Institut für Technische Informatik und Kommunikationsnetze
Computer Engineering and Networks Laboratory

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

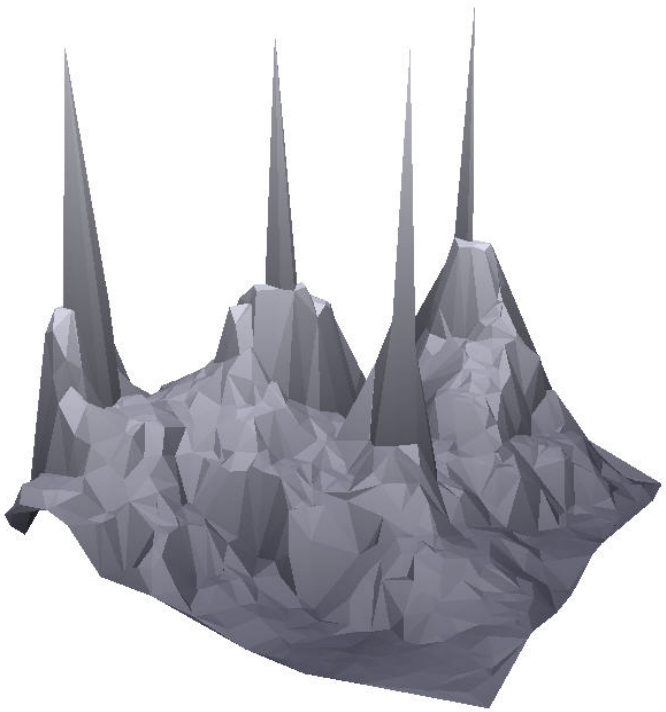
3D-Visualisierung von Field Based Routing

Semesterarbeit, SS2006

Niels Kistler, kistlern@student.ethz.ch

Betreuer: Rainer Baumann, baumann@tik.ee.ethz.ch
und Mario Strasser, strasser@tik.ee.ethz.ch
Professor: Prof. Dr. Bernhard Plattner, plattner@tik.ee.ethz.ch

10. Juli 2006



Abstract

Ziel dieser Semesterarbeit war die Entwicklung eines Software-Tools, welches die für Menschen wenig aussagekräftigen Simulationsdaten von Field Based Routing in eine animierte, grafisch ansprechende 3D-Grafik umsetzt.

Inhalt

1. Problemstellung **5**
2. Design **6**
2.1 Grundlagen 6
2.2 Ablauf 6
2.3 Kriterien 6
3. Implementierung **7**
3.1 Grafikbibliothek / Programmiersprache 7
3.2 Grafisches Userinterface 7
3.3 Algorithmus für die Oberflächenberechnung 7
3.4 Darstellung der Paket-Routen 8
3.5 Kontinuierliches Durchlaufen der Simulation 8
4. Handbuch **9**
4.1 Ausführen des Programms 9
4.2 Bedienung 10
4.2.1 Grundlegendes 10
4.2.2 Ablauf-Steuerung (unteres GUI-Fenster) 10
4.2.3 Aussehen und Positionierung des Objekts (rechtes GUI-Fenster) .. 11
4.2.4 Pakete 12
4.2.5 Details 13
4.3 Daten-Files 13
4.3.1 Allgemeines 13
4.3.2 Format 14
4.3.2.1 Knoten-Files 14
4.3.2.2 Paket-Files 15
4.4 Technische Details 15
4.5 Probleme 16
5. Zusammenfassung **17**
6. Weiterführende Arbeiten **17**
7. Referenzen **17**

1. Problemstellung

Field Based Routing ist ein neues Routingprotokoll für drahtlose Netzwerke, welches an der ETH entwickelt wurde. Es basiert auf dem Konzept eines Feldes, analog zu elektrostatischen Feldern in der Physik.

Wie soll nun ein solches Protokoll evaluiert werden?

Simulationen liefern Testdaten, die den Zustand des Feldes und die momentanen Positionen von Paketen beschreiben. Dies sieht konkret so aus:

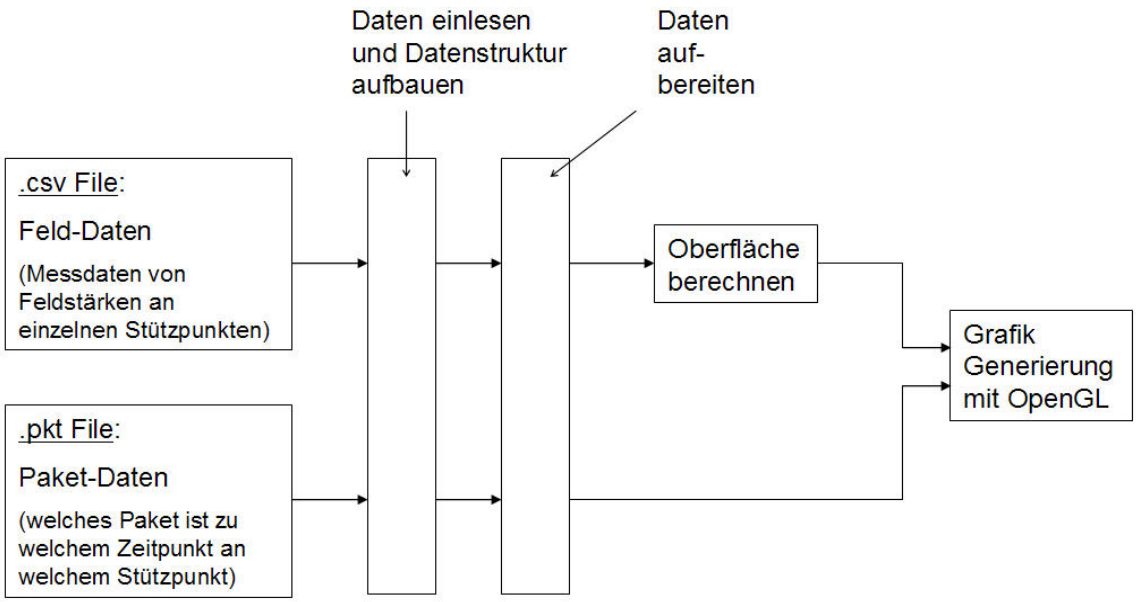
```
0;1249;1249;261;303;0;0;0;nan;0;0.000000000;1499;5.743295
1;0;0;0;0;1000;27069;0;nan;1000000;0.000000000;0;nan
2;612;0;0;0;1000;13158;0;nan;1000000;0.000000000;0;nan
3;0;0;0;0;1000;6338;0;nan;1000000;0.000000000;0;nan
4;721;0;0;0;1000;21935;0;nan;1000000;0.000000000;0;nan
5;177;0;0;0;1000;31665;0;nan;1000000;0.000000000;0;nan
6;0;0;0;0;1628;39859;12;nan;13745;817.767454491;0;nan
7;0;0;0;0;2149;35953;24;nan;4369;977.028087673;0;nan
8;14;0;0;0;1165;11401;7;nan;283;686.902816063;0;nan
9;0;0;0;0;1094;20163;19;nan;55516;964.761338720;0;nan
10;0;0;0;0;1679;32480;9;nan;9995;949.428916305;0;nan
11;0;0;0;0;1455;10839;12;nan;5637;999.232873482;0;nan
12;0;0;0;0;1000;24766;22;nan;0;0.000000000;0;nan
13;0;0;0;0;1000;9840;5;nan;0;0.000000000;0;nan
14;0;0;0;0;1514;11597;14;nan;14647;937.203293287;0;nan
15;0;0;0;0;2142;37859;19;nan;3053;999.449000649;0;nan
16;0;0;0;0;1000;6610;3;nan;0;0.000000000;0;nan
17;0;0;0;0;1000;12157;10;nan;0;0.000000000;0;nan
18;0;0;0;0;3621;54218;22;nan;11269;924.279193479;0;nan
19;0;0;0;0;1332;24162;10;nan;14718;951.374252574;0;nan
20;0;0;0;0;2405;26844;16;nan;2954;799.875209333;0;nan
…
```

Es sind riesige Daten-Files gefüllt mit Zahlenreihen. Diese für Computer gut lesbare Form gibt uns Menschen keine Informationen darüber, wie das Feld aussieht und wie sich die Pakete bewegen.

Ziel war also, solche Testdaten umzusetzen in eine animierte Grafik, die die räumliche Struktur des Feldes und die Veränderungen des Feldes sowie die Bewegungen der Pakete auf grafisch ansprechende Art sichtbar macht.

Eine Software, die das leistet, existiert unseres Wissens noch nicht.

2. Design



2.1 Grundlagen

Die Struktur des Feldes ist bestimmt durch die Feldstärken bei jedem Teilnehmer des Netzwerkes, in diesem Kontext Knoten genannt. Die Positionen der Knoten zusammen mit ihren Feldstärken werden hier als Stützstellen genommen, über die eine Oberfläche gelegt wird, dergestalt dass alle Stützstellen interpoliert werden.

Die Paket-Routen sind bestimmt durch eigene Daten-Files, in denen aufgelistet ist, welches Paket zu welchem Zeitpunkt an welchem Knoten ist.

2.2 Ablauf

Knoten-Dateien und Paket-Dateien werden zur Laufzeit vom Benutzer gewählt. Wird ein neues File angeklickt, werden die darin enthaltenen Daten eingelesen, in Datenstrukturen abgefüllt und vorverarbeitet (sortiert, verlinkt, etc.).

Beim Durchlaufen der Simulation werden zu jedem Zeitpunkt die dazugehörenden Daten interpoliert (siehe Abschnitt 3.5), aus den Knoten-Daten wird eine Oberfläche generiert und aus den Paket-Daten werden die Routen der zu diesem Zeitpunkt aktiven Pakete berechnet, und beides zusammen wird mit OpenGL gerendert.

2.3 Kriterien

Beim Durchlaufen der Simulation müssen in Zeiträumen von einem Zwanzigstel bis einem Vierzigstel einer Sekunde aus Millionen von Datensätzen die richtigen gefunden und interpoliert werden, und daraus eine Oberfläche und Paketrouten generiert und gerendert werden. Deshalb ist die wichtigste Anforderung an alle verwendeten Algorithmen und Datenstrukturen die höchst mögliche Effizienz.

3. Implementierung

In diesem Kapitel werden die wichtigsten implementierungs-technischen Gesichtspunkte der Software besprochen.

3.1 Grafikbibliothek / Programmiersprache

Zuerst musste entschieden werden, mit welcher Grafikbibliothek und in welcher Programmiersprache die Anwendung erstellt werden soll.

Die beiden am weitesten verbreiteten Grafikbibliotheken sind OpenGL und DirectX. Hier wurde OpenGL verwendet, da es frei erhältlich ist für praktisch alle gängigen Plattformen und bereits integriert ist in die meisten Betriebssysteme. Um die grösst mögliche Effizienz des Codes sicher zu stellen wurde in C++ programmiert, mit direkter Einbindung des OpenGL-API's. (Die wichtigste Alternative dazu ist die Verwendung von Java3D – welches aber für diese Anwendung zu langsam ist.)

3.2 Grafisches Userinterface

Hier war das Hauptproblem, dass OpenGL und seine Zusatzbibliotheken GLU und GLUT nur eine sehr einfach User-Interaktion ermöglichen - und keine GUI-Programmierung.

Zur Entscheidung stand:
1) Ein selbst programmiertes User Interface, das entsprechend wenig zu bieten hat und viel Programmierarbeit erfordert - dafür aber massgeschneidert ist und den geringst möglichen Overhead hat bei der Interaktion mit dem übrigen Programm.
2) Verwendung einer zusätzlichen Bibliothek speziell für GUI-Programmierung im Zusammenhang mit OpenGL-Programmen.
Dies ermöglicht wesentlich schnellere Programmierung mit wesentlich ansprechenderem Resulat - und dafür Einbussen an der Effizienz des Codes.

Die Wahl fiel schlussendlich auf PUI (Picoscopic User Interface) [1] - eine Zusatzbibliothek, die gut genug ist, um ein User Interface zu programmieren, welches für den Zweck dieser Simulation reicht, und gleichzeitig einfach und klein genug ist, so dass der Overhead bei der Ausführung zu verkraften ist.

3.3 Algorithmus für die Oberflächenberechnung

Die gelieferten Messwerte werden wie oben beschrieben als Stützpunkte einer Oberfläche interpretiert, die an jeder Stelle den wirklichen Wert des Feldes möglichst gut approximieren soll. Die Problemstellung ist analog zur Geländedarstellung, wo aus den Höhenmessungen an einzelnen Stützstellen die Oberflächenform angenähert werden soll.

Dafür hat sich die Delaunay-Triangulierung als geeignet erwiesen. Im Zweidimensionalen konstruiert sie ein Netz von Dreiecken aus der gegeben Punktemenge, so dass die Innenwinkel maximiert werden (bzw. so, dass jeder Umkreis eines Dreiecks keinen weiteren Punkt der Punktemenge enthält). Dies stellt sicher, dass die entstehenden Dreiecke so wenig wie möglich degenerieren. In 3D sind diese Eigenschaften nur noch teilweise gewährleistet - aber die Delaunay Triangulierung erweist sich auch da als beste Annäherung.

Für dieses Projekt wurde ein bestehender, bereits in C implementierter Delaunay-Algorithmus [2] abgeändert, so dass die Laufzeit in der Nähe des optimalen Wertes O(n*log(n)) liegt. Damit ist die Triangulierung (zusammen mit den zusätzlichen Berechnungen wie Interpolation der gegeben Daten, Normalen-Berechnungen für die Beleuchtung, etc.) schnell genug, um Simulationen mit 1000 Knoten in Echtzeit animieren zu können.

Als Datenstruktur für die Knoten wird ein Array von FRAME's verwendet, wobei jedes FRAME ein Array von dreidimensionalen Punkten ist, die einem Satz der Messdaten entsprechen. Die Daten werden beim Laden einer Datei gesamthaft eingelesen und gespeichert, um erstens den Ablauf der Simulation beliebig verändern zu können (an andere Stellen springen, rückwärts laufen), und zweitens keine Zeit zu verlieren mit Daten einlesen währen der Simulation.

Dadurch wird allerdings der Speicherbedarf zur Laufzeit entsprechend gross. Im Wesentlichen ergibt er sich zu:
Anzahl Knoten mal Anzahl Messungen mal 12 byte (je 4 für eine Koordinate).
Bei 1000 Knoten und 10'000 Messungen gibt das ca. 100MB. D.h. grössere Datenmengen werden am besten auf mehrere Files aufgeteilt.

3.4 Darstellung der Paket-Routen

Paketdaten-Files enthalten Tripel der Form:
Paket-ID, Zeitpunkt, Knotennummer.
Wenn ein Paket-File geladen wird, werden aus diesen Angaben alle Zeitintervalle extrahiert, die eindeutig den Weg eines Paketes von einem Knoten zu einem anderen beschreiben - und Intervalle, die erstens zum gleichen Paket gehören und zweitens von ihrer Zeitdifferenz her eindeutig als zu einer Route gehörend erkannt werden, werden untereinander verlinkt.

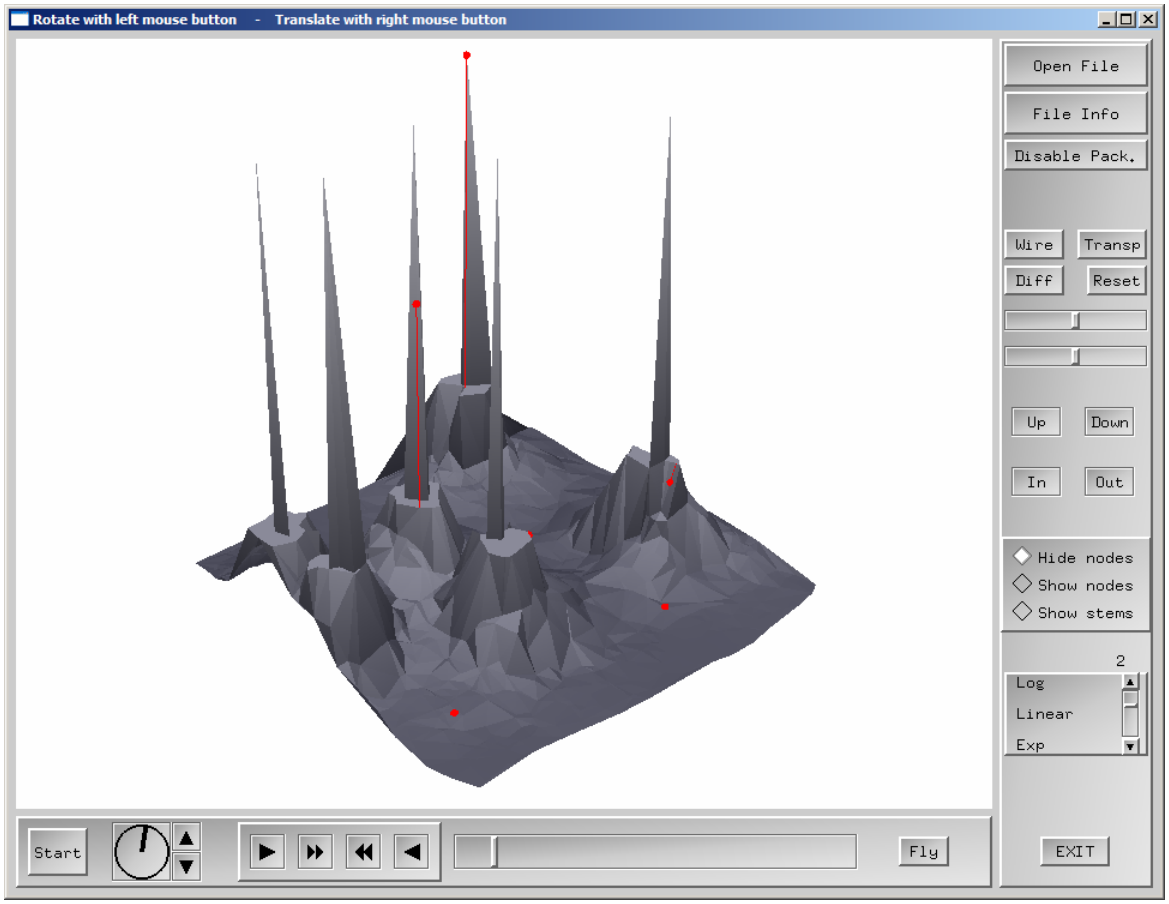
Beim Durchlaufen der Simulation wird zu jedem Zeitpunkt getestet, in welchen Paket-Zeitintervallen der jeweilige Zeitpunkt liegt, sprich welche Pakete gerade unterwegs sind, und von all diesen wird mittels der Verlinkung die ganze bereits zurückgelegte Route eruiert.

Als Datenstruktur dient ein Array von Pointern auf INTERVAL, wobei letztere Strukturen sind, die alle nötigen Informationen zu einem Intervall enthalten. Noch wenn das Paket-File geladen wird, werden die extrahierten Intervalle lexikographisch nach ihren Anfangs- und Endzeiten sortiert, um ein möglichst effizientes Durchlaufen und Durchsuchen der Intervalle in Vorwärts-Richtung zu ermöglichen.

3.5 Kontinuierliches Durchlaufen der Simulation

Die Messungen erfolgen typischerweise in Abständen von einer oder mehreren Sekunden. Wenn sie direkt nacheinander dargestellt werden, ergibt sich dadurch ein ruckartiger Ablauf der ganzen Simulation im Zeitraffertempo. Um einen kontinuierlichen Ablauf der Simulation zu ermöglichen, werden die Koordinaten und Feldstärken der einzelnen Knoten (sowie die Positionen der Pakete) zwischen aufeinanderfolgenden Messungen linear interpoliert. Das Tempo der Animation lässt sich dadurch stufenlos und beliebig verändern. (Lineare Interpolation wurde gewählt, weil sie einerseits am schnellsten zum Rechnen ist und andererseits der tatsächlichen Bewegung der Knoten in der Simulation am nächsten kommt.)

4. Handbuch



Das Handbuch enthält alle für die Benutzung der Software wichtigen Informationen.

4.1 Ausführen des Programms

Das Programm ist auf WindowsXP erstellt worden und für Windows geschrieben. Es basiert auf OpenGL, welches bereits in jeder neueren Windows-Version integriert ist, und benötigt die Hilfsbibliothek GLUT. Diese kann entweder installiert werden oder am gleichen Ort gespeichert werden wie die exe-Datei dieses Programms.

Die Daten der Simulation werden zur Laufzeit vom Benutzer ausgewählt und geladen. Das Fenster zum auswählen der Datei zeigt per Default das Verzeichnis an, in welchem sich die exe-Datei dieses Programms befindet. Dateien mit Knoten-Daten können beliebig geladen werden (das Laden einer neuen Datei löscht alle Informationen der zuvor geladenen Datei). Dateien mit Paket-Daten können nur geladen werden, wenn zuerst die entsprechende Knoten-Datei geladen wurde.

4.2 Bedienung

4.2.1 Grundlegendes

Nach starten des Programms erscheint das Hauptfenster, unterteilt in ein grosses Fenster für die Grafik und zwei kleinere Fenster für das User Interface.

Über den 'Open File' Button kann eine Datei ausgewählt werden, die dadurch sogleich ausgelesen und vorverarbeitet wird. Knoten-Files enthalten in ihrer Header-Zeile Angaben über die Grösse der Datei – dadurch kann das Fortschreiten des Ladevorgangs angezeigt werden. Dies ist bei Paket-Files nicht der Fall (bedingt durch die Entstehung der Daten in der verwendeten Netzwerk-Simulation).

Der 'File Info' Button zeigt die Namen der geladenen Dateien an, sowie die relevanten Informationen aus der Header-Zeile des Knoten-Files:
- Anzahl Knoten
- Simulationsdauer
- Zeitintervall zwischen den Messungen

Das angezeigte Objekt kann jederzeit mit der linken Maustaste gedreht und mit der rechten Maustaste verschoben werden.

Alle Pfeiltasten im unteren GUI-Fenster sowie die vier Knöpfe Up, Down, In und Out im rechten GUI-Fenster können entweder nur angetippt werden oder gedrückt gehalten werden für kontinuierliche Veränderung.

4.2.2 Ablauf-Steuerung (unteres GUI-Fenster)

Der Schieber zeigt die Position innerhalb des ganzen Ablaufs an. Durch klicken auf den Balken kann zu jeder Zeit an jede Stelle gesprungen werden. Der Schieber selbst kann ebenfalls mit der Maus gezogen werden.

Die Doppelpfeile vor und zurück ermöglichen das manuelle Traversieren aller originalen, sprich nicht interpolierten, Datensätze.

Die einfachen Pfeile vor und zurück ermöglichen das manuelle Traversieren der interpolierten Datensätze - mit variabler Geschwindigkeit.

Die Geschwindigkeit wird bestimmt durch die Anzahl Interpolationsschritte zwischen zwei aufeinanderfolgenden originalen Datensätzen.
Maximalgeschwindigkeit = keine Interpolation - d.h. es werden wie bei den Doppelpfeilen die originalen Datensätze nacheinander dargestellt.

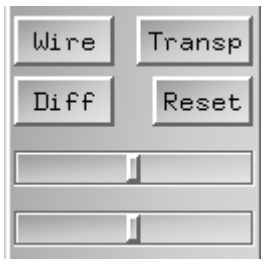
Die Geschwindigkeit wird angezeigt auf der Geschwindigkeitsuhr rechts neben dem Start-Knopf:
- Strich oben (= Ausgangsstellung) bedeutet Geschwindigkeit Null
- je weiter im Uhrzeigersinn, desto höher die Geschwindigkeit vorwärts
- je weiter im Gegenuhrzeigersinn, desto höher die Geschwindigkeit rückwärts
- zuunterst ist das Maximum.

Steuern lässt sich die Geschwindigkeit direkt mit der Uhr (am gewünschten Ort draufklicken oder Zeiger mit der Maus ziehen) - oder feiner mit den beiden Pfeiltasten rechts daneben:
- Pfeil nach unten: Geschwindigkeit erhöhen
- Pfeil nach oben: bremsen
(Die Pfeilrichtungen sind so gewählt, dass sie zur Bewegungsrichtung des Zeigers auf der Geschwindigkeitsuhr passen.)

Der Start-Knopf ganz links startet (bzw. stoppt) das Durchlaufen der Simulation, sprich das automatische Traversieren der interpolierten Datensätze mit der eingestellten Geschwindigkeit.

Der Fly-Knopf ganz rechts startet (bzw. stoppt) das Fliegen der Kamera um das Objekt herum. Dies vermittelt einen realistischeren 3D-Eindruck des Objektes.

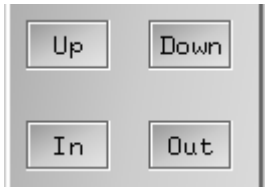
4.2.3 Aussehen und Positionierung des Objekts (rechtes GUI-Fenster):



Mit dem Wire-Knopf kann zwischen normalem und Wireframe-Modus umgeschaltet werden. Dies ermöglicht insbesondere zusammen mit dem Show-nodes-Modus, die Positionen der Knoten gut zu sehen.

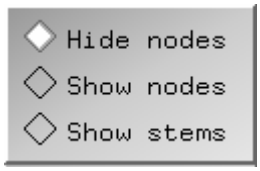
Der Transp-Knopf schaltet um zwischen normalem und semi-transparentem Modus.

Der Diff-Knopf schaltet um auf nur diffuses Licht - bzw. wieder zurück zu spekulärem Licht. Dies ist sinnvoll wenn Teile des Objektes überblendet werden durch zu starke Reflektion (was insbesondere bei hoch gelegenen, horizontalen Flächen geschieht). Mit den beiden Schiebereglern kann dann die Position der Lichtquelle in zwei zueinander senkrechten Richtungen verändert werden, um eine Beleuchtung zu finden, die die Beschaffenheit des Objekts möglichst gut zeigt. Mit Reset wird zurück auf die Default-Position der Lichtquelle geschaltet.



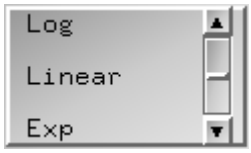
Up und Down kippen das Objekt hinauf bzw. hinunter (bis zur Horizontalen).

In und Out zoomen hinein bzw. heraus. Dies kann bei Anschluss einer externen Maus auch mit der mittleren Maustaste erreicht werden: Taste gedrückt halten und Maus bewegen.



Default-Modus ist 'Hide nodes', in dem nur die Oberfläche sichtbar ist, die durch die Knoten bestimmt ist. 'Show nodes' zeigt die Knoten als schwarze Kügelchen. 'Show stems' zeigt die Knoten sowie die Verbindungsgeraden vom Nullniveau bis zu den Knoten.

Die Knoten und die Verbindungslinien zu rendern braucht wesentlich mehr Zeit als nur die Oberfläche darzustellen - und verlangsamt dadurch das Durchlaufen der Simulation deutlich.



Per default werden alle Koordinaten linear gestaucht, um die Distanzverhältnisse zu erhalten. Die Höhenkoordinaten können aber auch logarithmisch oder exponentiell skaliert werden, um ein anderes Höhenrelief zu erzeugen.

In jede Richtung sind 5 Stufen möglich (mehr erschien nicht sinnvoll). Die Zahl oberhalb des Schiebers zeigt an, auf welcher Stufe von -5 bis 5 man sich befindet (Null = Default = lineare Stauchung).

Beim Verändern der Stufe werden sämtliche z-Koordinaten umgerechnet - d.h. bei grösseren Datenmengen kann das längere Zeit dauern.

Aus GUI-technischen Gründen war es nicht möglich, die Pfeiltasten so zu programmieren (wie die anderen Pfeiltasten), dass sie gedrückt gehalten werden können für kontinuierliche Wirkung. Aber für Springen zwischen mehreren Stufen auf ein Mal kann einfach an der gewünschten Stelle auf die Fläche des Schiebereglers geklickt werden.

4.2.4 Pakete

Wenn eine Paket-Datei eingelesen wird, wechselt das Program in den Paket-Modus, in dem die Pakete und ihre Routen dargestellt werden und die Bewegungskontrolle eine etwas andere Bedeutung hat. Es kann aber jederzeit hin- und hergeschaltet werden zwischen normalem und Paket-Modus mit dem Knopf 'Enable Packets' bzw. 'Disable Pack.' oben im rechten GUI-Fenster.

Im Paket-Modus erhält das Durchlaufen der Simulation eine semantisch andere Bedeutung: Es werden nicht mehr einfach die Datensätze des Knoten-Files durchlaufen, sondern es werden die Zeitintervalle durchlaufen, in denen Pakete aktiv sind.

Beim Laden eines Paket-Files bzw. beim Wechseln in den Paket-Modus springt der Ablauf zum ersten Zeitintervall, in dem ein Paket sichtbar ist, und mit allen Bewegungskontrollmöglichkeiten im unteren GUI-Fenster bewegt man sich von da an vor oder zurück, schnell oder langsam, immer in diesen Paket-Zeitintervallen. D.h. wenn das Ende eines Intervalls erreicht wird, springt der Ablauf zum nächsten. Dies ist notwendig, da die Zeitintervalle, in denen Pakete aktiv sind, derart kurz sind, dass man sie sonst gar nicht findet.

Die Geschwindigkeit des Ablaufs wird gleich gesteuert wie im normalen Modus, mit folgenden Unterschieden: Die Zeitschritte sind wesentlich kleiner, da die Pakete sich wesentlich schneller bewegen als die Knoten. Um dem Rechnung zu tragen und die Geschwindigkeit sinnvoll darstellen zu können, ist die Geschwindigkeitsuhr im Paket-Modus entsprechend anders skaliert.

Die Doppelfeiltasten durchlaufen im Paket-Modus nicht mehr die nicht-interpolierten Datensätze (siehe 4.2.2: Ablauf-Steuerung) sondern durchlaufen die Paket-Zeitintervalle einfach etwas schneller.

Um im grösseren Rahmen in der Paket-Simulation zu navigieren dient der Schieberegler. Mit ihm kann an eine beliebige Stelle geklickt werden und der Ablauf springt zum ersten Zeitintervall, welches diesen Zeitwert enthält – bzw. in das zeitlich nächstgelegene Zeitintervall, wenn der angeklickte Zeitpunkt in keinem Intervall liegt.

Die Paket-Routen werden nur angezeigt beim Durchlaufen der Intervalle in Vorwärts-Richtung.

4.2.5 Details

- Das Drehen des Objekts mit der linken Maustaste ist so programmiert, wie wenn die Maus am Rand einer horizontalen Scheibe greifen würde und diese (zusammen mit dem Objekt) um ihre vertikale Achse drehen würde. D.h. die Rotation funktioniert umso besser, je weiter aussen die Scheibe angepackt wird.

- Die GUI-Info-Boxen, die erscheinen wenn der 'Open File'- oder der 'File Info'-Knopf gedrückt werden, oder wenn eine Fehlermeldung angezeigt wird, blockieren den Rest des User Interfaces - es können dann also nur noch die Knöpfe im erschienenen Fenster bedient werden. Das Beenden des Programms ist deshalb zusätzlich mit der Escape-Taste möglich, die immer funktioniert.

Ebenfalls mit der Tastatur (Taste 's') lässt sich das Durchlaufen der Simulation starten und stoppen - genau gleich wie mit dem Start-Knopf ganz links. Dies deshalb, weil diese Funktionalität unter Umständen oft nacheinander gebraucht wird - und es mühsam ist, immer mit der Maus auf den entsprechenden Knopf zu gehen.

4.3 Daten-Files

4.3.1 Allgemeines

Die Eingabe-Daten sind getrennt in Files für Knoten-Daten und Files für Paket-Daten. Beide müssen textbasierte CSV-Files sein, mit allen Werten auf einer Zeile durch Strichpunkte getrennt. Knoten-Files müssen auf .csv und Paket-Files auf .pkt enden. Das Programm verarbeitet nur Files mit einer dieser beiden Endungen.

Wenn beim Einlesen einer Datei eine Zeile nicht dem unten angegebenen Format entspricht, wird der Vorgang abgebrochen und eine Fehlermeldung ausgegeben mit der Nummer der Zeile, auf welcher das Lesen der Datei fehlgeschlagen ist.

4.3.2 Format

4.3.2.1 Knoten-Files

1. Zeile (= Header):
Anzahl Knoten (integer)
Maximale x-Koordinate (integer)
Maximale y-Koordinate (integer)
Simulationsdauer (float)
Zeitintervall zwischen zwei Messungen (float)

Alle folgenden Zeilen:
Feldstärke (integer)
x-Koordinate (float)
y-Koordinate (float)

Von den Datentypen her also:
int;int;int;float;float
int;float;float
int;float;float
...

Knoten, die nicht dargestellt werden sollen, haben negative Feldstärke.

X- und y-Koordinaten müssen zwischen Null und den angegebenen Maximalwerten liegen.

Es folgen alle Datensätze für eine Messung nacheinander, und dann direkt anschliessend alle Datensätze für die nächste Messung, usw., wobei gleiche Knoten jeweils an gleicher Stelle innerhalb eines Datensatzes stehen (was für die Interpolation zwischen den Datensätzen entscheidend ist).

Beispiel:
File mit 1000 Knoten, Koordinaten-Maximalwerten von 7000, Simulationsdauer von 100 Sekunden, und Zeitintervall zwischen den Messungen von einer Sekunde:

```
1000;7000;7000;100.000000000;1.000000000
-1;5889.90;4469.74
0;6437.76;3761.98
0;1122.38;4160.15
0;4443.76;4555.71
0;1733.23;484.52
0;6591.68;2500.67
0;4500.17;2921.95
0;6375.48;2567.74
0;6693.28;5074.07
0;496.38;3174.35
[weitere 990 Zeilen für den ersten Datensatz]
[weitere 1000 Zeilen für den zweiten Datensatz]
…
```

4.3.2.2 Paket-Files

Alle Zeilen sind identisch und enthalten drei Werte:
Paket-ID (integer)
Zeitpunkt (float)
Knotennummer(integer)

Von den Datentypen her also:
int;float;int

Einträge mit gleicher Paket-ID müssen aufeinanderfolgen, sofern sie zur gleichen Route gehören, und der Zeit nach aufsteigend geordnet sein.

Beispiel:

```
0;77.5038081680;533
0;77.5129821160;73
0;77.5157302420;486
0;77.5183163970;500
613;428.0763860000;600
613;428.0785608550;956
613;428.0813110560;697
613;428.0842837490;2
100;157.0763860000;0
100;157.0764027080;5
100;157.0828545800;417
…
```

4.4 Technische Details

- Um die grösstmögliche Effizienz sicher zu stellen, wird überall mit dem Datentyp float gerechnet.

- Die Koordinaten der Knoten (x, y, und Intensität) werden auf Werte zwischen Null und Eins gestaucht. Dadurch wird die ganze Szene auf den Einheitswürfel reduziert, was eine einheitliche und wesentlich einfachere Darstellung des Objekts ermöglicht.

- Die verwendete Delaunay-Triangulierung erzeugt umso entartetere Dreiecke, je grösser die Neigung der Oberfläche ist - und bei Knoten, deren x- und y-Koordinaten zusammenfallen, deren z-Koordinaten aber verschieden sind, versagt sie ganz. Deshalb werden zuerst alle Knoten eliminiert, die zu nahe bei einem anderen Knoten liegen oder in ihren x- und y-Koordinaten mit einem anderen zusammenfallen. (Schwellwert ist ein Millionstel vom (tatsächlich vorkommenden) maximalen Koordinatenwert in x- oder y-Richtung.)

Dies stellt für die resultierende Grafik kein Problem dar, solange das Programm für seinen eigentlichen Zweck verwendet wird - denn die Feldstärken von Field Based Routing können sich gar nicht so schnell so stark ändern, dass ein Problem entsteht.
Wird das Programm aber für Daten verwendet, die in ihrer Topologie senkrechte oder fast senkrechte Flächen enthält, wird die Darstellung unter Umständen nicht korrekt sein.

4.5 Probleme

Je nach Computer, auf dem das Programm läuft, kann es zu verschiedenen störenden Effekten kommen – bedingt durch die jeweilige OpenGL-Implementation und andere systembedingte Parameter wie CPU-Leistung, Leistung der Grafikkarte, Betriebssystem-Version, Bildschirmauflösung, etc.

Aufgefallen sind folgende beiden Effekte:
- Auf dem IBM ThinkPad X31, auf dem die Software erstellt wurde, kommt es z.T. vor, dass bei Vergrösserung des Fensters das Programm nur noch verzögert reagiert - nach mehrmaligem Wechsel der Fenstergrösse aber wieder normal.
- Auf dem IBM ThinkPad T34p, auf dem das Programm getestet wurde, beginnt das Bild beim Wireframe- plus Fly-Modus zu flimmern - und ebenso bei Vollgrösse des Fensters und Fly-Modus (wohl bedingt durch die hohe Auflösung des Bildschirms und die dadurch bedingte Mehrarbeit beim Rendern der Grafik).

Ein Computer-unabhängiges Problem besteht, wenn die Simulation im halb-durchsichtigen Zustand des Objekts abläuft - dann kommt es zu einem stark störenden Flimmern der durchscheinenden Teile des Bildes (da die per-Pixel-Farbberechnung in OpenGL davon abhängt, welche Fläche zuerst gerendert wird - und diese Reihenfolge sich von Bild zu Bild ändert).

5. Zusammenfassung

Ausgangspunkt waren Daten aus der Simulation von Field Based Routing, die zwar prinzipiell alle Informationen enthalten, die aus der Simulation gewonnen werden sollten, aber in einer für Menschen unbrauchbaren Form.

Dafür wurde ein Software-Tool entwickelt, dass solche Daten grafisch umsetzt, so dass die darin enthaltenen Informationen für Menschen sichtbar werden.

Dies dient dazu, Field Based Routing besser verstehen und analysieren zu können – und auch vor Kunden präsentieren zu können.

6. Weiterführende Arbeiten

Das entwickelte Visualisierungs-Tool ist massgeschneidert auf Field Based Routing. Es könnte weiterentwickelt werden in eine Software, die auch andere Routing Protokolle visualisieren kann.

7. Referenzen

[1] http://plib.sourceforge.net/pui/
[2] http://astronomy.swin.edu.au/~pbourke/modelling/triangulate/