Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization (2003)
G.-P. Bonneau, S. Hahmann, C. D. Hansen (Editors)

Interactive Feature Specification for Focus+Context Visualization of Complex Simulation Data

Helmut Doleisch, Martin Gasser, Helwig Hauser

VRVis Research Center, Vienna, Austria
http://www.VRVis.at/vis/
Doleisch@VRVis.at, Martin.Gasser@VRVis.at, Hauser@VRVis.at

Abstract

Visualization of high-dimensional, large data sets, resulting from computational simulation, is one of the most challenging fields in scientific viualization. When visualization aims at supporting the analysis of such data sets, feature-based approches are very useful to reduce the amount of data which is shown at each instance of time and guide the user to the most interesting areas of the data. When using feature-based visualization, one of the most difficult questions is how to extract or specify the features. This is mostly done (semi-)automatic up to now. Especially when interactive analysis of the data is the main goal of the visualization, tools supporting interactive specification of features are needed.

In this paper we present a framework for flexible and interactive specification of high-dimensional and/or complex features in simulation data. The framework makes use of multiple, linked views from information as well as scientific visualization and is based on a simple and compact feature definition language (FDL). It allows the definition of one or several features, which can be complex and/or hierarchically described by brushing multiple dimensions (using non-binary and composite brushes). The result of the specification is linked to all views, thereby a focus+context style of visualization in 3D is realized. To demonstrate the usage of the specification, as well as the linked tools, applications from flow simulation in the automotive industry are presented.

1. Introduction

Visualizing high-dimensional data resulting from computational simulation is a demanding procedure, posing several complex problems which include, for example very large size of data sets and increased dimensionality of the results. In this paper, we present a formal framework that supports interactive and flexible analysis of complex data using a descriptive and intuitive language for defining features and multiple linked views with information visualization (InfoViz) and scientific visualization (SciViz). In the following, we shortly discuss a few key aspects, which are important for the new approach presented in this paper.

Feature-based visualization – visualization which focuses on essential parts of the data instead of showing all the data in the same detail at the same time, is called feature-based visualization. This kind of visualization gains increasing importance due to bigger and bigger data sets which result from computational simulation, so that not all of the data can be shown simultaneously. For feature-based visualization, proper feature extraction methods are essential.

Up to now, feature extraction mostly is done (semi-)automatically[14] with little interactive user intervention, often as a preprocessing step to the visualization. But for interactive analysis, in many cases, the question of what actually is (or is not) considered to be a feature refers back to the user: depending on what parts of the data the user (at an instance of time) is most interested in, features are specified accordingly. Therefore, flexible feature extraction requires efficient means of user interaction to actually specify the features.

Separating focus and context in InfoViz – when dealing with large and high-dimensional data sets in InfoViz, simultaneous display of all the data items usually is impossible. Therefore, *focus-plus-context* (F+C) techniques are often employed to show some of the data in detail, and (at the same time) the rest of the data, at a lower resolution, as a context for orientation[3, 6]. Thereby the user's attention is di-

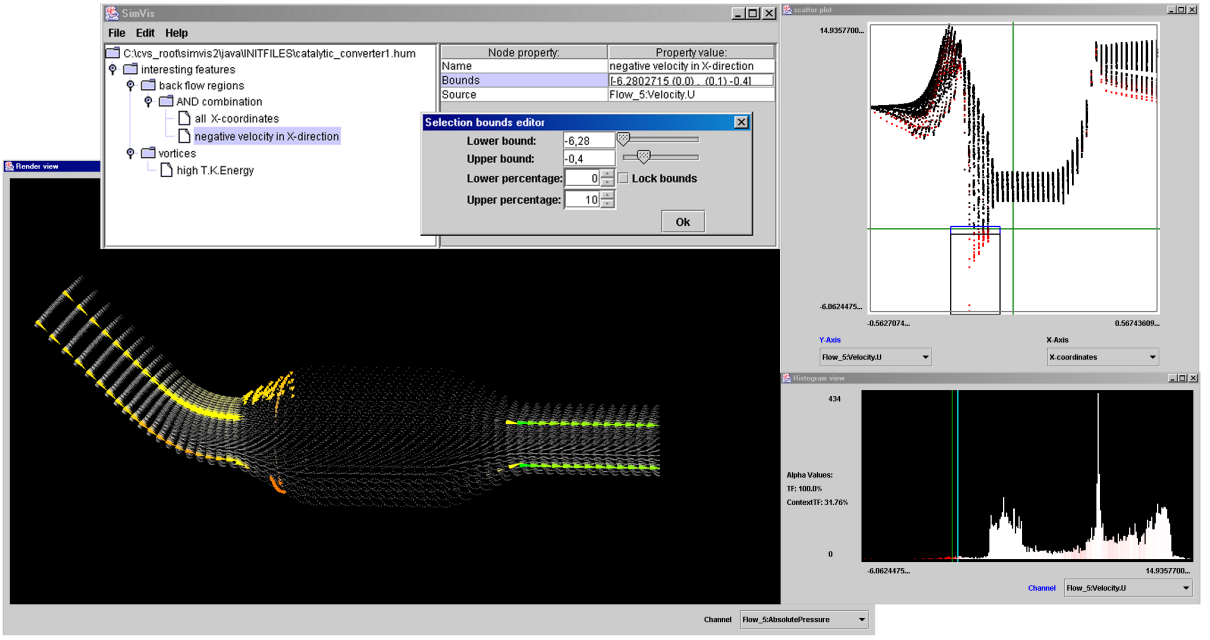

Figure 1: Flexible Feature Specification: *simulation data of a catalytic converter is shown, two features have been specified based on our feature definition language, using the different views for interaction and visualization. (see also colorplate)*

rected towards the data in focus (e.g., through visual enlargement), whereas the rest of the data is provided as context in reduced style (translucently, for example). This is especially useful when interacting with the data, or when navigating through the visualization.

To discriminate data in focus from context information, a so-called *degree of interest* (DOI) function can be used[6], assigning a 1D DOI-value out of the unit interval to each of the n-dimensional data items (1 represents "in focus", 0 is used for context information).

Defining the DOI function – in literature, implicit techniques for DOI-specification are described (e.g., focus specification through dynamic querying[15]) as well as explicit techniques, such as interactive object selection[9] or brushing[1, 17]. When brushing, the user actively marks a subset of the data set in a view as being of special interest, i.e., in focus, using a brush-like interface element.

In addition to standard brushing, several useful extensions to brushing have been proposed. Multiple brushes and composite brushes[12], and the use of non-binary DOI functions for smooth brushing[4] extend the available toolset for interactive DOI specification. Also, more complex brushes like those designed for hierarchical data[5], or such using wavelets[18] or relative information between different data channels[8] have been proposed recently.

Complex and high-dimensional feature definition – when analyzing simulation data, one very often encountered problem is the limited flexibility of current brushing and interaction techniques. Brushing is usually restricted to simple combinations of individual brushes, as well as missing support of high-dimensional brushes due to the tight coupling of GUI interactions and the representation of the brush data itself. For fast and flexible analysis of the usually large and high-dimensional simulation data, complex and also high-dimensional brushes are necessary. In this paper, we present a formal framework, that is very closely coupled to the data, allowing to define and handle such brushes interactively.

Linking multiple views – the combination of InfoViz and SciViz methods[7, 4], especially for the interactive visualization and analysis of simulation data, improves the understanding of the data in terms of their high-dimensional character as well as the data relation to the spatial layout. Linking several views[2] to interactively update all changes of the data analysis process in all views simultaneously is a crucial property for making optimal use of multiple (different) visualization views.

In previous work[4] we showed how a scatterplot (or a histogram) can be used to smoothly specify features in multi-dimensional data from simulation, and how this focus+context discrimination can be used for selective visualization in 3D. In this paper, we now present a formal framework (our feature definition language, see section 2)

for specifying features in simulation data together with advanced interaction techniques (see section 3), allowing for fast and flexible exploration and analysis of complex and high-dimensional data (application examples in section 4). Finally, a short overview about implementation details is given, as well as conclusions and some future work topics are presented.

2. Using a Feature Definition Language

When dealing with results from computational simulation, usually very large and high-dimensional data sets are investigated. Previous work already showed, that interactive specification of features with tight reference to the actual data attributes is very valuable for visualization of such data sets[7, 4]. For a fast and flexible analysis of these results, powerful and intuitive tools are needed – the here described approach provides flexibility in terms (a) of multiple options to differently view the data, and (b) a wide range of user interactions to construct and adapt feature specifications. Whereas previous work mainly focussed on viewing (a) so far, we mostly improve on interaction (b) in this paper.

To generalize the specification of features (enabling feature descriptions which are portable between data sets, for example) and to also formally represent the state of an analysis session, e.g., to allow for loading/saving of interactive visualization sessions, we present a compact language for feature specification, i.e., a *feature definition language*, here called FDL for short.

```
<FSpecData>:    Reference to data,
                <FeatureSets>          // mult. feat. sets possible
<FeatureSets>:  <FeatureSet>+          // only one active at a time
<FeatureSet>:   <Feature>+             // implicit OR of features
<Feature>:      <FeatureChar>+         // implicit AND of f.-chars.
<FeatureChar>:  <SimpleFChar>          // RxR-map (one channel->DOI)
                || <ComplexFChar>      // induces recursion
<SimpleFChar>:  Reference to channel, // direct data access
                mapping2DOI-info       // info for DOI calc.
<ComplexFChar>: AND <FeatureChar>+     // AND-comb. of subsequ.chars.
                || OR <FeatureChar>+   // OR-comb.  --"--
                || NOT <FeatureChar>   // NOT of subsequent char.
```

Figure 2: Feature definition language: sketch of its structure.

A sketch of the FDL-structure is presented in Fig. 2. Here the different key components of this language are shown, namely the feature specification itself (root), feature sets (level 1), features (level 2) and feature characteristics (level 3). In the following subsections, these four different hierarchical layers of the FDL are discussed in more detail.

2.1. Feature Specification

A description of a feature specification usually is closely coupled to a data set (the one that is to be analyzed). Alternatively, it could also be portable to similar data sets, when data semantics coincide. In the regular case, a feature specification therefore has a reference to the source data set, as well as to one or multiple feature sets (see below).

Our FDL is realized as an XML[13] language application, which makes it easy to handle and the resulting FDL-files readable. This also allows to save feature specifications as files, and load them again at any later point in time to resume an analysis session. Additionally, XML-files can be edited using a text-editor, which allows to re-adjust feature specifications also on a file level.

The explicit representation of feature specifications in the form of FDL-files makes using feature specifications on other data sets possible. Of course, care has to be taken that only data channels are referred to, which are available in all these data sets. With portable feature specifications it is possible to generate general feature definition masks, which can be applied very easily (and interactively adapted, if necessary).

2.2. Feature Sets

A feature set subsumes an arbitrary number of features which all are to be shown simultaneously (like an implicit logical OR-combination). Within each single view, always only one feature-set is used for F+C discrimination, all the other feature-sets are inactive at that time. Multiple feature sets can be used to interactively switch foci during an analysis session or to intermediately collect features in a "repository" feature set, not used at a certain point in time. Multiple views can be used for simultaneously showing different feature sets (one per view).

2.3. Features

Features are specified by one or multiple feature characteristics. The DOI function related to each feature is built up by an (implicit) AND-combination of all DOI functions of all associated feature characteristics. Multiple features are used to support named feature identification and intuitive handling of interesting parts of the data by the user. Each feature can be moved or copied from one feature set to any other.

In Fig. 1 two distinct features have been specified, one denoting areas of backflow, and another one, showing vortices. The latter one consists only of one simple feature characteristic (see below), brushing high values of turbulent kinetic energy, whereas the first feature consists of a logical combination of two separate feature characteristics.

2.4. Feature Characteristics

Feature chararateristics can be either simple or complex. Whereas simple feature characteristics store direct brushing information with respect to one data attribute (or channel) to derive a DOI function, complex feature characteristics imply a recursion.

Simple feature characteristics store a reference to the data channel which it is based on, as well as infomation about

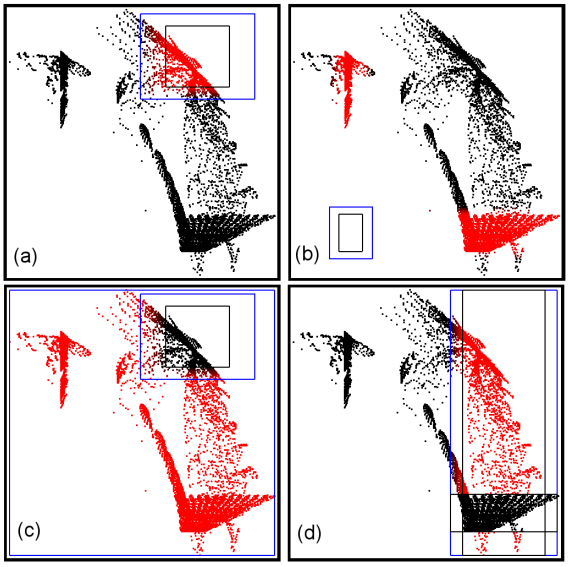

Figure 3: four examples of 2D brush types which users found useful during interactive analysis (catalytic converter example, pressure [x] vs. velocity [y]): *(a) "high velocity and high pressure" (logical AND), (b) "low velocity or low pressure" (log. OR), (c) "all but high vel. and high pressure" (NOT-AND), and (d) "high pressure but not low velocity" (SUB = AND-NOT). (see colorplate for shades of red)*

how the data of this channel is mapped to a DOI function (being the output of this characteristic). Especially the possibility for the user to directly interact with the data attributes by specifying feature charcateristics and modifying them interactively is very intuitive and straight-forward. In Fig. 1 a simple feature characteristic named "negative velocity in X-direction" is shown in the selection bounds editor. Simple feature characteristics support discrete and smooth brushing (via specifying percentages of the total brushing range, where the DOI-values decrease gradually).

Complex feature descriptions on the other hand provide logical operations (AND, OR, NOT) for the user to combine subsequent feature characteristics in an arbitrary, hierarchical layout. For combining smooth brushes, which can be interpreted as fuzzy sets, fuzzy logical combinations are used, usually implemented in form of T-norms and T-conorms[11]. We integrated several different norms for the above mentioned operations. By default, we use the minimum norm (T_M) in our implementation: this means, when doing an AND-operation of several values, the minimum value is taken, and for the OR-operation the maximum respectively.

In Fig. 3, four examples of 2D brush types, which users found useful during interactive analysis sessions, are shown. The data displayed in the scatterplot views comes from the catalytic converter application shown in Fig. 1 (which is also explained in more detail in section 4), pressure (x-axis) vs. velocity (y-axis) values are plotted. Interactive operations "NOT-AND" and "SUB" are mapped to "NOT"-"AND" and "AND"-"NOT" combinations in FDL, respectively.

3. Interaction

One main aspect of analyzing results from simulation is that investigation is often done interactively, driven by the expert working with the visualization system. Therefore, interaction is one of the key aspects that has to be considered when designing a system which should support fast and flexible usage (as described previously). Especially the task of searching for unknown, interesting features in a data set, and extracting them, implies a very flexible and intuitive interface, allowing new interaction methods. In the following subsections, we categorize the main types of interaction which our system supports. Note that these interactions are designed to meet users' most often requested requirements for such an analysis tool.

3.1. Interactive Feature Specification through Brushing

The first type of interaction that has to be considered when designing an interactive analysis tool for exploring simulation data, is *brushing*. In our system, interactive brushing of data visualization is possible in all views except for the 3D SciViz view, which is used for 3D F+C visualization of the feature specification results (see section 4). Brushing is used to define feature characteristics in the FDL interactively. As many types of applications also request non-binary brushing, we allow smooth-brushing[4] in all the interaction views. One example of using a 2D smooth brush, employing a logical AND operation of two simple feature characteristics is shown in Fig. 3 (a). Here, a region of relatively high velocity and high pressure values is brushed in a scatterplot view, defining (a part of) a feature. As can be seen from this figure, a smooth brush defines two regions. A core part of the brush is defined, where data of maximal interest is selected (mapped to DOI values of 1). It is padded by a border, where DOI values decrease gradually with increasing distance from the core part.

3.2. Interactive Feature Localization

Another very often used type of interaction is the so-called feature localization. It is usually provided in the context of simulation data, that has some spatial context. When analyzing this kind of data, the first interest is often, *where* features of specific characteristics are located in the spatial context of the data. Interactively defining and modifying features in different views, coupled with linking, the specification immediately results in a 3D rendering which provides fast localization of the features in the spatial context of the whole data set. For an example see Fig. 4 (a)-(c), where the backflow regions are interactively localized to be in the entrance of the catalytic converter chamber.

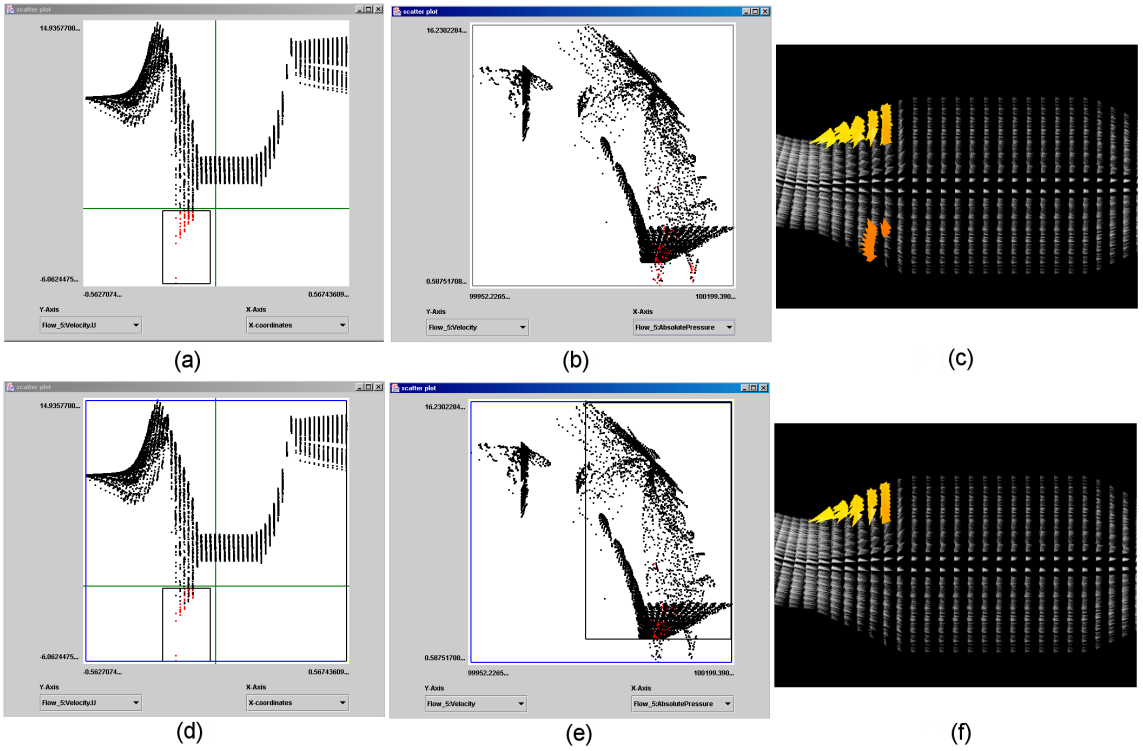

Figure 4: Interactive feature specification and refinement: *(a)-(c): first step: defining backflow region in a catalytic converter (see also Fig. 1) in a scatterplot view (a) by selecting negative x-flow values, direct linking to a second scatterplot view (b) and the 3D view (c). (d)-(f): second step: AND-refinement with a new selection in the second scatterplot view (e), back linking of the interaction via feedback visualization (color of points according to newly calculated DOI values) to the first scatterplot view (d). Now only the backflow region is selected, that exhibits general velocity above a specified threshold (f).*

3.3. Interactive FDL Refinements

After having defined multiple features via brushing and localized them, often interactive refinement of these features is the next step. Refining the feature specification can be either done by interactive data probing (see below) or by imposing further restrictions on the feature specifications, e.g., by adding additional feature characteristics to the actual state of a feature. One example of such an interactive FDL refinement is shown in Fig. 4 (d)-(f). As a first step (first row), all parts of the data, that exhibit backflow, have been selected, defining a feature that spans over two distinct regions in the spatial domain. In the refinement step (second row) a logical AND-combination of the first feature specification (a) with a new selection in a second scatterplot view of the same data (but showing two other data attributes) is performed (e). Thereby only those back-flow regions of the data are put into focus, which exhibit a general velocity above a specified threshold (f).

3.4. Interaction with Tree Viewer

Interaction with a tree viewer (see Fig. 1, left upper window) as a GUI for FDL is another very useful way to adapt or extend feature sets and features, as well as their characteristics. The tree viewer provides standard GUI elements, such as textfields for manual input of numbers or range sliders, for example. Naming of the different nodes of the FDL, as well as editing all the feature characteristics, and also the management of the tree structure (through copy, delete, or move of the different nodes and subtrees) are the most often used interaction methods in this viewer. It strongly depends on the nature of users of whether mouse-interactions or keyboard-input are preferred when specifying features. Sometimes, in the case of well-known thresholds, for example, the keyboard-input to the tree viewer is faster and more accurate then mouse-interaction to an InfoViz view.

3.5. Interactive Data Probing

Another form of interactively exploring features is using a data-probing approach. Thereby, after having specified a feature (via brushing, for example) the one or other feature characteristic can be changed interactively (e.g., by using a range slider). In all linked views (showing the same data and showing different data attributes) immediate feedback of DOI changes can give new insights into different data as-

pects. Especially for exploratively investigating value ranges and better understanding of associated patterns in the data sets, this interaction metaphor is very useful.

3.6. Interactive Management of Views

One key aspect of a system which provides multiple, different views of one data set, is the interactive management and linking of these views. Our system supports an arbitrary number of InfoViz views (currently scatterplots and histograms), as well as SciViz views. Views can be opened and closed at any point in time without distracting the feature specification. In the InfoViz views, the mapping which assigns data channels to the axes can be changed interactively. In the 3D SciViz view the mapping of a data attribute to rendering properties (color and/or opacity) via transfer functions can be interactively modified, too. Additionally, the different axes of all available views can be linked (and unlinked) interactively, allowing rapid updates in multiple views.

4. Visualization and Results from Applications

After having discussed our feature specification framework as well as the important role of interaction for analysis of simulation data, now the visualization part and typical applications are presented.

Below general aspects of visualization during analysis are presented. Then, two different application examples are described in detail. For high quality versions of the images presented here, as well as for additional examples and movies which illustrate the interactive behaviour of working sessions with our framework, please refer to *http://www.VRVis.at/vis/research/fdl-vis/* .

4.1. Visualization for Analysis

When visualization is used to support analysis of large, high-dimensional data sets, the use of multiple views, as well as of flexible views (with respect to data dimensionality) is very important. Our system supports an arbitrary number of each type of InfoViz views, as well as SciViz views. When interactively working with data, two types of views in a multiple views setup can be distinguished: *Actively linked views* are the views, which are primarily used for interaction purposes, i.e., for specifying the features, whereas *passively linked views* are primarily used for F+C visualization of the data, providing interactive updates.

3D SciViz views – The 3D SciViz views of our system are used as passively linked views for providing a F+C visualization and interactive feature localization. The F+C discrimination is mainly accomplished by using different transfer functions for focus and context parts (and interpolating inbetween, for smooth F+C discrimination). The transfer functions in use do not only specify color and opacity, but also the size of the glyphs, that are used to represent single data items (see Fig. 1, lower left window for a 3D SciViz view, showing a smooth F+C visualization).

Two main tasks of this F+C visualization can be identified. The support for feature localization and the visualization of data values through color mapping. Feature localization, as already described in section 3, plays a major role in interactive analysis based on features. By using a F+C visualization, the user attention is automatically drawn to the more prominently represented foci, i.e., the features. Value visualization is another very useful task of visualization in this view, and it is accomplished by coloring glyphs according to the associated data channel.

Of course, interactive user manipulation of rendering parameters (opacity, size of glyphs, or zoom and rotate) are necessary, very useful, and support the analysis task, too.

InfoViz views – Apart from supporting interaction, the InfoViz views (scatterplots & histograms in our system) are very valuable for visualization purposes, too. They visualize the data distribution (1D or 2D) and also give visual feedback of F+C discrimination. Points in the scatterplot views, for example, are colored according to the DOI value of the associated data item. Fully saturated red points are shown for data in focus, whereas the saturation and lightness of points decreases with decreasing DOI values, respectively (see Fig. 3 for examples).

In the InfoViz views it is especially useful that the mapped data attributes can be changed interactively. Mapping spatial axis information to one of the scatterplot axes, for example, is very intuitive in our applications (see below). Additionally, using several scatterplots, comparable to a (reduced) scatterplot matrix, often adds information about the data and internal relations of different data attributes.

4.2. Results from Air-Flow Analysis

We now want to give a step-by-step demonstration of how a typical analysis session takes place, especially to show the importance of interaction when analyzing simulation data.

(1) In a first step, a data set is loaded: in our example, results from air-flow simulation around a car (just on one central slice, from front to back of the car) are shown. To also cope with 2D-slices of 3D-data, we adapted our 3D-rendering view accordingly. It should be noted, that the general flow direction in this application is in X-direction, past the car from front to back. Before a tree viewer is opened automatically, an empty feature set is generated for preparation of an analysis session. A SciViz view is then opened interactively, to show the general spatial layout of the data (see Fig. 5 for the initial view setup). In this figure the unstructured grid of the data set is shown, overall velocity information is mapped to color (green denotes low, red relatively high velocity values).

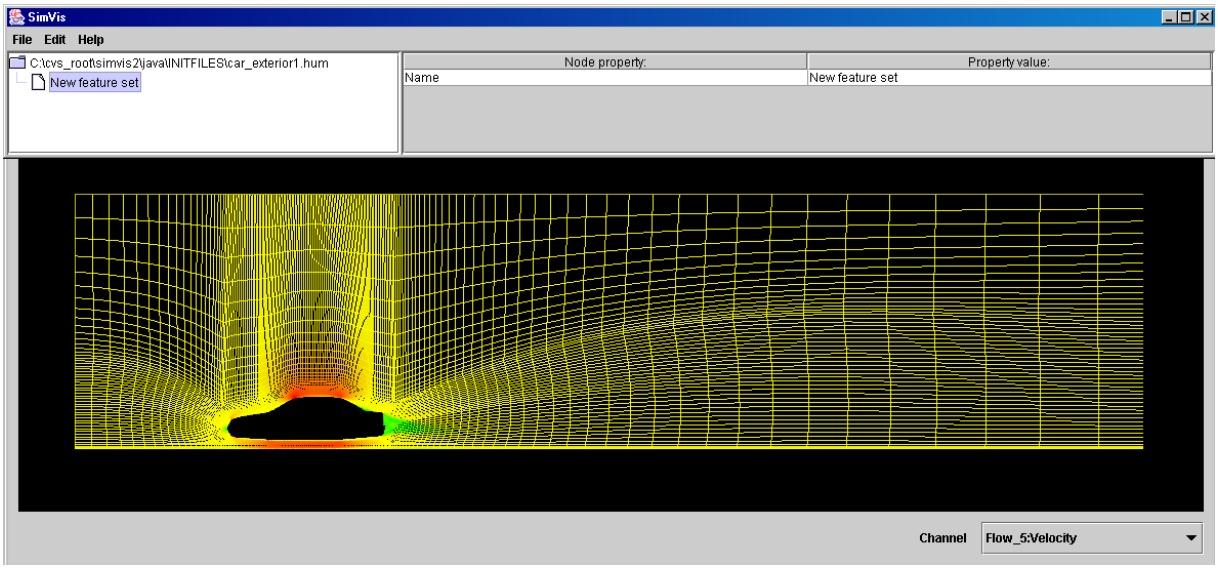

Figure 5: Air-Flow around a moving car: *After loading the data set, an empty feature set is created, and the spatial layout of the data is shown, overall velocity information is mapped to color (green denotes low, red high velocity).*

(2) As a first start into feature specification (focussing on non-horizontal, slow flow at this step of the analysis) a scatterplot view is opened, showing V-velocity (vertical component of overall velocity values), mapped to both axes. In this scatter plot an OR-brush is used to select relatively large positive V-flow, as well as relatively large negative one, too. Then the x-axis of the scatterplot view is changed to show overall velocity and an AND-refinement is done to limit the feature specification to slow flow (see Fig. 6, upper right view).

To furthermore visualize the feature specification up to this step, a second scatterplot view is opened, showing feature and context distribution with respect to the spatial X-coordinates and viscosity (mapped to y-axis of the view, see Fig. 6, lower right). In an interaction panel of the tree viewer, the restriction of V-velocity components is further adapted, to meet the user's needs (see Fig. 6 for a screen capture after this step).

(3) A further AND-refinement, restricting the feature specification to "high viscosity" values is added by using the second scatterplot view. As a result of this step, only features behind the car are part of the new focus (see Fig. 7).

(4) Yet another AND-refinement, further restricting the feature specification to high values of turbulent kinetic energy (a value also computed by the simulation), is performed in the tree viewer (see Fig. 8). This clips away parts of the previously selected features, leaving only the parts that exhibit stronger rotational behavior.

(5) To get a better idea of the vortical structures induced, interactive probing on one part of the feature specification (positive V-velocity) is performed. When limiting the focus to negative V-flow only, the downfacing parts of the upper as well as of the counterrotating, lower vortex become visible (see Fig. 9).

4.3. Results from Catalytic Converter Analysis

A second example presented here is an application, where the data comes from a simulation of a catalytic converter from automotive industry. The results of another analysis session are shown. The data is given on an unstructered grid in 3 spatial dimensions, and has 15 different data attributes for each of the approximately 12000 cells of the grid.

The data set and a corresponding feature specification is shown in several views in Fig. 1. The data set consists of basically three spatially distinct parts, the flow inlet on the left hand side, the chamber of the catalytic converter (middle), and the flow outlet on the right-hand side (see Fig. 1, left lower window for a 3D SciViz view). The other views shown in Fig. 1 include: the tree view for handling the FDL (including a pop-up window for changing the brush properties on the x-component of the velocity), a scatterplot view (right upper window) plotting x-velocity vs. x-coordinates for each data point, and a histogram, showing the distribution of x-velocity values over the data range.

Two distinct features have been specified using the InfoViz views and the FDL tree viewer. The first feature defines all backflow regions in the data set (with negative x-component of the velocity, as general flow is in x-direction). Two such regions are identified at the entrance of the chamber, a weaker one at the bottom of the catalytic converter,

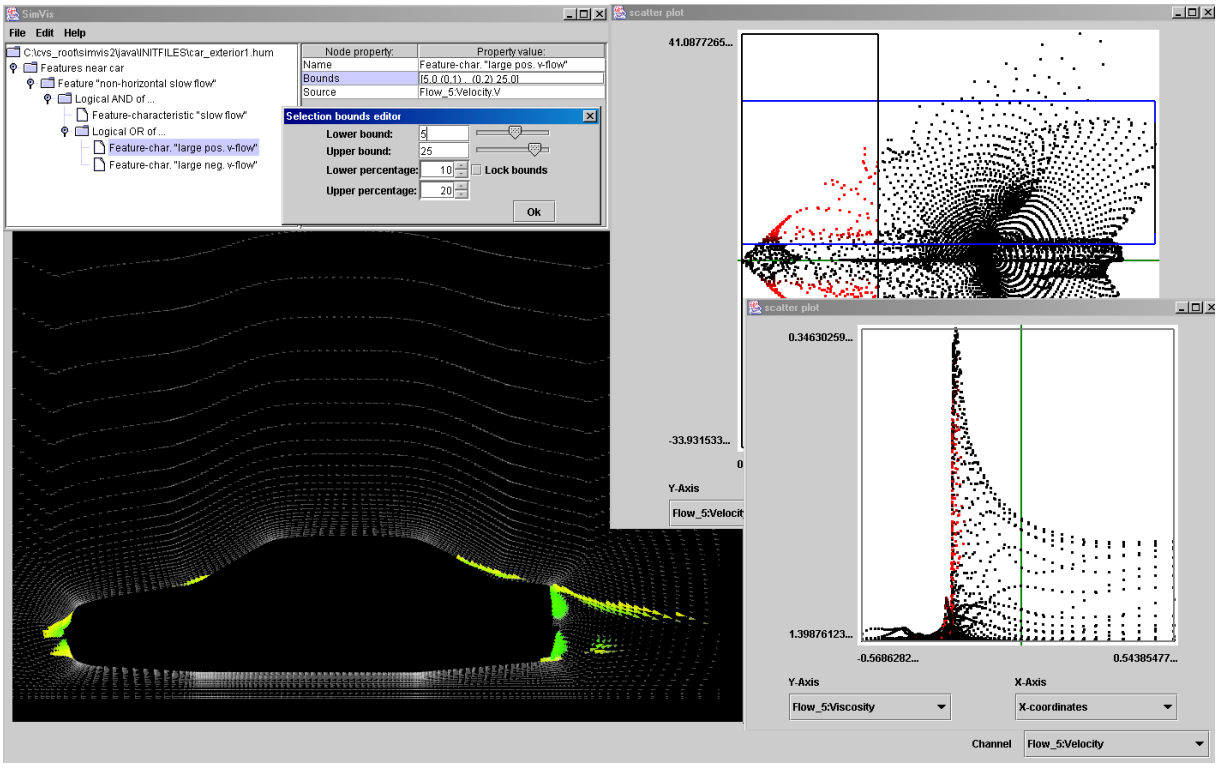

Figure 6: First step of analysis (non-horizontal slow flow): *a tree viewer showing the current feature specification in the upper left (interaction panel for adjusting a simple feature characteristic shown), a scatterplot view used for feature specification in the upper right (velocity vs. V-Velocity component), the SciViz view for f+c visualization in the lower left, a second scatter plot for visualization of f+c distribution (X-coordinates vs. viscosity).*

and a stronger one at the top. The second feature description defines all regions, with high turbulent kinetic energy, these are the regions, where vortices are appearing usually in the flow. As can be seen, two vortex cores are easily separated from the rest of the data at the inlet and outlet of the catalytic converter in this case.

Both, the vortices and the backflow regions have been brushed smoothly, to show some information about the gradient of the values in the 3D rendering view. Note, that the coloring in the 3D view is mapped from another data channel, namely data values of absolute pressure. This allows to visualize an additional data dimension for all the data, that was assigned to be in focus beforehand. In the here applied color mapping, green denotes relative low values of absolute pressure, and red corresponds to relative high values.

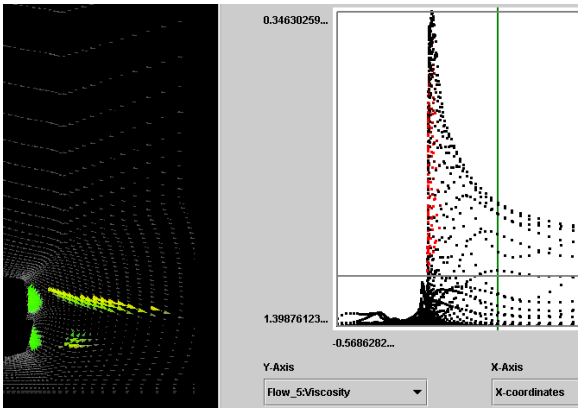

Figure 7: Step 2 of analysis: *AND-refinement, restricting feat. spec. to high viscosity values in the second scatterplot view. Only features behind the car are part of focus now.*

5. Implementation

The presented prototype system includes the described simulation data analysis tools and runs interactively on a standard PC (P3, 733MHz, 756MB of memory, GeForce2) for the data sets shown (in the range of 20.000 to 60.000 cells, 15 to 50 data attributes associated to each cell). The cells of the data are organized in unstructured grids. For the rendering of these grids a visibility algorithm was implemented,

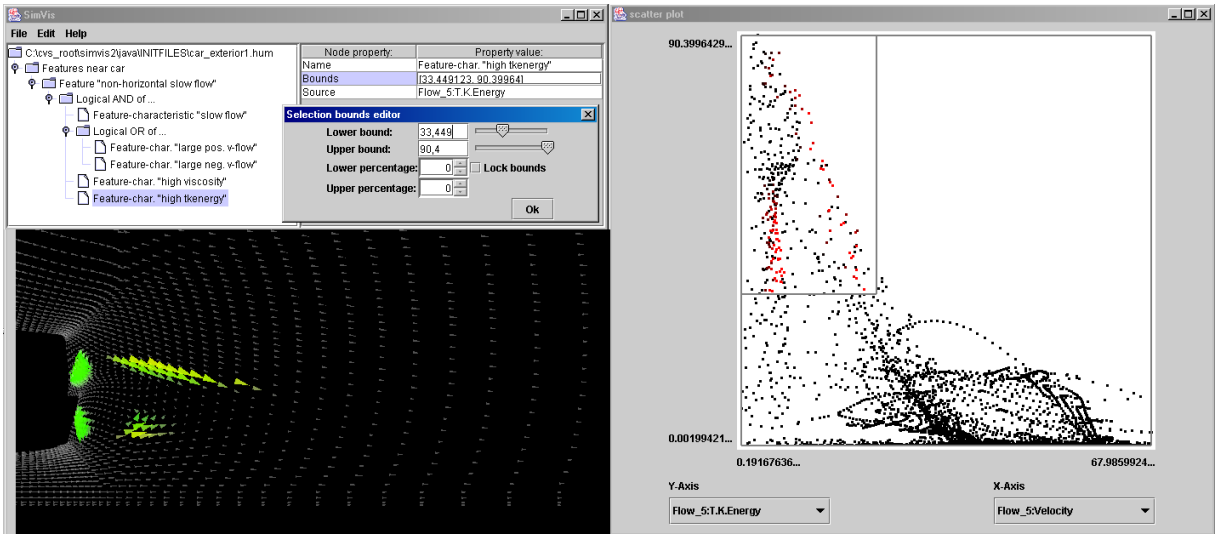

Figure 8: Step 3 of analysis: *another AND refinement, further restricting to high values of turb. kinetic energy, performed in the tree viewer. Only parts with strong rotational component are in focus. (see also colorplate)*

based on the XMPVO algorithm[16] presented by Silva et al. With newer, more powerful PC-setups we already managed to visualize data sets consisting of over a million data cells, but sorting for 3D rendering can not be performed interactively anymore.

For the implementation of our prototype, a hybrid approach was taken; UI Interaction and handling of the FDL is realized in Java, whereas mesh access and the rendering of the visualization views is implemented in native code (we used MS Visual C++). Native methods are called via the JNI API, and the gl4java package was used to make the GL rendering contexts available to the Java GUI toolkit. The mesh access has been realized by using our own data mesh format. Data coming from different data sources can be easily converted to this format via linked readers.

When designing the presented FDL, several considerations were taken, including for example: ease of implementation (close to the visualization sytstem and the data), allow for manual input by the user (preferably ASCII-based, with semantics), verification should be possible (to check for invalid definitions), and many more. To meet all these design considerations, is was decided to use the XML language[13] for storage of the FDL and as interface to other applications. For writing and reading feature specifications to and from FDL-files, the Apache Crimson parser (delivered with the SUN Java SDK) is used, but any other validating XML Parser could also be used. We use a *DTD* (Document Type Definition) for the verification of the FDL trees. The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements.

6. Conclusions and Future Work

We presented a framework for flexible and interactive, high-dimensional feature specification for data coming from computational simulation. For analyzing simulation data, a feature-based F+C visualization is a good approach, to cope with the data sets' large and high-dimensional nature and to guide the user and support interactive analysis. For F+C visualization interactive focus specification is very useful, if real-time updates of multiple linked views are available. Actual features in simulation data often only are captured with a complex type of specification (hierarchical specification, multiple data channels involved). This is why we believe, that using a simple language to define features hierarchically, namely our feature definition language, helps to extract and manage features during an interactive analysis session. In combination with using multiple InfoViz views (for data examination and feature specification) and SciViz views (for F+C visualization of the interactively extracted features) it is a very useful approach.

Future work will include extensions of the here presented FDL as well as of the analyzing tools. A parallel coordinates view[10] which has been developed earlier[8] can already be used passively to visualize the high-dimensional data, and will be integrated fully in the very near future, as well as new (hardware- and software-based) volume rendering techniques will be included, too. FDL extensions will mainly deal with including the views setting and couple it more tightly with the feature specification tree, as well as time-dependent issues. Currently only steady simulation data can be visualized and the logical next step will be, to enhance the FDL as well as all the corresponding visualization and inter-

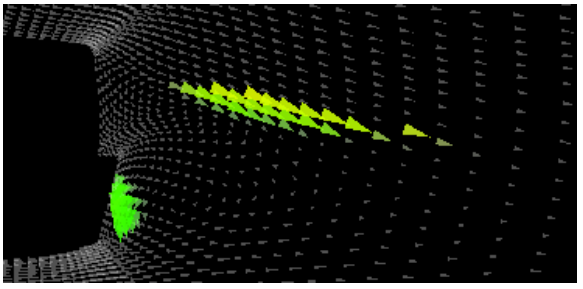

Figure 9: Step 5 of analysis: *interactive probing of V-velocity reveals different behavior of vortical structures, only downfacing parts are shown here.*

action views to cope with time-dependent data sets. Feature specification for time-dependent data sets will be one of the key-issues of future research.

Acknowledgements

This work has been carried out as part of the basic research on visualization at the VRVis Research Center in Vienna, Austria (http://www.VRVis.at/vis/), which partly is funded by an Austrian research program called Kplus. All data presented in this paper are courtesy of AVL List GmbH, Graz, Austria.

The authors would like to thank Robert Kosara, for his help with preparing this paper. Special gratitude goes also to our collegue Markus Hadwiger, who helped with parts of the implementation of the underlying mesh-library system, and the collegues from the Software Competence Center in Hagenberg, Austria, who helped with their knowledge about fuzzy sets and fuzzy combinations.

References

1. R. Becker and W. Cleveland. Brushing scatterplots. *Technometrics*, 29(2):127–142, 1987.
2. Andreas Buja, John A. McDonald, John Michalak, and Werner Stuetzle. Interactive data visualization using focusing and linking. In *Proc. of IEEE Visualization '91*, pages 156–163.
3. S. Card, J. MacKinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers, 1998.
4. Helmut Doleisch and Helwig Hauser. Smooth brushing for focus+context visualization of simulation data in 3D. In *Proc. of WSCG 2002*, Plzen, Czech Republic.
5. Ying-Huey Fua, M. O. Ward, and E. A. Rundensteiner. Structure-based brushes: A mechanism for navigating hierarchically organized data and information spaces. *IEEE Trans. on Visualization and Computer Graphics*, 6(2):150–159, 2000.
6. George W. Furnas. Generalized fisheye views. In *Proc. of the ACM CHI '86 Conf. on Human Factors in Computing Systems*, pages 16–23, 1986.
7. 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 IEEE Visualization 2000*, pages 489–492, 2000.
8. H. Hauser, F. Ledermann, and H. Doleisch. Angular brushing of extended parallel coordinates. In *Proc. of IEEE Symp. on Information Visualization*, pages 127–130, 2002.
9. H. Hauser, L. Mroz, G. I. Bischi, and E. Gröller. Two-level volume rendering. In *IEEE Transactions on Visualization and Computer Graphics*, volume 7(3), pages 242–252. IEEE Computer Society, 2001.
10. A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multidimensional geometry. In *Proc. of IEEE Visualization '90*, pages 361–378.
11. E. P. Klement, R. Mesiar, and E. Pap. *Triangular Norms*, volume 8 of *Trends in Logic*. Kluwer Academic Publishers, Dordrecht, 2000.
12. A. Martin and M. O. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proc. of IEEE Visualization '95*, pages 271–278.
13. Webpage of the World Wide Web Consortium on XML. See URL `http://www.w3.org/XML/`.
14. F.H. Post, H. Hauser, B. Vrolijk, R.S. Laramee, and H. Doleisch. Feature extraction and visualization of flow fields. In *Eurographics State of the Art Reports*, pages 69–100, 2002.
15. Ben Shneiderman. Dynamic queries for visual information seeking. Technical Report UMCP-CSD CS-TR-3022, Department of Computer Science, University of Maryland, College Park, Maryland 20742, U.S.A., January 1994.
16. C. Silva, J. Mitchell, and P. Williams. An exact interactive time visibility ordering algorithm for polyhedral cell complexes. In *Proc. of IEEE Symp. on VolVis '98*, pages 87–94, 1998.
17. M. O. Ward. XmdvTool: Integrating multiple methods for visualizing multivariate data. In *Proc. of IEEE Visualization '94*, pages 326–336.
18. Pak Chung Wong and R. Daniel Bergeron. Multiresolution multidimensional wavelet brushing. In Roni Yagel and Gregory M. Nielson, editors, *Proc. of the Conf. on Visualization*, pages 141–148, Los Alamitos, October 27–November 1 1996. IEEE.

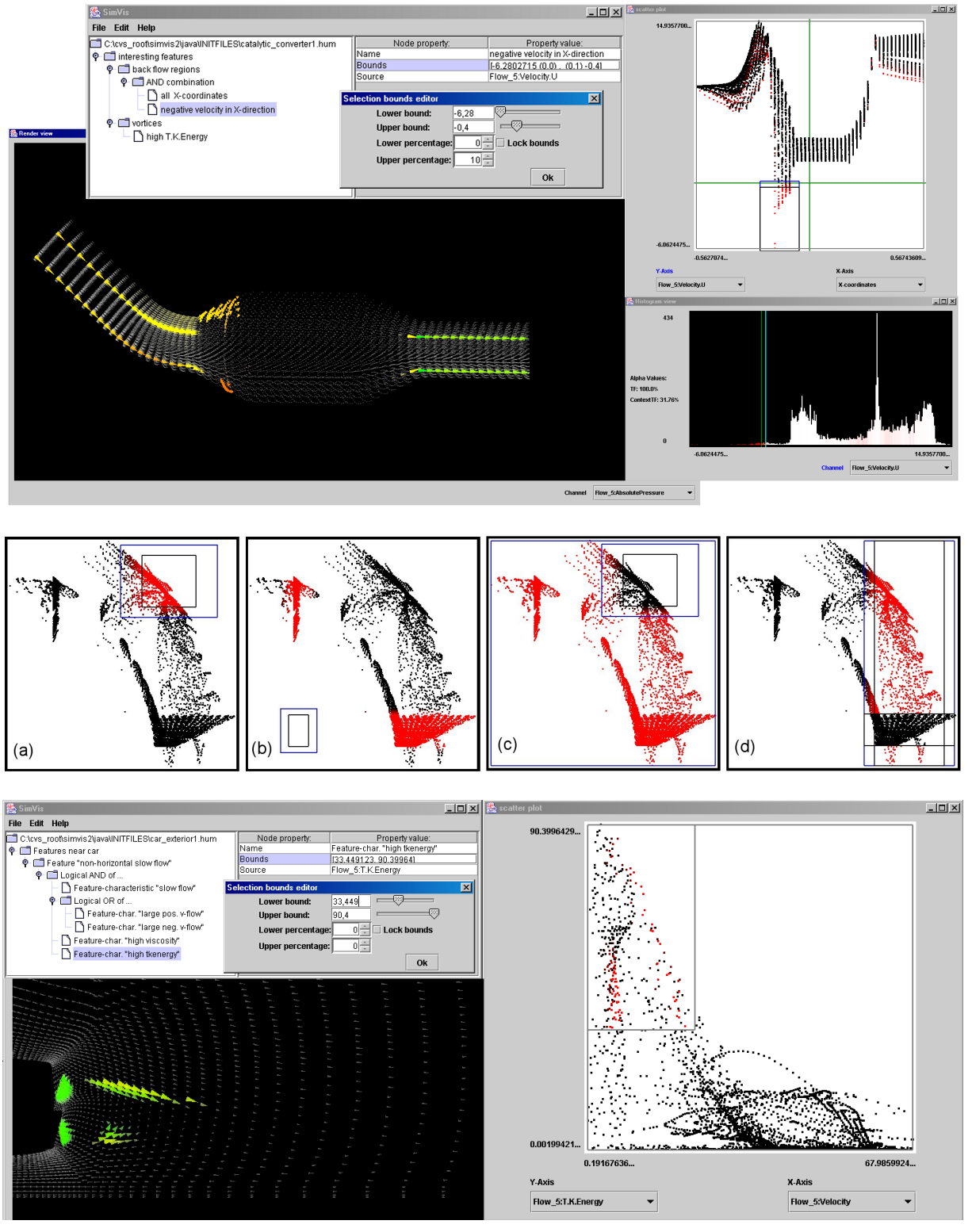

Figure 10: Two examples for feature-based flow visualization using our framework for interactive feature specification and four illustrations of different combination modes for smooth brushes (middle row) *(for verbose captions see figures 1, 3, and 8).*