Instructor: Stefan Seipel
ssl@hig.se

Geovisualisering och bildbehandling

Project, part 2: Visualization

The assignment is mandatory and should be handed in due to deadline. This part of the project builds upon results from the first project part, hence finishing essential requirements from part 1 is necessary to proceed with this part of the project. A total of 10 points can be obtained for this part of the project and these points will be weighted into calculation of the final course grade. The submission deadline for this part is set to **16/1-2015**. Handing in later than this date will result in withdrawal of 2 points.

Project objectives

In the first part of the project you used image analysis methods to automatically extract edges and contours of floods banks from discrete 2D image data (Landsat images) at different points in time (before and after a flood).

The general objective of the visualization part of this project is to provide a visualization, which helps the user to explore the spatial and temporal relationship in the change of river banks. In particular the visualization should allow exploring specific geographic locations and the proximity to the river/floods at arbitrary times. One example of a question, which the visualization is supposed to answer, could be:

Q: When was the flood/river closest to a specific geographic location and how close was the flood then?

Due to the fact that raw data, as retrieved in the first part of the project, describes flood lines only at few discrete steps in time, the user’s task requires some sort of mental extrapolation/interpolation of positions and times.

Visualization approach and data

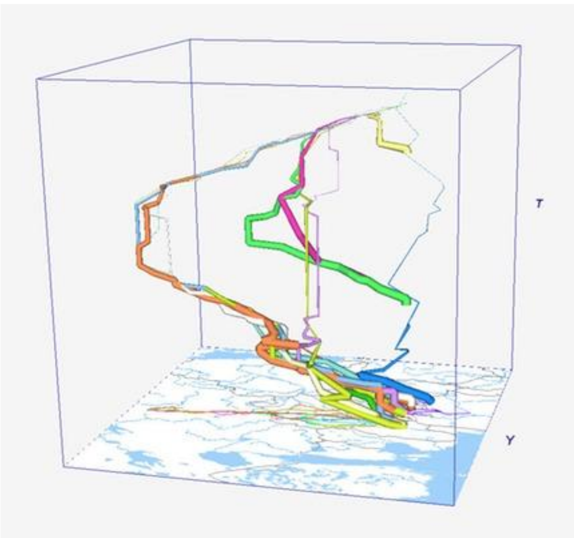

Figure 1: A Space-Time-Cube visualization of moving entities: Image http://www.itc.nl/personal/kraak/move/

Instructor: Stefan Seipel
ssl@hig.se

In the field of time-geography, a number of interesting visualization metaphors have been proposed to depict phenomena that occur in geographic context over time. The time-space-cube is a visualization method that is based on a 3D rectilinear context, where the 2D geographical coordinate system is extended with a time-axis pointing along the 3rd Cartesian axis. Figure 1 shows an example of a time-space-cube visualization wherein the motion of a number of entities (in geography and time) is depicted through a 3D trajectory in the time-space-cube.
This form of visualization can be applied to other phenomena. A 2D contour in geography that changes over time can be evolved into a surface within a 3D time-space-cube visualization.
Visual exploration of temporal and spatial relationships can then be facilitated through interactive cursors (e.g. 3D crosshairs) which enable to identify coincidences and distances in all dimensions (temporal and spatial).

Data needed for this type of visualization comprises:

a) A cartographic (2D) map that provides geographic reference. Here, digital images from LandSat or other registered aerial images can be used. Those discrete maps must be imported into a visualization using suitable formats and are subsequently treated as texture images when rendering the visualization.

b) Contours that describe the observed phenomenon (river banks, river lines). They form the basis for extruding a surface into the 3D time-space cube. If several contours are used, identifying information must be available to associate them with one another. Also, time stamps are needed for every contour in order to maintain correct connectivity when building contour surfaces in time-space.
   Contours must be represented as explicit models, that is, a list of vertices. Contours should be reasonably dense sampled (not too sparse to preserve curvature, and not too dense to avoid huge polygon models).

Tasks

Implement an interactive visualization in Vizard (http://www.worldviz.com/) that allows to import data extracted from the first part of the project and to visually explore time-space relationships using a time-space cube visualization. The visualization should be interactive in the sense that it provides at least intuitive view-control (seeing the visualization from different directions and distances) and some effective means of interaction for identifying geographic locations and their relation to flood borders/lines at arbitrary points in time. Several steps are required to be solved here:

Step 1: Map import

Write a Python script in Vizard to import a geographic map as a texture map and to visualize this using one or several texture mapped polygons. To build the texture-mapped geometry, use the concept of **Layers/On-The-Fly Nodes** in Vizard – they enable you to create graphical objects on a very low level. Define coordinate systems/scale levels according to your own preference once for all and be consistent with this throughout the remainder of the project. At this stage it is not necessary that the texture/geographic map is correctly geo-referenced. But care should be taken so that segmented river banks/flood lines actually coincide with the corresponding geographic positions in the map.

Instructor: Stefan Seipel
ssl@hig.se

```
#Example of creating an on-the-fly node object

viz.startlayer(viz.LINES)
viz.vertex(0,0,0)
viz.vertex(0,1,0)
lines = viz.endlayer()

#Add another line
lines.addVertex([1,0,0])
lines.addVertex([1,1,0])
```

Step 2: Simple navigation

Once you have made visible your map on screen you will find that normal scene navigation using the built-in flying-navigation functionality will get you lost very quickly. Implement therefore an alternative way for view control by treating the visualization as an object that is rotated/re-sized etc. with a virtually static observer. Add also some lines and graphical elements to the visualization that might support navigation and that outline the limits of your time-space cube visualization.

Step 3: Creation of contour surfaces

Implement a script that reads contours from a file (as saved from the first part of the project) and stores them as lists of vertices. Maintain additional information along with each contour (time-stamp and contour identifier) in a way suitable to process contours in an ordered way. Implement a script that builds up one or several 3D surfaces (depending on number of contours) that represent how the contour is evolving over time into the 3rd dimension. This surface is in its simplest form a polygon surface where vertices from any two temporally subsequent contours are connected to form either a *triangle-strip* or a *quad-strip*. At this point it is becomes clear that contour-lines must be identifiable and have a time stamp in order to know how to build up the time-space surface. Also, take care that contour lines in subsequent times have the same number of vertices. This will make the task of building quad-strips/triangle-strips much easier.

When building up 3D contour surfaces try different time-scales, colors, and transparency until you arrive at a point where you are satisfied with the visual appearance. Again, for building the geometry, use the concept of Layer-Nodes in Vizard that allows creation of custom geometry at run-time.

Step 4: Interactive exploration

To explore coincidence of geographical locations with a contour in 3D space, provide some intuitive interaction tool. As proposed above, a 3D cross-hairs cursor could be used to mark readings in the map and along the temporal axes. Other tools (like semi-transparent cut planes etc..) are also possible to support this task. In addition to control and visual feedback of the interaction tool print out current readings in some appropriate way.

Step 5: Alternative solutions

Now that you have implemented an interactive time-space cube visualization, take a step back and think of other visualizations/ways in which the question described above could be answered! Discuss how such a possible alternative solution would conceptually work. Discuss how you would evaluate/compare this method with the time-space cube visualization.

Step 6: Test / validation

Using the given scenario (data) that you were working with, identify three different hypothetical locations A, B, and C in the map by adding some graphical elements to the corresponding map positions. Ask someone near to you to volunteer as a test person to use your interactive visualization to answer the question:

Q: When was the flood/river closest to A (likewise for B and C) and how close was the flood then?

Observe the test person and annotate the answers given by the test person. Also note time. From that, summarize in your report what can you conclude. In particular

- Was the test person able to solve the task i.e. answering the above question?
- Was your visualization tool intuitive to use by the test person?
- What changes/additions would be required to improve your visualization?

Grades

The assignment comprises two deliverables: a) Python/Vizard scripts and data to run your visualization and for verification on how you solved steps 1-4. b) A report that briefly summarizes/describes your visualization application and, more importantly, discusses step 5 and presents step 6. Based on the code and report goal achievement will be assessed and the following maximum points will be given:

Step 1 : 1 point

Step 2 : 1 point

Step 3 : 3 points

Step 4 : 2 points

Step 5: 1 point

Step 6: 2 points

At most 10 points can be obtained. Failing to deliver the project in time will result in a reduction of 2 points on the obtained points. Submission deadline is **16/1-2015**.