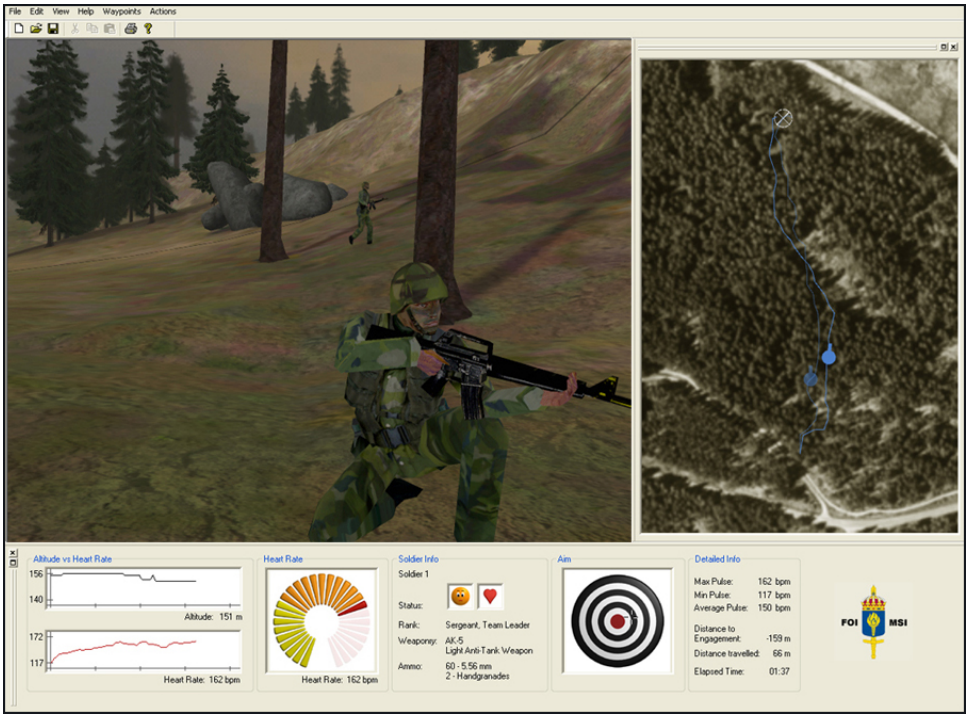
FOI-R--0978--SE
August 2003
ISSN 1650-1942
Technical report

Mattias Kindström

Geographical Mapping and Visualization of Soldiers' Heart-Rate



Command and Control Systems
SE-581 11 Linköping

SWEDISH DEFENCE RESEARCH AGENCY

Command and Control Systems
Box 1165
SE-581 11 Linköping

FOI-R--0978--SE

August 2003

ISSN 1650-1942

Technical report

Mattias Kindström

Geographical Mapping and Visualization of Soldiers' Heart-Rate

Issuing organization	Report number, ISRN	Report type
FOI – Swedish Defence Research Agency Command and Control Systems Box 1165 SE-581 11 Linköping	FOI-R--0978--SE	Technical report
	Research area code 2. Operational Research, Modelling and Simulation	
	Month year August 2003	**Project no.** E 7082
	Customers code 5. Contracted Research	
	Sub area code 21. Modelling and Simulation	

Author/s (editor/s)	Project manager
Mattias Kindström	Håkan Hasewinkel
	Approved by Håkan Hasewinkel
	Sponsoring agency Swedish Armed Forces
	Scientifically and technically responsible Mattias Kindström, Otto Carlander

Report title

Geographical Mapping and Visualization of Soldiers' Heart-Rate

Abstract (not more than 200 words)

Within the Swedish Armed Forces and the Swedish Defense Research Agency there has been an increasing emphasis put on modeling and simulation. However, a majority of the projects do not include any simulation of human behavior. The aim for this report is to produce an application presenting a concept of how to visualize a soldier's status and relate it to a position in the terrain. The status of the soldier was simulated using heart-rate.

The implemented application displays soldiers' geographical position using an animation on a 2D map, and a visualization of the soldiers' status achieved from their respective heart-rate. It also includes a 3D view of the soldiers advancing through the terrain. Two separate sound tracks, simulated heart-beats and ambient sounds, were also implemented.

The application is based only on one dataset but displays several different techniques that can be used to visualize a soldier's status. Different dataset could be connected to the different visualizations to create a more informative application.

Keywords

Geographical mapping, visualization, heart-rate, DI Guy

Further bibliographic information	Language English
ISSN 1650-1942	**Pages** 40 p.
	Price acc. to pricelist

Utgivare	Rapportnummer, ISRN	Klassificering
Totalförsvarets Forskningsinstitut - FOI	FOI-R--0978--SE	Teknisk rapport
Ledningssystem	**Forskningsområde**	
Box 1165	2. Operationsanalys, modellering och simulering	
SE-581 11 Linköping	**Månad, år**	**Projektnummer**
	Augusti 2003	E 7082
	Verksamhetsgren	
	5. Uppdragsfinansierad verksamhet	
	Delområde	
	21. Modellering och simulering	

Författare/redaktör	Projektledare
Mattias Kindström	Håkan Hasewinkel
	Godkänd av
	Håkan Hasewinkel
	Uppdragsgivare/kundbeteckning
	Försvarsmakten
	Tekniskt och/eller vetenskapligt ansvarig
	Mattias Kindström, Otto Carlander

Rapportens titel (i översättning)

Geografisk Mappning och Visualisering av Soldaters Puls

Sammanfattning (högst 200 ord)

Inom armén och FOI har det skett en ökning av intresset för modellering och simulering. Dock har många projekt enbart genomfört experiment för datainsamlande eller enbart simulering/animering. Målet med detta projekt är att skapa en applikation för att presentera ett koncept för visualisering av en soldats status och relatera denna status till en geografisk position.

Applikationen visar soldatens position med en animering i en 2D vy, och en visualisering av soldatens status baserad på soldatens puls. Applikationen innehåller även en 3D vy där framryckningen ses i ett 3D landskap. Två ljudspår implementerades, ett med pulsljud och ett som består av omvärldsljud.

Applikationen använder sig endast av ett dataset men visar flera tänkbara tekniker för hur man skulle kunna visualisera en soldats status. Fler dataset kan användas för att skapa en mer informativ och utförlig applikation.

Nyckelord

Geografisk mappning, visualisering, puls, DI Guy

Övriga bibliografiska uppgifter	Språk Engelska

ISSN 1650-1942	Antal sidor: 40 s.
Distribution enligt missiv	Pris: Enligt prislista

Table of Contents

1 Introduction

Within the Swedish armed forces today, a large number of visual 2D- and 3D-simulations are used not only to train and educate military personnel, but also to study and try new ideas and concepts. In most of these simulations the representation of human behavior has not been the focus of attention. In the simulations where human behaviors were considered, they are often not prioritized or wrongly represented.

As of recent time the armed forces have emphasized an increase in the research in the area of simulations. The behaviors of individual soldiers and soldiers in groups have been receiving more focus as new sensors and weapon-systems are under development, e.g. Network Centric Warfare (NCW) and the future soldier. These changes produce an increasing need to be able to simulate and visualize human behavior. Examples of these needs include:

- Trying and testing new equipment and weapon-systems, e.g. performing tests with heavy backpacks, protection vests etc, without endangering conscripts.
- Simulating real-life training environments, e.g. training soldiers in real-life situations for the purpose of minimizing possible risks.
- Trying ideas and concepts for NCW.

The list of areas where there is a need for simulation and visualization of human behavior can be made much longer; the areas listed above are only some examples. The process of simulating human behavior often involves trying to relate a soldier's mental and physical status to a position in the terrain, given the specific task, equipment, time etc.

This report consists of a detailed description of the creation of an application. The application includes visualizing the heart-rate and both 2D and 3D animations of the movement in the terrain.

1.1 Limitations

The initial intention was to create a 3D environment of the area of interest. However, no sufficient altitude data or terrain model could be produced within the time limits, therefore an arbitrary map had to be used which reduces the final impression. Other projects deal with creation of 3D environments why this simple solution was satisfactory for this project (Ahlberg et al., 2001; Allberg, 1997).

Another limitation was the fact that no testing or evaluation of the interface was performed. The application is merely considered to be a prototype of one possible solution of how to visualize a subject's heart-rate and mapping it to a position in the terrain.

2 Background

The data used, for the visualizations, in this application was collected at the Army Combat School at Kvarn. However, the methods used and a description of the procedure of the data collection are not included in this report. This report focuses only on the implementation of the application. The application is implemented as a stand-alone application. However, the idea for the future is to be able to use it as a plug-in to the MIND-system. The MIND-system is explained in section 2.1. The version of the application described in this report is not compatible with the MIND-system, but after future development this should be possible.

2.1 The MIND-system

The MIND-system is developed at the Department of Command and Control Systems. The main concept with the MIND-system is to register different events, both manually and automatically, during training and present the registered data for to simplify feedback and evaluation. The presented material could be used for e.g. educational purposes or as directives for future exercises (Thorstensson et al., 2002).

The purpose with simulations performed in the MIND-system is to increase the realism of simulation. The system supports different types of exercises in different areas, e.g. military exercises or rescue operations (Fjordén, 2002). The MIND-system consists of four parts from where data could be collected (Jenvald et al., 1996). These parts are:

- The instrument part
- The umpire part
- The fire control part
- The tactical radio part

The data from these four parts could then be combined and used to create a visualization of the exercise.

3 Implementation

The implementation chapter includes a presentation of the different parts of the application, and explanation of issues concerning the creation of the application. It also discusses and displays the interface.

3.1 Application Overview

The application consists of three views displayed in one window. The main part for this thesis is the part displaying the visualization of the collected heart-rate data. The other parts include a 2D animation based on the collected data and a 3D view. Figure 3.11 displays a screenshot of the different views.

The application is created as a MFC project (For more information on MFC, see Appendix A). It consists of one 3D part, implemented using DI-Guy (For more information on DI-Guy, see Appendix A), and two parts containing the animation and the visualization. Figure 3.1 displays an UML (Unified Modeling Language) class diagram describing the basic structure of the application. Other UML diagrams could be found in Appendix B.

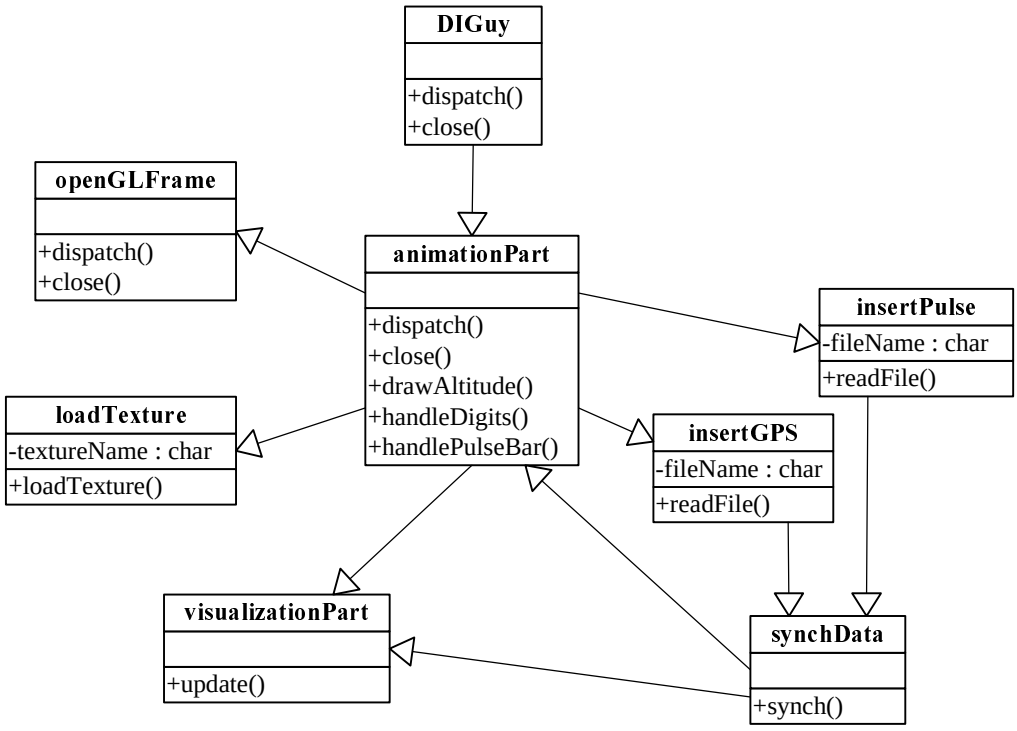


Figure 3.1. UML class diagram.

3.2 Data Processing

The data processing part includes explanations of manipulations that had to be performed in order to load the data into the application.

3.2.1 GPS Data

The received GPS-data was loaded into the computer from the GPS receiver to a software called OziExplorer (The Official OziExplorer Web Site, 2003 [Online]). The data was stored as an OziExplorer Track file (*.plt*) containing the entire path. For this project the *.plt* files could not be loaded into the created application. Therefore a conversion had to be made. The first step was to export the OziExplorer Track file with file extension *.plt* to a universal Track file, a *.trk* file.

Once the data was in *.trk* format a software called GpsLogger 1.8.1 were used. The GpsLogger was received from and developed by personnel from the MIND-project. The GpsLogger converts the *.trk* file to an XML document (For more information on XML, see Appendix A. An example of an XML document used for the project can be found in Appendix D) and the XML files were then used to load the GPS position into the application (For more information on GPS, see Appendix A).

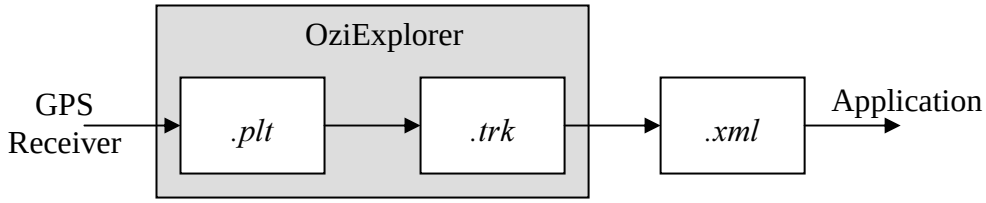


Figure 3.2. Overview of format conversions for the GPS data.

3.2.1.1 Unit Conversion

GPS data is generally stored as longitude and latitude positions. To get more understanding and to get a more manageable dataset the longitude and latitude coordinate system were transformed into a coordinate system containing the positions in meter. For this, the Transversal Mercator projection was introduced. The Transversal Mercator projection transforms the longitude and latitude positions into a coordinate system using the metric system (Reit, 2001). There are a few steps that need to be performed in order to transform these coordinates. For detailed information of these steps see Appendix C.

3.2.1.2 Position Manipulation

The GPS positions received did not provide usable position data that could be used without manipulation. The errors in measurements made by the GPS receivers produced unwanted soldier behaviors. Errors that occurred were e.g. unwanted movements when the soldier did not move and unnaturally large movements between adjacent samples. To correct this and create a correct simulation several steps had to be made to manipulate the data. These steps are not universally applicable for this first prototype. However, future versions of the application should involve creating general algorithms for position manipulation.

- Position equalizing
- Normalization
- Negative value removal
- Clustering
- Extraction of key points
- Interpolation
- Generate start/stop movement
- Rotation handling

The first step was to equalize the position of the two soldiers. Errors caused by the GPS created large distances between the two soldiers. In some datasets one soldier, throughout the entire or most parts of the exercise, were positioned more than 30-40 meters ahead of the other. This error had to be removed to create a correct simulation, and it was done by taking each sample from one soldier's dataset and comparing it to the respective sample from the other soldier's dataset. If the difference was larger than two meters, in the latitude direction, the sample was moved closer to the reference sample. The new sample was calculated to be close to the reference sample using a random position to avoid creating a static appearance. This procedure was repeated for all samples in the dataset. Figure 3.3 displays an example of the position equalization.

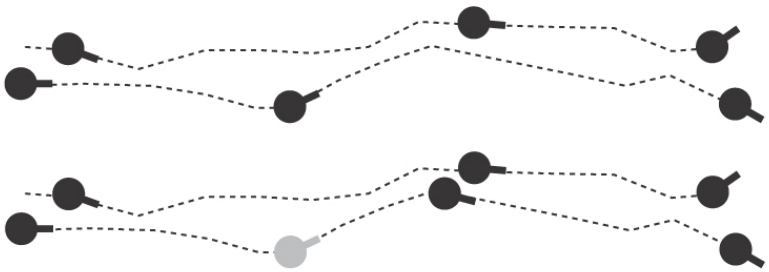


Figure 3.3. Position equalization, top original, bottom corrected.

Once all errors between the two soldiers were removed large differences in adjacent positions had to be corrected. To ensure that large unnatural movements were removed and that the movements got smoother, normalization of the data was performed. This was achieved by calculating the current position using the adjacent positions. Since the subjects' movement was in a strictly northern direction the latitude and longitude positions were calculated slightly differently.

In the longitude case, the mean value of the four values; previous, current, forward and two in front, was calculated. The resulting value replaced the current position, which created a smoother path with less large steps sideways.

In the latitude direction the mean difference between the four values where used to calculate a smoother forward movement. Since the movements were restricted to a northern direction the mean value of the positions could not yield a correct movement. Instead, the mean movement between adjacent positions

was used to create the new positions. Figure 3.4 displays an example of the normalization.

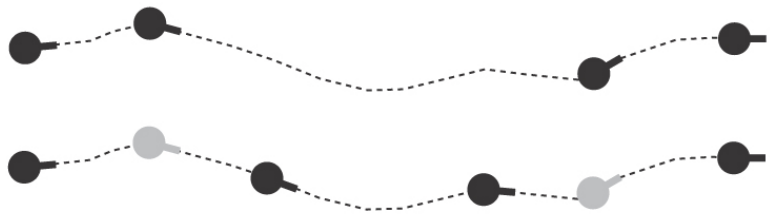


Figure 3.4. Normalization, top original, bottom corrected.

Since the exercise only involved advancing in one direction (forward, no retreat permitted), negative motion was considered errors created by the GPS receiver and had to be removed. This was achieved by comparing the current position with the previous position. If the difference was negative, the position of the previous position was stored where the current would. Figure 3.5 displays an example of the removal of negative movement.

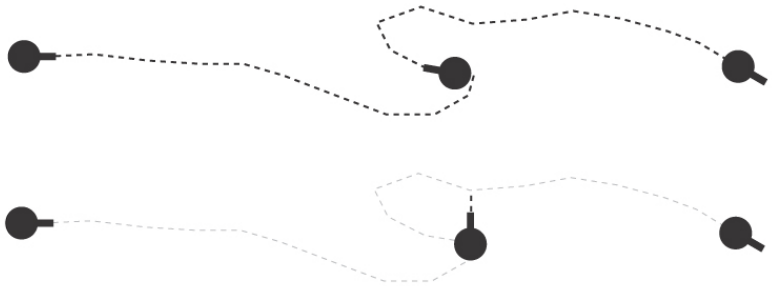


Figure 3.5. Negative movement removal, top original, bottom corrected.

The next manipulation was to remove small movements created by GPS receiver errors. When the actual position given from the data collection was fixed, the recorded position varied depending on small errors in the GPS receiver. To eliminate these small movements some form of clustering had to be implemented. This was achieved by comparing the current position with the following positions until the difference in longitude and latitude exceeds five meters. The positions that fulfill these criteria are replaced by the position stored in the current position. Figure 3.6 displays an example of the clustering of similar values.

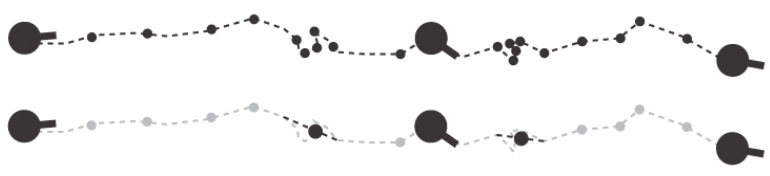


Figure 3.6. Clustering , top original, bottom corrected.

To stop the simulated soldier from changing direction too frequently, some key points had to be extracted. A sample density was to set five seconds and a sample value was added every five seconds. The values between each of these key points were the same as the previous sample density point. Figure 2.7 displays an example of the extraction of key points.

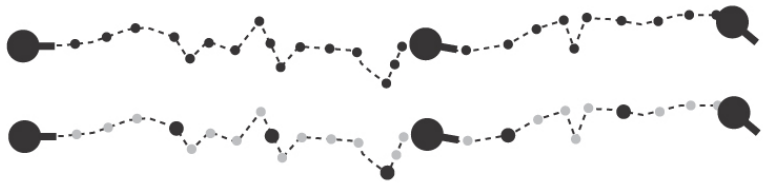


Figure 3.7. Key points extraction, top original, bottom corrected.

Once the key points were extracted the points between the key points had to be recreated. The samples between each key point had to be interpolated in order to maintain the synchronization with the heart-rate values. This was achieved by calculating the difference between two adjacent key points. The calculated difference was divided by the number of samples between the key points, in this case five. The difference was then added to the respective samples depending on their location in reference to the key points. Figure 3.8 displays an example of the interpolation of new samples.

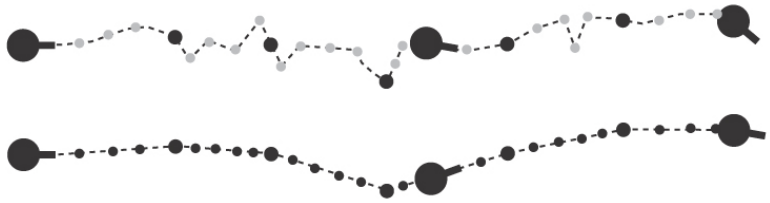


Figure 3.8. Interpolation, top original, bottom corrected.

In the exercise the subjects moved in sequences. The two soldiers tactically advanced which means that if one soldier advanced the other was in a firing position, i.e. he/she did not move. To recreate this behavior a start and stop algorithm had to be implemented. When one soldier advanced the respective samples for the other soldier were replaced with the sample at the start of the movement. Once the first soldier had finished moving the second soldier starts advancing. Figure 3.9 displays an example of the implementation of the start and stop algorithm.

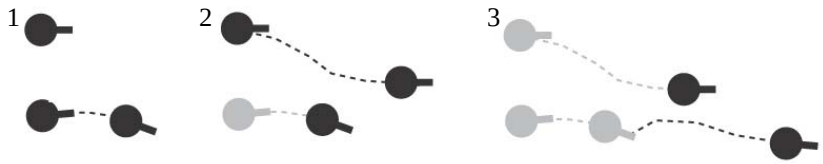


Figure 3.9. Creating start/stop movement, frame by frame.

3.2.1.3 Angle of Direction

In the XML file only the positions in the longitude and latitude directions were stored. To be able to perform an accurate simulation not only the positions needed to be known but also the direction in which the soldier was facing. To calculate this direction at a certain position the longitude and latitude values in two adjacent points, the current and the following one, were needed. There are eight different ways that these two points can represent an angle of direction. The different cases and how they are calculated are shown in figure 3.10.

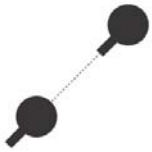
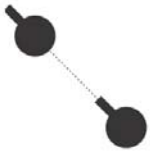
 Longitude = 0 Latitude = + Angle = 90°	 Longitude = 0 Latitude = - Angle = 270°	 Longitude = + Latitude = 0 Angle = 0°	 Longitude = - Latitude = 0 Angle = 180°
 Longitude = - Latitude = - Angle = 180 - 270°	 Longitude = + Latitude = + Angle = 0 - 90°	 Longitude = - Latitude = + Angle = 90 - 180°	 Longitude = + Latitude = - Angle = 270 - 360°

Figure 3.10. The eight possible angle calculations.

3.2.1.4 XML-Parser

Once the GPS-data was stored as an XML-file it was possible to load it into the application. This was done using a so-called XML-parser. An XML-parser parses through the XML-file searching for user-defined strings of characters. The parser written for this project was specially created to read the specific XML-file created by GpsLogger 1.8.1. See Appendix D for an example of an XML-file used in the project. Once the string of characters was found, the parser extracts the data found after the string and performs a specific task, in this case, store it in a vector for later use.

The XML-files used in this project have a specific structure that makes it possible to search for several strings in a specific order. Three different parts of information are extracted from the XML-files: time, longitude value and latitude value. The parser searches through the file once and extracts the three parts of information and stores them in vectors. The time, longitude and latitude values are stored in three different vectors.

3.2.2 Heart-Rate Data

The heart-rate data were, as the GPS data, stored in an internal file format. The file format was exclusive for the used heart-rate monitor and had to be converted before it was loaded into the application. This was solved by cutting the data from the program and pasting it into an arbitrary Word Processing program. In this project the file format used to load the heart-rate data into the application was a regular text file (extension *.txt*). Therefore, programs like Microsoft Word or Notepad could be used.

After the file had been saved as a text file it was possible to load it into the application and extract the relevant information. This was done in a similar way as with the GPS data. Since the text file not only contained pure data, but also other information, the first thing was to find where in the file the information was located. Once the correct location in the file was found the different bits of information were stored in vectors, as with the GPS data mentioned above. The information extracted from the heart-rate file was time, heart-rate and altitude. These were stored in separate vectors for easier access. For more information on heart-rate monitors, see Appendix A.

3.3 The Interface

The interface section contains information on the general interface as well as a detailed description of the different parts included.

3.3.1 General Interface Issues

When designing an application it is important to create an interface that is not only easy to understand but also pleasing to the eye. There is a number of different ways in which an interface can be designed, and a wide range of rules to follow. The application in this report consists of three parts, the 3D view and the animation and visualization parts. The 3D view and the animation part are solid frames but the visualization part consists of several smaller components. However, data on the screen should be grouped into meaningful parts (Preece, 1994). Therefore, the screen is divided into three parts where the visualization components are in the same frame.

The 3D view is the most space consuming frame. It is also the main frame of the application and is therefore located in the center, slightly to the left. The animation part is placed on the right next to the 3D view since it contains information that may not be interesting throughout the entire animation. Finally, the visualization part is placed along the bottom of the screen. Since it contains several smaller components it was easy to position them side-by-side. Figure 3.11 shows a screen shot of the application.

Since only one set of psychophysiological data, heart-rate was used; the different visualization techniques display the same thing. The application is thought to be a prototype displaying visualizations of someone's status.

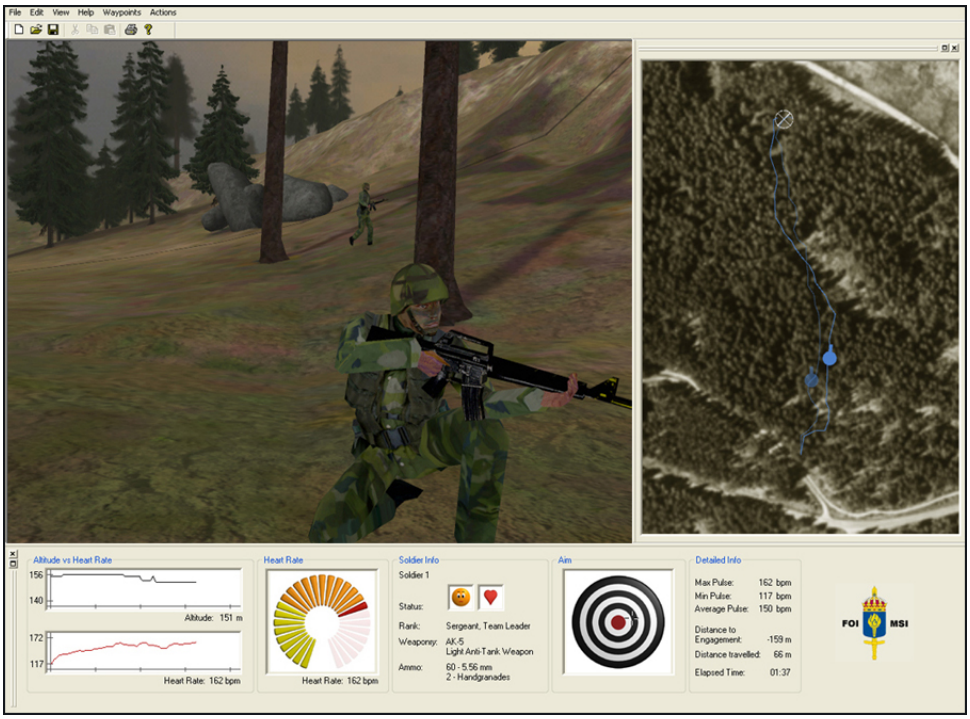


Figure 3.11. The application.

3.3.2 The Heart-Rate Visualization

The heart-rate visualization displays the heart-rate for the selected soldier. To complement a text-indicator displaying the heart-rate, a visualization was used. Today's computer systems are widely based on metaphors, e.g. the trashcan and the folder system (Preece, 1994). Based on these theories a circular heart-rate bar similar to a rev counter was designed. The heart-rate bar alters clockwise starting at the bottom left corner and ends at the bottom right corner. Below the heart-rate bar a label with the actual heart-rate value is displayed.

The color coding of the heart-rate consists of three segments, light yellow, orange and red. The colors were chosen to match the color of the faces described in Section 3.3.5. The light yellow segment is in the first third of the bar, the orange is in the middle and the red is in the final third. When the heart-rate is at a low level the bar is light yellow and when the heart-rate is in the middle region the bar turns orange. Finally, the bar turns red when the heart-rate reaches a critical level. The colors green, yellow and red could be used, to achieve an analogy to e.g. a traffic light (Hollands, Wickens, 1999), but to match the color of the faces in Section 3.3.5 the green was changed to a light yellow color. Another reason for not using green is the fact that seven percent of the male population and one percent of the female population is colorblind, and the most common colorblindness is the red-green combination (Medicular Ögonklinik, 2003 [Online]).

The color coding is intended to complement the number of filled segments at a specific heart-rate. The different color-levels as well as the max value of the meter should be individually implemented for each soldier. However, for this application only a standard value, the same for all soldiers, were used. This means e.g. that if the bar is on red, it does not mean that the soldier has a specific status. It merely means that the pulse has surpassed a certain level and that it is at a relatively high level.

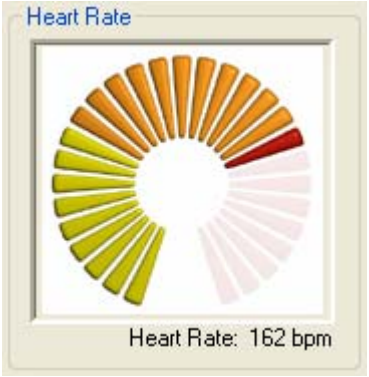


Figure 3.12. The heart rate visualization.

3.3.3 Altitude and Heart-Rate Visualization Over Time

The changes in heart-rate and altitude over time are important features. It is of interest to be able to see if a high heart-rate is caused by a recent increase in altitude or if it is caused by other things, such as stress. It is also of use since a single heart-rate value does not provide much information. To get a good reference for the current heart-rate, the history of the heart-rate's changes is needed. Therefore, two graph frames were implemented, one displaying the altitude change over time and one displaying how the heart-rate alters during the simulation. The current heart-rate and altitude value are displayed below the respective frame. Also, to get a reference to the current, the maximum and minimum values of both graphs are displayed on the left side of each graph.

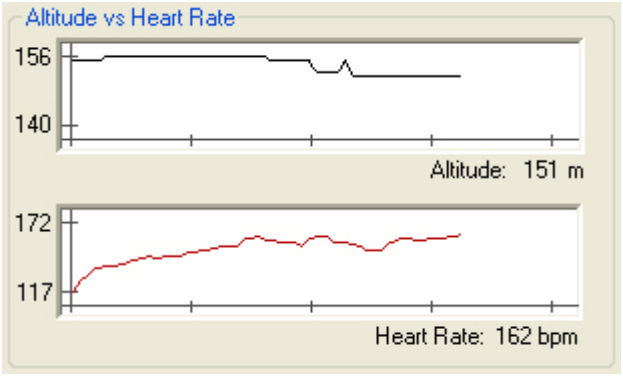


Figure 2.13. The altitude and heart-rate graphs.

The graph frames were based on the same concept as a line-graph, where the focus was to make it easy to see trends and changes during the simulation,

since line-graph is a good choice when displaying variables that change over time (Preece, 1994). The colors used were black on a white background for the altitude graph, and dark red on a white background for the heart-rate graph. These colors were chosen because a dark-white combination provides a better amount of contrast than e.g. a red-green or blue-yellow combination (Ware, 2000).

3.3.3.1 Axis Scaling

There was a large difference in the input values for the two graphs. Heart-rate values are often in the region around 100, while altitude values range from zero to a couple of thousands depending on the terrain. Also, a difference in altitude of e.g. two meters requires a different scaling than an altitude difference of 200 meters. If the same scaling is used for graphs with large differences small changes might not be visible. In order to get a similar look on the two graphs and a similar slope on the curves the scaling of the axis had to be controlled. This was done using equation 3.1 where Q is the output value and P is the input value (Gonzalez, Woods, 2002).

$$Q = \frac{P - P_{\min}}{P_{\max} - P_{\min}}$$

Equation 3.1. Scaling equation for the axis.

Equation 3.1 changes the range of the input values, independent of the current range, to be between zero and one. Once the values are scaled to be in the zero/one region it is easy to multiply the two new datasets with a suitable number to fit the size of the frame. The maximum and minimum values of the two graphs were positioned on the same place in the two frames, and the values used the maximum amount of space available in the frames.

The scaling along the time axis was set to be the same for the two graphs. However, the entire dataset was too large to display. Therefore, the time axes only display the data from the two previous minutes. The first two minutes new samples are added at the end of the graph. After two minutes, samples older than two minutes are not displayed.

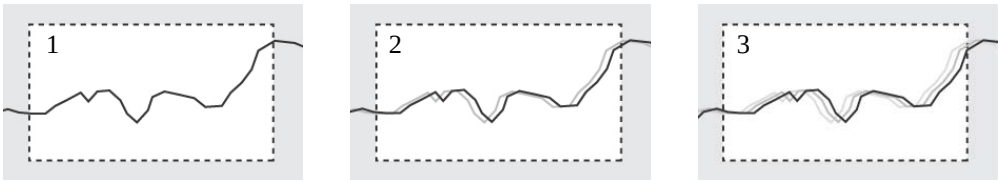


Figure 3.14. Frame 1: Two minutes reached. Frame 2: New value added on the right side, entire graph is moved one step the left. Figure 3: New value added, graph moves one step.

3.3.4 The Sight Visualization

The sight visualization is used to provide the user with information showing the possibility of the selected soldier hitting a target at that moment. Studies performed at the MSS Kvarn Combat School have shown that a sniper has best

hit percentage if the heart-rate is between 60 and 70 bpm. At a heart-rate of 90 bpm there is a risk that the sniper will not be able to have an acceptable hit percentage. If the heart-rate exceeds 100 bpm, hitting the target is extremely difficult (Hasewinkel, 2002). However, in this project, these tests were only used as a theoretical foundation and a source of inspiration. No snipers were involved and the heart-rate is not of the same importance for regular soldiers as for snipers.

Although the heart-rate is not of the same importance to the hit percentage of regular soldiers as it is to snipers, the heart-rate could have an effect on the hit percentage. The major factor deciding if a soldier hits the target or not is the amount of breathing. If a soldier breathes heavily the hit percentage is low and if the same soldier breathes calmly the hit percentage is high (*Skjutreglemente för Armén*, 1994). The breathing is in many cases related to the heart-rate. If it is not, then something else might be affecting the soldier, e.g. stress.

The sight visualization is not considered a main visualization. Instead it could be used as a bi-visualization displaying a certain aspect of the soldier's mental status. However, a better reference of what is a good value and what is a bad value is needed. It could be difficult for a user to know how the sight should move if the soldier's status is ok or if the status is bad. Also more data analysis is needed. E.g. if the soldier's breathing could be measured and the soldier has a high heart-rate as well as a heavy breathing the soldiers status could be alright. If the soldier on the other hand has a high heart-rate and calm breathing the status could be poor.

The sight icon and the target were designed as simple as possible to simplify the viewing of the animation. The sight's position was calculated based on value of the heart-rate. Using the heart-rate, a specific amount of randomized movement and a simulated shakiness were added to move the sight away from the center. The amount the sight moved away from the center was also altered depending on the heart-rate. High heart-rate gives high randomized movement, much shakiness and a large distance from the center.

Figure 3.15. The sight visualization.

3.3.5 Status Information

The status information part displays information about the selected soldier. At the top of the frame the selected soldier is displayed. Just below are two visualizations of the soldier's status, one face frame and one heart frame. The mental status in this project is represented by the heart-rate value of the soldier.

Facial expressions play a crucial role in non-verbal communication. They express a subject's emotional status (Du, Lin, 2002). Therefore, facial expressions can be used to display a subject's condition. There are six different basic expressions: joy, sadness, surprise, fear, disgust and anger (Ruttkay et al., 2003). For the application the characteristics for joy and sadness are used as a basis for declaring the subject's status. Characteristics for a face expressing joy are an open face with raised slightly arched eyebrows and an upwards pointing mouth. Similar, characteristics for a face expressing sadness are a slightly closed face with lowered slightly tilted eyebrows and a downwards pointing mouth (Grammer et al., 2002).

In the application the subject's status is visualized using 11 different facial expressions. There is no specific reason why exactly 11 different facial expressions were used. The simplest way to display the state could be to use only two faces, one happy face and one sad face. However, to emphasize that the status is continuously changing as the heart-rate changes more faces were used. By doing this the user understands that there are more levels to a person's mental status than just good or bad.

The 11 different facial expressions are based on the collected data. If the status of the soldier is good there is a happy expression and if the soldier's status is poor there is a sad expression. To emphasize the expressions the face images were colored differently depending on the status. The face expressing joy was colored yellow and the face expressing sadness were colored red and the stages between were colored in various orange colors. In figure 3.16 below the designed face images are displayed.

Figure 3.16. The 11 images showing different facial expressions.

The heart frame displays a pulsating heart. It is similar to the face frame based on the soldier's heart-rate. It simulates the soldier's heart rate by pulsating at different speeds depending on the heart-rate value. If the heart-rate is high the pulsation is fast and if the heart-rate is slow the pulsation is slow.

The status information frame also displays some information about the soldier not based on the animation. The rank of the soldier is displayed as well as the weaponry the soldier carries and the amount of existing ammunition.

Figure 3.17. Soldier information.

3.3.6 Detailed Information

The part with the detailed information consists of continuous representation of some variables from the simulation. The maximum, minimum and the mean heart-rates of the selected soldier are displayed. Also, the distance the soldier has traveled and the distance remaining to the point of engagement are displayed. At the bottom of the frame a counter is placed, displaying the time elapsed since the beginning of the exercise.

The detailed information is used to give additional information about the soldiers that otherwise might not have be seen in the simulation. Some data that might be interesting in the evaluation of the exercise has been included. It is seen as a help function and not as a center part of the visualization. This is why it has been placed in the bottom right corner.

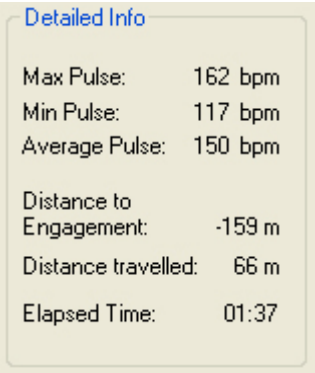


Figure 3.18. Detailed information.

3.3.7 The Animation Part

The animation part was created using OpenGL (For more information on OpenGL, see Appendix A). The reason for using OpenGL is the simplicity in creating scenes and performing animations. The OpenGL scene was displayed in a separate window and consists of two main parts, the map and the soldiers.

The map consists of a plane with a texture added to it. Normally the materials of objects, in OpenGL, have a single color but in this case Bitmap image-files were used as material. This created a plane image similar to an ordinary map. There were two different images used. The default image was an ortho-photo of the exercise area. It was used to get a better understanding of the terrain. It is easier to see if the soldiers are advancing in e.g. a forest or a meadow. Apart from the default image, an ordinary map was used to get a different view of the terrain, e.g. to view height differences. The plane with the maps was rotated and translated to be parallel to the screen to face the viewer. The images for the maps were obtained from the National Land Survey of Sweden (The National Land Survey of Sweden, 2003 [Online]).

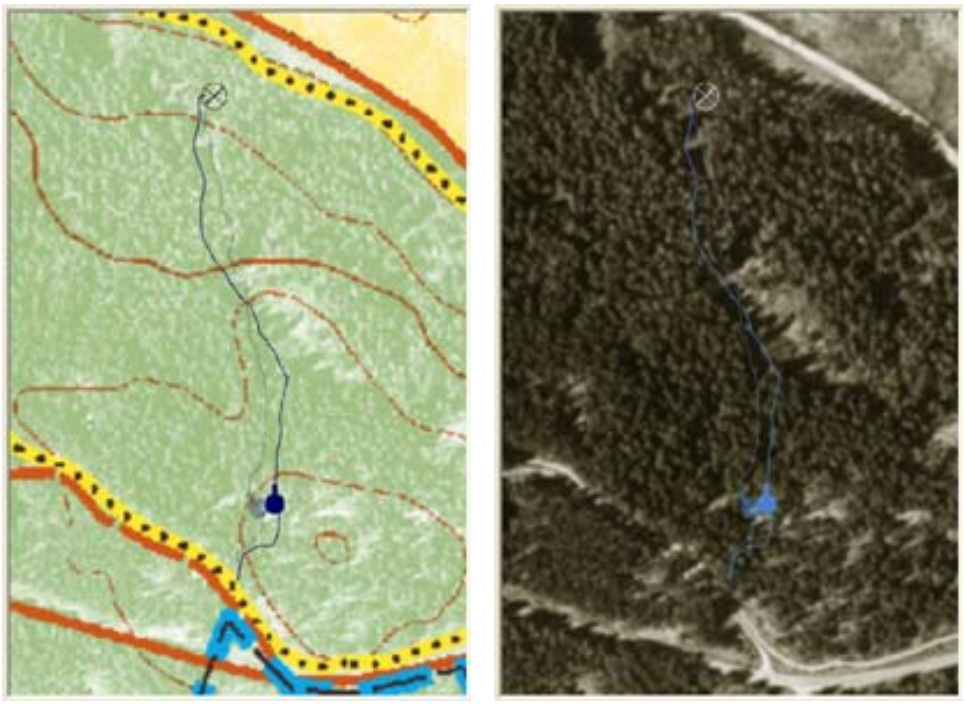


Figure 3.19. The animation part, on the left: the ordinary map, on the right: the ortho-photo.

The two soldiers were created using standard OpenGL commands, and they were represented with the military symbol for a “soldier”. The selected soldier has a solid color while the other is semi-transparent. The colors depend on the displayed map, since the map is bright and the ortho-photo is dark. If the background is dark the figure should be bright and vice versa (Preece, 1994). The color of the soldiers is blue, since symbols for friendly troops, within the Armed Forces, are marked as blue. If the ortho-photo is selected the blue color is light and if the map is selected the color is dark blue.

The actual animation was achieved by using the animation loop within OpenGL. Each time OpenGL updates the scene, a small translation and/or rotation of the soldiers is made. Key frames achieved from the dataset are used and in each loop the translation and/or rotation are interpolated to perform a smooth movement.

Apart from the animated soldiers the user can display other informational symbols. The entire path of both soldiers could be displayed. The color of the paths is the same as the color of the respective selected soldier, i.e. blue if selected and transparent blur if not selected. The user can also decide to display the position of engagement and the enemy position. They are both displayed as a cross with a surrounding circle. The color of the position of engagement is white if the dark ortho-photo is selected and black if the bright map is selected. The color of the enemy is red, in line with the symbol of enemy troops within the Armed Forces.

3.3.8 The 3D View

The 3D view consists of an arbitrary terrain model chosen to be as similar as the terrain used as possible. The initial idea was to use a 3D model of the terrain where the data collection took place but this was not possible. Therefore a 3D model from the PC-game Battlefield was used.

In the terrain model, key points so-called waypoints were inserted to create the paths for the soldiers. The waypoints were manually inserted in the 3D world using the mouse. Also, action keys were inserted to define where different actions, e.g. run or kneel aim, should be performed. The lengths of the actions were also specified in each action key point, e.g. if a soldier should kneel for 10 seconds it had to be specified. Using waypoints and action a path was created to be as similar as the path based on the dataset shown in the 2D view.

Figure 3.20. On the left: path created using waypoints.
On the right: actions (in blue) added to some waypoints.

3.4 Sound Effects

To show that sound could be included in applications similar to this, two different sound tracks were implemented, one heart-rate track and one track with ambient sounds. The user has the opportunity to switch between the different tracks in real-time or to play them simultaneously.

The sound was not considered a major part of the project, but it was used to add an extra sense of reality to the animation. Studies have shown that using sound, expected by the user, increases the user's sense of presence (Chueng, Marsden, 2002). Therefore, a heart-beat sound was used to add another dimension to the bar displaying the heart-rate. It provides the user with a better sense of the changes of the heart-rate. The ambient sounds were used to provide the user with sound that could be heard during a similar situation.

4 Result and Discussion

This chapter involves a discussion on the programming issues that occurred during the project. It also includes a discussion about the implemented application.

4.1 Programming Issues

The application was developed in C++, which is a widely known computer language that simplifies the implementation process. Visual C++ that was used to create the actual interface is an easy tool to use for the creation of an interface. However it can be quite lengthy for control and manipulation of the interface.

Since the GPS data could not be used without manipulation there is a risk that the simulated path does not exactly correspond to what was performed during the collection of the data. Since the GPS data needed to be manipulated before use, there is a risk that important soldier movements were lost in the process.

The animation part was created using OpenGL, which was used for its abilities to create animations. The OpenGL frame was easy to implement and control from the Visual C++ interface. One drawback with OpenGL is the difficulties in creating detailed objects, which gives the application a simple OpenGL animation.

Boston Dynamics' software, DI Guy, was used to create the 3D view. Since there were no way to insert the position coordinates, the key-points hade to be manually placed in the 3D world, which is very time-consuming task. DI Guy is also hardware dependent, which means long initiating-times and that a high performing computer system is needed.

Since a 3D model of the actual terrain where the data collection took place could not be used, the 3D view in the application is not correct. An arbitrary 3D world was used to show that the concept could be used. This means that the collected altitude data is not used in the 3D view. Also, the terrain is not the same as on the correct 2D map which makes e.g. trees and rocks appear in the wrong places.

4.2 The Application

The layout of the interface was designed after how much space they required. The 3D view is probably the most interesting frame for continuous viewing, while the 2D view and the visualization frame are the most informational frames. Also the 3D view needs a lot of screen space in order to be viewed acceptable. The 2D frame could use a zooming effect to display the information in more detail and the visualization part needs less space to supply the displayed information.

Since only one data set containing heart-rate values was used, the different sub parts within the visualization part displays the same data. The different visualization techniques included in the application could help a user to interpret data. The techniques could also be used to display different measures, but for this application they serve as concepts on the possibilities to visualize status.

5 Future Work

This report did not intend to result in a final application; more to try the general concept and to create a prototype. The project showed that the concept could be improved and the application could be implemented on a number of areas.

One issue that could be interesting to develop further is to be able to run an application similar to this in real time. The application implemented in this project was intended for monitoring the exercise after the data had been collected. A future development is to be able to collect psychophysiological and geographical data in real time, from a number of subjects simultaneously, and visualize it in single application. As mentioned earlier, it could also be possible to include similar visualization techniques and a 3D view in the MIND-system for monitoring and evaluation of larger military exercises.

Another interesting idea is to add artificial intelligence to the application. This would be a major part in a future application. Computer Generated Forces (CGF) could be used to evaluate the status of soldiers in a certain terrain during a certain exercise. CGFs could also be used to add third-party forces to create an environment similar to combat scenes. To create satisfactory simulations a future application needs an interface where the user could specify data for the soldiers in the simulation, e.g. weight, age, stress durability. This feature could be used to create simulations with real subjects, to see how personnel handle different tasks and to test new equipment.

Another area within the project that could be developed further is the storage of the collected data. For this version, the data is stored as text in one XML-file and one Text-file. In the future it would be preferable to store the collected data in a database to simplify storage and data access.

Several aspects of the 3D part of the application could be implemented further. For the prototype produced in this project the 3D view merely emphasized the possibilities with 3D computer graphics within modeling and animation. The actual terrain where the data collection was made was not used in the application. For future development the correct environment would improve the overall accuracy of the application. Also a transformation of the collected coordinates to the 3D view could be improved. Since no height data was available for the 3D world used in the application, the coordinates collected could not automatically be inserted into the 3D world. This had to be done by hand, which lead to errors in the similarities between the 2D view and the 3D view. A fully automated conversion would be preferable in a future application.

Another aspect of the 3D view is the fact that positional data alone could not produce a correct animation. For this application the motions of the soldiers were limited to the motions available in DI Guy, which is not enough to create correct animation of movement. Future development could include a motion capture system to expand the positional data with data including body movement.

References

Ahlberg S., Elmqvist M., Hermansson P., Jacobsson J., Persson Å., Söderman U. (2001), *Synthetic Environments and Sensor Simulation*, FOI-R—0292—SE, Försvarets Forskningsinstitut.

Allberg H. (1997), *Laser-Radar Images for Generation of Up-To-Date Digital Terrain Models*, FOA-R—97-00399-408—SE, Försvarets Forskningsinstitut.

Boston Dynamics (2003) [Online], [Accessed 15th July 2003], <http://www.bostondynamics.com>.

Chueng P., Marsden P. (2002), *Designing Auditory Spaces to Support Sense of Place: The Role of Expectation*.

Du Y., Lin X. (2002), *Mapping Emotional Status to Facial Expressions*.

Encyclopedia Britannica (2003) [Online], [Accessed 28th April 2003], <http://www.britannica.com>.

Feuer A. (1997), *MFC Programming*, ISBN 0-201-63358-2.

Fjordén J. (2002), *Representation and Visualisation of Casualty Flows in Rescue Operations,* FOI-R—0496—SE, Försvarets Forskningsinstitut.

Fowler M., Scott K (2002), *UML Distilled*, Second Edition, ISBN 0-201-65783-X.

Gonzalez R., Woods R. (2002), *Digital Image Processing*, ISBN 0-13-094 650-8.

Grammer K., Tessarek A., Hofer G. (2002), *From Emotions to Avatars: the Simulation of Facial Expression*.

Hasewinkel H. (2002), *Hur ska vi bygga en avsutten soldat för användning i 3D simuleringar?*, FOI diarenummer:02-3200, Försvarets Forskningsinstitut.

Hollands J., Wickens C. (1999), *Engineering Psychology and Human Performance,* Third Edition, ISBN 0-321-04711-7, pg 100-101, pg 465-467, pg 481-507.

Jenvald J., Morin M., Worm A., Örnberg G. (1996), *MIND – ett instrument för värdering, utveckling och utbildning av krigsförband*, FOA-R—96-00351-3.8—SE, Försvarets Forskningsinstitut.

Josuttis N. (2000), *The C++ Standard Library,* ISBN 0-201-37926-0.

Leick A. (1995), *GPS Satellite Surveying*, Second Edition, ISBN 0-471-30626-6, pg 1-10.

Lindén M. (2002), *Handbok i GPS*, ISBN 91-89564-03-0, pg 18-31.

McLaughlin B. (2000), *Java and XML*, First Edition, ISBN 0-596-00016-2, pg 2-4.

Medicular Ögonklinik (2003) [Online], [Accessed 24th August 2003], <http://www.medicular.se>.

Mendosa R., *Medical Writer and* Consultant (2003) [Online], [Accessed 24th May 2003].

Preece J. (1994), *Human-Computer Interaction*, ISBN 0-201-62769-8, pg 89-91, pg 100-105, pg 145-149.

Reit B-G. (2001), *Gauss conformal projection (Transverse Mercator)*, The National land survey of Sweden.

Ruttkay Z., Noot H., ten Hagen P. (2003), *Emotion Disc and Emotion Squares: Tools to Explore the Facial Expression*, Volume 22 pg 49-53.

Skjutreglemente för Armén, Pansarskott, Eldhandvapen och Kulsprutor Del 2 (SkjutR A PEK 2), (1994), pg 36-37.

The National Land Survey of Sweden (2003) [Online], [Accessed 12th May 2003], <http://www.lantmateriet.se>.

The Official Website of Trimble Navigation Ltd (2003) [Online], [Accessed 3rd May 2003], <http://www.trimble.com/gps>.

The Official OziExplorer Web Site (2003) [Online], [Accessed 25th July 2003], <http://www.oziexplorer.com>.

Thorstensson M., Jenvald J., Morin M. (2002), *Modellering och Visualisering av Marin Förband*, FOI-R—0524—SE, Försvarets Forskningsinstitut.

Ware C. (2000), *Information Visualization – Perception for design*, ISBN 1-55860-511-8, pg 3-4, pg 118-136.

Woo M., Neider J., Davis T., Shreiner D. (2001), *OpenGL Programming Guide*, Third Edition, ISBN 0-201-60458-2.

Wright Jr R., Sweet M. (2000), *OpenGL SuperBible*, Second Edition, ISBN 1-57169-164-2.

Appendix

A. Technologies

A.1 GPS – Global Positioning System

The Global Position System (GPS) is a satellite-based navigation system that can be divided into three parts: the space segment, the control segment and the user segment (Leick, 1995):

- The space segment is the foundation of the system. It is made up of a network consisting of 24 satellites. These satellites are arranged in a way that a user segment can receive information from at least four satellites at all times. The satellites were placed in an orbit, at a height of approximately 19.3 kilometers, by the U.S. Department of Defense.
- The control segments control the satellites by tracking them and providing them with correct orbital information and a time stamp. There are five control stations situated around the world that handles the control of the 24 satellites.
- The user segment is the GPS receiver used to track the signals transmitted from the satellites.

The U.S. Department of Defense started building the GPS in 1973, and during the 1980s several satellites were launched. However not until 1995 did the system reach its full operational capacity. GPS is today the only positioning system accessible for civilians. GPS is intended to work in any weather conditions, any time of day all year around anywhere on the planet (Lindén, 2002). GPS receivers are also relatively cheap to buy and widely used in society today which simplifies the using of the technology.

A.1.1 How Does GPS Works?

The idea behind GPS is to use satellites in space to calculate the position on earth. The satellites orbit the earth two times a day and continuously transmit signals down to earth. A GPS receiver receives the signal and compares the signal's time stamp with the current time. This time difference and the fact that the signal's speed is the speed of light, makes it possible for the receiver to calculate the distance to the satellite. If the GPS receiver connects with three satellites the receiver can determine a 2D position, i.e. longitude and latitude, and if the receiver connects to four or more satellites it is possible to add a third dimension of precision, in this case altitude (Leick, 1995).

Example A.1:
Suppose the distance from a satellite is 20 kilometers. This gives a location of any position on a sphere around the satellite with a radius of 20 kilometers. The receiver then measures the distance from a second satellite and finds that the distance is 22 kilometers. The second satellite tells us that the position is not only somewhere on the first satellite's sphere but also on the sphere of the second satellite. This narrows down the position to the intersection between the

two spheres, in this case a circle. If a third satellite is used it is possible to narrow down the selection of possible positions to the two points where all three spheres intersect. Although it is often trivial to rule out one of the two points as unrealistic (not on the surface of the earth) adding a fourth satellite will give one exact point in space. The process is described in figure A.1 below (The Official website of Trimble Navigation Ltd, 2003 [Online]).

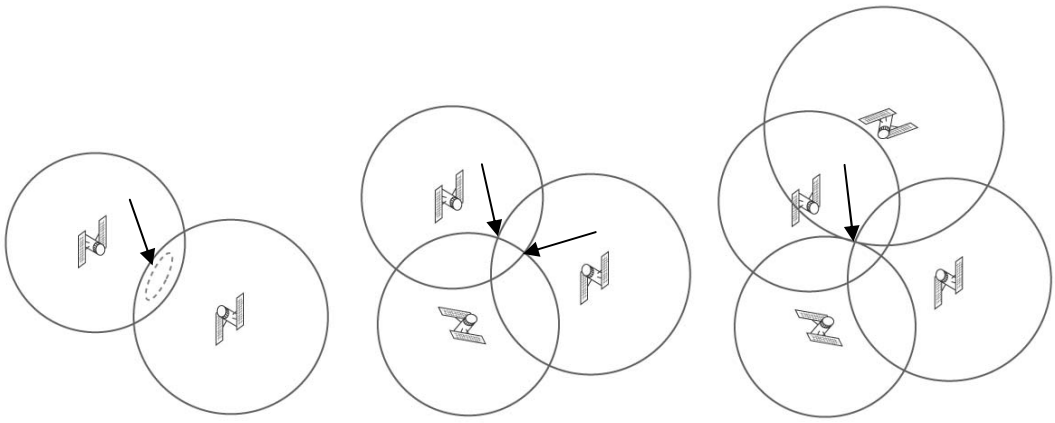


Figure A.1. Left figure: Dotted ellipse displays the possible positions with only two satellites. Middle figure: Two possible solutions at the intersection of the three satellites. Right figure: Only one possible solution using four satellites.

A.2 Heart-Rate Monitors

Heart-rate monitors are simplified portable electrocardiogram (ECG). The difference is the accuracy of the measurements. A regular ECG records every single heartbeat while a heart-rate monitor records as many heartbeats as it can detect and interpolates the missing beats. An ECG works by recording the electrical activity within the heart using various electrodes placed on the surface of the skin. Every heartbeat generates an electrical signal that can be measured on the skin (Encyclopedia Britannica, 2003 [Online]).

Heart-rate monitor transmitters contain similar electrodes as a regular ECG but the electrodes are mounted in a belt that is attached to the chest. The transmitter detects the voltage differential on the skin from every heartbeat and transmits the signal to a receiver, commonly a device similar to a wristwatch. The receiver displays the heart-rate in beats per minute (bpm) (Mendosa R., Medical writer and consultant, 2003 [Online]).

A.2.2 Barometric Pressure

On some heart-rate monitors it is also possible to monitor the altitude at every sample point, to view changes in the altitude during the exercise. This is often done by measuring the barometric pressure. The heart-rate monitor is reset at a known altitude, often the sea level, to get the correct barometric pressure at a specific altitude. As the person wearing the heart-rate monitor moves, and the altitude changes, the barometric pressure changes and the heart-rate monitor produce a physical altitude from these changes.

A.3 XML

XML is an abbreviation for Extensible Markup Language. XML is a language used to define other languages. XML is like the more widely spread HTML based on sets of tags. The set of tags defines the markup tags the interpreting language uses to make sense of the code. A tag set consists of a starting tag and an end tag, where the code between these is affected by the attributes given by the tag. Some tags however, can be in the form of attributes where no end tag is required. In XML there is no limit to the amount of different tags that can be created, unlike HTML where only a finite set of tags is available. XML is like the name suggests completely extensible (McLaughlin, 2000).

Example A.2: Sample of an XML-file. In the example the tag <book> is an example of a regular tag and <title=…> is an example of an attribute tag.

```
<book>
     <cover>
     <title=”The title of the book”>
          <cover color=”BLUE”>
          <cover type=”paperback”>
     </cover>

     <pages>
          <paper quality=”4”>
          <amount=”432”>
          <size=”A3”>
     </pages>
</book>
```

XML can be used in a variety of ways spanning from creating a regular webpage to defining new specialized languages. In this project XML was used to create a document from which it was easy to extract information. An example of an XML file used in the project can be found in Appendix A.

A.4 OpenGL

OpenGL is a software interface for a graphics hardware that is used to specify objects and operations to produce interactive 3D applications. It consists of approximately 250 distinct commands used to specify the objects and operations needed to produce interactive 3D applications. Objects in OpenGL are created from a small set of geometric primitives, points, lines and polygons (Woo et al., 2001).

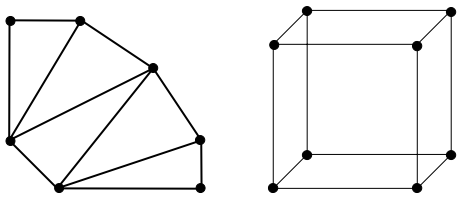


Figure A.2. Examples of object creation in OpenGL. Left: triangle strip, right: cube created from six squares.

Once the models, or objects, have been defined the scene is rendered. Rendering is the process where the computer creates an image of the scene. The computer translates the 3D scene to a 2D image consisting of pixels drawn on the screen. Interactivity and animation in OpenGL is achieved by redrawing the calculated image over and over again. An animation is created if objects in the scene are translated, rotated or scaled in each loop (Woo et al., 2001). Figure 2.3 shows a simple animation where the object is translated and rotated in each frame.

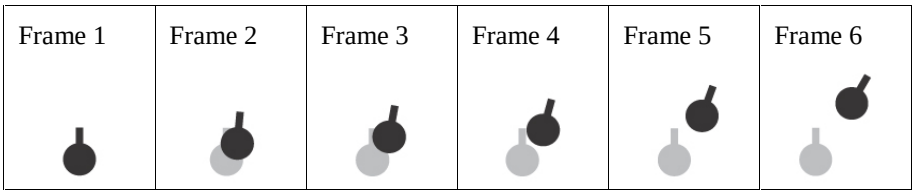


Figure A.3. The OpenGL animation loop.

A.5 Microsoft Foundation Classes

The Microsoft Foundation Classes (MFC) is a C++ class library used to create Windows applications. It provides an object-oriented wrapper around the Windows API. MFC is one of several Windows class libraries available to simplify Win32 programming. The native interface to Win32 is the C Application Program Interface (API). The MFC library calls the C API to access Win32 (Feuer, 1997).

MFC programming involves deriving new classes from classes already in the Windows library, inheriting and replacing behavior as appropriate. These high-level classes are known as an application framework. The framework part of MFC is where the programming is done and the wrapper part is the translator that communicates with the Windows API (Feuer, 1997).

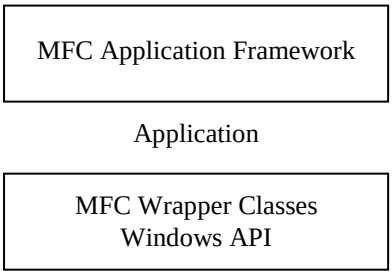


Figure A.4. Overview of an application written using MFC.

A.6 DI-Guy

DI-Guy is a software for adding lifelike human characters to real-time 3D-simulations developed by Boston Dynamics (for more information on Boston Dynamics see (Boston Dynamics, 2003 [Online])). DI-Guy includes a real-time motion engine, a number of characters that can be included in various

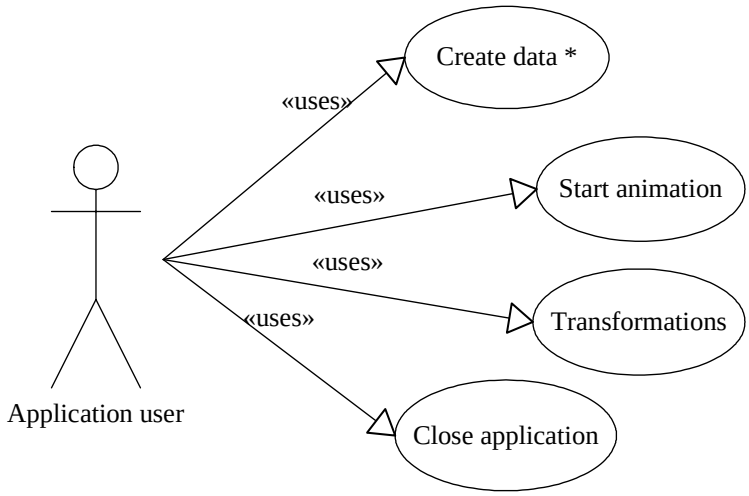
simulations and an API that lets the user manipulate the characters or the simulation in a variety of ways.

With the API, characters can be added using high-level functions and manipulated for special behaviors using a variety of available function calls. DI-Guy automatically creates natural-looking smooth behavior based on the used function calls. DI-Guy contains a wide array of characters that can be added to an application and more than 1000 different motions that can be used to animate the characters (Boston Dynamics, 2003 [Online]).

B. UML

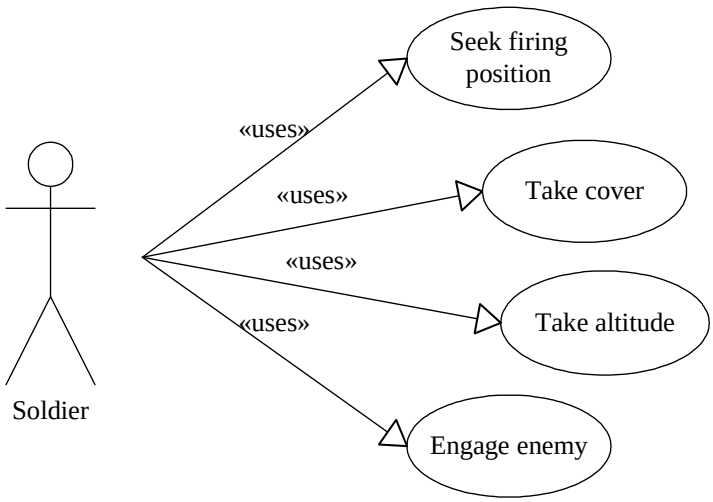
B.1 Use Cases

B.1.1 Using the Application



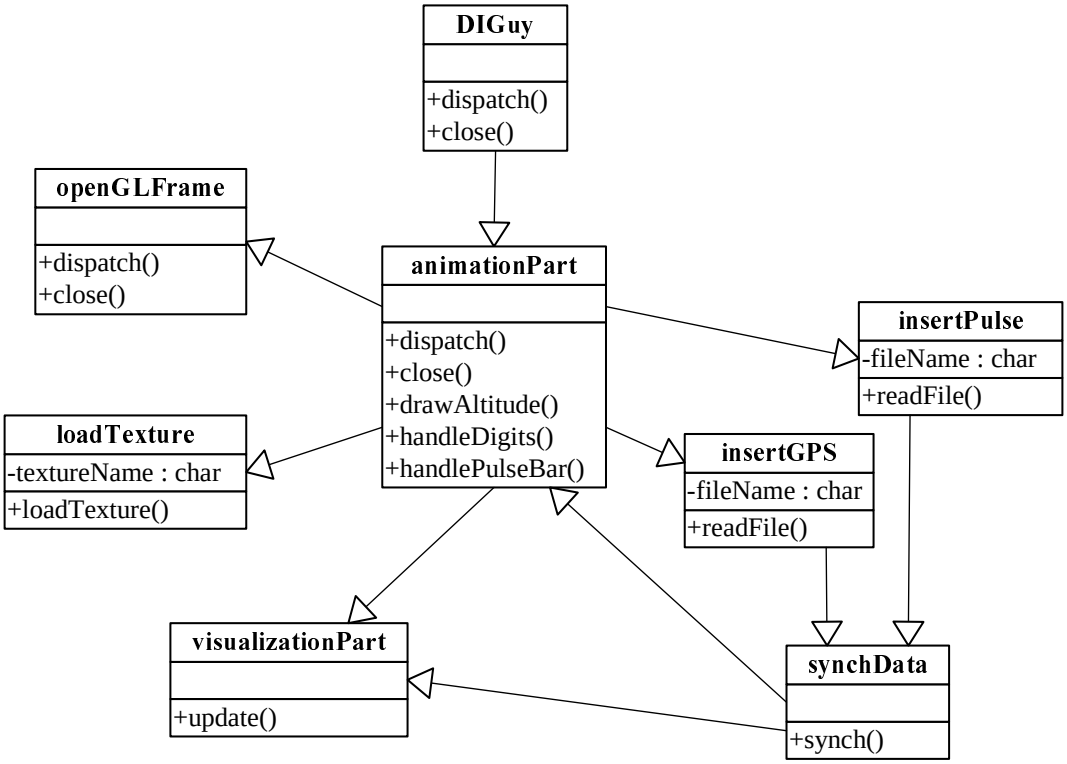
Scenario: Using the application
1. An user launches the application.
4. The animation is initialized.
5. View transformations are made, zooming and translation of the map.
6. When the user is done he/she closes the application.

B.1.2 Create Data

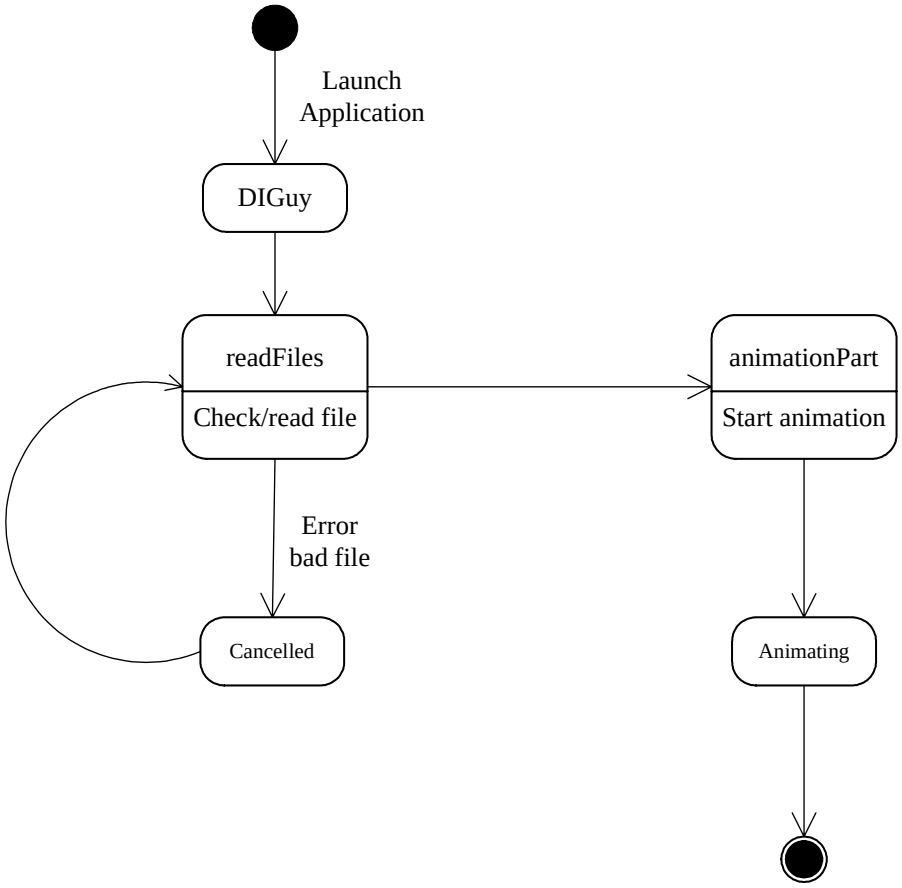


Scenario: Create data
1. A fire and maneuver team moves through a piece of woodland.
2. They perform tactical advances towards a small height.
3. When the team reaches the height, they establish a position for engagement and wait.
4. The team identifies the enemy ammunition support vehicle.
5. The enemy ammunition support vehicle is engaged.

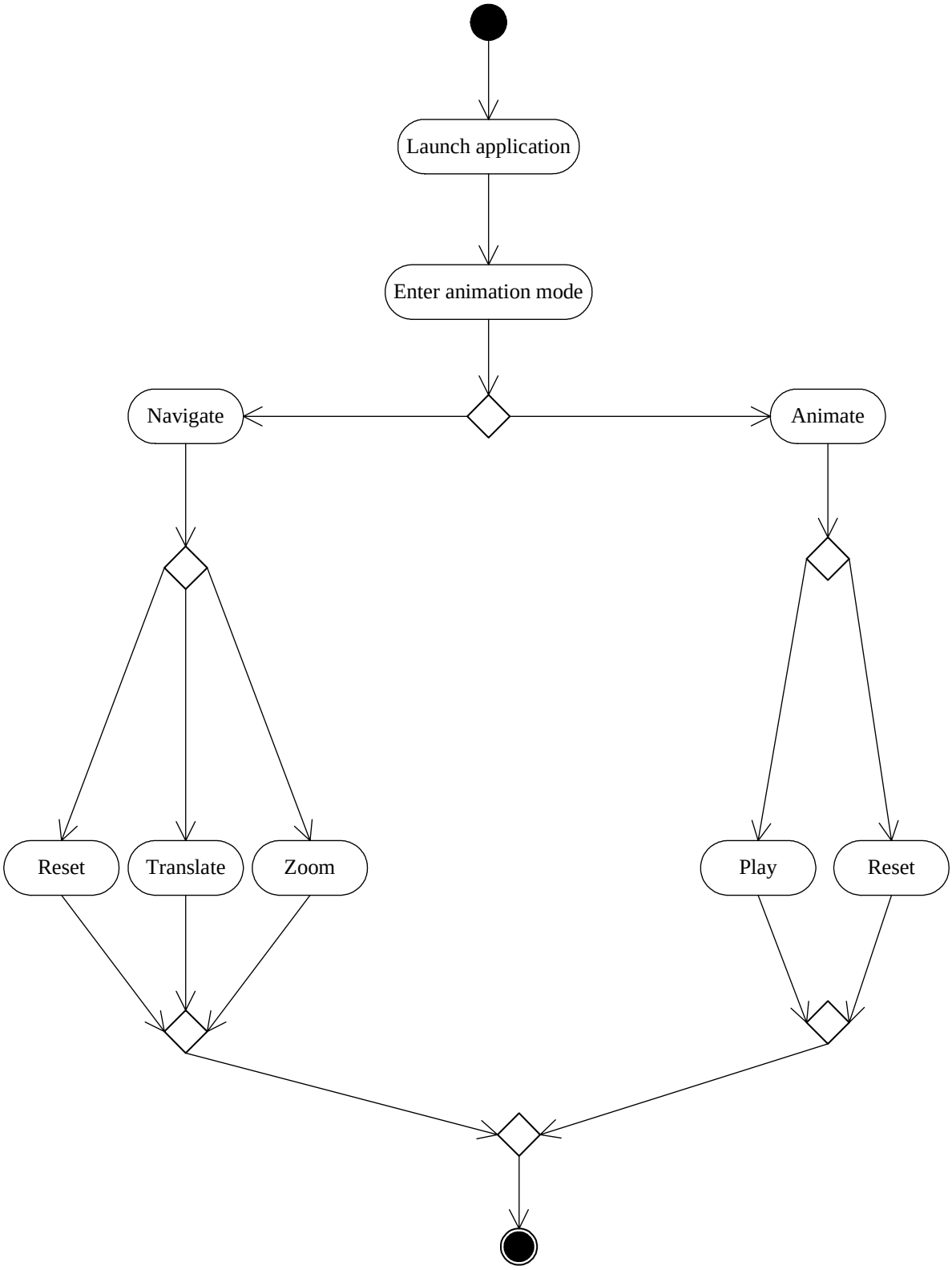
B.2 Class Diagram



B.3 State Diagram



B.4 Interaction Diagram



C. Coordinate conversion

Below are the variables needed for the transformations explained:

a	The earth's semi-major axis
f	The earth's flattening
e^2	First eccentricity squared
φ	Latitude position, north/south (GPS-coordinates)
λ	Longitude position, east/west (GPS-coordinates)
x	Latitude coordinates in meter
y	Longitude coordinates in meter
λ_0	Longitude of the central meridian
k_0	Scale factor along the central meridian
$\delta\lambda$	Difference in longitude $\lambda - \lambda_0$
FN	False northing (redefine the origin in the x-direction, in the new coordinate system)
FE	False easting (redefine the origin in the y-direction, in the new coordinate system)

All latitude and longitude values should be expressed in radians. One important observation is that both the latitude/longitude coordinate system and the x/y coordinate system have a positive axis in the north direction and in the east direction.

The value of the constants:

$$a = 6378137.0\ \text{m}$$

$$f = \frac{1}{298.257222101}$$

$$k_0 = 1.00000254$$

$$FN = -6226307.8640\ \text{m}$$

$$FE = 84182.8790\ \text{m}$$

$$\lambda_0 = 13\ \ 35\ \ 7.6920\ \text{deg min sec} = \left(13 + \frac{35}{60} + \frac{7.6920}{3600}\right) * \frac{pi}{180}\ \text{radians}$$

$$e^2 = f(2 - f)$$

$$n = \frac{f}{(2 - f)}$$

$$\hat{a} = \frac{a}{(1 + n)}\left(1 + \frac{1}{4}n^2 + \frac{1}{64}n^4 + \ldots\right)$$

Some variables contain series where only the necessary amount of terms has been used in order to get the desired precision. All longitude and latitude values should be expressed in radians which is why the necessary conversions have to be made.

The conformal latitude:

$\varphi^* = \varphi - \sin\varphi\cos\varphi\left(A + B\sin^2\varphi + C\sin^4\varphi + D\sin^6\varphi + ...\right)$ where

$A = e^2$

$B = \frac{1}{6}\left(5e^4 - e^6\right)$

$C = \frac{1}{120}\left(104e^6 - 45e^8 + ...\right)$

$D = \frac{1}{1260}\left(1237e^8 + ...\right)$

Define $\delta\lambda = \lambda - \lambda_0$ and

$\xi = \arctan\left(\frac{\tan\varphi^*}{\cos\delta\lambda}\right)$

$\eta = \operatorname{arctanh}\left(\cos\varphi^* * \sin\delta\lambda\right)$

$\beta_1 = \frac{1}{2}n - \frac{2}{3}n^2 + \frac{5}{16}n^3 + \frac{41}{180}n^4 + ...$

$\beta_2 = \frac{13}{48}n^2 - \frac{3}{5}n^3 + \frac{557}{1440}n^4 + ...$

$\beta_3 = \frac{61}{240}n^3 - \frac{103}{140}n^4 + ...$

$\beta_4 = \frac{49561}{161280}n^4 + ...$

The final *x* and *y*-positions are calculated by the following equations:

$$x = k_0\hat{a}\left(\begin{array}{l}\xi + \beta_1\sin 2\xi\cosh 2\eta + \beta_2\sin 4\xi\cosh 4\eta + \\ \beta_3\sin 6\xi\cosh 6\eta + \beta_4\sin 8\xi\cosh 8\eta + ...\end{array}\right) + FN$$

$$y = k_0\hat{a}\left(\begin{array}{l}\xi + \beta_1\cos 2\xi\sinh 2\eta + \beta_2\cos 4\xi\sinh 4\eta + \\ \beta_3\cos 6\xi\sinh 6\eta + \beta_4\cos 8\xi\sinh 8\eta + ...\end{array}\right) + FE$$

As mentioned above, only the necessary series are taken into account in the calculations. The result will be coordinates in meter measured from the equator in the *x*-direction and from the zero-meridian in the *y*-direction (Reit, 2001).

D. XML File

```
<Root>
     <Positions>
          <Source Type="Garmin eTrex">FOI_MSI</Source>
          <ObjectId Type="Name">FOI Linkoping</ObjectId>
          <Sample Date="2003-04-11" Time="08:06:00">
                 <Position Type="WGS84">
                 <Latitude CardinalPoint="North">58.63154</Latitude>
                 <Longitude CardinalPoint="East">15.3242</Longitude>
                 </Position>
          </Sample>
          <Sample Date="2003-04-11" Time="08:06:02">
                 <Position Type="WGS84">
                 <Latitude CardinalPoint="North">58.63154</Latitude>
                 <Longitude CardinalPoint="East">15.3241</Longitude>
                 </Position>
          </Sample>
          <Sample Date="2003-04-11" Time="08:06:04">
                 <Position Type="WGS84">
                 <Latitude CardinalPoint="North">58.63154</Latitude>
                 <Longitude CardinalPoint="East">15.3242</Longitude>
                 </Position>
          </Sample>
          <Sample Date="2003-04-11" Time="08:06:06">
                 <Position Type="WGS84">
                 <Latitude CardinalPoint="North">58.63159</Latitude>
                 <Longitude CardinalPoint="East">15.3241</Longitude>
                 </Position>
          </Sample>
          <Sample Date="2003-04-11" Time="08:06:08">
                 <Position Type="WGS84">
                 <Latitude CardinalPoint="North">58.63161</Latitude>
                 <Longitude CardinalPoint="East">15.3242</Longitude>
                 </Position>
          </Sample>
     </Positions>
</Root>
```