Eidgenössische Technische Hochschule Zürich

INSTITUTE of CARTOGRAPHY

Vertiefungsblock K7
Wintersemester 2000/2001

3-D-Visualisierung der SOLA-Stafette im Internet

Technischer Bericht von
Andreas Meister

Leitung: Prof. Dr. L. Hurni
Betreuung: A. Terribilini

Vorwort

Diese Arbeit entstand im Rahmen eines Vertiefungsblocks am Institut für Kartographie. Auftraggeber war allerdings der ASVZ, der jährlich die SOLA-Stafette organisiert. Ziel war es, ein dreidimensionales, interaktives Modell der einzelnen Laufstrecken zu erstellen, welches dann aufs Internet geladen wird. Es ist bei Semesterarbeiten selten der Fall, dass sie eine reelle Anwendung finden: Meistens wird die Aufgabe bloss als Übung angesehen. Diese kann durchaus höchst interessant sein und auch ein grosses Entwicklungspotential aufzeigen. Viel herausfordernder jedoch ist es, wenn nach getaner Arbeit Leute das Resultat beliebig ansehen und benutzen können.

Der Reiz, während dieses Semesters möglichst weit zu kommen, damit das Resultat einwandfrei ist, war demzufolge gross - der Aufwand, da ich die Arbeit alleine zu bewältigen hatte ebenfalls.

Dieser Bericht gibt einen Überblick über die Leistung die ich während des Semesters erbrachte. Ich empfehle aber natürlich einen Besuch der Homepage, um das Resultat auch konkret mal anzuschauen und ausprobieren zu können.

Ich möchte an dieser Stelle noch all jenen danken, die mir während des Semesters geholfen und mich unterstützt haben. Dazu gehören insbesondere:

- Prof. Dr. L. Hurni für die Leitung des Vertiefungsblockes;
- Thomas Mörgeli für die Koordination mit dem ASVZ;
- Andrea Terribilini für die stetige Unterstützung und die vielen guten Tips;
- Andreas Neumann, Barbara Schneider und die anderen Assistenten am IKA.

Zürich, den 8. Februar 2001

Der Autor, Andreas Meister

Zusammenfassung

Ziel dieser Arbeit war das Erstellen eines dreidimensionalen Modells für die Strecken der SOLA-Stafette. Dieses Modell war für das Internet bestimmt, was die Datenmenge beschränkt. Trotz dieser Einschränkung sollte das Modell möglichst realistisch aussehen, damit es für den Anwender ein attraktives Produkt darstellt.

Nachdem die Grundlagen der verwendeten Programmiersprache - VRML - erlernt und die nötigen Daten gesammelt waren, konnte mit der Modellierung begonnen werden. Die vielen Vektordaten wurden thematisch unterteilt: Gelände, Strassen, Gebäude, Einzelobjekte. Der grösste Arbeitsaufwand forderte die Bearbeitung der Daten. Diese mussten generalisiert, bereinigt und transformiert werden, bevor sie in VRML umgewandelt wurden.

Um eine virtuelle Realität entstehen zu lassen, reichen geometrische Daten alleine nicht. Deshalb wurden die VRML-Dateien mit weitere Angaben ergänzt, wie z.B. die Beleuchtung und die Hintergrundfarbe.

Das so entstandene Modell wurde in eine Homepage eingebunden, welche neben der virtuellen Welt noch zusätzliche Informationen wie eine Legende beinhaltet.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

1.1 Situation

Der ASVZ, der Organisator der SOLA-Stafette, erneuert momentan das Erscheinungsbild rund um die SOLA-Stafette. Rechtzeitig zur SOLA 2001 ist bereits eine neue Broschüre herausgekommen, und für das folgende Jahr ist ein Internetauftritt mit einer 3D-Visualisierung der Laufstrecken geplant. Nach Absprache mit dem IKA entstand der Entscheid, die Realisierung dieses Projekts als Vertiefungsblock anzubieten. Die SOLA-Stafette ist in 14 Teilstrecken unterteilt, und findet im Grossraum Zürich statt. Dies bringt den grossen Vorteil mit sich, dass für dieses Gebiet schon viele Daten in Vektorform vorhanden sind.

1.2 Aufgabenstellung

Die Aufgabenstellung wurde folgendermassen ausgeschrieben:

"3D-Visualisierung der SOLA-Laufstrecken im Internet

Der Akademische Sportverband Zürich (ASVZ) organisiert jeden Frühling im Grossraum Zürich die SOLA-Stafette, die sich aus 14 Teilstrecken zwischen 4 und 15 km zusammensetzt. Für die SOLA 2001 wird geplant, dass die Läufer/innen im Vorfeld der Stafette die einzelnen Streckenpläne im Internet abrufen können. Ziel dieser Arbeit ist die 3D-Modellierung dieser Strecken im WWW mittels VRML oder ähnlicher Techniken. Weiter soll es möglich sein, die Strecken virtuell abzulaufen. Voraussetzungen: Kenntnisse im Umgang mit WWW-Techniken, VRML, HTML, Java-Script; resp. Zeit und Wille sich in diese Techniken einzuarbeiten.

Kontakt: A. Neumann, C. Oberholzer, B. Schneider, T. Mörgeli (ASVZ)".

1.3 Zielsetzung

1.3.1 Ziele

In der Aufgabenstellung stand "3D-Visualisierung der SOLA-Streckenabschnitte". Es war aber ziemlich schnell klar, dass die Modellierung aller Strecken den Rahmen des Vertiefungsblockes sprengen würde. Nach Absprache mit der Assistenz und Herrn Mörgeli wurde vereinbart, dass nur eine Strecke quasi als Prototyp während des Semesters erarbeitet werde.

Die Ziele dieser Arbeit wurden wie folgt formuliert :

- Es soll ein realitätstreues Modell eines Streckenabschnittes entstehen - realitätstreu, damit es für den Anwender möglichst attraktiv ist. Es soll erkennbar sein, ob man sich im Wald, auf offenem Land oder in Siedlungsgebiet befindet. Auch die Strassenbeschaffenheit (Teer oder Naturstrasse) soll erkennbar sein.

- Das Modell soll in eine Homepage eingebunden werden, wo es dann auf dem Internet zugänglich und abrufbar ist.

1.3.2 Randbedingungen

Eine Randbedingung bei der Realisierung ist einerseits die beschränkte Datenmenge. Da das Modell von den Benutzern heruntergeladen werden muss, erschien es von Anfang an sinnvoll, die Grösse der Dateien auf ein erträgliches Mass zu begrenzen. Andererseits wurde der Umfang dieser Arbeit durch den Zeitrahmen - zwölf Wochenstunden während eines Semesters - beschränkt.

Es lässt sich erahnen, dass sich die Forderung nach einer grossen Realitätstreue, sprich Detailreichheit, nur schwer mit einer möglichst geringen Datenmenge vereinbaren lässt. Es musste also stetig überlegt werden, wo man grosszügiger generalisieren konnte um Speicherplatz zu sparen und welche Informationen ausgelassen werden konnten.

Es soll an dieser Stelle noch erwähnt werden, dass es sich beim Resultat um ein kartographisches Werk handelt, in dem bewusst Sachen ausgelassen und andere generalisiert wurden, und nicht um eine photorealistische Show, welche sicher auch sehr schön wäre, aber nicht dieser Arbeit entspricht.

1.4 Geforderte Resultate

Es wird erwartet, dass am Ende des Semesters ein Modell besteht, das den Zielsetzungen entspricht. Zudem müssen die Arbeiten in einer Präsentation vorgestellt werden. Ende des Semesters muss ein Bericht abgegeben, ein Plakat erstellt und eine CD gebrannt werden. Da es schon als Zielsetzung formuliert ist, wird auch eine Homepage erstellt.

2 Organisation

2.1 Terminplanung, Vorgang

Die Einführung in den Vertiefungsblock fand am 26. Oktober 2000 statt. In der folgenden Woche musste das vervollständigte Konzept sowie weitere Ideen abgegeben werden.

Danach waren jede Woche der Donnerstag sowie der Freitag Morgen für die Bearbeitung reserviert.

Am 7. Dezember 2000 fand eine Zwischenbesprechung statt und der 1. Februar 2001 war der Tag der Schlusspräsentation. Abgabetermin für den Bericht, das Poster und die CD ist der 8. Februar 2001.

Zusätzlich wurden noch zwei Besprechung mit dem Vertreter des ASVZ, Herrn Mörgeli, durchgeführt, um die Erwartungen des Sportverbandes zu klären.

Der Arbeitsvorgang wurde wie folgt gegliedert:

Zuerst mussten die VRML-Grundlagen erlernt werden. Dieser Schritt war erstens notwendig, um überhaupt ein Modell in dieser Sprache entwerfen zu können, und zweitens, um sich vorstellen zu können, was realisierbar ist und wie das Resultat aussehen kann. In einer zweiten Etappe wurden die nötigen Daten gesammelt. Diese mussten dann verarbeitet werden, wie dies im Abschnitt "Modellierung" ausführlich erläutert ist. In einem nächsten Schritt galt es, die verarbeiteten Daten in VRML umzuwandeln und alle nötigen Angaben zu schreiben, die für die Vervollständigung des Modells nötig sind. Schliesslich wurde die Homepage aufgebaut, die Präsentation vorbereitet und die abschliessenden Arbeiten - Bericht, Poster, CD - erstellt.

2.2 Infrastruktur

Zur Bearbeitung standen die Computer des IKA zur Verfügung, auf welchen die nötige Software installiert ist. Folgende Software wurde zu folgenden Zwecken verwendet:

- ESRI ArcInfo: Bearbeitung von GIS-Daten
- AutoCAD: Bearbeitung aller Vektordaten
- VRMLpad (© ParallelGraphics): Erarbeitung der WRL-files

- Netscape: Visualisierung der WRL-files sowie der Homepage
- Cosmo Player (© Cosmo™ Software) und Cortona VRML Client (© ParallelGraphics): Visualisierung der virtuellen Welten
- Adobe Photoshop: Bearbeitung von Bilddaten
- UltraEdit: Editieren der HTML- und CSS-files
- MS Excel: Berechnungen
- MS Word: Erstellen des Berichtes
- MS Powerpoint: Folienbearbeitung für die Präsentation
- Freehand: Erstellen des Posters
- Acrobat Distiller: Umwandlung des Berichtes in PDF-Format

Für nähere Angaben über diese Programme sei an dieser Stelle auf die entsprechende Literatur verwiesen.

Zusätzlich wurden noch einige Arbeiten am IGP ausgeführt, wie dies im Bericht näher erklärt wird. Weitere Arbeiten wurden in Zusammenarbeit mit Andrea Terribilini gemacht, der im Rahmen seiner Doktorarbeit verschiedene Programme geschrieben hat, die die Verarbeitung von Geodaten ermöglichen. Es war für ihn einfach, diese Programme umzuschreiben, um DXF- oder TIN-Dateien in das VRML-Format zu konvertieren.

3 Grundlagen

Dieses Kapitel basiert im Wesentlichen auf Dässler (1999).

3.1 Was ist VR?

VRML ist eine Modellierungssprache im Bereich der virtuellen Realität. Unter virtuellen Realität - kurz VR - versteht man computergestützte Konzepte, die unsere natürliche audiovisuelle und räumliche Wahrnehmung simulieren. Es gibt verschiedene Systeme, die verschiedene Immersionsgrade aufweisen. Im Idealfall ist der Benutzer vollständig davon überzeugt, dass die künstliche Umgebung real ist. Dieses Projekt beschränkt sich allerdings auf ein sogenanntes "Desktop-VR", welches 3D-Objekte auf einem normalen flachen Computermonitor darstellt.

3.2 Rendering

VRML-Browser benutzen verschieden Methoden, um räumliche Objekte auf einem flachen Bildschirm darzustellen. Wenn man sich im virtuellen Raum bewegt (was als Navigation bezeichnet wird), muss die gesamte Szene mit allen Objekten für jede einzelne Phase dieser Bewegung neu berechnet werden, und als einzelnes Bild - auch Frame genannt - auf dem Computerbildschirm angezeigt werden.

Um flüssige Bewegungen zu ermöglichen, sollte die Framerate, gemessen in Frames pro Sekunde (fps), bei über 30 fps liegen. Bei ungenügender CPU-Leistung besteht die Gefahr, dass die Szenen ruckartig verlaufen.

Die Grundlage jeder räumlichen Modellierung ist ein 3D-Koordinatensystem. Die Gestalt eines Objektes wird durch Punkte in diesem Koordinatensystem beschrieben. Verbindet man die Koordinatenpunkte miteinander, entsteht ein Drahtmodell. Um die Raumorientierung des Objektes eindeutig festzulegen, werden zwei einfache Verfahren angewendet: die Perspektive und die Verdeckung.

3.3 Die 3D-Grafikpipeline

Um aus dem Programmcode eine virtuelle Welt entstehen zu lassen, durchlaufen die Daten eine so genannte 3D-Grafikpipeline. Diese besteht aus zwei Komponenten: einem Geometrieteil und der eigentlichen Rendermaschine.

Der Geometrieteil besteht aus folgenden Operationen:

- Translation
- Transformation: Algorithmen zur Änderung der Objektposition und -lage, sowie zur Anpassung der Objektgrösse
- Beleuchtung
- Tesselation: Zerlegung komplexer Polygone in reguläre Formen (Dreiecke)

Renderingoperationen:

- Schattierung
- Z-Buffer: Überprüfung, ob ein Objekt durch ein anderes verdeckt wird
- Textur-Mapping: Verdeckung der Oberflächen mit Bilder
- Mip-Mapping: Speicherung mehrerer Versionen einer Textur, die sich in ihrem Detaillierungsgrad unterscheiden
- Alpha Blending: Kontrolle der Transparenz eines Objektes
- Antialiasing: Verfahren zur Glättung von Kanten und zur Vermeidung unerwünschter Treppenmuster.

3.4 Was ist VRML?

VRML - das Kürzel steht für Virtual Reality Modelling Language - ist keine Programmiersprache im herkömmlichen Sinn, sondern eine Beschreibungssprache, mit der man definiert, wie dreidimensionale Objekte auszusehen haben, wie sie in einer virtuellen Welt angeordnet sind, und welche Wechselwirkung sie miteinander haben. Der aktuelle Standard VRML97 ermöglicht darüber hinaus die Steuerung zeitlicher Abläufe in dieser Welt. Aus statischen Anordnungen von Objekten werden so dynamische Welten mit Interaktion und Animation.

Abbildung 1 - VRML-Logo

Zur Darstellung von VRML-Welten benötigt man neben einem Webbrowser ein VRML-Plug-In. VRML ist wie HTML plattformunabhängig.

3.5 VRML - Die Geschichte

Die Geschichte von VRML beginnt im Jahr 1994. Das Ziel war die Realisierung virtueller Welten im Internet. Nachdem 1995 eine erste offizielle Version - VRML 1.0 – erschienen ist, wurde 1996 bereits VRML 2.0 veröffentlicht. Mit geringen Änderungen wurde VRML 2.0 im September 1997 unter dem Namen VRML97 zum Industriestandard der International Standards Organisation (ISO) erklärt.

Heutzutage werden 3D-Webtechnologien vom Web3D-Konsortium weiterentwickelt. Als Standard für die nächste Generation von 3D-Applikationen ist eXtensible-3D (X3D) angekündigt.

3.6 Anwendungsbeispiele

Das Spektrum der Anwendungen reicht von virtuellen Einkaufszentren über Produktpräsentationen bis zu Architekturmodellen und virtuellen Landschaften. Zwei wichtige Anwendungsbereiche sind zum einen die Produktpräsentation und zum anderen die Geovisualisierung. Ein gutes Beispiel dafür ist Heiko Mundles Doktorarbeit über kartografische Online-3D-Plattform (2000). VRML ist demzufolge ein geeignetes Mittel, um die SOLA-Streckenabschnitte zu visualisieren.

3.7 Navigation

Zur Navigation in der virtuellen Welt bedarf es eines VRML-Browsers. Diese werden als Plug-Ins für die Standard-Webbrowser angeboten. Sie können gratis vom Netz heruntergeladen werden.

Wenn eine virtuelle Welt geladen ist, gibt es drei grundsätzliche Bewegungsmöglichkeiten. Der Anwender kann gehen, fliegen, oder Objekte untersuchen. Im ersten Fall (Walk) bewegt sich der Anwender ausschliesslich auf der Oberfläche, auf die er ständig durch eine simulierte Schwerkraft hinabgezogen wird. Im Fly-Modus kann sich der User beliebig auch in die dritte Dimension bewegen. Im Examine-Modus bewegt sich der Anwender nicht selbst, sondern kann diejenigen Objekte bewegen, die sich vor ihm befinden. Die Kollisionserkennung verhindert das unbeabsichtigte Eindringen in ein Objekt.

3.8 Syntax

Auf eine Art Handbuch, um die Syntax von VRML zu verstehen, wird hier verzichtet. Es gibt genügend Literatur, auf die an dieser Stelle zurückgegriffen werden kann, wie z.B. „Floppy's VRML Guide", das auf dem Internet verfügbar ist.

Es werden lediglich einige Begriffe erklärt, welche für das weitere Verständnis notwendig sind.

- LOD (Level Of Detail): Der LOD beschreibt ein Verfahren, das die Detailtreue eines Objektes in Abhängigkeit von der Entfernung zwischen Objekt und Beobachter definiert. Im LOD-Knoten werden einerseits die Objekte definiert, die dargestellt werden sollen, und andererseits die Entfernungswerte, bei denen von einem Objekt zum nächsten Objekt mit einem anderen Detaillierungsgrad gewechselt wird.

- IndexedFaceSet: Geometrieknoten zur Modellierung von Flächen. Im IndexedFaceSet sind alle Eckpunkte einer Fläche in einer Liste aufgeführt. In einer zweiten Liste werden diejenigen Punkte miteinander verbunden, welche jeweils eine ebene Teilfläche bilden.

- Inline: der Inline-Knoten ermöglicht das Zugreifen auf eine extern gespeicherte Datei. Auf diese Art ist es möglich, eine grosse Datei in mehrere kleineren aufzuteilen.

4 Modellierung

Als erster Schritt für den Modellaufbau galt es, eine Teilstrecke der SOLA-Stafette zu wählen. Es wurde die Strecke zwei berücksichtigt, da diese alle Elemente beinhaltet, die in den Vorbesprechungen als relevant erschienen. Die Strecke führt durch Wald, offene Landschaften sowie Siedlungsgebiet. Die Läufer müssen geteerte sowie Naturstrassen betreten. In der Mitte der Strecke befindet sich eine Verpflegungsstelle, wo sich die Läufer erfrischen können.

4.1 Gelände und Primärflächen

4.1.1 Kartenausschnitt

Nachdem eine Strecke gewählt wurde, musste ein passender Kartenausschnitt bestimmt werden, der die Grenzen des Modells definiert. Dieser wird durch die Koordinaten CH-675'000 im Westen, CH-681'000 im Osten, CH-244'000 im Süden und CH-253'000 im Norden begrenzt. Man hat sich auf Tausenderkoordinaten beschränkt, um sich die Zahlen besser merken zu können, da sie während der Datenverarbeitung in AutoCAD immer wieder gebraucht wurden.

Abbildung 2 - Kartenausschnitt mit der SOLA-Strecke zwei

Die Strecke führt von der ETH Hönggerberg (oben rechts, siehe Abb. 2) bis zur Sportanlage Buchlern in Altstetten (in der Mitte). Der Ausschnitt wurde noch weiter gegen Süden ausgedehnt, damit der Uetliberg ins Modell aufgenommen werden konnte (siehe roter Pfeil). Dieser prägt mit seinem 180 m hohen Turm die Umgebung von Zürich sehr stark. Daher wurde er in das Modell mit eingebaut.

Man sieht auf diesem Kartenausschnitt der Landeskarte sehr gut, dass ein grosser Teil dichtes Siedlungsgebiet ist. Jedes Objekt, sei es eine Strasse, ein Gebäude, ein Baum, usw., das man in das Modell hineinnimmt, vergrössert die Datenmenge. Es ist daher unmöglich, alle Gebäude und Strassen genau so, wie sie auf der Landeskarte 1:25'000 vorkommen zu übernehmen. Die SOLA-Strecke sollte möglichst genau modelliert werden. Das umliegende Gebiet hingegen benötigt nicht eine so hohe Detailtreue, und darf durchaus gröber generalisiert werden. Um dieses Ziel zu erreichen - grosse Detailliertheit entlang der Strecke, Generalisierung im übrigen Gebiet - wurde ein 500 m breiten Korridor entlang der Strecke ausgeschieden. Das Resultat dieser Operation ist in Abbildung 3 dargestellt.

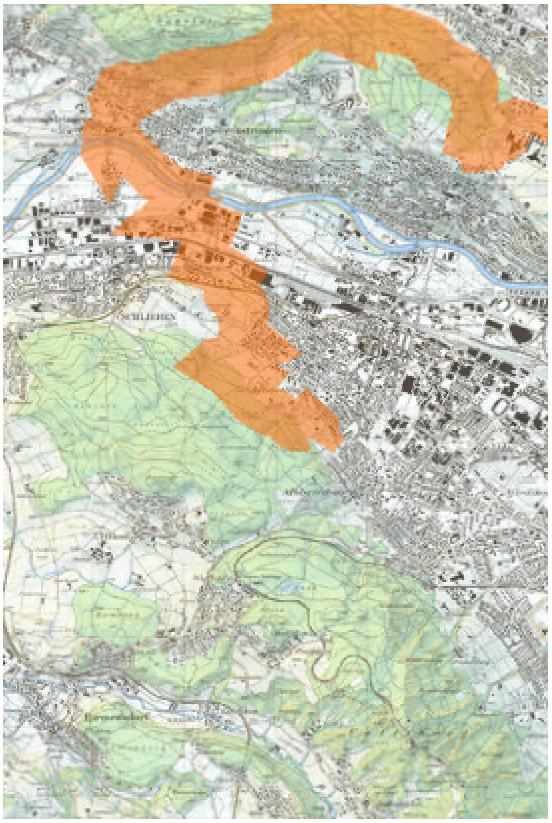

Abbildung 3 - Kartenausschnitt mit dem Streckenbereich

4.1.2 Gelände

Für das Gelände wurden die Daten aus dem DHM25 (© Landestopographie, Wabern) verwendet. Dieses wurde in ein TIN umgewandelt und so generalisiert, dass der maximale Höhenfehler 10 m beträgt.

4.1.3 Primärflächen

Für die Primärflächen wurden die Daten aus Vector25 (© Landestopographie, Wabern) verwendet. Auch hier war der Detaillierungsgrad grösser als gefordert. Deshalb wurden einerseits die Konturen vereinfacht, und zwar je stärker desto weiter weg sie sich von der Strecke befinden. Andererseits wurde die Anzahl der Klassen reduziert. Die 28 ursprünglichen Bodenbeschaffenheitsarten wurden auf sechs Klassen reduziert. Diese sind Siedlungsfläche, Wald, Wiese und Ackerland, Rebberg, Fluss und See.

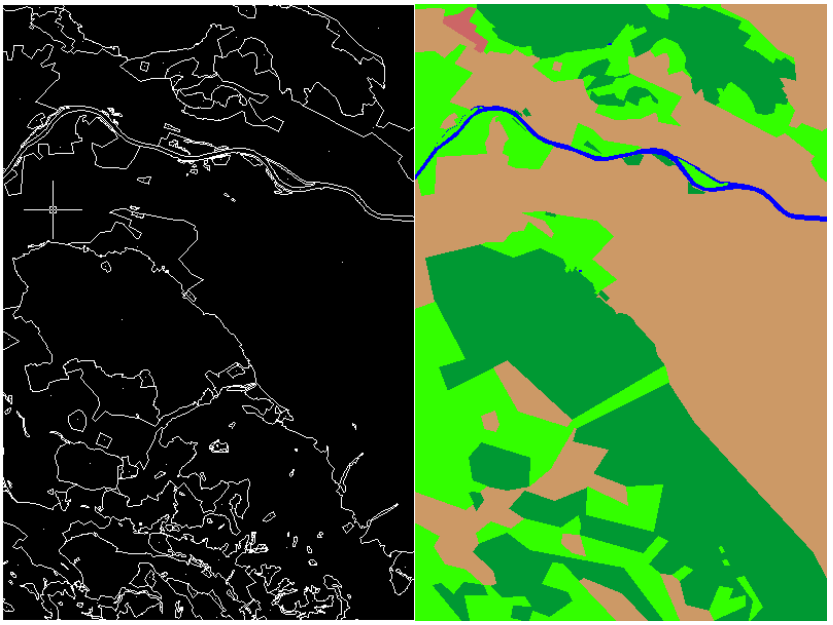

Abbildung 4 - Primärflächen, vor und nach der Bearbeitung

Die bearbeiteten Primärflächen wurden anschliessend mit dem Gelände verschnitten. Das DTM enthält nun als zusätzliche Information die Primärflächengrenzen. Als letzter Schritt wurde das DTM, das als TIN vorlag, in ein IndexedFaceSet im VRML-Format konvertiert.

Den Primärflächen wurden Farben zugeteilt, die nicht der Landeskarte sondern möglichst der Natur ähnlich sind.

4.2 Gebäude

Bei den Gebäuden galt es, wie schon angesprochen, die Aufteilung zwischen dem Streckenbereich (grosse Detailliertheit) und Fernbereich (starke Generalisierung) durchzuführen.

Für den Fernbereich wurden die Daten aus Vector200 (© Landestopographie, Wabern) verwendet. Diese erfüllen die Ziele sehr gut, da sie von weitem die Siedlungsstruktur gut zum Vorschein bringen, aber gleichzeitig stark generalisiert sind, so dass sie nur einen beschränkten Speicherplatz beanspruchen.

Für den Streckenbereich wäre es einfach gewesen, die Daten auf die selbe Art von Vector25 zu übernehmen. Leider existieren diese noch nicht. So musste nach einer anderen Lösung gesucht werden. Es wäre sicher möglich gewesen, die Gebäude in einem CAD-Programm mit Hilfe eines Hintergrundbildes (gescannte Landeskarte 1:25'000 (© Landestopographie, Wabern)) nachzuzeichnen. Dies ist aber eine nicht besonders rationelle Methode. Es war hingegen bekannt, dass im IGP Programme entwickelt wurden, welche Gebäude auf einer Pixelkarte erkennen und diese in Vektordaten umwandeln. Nach Absprache mit der dortigen Assistenz wurde diese Lösung bevorzugt.

Zuerst wurden mit dem Programm KAMU die schwarzen Flächen aus der Landeskarte 1:25'000 erkannt. Da die Einstellungen des Programms nur das Erkennen der Farbflächen erlaubten, mussten in einem zweiten Schritt mit Hilfe des Programms Ravis die fälschlicherweise erkannten Flächen (wie z.B. Buchstaben) wieder gelöscht und die nicht erkannten Gebäude hinzugefügt werden. Als die Gebäude im Programm mit denen auf der Landeskarte übereinstimmten, mussten sie noch vektorisiert werden. Dies geschah mit dem Programm RobVec.

Nach diesen Arbeiten waren alle Gebäude sowohl in Vector200 wie auch in Vector25 für den ganzen Ausschnitt vorhanden. Die überflüssigen Gebäude - aus Vector200 im Streckenbereich, aus Vector25 im restlichen Teil - mussten ausradiert werden. Beim Übergang zwischen den zwei Bereichen musste diese Operation mit grosser Sorgfalt ausgeführt werden, damit es keine Überlappungen oder unreelle Lücken gab. Des Weiteren musste auch überprüft werden, dass die Gebäude nicht in die Strassen hinausragten. War dies der Fall, musste das jeweilige Gebäude etwas zurückversetzt werden.

Anschliessend wurden die Eckkanten auf das Gelände projiziert. Von diesen Eckkanten wurden die höchste und die tiefste bestimmt. Bei der höchsten Ecke wurden 10 m hinzugerechnet, um die Hausoberkante zu erhalten. Bei der

untersten Ecke wurde nach unten eine gewisse Tiefe addiert, um sicher zu gehen, dass unter den Häusern keine Löcher entstehen.

Diese Häuserkanten wurden dann in VRML in ein Indexed-FaceSet umgewandelt.

4.3 Strassen

Die Strassen wurden wie die Gebäude in den Fern- und Streckenbereich aufgeteilt. Die Datenquellen waren auch hier Vector200 und Vector25.

Die vielen Layers mussten zusammengefügt - z.B. Strassen 1. Klasse aus Vector200 mit den Strassen 1. Klasse aus Vector25 - und bereinigt werden. Es gibt gewisse Klassen (wie z.B. Quartierstrassen), die als Solches keinen Sinn machten. In diesem Modell sind das Verkehrsaufkommen oder die rechtliche Lage nicht von Belangen. Es interessierten eigentlich einzig die Breite und der Belag der Strasse. Dazu wurde eine Unterteilung vorgenommen zwischen den Strassen, die zur SOLA-Strecke gehören, und jenen die nur zur Orientierung dienen.

Die schlussendliche Unterteilung in die verschiedenen Klassen ist in Tabelle 1 ersichtlich

Tabelle 1 - Verwendete Strassenklassen

Breite der Strasse	SOLA-Strecke		nicht-SOLA-Strecke		Eisenbahn
	geteert	Naturstrasse	geteert	Naturstrasse	
2 m	S-Klasse 7	S-Klasse 6	Fussweg	Klasse 6	-
4 m	S-Klasse 4	S-Klasse 5	Klasse 4	Klasse 5	-
6 m	S-Klasse 3	-	Klasse 3	-	-
8 m	S-Klasse 2	-	Klasse 2	-	-
10 m	S-Klasse 1	-	Klasse 1	-	Eisenbahn
20 m	-	-	Autobahn	-	-

Es resultierten 16 Klassen. Alle Sonderklassen wie Quartierstrassen oder Autobahnauffahrten wurden in andere bestehende Klassen integriert. Die Absonderung der Strassenabschnitte, die zur SOLA-Strecke gehören war notwendig, denn sie erhielten eine andere Farbe. Wären sie auf den selben Layers wie die übrigen Strassen, wäre dies nicht möglich.

In den Vector-Daten sind die Strassen lediglich durch ihre Achsen bestimmt. Im Modell will man aber die Strassen als Flächen darstellen. Um dies auf einfache Art zu erzielen, wurde ein kleines Programm geschrieben, welches für jeden

Abschnitt den Richtungsvektor bestimmt. Wenn man die Komponenten dieses Vektors vertauscht und bei einer Komponente noch das Vorzeichen wechselt, resultiert ein Normalvektor. Wenn man diesem Vektor nun die halbe Breite der Strasse als Länge gibt und die selbe Operation auf die andere Seite sowie am anderen Ende des Abschnittes durchführt, erhält man die vier Eckpunkte, die den Strassenabschnitt darstellen.

Die Höhe der Punkte wurde durch die Höhe der Achse bestimmt, damit die Strassenabschnitte flach bleiben. Zuerst wurden die Schnittpunkte einfach auf das Gelände projiziert. Bei einer Kuppe kam es dann vor, dass die Strasse unter dem Gelände verschwand. Um dies zu verhindern, mussten die Strassen mit dem Gelände verschnitten werden. Dadurch wurde die Datenmenge um einiges vergrössert, aber die Geometrie stimmte wenigstens.

Die Strassen wurden um einige Dezimeter erhöht, damit sie sich nicht mit dem Gelände verschneiden und teilweise darunter verschwinden.

Diese Art der Modellierung der Strassen ist relativ einfach und auch für das VRML sinnvoll, da die Geometrie nicht allzu kompliziert wird. Der Nachteil ist, dass sich bei Kurven zwischen den Abschnitten unschöne Lücken bilden.

Abbildung 5 - Beispiel von Lücken in Strassenkurven

4.4 Einzelobjekte

4.4.1 Markante Landschaftsobjekte

Einzelobjekte können zum Beispiel markante Landschaftsobjekte wie der Uetliberg-Turm sein (siehe Abbildung 6). Dieser wurde durch einen rot-weiss gestreiften Kegel modelliert. Die Plattform, die sich auf ca. 1/4 der Höhe befindet, wird durch einen flachen Zylinder dargestellt. In der VR fiel aber auf, dass die feine Spitze des Turmes bei Betrachtung von grosser Entfernung nicht immer sichtbar war, und teilweise sogar durch Verschwinden und Wiedererscheinen blinkte. Um diesem Problem entgegenzukommen wurde ein LOD eingeführt, in dem der Turm von Nahem wie schon beschrieben dargestellt wird, und von Weitem als Zylinder erscheint, um die feine Spitze zu eliminieren.

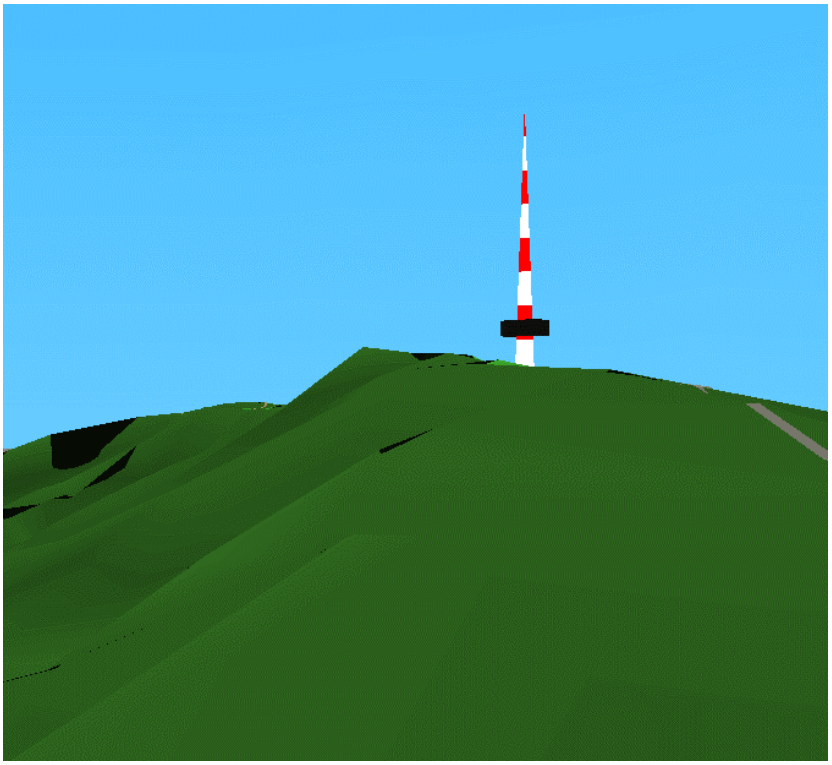

Abbildung 6 - Uetliberg-Turm, wie er im Modell aussieht

In der Hardau, einem Gebiet das sich noch auf dem Kartenausschnitt befindet, gibt es vier Hochhäuser, die ebenfalls von weitem sichtbar sind. So wurde entschieden, auch diese in das Modell zu nehmen. Da sie als Gebäude schon vorhanden waren, wurden sie einfach im Quellcode ausfindig gemacht und um 50 m erhöht.

4.4.2 Spezifische Objekte für die SOLA-Stafette

Es war vorgesehen, in das Modell auch Objekte einzubauen, welche für die Läufer von Belangen sind. In den Start- und Zielarealen sind dies insbesondere die Standorte, wo sie ihre Taschen hinlegen und wieder abholen können, sowie der Sanitätsposten. Die dazu nötigen Daten stammen aus der SOLA-Broschüre (ASVZ, 2000).

Vorgaben für das Aussehen dieser Objekte gab es keine. Für die Gepäckstandorte wurde eine Art Taschenhaufen aus einfachen Grundformen aufgebaut. Um das Rendern nicht stark zu beeinträchtigen, wurde auch hier ein LOD eingesetzt. Bei grösserer Entfernung erscheint der Taschenhaufen nur noch als Zylinder. Um sicher zu gehen, dass die Anwender auch verstehen, worum es sich handelt, wurde alles noch mit einem Textfeld beschriftet, welches sich immer in den Blickwinkel des Beobachters dreht.

Der Sanitätsposten auf dem Hönggerberg wurde durch ein Zelt modelliert, welches sich bei grösserer Entfernung in eine einfache Schachtel vereinfacht. In der Buchlern, wo sich der Sanitätsposten im Gebäude befindet, wurde vor dem Gebäude ein weisses Rechteck mit dem üblichen roten Kreuz hingestellt.

Des weiteren gibt es auf der Strecke zwei einen Verpflegungsposten. In Realität ist dieser ein Tisch, auf dem Wasserbecher stehen. Da dies in einem Modell kaum erkennbar wäre, wurde stattdessen ein überdimensionales Wasserglas mit einem Trinkhalm entworfen und beschriftet (siehe Abbildung 7).

Abbildung 7 - Wasserglas als Symbol für die Verpflegungsstelle

4.5 VRML-Angaben

VRML-Angaben sind notwendig, damit die virtuelle Welt überhaupt als solches wahrgenommen wird, und damit dem Anwender das Leben erleichtert wird.

Zu erwähnen ist die Beleuchtung. Dazu wurde eine Sonne modelliert, die die Eigenschaften ihrer reellen Schwester möglichst nachahmt. Wichtig ist auch der Himmel, der am Tag der SOLA möglichst blau wie im Modell sein soll.

4.5.1 Viewpoints

Als weitere Elemente wurden Viewpoints definiert. Viewpoints sind Kamerapositionen. So kann der Anwender seine Position mit einem Mausklick an einen vordefinierten Standort wechseln. Viewpoints wurden an Schlüsselpositionen der Strecke hingefügt. Diese sind der Start und das Ziel am Hönggerberg respektive an der Buchlern. Zusätzlich befindet sich noch einer an der Verpflegungsstelle. Als wietere Viewpoints wurden eine Aufsicht definiert, welche einen Überblick über das ganze Gebiet erlaubt, sowie eine panoramische Sicht von Süden her.

Der letzte Viewpoint ist nicht fest, sondern animiert, und ermöglicht einem Anwender, der mit der Navigation nicht gut zurecht kommt, die ganze Strecke wie in einem Film abzufliegen. Dazu wurde die Strecke in AutoCAD als Polylinie nachgezeichnet. In der Animation überfliegt der Anwender jeden Punkt der Polylinie. Damit man nicht immer in die selbe Richtung schaut, was in VRML als Standard gilt, musste zusätzlich für jeden Streckenabschnitt die Richtung berechnet werden. Diese wurde anschliessend in VRML als Orientierung der Animation hinzugefügt.

Dieser letzte Viewpoint wird erst gestartet, wenn der Anwender auf einen Knopf druckt, der neben dem Start platziert wurde.

4.5.2 Skripte

Um die ganze Anwendung interaktiv zu gestalten, bedarf es einiger Skripte, welche in Javascript geschrieben wurden. Eines dieser Skripte ermöglicht, dass man die Animation, wie sie oben beschrieben ist, überhaupt realisieren kann. Der Einsatz von Skripte ermöglicht insbesondere, von aussen her auf gewisse Knoten im VRML-Code zurückzugreifen und diese zu verändern.

Weitere Skripte sind direkt in der Homepage eingebaut.

5 Resultat, Anwendung

Wie dies in den Zielen formuliert wurde, soll die VR in eine Internetseite eingebaut werden, wo sie dann abgerufen und betrachtet werden kann.

5.1 Homepage

Die Internetseite, von der man die Strecke betrachten kann, ist in verschiedene Frames unterteilt (siehe Abbildung 8). Der obere Teil ist eine Art Titelleiste, in der steht um welche Strecke es sich handelt. Zusätzlich befindet sich dort auch ein Bezug auf die Hauptseite der SOLA-Stafette.

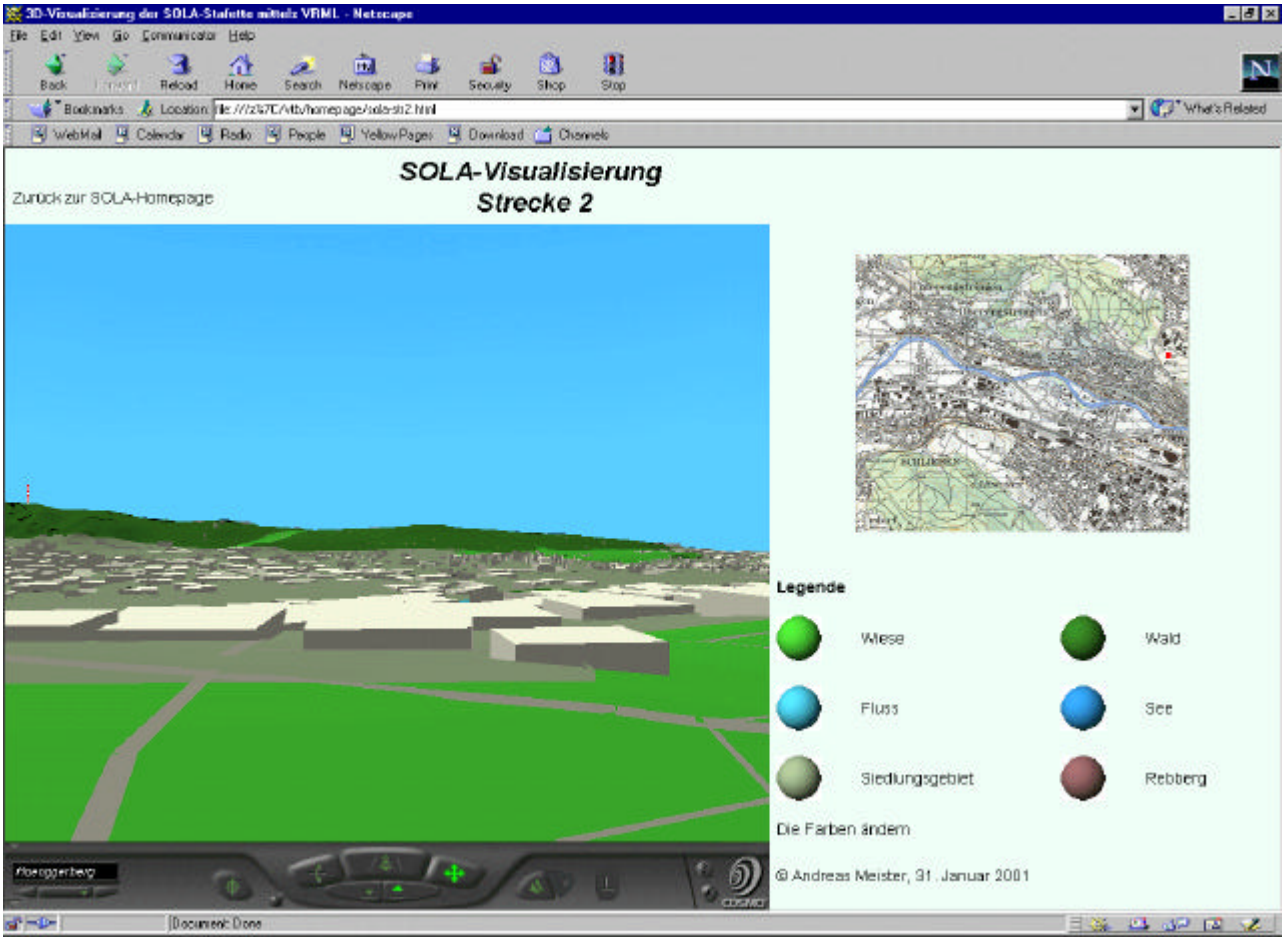

Abbildung 8 - Homepage

5.1.1 Plug-In

Der untere Teil ist in zwei vertikale Teile zerlegt. Im linken Teil wird das Plug-In gestartet, in dem die Navigation in der VR stattfindet. Bevor dies aber funktionieren kann, muss der Anwender zuerst das entsprechende Plug-In (Cosmo Player oder Cortona) heruntergeladen haben. Ein Hinweis dazu soll in einer vorgehenden Seite gegeben werden.

5.1.2 Kartenausschnitt

Rechts des Navigationsfensters befindet sich unter anderem ein Kartenausschnitt, auf dem ein roter Punkt die jeweilige Position des Beobachters wiedergibt. Diese Aktion wird durch ein Skript gesteuert, welches die Position aus dem Plug-In holt, um alle Sekunden den Punkt auf der Karte zu verschieben.

5.1.3 Legende

Im Weiteren befindet sich auf der rechten Bildschirmseite eine Legende. Da die Welt dreidimensional ist und die Farben je nach Neigung und Beleuchtung anders aussehen, wäre es sinnlos gewesen, in der Legende flache Farbflächen darstellen zu wollen. Diese hätten die Farben in der VR nicht wirklichkeitsgetreu widerspiegelt. Deshalb wurden für die verschiedenen Primärflächen statt kleine Farbfelder je ein kleines Plug-In definiert. Die Welt, die in diesen kleinen Plug-Ins dargestellt wird, besteht lediglich aus einer Farbkugel, die die gleichen Farbwerte wie die Flächen im Modell aufweisen. Die Beleuchtung wurde aus dem Modell übernommen, damit die Ähnlichkeit perfekt ist. Damit die Konsole des Plug-In's in der Legende nicht sichtbar ist, wurde sämtliche Navigation unterbunden.

Im Legendenteil befindet sich ein Link "Farbe ändern". Klickt man diesen an, öffnet sich ein neues Fenster, in dem man für jede Farbfläche die RGB-Werte ändern kann (siehe Abbildung 9). Bevor man die Farbe dem Modell zuordnet, kann man sie ansehen, und falls nötig wieder abändern. Wenn sie gesetzt wird, ändern sich die Farben für die gewählte Fläche sowohl im Modell als auch in der Legende. Diese Aktion wurde mit Javascript implementiert.

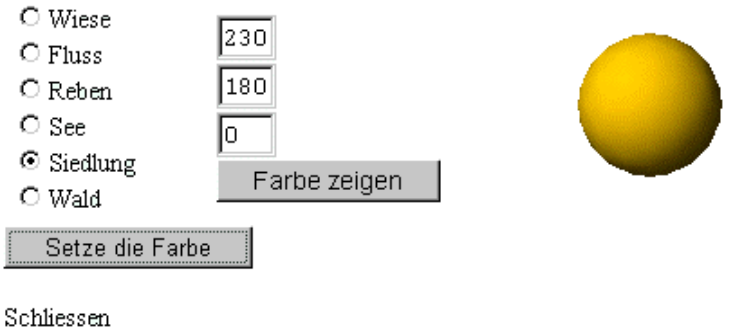

Abbildung 9 - Frame "Farbe ändern"

5.2 Internet

Zur Zeit befindet sich das Modell erst auf einer HTML-Seite, die zuerst noch auf das Netz geschaltet werden muss. Deshalb wird das Ganze auf den Server des IKA geladen, damit es von der Seite, wo die Studentenarbeiten aufgelistet sind, abgerufen werden kann. Ferner ist es vorgesehen, die Seite ebenfalls auf die Homepage der SOLA-Stafette zu laden.

5.3 Quellcode

Die Quellcodes zu den VRML- sowie HTML- und CSS-Dateien befinden sich im Anhang. Lange Koordinatenlisten und Ähnliches wurden gekürzt. Die vollständigen Dateien können aber mit einem Texteditor ab der CD geöffnet werden.

An dieser Stelle sei noch erwähnt, dass der gesamte VRML-Code nicht in einer Datei, sondern in mehreren abgespeichert ist. Im Plug-In führt der Link zur Hauptdatei, 'SOLA-Strecke2.wrl', welche die anderen Dateien mittels Inline-Knoten aufruft.

6 Verbesserungsmöglichkeiten und Nachführungsvorschläge

6.1 Verbesserungen

Einige Verbesserungsmöglichkeiten wurden schon erwähnt. Hier sind sie nochmals zusammengefasst und durch Weitere ergänzt.

6.1.1 Strassen

Bei den Strassen wurde das Problem der Lücken in den Kurven nicht gelöst. Es wäre durchaus möglich, eine Routine zu entwerfen, welche aus den Lücken kleine Dreiecke baut. Diese zusätzliche Geometrie würde zuviel Speicherplatz brauchen und zusätzlich das Rendern verlangsamen.

Bei den Hauptstrassen ist es für den Anwender nicht ersichtlich, ob ihm die ganze Strassenbreite zur Verfügung steht, oder ob er auf dem Trottoir laufen muss, welches übrigens nicht implementiert ist.

Ein weiterer Punkt, der verbesserungswürdig ist, sind die Strassenüber- und -unterquerungen. Die Unterführungen auf der Strecke zwei sind im Modell nicht vorhanden, da es sie im DHM nicht gibt. Um eine saubere Lösung zu erhalten, müsste man das Höhenmodell abändern. An Übergängen werden die Läufer von der Strassenpolizei über die Strassen geleitet, auf kleinen Wegen hingegen muss der Läufer selber auf den Verkehr aufpassen. Diese Unterschiede werden im Modell nicht berücksichtigt.

6.1.2 Gebäude

Die Gebäude im Modell sehen alle gleich aus. Sie haben dieselbe Farbe sowie dieselbe Höhe. Dies entspricht kaum der Realität, und sieht im Modell auch etwas eintönig aus. Es wäre auf relativ einfache Weise möglich, die Gebäude nach ihrer Höhe zu unterteilen: So wie auch die Strassen nach ihrer Breite unterteilt wurden, könnten dann auch diesen verschiedenen Gebäudeklassen unterschiedliche Farben zugeordnet werden.

6.1.3 Wald

Zum Wald wurde bis hier noch nicht viel gesagt, ausser dass er zu den sechs Primärflächen gehört. Als Solches unterscheidet er sich vom umgebenen Land durch nichts weiteres als seine Farbe. Dies ist leider eine nicht ganz befriedigende Lösung. Verschiedene Verbesserungsmöglichkeiten wurden in Betracht gezogen. Man könnte die Waldfläche extrudieren, aber dann würde man auf den Wald-

wegen wie zwischen zwei Wänden laufen. Es wäre auch möglich, den Wald zu bepflanzen, was jedoch viel Speicherplatz beanspruchen sowie eine erhöhte Rechenleistung nach sich ziehen würde. Die beste Lösung scheint hier den Gebrauch von LOD's zu sein. Man könnte entlang der Strecke Bäume setzten, die erst erscheinen, wenn der Anwender in die Nähe kommt. Dies würde zwar zusätzlichen Speicherlatz benötigen, aber das Rendern würde nicht zu stark beeinträchtigt.

Zusätzlich könnten allgemein mehr Bäume oder auch Hecken gepflanzt werden. Wie dies mit der Buchenreihe auf dem Hönggerberg (siehe Abbildung 10) ersichtlich ist, erhöht eine solche Zusatzinformation das Realitätswahrnehmen wesentlich.

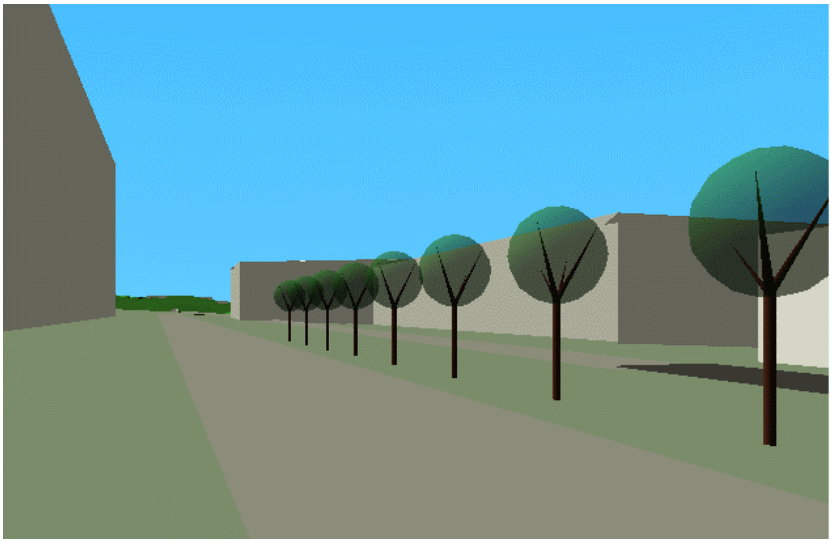

Abbildung 10 - Buchenreihe auf dem Hönggerberg

6.1.4 Datenstruktur

Die Daten sind zum Teil mehrfach vorhanden. Z.B. die Gepäckstandorte: Diese sind durch Taschenhaufen symbolisiert und in vier verschiedenen Dateien abgespeichert. Die ganze Geometrie ist somit ebenfalls viermal gespeichert. Das könnte man auf einfache Art korrigieren.

6.2 Ergänzungen

6.2.1 Animationen

Die Möglichkeiten, die VRML bietet, wurden in diesem Modell bei weitem nicht ausgereizt. Eine der grossen Stärken dieser Sprache sind die vielseitigen Animationsmöglichkeiten. Diese Wurden, abgesehen vom Flug über die Strecke, nicht eingesetzt. Man könnte sich vorstellen, dass auf den Strassen, welche die Läufer überqueren müssen, Verkehr fahren würde. An den Bushaltestellen nahe den Über-

gabestellen könnte man z.B. einen an- oder abfahrenden Bus simulieren. Mit etwas Phantasie könnte man hier noch vieles hinzufügen

6.2.2 Klänge

Eine gute VR sollte zusätzlich auch Klänge beinhalten. VRML unterstützt solche. Es wäre also durchaus machbar, den Lärm von Verkehr zu simulieren. Man könnte ebensogut im Wald Vögel zwitschern lassen oder im Zielgelände die Zuschauer klatschen lassen.

6.2.3 Interaktive Legende

Wie schon erklärt wurde, kann man in der Legende die Farbe der Primärflächen interaktiv verändern. Diese Möglichkeit könnte auf andere Elemente, insbesondere die Strassen, erweitert werden.

Zusätzlich zu den Farbänderungen könnte man von der Legende aus noch weitere Aktionen steuern, wie zum Beispiel das Setzen der Viewpoints. Von der Technik her wäre es möglich, quasi jedes einzelne Element sowie jede Eigenschaft dieser Elemente über Javascript zu steuern.

7 Schlusswort

Das Projekt einer 3D-Visualisierung im Internet begeisterte mich von Anfang an sehr. Zum Einen bin ich von Webtechnologien fasziniert. Ich denke, dass heutzutage Kenntnisse übers Internet unentbehrlich sind. VRML und 3D-Modellierungen waren für mich allerdings neu. So musste ich mich zuerst in diese Techniken einarbeiten, was ziemlich Geduld brauchte. Zum Anderen bin ich ein begeisterter Amateurläufer, und nahm in den letzten Jahren regelmässig an der SOLA-Stafette teil. Spannende Techniken mit einem tollen Thema kombiniert durften also nur ein gutes Resultat ergeben. Somit war für mich dieser Vertiefungsblock eine besondere Herausforderung.

Da ich den Aufwand und die Schwierigkeiten eher unterschätzt habe, musste ich mich am Anfang immer wieder motivieren und das Ziel im Auge behalten. Als das Resultat allmählich Form annahm, wurde es für mich immer spannender. Ich wollte dann immer Neues ausprobieren und das Modell verbessern. Der Ehrgeiz, ein einwandfreies Resultat vorführen zu können, war auch deshalb so gross, weil das Modell auf dem Internet abrufbar sein wird.

Es freut mich natürlich ausserordentlich, wenn es soweit ist. Ich hoffe, dass viele Läufer von der Möglichkeit Gebrauch machen werden, die Strecke virtuell abzulaufen, und dass sie Spass daran haben werden.

8 Verwendete Abkürzungen

ASVZ	Akademischer Sportverband Zürich
CAD	Computer Aided Design
CPU	Central processing unit
CSS	Cascading Style Sheet
DHM	Digitales Höhenmodell
DTM	Digitales Terrainmodell
DXF	CAD-Vektordatenformat (© Autodesk)
GIS	Geoinformationssystem
HTML	Hyper Text Markup Language
IGP	Institut für Geodäsie und Photogrammetrie
IKA	Institut für Kartographie
LOD	Level Of Detail
PDF	Portable Document Format (© Adobe)
TIN	Triangulated Irregular Net
VR	Virtuelle Realität
VRML	Virtual Reality Modelling Language
WRL	World (Kürzel für VRML-Dateien)
WWW	World Wide Web

9 Literaturverzeichnis

Akademischer Sportverband Zürich (2000)
SOLA 2001, Samstag 5. Mai, 28. SOLA-Stafette.
Informationsbroschüre. TA-Media AG, Zürich

Dässler, R. (1999)
VRML, Das Einsteigerseminar. bhv Verlag, Kaarst

Mundle, H. (2000)
Kartografische Online-3D-Plattform, Methana 3D - eine kartografische Online-3D-Plattform mit Datenbankanbindung der griechischen Halbinsel Methana.
Karlsruher Geowissenschaftliche Schriften A 13, Karlsruhe

Smith, J. (2000)
Floppy's VRML Guide.
http://www.vapourtech.com/vrmlguide/

10 Links

ASVZ: http://www.asvz.ch

IGP: http://www.gis.ethz.ch

IKA: http://www.karto.ethz.ch

Landestopographie: http://www.landestopographie.ch

SOLA-Stafette: http://www.sola.asvz.ethz.ch

Web3D-Konsortium: http://www.web3d.org

Inhaltsverzeichnis Anhang

A. VRML-Dateien

SOLA-Strecke2.wrl

```
#VRML V2.0 utf8
NavigationInfo {
     type["ANY"]
     headlight FALSE
     speed 10
}
DEF Materialeingenschaften Group {
     children [
          DEF Uebrig_mat Material { diffuseColor .298 .831 .239 }
          DEF Fluss_mat Material { diffuseColor .298 .8 1 }
          DEF Reben_mat Material { diffuseColor .58 .39 .44 }
          DEF See_mat Material { diffuseColor 0.2 0.6 1.0 }
          DEF Siedl_mat Material { diffuseColor .62 .71 .6 }
          DEF Wald_mat Material { diffuseColor .22 .49 .16 }
          DEF Fluss Appearance { material USE Fluss_mat }
          DEF Reben Appearance { material USE Reben_mat }
          DEF See Appearance { material USE See_mat }
          DEF Siedl Appearance { material USE Siedl_mat }
          DEF Uebrig Appearance { material USE Uebrig_mat }
          DEF Wald Appearance { material USE Wald_mat }
     ]
}
#Sonnenlicht
DirectionalLight {
     direction -0.3 -1.0 -0.5
     ambientIntensity 0.5
     color 1.0 1.0 0.9
     intensity 0.8
}
#Hintergrund
Background {
     skyColor [
          0.0 0.5 1.0
          0.2 0.7 1.0
          0.4 0.8 1.0
     ]
     skyAngle [ 0.82 1.6416 ]
     groundColor  [
          0.0 0.0 0.0,
          0.3 0.3 0.3,
          0.5 0.5 0.5,
     ]
     groundAngle  [ 0.75, 1.5 ]
}
#Position wird für das Pixel auf dem Kartenausschnitt gebraucht
DEF Position ProximitySensor {
     center 678000 2000 -248500
     size 100000 100000 100000
}
#Viewpoints
DEF Kameras Group {
     children [
          Viewpoint {
               position 678000 12000 -248500
               orientation 1 0 0 4.712
               description "Draufsicht"
          }
          Viewpoint {
               position 678000 2000 -240000
               orientation 0 1 0.3 0
               description "panorama-sued"
          }
          Viewpoint {
               position 680700 530 -251450
               orientation 0 1 0 -0.44
               description "Hoenggerberg"
               fieldOfView 1
          }
          Viewpoint {
```

```
            position 678419 470 -248229
            description "Buchlern"
            fieldOfView 1
        }
        Viewpoint {
            position 676975 400 -250831
            orientation 0 1 0 3
            description "Verpflegung"
            fieldOfView 1
        }
        DEF  Strecke2 Transform {
            children DEF Strecke2viewpoint Viewpoint {
                fieldOfView 0.7
            }
        }
    ]
}
DEF  PI_Strecke2 PositionInterpolator {
    key [ … ]
    keyValue [ … ]
}
DEF  OI_Strecke2 OrientationInterpolator {
    key [ …  ]
    keyValue [ … ]
}
DEF  Uhr_Strecke2 TimeSensor {
    cycleInterval 900
}
#Objekte
DEF  Objekte Group {
    children [
    #Start, Ziel, Verpflegung
        Inline { url "start.wrl" }
        Inline { url "ziel.wrl" }
        Inline { url "verpflegung.wrl" }
    #Gebäude
        DEF Gebaeude Transform {
            children Inline { url "sola_geb.wrl" }
        }
    #HIP-HIQ
        Inline { url "hip.wrl" }
        Inline { url "hiq.wrl" }
    #Uetlibergturm
        Inline { url "uetliberg-turm.wrl" }
    #Sanität Hönggerberg und Buchlern
        Inline { url "sani-hoebe.wrl" }
        Inline { url "sani-buchl.wrl" }
    #Gepäck Hönggerberg von Bucheggplatz
        Inline { url "t-hoebe-v.wrl" }
    #Gepäck Hönggerberg nach Buchlern
        Inline { url "t-hoebe-n.wrl" }
    #Gepäck Buchlern von Hönggerberg
        Inline { url "t-buchl-v.wrl" }
    #Gepäck Buchlern bleibt hier
        Inline { url "t-buchl-b.wrl" }
    #Wasserstelle
        Inline { url "wasser-2.wrl" }
    #Buchenreihe auf dem Hönggerberg
        Inline { url "buchen-reihe.wrl" }
    #Strassen
        Inline { url "str-1.wrl" }
        Inline { url "str-2.wrl" }
        Inline { url "str-3.wrl" }
        Inline { url "str-4.wrl" }
        Inline { url "str-5.wrl" }
        Inline { url "str-6.wrl" }
        Inline { url "str-auto.wrl" }
        Inline { url "str-bahn.wrl" }
        Inline { url "str-fuss.wrl" }
        Inline { url "str-s-1.wrl" }
        Inline { url "str-s-2.wrl" }
        Inline { url "str-s-3.wrl" }
        Inline { url "str-s-4.wrl" }
        Inline { url "str-s-5.wrl" }
        Inline { url "str-s-6.wrl" }
        Inline { url "str-s-7.wrl" }
    #Startknopf
        Transform {
```

```
            children [
                Shape {
                    appearance Appearance {
                        texture ImageTexture { url "startknopf.jpg" }
                    }
                    geometry Box { }
                },
                DEF Startbefehl TouchSensor { }
            ]
            translation 680732 524 -251528
        }
    ]
}
DEF tourScript Script {
    eventIn SFTime doTour
    eventIn SFBool isActive
    field SFBool boolValue TRUE
    eventOut SFBool viewBind
    eventOut SFTime startTime
    url "javascript:
    function doTour(value) {
        viewBind = true;
        startTime = value;
    }
    "
}
ROUTE Startbefehl.touchTime TO tourScript.doTour
ROUTE tourScript.viewBind TO Strecke2viewpoint.set_bind
ROUTE tourScript.startTime TO Uhr_Strecke2.startTime
ROUTE Uhr_Strecke2.fraction_changed TO PI_Strecke2.set_fraction
ROUTE PI_Strecke2.value_changed TO Strecke2.translation
ROUTE Uhr_Strecke2.fraction_changed TO OI_Strecke2.set_fraction
ROUTE OI_Strecke2.value_changed TO Strecke2.rotation
DEF coord_list Coordinate {
    point [ … ]
}
DEF TIN Transform {
    children [
        Shape {
            appearance USE Fluss
            geometry IndexedFaceSet {
                solid FALSE
                coord USE coord_list
                coordIndex [ … ]
            creaseAngle 3
            }
        }
        Shape {
            appearance USE Reben
            geometry IndexedFaceSet {
                solid FALSE
                coord USE coord_list
                coordIndex [ … ]
            creaseAngle 3
            }
        }
        Shape {
            appearance USE Uebrig
            geometry IndexedFaceSet {
                solid FALSE
                coord USE coord_list
                coordIndex [ … ]
            creaseAngle 3
            }
        }
        Shape {
            appearance USE See
            geometry IndexedFaceSet {
                solid FALSE
                coord USE coord_list
                coordIndex [ … ]
            creaseAngle 3
            }
        }
        Shape {
            appearance USE Siedl
            geometry IndexedFaceSet {
                solid FALSE
```

```
                coord USE coord_list
                coordIndex [ … ]
            creaseAngle 3
            }
        }
        Shape {
            appearance USE Wald
            geometry IndexedFaceSet {
                solid FALSE
                coord USE coord_list
                coordIndex [ … ]
            creaseAngle 3
            }
        }
    ]
}
```

start.wrl

```
#VRML V2.0 utf8
DEF Start Transform {
    children Shape {
        appearance Appearance {
            material Material { emissiveColor 1.0 1.0 1.0 }
        }
        geometry Text {
            string [ "Start" ]
            fontStyle FontStyle {
                style "SANS"
            }
        }
    }
    translation 680735 527 -251527
    rotation 0 1 0 -0.44
    scale 2 2 2
}
```

ziel.wrl

```
#VRML V2.0 utf8
DEF Start Transform {
    children Shape {
        appearance Appearance {
            material Material { emissiveColor 1.0 1.0 1.0 }
        }
        geometry Text {
            string [ "Ziel" ]
            fontStyle FontStyle { style "SANS" }
        }
    }
    translation 678415 451 -248223
    rotation 0 1 0 -3.77
}
```

verpflegung.wrl

```
#VRML V2.0 utf8
DEF sensor ProximitySensor {
    size 100 100 100
    center 676950 392 -250789
}
DEF verpflegung Transform {
    children Shape {
        appearance Appearance {
            material Material { emissiveColor 1 1 1 }
        }
        geometry Text {
            string [ "Verpflegung" ]
            fontStyle FontStyle {
                style "SANS"
                justify "MIDDLE"
            }

        }
    }
```

```
    translation 676950 400 -250789
    rotation 0 1 0 0.4
}
ROUTE sensor.orientation_changed TO verpflegung.rotation
```

sola_geb.wrl

```
#VRML V2.0 utf8
Shape {
    appearance DEF GebMat Appearance {
        material Material {
            diffuseColor 0.8 0.8 0.8
            ambientIntensity 1
            shininess 0.5
        }
    }
    geometry PointSet {
        coord Coordinate {
            point [ 680000 500 -250000 ]
        }
    }
}
Shape {
    appearance DEF GebMat2 Appearance {
        material Material {
            diffuseColor 0.6 0.4 0.0
        }
    }
    geometry PointSet {
        coord Coordinate {
            point [ 680000 500 -250000 ]
        }
    }
}
Shape {
    appearance USE GebMat2
    geometry IndexedFaceSet {
        solid FALSE
        coord Coordinate {
            point [
                680877.4 487.5 -248486.3,
                680903.1 487.5 -248470.9,
                680886.4 487.5 -248442.8,
                680860.7 487.5 -248458.2,
                680877.4 397.0 -248486.3,
                680903.1 397.0 -248470.9,
                680886.4 397.0 -248442.8,
                680860.7 397.0 -248458.2,
            ]
        }
        coordIndex [
            0, 1, 2, 3,0, -1,
            0, 1, 5, 4 ,0, -1,
            1, 2, 6, 5 ,1, -1,
            2, 3, 7, 6 ,2, -1,
            3, 0, 4, 7, 3, -1,
        ]
    }
},
```

An dieser Stelle folgen die ca. 1600 anderen Gebäude.

hip.wrl

```
#VRML V2.0 utf8
DEF HIP Transform {
    children [
        DEF GrundQuader Transform {
            children [
                Shape {
                    appearance DEF hipMat Appearance {
                        material Material {
                            diffuseColor 1 1 1
                            ambientIntensity 1
                        }
                    }
```

```
                    geometry Box { size 42 4 8 }
                }
            ]
        },
        DEF  Dach Transform {
            children [
                Shape {
                    appearance   USE hipMat
                    geometry IndexedFaceSet    {
                        solid FALSE
                        coord Coordinate {
                            point [
                                -21 0 4,      #0 vorne links
                                21 0 4,       #1 vorne rechts
                                21 0 -4,      #2 hinten rechts
                                -21 0 -4,     #3 hinten links
                                -21 4 4,      #4 oben links
                                21 4 4,       #5 oben rechts
                            ]
                        }
                        coordIndex [
                            0, 1, 5, 4, 0, -1,     #Fläche vorn
                            0, 3, 4, 0, -1,        #Dreieck links
                            1, 2, 5, 1, -1,        #Dreieck rechts
                            3, 4, 5, 2, 3, -1,     #Schrägfläche
                        ]
                    }
                }
            ]
            translation  0 2 0
        }
    ]
    translation  680716 524   -251536
    rotation 0 1 0 1.13
}
```

hiq.wrl

```
#VRML V2.0 utf8

DEF HIP Transform {
    children [
        DEF GrundQuader Transform {
            children [
                Shape {
                    appearance DEF hipMat Appearance {
                        material Material {
                            diffuseColor 1.0 1.0 1.0
                            ambientIntensity 1
                        }
                    }
                    geometry Box { size 42 4 8 }
                }
            ]
        },
        DEF Dach Transform {
            children [
                Shape {
                    appearance USE hipMat
                    geometry IndexedFaceSet {
                        solid FALSE
                        coord Coordinate {
                            point [
                                -21 0 4,      #0 vorne links
                                21 0 4,       #1 vorne rechts
                                21 0 -4,      #2 hinten rechts
                                -21 0 -4,     #3 hinten links
                                -21 4 4,      #4 oben links
                                21 4 4,       #5 oben rechts
                            ]
                        }
                        coordIndex [
                            0, 1, 5, 4, 0, -1,     #Fläche vorn
                            0, 3, 4, 0, -1,        #Dreieck links
                            1, 2, 5, 1, -1,        #Dreieck rechts
                            3, 4, 5, 2, 3, -1,     #Schrägfläche
                        ]
```

```
                    }
                }
            ]
            translation 0 2 0
        }
    ]
    translation 680696 524 -251546
    rotation 0 1 0 1.13
}
```

uetliberg-turm.wrl

```
#VRML V2.0 utf8
DEF Uetliberg-Turm Transform {
    children [
        LOD {
            level [
                Group {#Turm
                    children [
                        DEF Turm Transform {
                            children [
                                Shape {
                                    appearance Appearance {
                                        texture ImageTexture { url "uebt.gif" }
                                    }
                                    geometry Cone {
                                        bottomRadius 6.0
                                        height 180.0
                                    }
                                }
                            ]
                        },
                        #Plattform
                        DEF Plattform Transform {
                            children [
                                Shape {
                                    appearance Appearance {
                                        material Material {
                                            diffuseColor 0.5 0.5 0.5
                                        }
                                    }
                                    geometry Cylinder {
                                        radius 15.0
                                        height 10.0
                                    }
                                }
                            ]
                            translation 0.0 -60.0 0.0
                        }
                    ]
                },
                DEF  Turmweit Transform {
                    children Shape {
                        appearance Appearance {
                            texture ImageTexture { url "uebt.gif" }
                        }
                        geometry Cylinder {
                            radius 8
                            height 180
                        }
                    }
                }
            ]
            range [ 4000 ]
        }
    ]
    translation 679455 932 -245050
}
```

sani-hoebe.wrl

```
#VRML V2.0 utf8
DEF Sanitaet Transform {
    children [
        LOD {
            level [
```

```
Transform {
    children [
        #1. Seite
        Shape {
            appearance Appearance {
                texture ImageTexture { url "sani.gif" }
            }
            geometry IndexedFaceSet {
                solid FALSE
                coord Coordinate {
                    point [
                        0 0 0, #0
                        3 0 0, #1
                        3 0 3, #2
                        0 0 3, #3
                        0.5 2 0.5, #4
                        2.5 2 0.5, #5
                        2.5 2 2.5, #6
                        0.5 2 2.5, #7
                        1.5 2.5 1.5, #8
                    ]
                }
                coordIndex [
                    0, 1, 5, 4, 0, -1,
                ]
            }
        },
        #2. Seite
        Shape {
            appearance Appearance {
                texture ImageTexture { url "sani.gif" }
            }
            geometry IndexedFaceSet {
                solid FALSE
                coord Coordinate {
                    point [
                        0 0 0, #0
                        3 0 0, #1
                        3 0 3, #2
                        0 0 3, #3
                        0.5 2 0.5, #4
                        2.5 2 0.5, #5
                        2.5 2 2.5, #6
                        0.5 2 2.5, #7
                        1.5 2.5 1.5, #8
                    ]
                }
                coordIndex [
                    1, 2, 6, 5, 1, -1,
                ]
            }
        },
        #3. Seite
        Shape {
            appearance Appearance {
                texture ImageTexture { url "sani.gif" }
            }
            geometry IndexedFaceSet {
                solid FALSE
                coord Coordinate {
                    point [
                        0 0 0, #0
                        3 0 0, #1
                        3 0 3, #2
                        0 0 3, #3
                        0.5 2 0.5, #4
                        2.5 2 0.5, #5
                        2.5 2 2.5, #6
                        0.5 2 2.5, #7
                        1.5 2.5 1.5, #8
                    ]
                }
                coordIndex [
                    2, 3, 7, 6, 2, -1,
                ]
            }
        },
        #4. Seite
```

```
                        Shape {
                            appearance Appearance {
                                texture ImageTexture { url "sani.gif" }
                            }
                            geometry IndexedFaceSet {
                                solid FALSE
                                coord Coordinate {
                                    point [
                                        0 0 0, #0
                                        3 0 0, #1
                                        3 0 3, #2
                                        0 0 3, #3
                                        0.5 2 0.5, #4
                                        2.5 2 0.5, #5
                                        2.5 2 2.5, #6
                                        0.5 2 2.5, #7
                                        1.5 2.5 1.5, #8
                                    ]
                                }
                                coordIndex [
                                    3, 0, 4, 7, 3, -1,
                                ]
                            }
                        },
                        #Dach
                        Shape {
                            appearance Appearance {
                                material Material { diffuseColor 1.0 1.0 1.0 }
                            }
                            geometry IndexedFaceSet {
                                solid FALSE
                                coord Coordinate {
                                    point [
                                        0 0 0, #0
                                        3 0 0, #1
                                        3 0 3, #2
                                        0 0 3, #3
                                        0.5 2 0.5, #4
                                        2.5 2 0.5, #5
                                        2.5 2 2.5, #6
                                        0.5 2 2.5, #7
                                        1.5 2.5 1.5, #8
                                    ]
                                }
                                coordIndex [
                                    4, 5, 8, 4, -1,
                                    5, 6, 8, 5, -1,
                                    6, 7, 8, 6, -1,
                                    7, 4, 8, 7, -1,
                                ]
                            }
                        },
                    ]
                },
                Transform {
                    children [
                        Shape {
                            appearance Appearance {
                                texture ImageTexture { url "sani2.gif" }
                            }
                            geometry Box {
                                size 2.8 2.2 2.8
                            }
                        }
                    ]
                    translation  1.5 1.25 1.5
                }
            ]
            range [ 50 ]
        }
    ]
    translation 680756 522.5 -251590
}
```

sani-buchl.wrl

```
#VRML V2.0 utf8
DEF Sanitaet Transform {
    children Shape {
        appearance Appearance {
            texture ImageTexture { url "sani.gif" }
        }
        geometry IndexedFaceSet {
            solid FALSE
            coord Coordinate {
                point [
                    0 0 0,
                    2.6 0 0,
                    2.6 2.8 0,
                    0 2.8 0,
                ]
            }
            coordIndex [ 0, 1, 2, 3, 0, -1, ]
        }
    }
    translation 678435 446 -248302
    rotation 0 1 0 1.57
}
```

t-hoebe-v.wrl

```
#VRML V2.0 utf8
DEF Sensor ProximitySensor {
    size 200 200 200
    center 680720 521.5 -251595
}
DEF Taschenhaufen Transform {
    children [
        #Beschriftung
        LOD {
            level [
                DEF Beschriftung_Taschenhaufen Transform {
                    children [
                        Shape {
                            appearance Appearance {
                                material Material { emissiveColor 1.0 1.0 1.0 }
                            }
                            geometry Text {
                                string [ "Gepaeck von", "Bucheggplatz" ]
                                fontStyle FontStyle {
                                    family "SANS"
                                    size 0.5
                                    horizontal TRUE
                                    justify "MIDDLE"
                                }
                            }
                        }
                    ]
                    translation 0 2 0
                },
                Shape {
                    geometry PointSet {
                        coord Coordinate { point [ 0 0 0 ] }
                    }
                }
            ]
            range [ 40 ]
        }
        #Taschenhaufen
        LOD {
            level [
                Group {
                    children [
                        DEF T1 Transform {
                        children DEF ts1 Shape {
                            appearance Appearance {
                                material Material { diffuseColor 0.8 0.1 0.0 }
                            }
                            geometry Box { size 0.6 0.3 0.3 }
```

```
        }
        translation 0.0 0.15 0.0
    },
    DEF T2 Transform {
        children DEF ts2 Shape {
            appearance Appearance {
                material Material { diffuseColor 0.0 0.4 0.1 } #grün
            }
            geometry Box { size 0.5 0.2 0.4 }
        }
        translation 0.3 0.3 0.1
        rotation 0.5 1.0 -0.8 0.4
    },
    DEF T3 Transform {
        children DEF ts3 Shape {
            appearance Appearance {
                material Material { diffuseColor 0.0 0.0 0.8 } #blau
            }
            geometry Box { size 0.2 0.3 0.5 }
        }
        translation 0.2 0.1 0.5
        rotation 0.2 0.7 0.3 0.3
    },
    DEF T4 Transform {
        children DEF ts4 Shape {
            appearance Appearance {
                material Material { diffuseColor 1.0 0.4 0.0 } #orange
            }
            geometry Cylinder { radius 0.2 height 0.5 }
        }
        translation -0.2 0.2 -0.3
        rotation 0.7 0.1 0.7 0.7
    },
    DEF T5 Transform {
        children DEF ts5 Shape {
            appearance Appearance {
                material Material { diffuseColor 0.9 0.9 0.9 } #weiss
            }
            geometry Box { size 0.2 0.3 0.3 }
        }
        translation -0.3 0.2 0.4
        rotation 0.3 0.8 0.2 0.6
    },
    DEF T6 Transform {
        children DEF ts6 Shape {
            appearance Appearance {
                material Material { diffuseColor 0.0 0.0 0.0 }
            }
            geometry Cylinder { radius 0.15 height 0.4 }
        }
        translation 0.8 0.15 -0.8
        rotation 1.0 0.8 0.2 0.8
    },
    DEF T7 Transform {
        children USE ts1
        translation 0.5 0.5 -0.4
        rotation 0.1 0.2 0.3 0.4
    },
    DEF T8 Transform {
        children USE ts2
        translation 0.8 0.4 0.6
        rotation 0.5 0.6 0.7 0.8
    },
    DEF T9 Transform {
        children USE ts3
        translation -0.7 0.2 -0.1
        rotation 0.9 1 0 1.2
    },
    DEF T10 Transform {
        children USE ts4
        translation 1.3 0.3 -0.1
        rotation 0.2 0.3 0.4 1.6
    },
    DEF T11 Transform {
        children USE ts5
        translation -0.1 0.2 -1.1
        rotation 0.6 0.7 0.8 2.0
    },
```

```
                    DEF T12 Transform {
                        children USE ts6
                        translation -0.1 0.3 1.1
                        rotation 0 0.1 0.2 2.4
                    },
                    DEF T13 Transform {
                        children USE ts1
                        translation -0.9 0.2 0.7
                        rotation 0.4 0.5 0.6 2.8
                    },
                    DEF T14 Transform {
                        children USE ts2
                        translation -0.7 0.4 -0.7
                        rotation 0.8 0.9 1 3.2
                    },
                    DEF T15 Transform {
                        children USE ts3
                        translation 1.5 0.2 -0.7
                        rotation 0.2 0.3 0.4 3.6
                    },
                    DEF T16 Transform {
                        children USE ts4
                        translation 0.5 0.2 1.5
                        rotation 0.6 0.7 0.8 4
                    },
                    DEF T17 Transform {
                        children USE ts5
                        translation 1.7 0.3 0.5
                        rotation 1 0 0.1 4.4
                    }
                    DEF T18 Transform {
                        children USE ts6
                        translation -1.5 0.3 -0.2
                        rotation 0.2 0.3 0.4 4.7
                    },
                    DEF T19 Transform {
                        children USE ts1
                        translation 0.7 0.5 1
                        rotation 0.75 0.96 0.13 3
                    },
                    DEF T20 Transform {
                        children USE ts2
                        translation -0.5 0.5 0.8
                        rotation 0.4 0.5 0.3 3.5
                    },
                    DEF T21 Transform {
                        children USE ts3
                        translation -0.8 0.3 1.4
                        rotation 0.8 0.6 0.9 4
                    },
                    DEF T22 Transform {
                        children USE ts4
                        translation -1.1 0.4 0.3
                        rotation 0.4 0.3 0.5 4.5
                    },
                    DEF T23 Transform {
                        children USE ts5
                        translation 0.5 0.3 -1.1
                        rotation 0.6 0.5 0.4 5
                    },
                    DEF T24 Transform {
                        children USE ts6
                        translation 1.2 0.6 0.4
                        rotation 0.1 0.5 0.9 5.5
                    },
                ]
            },
            Shape {
                appearance Appearance {
                    material Material { diffuseColor 0.1 0.7 0.9 }
                }
                geometry Cylinder {
                    radius 2.0
                    height 1.0
                }
            }
        ]
        range [ 40 ]
```

```
        }
    ]
    translation 680720 521.5 -251595
}
ROUTE Sensor.orientation_changed TO Beschriftung_Taschenhaufen.rotation
```

Die Dateien mit den anderen Gepäckstandorte, t-hoebe-n.wrl, t-buchl-v.wrl und t-buchl-b.wrl besitzen die selbe Geometrie wie die oben aufgelistete Datei t-hoebe-v.wrl, und werden hier deshalb nicht wiederholt.

Wasser-2.wrl

```
#VRML V2.0 utf8
DEF Becher Transform {
    children [
        #Wasser
        Transform {
            children [
                Shape {
                    appearance Appearance {
                        material Material {
                            diffuseColor 0 1 1
                            transparency .2
                            emissiveColor 0 .1 .08
                            ambientIntensity 0.8
                            shininess 0
                        }
                    }
                    geometry Cone {
                        bottom TRUE
                        height 10
                        bottomRadius 1
                    }
                }
            ]
            rotation 1 0 0 3.14
        },
        #Glas
        Transform {
            children [
                Shape {
                    appearance Appearance {
                        material Material {
                            diffuseColor 0.8 0.8 0.9
                            transparency 0.8
                        }
                    }
                    geometry Cone {
                        bottom FALSE
                        side TRUE
                        height 10.5
                        bottomRadius 1.1
                    }
                }
            ]
            rotation 1 0 0 3.14
        },
        #Trinkhalm
        Group {
            children [
                Transform {
                    children Shape {
                        appearance DEF Halm Appearance {
                            material Material {
                                diffuseColor 1 0 0
                                ambientIntensity 0.5
                            }
                        }
                        geometry Cylinder {
                            height 3
                            radius 0.1
                        }
                    }
                    rotation 0 0 1 0.6
                    translation -0.2 4.2 0
                },
```

```
                Transform {
                    children Shape {
                        appearance USE Halm
                        geometry Cylinder {
                            height 1
                            radius 0.1
                        }
                    }
                    rotation 0 0 1 -0.8
                    translation -1.4 5.1 0
                }
            ]
        }
    ]
    translation 676950 392 -250789
}
```

buchen-reihe.wrl

```
#VRML V2.0 utf8
DEF Buchenreihe Transform {
    children [
        Transform {
            children Inline { url "buche-klein.wrl" }
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 5 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 10 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 15 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 20 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 25 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 30 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 35 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 40 0 0
        },
        Transform {
            children Inline { url "buche-klein.wrl" }
            translation 45 0 0
        },
    ]
    translation 680729 523.6 -251494
    rotation 0 1 0 1.13
}
```

buche-klein.wrl

```
#VRML V2.0 utf8
DEF  Buche-klein Transform {
    children [
        #Stamm
        Transform {
            children Shape {
                appearance Appearance {
                    material DEF Stamm Material { diffuseColor 0.6 0.3 0.2 }
```

```
                }
                geometry Cylinder {
                    radius 0.08
                    height 2
                }
            }
            translation 0 1 0
        }
        #Krone
        Transform {
            children Shape {
                appearance Appearance {
                    material Material {
                        diffuseColor 0.2 0.6 0.2
                        transparency 0.4
                    }
                }
                geometry Sphere { radius 1 }
            }
            translation 0 2.95 0
        }
        #Äste
        Transform {
            children [
                Transform {
                    children Shape {
                        appearance Appearance {
                            material DEF Ast Material { diffuseColor 0.6 0.2 0.2 }
                        }
                        geometry DEF Cone Cone {
                            bottomRadius 0.04
                            height 1
                        }
                    }
                    translation 0 2.5 0
                    center 0 -0.5 0
                    rotation 0 0 1 0.7
                }
                Transform {
                    children Shape {
                        appearance Appearance { material USE Ast }
                        geometry Cone {
                            bottomRadius 0.06
                            height 1.8
                        }
                    }
                    translation 0 2.9 0
                    center 0 -0.9 0
                    rotation 1 0 1 -0.5
                }
                Transform {
                    children Shape {
                        appearance Appearance { material USE Ast }
                        geometry Cone {
                            bottomRadius 0.05
                            height 1.4
                        }
                    }
                    translation 0 2.7 0
                    center 0 -0.7 0
                    rotation -1 0 1 0.8
                }
                Transform {
                    children Shape {
                        appearance Appearance { material USE Ast }
                        geometry Cone {
                            bottomRadius 0.07
                            height 1.8
                        }
                    }
                    translation 0 2.9 0
                    center 0 -0.9 0
                    rotation -1 0 0.6 -0.5
                }
            ]
        }
    ]
}
```

str-1.wrl

```
#VRML V2.0 utf8
Shape {
     appearance DEF k1_KLASSMat Appearance {
          material Material {
               diffuseColor .7 .7 .7
               ambientIntensity 0.2
               shininess 0.2
          }
     }
     geometry IndexedFaceSet {
          solid FALSE
          coord Coordinate {
               point [ … ]
          }
          coordIndex [ … ]

     }
},
```

usw.

Alle anderen Strassen-Dateien (str-2.wrl, str-3.wrl, str-4.wrl, str-5.wrl, str-6.wrl, str-auto.wrl, str-bahn.wrl, str-fuss.wrl, str-s-1.wrl, str-s-2.wrl, str-s-3.wrl, str-s-4.wrl, str-s-5.wrl, str-s-6.wrl, str-s-7.wrl) sind auf ähnlich aufgebaut. Es ändert sich lediglich die Appearance, sowie natürlich die Geometrie.

sphere.wrl

```
#VRML V2.0 utf8
NavigationInfo {
     headlight FALSE
     type "NONE"
}
Background {
     skyColor 1 1 1
}
DirectionalLight {
     direction 1 -1 -2
     ambientIntensity 1
     color 1.0 1.0 0.9
     intensity 1
     on TRUE
}
DEF matSphere Material { diffuseColor 1 1 1 }
Shape {
     appearance Appearance {
          material USE matSphere
     }
     geometry Sphere { }
}
Viewpoint {
     fieldOfView 0.2
}
```

B. CSS- und HTML-Dateien

style.css

```
/* style-sheet für die SOLA-3D-Visualisierung-Homepage */
/* Andreas Meister, 7. Februar 2001 */
body {       background: #effff7; /* hellgrün */
             color: #000000;
             font-family: Helvetica, Arial; }
h1, #titre1 {font-family: 'Formal436 BT', Helvetica, Verdana;
             font-size: 18pt;
             font-style: italic;
             font-weight: medium;
             text-align: center; }
h2, #titre2 {font-family: 'Formal436 BT', Helvetica, Verdana;
             font-size: 16pt;
             font-weight: bold; }
h3, #titre3 {font-family: Helvetica, Verdana;
             font-weight: bold;
             font-size: 14pt; }
caption {    font-size: 12pt;
             font-style: bold; }
p, br, td, div, li { font-size: 11pt; }
a:link {     text-decoration: none;
             color: #000000; }
a:active {   text-decoration: none;
             color: #000000; }
a:visited {  text-decoration: none;
             color: #000000; }
a:hover {    text-decoration: underline;
             color: #3399ff; /* hellblau */ }
```

sola-str2.html

```
<html>
<head>
    <title>3D-Visualisierung der SOLA-Stafette mittels VRML</title>
    <meta http-equiv="author" content="Andreas Meister">
    <meta http-equiv="distribution" content="global">
    <meta http-equiv="resource-type" content="document">
</head>

<frameset rows="70,*" border=0 frameborder=0 framespacing=0>
    <frame src="titelleiste.html" name="titelleiste" scrolling='NO' noresize marginheight=2
    marginwidth=2>
    <frameset cols="60%,40%" border=0 frameborder=0 framespacing=0>
        <frame src="../vrml-files/SOLA-Strecke2.wrl" name="plugin" scrolling='NO' noresize
        marginheight=2 marginwidth=2>
        <frameset rows="50%,50%" border=0 frameborder=0 framespacing=0>
            <frame src="kartenausschnitt.html" name="kartenausschnitt" scrolling='NO'
            noresize marginheight=2 marginwidth=2>
            <frame src="legende.html" name="legende" noresize marginheight=2 marginwidth=2>
        </frameset>
    </frameset>
</frameset>
</html>
```

titelleiste.html

```
<html>
<head>
<title>Titelleiste</title>
<link rel=stylesheet type="text/css" href="style.css">
</head>

<body>
<table height=100%>
<tr valign=center>
<td width=20%>
<a href="http://www.sola.asvz.ethz.ch/" target="_top">Zur&uuml;ck zur SOLA-Homepage</a></td>
<td width=60% align=center><h1>SOLA-Visualisierung<br>Strecke 2</h1></td>
```

```
<td width=20%> </td>
</tr>
</table>
</body>
</html>
```

kartenausschnitt.html

```
<html>
<head>
<title>Kartenausschnitt</title>
<script language="javascript">
<!--
var isNS = (navigator.appName == "Netscape" && parseInt(navigator.appVersion) >= 4);
var objet;
var posNode;
var mapBrowser;
function animation()
{
    pos=posNode.getEventOut("position_changed").getValue();

    objet.left=(pos[0]-678000)/20+parent.frames["kartenausschnitt"].innerWidth/2;
    objet.top= (pos[2]+250500)/20+parent.frames["kartenausschnitt"].innerHeight/2;
    §setTimeout("animation()", 3) ;
}
function start()
{
    if (isNS) {
        mapBrowser=parent.frames["plugin"].document.plugins[0];
        posNode =mapBrowser.getNode("Position");
        objet=document.layerobjet;
        animation();
    }
}

// -->
</script>
</head>

<body onload="start()">
<table width=100% height=100%>
<tr valign="middle"><td><center>
<img src="Ausschnitt_klein.gif">
<layer name=layerobjet height=20 width=20 ><img SRC="pointer.gif" NAME="tirob" ></layer>
</center></td></tr>
</table>
</body>
</html>
```

legende.html

```
<html>
<head>
<title>Legende</title>
<link rel=stylesheet type="text/css" href="style.css">
</head>

<body>
<table width=100% height=100%>
<tr><td colspan=4>
<b>Legende</b>
</td></tr>
<tr><td><embed src="sphere.wrl" name="cosmo_wiese" width=40 height=40></td><td>Wiese</td>
<td><embed src="sphere.wrl" name="cosmo_wald" width=40 height=40></td><td>Wald</td></tr>
<tr><td><embed src="sphere.wrl" name="cosmo_fluss" width=40 height=40></td><td>Fluss</td>
<td><embed src="sphere.wrl" name="cosmo_see" width=40 height=40></td><td>See</td></tr>
<tr><td><embed src="sphere.wrl" name="cosmo_siedl" width=40
height=40></td><td>Siedlungsgebiet</td>
<td><embed src="sphere.wrl" name="cosmo_reben" width=40 height=40></td><td>Rebberg</td></tr>
<tr><td colspan=4>
<a href="Farbenwahl.html" target="kartenausschnitt">Die Farben &auml;ndern</a>
</td></tr>
<tr valign="bottom"><td colspan=4>
&copy; <a href="mailto:andreas@guv.ethz.ch?subject=SOLA-Visualisierung">Andreas Meister</a>,
31. Januar 2001
</td></tr>
```

```
</table>
</body>

<script language="JavaScript">
var matNode;
var mapBrowser,legBrowser;
mapBrowser=parent.frames["plugin"].document.plugins[0];
matNode   =mapBrowser.getNode('Uebrig_mat');
matCol    =matNode.getEventOut("diffuseColor");
new_color =matCol.getValue();
legBrowser=parent.frames["legende"].document.plugins['cosmo_wiese'];
legNode   =legBrowser.getNode("matSphere");
legNode.getEventIn("set_diffuseColor").setValue(new_color);
matNode   =mapBrowser.getNode('Wald_mat');
matCol    =matNode.getEventOut("diffuseColor");
new_color =matCol.getValue();
legBrowser=parent.frames["legende"].document.plugins['cosmo_wald'];
legNode =legBrowser.getNode("matSphere");
legNode.getEventIn("set_diffuseColor").setValue(new_color);
matNode   =mapBrowser.getNode('Fluss_mat');
matCol     =matNode.getEventOut("diffuseColor");
new_color = matCol.getValue();
legBrowser=parent.frames["legende"].document.plugins['cosmo_fluss'];
legNode =legBrowser.getNode("matSphere");
legNode.getEventIn("set_diffuseColor").setValue(new_color);
matNode   =mapBrowser.getNode('See_mat');
matCol     =matNode.getEventOut("diffuseColor");
new_color = matCol.getValue();
legBrowser=parent.frames["legende"].document.plugins['cosmo_see'];
legNode =legBrowser.getNode("matSphere");
legNode.getEventIn("set_diffuseColor").setValue(new_color);
matNode   =mapBrowser.getNode('Siedl_mat');
matCol     =matNode.getEventOut("diffuseColor");
new_color = matCol.getValue();
legBrowser=parent.frames["legende"].document.plugins['cosmo_siedl'];
legNode =legBrowser.getNode("matSphere");
legNode.getEventIn("set_diffuseColor").setValue(new_color);
matNode   =mapBrowser.getNode('Reben_mat');
matCol     =matNode.getEventOut("diffuseColor");
new_color = matCol.getValue();
legBrowser=parent.frames["legende"].document.plugins['cosmo_reben'];
legNode =legBrowser.getNode("matSphere");
legNode.getEventIn("set_diffuseColor").setValue(new_color);
</script>
</html>
```

Farbenwahl.html

```
<html>
<head>
<title>Farbenwahl</title>
<link rel=stylesheet type="text/css" href="style.css">
</head>

<script language="JavaScript">
<!--
var matNode,legNode,legCol;
var mapBrowser,legBrowser;
mapBrowser=parent.frames["plugin"].document.plugins[0];
function werte_zeigen(typ)
{
    // get the selected material node
    matNode   =mapBrowser.getNode(typ);
    matCol     =matNode.getEventOut("diffuseColor");
    new_color = matCol.getValue();
    legBrowser=parent.frames["kartenausschnitt"].document.plugins[0];
    legNode =legBrowser.getNode("matSphere");
    legNode.getEventIn("set_diffuseColor").setValue(new_color);
    // show the values
    document.forms[0].red.value  =Math.round(new_color[0] * 255);
    document.forms[0].green.value=Math.round(new_color[1] * 255);
    document.forms[0].blue.value =Math.round(new_color[2] * 255);
    return 1;
}
function new_sphere()
{
    // get the new color values
```

```
    new_color = matCol.getValue();
    new_color[0]=parseFloat(document.forms[0].red.value)/255;
    new_color[1]=parseFloat(document.forms[0].green.value)/255;
    new_color[2]=parseFloat(document.forms[0].blue.value)/255;
    legBrowser=parent.frames["kartenausschnitt"].document.plugins[0];
    legNode =legBrowser.getNode("matSphere");
    legNode.getEventIn("set_diffuseColor").setValue(new_color);
}
function werte_setzen()
{
    // get the new color values
    new_color[0]=parseFloat(document.forms[0].red.value)/255;
    new_color[1]=parseFloat(document.forms[0].green.value)/255;
    new_color[2]=parseFloat(document.forms[0].blue.value)/255;
    // set the new color
    matNode.getEventIn("set_diffuseColor").setValue(new_color);
    legNode.getEventIn("set_diffuseColor").setValue(new_color);
    // legende
    matNode   =mapBrowser.getNode('Uebrig_mat');
    matCol     =matNode.getEventOut("diffuseColor");
    new_color = matCol.getValue();
    legBrowser=parent.frames["legende"].document.plugins['cosmo_wiese'];
    legBrowser.getNode("matSphere").getEventIn("set_diffuseColor").setValue(new_color);
    matNode   =mapBrowser.getNode('Wald_mat');
    matCol     =matNode.getEventOut("diffuseColor");
    new_color = matCol.getValue();
    legBrowser=parent.frames["legende"].document.plugins['cosmo_wald'];
    legBrowser.getNode("matSphere").getEventIn("set_diffuseColor").setValue(new_color);
    matNode   =mapBrowser.getNode('Fluss_mat');
    matCol     =matNode.getEventOut("diffuseColor");
    new_color = matCol.getValue();
    legBrowser=parent.frames["legende"].document.plugins['cosmo_fluss'];
    legBrowser.getNode("matSphere").getEventIn("set_diffuseColor").setValue(new_color);
    matNode   =mapBrowser.getNode('See_mat');
    matCol     =matNode.getEventOut("diffuseColor");
    new_color = matCol.getValue();
    legBrowser=parent.frames["legende"].document.plugins['cosmo_see'];
    legBrowser.getNode("matSphere").getEventIn("set_diffuseColor").setValue(new_color);
    matNode   =mapBrowser.getNode('Siedl_mat');
    matCol     =matNode.getEventOut("diffuseColor");
    new_color = matCol.getValue();
    legBrowser=parent.frames["legende"].document.plugins['cosmo_siedl'];
    legBrowser.getNode("matSphere").getEventIn("set_diffuseColor").setValue(new_color);
    matNode   =mapBrowser.getNode('Reben_mat');
    matCol     =matNode.getEventOut("diffuseColor");
    new_color = matCol.getValue();
    legBrowser=parent.frames["legende"].document.plugins['cosmo_reben'];
    legBrowser.getNode("matSphere").getEventIn("set_diffuseColor").setValue(new_color);
}
// -->
</script>
<body>
<form>
<table width=100%>
<tr><td>
<input type="radio" name="prim" value="Wiese" onClick="werte_zeigen('Uebrig_mat')">Wiese<br>
<input type="radio" name="prim" value="Wald" onClick="werte_zeigen('Fluss_mat')">Fluss<br>
<input type="radio" name="prim" value="Wald" onClick="werte_zeigen('Reben_mat')">Reben<br>
<input type="radio" name="prim" value="Wald" onClick="werte_zeigen('See_mat')">See<br>
<input type="radio" name="prim" value="Wald" onClick="werte_zeigen('Siedl_mat')">Siedlung<br>
<input type="radio" name="prim" value="Wald" onClick="werte_zeigen('Wald_mat')">Wald<br>
</td>
<td>
<input type="text" name="red" size="3"><br>
<input type="text" name="green" size="3"><br>
<input type="text" name="blue" size="3"><br>
<input type="button" onclick="new_sphere()" value="Farbe zeigen"><br>
</td>
<td>
<embed src="sphere.wrl" name="cosmo" width=80 height=80>
</td>
</tr>
</table>
<input type="button" onclick="werte_setzen()" value="Setze die Farbe"><br>
</form>
<p><a href="kartenausschnitt.html">Schliessen</a></p>
</body>
</html>
```