Interactive Visual Analysis of Multi-Parameter Families of Function Graphs

Krešimir Matković,
Josip Jurić, Zoltán Konyha
VRVis Research Center
{Matkovic,Juric,Konyha}@VRVis.at

Jürgen Krasser
AVL List GmbH
Advanced Simulation Technologies
juergen.krasser@avl.com

Helwig Hauser
VRVis Research Center
Hauser@VRVis.at

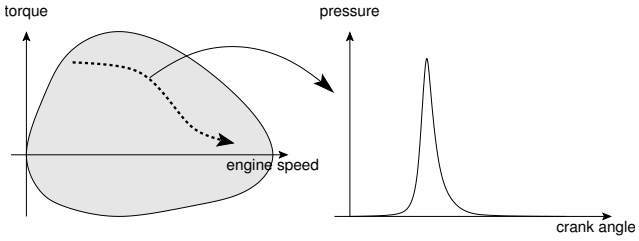

Figure 1. Left: an illustrative example of an engine characteristic diagram that shows torque versus engine speed. The engine's working point is always within the area shown in this diagram. The dotted line indicates the motion of the engine's working point as the car accelerates. Right: gas pressure in one of the cylinders versus crankshaft angle at one specific point in the engine characteristic diagram.

Abstract

The paper describes a method developed for interactive data visualization and exploration with applications in the automotive industry. The input data set contains a large number of function graphs. Each of the graphs is characterized by a set of basic attributes. The technique that is used for visualization includes two linked views: a map view (or attribute space view), where all function graphs are represented as a point or an icon on the map, and a linked function graph view. The map view provides additional visualization possibilities and allows user interaction. Additional features like brushing in both views, graph management, and related issues like interpolation of the graphs are described.

Keywords— **interactive visualization, linked views, brushing, time series data, interactive exploration, interpolation, Kennfeld diagram**

1. Introduction

The amount of information we are being confronted with nowadays increases constantly. Effectively exploring and analyzing huge amounts of data is a challenging problem, perhaps even more than collecting or processing this data.

Plain data tables or simple 2D charts are not always very useful when it comes to large data sets. Furthermore, people often look for various correlations and want to compare specific data. Information visualization tries to make it easier for the user to analyze and explore large data sets by cleverly and interactively displaying information [5, 3, 12, 13].

Information to be explored can come from various disciplines and application areas and there is no universal solution that can fit all of those areas. In this paper we describe a method which can be used to analyze and explore data sets containing families of function graphs, where each of them corresponds to a point in a 2D domain. The data set looks like this: we have a 2D domain and for each point in this domain (or for some of the points) there is a corresponding graph describing an attribute in tw0 other dimensions.

One easily comprehendable example of such a data set is a diagram of temperature over time measured on a number of places within a certain geographical area. The geographical map represents the 2D domain, and the temperature vs time data at specific points on the map represent additional two dimensions of the data set. Figure 2 illustrates the example.

Another example of a similar data set comes from automotive engine design. An *engine characteristic diagram* is a 2D diagram that shows engine speed (in rpm) on the X axis and torque at the crankshaft on the Y axis. At every engine speed there is a maximum torque that the engine can produce when the the accelerator is fully depressed. There is also a minimum torque (with the accelerator not depressed at all) which is negative and can be used to decelerate the

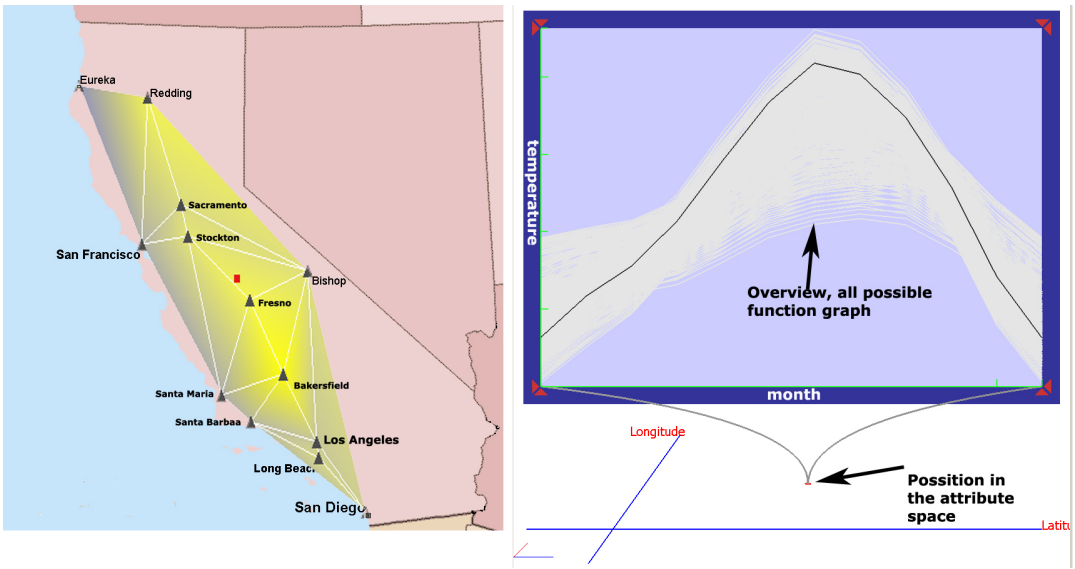

Figure 2. Left: attribute space view and right: function graph view.

car. While driving a car the working point of the engine follows a path within the engine characteristic diagram.

An example of the engine characteristic diagram is shown in Figure 1. For each point of this diagram there are several interesting properties to be investigated, for instance gas pressure in cylinders or main bearing forces as functions of crank angle.

Time-series are a special kind of function graph. This special case has been dealt with before. *Time Searcher* by Hochheiser et al. [9] is used to interactively explore time-series data. Time-boxes are used to specify queries. Time-boxes can be manipulated to specify various times.

Van Wijk and van Sellow [14] have used calendar view together with time-series data. The main scope was again various time series data. Since data was gathered for various days, the calendar was used to visualize time-series data origins. The authors also showed how clusters can help in finding similar time-series in the data set.

Li Zhang at al. [15] used higher Fourier harmonics to enhance the visualization of time series data.

2. Motivation

When users work with large sets of function graphs associated with points in the attribute space they often want to explore the data first. They try to find locations in attribute space that generate function graphs which have some interesting or unusual property, or conversely, ones that exhibit some regularity or follow a pattern. They often have little a priori knowledge about the data set. They do not exactly know what pattern or property they are looking for, or where the feature is likely to appear in the attribute space.

This calls for a visualization method that allows quick and easy, yet accurate exploration of the data. Two simultaneous views of the data set should be provided: (1) a good overall view of all function graphs to support navigation in the data set as well as (2) a detailed view which offers more information about specific graphs. Picking graphs from the attribute space must be made as effortless as possible and should require minimum user interface action.

After some interesting locations have been identified in the attribute space, another typical task is the detailed comparison of graphs associated with these areas. In order to perform this efficiently one must be able to observe the specific graphs at the same time and still not loose the context where the graphs belong [4].

Working with static 2D charts is obviously not practical if the attribute space is large. For example, numerical simulations where a large number of parameterized cases are calculated produce large sets of result graphs. These graphs must be examined in search of model parameters satisfying a certain criteria. It is clearly impossible to explore a set of thousand function graphs without the possibility of navigating the data set interactively. More complex queries like ”show me points in the attribute space whose function graphs fulfill this-and-that criteria” are obviously even less feasible without interactive visualization methods.

Although displaying the data set as a 3D animated sequence where the time represents one of the coordinates is an obvious approach, it can only be useful in some cases. It is suitable to offer an overview of the data set but it is less practical for interactive navigation. It is even less suitable to show detailed and accurate information about graphs of specific points in the attribute space. In cases when the spatial distribution of the data at a specific position in time is less important than the temporal behaviour of a specific point in the attribute space, this way of displaying the data may even be inappropriate.

Therefore we suggest a different approach. Naturally, one must note that there is no perfect solution which could aid all visualization tasks even for one specific type of data. Existing methods that show animated views for a better overview in time domain lack numeric accuracy. Sheets of numeric data and charts are accurate but lack overview. We assume that the method suggested in this paper could complement these existing methods and offer an overview and detailed information simultaneously.

This is a way to explore the data, in particular when many variants have to be examined, which is a typical scenario in optimization, for example. Interaction opens new ways of data exploration.

3. Technology content

The main idea is to have two linked views: the attribute view and the function graph view. The attribute view displays the domain (map, engine characteristic diagram,...) while the function graph view displays the data available for each point of the domain (temperature over time, cylinder pressure over crank angle,...). The attribute view can contain additional information such as glyphs for points where actual data is available, or a color-map of some interesting data characteristic (e.g. the maximum, the average, or mean value of the data). The system is interactive and allows the user to navigate and select points both in the attribute view and in the function graph view. The function graph view can display more graphs simultaneously in 2D or, optionally, in 3D.

Furthermore, the function graph view contains an overview part (showing the actual position of the selected point in the attribute space) and the graph corresponding to that point. Besides multiple selection in the attribute view, a collecting principle is described as well. The user can collect various graphs during the exploration, and compare them afterwards.

Finally, we describe an application which is implemented based on above mentioned principles. The application is used to visualize engine characteristic diagram. It has been developed together with AVL [2], one of the leading companies in the field of powertrain simulation, measurement and design.

3.1. Basic idea

We describe a general framework for interactive analysis and exploration of data sets containing a family of function graphs each of them belonging to a point in 2D domain. The main idea is to have an interactive visualization method, which will make it possible to easily select a point in the attribute space and get the corresponding graph. Furthermore, the inverse process, selecting a function graph with a certain characteristic in order to locate similar graphs in the domain should also be possible.

3.2. Function graphs and interpolation

The data set we concentrate on consists of a number of function graphs:

$$f_P(x)$$

Each of the function graphs is characterized by a set of parameters (or attributes) $P = \{p_1, p_2, ..., p_N\}$. In a general case this gives N-dimensional attribute space. The function graphs from the input data set are represented as points in this N-dimensional space and they are scattered non-uniformly across this space. The complete input data set can be formalized as:

$$\mathcal{F} = \{f_{Pi}\}$$

In this paper we focus on a case where 2 out of N parameters are visualized in the attribute view. This gives us a 2D space which can be thought of as a map. The other linked view displays interactively selected function graphs.

In most cases the function graphs are not defined for each point in the map view. If the defined points are dense enough, it might be sufficient to choose the nearest defined point. However, there might be some applications where only few points in the 2D domain are available. These points can be scattered in the parameter domain on an unstructured grid and just picking the nearest defined graph when performing a selection in the map view is not always satisfactory.

The natural solution is to provide interpolated curves at all points within the map view for which the data is not available [11]. Of course, the interpolation can only be done if the nature of the data allows it. The first requirement is that the data set is continuous.

Let us introduce the interpolation operator Ω :

$$\begin{aligned} \hat{f}_P(x) &= \Omega(\mathcal{F}, P) \\ &= \Omega_P \mathcal{F} \end{aligned}$$

The unknown function graph $\hat{f}_P(x)$ can be interpolated from the graphs of the neighboring data points with a method that best fits the data.

To enable efficient interpolation, the Delaunay triangulation is performed as a first step on all the defined points within the 2D map .

If a graph $\hat{f}_P$ at point $\hat{P}$ needs to be interpolated the following steps are performed:

1. The triangle $\triangle(P_1, P_2, P_3)$ containing the point $\hat{P}$ is located.

2. The homogeneous barycentric coordinates (t_1, t_2, t_3) of the point $\hat{P}$ are computed. Homogeneous barycentric coordinates are barycentric coordinates normalized such that they become the actual areas of the subtriangles $\triangle(\hat{P}, P_2, P_3)$, $\triangle(P_1, \hat{P}, P_3)$, $\triangle(P_1, P_2, \hat{P})$ normalized by the area of the original triangle $\triangle(P_1, P_2, P_3)$. These coordinates are also called *areal coordinates* [6].

3. The resulting function graph is computed as the weighted sum of the three related function graphs using homogeneous barycentric coordinates as weighting factors.

$$\hat{f}_P(x) = t_1 f_{P1}(x) + t_2 f_{P2}(x) + t_3 f_{P3}(x)$$

Using linear interpolation for interpolating temperature data on a map is likely to be a valid approach, assuming the data is not sampled too sparsely. In this case trilinear interpolation of the graphs of the three nearest points on the map is an acceptable way to fill in the missing data.

Interpolating gas pressure curves in an engine characteristic diagram is an interesting example where a more sophisticated interpolation method is required. Using a simple linear interpolation may lead to incorrect results due to the specific shape of the curve (Figure 3), and possible phase shifts between the cylinder pressure curves at various engine speeds.

One possible solution to the problem is to align all involved cylinder pressures by cyclic shifting so that the x - coordinate of the peak value for all involved curves matches. The phase shift of the resulting graph is interpolated between the phase shifts that were used for the initial alignment using the same weighting factors (t_1, t_2, t_3) (Figure 4).

Another solution to this problem is to use interpolation in frequency domain instead of direct interpolation. The function graphs are transformed to the frequency domain first, then the weighted sum using homogeneous barycentric coordinates is calculated for phase and amplitude data before converting them back to the original domain. Both methods are specific and usable for cylinder pressure diagrams because the nature of the diagram allows and requires such interpolation.

3.3. Visual metaphor

Since there is an attribute space and a family of graphs we propose to use two views for visualization [14].

The attribute space view is used to depict the domain. Points where function graphs exist are of main interest in the attribute space. If there are just a few points, all of them will be depicted. In case of large number of points (maybe even more points per pixel), only a subset will be shown.

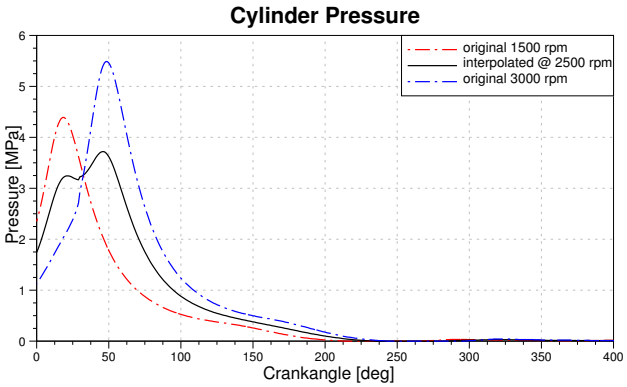

Figure 3. An erroneous interpolation of cylinder pressure. There is a phase shift between the pressure graphs (dashed lines) of different crankshaft angles, thus peaks are at different angles. Simple linear interpolation produces a graph (plain line) with two local maxima—completely unrealistic of cylinder pressure.

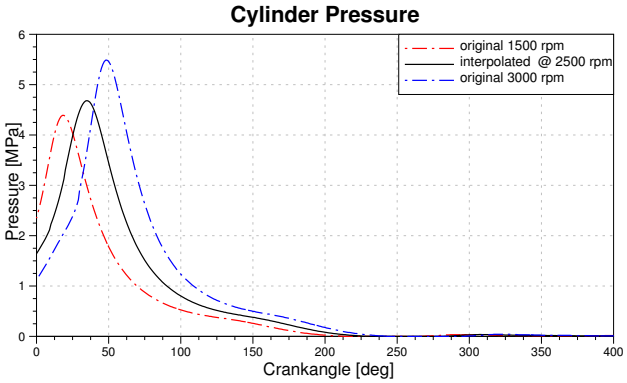

Figure 4. The correct interpolation of cylinder pressure. The peak values of the pressure graphs are aligned before linear interpolation. The resulting interpolated pressure graph has a very natural shape. Compare this to Figure 3

Furthermore, one characteristic of the corresponding function graph, such as maximum value, minimum value, average, etc. can be shown in the attribute space as well. Trilinear color interpolation will be used to show this characteristic. If the points are too sparse, additional points have to be inserted into the triangulation in order to avoid the visual artifacts of the trilinear interpolation. To create these additional nodes the interpolation described in Section 3.2 is used–same as when interpolating the data for the function graph view.

The second view is used to show function graphs. It shows function graphs in a plane. Besides the current function graph, all function graphs are shown in light gray in the

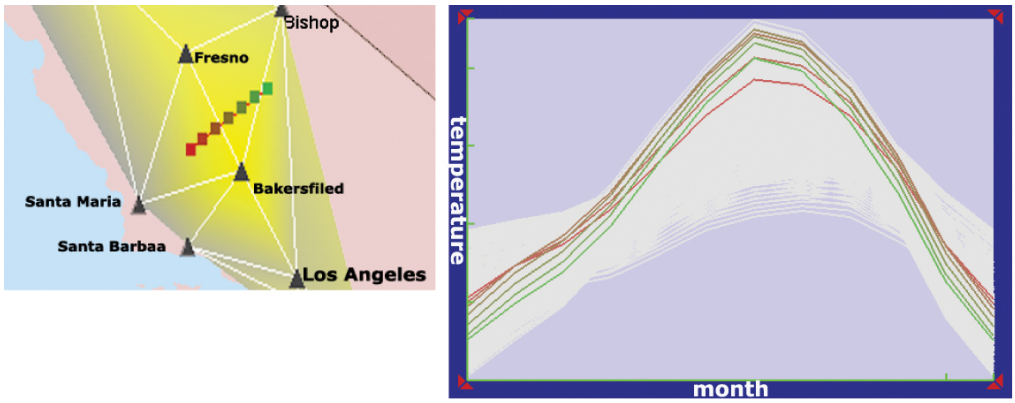

Figure 5. A 2D display of function graphs that belong to a line in the attribute space. Note that the overview is depicted as context and the colors of the seven function graphs match those of the points in the line they are associated with.

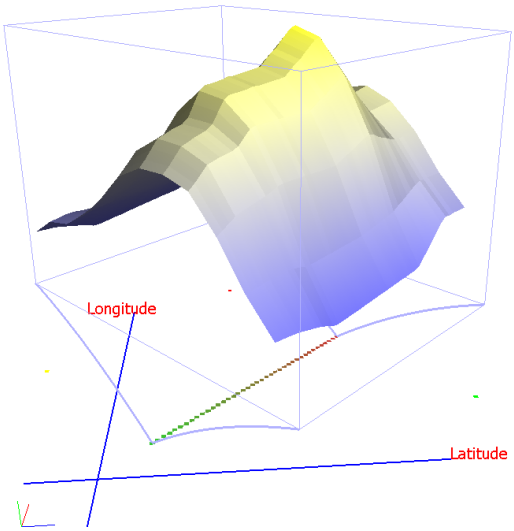

Figure 6. A 3D display of function graphs that belong to a line in the attribute space.

same plane as an overview. Furthermore the 2D attribute space is depicted perpendicular to the plane, and the origin point of the highlighted graph is shown as well. This additional overview information helps the user in seeing a particular graph in the overall context. Note this graph's source is highlighted in attribute space itself as well. This dual highlighting helps the user to keep concentrated on the graph view, and still have an overview of the graph's location in the attribute space. Without the overview feature, the user would have to repeatedly change focus from graph view to the attribute view. Figure 2 shows the attribute view, with the maximum value depicted using a color scale, and the function graph view with overview and one curve highlighted. A real map image has been used as background for the attribute view.

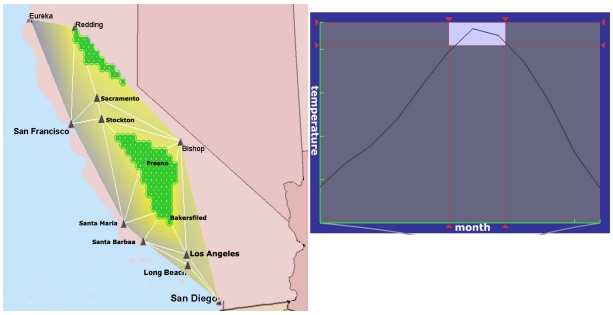

Figure 7. A brushing example. In the function graph view the user has highlighted a rectangle to select very high temperature in July. All points whose temperature graphs pass through the highlighted rectangle are marked in the attribute view to display areas of high July temperature.

3.4. Workflow and interaction

The usual workflow while exploring a data set consisting of family of function graphs involves two main actions:

- attribute based curve exploration, and
- locating graphs in attribute space based on their properties.

This means that user usually browses through the attribute space and observes the function graphs, and later selects some characteristics in the function graphs and finds corresponding positions in the attribute space. The straightforward interaction for the first action is "mouse-over", which depicts a function graph corresponding to the point that is currently under the mouse pointer. Depending on the mouse pointer position, this can be either an existing function graph or an interpolated one. This information should also be displayed to the user.

Seeing only one graph at a time makes it difficult to compare the curves. A method for selection in attribute space is

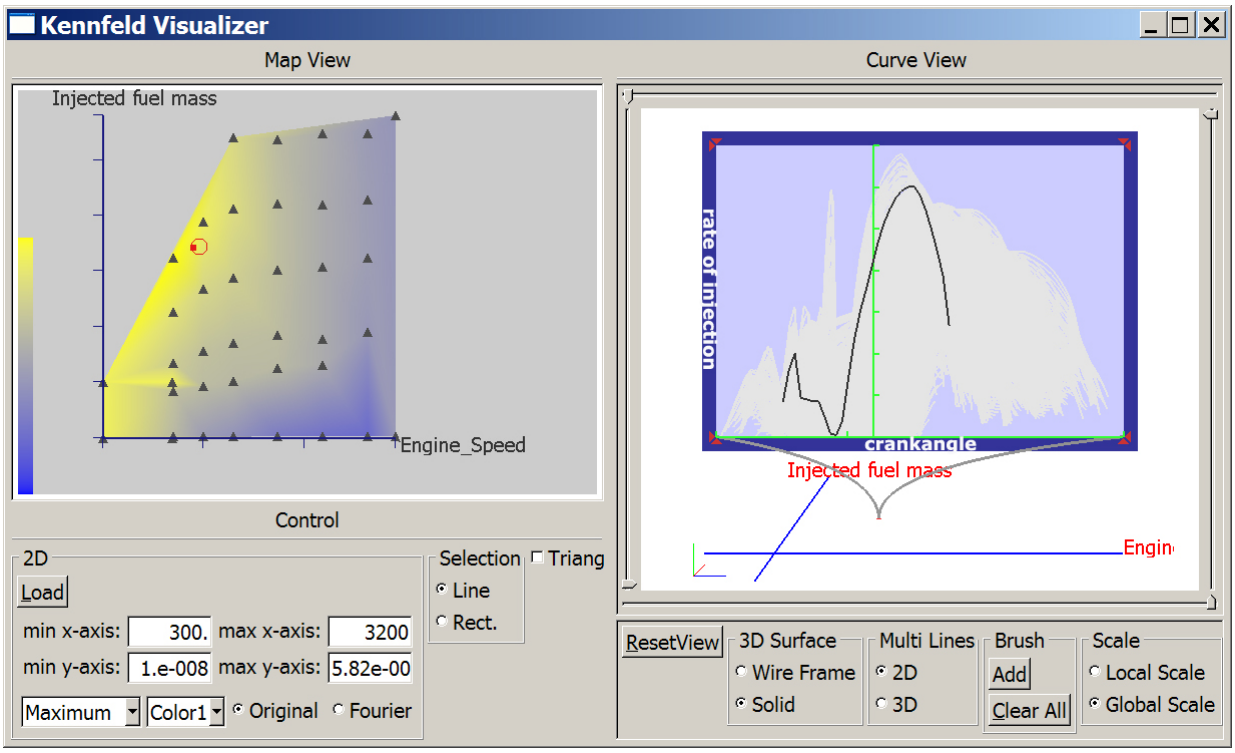

Figure 8. The typical shape of an injection curve. Two peaks, the smaller pre-injection and the main injection, are clearly visible.

required. There are numerous possibilities to select multiple points in attribute space. A line selection method which allows the user to select a line in the attribute space is the simplest. If we want to have separable, independent axes in the attribute selection, a rectangle selection with the axes aligned with the attribute space is needed. With both line and rectangle selection, the selection can be colored using a color gradient. The color gradient provides additional information about the sources of the function graphs. The color range for this gradient is chosen to be distinct from the color coding used in the attribute view.

When the user selects multiple points in the attribute view, more function graphs are displayed simultaneously. Each graph is drawn in the color of its selection point. Figure 5 illustrates a simple line selection, using the temperature data set. Note that the colors of the function graphs are the same as that of the corresponding points in the selection in attribute space. Having multiple function graphs corresponding to a line in the attribute space, the resulting curves can be extruded to create a 3D surface. 3D surfaces can offer new insights into the data. Figure 6 shows another line selection depicted as 3D surface. Note that extrusion to 3D is possible only for line selections in the attribute view.

It is possible to combine more selections using Boolean algebra, which adds additional possibilities to the data exploration and analysis. Selections are very powerful for exploration, but they have a limited lifetime. When the user makes a new selection the old one is lost. That is why a collection principle is introduced. Collections are actually containers that hold a number of graphs. As the user browses the attribute space and finds a graph of interest he can add it to a collection and continue browsing. Graphs can be added to existing collections at any later point. Several collections may be defined and used simultaneously, much like having multiple clipboards in a text editor. It is also possible to create new collections via Boolean operations on other collections. After the user has created collections he can simply display and explore them.

Once a selection is made or a collection is displayed, the user can start working with the function graph view. The user often wants to locate graphs in the attribute space with certain characteristics. The use of a rectangular brush [1] [7] for selection of the interesting areas within the function graph view provides an excellent way to do this kind of analysis. The points corresponding to the graphs passing through the selected rectangle are highlighted in the attribute space. For instance, the use of brushing makes it easy to find areas on the map that have average temperatures in July over some threshold. Figure 7 shows such an example.

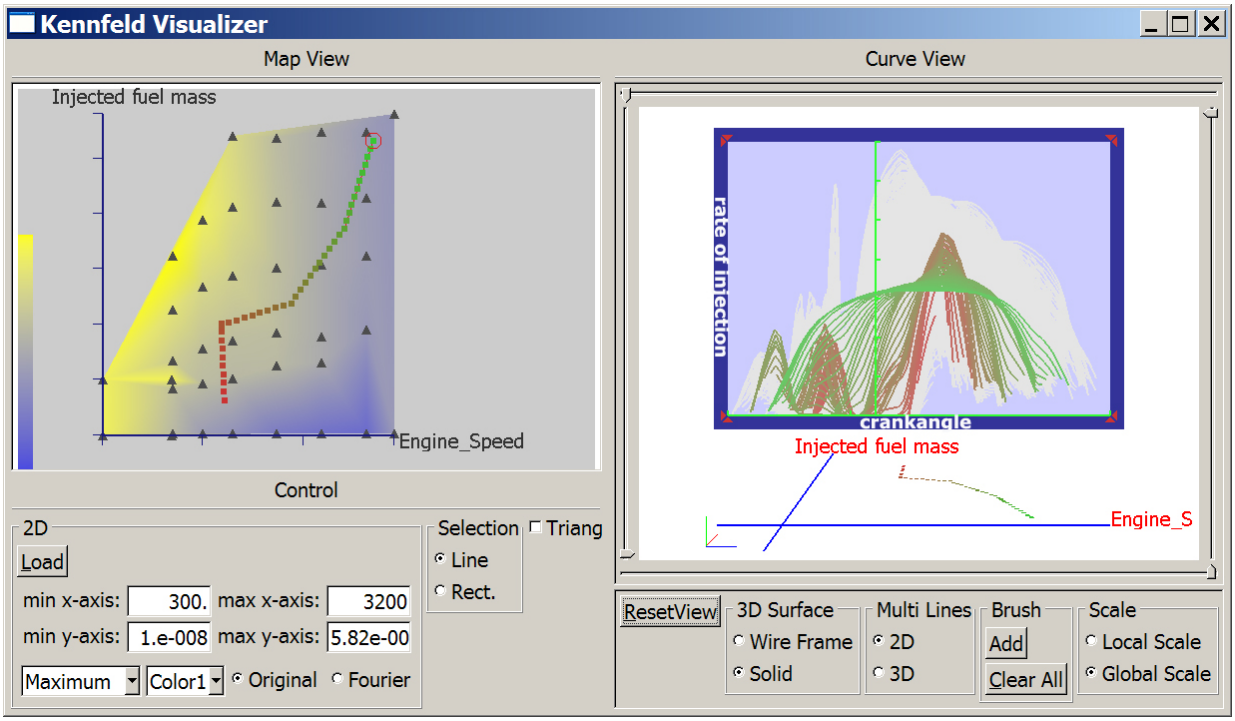

Figure 9. Polyline selection in the attribute space. The data points of the line are drawn in colors ranging from red to green. Each function graph is drawn in the same color as the point it is associated with to help the user identify which point the graph belongs to.

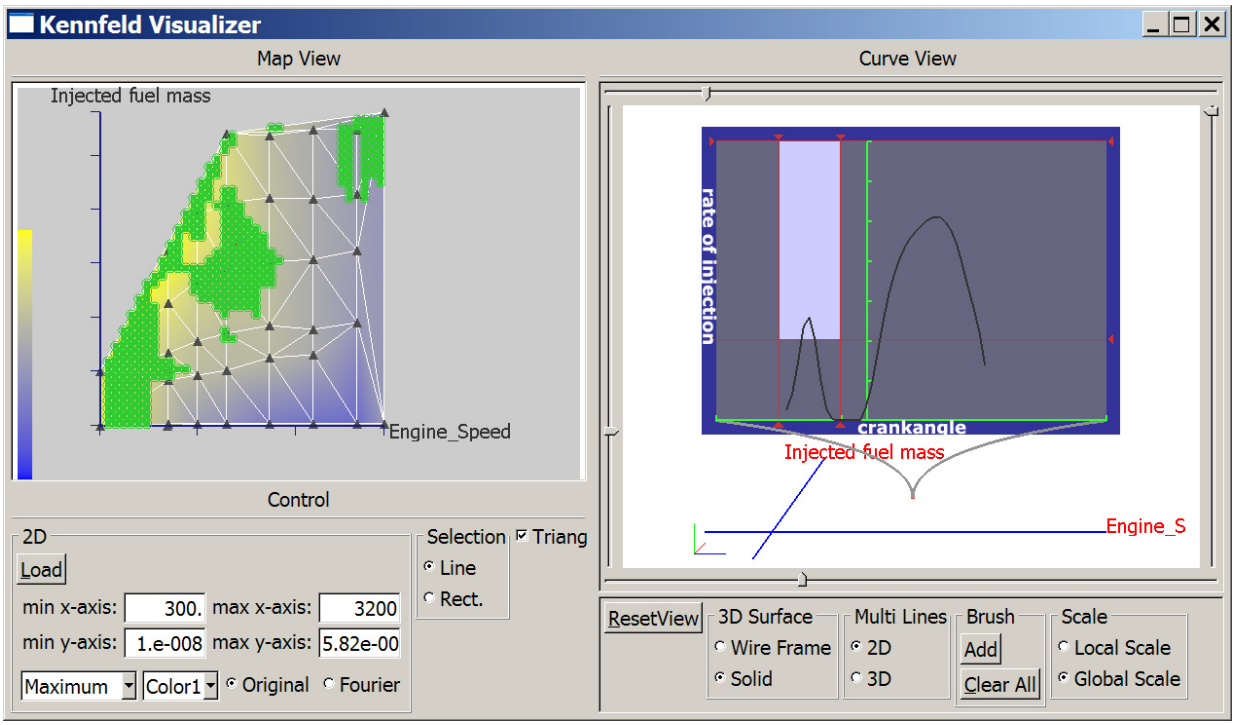

Figure 10. A brushing example. The user has highlighted areas where pre-injection ratio is above a certain threshold.

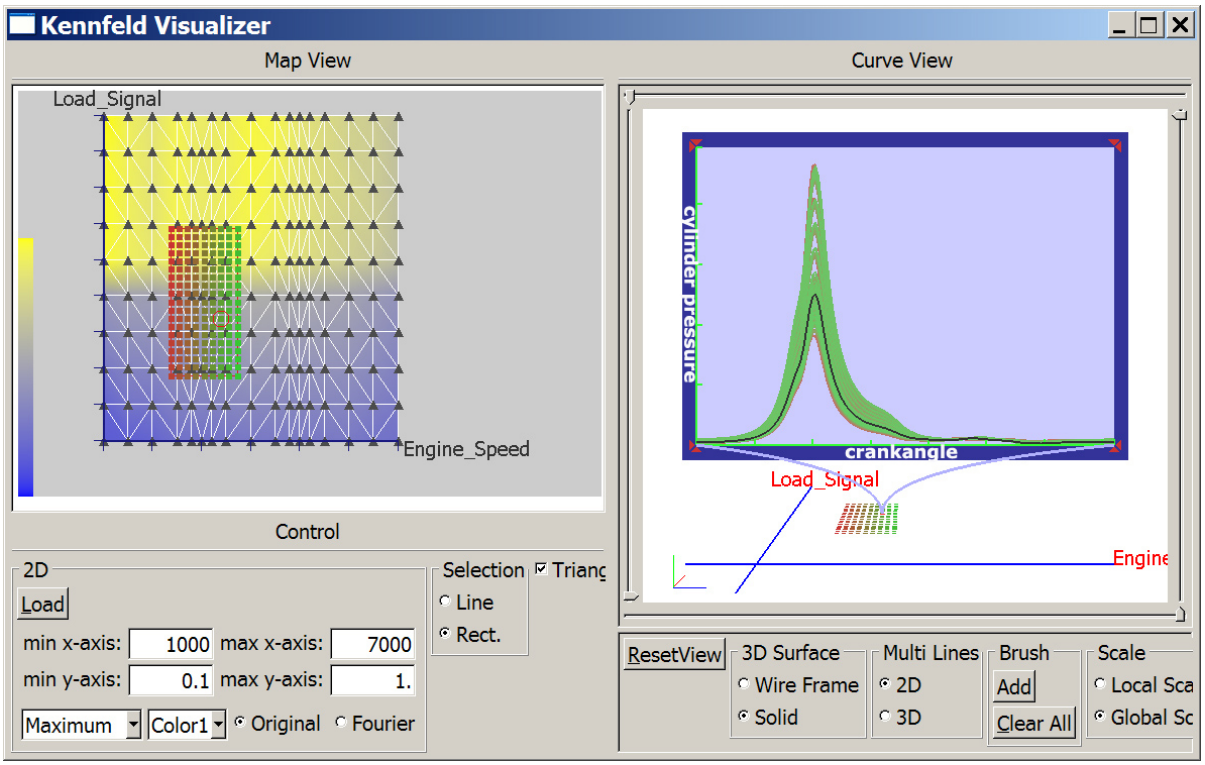

Figure 11. Rectangle selection in attribute space. The curves in this example depict pressure in cylinders versus crankshaft rotation angle. The attribute space is defined with load signal (which is proportional to the pressure on the accelerator) and engine speed in rpm.

4. Application

We have applied the above described principles in the *KennfeldView* tool developed together with AVL. The application is implemented in C++ using OpenGL [10] and FOX Toolkit [8]. The tool is available on several platforms, including MS Windows and Linux. Primary application of the tool is the exploration and analysis of engine characteristic diagrams. After the data is gathered either by simulation or measurement on real engines, the user loads the data, and starts exploration. The attribute view can display maximum, minimum, average and mean value of the corresponding graphs as underlying color map. The *KennfeldView* supports three types of interpolation: linear, modified linear and interpolation in Fourier space. A proper interpolation has to be selected based on the input data. In order to perform the interpolation the input points in the attribute view are tessellated. The user can choose to display the tessellation of the attribute view, or the icons on the points where graph data is available, or both.

Once the data is loaded and interpolation type, color scale and an underlying map are chosen, the user can start data exploration. While moving the mouse pointer over the attribute view the corresponding graphs are instantly displayed in the graph view. Single or multiple selections, as described in Section 3.4 are supported. If the user chooses to display multiple selections as a 3D surface (possible only for line selection) the surface can be displayed either as a surface or a wire-frame. The camera can be freely moved in this mode. A 2D display of multiple selections does not allow camera movement, but allows brushing in the graph view, which is a very important feature for engineers.

Let us show how an actual data exploration might look like: we have two data sets, the first describing the rate of injection as a function of a crankangle. Rate of injection indicates how fuel mixture is injected into the cylinder. Figure 8 shows a characteristic injection curve. Two peaks are clearly visible, the first one, so called *pre-injection*, and the second one, the *main injection*. Note that this form of injection curve is desired and engine designers try to achieve such engine characteristics. On the left hand side in the figure 8 the corresponding attribute space is shown, which depicts injected fuel mass and engine speed in rpm. The depicted attribute space represents the working range of the engine. The working point of the engine can be only be located within the depicted area in the attribute view. The engineers can use the tool to explore injection curves at various working points and to quickly see if there are some areas where the curve has only one peak (undesired curve shape), or has too low pre-injection ratio. Figure 9 shows an

example of a polyline selection in attribute space. Note how the color changing from red to green aids the user in identifying areas that have an undesired (single) peak curve shape in the selection.

An inverse kind of exploration is shown in Figure 10. In this case we have selected the area where the pre-injection amount exceeds certain value. We can clearly see all the areas in the attribute space where function graphs having this property are located.

Finally, Figure 11 shows the cylinder pressure in the function graph view and the working area of the engine in the attribute view described by the load signal and engine speed. The attribute space determines the domain where the operating point of the engine may be while the engine is running. The load signal is directly proportional to the pressure on the accelerator. A rectangle selection with gradient brush is shown, which is a good way to explore tendencies with this kind of data.

The *KennfeldView* tool has been recently demonstrated to a group of mechanical engineers at AVL List GmbH. They had a good initial impression about the application and are interested in integrating it into one of AVL's software products.

5. Summary and conclusion

We have presented a method to visualize a family of function graphs where each of them is associated with point in a 2D domain. Such data sets are ubiquitous, from all kinds of data bound to the map of an area (meteorological data, population, pollution, etc.), to special cases in engineering, like the working domain of the car engine linked to various crank angle dependent attributes at each operating point. We have described an interactive, dual linked view solution consisting of an attribute view and a graph view. Users can browse through the data by simply moving the mouse pointer over the attribute view and explore related function graphs in the graph view. Brushing in the graph view adds additional dimension to the exploration by highlighting the locations of graphs with certain characteristics on the 2D map. The engineers at AVL dealing with engine design are pleased with the new interactive solution. It has many advantages over the previously used static 2D charts. *KennfeldView* may evolve to become a component of AVL's software suite.

Additional color images are available for download at `http://www.vrvis.at/scivis/kennfeld`.

6. Acknowledgements

The authors express special thanks to Andreas Münzer of AVL for supplying the engine characteristic diagram data set. We also thank Robert S. Laramee of VRVis for his for his valuable feedback. Parts of this work have been carried out in the scope of applied and basic research at the VRVis Research Center which is funded by an Austrian governmental research program called K plus (`www.kplus.at`).

References

[1] C. Ahlberg and B. Shneiderman. Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *CHI '94: Conference companion on Human factors in computing systems*, page 222. ACM Press, 1994.

[2] AVL List GmbH, http://www.avl.com.

[3] B. Bederson and B. Shneiderman, editors. *Readings in Information Visualization: using Vision to Think*. Elsevier Science & Technology Books, 1999.

[4] A. Buja, J. A. McDonald, J. Michalak, and W. Stuetzle. Interactive data visualization using focusing and linking. In *Vis '91: Proceedings of the 2nd conference on Visualization '91*, pages 156–163. IEEE Computer Society Press, 1991.

[5] S. K. Card, J. Mackinlay, and B. Shneiderman, editors. *The Craft of Information Visualization: Reading and Reflections*. Morgan Kaufmann Publishers, 2003.

[6] H. Coxeter. *Introduction to Geometry (2nd ed.)*. Wiley & Sons, 1969.

[7] K. Fishkin and M. C. Stone. Enhanced dynamic queries via movable filters. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 415–420. ACM Press/Addison-Wesley Publishing Co., 1995.

[8] FOX Toolkit, http://www.fox-toolkit.org.

[9] H. Hochheiser, E. H. Baehrecke, S. M. Mount, and B. Shneiderman. Dynamic querying for pattern identification in microarray and genomic data. In *Proceedings IEEE International Conference on Multimedia and Expo*, Baltimore, MD, 2003. IEEE Computer Society.

[10] OpenGL, http://www.opengl.org.

[11] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524. ACM Press, 1968.

[12] E. Tufte. *Envisioning Information*. Graphics Press, Cheshire, CT 06410, 1990.

[13] E. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT 06410, second edition, 2001.

[14] J. J. V. Wijk and E. R. V. Selow. Cluster and calendar based visualization of time series data. In *InfoVis '99: Proceedings of the 1999 IEEE Symposium on Information Visualization*, page 4, Washington, DC, USA, 1999. IEEE Computer Society.

[15] L. Zhang, A. Zhang, and M. Ramanathan. Enhanced Visualization of Time Series through Higher Fourier Harmonics. In M. J. Zaki, J. T. L. Wang, and H. T. T. Toivonen, editors, *Proceedings of the 3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics*, 2003.