**Project Plan**

DV1478 Kandidatarbete i datavetenskap

2017-04-30

|  |  |  |
| --- | --- | --- |
| Thesis | Tentative title | Procedural city generation viable in games |
| Classification | Theory of computation, Randomness, geometry and discrete structures, Computational geometry |
| 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 | Hans Tap |
| Email | hans.tap@bth.se |

**1 Introduction**

Exploring a huge open world environment is a desirable feature in games. But creating a big open city such as in the *Grand Theft Auto*[1] series and *Batman: Arkham City*[2] 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 algorithmically based solutions exist, *Procedural Content Generation* (PCG). PCG was in the past used to minimize the disk space required for games. *.kkrieger*[3] is an excellent example of this. It has since evolved into a method to minimize workforce required for content. *No man’s sky*[4] 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 generating a city viable to use in games.

MEASURE VIABILITY WITH HUMAN TEST SUBJECTS – viktigaste mätningen

## 1.1 Techniques

There are many techniques to achieve PCG such as ray marching, squarified treemaps[5], Perlin noise [12], fractals, L-systems [9][10], Shape grammar[20] and rule based subdivision[10] among others.

## 1.2 Hierarchical PCG

First we generate the districts with a Perlin noise. Later the roads and blocks are generated with a Perlin noise, but taking the districts and its parameters into consideration. Like this the houses are generated and using information from the roads, blocks and districts for its generation. This is what we refer to as Hierarchical PCG. Säg att det är top-down.

**2 Aim and objectives**

* Collect relevant data for analysis to answer the research question.
* Bra test med människor som svarar på enkäter, visa att hela staden kan genereras med en metod, nosie?

**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? ÄNDRA MIG?

**4 Method**

Implementation, People testing implementation, INPUT -> technique -> Output, measurements of data. How is city viable? District viable, roads viable. Research design (how will testing with humans be done)

## 4.1 Constraints

* Bara 3 district bara 3 parametrar.
* Inte L-systems eller Shape grammar
* Do not implement optimizing techniques for the rendering pipeline.
* Limit content generation to the city (i.e. no terrain generation etc.).
* Do not implement property generation (i.e. no cars or street signs etc.)

## 4.2 Early implementation plan

To generate a city viable to use in games, three different generation stages have been recognized. District, block house. 3D, flat ground, square city, Bilder för att visa stuff? Perlin noise för allting. Visa ordningen det genereras.  
OM VI SKA HA VÄLDIGT FÅ MODELLER KOMMER INTE HUSEN SE FÖRJÄVLIGA UT DÅ? IMPLEMENTERA SOM I ” *Real-time procedural generation of `pseudo infinite' cities” ELLER NÅGOT FÖR ATT FÅ SKITEN LITE MER INTRESSENT ÄN FYRKANTER I OLIKA HÖJDER? Om husen ser förjävliga ut kommer alla svara på enkäten att staden är dålig, men det är ju valid resultat det med.*

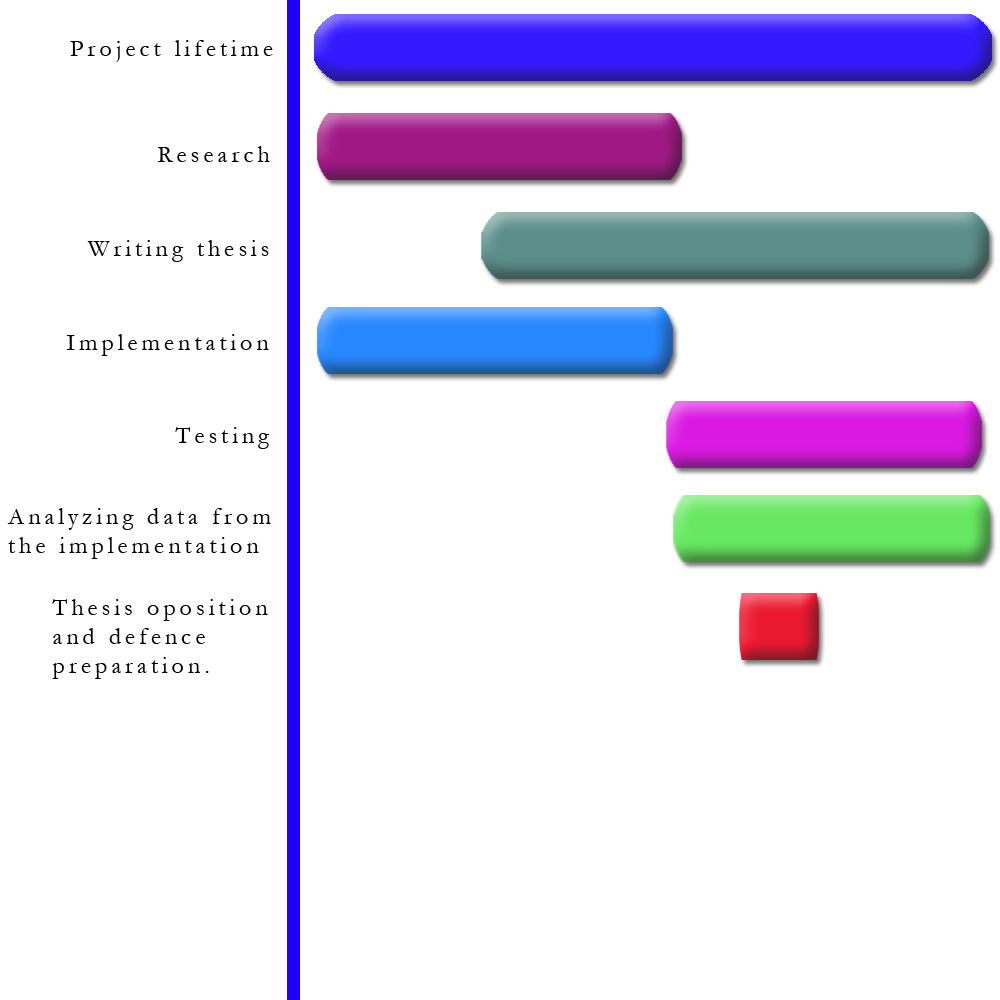
### 4.2.4 Rendering pipeline and libraries

OpenGL[21] will be used to render the city and GLM [16] for 3D math. QT [15] will be used for the user interface. The user interface is connected to the core PCG and rendering application through an interface. The user interface will modify all the previous mentioned parameters and control what the core application is generating.

**5: Expected outcomes**

Perlin noise for city is good. People will enjoy city and think it would be viable in game. Bild på stad?

**6: Time and activity plan**

  
*Figure 3: Visual representation of 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 rendering 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, data gathering and analysis of this data will begin. This data is crucial to answer the research question and conclude the thesis.   
MINDRE TEXT HÄR?

* Project lifetime 2017-03-27 – 2017-06-11
* Research 2017-03-27 – 2017-05-14
* Writing thesis 2017-05-01 – 2017-06-11
* Implementation 2017-03-27 – 2017-05-14
* Testing 2017-05-14 – 2017-06-11
* Analyzing data 2017-05-14 – 2017-06-11
* Thesis opposition and defense 2017-05-22 – 2017-05-28

**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. |
| 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. |
| Defective construction of the implementation. | 2 | 4 | Before coding the implementation make sure there is a good plan to follow. |

# References

[1] Rockstar Games. (1997-2013). Game. *Grand Theft Auto Series.*

[2] Rocksteady Studios. (2011). Game. *Batman: Arkham City.* Warner Bros, Square Enix.

[3] Farbrausch. (2004). Game. *.kkrieger.*

[4] Hello Games. (2016). Game. *No man’s sky .* Hello Games, Sony.

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

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

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

[8] Ubisoft Montreal. (2007-2015). Game. *Assassin's Creed series.* Ubisoft.

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

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

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

[12] Ken Perlin (2002), *Improving noise. SIGGRAPH 02*

[13] *Procedural Content Generation for Games: A Survey*, Mark Hendrikx, Sebastiaan Meijer, Joeri Van Der Velden, and Alexandru IOSUP Delft, University of Technology, the Netherlands

[14] *Interactive Geometric Simulation of 4D Cities*, Basil Weber,Pascal Müller,Peter Wonka, Markus Gross, computer Graphics forum

[15] https://www.qt.io/ui/ (2017). Website

[16] http://glm.g-truc.net (2017). Website

[17] *A caching approach to real-time procedural generation of cities from gis data*, Brian Cullen, Carrol O’Sullivan

[18] *Generating autonomous time-varying virtual cities*, M. Honda, K. Mizuno, Y. Fukui

[19] http://www.esri.se/Produkter/cityengine (2017). Website, CityEngine

[20] *Advanced procedural modeling of architecture*, Pascal Müller, Michael Schwarz, ACM

[21] https://www.opengl.org/about/. Website