.

**2: Aim and objectives**

* Research and explore what techniques to use when procedurally generating a city.
* Find a way to generate a city viable to use in games.
* Explore what constraints a city must have to be viable in games.
* Discover exactly what steps need to be done to generate a city and generating different cities depending on some set of parameters.

**3: Research question**

* Can PCG techniques be combined in a hierarchical manner to procedurally generate a city that is viable in games according to set constraints a real-time application such as games have?

**4: Method**

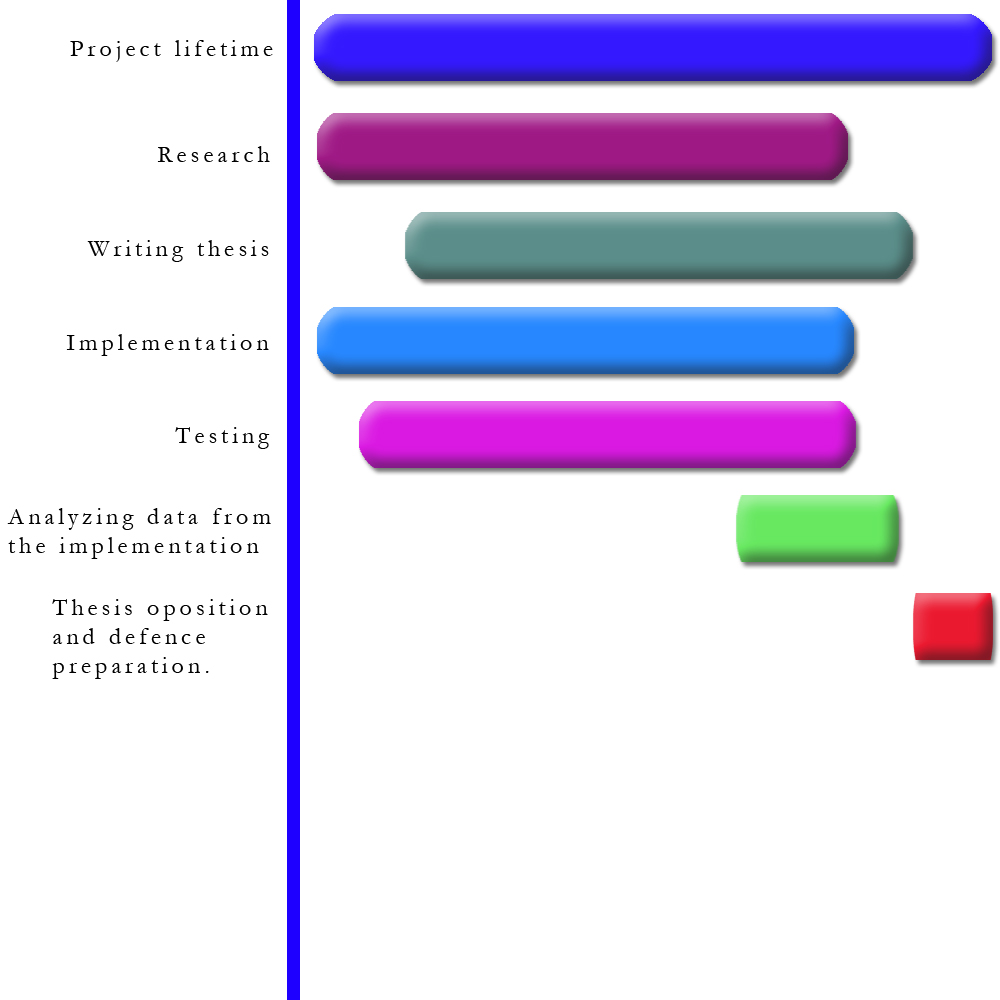
The method used to answer these questions will be studying and researching of PCG techniques along with an implementation of said techniques to generate a city viable in games. This implementation will be able to generate cities according to a set of variables, the performance of the implementation will be measured with different input resulting in different cities we are able to measure according to the constraints. The result of these tests of the implementation will be analyzed and will hopefully answer our research questions.

A minimum of ten different tests will carried out and analyzed. These tests will cover a vast number of circumstances for different cities allowing for a good analysis of pros, cons and if the city is viable to use in games.

**5: Expected outcomes**

From the implementation, we will gather a good amount of data making it possible to analyze and answer the research question. Furthermore, the research and experience gained from making this implementation will be valuable to continue our path of great game programmers.

**6: Time and activity plan**

****

The implementation and research will be the first things to be done and they will be done simultaneously. Some parts of the implementation require research before we have the knowledge to implement it, but all the parts of the implementation such as the OpenGL pipeline that do not require any research will be done in conjunction with the research.

The writing of the thesis will begin shortly after we have some basic research and base implementation done. The thesis will be worked on every week until completion. When the implementation has enough features, gathering of data and analysis will begin. This data is crucial concluding the thesis and answering the research question.

**7: Risk management**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability (1-5)** | **Severity (1-5)** | **Mitigation** |
| Procedurally generating a city is difficult and consumes too much time. | 3 | 5 | Only implement the most crucial features to answer the research question. Ask supervisor for guidance. |
| Failing to communicate with thesis partner. | 2 | 4 | Have daily meetings and work together when possible. |
| Technical issues | 2 | 5 | Make sure all work is on several hard drives. Use git as source control. |
| Collect data from the implementation wrong, twisting the results. | 3 | 4 | Carefully decide what parameters in the implementation to collect data from and in what form to collect this data. |
| Not enough contact with supervisor. | 2 | 3 | Contact supervisor at regular time intervals. |
| Improper analysis of the data. | 2 | 5 | Do not tweak the results to get the expected outcome. |
| Health problems for students or supervisor. | 1 | 4 | Work to the best of our ability anyway. |
| Lack of time or confusion about time management. | 3 | 4 | Follow the project plan. Contact the supervisor. Look for work that can be mitigated. |

# Referenser

Carlos A. Vanegas, I. G.-D. (2012). *Inverse design of urban procedural models.* New York: ACM Transactions on Graphics.

Fernando Marson, S. M. (2010). *Automatic Real-Time Generation of Floor Plans Based on Squarified Treemaps Algorithm.* Brazil: International Journal of Computer Games Technology.

Markus Steinberger, M. K. (2014). *On-the-fly generation and rendering of infinite cities on the GPU.* Computer Graphics forum.

Muller, P. (2006). *Procedural Modeling of Cities Part VI.* Boston, Massachusetts: ACM SIGGRAPH.

Müller, P. (2001). *Procedural modeling of cities.* New York: SIGGRAPH '01.

Stefan Greuter, J. p. (2003). *Real-time procedural generation of `pseudo infinite' cities.* Australia: Graphite.