Interactive Visual Analysis of Families of Function Graphs

Zoltán Konyha, Krešimir Matković, Denis Gračanin, Mario Jelović and Helwig Hauser

Abstract— **The analysis and exploration of multidimensional and multivariate data is still one of the most challenging areas in the field of visualization. In this paper, we describe an approach to visual analysis of an especially challenging set of problems that exhibit a complex internal data structure. We describe the interactive visual exploration and analysis of data that includes several (usually large) families of function graphs $f_i(\mathrm{x}, t)$. We describe analysis procedures and practical aspects of the interactive visual analysis specific to this type of data (with emphasis on the function graph characteristic of the data). We adopted the well-proven approach of multiple, linked views with advanced interactive brushing to assess the data. Standard views such as histograms, scatterplots, and parallel coordinates are used to jointly visualize data. We support iterative visual analysis by providing means to create complex, composite brushes that span multiple views and that are constructed using different combination schemes. We demonstrate that engineering applications represent a challenging but very applicable area for visual analytics. As a case study, we describe the optimization of a fuel injection systems in Diesel engines of passenger cars.**

Index Terms— **visual exploration, composite brushing, linked views, time series data, fuel injection system**

I. Introduction

THE development of effective visualization and interaction techniques requires the understanding of the properties of the data and the typical tasks the users want to perform [1]. Unfortunately, this requirement is not always met, often because of insufficient collaboration and communication between visualization experts and the users. The users' ultimate goal is always to find expected phenomena to support (or reject) their hypotheses or to discover unexpected results that question their assumptions or the validity of the data acquisition process. That can lead to the generation of new hypotheses.

The challenges of data analysis and exploration are associated with very large data sets, increased dimensionality and the consideration of data semantics, including features, focus and context [2]. Therefore, a visualization tool should be designed in close collaboration with potential users. Tool developers must be aware of the users' actual requirements, the usual tasks they need to solve, the shortcomings of their previously used tools, and their feedback on new ideas. A part of that process is a development of intuitive and effective visualization and interaction techniques based on a common data model. If designed well, the same principles can be used across several application domains, from real-time data monitoring to engineering design applications, including simulations.

Modern simulation software can generate massive amounts of data that require suitable analysis techniques to get an insight into the practical implications of the results. Simulation is increasingly used to assess the quality and potential of new designs early in the process (e.g. aircrafts and cars). Building real prototypes is time-consuming and expensive. Even though measurements on test bed systems are likely to remain an important way to verify designs in the future, the use of computational simulation in the design and production process can help to minimize the costs of the development and shorten the time-to-market for new products.

For example, in the automotive industry many different aspects of new designs are checked using simulation long before a new car is manufactured. Examples include mixture formation and combustion, engine cooling and filter regeneration, air conditioning in the passenger cabin and front shield deicing, and many others. The increasing complexity of automotive subsystems, e.g., the power train, the intake and exhaust system, or the fuel injection subsystem, also requires simulation for optimization. Tuning of an injection system for modern cars is an example of a multi-parameter optimization process. The operation of the injection system depends, in a very indirect way, on several parameters. Therefore, optimization by experience and/or intuition is usually not possible.

In this paper, we present a new approach to the interactive visual exploration and analysis of measurement and simulation data. This approach is general enough for a number of application scenarios that share the same characteristics (including multi-parameter tuning problems). A major challenge (in general) is how to visually relate the multivariate dependent variables to their multidimensional reference parameters (independent variables). We suggest a combination of different kinds of views with specific brushing interactions, all adapted to work well for the families of function graphs in order to facilitate the interactive visual exploration and analysis of such data sets. We have investigated the usability of our ideas in two very different settings: the analysis of road traffic data and the optimization of a fuel injection system. The road traffic data set serves as an illustrative example for the introduced concepts while the fuel injection system data set provides a case study, described in detail in Section VI.

The remainder of the paper is organized as follows. Section II provides an overview of related work. Section III gives a brief description of the data model used and the exploration procedures. Section IV describes our proposed tools and

Z. Konyha, K. Matković and H. Hauser are with the VRVis Research Center, Donau-City-Strasse 1, A-1220 Vienna, Austria. Email: {Konyha, Matkovic, Hauser}@VRVis.at.

D. Gračanin is with the Department of Computer Science, 660 McBryde Hall, Virginia Tech, Blacksburg, VA 24061, USA. Email: gracanin@vt.edu.

M. Jelović is with AVL-AST d.o.o., Av. Dubrovnik 10/2, HR-10020 Zagreb, Croatia. Email: mario.jelovic@avl.com.

methods for supporting these tasks. Section V introduces the typical tasks in the analysis of such data sets. Section VI describes the use of the developed approach for a real world automotive engine design task. Section VII provides closing remarks and directions for future work.

II. Related Work

Interactive visual exploration and analysis can benefit from previous results in many areas. We first address visual analytics and then provide an overview of visualization techniques for high dimensional data and the linked views principle.

Thomas and Cook [1] define visual analytics as "*the science of analytical reasoning facilitated by interactive visual interfaces*." It is a wide-ranging field of science that involves visualization and interaction methods combined with analytical reasoning, data representation and transformation as well as production and presentation of the results. Therefore it is difficult to find a previously published work that encompasses all aspects. We are able to collect and generate data at an increasingly fast rate, but capability of analyzing the collected data lags behind [3]. We focus on how users gain insight into data, find expected and unexpected features and make decisions using visual tools.

Trafton et al. [4] present a study of how experienced weather forecasters interpret complex visualizations to build their own qualitative mental models of weather conditions which in turn are the basis of the weather report they produce. In their experiments they show how the users perform *convergent thinking* (assembling evidence to support a hypothesis) and *divergent thinking* (thinking creatively to identify alternatives) at various stages of their work. Saraiya et al. [5] evaluate five visualization tools to determine which one provides the best insight into the specific data. González et al. [6] describe how an information visualization system was used by administrative data analysts. Ahlberg et al. [7] introduce a visual information seeking technique which focuses on rapid filtering and progressive refinement of search parameters.

There are numerous publications of having scientific visualization (SciVis) techniques applied to the visualization of simulation data with an engineering perspective. Laramee et al. [8] provide a thorough visual analysis of the coolant flow through the cooling jacket of a car engine by using various different flow visualization methods to reveal different aspects of the simulation data. Konyha et al. [9] propose 3D icons for the analysis of simulation data of chain and belt drives.

There are simulation data types that are more effectively explored by even more abstract visualization methods. In those cases the user may be able to gain more insight if information visualization (InfoVis) techniques are used instead of or together with SciVis methods. Matković et al. [10] describe a method for the analysis of a fuel injection system. The simulation time series data is reduced and described by a set of scalars which results in a highly abstract view of the injection system. The case study illustrates in detail how engineers can use the interactive, linked InfoVis views to explore and analyze simulation data of a fuel injection system.

The body of literature about the visualization of high dimensional data is vast. Following the terminology of Wong et al. [11] we focus on the visualization of *multidimensional* data. Keim [12] classifies methods for visualization of high dimensional data into four groups: geometric projections, hierarchical methods, iconic methods and pixel-based techniques. In our work we have considered only geometric projections so far. Geometric projections include two of the most popular information visualization techniques: scatterplots [13], [14] and parallel coordinates [15], [16], [17], [18], [19].

Using multiple, interactively linked views of the same data set allows the user to productively combine the information he or she gathers from the different views. The Attribute Explorer [20], [21] uses linked histograms to simultaneously represent the interaction between attributes and allow the user to narrow the focus by defining limits on certain attributes. The Influence Explorer [22] allows exploration of data computed from a model using given sets of parameter values as input. The user can select a set of points in either the parameter or the result spaces and see how this set corresponds to points in other dimensions in both spaces. Gresh et al. [23] present an approach that links 3D visualizations to statistical representations to facilitate effective exploration of medical data. Doleisch et al. [24] use multiple linked views, including adapted information visualization views in the analysis of CFD simulation data. Piringer et al. [25] interlink 2D/3D scatterplots and histograms with smooth brushing. Schafhitzel et al. [26] link several texture-advected flow visualizations on slices with the 3D view of the vector field in an attempt to overcome occlusion problems. Matković et al. [27] use Timebox-like [28] brushing to link the graph of a function to a scatterplot display of its parameter space. More advanced multiple view visualization systems can be configured freely to suit various data sets [29] and allow flexible coordination of views [30]. As the number of linked views and the amount of coordination increases it may become necessary to visualize the visualization's structure and operation [31].

III. Data Model

Generally speaking, a data model consists of a data definition and a manipulation language (structuring and operational definitions) [32]. Data definitions that result from an engineering simulation, a real-world sensor data set, or intelligence data may be very similar. Consequently, the data sets under consideration share some common characteristics. The data sets contain values for m independent variables and n dependent variables.

The independent variables $\mathbf{x} = [x_1, \ldots, x_m]$ and their values define a subset I of the data set. A member of $I \subseteq \Re^m$ represents a specific set of values $\mathbf{x}_i$ of independent variables. For each $\mathbf{x}_i$, the corresponding set of values of dependent variables is provided. There are two types of dependent variables, regular and function graphs. While regular variables have a singular value for each $\mathbf{x}_i$, function graph variables use time as an additional independent variable to provide a set of values for each $\mathbf{x}_i$. A function graph can be visualized as 2D plot that shows how the value of a dependent variable changes over time. In other words, the regular variables $\mathbf{r} = [r_1, \ldots, r_{n_r}]$ depend only on $\mathbf{x}$ while the function graph variables $\mathbf{f} =$

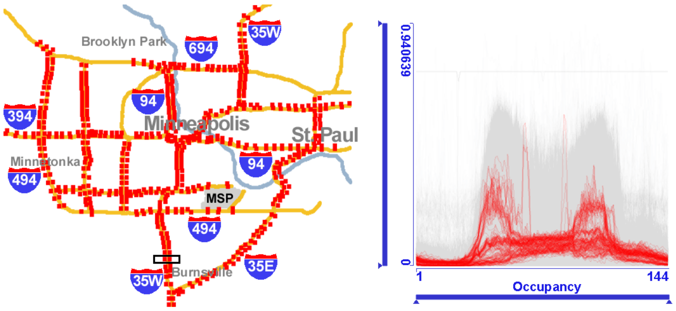

Fig. 1. Left: the map of all sensor locations in the Minneapolis freeway system traffic data. A schematic map is underlayed to provide context information. Each red dot marks the location of one *station* (which usually encompasses several sensors, *detectors*, one per lane). Right: road traffic occupancy family of function graphs. Data from some sensors (marked with the black rectangle on road 35W in the map) is highlighted in red. Occupancy is defined as a percentage of time a detector detects vehicles. It is measured in ten minute intervals. An occupancy value of 0.7 means that for seven out of ten minutes a sensor detected vehicles.

$[f_1, \ldots, f_{n_f}]$ depend on $\mathbf{x}$ and time $t \in \Re$. For a specific set of values $\mathbf{x}_i$ of independent variables and fixed time t_j we can define the set of values of dependent variables as $\mathbf{d} = [r_1(\mathbf{x}_i), \ldots, r_{n_r}(\mathbf{x}_i), f_1(\mathbf{x}_i, t_j), \ldots, f_{n_f}(\mathbf{x}_i, t_j)]$, $n_r + n_f = n$. The dependent variables and their values (possibly, over time) define a subset D of the data set. For a given function graph variable, $f_j(\mathbf{x}, t)$, we define a family of function graphs as a set of function graphs for each possible value of $\mathbf{x}$, $\{f_j(\mathbf{x}_i, t) | \forall \mathbf{x}_i \in I\}$.

Once the data set is defined, the question is how to analyze the data. In our data model, the manipulation language is an exploration language that enables search and pattern discovery without modifying the data set. From the visual analytics point of view, the goal is to discover, in an iterative manner, trends, tendencies and outliers in the data and to see how patterns in D map to the corresponding subsets in I and vice-versa. In order to achieve that, data exploration techniques must be conceptually simple, easily combined and visually intuitive.

The visualization framework is based on the described data model and a set of visual operators (brushing techniques) and views (histograms, scatterplots, parallel coordinates, etc.) that are linked together. The design of interactive visual analysis within this framework is based on the following principles. The analyst can select a varying number of views. Within each view, the variables of interest can be selected and the corresponding values displayed. The visual operators are used to select a subset of "interesting" values for the specific variables in the view. The selection is immediately displayed in all other views. Families of function graphs are of special importance in providing a visual space for patterns. Within a family of function graphs, we would like to select function graphs based on their shapes. It is possible to use a combination of function graph values to specify the desired shape of a function graph, i.e. the pattern.

We will use a real-world road traffic measurements data set to illustrate the concepts described in Sections IV and V. The data set is provided by the Traffic Management Center of Minnesota Department of Transportation [33] that maintains an archive database of road traffic measurements from the freeway system in the Twin Cities metropolitan area. The data set contains 28 days of measurements from approximately 4,000 sensors grouped into about 1,000 stations covering ten main roads. Opposite directions on a road (e.g. north-bound vs. south-bound) are treated separately, thus effectively creating 20 one-way roads. I consists of the positions of the sensors, road numbers and weekdays. The sensors report traffic volume and occupancy, thus D consists of two families of function graphs in this data set.

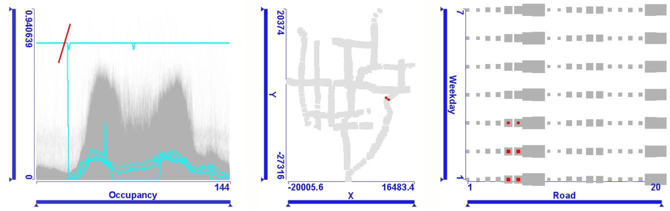

Fig. 2. Several occupancy function graphs of atypical shape have been selected by the red line brush. We conclude from very high occupancy values that those function graphs indicate malfunctioning sensors. In the linked map view (scatterplot view of sensor coordinates) we can see that there are two malfunctioning sensors next to each other. In another linked scatterplot view weekdays and road numbers are displayed. Each column represents one direction (for instance, south-bound) of a road. We can see that those sensors are on road 35E and that they did not work for three days.

IV. Tools for Analysis of Families of Function Graphs

We have developed a tool based on premises described in Section III. The combination of basic, highly interactive views is sufficient to carry out a wide range of sophisticated analysis tasks. Interactivity plays a crucial role in analysis. Important and novel aspects that support interactive procedures are described in the following.

We currently offer up to six linked views including histograms, scatterplots, parallel coordinates and function graphs. We do not make any assumptions about independent and dependent variables in the sense that we would restrict any of the basic view to display either of them. The inputs of the views can be mapped to any attribute of the data set, both independent and dependent variables. The user can arrange the views as desired, can have more than one instance of the same view type showing the same or different attribute sets. It is possible to temporarily maximize one view for more detailed examinations. Histograms, parallel coordinates and scatterplots are standard, well known views [34], thus we do not describe them here in detail. However, it is worth mentioning that the point size in scatterplot views can optionally be proportional to the number of data items represented by a single point. The more items a point represents the larger it is. An example is shown in Fig. 2: larger points indicate more sensors on the road. Similarly, the sizes of points highlighted in the focus set are also proportional to the number of items brushed (Fig. 11). Thus the ratio of brushed items versus context represented by a point in the scatterplot is indicated by point sizes.

The function graph view displays a family of function graphs at once. If the number of function graphs in the family is large then the display can become visually cluttered and

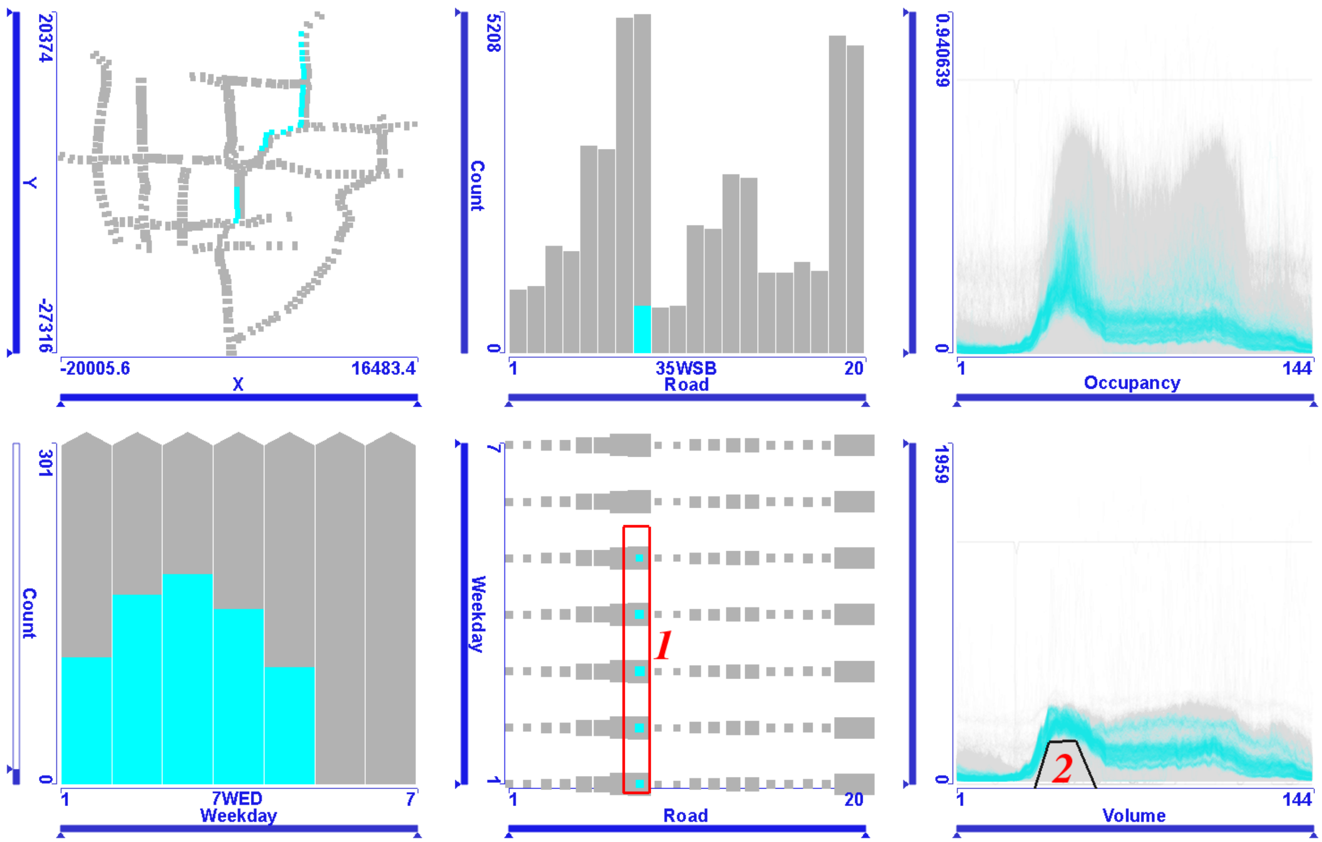

Fig. 3. A snapshot from an interactive visual analysis session of traffic data in the Minneapolis metropolitan area. The brush creation order is indicated by red numbers. Here we look for locations of high volume morning traffic on a given road on weekdays. The user has selected the road and weekdays in the scatterplot (brush 1) and then removed low traffic volume graphs intersecting brush 2 using SUB operation. The locations are highlighted in the linked map. The road number (35WSB, i.e. 35W south-bound) is shown on mouse-over. We can see that heavy morning traffic on south-bound 35W mainly occurs mainly north of the downtown, i.e., towards it.

non-informative. In order to represent the characteristics of the family better, we can (optionally) render the pixels through which more graphs pass through with higher intensity.

In the following we will describe generic interaction principles and elaborate on the specific requirements of brushing and linking in various views.

A. Generic Interaction Features

If the basic views listed above are independent then they provide a limited insight into the data set. However, if areas of focus can be highlighted with applicable brushing techniques and this focus area is linked to the other views, then correlations and dependencies in the data can be revealed. Our system supports interactive brushing and linking and the number of currently brushed data items versus total is always indicated. The user can perform brushing in any of the views and all the other views will also highlight the brushed items while the context is shown in a different, less saturated color. Whenever applicable, the view can be zoomed to show the brushed region only. The brushes can be resized and dragged to new locations which helps in the interactive data exploration. A tabular display of the currently brushed items can be opened when the user needs detailed numeric information.

With simple brushing and linking it is usually a problem to locate the matching brushed data items in different views. If more data items are brushed in a view then all corresponding items are highlighted in other views, but we cannot visually identify the same item in the different views. We have applied an optional *color gradient* along the brush and used this color gradient in the linked views to establish a visual identification of the correlated data items. That aids the user in discovering tendencies in the data set. Fig. 9 provides an illustration of the gradient brush.

Another improvement is *composite brushing*, a query tool which is a result of logical operations performed on brushes. Composite brushing makes it possible to build queries that specify several overlapping or intersecting ranges of criteria in the same or different views. We could have chosen to offer AND, OR and NOT operations to composite brushes and add a formula editor to allow controlling the order of operations by bracketing. In contrast, we use composite brushing similar as in the SimVis system [35] by offering AND, OR and SUB operations where the first operand is always the result of the latest composition. This allows a simplified, intuitive, and more iterative workflow compared to working with a formula editor. The user defines the first brush, then (optionally) selects a Boolean operation, adjusts the composition setting (to either AND, OR, or SUB) and then defines the next brush to adjust the current selection. Following brushes and operations will be applied to the result of prior brushes only. Iteratively, every new brush alters the current selection status according to the composition rule in use. The process continues in this way: new brushes and operations are applied to the latest state only. Each new brushing operation provides immediate visual feedback and the user can interactively refine (using AND and SUB) or broaden (using OR) the current selection and steer the information drill down. The user can also resize or move any existing brush in the chain to gain even more flexibility.

Brushing and linking is a powerful feature in understanding how outputs depend on inputs and finding input parameter sets when desired properties of outputs are known. Because we treat all parameters in the same manner, one can brush in views of independent parameters and study how dependent parameters change in other views, or perform the inverse kind of investigation to find suitable inputs for specified results by brushing in the views showing output parameters.

Brushing conventional views is quite straightforward. The user can select histogram bins, rectangular areas in scatterplot views or ranges of a parallel coordinates axes. We have introduced a novel brushing tool in the function graphs view and it will be described in more detail.

B. Brushing Function Graphs

We suggest two brushing methods to meet the specific requirements of queries on families of function graphs.

A *line brush* is a simple line segment drawn in the function graph view. It selects all function graphs that intersect the line Fig. 2 shows an example of selecting several graphs that have high and constant occupancy value indicating malfunctioning sensors. Linking them to the corresponding points in I in the map, we identify those sensors. It is very easy to exclude outliers in a family of function graphs or to isolate function graphs with desired characteristics with just a few line brushes (brush 2 in Fig. 3). Additionally, it is very useful to provide a polyline brushing opportunity, i.e., a brush in the form of a polyline which selects all function graphs which intersect any of the polyline segments. The line brush, together with the above-mentioned composition functionality, assist the user when looking for function graphs whose approximate shape is known. A logical operation can be defined individually for each line brush which supports very complex queries. An example of composite brushing is provided in Fig. 5. A complex combination of line brushes is used to include and remove various function graph shapes in the focus set. We have found compositions of line brushes very intuitive and effective in brushing function graphs.

A *rectangular brush* selects all curves which pass through the rectangle. The timebox widget [28] is an analogy to rectangular brushing. We have enhanced the original idea by allowing the user to optionally limit the brushing to function graphs that enter and leave the rectangular brush at given edges. Probably the most useful ones are those where the function graph is required to enter and leave at the bottom or on the top edge of the rectangle. These function graphs have a local maximum or minimum inside the rectangle, which is often a criterion in time series data analysis. This is especially useful if the display of a family of function graphs is dense and areas of maxima and minima are overlapped by other function graphs. The rectangular brush can be represented as a composition of line brushes.

V. Analysis Procedures

The shapes of function graphs depend on the independent variables $\mathbf{x}$ and in practical cases the shapes usually exhibit similarities for slight variations in the variable values, albeit this correlation may be quite indirect. For example, complex physical systems can be considered "black boxes" that return output for an input parameter set, but their exact dependencies on inputs are unknown. This can also happen if a system is simulated using a computer: the boundary conditions can have so diverse effects on the results that in analysis of such systems it is more feasible to reconstruct the black box by exploration rather than by trying to deduce its internals from a simulation process. Analysis and exploration of this class of data involves several types of procedures, including discovering trends and tendencies or finding outliers in D. For certain data sets, similar analysis of I can also be of interest. However, in this section we focus on finding patterns and dependencies in the union of I and D.

A. Black Box Reconstruction

We call the process of understanding the influence of independent variables on dependent function graph variables *black box reconstruction*. To accomplish this, we usually need to have an overview of the entire data set, following the principles of Schneiderman's Visual Information Seeking Mantra: overview first, zoom and filter, then details-on-demand [36]. We are interested in how function graphs change as values of independent variables are changed. We want to fix values of some independent variables to reduce the focus area and vary other independent variables while studying the corresponding function graphs. This is an interactive and iterative data exploration process: brushes are created and moved to areas of interest. When we have built up an overview of the dependencies we want to zoom in on details in both I and D in order to discover more subtle correlations in the data. In case of function graphs it is especially important to provide context information so that changes in shape are more obvious as various ranges of values are brushed.

An example of black box reconstruction is shown in Fig. 4. We are interested in evening traffic characteristics entering the Minneapolis area freeway system. The freeway system has entrances from North, South, East and West. We first brush these entry points in the map view with a logical OR combination of four brushes labeled 1 through 4. Then low traffic volume in the evening is excluded from the focus set using a line brush with SUB operation (5). Finally, we create a sixth, larger brush in the map using the AND operation to restrict the focus set to one of the entries. By dragging this last brush to the other three entries we can quickly change the focus and compare traffic patterns of the four entry points, while still being accurate with respect to brushes 1 to 4. After

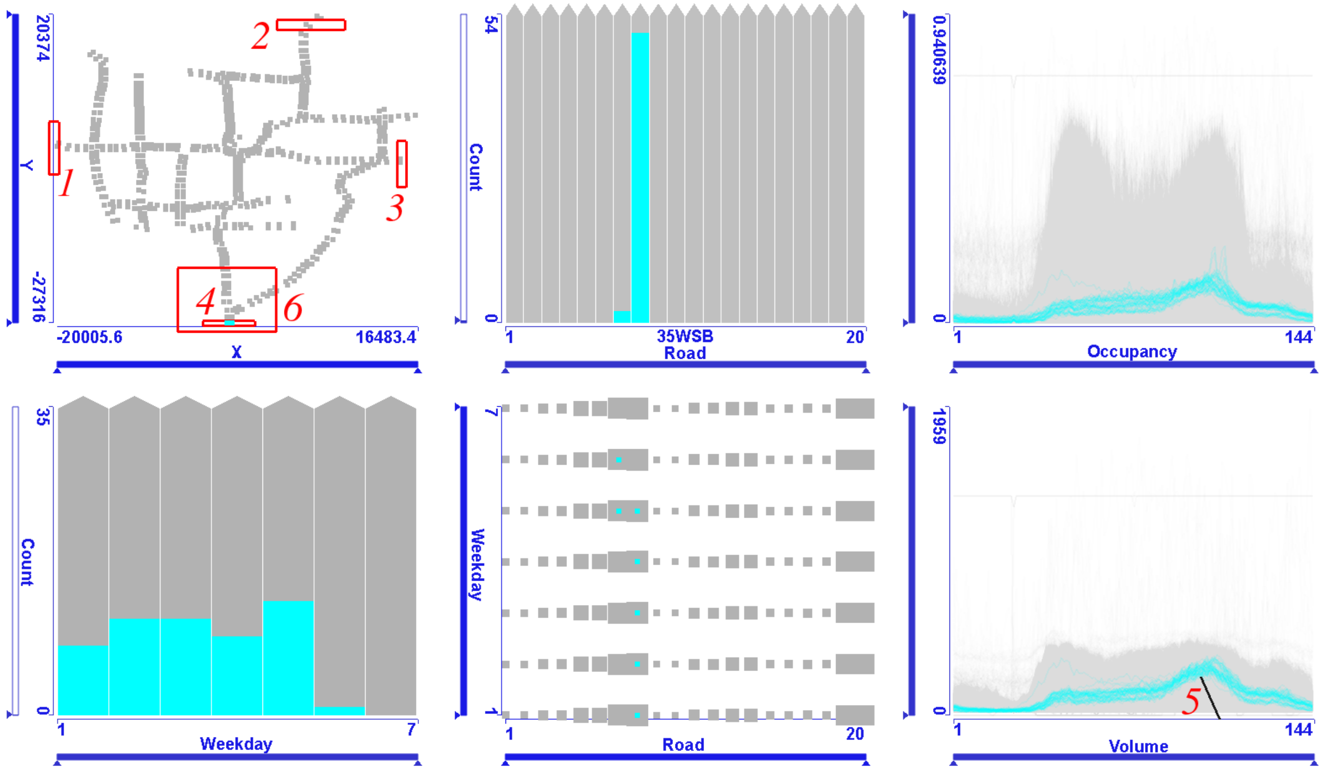

Fig. 4. Another snapshot of an iterative, interactive visual analysis of the Minneapolis traffic data. The creation order of brushes is indicated by red numbers. First, entries into the freeway system are selected using the union (OR) of four brushes (labeled 1 to 4) in the top left map. Next, low volume evening traffic is excluded from the focus by subtracting graphs which intersect brush 5. Using logical AND with brush 6 in the map view we can restrict the investigation to one specific entry. By dragging this brush to other entries we can quickly change the focus to one of the four entries. Thereby a comparison of traffic patterns with respect to the four entries is possible. In this snapshot evening traffic on the south entry is shown. The highlighted points in the lower middle scatterplot and the linked histograms reveal that heavy traffic direction is south-bound from Mondays to Thursdays, but interestingly, shifts to north-bound on Fridays and Saturdays (road number and direction are displayed on mouse-over).

each interaction step all views are immediately updated in order to support iterative analysis.

B. Analysis of Families of Function Graphs

In this type of analysis we (approximately) know the desired or expected shape of function graphs and our goal is to find combinations of independent variables that produce those shapes. We also want to exclude combinations that produce undesirable or invalid function graphs and we want to find out how the deviations from the desired shape depend on the independent variables. This could be considered an inverse investigation compared the one in Section V-A. However, this type of analysis requires that the we have an idea of the operation of the black box so that we avoid erroneously identifying dependencies that are a mere coincidence.

The procedure requires focus+context views of the graphs where criteria can be defined to select graphs of specific shapes. The desired shapes of graphs can be characterized by brushing. The typical procedure is to locate invalid or undesired function graphs first, as illustrated in Fig. 2. We brush them in the function graph view and find the related values of independent variables, in this case locations of the malfunctioning sensors. We will exclude these items from further analysis.

The desired properties of a function graph can be defined by line brushes, as shown in Fig. 3. Here we look for locations of high volume morning traffic on a given road on weekdays. This can be accomplished by selecting the road and weekdays in the scatterplot (brush 1) and then removing low traffic volume function graphs using SUB operation (brush 2). The locations are highlighted in the linked map.

C. Multidimensional Relations

Another interesting aspect of the analysis is the correlation between various families of function graphs. We want to investigate features of one family of function graphs depending on the properties of a set of function graphs in another family, for example, relationships between traffic volume and occupancy. This analysis within multidimensional time series data requires that families of function graphs are displayed simultaneously and the user can interactively brush specific groups of function graphs in one family and study the corresponding ones in the other families. We may also want to narrow the search by specifying filters on the function graph's independent variables $\mathbf{x}$. Furthermore, we want to be able to define criteria for various families of function graphs.

As an illustrative example let us consider the following query on the traffic data: we look for areas where traffic is strong, but still moving both in the morning and in the after-

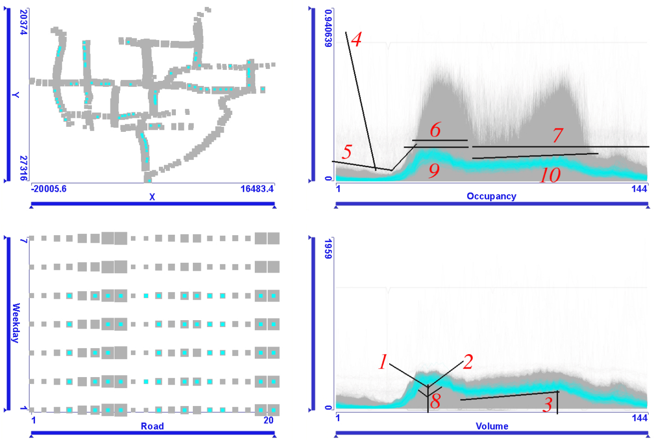

Fig. 5. We look for roads where traffic is high both in the morning and in the afternoon. First we select heavy morning traffic (1 AND 2), then remove low evening traffic volume (SUB 3). Data from malfunctioning sensors is removed (SUB 4). Next we remove high occupancy function graphs by subtracting brushes 5, 6 and 7. The desired function graph shapes are further refined by removing function graphs which intersect brushes 8, 9 and 10. The combined criteria on the two function graphs reveals the roads in question on the map. We can also see in the lower scatterplot view that interestingly, on specific roads, no points are highlighted on Wednesdays. This means traffic on those roads does not follow this pattern.

noon. Those are heavily used roads without traffic jams. If cars travel at higher speeds then many cars pass over the sensors but with relatively large gaps. This is indicated by high volume and relatively low occupancy values. In the analysis tool this is expressed as a combination of brushes in both families of function graphs. There is no direct correlation between the two function graphs. The investigation is demonstrated in Fig. 5. We brush in the volume and occupancy function graph views and study the linked map. First large morning traffic volume is brushed using two line brushes (1 AND 2). Then we remove function graphs with low traffic volume in the evening (SUB 3). Now we narrow down the search in the occupancy function graph view. Data from malfunctioning sensors is excluded (SUB 4). We then limit the occupancy by removing function graphs which intersect brushes 5, 6 and 7. Now we have a view of the areas where traffic is strong but moving in the morning and in the evening. We can further refine the desired traffic volume shape in the morning by removing the function graphs which intersect the three line brushes (SUB 8). Finally, we limit the occupancy to even lower ranges by removing function graphs that intersect brushes 9 and 10.

D. *Hypothesis Generations via Visual Analysis*

There is a particularly strong need in engineering applications to perform automatic optimization of designs using several simulation iterations with suitably varied boundary conditions. The automatic optimization process must have an approximate model of the simulation so to know how boundary conditions should be adjusted in search for an optimum. Visual analysis can be used to create hypotheses and rules that the automatic optimization can use in its simplified model and also to find out if the optimization misses some families of function graphs while searching for an optimum.

Gaining insight into the current design and setting up hypotheses about its operation via visual analysis has a very important additional advantage over pure numeric optimization. When designing a new component, engineers almost never start from scratch, but the new design evolves from an old one. Because of this iterative nature of design in engineering, the insight gained from analysis of previous designs can be useful in improving future ones. It also implies that simulation models of new designs are not radically different from that of the old ones and their results are comparable to some extent. By analyzing the relationships between the two, tendencies can be found and extrapolated to improve future designs.

VI. Engineering Application: Fuel Injection System Simulation

The science of visual analytics is very applicable in engineering applications. Simulation and measurement data sets are vast, optimization goals are often conflicting, the tendencies and dependencies in the data can be indirect and engineers need to find optimal configurations. Designers must make defensible and responsible decisions because design mistakes can have very expensive consequences if shortcomings are discovered during production. Time-to-market for new designs needs to be short, so designers must work under time pressure and communicate their findings to collaborating teams. In this section we demonstrate the applicability of our approach to the analysis of Diesel injection system simulation data.

A. *Diesel Common Rail Injection Systems*

There are many (often conflicting) goals of Diesel engine design including the need for high power and good fuel efficiency, meeting emission regulations, reducing noise levels and improving driveability (steady and reliable torque at various engine speeds). The fuel injection system is the key Diesel engine component to achieve those goals. The following properties are considered important in the fuel injection procedure:

- high injection pressure for good atomization and combustion,
- flexible timing of the injection,
- short pre-injection before the main burst to reduce combustion noise,
- accurate control of injected fuel quantity,
- ability to inject small amounts of fuel to achieve economical operation and good emission properties.

A specific type of injection systems, the common rail injection system can be controlled in a very flexible way. Injection pressure and quantity can be controlled with a high degree of flexibility, multiple fuel injections are possible within one injection cycle and the time and duration of the injections can be controlled precisely by the engine control unit based on the engine speed and load. These properties are key factors in meeting current and future very stringent emission regulations. Therefore, common rail injection systems are seen as a very popular option by many manufacturers. In our case study we use the simulation results of a conventional series common rail Diesel fuel injection system [10], [37].

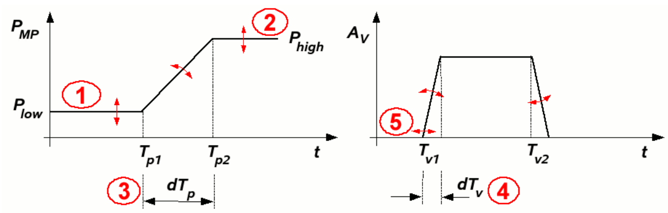

Fig. 6. Control parameters for the simulation. Left: inlet pressure characteristics are pressure levels P_{low}, P_{high} and the time interval of pressure increase dT_p. Right: injector valve opening/closing properties are the time T_{v1} when the valve starts to open and the opening/closing time interval dT_v.

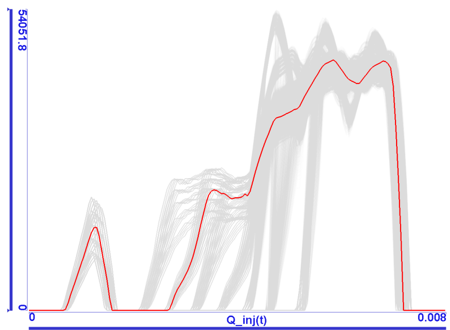

Fig. 7. A typical shape of the fuel injection rate function graph is highlighted in red. There is a short pilot injection first, followed by the main injection. The function graphs resulting from other combinations of control parameters are gray.

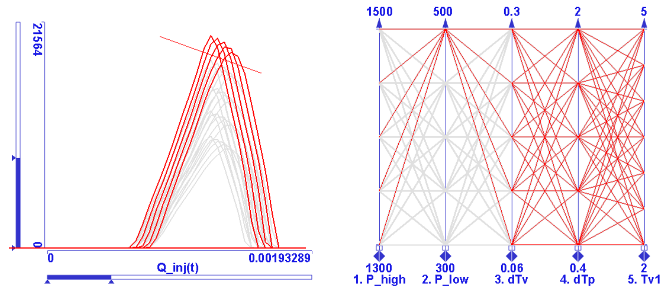

Fig. 8. Pilot injections with high amount of fuel are brushed with a line brush. As seen in the parallel coordinates view of the independent variables, these all correspond to high P_{low} values.

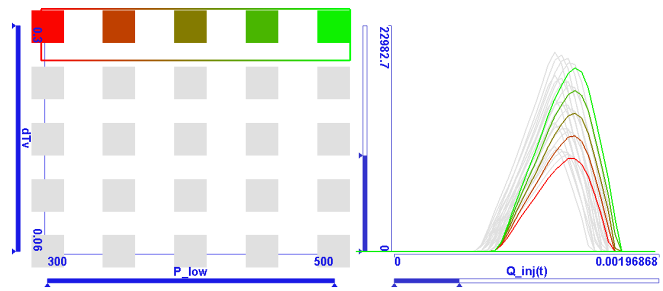

Fig. 9. High dT_v and all P_{low} simulation parameters are brushed in the scatterplot diagram using a gradient brush. The correlation of P_{low} and the amount of pilot injection is revealed by the color gradient.

B. Fuel Injection Simulation

The fuel injection simulation data is from AVL-List GmbH. The simulation is based on the theory of 1D fluid dynamics and 2D vibrations of multi-body systems. In 1D fluid dynamics pressure is uniform on pipe slices perpendicular to the axis. The simulation was run for a number of *cases*. Each case is represented by its own set of simulation control parameter values. The software can automatically loop the parameters over a specified range and run a simulation variant for each resulting case. The engineers study the resulting output data and attempt to find ideal simulation input parameters for various engine operating situations. This data set follows the model introduced in Section III: I consists of the simulation control parameters and D consists of the simulation output. In the following, we provide a detailed description of the dependent and independent variables in this data set.

1) Simulation Control Parameters: The injection shape depends mainly on three factors: the nozzle geometry, injection pressure and timings for valve opening and closing procedures. The influence of control parameters related to nozzle geometry has already been investigated in our previous work [10]. Once a (nearly) optimal nozzle geometry is found and it goes into production it cannot be changed very often because that would be too expensive. The focus of fuel injection optimization afterwards is usually on varying the remaining two factors.

Therefore, the independent variables in our current investigation are related to injection pressure and injector valve timings only. The injection pressure is controlled by the injection pressure modulation device which is positioned between the rail and injector. In our investigations this device is not modeled in detail, but we take the modulated pressure as input. The characteristics of the pressure on the injector's inlet are described by three independent variables (Fig. 6). The injector valve actuator that controls the injection timing is described by its opening/closing times and velocities. Although this is a simplified model, it allows the simulation of various types of valve actuators including the popular solenoid type or the more recent piezoelectric ones. Consequently, we have I of five independent variables. In parenthesis we indicate the number of variations for each independent variable.

1) P_{low}: low pressure on the injector inlet (5),
2) P_{high}: high pressure on the injector inlet (5),
3) dT_p: time interval of modulated pressure increase on the injector's inlet (5),
4) dT_v: time interval of the injector valve opening and closing (5),
5) T_{v1}: injector valve opening time (7).

The total number of variations of the independent variables is $5^4 \times 7$, which means 4375 different sets of simulation boundary conditions.

2) Simulation Output: For each combination of the independent variables the simulator computes three sets of time-dependent results: $Q_{inj}(t)$: injection rate, $P_{inj}(t)$: injection pressure and $A_n(t)$: needle lift. In other words, there are three families of function graphs in this data set. Furthermore, the following regular dependent variables are computed: Q_p: amount of fuel injected during pilot injection, Q_m: amount of fuel injected during main injection, Q_{vo}: amount of fuel flowing back to the fuel tank, V_{open}: needle opening velocity, V_{close}: needle closing velocity, L_p: spray penetration depth, P_{ia}: average injection power.

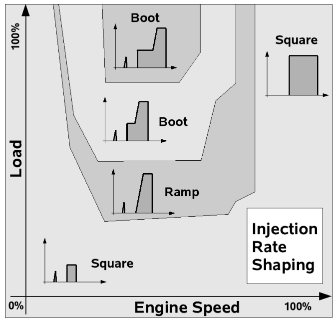

Fig. 10. Ideal shape of the main injection for various engine operating points defined by engine speed and load. The shape can be classified into three types: square (injection rate steeply increases to a maximum level), ramp (the slope is more gentle) and boot (following a nearly horizontal segment injection rate rapidly increases to maximum level at the moment of ignition). This classification is somewhat arbitrary, since the shape changes from square to boot in a continuous manner as the control parameters vary.

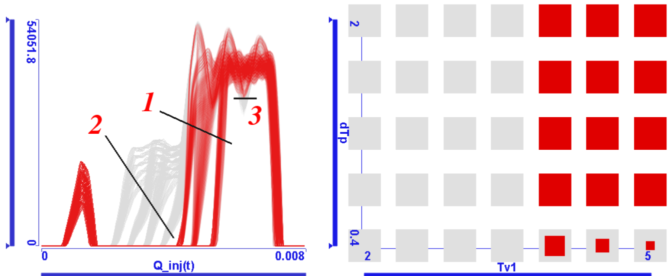

Fig. 11. Ranges of control parameters that produce square shaped injections. First the user attempted to select square shaped graphs with brush 1. This brush selects several not square shaped graphs, too. These are removed by subtracting graphs that intersect brush 2. Brush 3 removes graphs that drop under a certain threshold in the main injection part. This property is a result of vibrations in the fuel line, which are to be avoided.

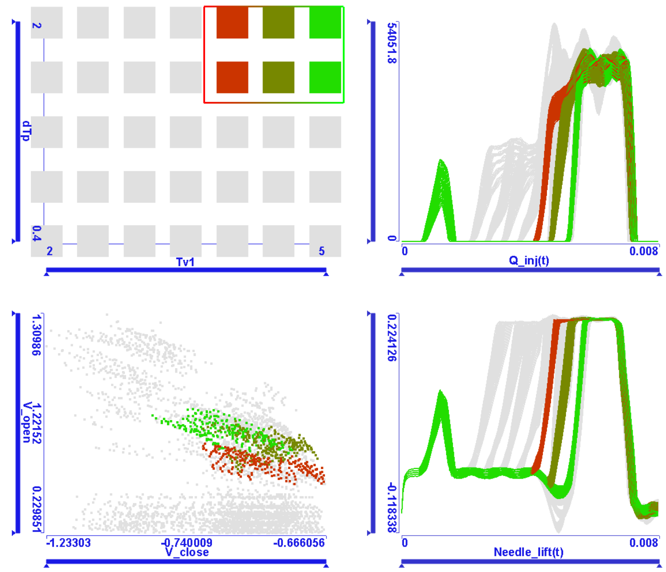

Fig. 12. Required needle movement characteristics for square shaped main injections. Top left: control parameters that produce square shaped injections are brushed. Top right: red to green gradient shows that earlier valve opening times cause deviation from the ideal square shape. Bottom left: needle opening and closing velocities must be fairly high for this shape. Bottom right: the shape of the needle lift function graph is closely correlated to that of the injection rate function graph.

C. Analysis of Fuel Injection Simulation Data

Arbitrary shaped injection rate function graphs cannot be produced in simulation because of the physical requirements of the combustion in the engine. The engine will not run properly if the shape of injection rate function graph does not follow the ones in Fig. 7. There are usually one (as in Fig. 7) or two small peaks called *pilot injection* in the first quarter of the injection procedure in order to reduce combustion noise and NO_X emission in combination with the main injection. The goal is to find combinations of simulation parameters that control the volume of the pilot injection and produce the desired shape of the main injection.

1) Analysis of Pilot Injection: We investigate how the amount of fuel and the timing of the pilot injection depend on the control parameters. We zoom to the pilot injection part of the $Q_{inj}(t)$ function graph and select function graphs with high peaks using a line brush (Fig. 8). The corresponding items are highlighted in the parallel coordinate view of the input parameters We can see that strong pilot injections are linked with high P_{low} values. We suspect that there is a direct correlation between P_{low} and the amount of fuel injected during pilot injection. To support this hypothesis we brush all P_{low} and high dT_v values with a gradient brush in a scatterplot view (Fig. 9). The color gradient from red to green establishes visual links between the brushed items in the scatterplot view of the injection control parameters and the graph view of the injection rate.

Next, we try to find the parameters that determine the timing of the pilot injection. We brush the peaks of the function graph with a line brush and examine the parallel coordinate view of the control parameters. We conclude that time of pilot injection's peak depends on dT_v. This hypothesis can be counter-checked in an interactive way. A brush is panned over the scatterplot diagram of P_{low} and dT_v and the highlighted injection rate function graphs are studied. We find that large dT_v values cause the pilot injection to start later and also to have slightly lower volume. The fully covered axes of the three other control parameters in the parallel coordinates suggest that the pilot injection's shape does not depend on them.

2) Analysis of Main Injection: The optimal shape of main injection is different for each particular engine operating point (Fig. 10). The engine control unit (ECU) measures engine speed and load to determine the current operating point. For each operating point the ECU contains a lookup table of injection control parameters used to control the injection system. The goal is to find suitable sets of control parameters for characteristic points in the diagram and understand how various properties of the injection rate function graph can be controlled. In the following we investigate how suitable control parameters can be found for specific main injection shapes. For each case we also demonstrate some additional dependencies and tendencies in the data set.

a) Square: Square main injection shape is desirable when load is very low or when the engine speed and load are both high. We used a combination of three line brushes to select square shaped injection rate function graphs (Fig. 11).

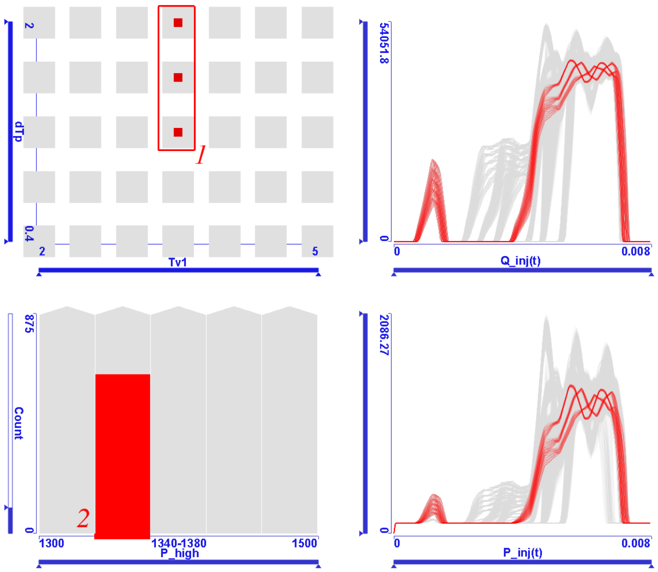

Fig. 13. Top left: brush 1 selects control parameters for ramp shaped main injections. Bottom left: brush 2 is probed in the histogram of the high pressure on injector inlet using the AND operation. Top right: injection rate function graphs of the brushed items. Bottom right: injection pressure function graphs of the brushed items. By dragging brush 2 and studying the linked function graph view we observe that injection rate function graphs have similar shapes but different maxima.

The aim of brush 3 is to exclude undesired shock wave reflections.

Using steps similar to the ones used when investigating the pilot injection we discover that T_{v1} is high for the brushed function graphs. That means the injector valve opens late when the pressure on its inlet is already very high. This leads to a sudden increase of injection rate, creating square shaped injection rate function graphs. As brush 3 is created we observe in the linked scatterplot diagram that most of the items with low dT_p (time interval of modulated pressure increase) are removed from the focus. This means dT_p must not be very low in order to avoid shock wave reflections.

We also study the desired needle opening and closing velocities and the correlations between the injection rate and the needle lift function graphs for this case. In order to do so, high T_{v1} and dT_p are brushed in the scatterplot diagram in Fig. 12. The highlighted points in the V_{close}/V_{open} scatterplot diagram show that fairly fast needle opening and closing is required for square shaped injections. The needle lift function graph (bottom right) is also linked and the color gradient of the brush shows a strong correlation between the needle lift and the injection rate graphs.

b) Ramp: Ramp-shaped main injection is desirable when the engine speed and load are in a mid-range. In the previous case we have found a correlation between T_{v1} and the shape of the injection rate function graph. We also know that the time interval of the modulated pressure increase on injector's inlet should be fairly high to avoid reflections. Based on this we start the investigation by brushing cases when the injector valve starts opening a little later and we exclude low dT_p ranges (brush 1 in Fig. 13).

The histogram of P_{high} is also brushed (brush 2) and the intersection of the two brushes is studied in the injection rate and injection pressure function graphs. We observe that the corresponding function graphs of injection rate and pressure are similar in shape but differ in their maxima as P_{high} is varied (Fig. 13).

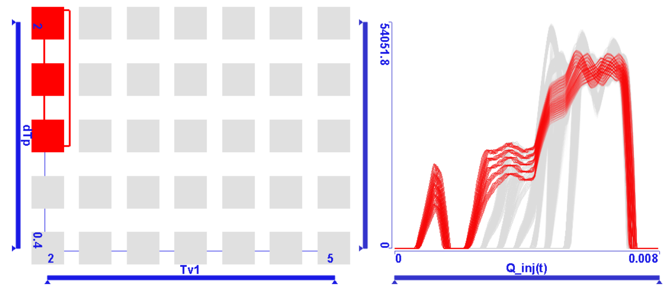

Fig. 14. If the injector valve is opened very early than the injection rate quickly increases to the "boot" level. It reaches its maximum when the mixture is ignited in the combustion chamber.

c) Boot: Boot-shaped main injection is desirable for engine operating points of mid-range engine speeds and high load. From our previous experience we assume that the injector valve has to be opened very early to achieve this shape. This assumption is verified in Fig. 14 by brushing the corresponding region in the scatterplot diagram of T_{v1} and dT_p.

Now we investigate how desired amounts of fuel in the main injection and various injection penetration levels are achieved. The scatterplot diagram of these dependent variables is brushed and the brushed items are observed in the linked views in Fig. 15. We observe that the brushed injection rate function graphs are all boot shaped. In the parallel coordinates view it is obvious that boot shaped injection does not require fast injection rate increase, but fast needle closing and injection rate decrease are still preferred. We also discover that for deep fuel spray penetration and high injection powers (brushed in green in Fig. 15) fast needle closing velocities are required. The injected fuel mass (Q_m) and the amount of fuel returned to the fuel tank (Q_{vo}) are both fairly high. This matches our expectations, since we see in the parallel coordinate view that fuel pressure P_{high} was also quite high in these cases.

3) Insight Gained from Analysis: In this example we have gained valuable insight into the fuel injection simulation data set and thereby into the fuel injection process.

We found that the amount of injected fuel in the main injection stage can be controlled by adjusting P_{high}. The amount of pilot injection is controlled mostly by P_{low}, but dT_v also has some influence on it. We observed that choosing dT_p and T_{v1} is the key to achieving the desired injection shapes for various engine operating conditions. When pressure increases too fast on the injector's inlet then the resulting wave can be reflected into the fuel line which impairs our control over the injection's shape. By studying the needle lift function graph and the related V_{open} and V_{close} simulation outputs we can define the desired needle characteristics for specific injection shapes and see how tightly the injection rate and the needle lift function graphs are correlated.

Additional images and supporting videos are available at `http://www.vrvis.at/scivis/graphs-analysis/` .

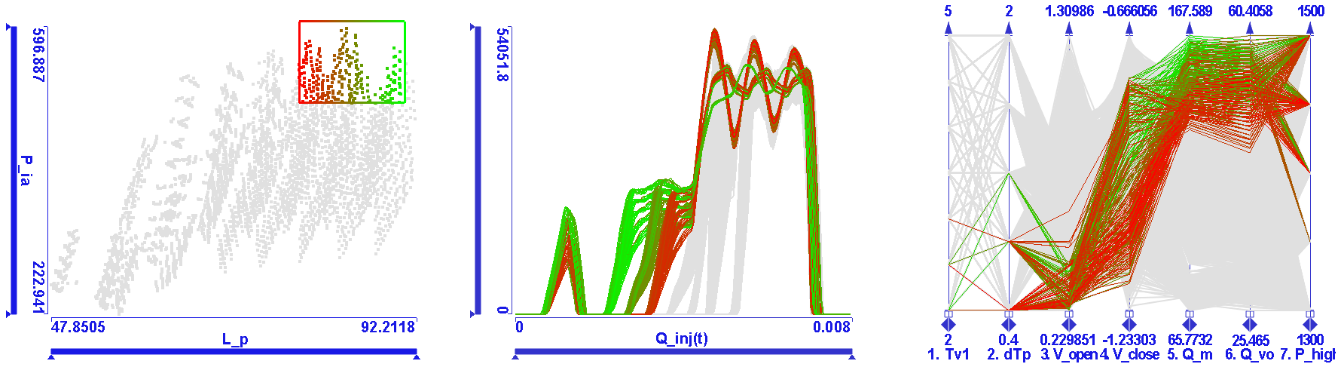

Fig. 15. We investigate the conditions when fuel is injected deep in the combustion chamber and with high power. The corresponding items are brushed in the scatterplot diagram. The linked injection rate function graphs show that this requires boot shaped main injections. The desired needle opening and closing velocities are highlighted in the parallel coordinate view.

VII. Conclusion

The analysis of relationships between families of function graphs is a common task in many application domains. A novel combination of established visualization techniques, linked views and advanced brushing features represents a valuable tool for interactive visual exploration and analysis of data sets that include families of function graphs. Independent and dependent variables in the data set are treated the same, providing improved support for iterative exploration and analysis of the entire data space. Multiple, linked views enable simultaneous viewing of independent and dependent parameters with immediate feedback.

Brushing proved to be especially effective, since it allows the interactive exploration of relations between independent and dependent variables. The color gradient improves the visual connection of the brushed items to the linked focus+context visualizations. The composite brushing with AND, OR, and SUB operations supports the iterative refinement of information drill-down and the detection or extraction of patterns from the application domain. The line brush technique proves to be especially useful in selecting function graphs. It is intuitive, easy to use and very effective. Fig. 5 shows how a composition of nearly a dozen line brushes is used to identify a pattern in a family of traffic volume function graphs.

The process of the composite brush construction captures the essence of visual analytics procedures: it is interactive and iterative. The initial brush provides the initial data selection in one view. That selection is immediately displayed in the linked views where it can be analyzed from different perspectives to formulate a hypothesis. That hypothesis is then tested using new brushes. During this iterative procedure new, possibly unexpected patterns can be found. Fig. 2 shows a discovery of a pattern in D (constant high occupancy) that indicates a pattern in I (malfunctioning sensors). Such discoveries are more difficult or even impossible without interactive visual analysis.

Future work will proceed in three directions. First, we will expand the data model to include input time series and time-dependent input parameters as well as first and second order derivatives of times series. We will explore what impact this has on the required analysis procedures and try to find tools to support the new tasks. Finally, we will explore the use of large-scale displays and usability issues related to manageability and arrangement of large number of views.

Acknowledgments

The authors would like to thank the reviewers for their valuable comments and suggestions. The fuel injection simulation data is courtesy of AVL-List GmbH (`http://www.avl.com/`). Parts of this work was done through applied and basic research at the VRVis Research Center funded by an Austrian governmental research program called K plus (`http://www.kplus.at/`).

References

[1] J. J. Thomas and K. A. Cook, Eds., *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Center, 2005.

[2] H. Doleisch, M. Gasser, and H. Hauser, "Interactive feature specification for focus+context visualization of complex simulation data," in *Proc. of the Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization*, G.-P. Bonneau, S. Hahmann, and C. D. Hansen, Eds., 2003.

[3] J. J. Thomas and K. A. Cook, "A visual analytics agenda," *IEEE Comput. Graph. Appl.*, vol. 26, no. 1, pp. 10–13, 2006.

[4] J. G. Trafton, S. S. Kirschenbaum, T. L. Tsu, R. T. Miyamoto, J. A. Ballas, and P. D. Raymond, "Turning pictures into numbers: Extracting and generating information from complex visualizations," *Int. J. Hum.-Comput. Stud.*, vol. 53, no. 5, pp. 827–850, 2000.

[5] P. Saraiya, C. North, and K. Duca, "An evaluation of microarray visualization tools for biological insight," in *INFOVIS '04: Proc. of the IEEE Symp. on Information Visualization (INFOVIS'04)*, 2004, pp. 1–8.

[6] V. González and A. Kobsa, "Benefits of information visualization systems for administrative data analysts," in *Proceedings of Information Visualization 2003*. IEEE Computer Society, 2003, pp. 331–336.

[7] 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*. New York, NY, USA: ACM Press, 1994, pp. 313–321.

[8] R. S. Laramee, C. Garth, H. Doleisch, J. Schneider, H. Hauser, and H. Hagen, "Visual Analysis and Exploration of Fluid Flow in a Cooling Jacket," in *Proc. of the IEEE Visualization 2005*, 2005, pp. 623–630.

[9] Z. Konyha, K. Matković, and H. Hauser, "Interactive 3D Visualization Of Rigid Body Systems," in *Proceedings IEEE Visualization 2003*. IEEE Computer Society, 2003, pp. 539–546.

[10] K. Matković, M. Jelović, J. Jurić, Z. Konyha, and D. Gracanin, "Interactive Visual Analysis and Exploration of Injection Systems Simulations," in *Proc. of the IEEE Visualization 2005*, 2005, pp. 391–398.

[11] P. C. Wong and R. D. Bergeron, "30 years of multidimensional multivariate visualization," in *Scientific Visualization, Overviews, Methodologies, and Techniques*, 1997, pp. 3–33.

[12] D. A. Keim, “Designing pixel-oriented visualization techniques: Theory and applications,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 6, no. 1, pp. 59–78, 2000.

[13] E. R. Tufte, *The Visual Display Of Quantitative Information*. Graphics Press, 1983.

[14] W. S. Cleveland, *The Elements of Graphing Data*. Belmont, CA, USA: Wadsworth Publ. Co., 1985.

[15] A. Inselberg and B. Dimsdale, “Parallel coordinates: a tool for visualizing multi-dimensional geometry,” in *Proc. of the 1st conf. on Visualization ’90*, 1990, pp. 361–378.

[16] H. Theisel, “Higher order parallel coordinates,” in *Proc. of the 5th Fall Workshop on Vision, Modeling and Visualization*, 2000, pp. 415–420.

[17] Y.-H. Fua, M. O. Ward, and E. A. Rundensteiner, “Hierarchical parallel coordinates for exploration of large datasets,” in *VIS ’99: Proceedings of the conference on Visualization ’99*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1999, pp. 43–50.

[18] E. Kandogan, “Visualizing multi-dimensional clusters, trends, and outliers using star coordinates,” in *Proc. of the 7th ACM SIGKDD int. conf. on Knowledge discovery and data mining*, 2001, pp. 107–116.

[19] P. Hoffman, G. Grinstein, and D. Pinkney, “Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations,” in *NPIVM ’99: Proc. of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*, 1999, pp. 9–16.

[20] L. Tweedie, B. Spence, D. Williams, and R. Bhogal, “The Attribute Explorer,” in *CHI ’94: Conference companion on Human factors in computing systems*, 1994, pp. 435–436.

[21] R. Spence and L. Tweedie, “The attribute explorer: information synthesis via exploration.” *Interacting with Computers*, vol. 11, no. 2, pp. 137–146, 1998.

[22] L. Tweedie, R. Spence, H. Dawkes, and H. Su, “Externalising abstract mathematical models,” in *CHI ’96: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM Press, 1996, pp. 406–412.

[23] D. L. Gresh, B. E. Rogowitz, R. L. Winslow, D. F. Scollan, and C. K. Yung, “WEAVE: A system for visually linking 3-D and statistical visualizations applied to cardiac simulation and measurement data,” in *Proc. of the IEEE Visualization*, 2000.

[24] H. Doleisch, M. Mayer, M. Gasser, P. Priesching, and H. Hauser, “Interactive feature specification for simulation data on time-varying grids,” in *Proc. of the Conference Simulation and Visualization 2005 (SimVis 2005)*, 2005, pp. 291–304.

[25] H. Piringer, R. Kosara, and H. Hauser, “Interactive focus+context visualization with linked 2D/3D scatterplots,” in *CMV ’04: Proceedings of the Second International Conference on Coordinated & Multiple Views in Exploratory Visualization (CMV’04)*, 2004, pp. 49–60.

[26] T. Schafhitzel, D. Weiskopf, and T. Ertl, “Interactive exploration of unsteady 3D flow with linked 2D/3D texture advection,” in *CMV ’05: Proc. of the Coordinated and Multiple Views in Exploratory Visualization (CMV’05)*, 2005, pp. 96–105.

[27] K. Matković, J. Jurić, Z. Konyha, J. Krasser, and H. Hauser, “Interactive visual analysis of multi-parameter families of function graphs,” in *CMV ’05: Proc. of the Coordinated and Multiple Views in Exploratory Visualization (CMV’05)*, 2005, pp. 54–62.

[28] H. Hochheiser and B. Shneiderman, “Dynamic query tools for time series data sets: timebox widgets for interactive exploration,” *Information Visualization*, vol. 3, no. 1, pp. 1–18, 2004.

[29] C. North and B. Shneiderman, “Snap-together visualization: a user interface for coordinating visualizations via relational schemata,” in *Proc. of the working conf. on Adv. vis. interfaces*, 2000, pp. 128–135.

[30] C. Weaver, “Building highly-coordinated visualizations in improvise,” in *Proc. of the IEEE Symp. on Information Visualization (INFOVIS’04)*, 2004, pp. 159–166.

[31] ——, “Visualizing coordination in situ,” in *Proc. of the IEEE Symp. on Information Visualization (INFOVIS’05)*, 2005, pp. 165–172.

[32] E. Thomsen, *OLAP Solutions: Building Multidimensional Information Systems*. New York: John Wiley & Sons, Inc., 1997.

[33] “Twin cities traffic data archive.” [Online]. Available: http://www.d.umn.edu/~tkwon/TMCdata/TMCarchive.html

[34] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., 1999.

[35] H. Doleisch, M. Gasser, and H. Hauser, “Interactive feature specification for focus+context visualization of complex simulation data,” in *Proc. of the 5th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2003)*, 2003, pp. 239–248.

[36] B. Shneiderman, “The eyes have it: A task by data type taxonomy for information visualizations,” in *Proc. of the 1996 IEEE Symp. on Visual Languages*, 1996, p. 336.

[37] W. Boehner and K. Hummel, “Common Rail Injection System for Commercial Diesel Vehicles,” *SAE Transactions, SAE 970345*, 1997.