Liquid 2D Scatter Space for file system browsing

Carsten Waldeck

Infoverse.org and ZGDV - Computer Graphics Center,
Dept. Z3 (MIV: Mobile Information Visualization)

E-mail: waldeck@infoverse.org

Abstract

This (video-) paper describes a user interface concept, which facilitates multidimensional visual queries, filtering and browsing of the file system at the same time by means of the "Liquid Scatter Space" (LSS) concept ([1] Waldeck 2004) .

The UI concept is based on an advanced star field display ([2] Ahlberg & Shneiderman 1994) using liquid browsing [1] and selection based (sketch-) queries. It allows realtime search and browsing at the same time (integrated into one single user interface) and can visualize the most important file system meta data dimensions simultaneously: 1. filename, 2. creation/modification date, 3. filesize, 4. filetype and 5. label. LSS provides very fast and easy visual data mining possibilities for the desktop and makes it possible to perceive complex dependencies between the most important file system meta data properties at a glance. It also overcomes the "hidden-deep-down-in-hierarchy-structures"-problem by making it possible to use folders for structuring but not being bound to them.

This paper focuses on visual design and interaction aspects and emphasizes the importance of paying regard to visual interactive details for information visualization and interaction interfaces and the need of making it easy to use.

Keywords: liquid browsing, focus and context, visual interactive information spaces, mobile computing, user interface design, dynamic interactive visualization, visual data mining, file system browsing.

1. Introduction and Problem Description

As harddrive capacities become larger and larger, enabling us to to store significantly more data and user tasks becoming more and more complex and mobile, the need for handling a far greater amount of files, even on smaller screens (1.1.), and finding them in very deep and complex hierarchy structures (1.2.) is growing.

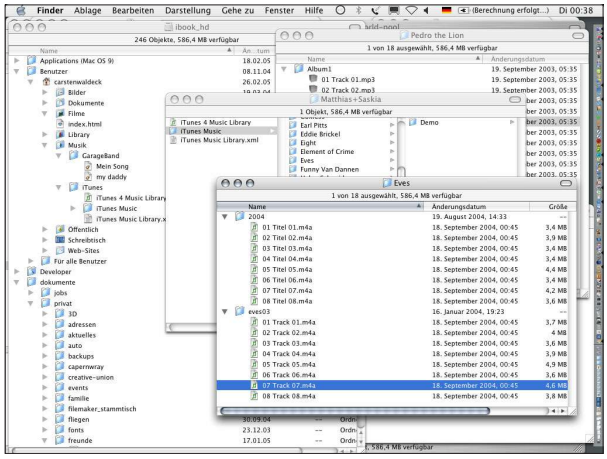

Figure 1: everyday situation on a normal desktop.

1.1. A table view (which is the most common view we use today) allows us to only look at 40-80 objects (files) at a time without scrolling. For mobile devices or smaller screens this number cuts down to 15-30 objects. Focus and context solutions ([3] T. Keahey, 1998) like the table lens ([4] Rao & Stuart, 1994) have been proposed to solve this problem.

Furthermore Table views only allow us to sort and compare according to one single criterion, which forces the user to change sorting quite often and makes it very hard to find interdependencies between the different metadata dimensions.

1.2. The hierarchical filestore is normally too large to be fully displayed on the user's screen. The common folder view tries to solve this problem by scrolling and the possibility to open and close folders. This metaphor has a clear disadvantage: Very often users can't find files that are hidden somewhere deep down in complex hierarchy structures. Alternative visualization and interaction methods have been proposed, such as Cone Trees [5] and the Hyperbolic Browser [6]. Until now those methods have not been integrated into your everyday operating systems due to poor navigational performance advantage (in some cases even disadvantages) over standard mechanisms when being used for OS file management tasks. Treemaps ([7] Johnson B, Shneiderman B, 1991) can give a good space-filling overview, but browsing single files and detail navigation can be quite difficult to perform in that case.

Also it can be a very hard task to find a file when the user has forgotten the filename. Of cause there are possibilities for metadata search in operating systems today, but using them is still far from being direct and intuitive.

2. Solution

An interactive 2D scatter plot approach allows very large amounts of information to be visualized (thousands of objects) and is simultaneously sortable based on two criteria. In addition, comparisons based on a multitude of criteria are possible at the same time (according to mapping and expressivity of the visual parameters: size, opacity, color value, shape, orientation, ...or changes of all these parameters over time, resulting in animation).

The "Liquid 2D Scatter Space" [1] makes it possible to perceive very high information densities with multiple dimensions (2-7) of metadata simultaneously.

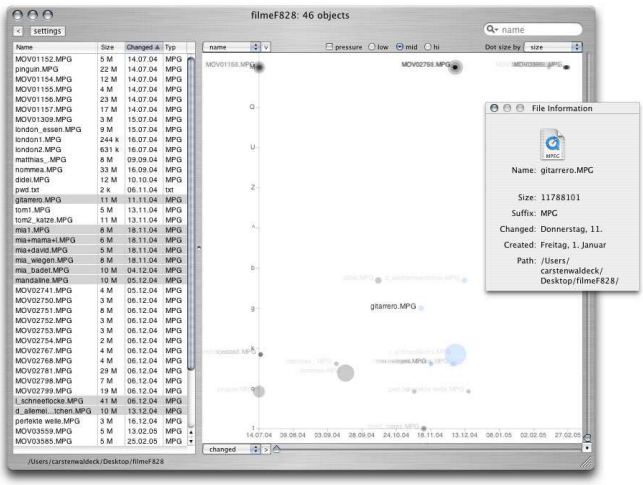

Figure 2: LSS window with syncronized table view and info details view

These qualities make it very useful especially for different kinds of visual metadata browsers, but also applied to the file system it offers a lot of advantages over the common table view based Finder (or Windows Explorer) way:

a) The LSS-view based file system visualization can provide an overview even with thousands of files while being easily navigable with the help of the liquid browsing interaction method to get rid of the overlapping problem of scatterplots and providing focus and context functionality (please refer to the video for this subject).

b) It is possible to visualize the most important file system meta data simultaneously, making it possible to perceive dependencies of the metadata dimensions at a glance.

c) LSS enables the user to search and browse at the same time by sketching visual queries into the scatterspace.

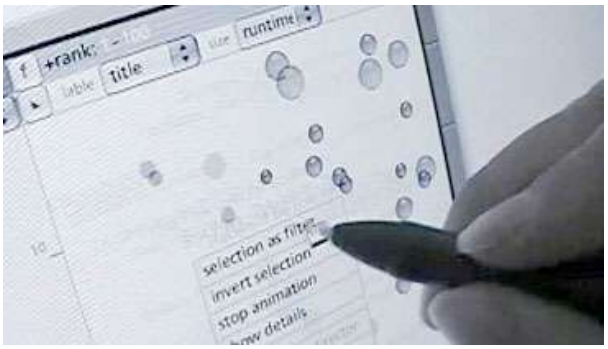

Figure 3: setting 4 filter values in two dimensions with one scetch

By optionally combining the scatter space with a table view (figure 2) and applying linking and brushing functionality we can extend the possibilities: Selecting one or more files in the table view selects the according items in the scatterspace and vice versa. Selecting multiple files in the scatterspace by drawing a selection rectangle makes the table view show only all the selected items.

By synchronizing the scatter space with a hierarchy list we get an even more powerful tool (figure 4): Selecting a folder in the hierarchy view selects all files in the scatterspace which are contained in that folder (or any of its sub-folders). Selecting an item (representing a file) in the scatterspace opens the folder hierarchy and shows the file in its containing folder(s). The user can set default preferences to define, if he also wants the files to be visualized within the hierarchy (like the MacFinder) or only see the plain hierarchy structure without files (the Windows Explorer way).

2.1. Redesign: Cleaning up the UI

Because we have the possibility to visualize the most important file system meta data simultaneously, there is not really a need to change the values on the axes of the scatterplot. By assigning a) the filename to the vertical y-axis, b) the creation/modification date/time to the horizontal x-axis, and c) the filesize to the size of the dots, we can achieve a very intuitive and easy to use visualization:

a) the filename is used just the same way the user is familiar with by working with the common table view.

b) the date assigned to the horizontal axis feels just like a common timeline metaphor from left to right.

c) representing the file size by the size of the dots is very direct and straight forward ([8] Bertin 1983).

There is also another important reason for assigning the axes in the described way: There can never be a case of total overlapping (one of the main problems of the scatter space), because a file can never have exactly the same name and modification date (unless it is exactly the same file, just being in different folders).

When working with the file system, users very often have to change the sorting of the tableview. In the majority of cases they have to change it from filename to modification date and vice versa (to find their newly created files). When using the LSS in the described way we don't need to change the sorting to fulfill the same tasks., But what is more: we do not even need to provide a possibility to change the sort order (ascending/descending) for the scatter space, because the standard setting is space-filling. That means, you can see all of the files without scrolling and it doesn't matter if you begin to look for your file on the top or on the bottom of the scatterspace window. With LSS, the files always stay at their original place, which enables the user to find the wanted files very fast and intuitively with the help of their spatial memory: the new ones are always right and his files that begin with "a" are at the top.

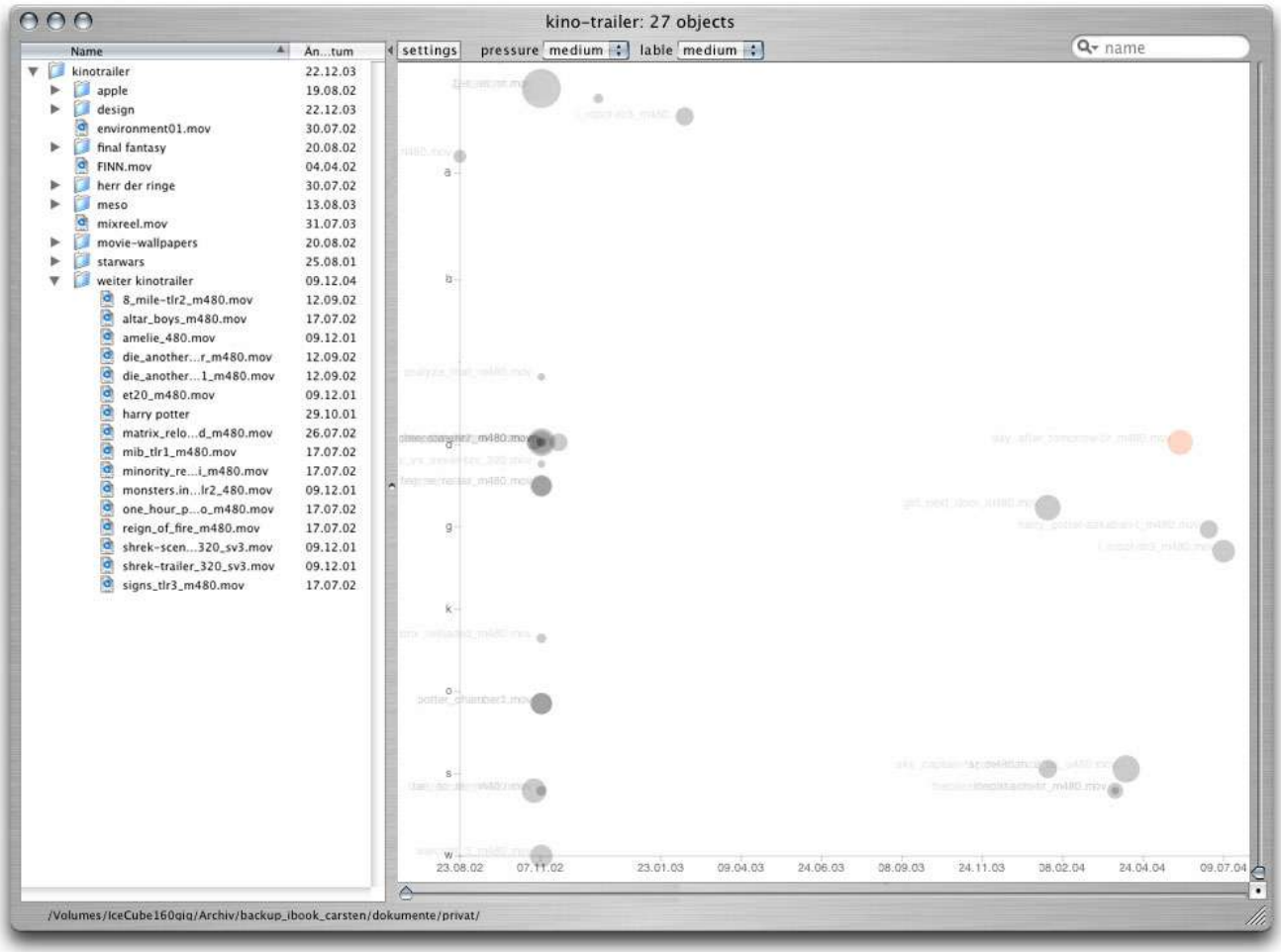

Figure 4: The new LSS window with syncronized tree/table/hierarchy list view

2.2. The ScatterTable

The ScatterTable approach goes one step further with the integration of the scatter space and common table- or hierarchy view. In this case they are not only synchronized, but they are merging and become a new view with slightly different characteristics.

You can think of it either as a liquid scatter space view with left-aligned lable or as a (super-) table view with a scatter space column.

One of the great advantages is of cause the reduced screen space consumption (figure 5). The Finder view on the left shows only 67% of the files (see scrollbar), while the ScatterTable shows 100% with only using less than 10% of the screen space. Similar to a Table Lens [4] or Fisheye Menu [13] the lable size grows, when you get close to its vertical position (the lable or corresponding dot or anywhere in-between). The enlarged lable can grow even over the scatter column, because the dots are rendered in a way (10-30% opacity) that they do not hinder legibility. That means, that the overall width of the window only has to be as wide as the string the user needs to read when scaled up to his favorite font size on rollover.

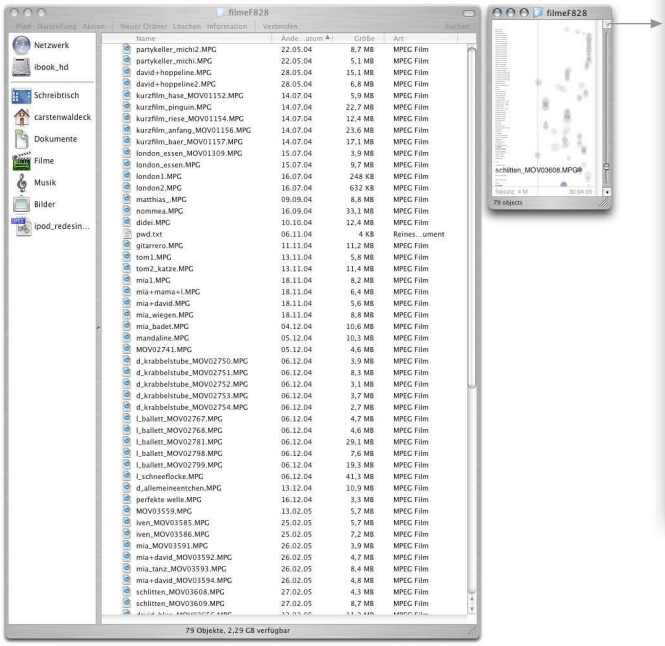

Figure 5: comparison: size of Finder window vs. ScatterTable

One disadvantage of the ScatterTable is the fact, that it is more space consuming in vertical direction than a pure scatter space approach. The maximum amount of objects that can be visualized this way without scrolling is the window height in pixels divided by 2. For example, on a standard XGA display (1024x768 px) it only makes sense to visualize 300-400 objects. In that case the lable tags are only visible as tiny 1 pixel lines and after that they will disappear. The scatter column can handle much higher densities that this. The good news is, that the user can fit about 4 of these ScatterTable views next to each other on one XGA display without overlapping of the windows.

Realtime filtering and representation adjustment can be applied with the help of the adjustment drawer (figure 6) or context menu within the same interface, while the user can continue to browse the files on the main ScatterTable window.

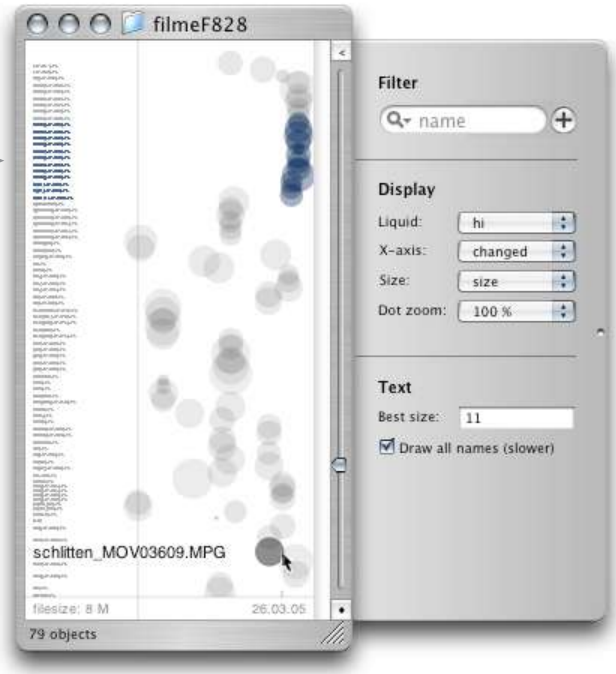

Figure 6: ScatterTable with extended adjustment drawer (simple version)

2.3. The ScatterTree

The ScatterTree approach is about the fusion of scatter space and hierarchical table view. This view provides all the functionality of the ScatterTable plus the possibility of hierarchical browsing within the same interface.

One very important subject of this approach is, that the user can decide if he wants to see all the content that is within closed folders (called flat or cumulative browsing, figure 7), or if he wants to see it the common non-cumulative way, having to open the folder to see its content. When browsing the files the cumulative way, the user will see all the files in an unstructured way and then he can control the level of structure he needs by opening the relevant folders. Another interesting point about browsing this way, is the fact, that clicking on an object in the scatter column can open the corresponding folders in the hierarchy column and show the complete path and the file within the containing folder(s) in realtime.

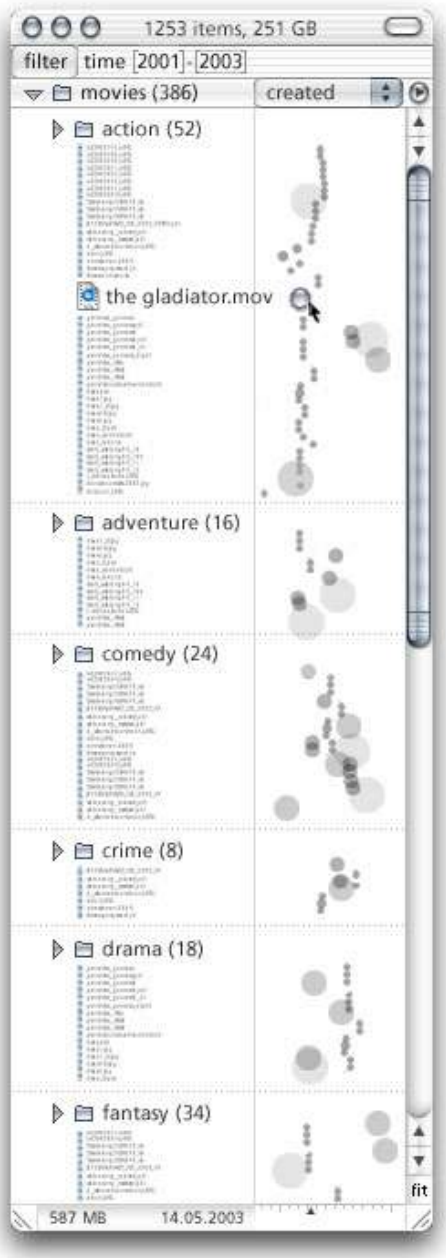

Figure 7: ScatterTree with activated cumulative view

4. Implementation

Our first LSS for a file system was implemented in ObjectiveC for MacOSX [9]. The current versions are realized as stand-alone application. When you start the program for the first time you get an empty window (otherwise it is showing your last state). You can drag any folder or complete hard drive onto this window (or choose it with the open panel) to visualize all its containing files. All of the OSX Finder functions (copy, move, select, open,...) work exactly like in the original Finder view. That means you can for example simply click a file and drag it to the desktop or another Finder window while pressing the alt key to copy the file to the new location.

You can find a movie about it at:
http://www.infoverse.org

The application can be downloaded at:
http://www.i-stuff.de

The Liquid 2D Scatter Space version that is shown in the movie was running on an 800MHz G4 Mac attached to a Wacom Cintiq (Interactive Pen Display). Since most of the people do not have this kind of display and Apple Computer has not built something like a TabletPC yet, we also provide a version, that can be browsed with a regular mouse and manual pressure adjustment.

5. Conclusion and Future Work

This paper has introduced the first use of LSS on the file system. Though extensive empirical studies still have to be done, first user tests have shown, that only after a few minutes of familiarization with the LSS, users where much faster in finding files, especially when having a rough feeling for 2 or more metadata dimensions (like: "it was a small, newly created file). The more files they had to handle, the bigger the temporal advantage became.

Some more complex questions, that were clear to the LSS test persons at a glance, couldn't even be solved at all by the test persons doing it the conventional way or it took exceptionally long: How has the filesize of my different document versions changed over time? When did i create or copy many documents? Where are the big files (hidden in many different subfolders)? Which documents were changed? Which are still original?

This first step of the LSS for file systems was implemented as a stand-alone application. Future versions should be directly embedded into the operating system,

making it possible to simply choose from the standard OS visualizations (icons, list or column) also a visualization called either scatter, scattertable, scatterspace or simply LSS. Another important issue is the improvement of the performance of the system.

The Liquid Scatter Space is a part of the "iworld" project, which is about a versatile multiple view knowledge browser with a special focus on semantic web browsing.

6. Acknowledgments

I would like to thank Iven Schmidt, Daniel Hess and Dieter Schuster. Without them the application for MacOSX would not exist in that quality at this point of time. Thank you also for everyone out there who supports our work by buying an iSaver or one of our other products from iStuff (www.i-stuff.de).

7. References

[1] Waldeck, Carsten: *Mobile Liquid 2D Scatter Spaces*. Proc. of Information Visualisation 2004, IEEE, London.

[2] Ahlberg, Christopher and Shneiderman, Ben (1994): *Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Star field Displays*. Proc. of CHI 1994. ACM, New York, pp 313-317.

[3] T. Keahey, *The Generalized Detail-In-Context Problem*. Proc. of IEEE Visualization '98, Information Visualization Symposium, IEEE Press, 1998.

[4] R. Rao, C. Stuart, *Exploring large tables with the Table Lens*, Human Factors in computing systems (CHI) - Conference Proceedings v.2., ACM Press, 1994, pp. 222.

[5] G. G. Robertson, J.D. Mackinlay, and S.K. Card, *Cone Trees: Animated 3D visualisations of hierarchical information*, Proc. of CHI 1991, ACM, pp. 189-194.

[6] J. Lamping and R. Rao, *The hyperbolic browser: a focus + context technique for visualizing large hierarchies*, Journal of Visual Languages 7, 1996, Academic Press.

[7] Johnson B, Shneiderman B (1991) *Treemaps: a space-filling approach to the visualization of hierarchical information structures*, Proc. of the international IEEE Visualization Conference 1991, San Diego.

[8] Bertin, Jacques. *Semiology of Graphics*. The University of Wisconsin Press, 1983.

[9] Apple Computer: http://developer.apple.com/cocoa/

[10] Pirolli P, Card SK, Van Der Wege MM (2003), *The effects of information scent on visual search in the hyperbolic tree browser*. ACM Trans Comput Hum Interact 10(1):20–53

[11] Cockburn A, McKenzie B, *An evaluation of cone trees*. In: People and computers XIV: British computer society conference on human computer interaction, Sunderland.

[12] Bederson, Benjamin and Hollan, James: *Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics*. Proc. of UIST 1994, ACM, New York.

[13] Bederson, Benjamin, *Fisheye Menus.* Proc. of UIST 2000), pp. 217-226, ACM Press.

[14] Sarkar, Manojit and Brown, Marc. *Graphical Fisheye Views of Graphs*. Proc. of CHI 1992, ACM, New York, pp 83-91.

[15] B.A. Nardi and C.L. Zarmer, "Visual Formalisms in User Interface Design", Journal of Visual Languages 4, 4 (March 1993), Academic Press, pp. 5-33.

[16] S.K. Card, J.D. Mackinlay, and B Shneiderman, *Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers, San Francisco, California, pp. 1-34, 1999.

Contact Information:

Carsten Waldeck
ZGDV - Computer Graphics Center
Dept. Z3, MIV (Mobile Information Visualization)

Fraunhofer Str. 5
64283 Darmstadt, Germany
Tel.: +49 (0)6151-155-623

http://www.zgdv.de
http://www.infoverse.org
carsten@infoverse.org