Time-based Visualization of Large Data-Sets An Example in the Context of Automotive Engineering

Werner Sturm, René Berndt, Andreas Halm, Torsten Ullrich, Eva Eggeling

Fraunhofer Austria Research GmbH
Visual Computing, Graz, Austria
&
Institute of ComputerGraphics and KnowledgeVisualization,
Graz University of Technology, Austria

Email: {rene.berndt, andreas.halm torsten.ullrich, eva.eggeling}@fraunhofer.at, werner.sturm@student.tugraz.at

Dieter W. Fellner

Institute of ComputerGraphics and KnowledgeVisualization,
Graz University of Technology, Austria
&
Fraunhofer IGD and TU Darmstadt
Darmstadt, Germany

Email: d.fellner@igd.fraunhofer.de

Abstract—**Automotive systems can be very complex when using multiple forms of energy. To achieve better energy efficiency, engineers require specialized tools to cope with that complexity and to comprehend how energy is spread and consumed. This is especially essential to develop hybrid systems, which generate electricity by various available forms of energy. Therefore, highly specialized visualizations of multiple measured energies are needed. This paper examines several three-dimensional glyph-based visualization techniques for spatial multivariate data. Besides animated glyphs, two-dimensional visualization techniques for temporal data to allow detailed trend analysis are considered as well. Investigations revealed that Scaled Data-Driven Spheres are best suited for a detailed 3D exploration of measured data. To gain a better overview of the spatial data, Cumulative Glyphs are introduced. For trend analysis, Theme River and Stacked Area Graphs are used. All these visualization techniques are implemented as a web-based prototype without the need of additional web browser plugins using X3DOM and Data-Driven Documents.**

***Keywords*–scientific visualization; spatio-temporal multivariate; glyph based; trend analysis; web-based.**

I. Introduction

Scientific visualization is a growing field of research in computer graphics. Recent visualization techniques have been presented, among others, at the *International Conference on Creative Content Technologies* (CONTENT), where the foundation of this article has been discussed [1].

To make fast decisions based on the data, scientists and engineers need to interpret valuable information efficiently. The complexity of data constantly increases and therefore highly specialized and individual visualization methods are needed [2].

Especially automotive engineering encounters the need to develop more efficient energy saving systems to provide longer duration with a given amount of energy. Thus, automotive engineers require specialized tools that support them to achieve this goal. An increasing research field of automotive engineering are hybrid vehicles (e.g., combination of combustion engine and electric propulsion system). As electricity is used for propulsion, several forms of energy can be used to generate electricity [3] (e.g., heat, kinetic energy or sun light). A hybrid vehicle that uses multiple forms of energy can be a very complex system. Every single component that outputs or consumes energy must be considered to ensure high efficiency. To cope with such complex systems, engineers need to be able to understand the global behavior of such hybrid systems.

Measured values of several forms of energy can be visualized to get a better understanding how energy is consumed and emitted. A measurement might be a recording of a specific scenario (e.g., start the engine and driving up the hill at 40°C temperature), thus engineers are interested in the energy trend as well. Considering that, the visualization should also represent the systems behavior over time.

In this article, several multivariate visualization techniques (two- and three dimensional) are evaluated. In addition to that, a prototypal implementation using X3DOM [4] and Data-Driven Documents (D^3) [5] technology is discussed. This interactive prototype demonstrates two glyph-based three-dimensional (3D) visualization methods including Scaled Data-Driven Spheres (SDDS) and three different two-dimensional (2D) visualization methods called ThemeRiver [6], Stacked Area Graph and Stacked Cumulative Percent Plots [7].

II. Related work

There exist many visualization techniques for multivariate data like Geometric Projection, Pixel-Oriented Techniques and Hierarchical Display [8]. The given data model is not only a general multivariate data set, but it also has temporal and spatial components, which play an important role.

Temporal or so-called time-oriented data have become a separate research field. There are various visualization techniques for temporal data and different purposes (e.g., ThemeRiver, TimeWheel) [9]. Generally, they have in common the depiction of the change over time.

On the other side, previous research also found multiple visualization methods for spatial-multivariate data, which are mostly used for scientific visualizations. These techniques include surface based rendering, direct volume rendering and glyph based techniques [10]. A large field of application for visualization methods for spatial-multivariate data are medical data visualizations (e.g., data from Nuclear Magnetic Resonance (NMR)). In the next Section, several related visualization techniques are presented, which have been considered for visualizing various forms of energy for automotive engineering.

A. Quantitative Texton Sequences

Quantitative Texton Sequences (QTonS) is a bivariate visualization technique. It maps one dimension to a spectrum color sequence and the other to a QTonS sequence (see Figure 2). These two corresponding dimensions are distributed over an 2D area. A Quantitative Texton Sequence is a set of textures (textons) as illustrated in Figure 1.

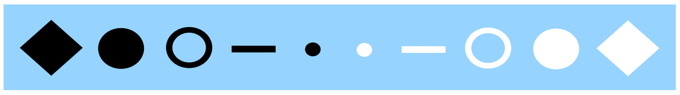
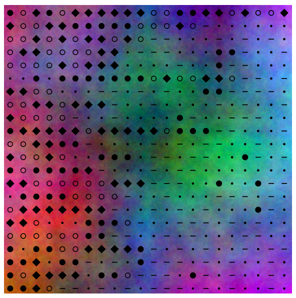
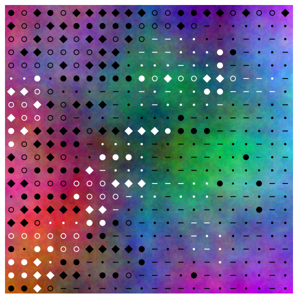


Figure 1. Example of a Quantitive Texton Sequence.

"A texton is here defined as a small texture element that when presented in a dense field forms a texture." [11].



(a) QToS with only negative values



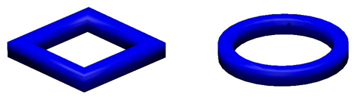
(b) QToS with both negative and positive values

Figure 2. Illustration of a Quantitive Texton Sequence visualization (QToS). Color spectrum represents the first variable and textons, placed at the same position, represent the corresponding additional variable.

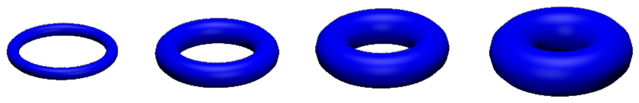
Fundamentally, values are mapped to individual Textons. The higher the absolute value is, the bigger the Textons surface will be. This means, that an area covered by Textons densily represents an area of high absolute values. To differentiate negative and positive values, black and white Textons are used. In this case, black Textons represent negative values (see Figure 2).

B. Superquadric Glyphs

Superquadrics are a category of geometrical shapes, which basically are derived from ellipsoids and quadrics. Their appearance can be changed by changing their parameters (e.g., roundness of corners, size, color, etc.). This property can be exploited to map values to the glyphs appearance.



(a) Cornered and round supertorus.



(b) Supertorus of various thickness.

Figure 3. Glyph-based visualization using superquadrics. Cornered supertorus represent negative values, round ones represent positive values (a). Absolute values are mapped to thickness of glyph (b).

Generally, this method is called glyph-based visualization and it enables a wide range of possibilities for displaying spatial multivariate data. A supertorus can be manipulated by its size (major radius), thickness (minor radius), roundness and color (as illustrated in Figure 3b and 3a). Thus, Superquadric Glyphs can be used to visualize up to four dimensions in addition to the glyphs position.

C. Scaled Data-Driven Spheres

Scaled Data-Driven Spheres is another glyph based visualization technique, which extends the 2D Data-Driven Spots [12] to 3D. SDDS use spheres as glyphs and it implies the properties of superquadrics like color and size. In addition to that, they are easier to interpret due to their simple shape and viewing angle independence. A previous user based evaluation of superquadrics and SDDS revealed that SDDS result to a lower error rate in value estimation and relationship identification [13].

Basically, SDDS use the spheres size to represent the value (see Figure 4). Scaled Data-Driven Spheres can visualize

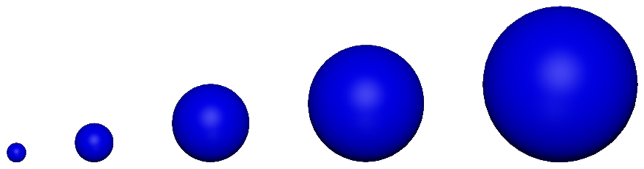


Figure 4. Glyph based visualization using Scaled Data-Driven Spheres (SDDS). Linear mapping of values to SDDS' size.

two additional dimensions (beside its position) using their properties color and size:

$$r = r_{max} \frac{v - v_{min}}{v_{max} - v_{min}}, \tag{1}$$

whereas r is the resulting radius of the sphere based on same input value $v \in [v_{min}, v_{max}]$.

D. Theme River

Theme River is a two-dimensional visualization technique designed for trend analysis over a serial dimension (e.g., time) [6], [14]. Theme River enables users to compare different identities, which occur parallel. Every identity is represented by a colored current within the river (see Figure 5).

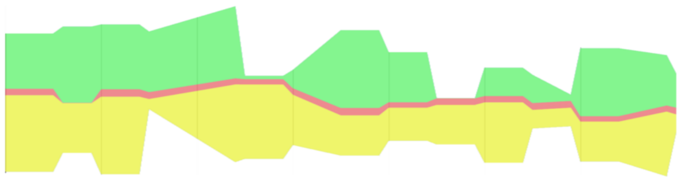


Figure 5. Illustration of Theme River with three currents. The width of current represents the corresponding value. It is primarily used to identify trends. Thus, labels indicating the exact value are not essential.

The higher a value is, the wider the corresponding current will be. If a value changes along the serial dimension, the currents width will proportionally change to it. Hence, abrupt changes and long term trends can be recognized easily. If a value equals zero, the appropriate current will disappear until it changes its value again.

E. Stacked Area Graph

Stacked Area Graph represents serial values as areas. The x-axis represents a serial dimension and the y-axis represents the corresponding value. When values change over a serial dimension, the height of the corresponding area will change as well. If multiple values are depicted, areas are not overlapping. They are arranged as a stack. Thus, the total sum of all depicted values is represented by the total hight of the stack (see Figure 6).

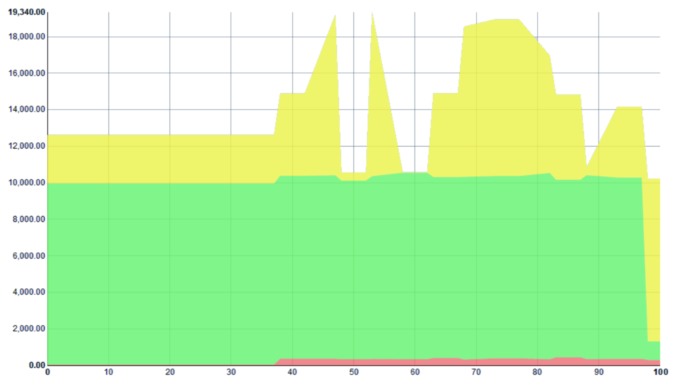


Figure 6. Illustration of Stacked Area Graph with three elements. The height of stack element expresses the corresponding value.

This enables the user to recognize the total amount easily. Like Theme River - if a value changes to zero, the corresponding area will disappear. If the value changes again, the area will appear at the same stack level where it disappeared before.

F. Stacked Cumulative Percent Plot

Stacked Cumulative Percent Plot (SCPP) is another graphical method for visualizing trends [7]. In contrast to Theme Rivers and Stacked Area Graphs, SCPP depicts the percentage instead of the value as stacked areas.

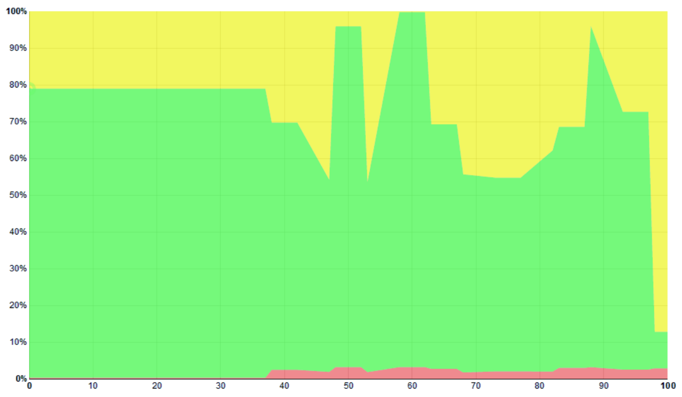


Figure 7. Illustration of Stacked Cumulative Percent Plot with three elements. The height of stack element represents the values percentage of total sum.

The total graph is always stacked up to 100% (as in Figure 7). When values are normalized, it enables the user to identify how much a single component contributes to the total amount.

G. X3DOM

As these visualization techniques have to be implemented with the aid of current web technologies, X3DOM is used to render the 3D visualization. Generally, X3DOM is a new approach to enable web browsers to render 3D content without the need of further plugins [4]. The current HTML5 standard defines X3D (derived from VRML97) to be used to provide 3D content, but it does not define how this 3D content has to be handled. For that, X3DOM has been approached to connect the web browsers *Document Object Model* (DOM) containing the X3D data, with the internal X3D runtime. The X3D runtime does all 3D tasks, which are needed to render the scene defined by the X3D data. X3DOM enables web developers to create 3D content in a declarative way as used for creating SVG and common HTML elements. Thus, there is no need to struggle with low-level graphics interfaces like OpenGL provided by WebGL [15]. Due to its observer architecture, it also supports dynamic DOM changes, which effect an analogous update of the X3DOM runtime content. X3DOM has not been declared as a web standard, but like XML3D [16], it runs for it.

H. JQuery

JQuery is a JavaScript framework, which simplifies the development of dynamic web pages [17]. It provides functionalities to modify the web browsers DOM and to invoke remote procedures via AJAX. It's *selector* enables programmers to select DOM elements by their CSS attributes and update them via implicit iterations. JQuery has become a first choice for web developers.

I. Data-Driven Documents

Data-Driven Documents (D^3) is a JavaScript library for manipulating the DOM based on data (data-driven) [5]. It is designed to cope with large data sets and it allows to bind data to the DOM combined with dynamic properties for animations and user interactions. It uses several well known web technologies like SVG, CSS3 and HTML5 to generate a representation of the defined data. Therefore, no further libraries or plugins are needed. Moreover, it uses similar methods called selectors as JQuery does.

III. Preconditions

A. Data model

Problem-specific visualizations are tailored for specific data. Thus, it is important to know the underlying data structure. In our case, we have spatio-temporal multivariate data. The data consist of the energy type, the 3D position where the data were collected, the data acquisition time and the measured value itself. The value can be positive or negative. A single measurement represents a period and consists of various value sequences. As illustrated in Figure 8,

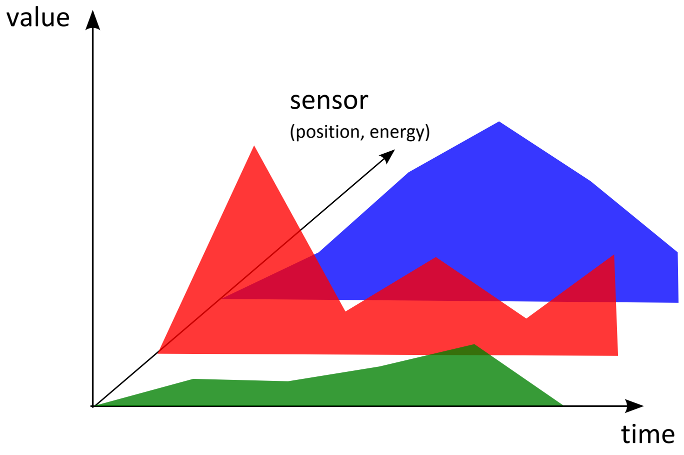


Figure 8. Representation of the data model. Dimension *sensor* combines 3D position and form of energy.

a single value sequence represents the recording of a single sensor (measuring point). All values are measured parallel and it can be assumed that all sensors have been synchronized. Hence, the n^{th} value of every value array represents the same time.

The multivariate data have six dimensions:

- form of energy
- value
- time
- 3D position (one scalar for each dimension)

B. Visualization

The visualization must be capable to display five distinct forms of energy and an arbitrary number of sensors in an explorable way. These forms of energy should be easy to interpret, to distinguish and to compare. In addition, the trend over time must be easy to investigate to satisfy the requirements of automotive engineers. To achieve that, both 2D and 3D visualization techniques will be considered.

Summing up, an optimal visualization fulfills these requirements:

- displaying at least five distinct forms of energy measured by several hundred sensors
- easy exploration of data, especially considering spatial and temporal dimensions
- opportunities to compare measuring points and different forms of energy
- gives a global overview about the data
- opportunities for detailed trend analysis

IV. Examination

Basically, the human's perception capabilities are limited to a 3-dimensional spatial sense plus a perception of time. This results to a 4D world, in which we live in. This limitation also narrows the possibilities of beneficial visualization methods, while the perception of time can only be used via animations. Thus, a single image can not exploit the perception of time. Regarding this fact, properties of geometrical shapes (glyphs) (e.g., color, size, roundness, pattern and orientation) can be used to gain the feasibility to visualize even more dimensions. This is needed to display multi-dimensional data.

An investigation of existing techniques revealed that a combination of 2D- and 3D visualization techniques lead to a better global insight into the measuring data. Moreover, this allows the user to investigate the data from different points of interest.

A. 3D visualization

If the data set has a spatial component and it is important to know its location, it is common to use a visualization technique that retains this spatial information. The drawback of this decision is, that every visualized dimension consumes a single dimension in the visualization space. For our problem, all three dimensions of the spatial variable are used because the cognition of the energy's location is essential. Therefore, no spatial dimension for visualizing time is left. Three-dimensional visualizations also allow more freedom to explore the spatial data in a natural way.

1) Quantitative Texton Sequences: QTonS are considered to be used as a render mode of the three-dimensional space to represent the two dimensions *value* and *form of energy*. While exploring the 3D-space, the view is updated by the QTonS technique to represent the amount of energy in the viewed region. The mapped colors represent the *form of energy* and the texons represent the *value* of the data (see Figure 2). The color of textons indicates the sign of its *value*. White textons represent positive values and black textons represent negative values. This combination of 3D and QTons visualizes all needed information of the measurement at a single point of time. Therefore, a control to navigate through the measuring period is needed.

The drawback of this approach is that the location of the visualized energies is not clearly evident. The user can only perceive how much energy occurred in a specific direction calculated from the viewpoint. Therefore, it is not possible to investigate a single location. This results to an ineffective explorable visualization. Moreover, it is rather suitable for data sets of high spatial density. Such a high density might be reached by some measurements but this is not the common case. It is assumed that common measurements have up to several hundred measuring points distributed over the vehicle.

2) Superquadric Glyphs: Here, the approach called torus-based Superquadric Glpyh (supertorus) is examined, which is discussed in [18]. Compared to QTonS', this technique is more meaningful for displaying 3D-spatial data, because glyphs represent the spatial information by their 3D-position implicitly. The dimension *form of energy* is mapped to the glyphs color and the corresponding absolute *value* is mapped to its thickness as shown in Figure 3b. To indicate if a *value* is either positive or negative the roundness is used therefore (see Figure 3a). A round supertorus represents a positive value.

Superquadric Glyphs scale very well from few data sets to several hundred. The user can easily identify regions with high occurrences of a specific form of energy. If a scale is provided, the user can estimate the value of a single glyph. Therefore, glyphs can also be compared to understand relationships of different energies. To preserve the size of glyphs perspective-independent, an orthogonal projection is used. Contrasting colors should be chosen as discussed in [19] and [20] to be distinguishable easily.

A serious disadvantage of superquadrics is that users can not easily interpret the glyphs roundness and thickness when glyphs are placed densely. In addition to that, superquadrics are not viewing angle independent. According to the user study [13], Superquadric Glyphs are not as effective as Scaled Data-Driven Spheres (SDDS). The next Section will discuss SDDS in detail.

3) Scaled Data-Driven Spheres: In contrast to Superquadric Glyphs, Scaled Data-Driven Spheres are viewing angle-independent. To distinguish different types of spheres, contrasting colors are used. According to [20], humans perception struggles to differentiate more than 12 color values. Therefore, up to 11 (plus one for background) colors can be used to represent different glyphs. In this case, up to 5 different forms of energy are needed to be displayed. This allows to use 2 colors for each form of energy to distinguish positive and negative values (10 colors). To clarify that these two colors belong to the same form of energy they should have the same ground color (e.g., light red and dark red). Therefore, light colored spheres represent positive values and negative values are represented by dark colored spheres (as in Figure 9). The contrasting base colors of all five *forms of energy* are green, yellow, red, blue and magenta.

The remaining free color value can be used to display the outline of a car as a semitransparent model. This assists the user to interpret 3D positions easier. The car model should not disturb the perception of the visualized measuring data. Thus, light gray is used since gray is not a signaling color. As a result, all dimensions of the data at a specific moment can be visualized in a comprehensible way.

However, SDDS underly several side effects of the 3D rendering process. To achieve a three-dimensional impression, objects have to be shaded, which effects a distortion of the objects original color. When using a semitransparent car model, the color of the car model also distorts the color of SDDS that are located behind the model. This can result to an unclear color identification and should be considered.

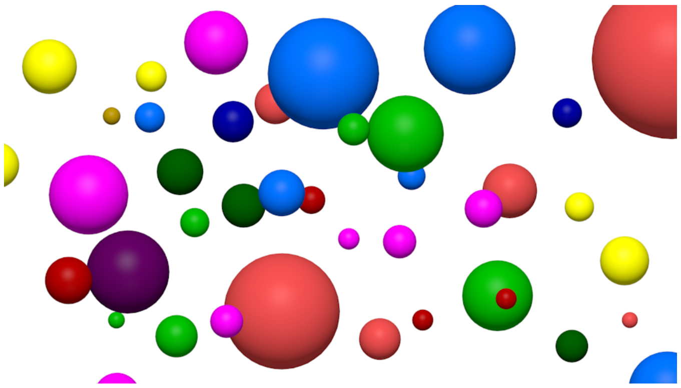


Figure 9. Illustration of cluttered Scaled Data-Driven Spheres (SDDS) using contrasting colors. Each base color represents a distinct form of energy.

To illustrate the trend over time, a key frame animation showing the change of measuring values is used. Every key frame equals one measuring time, which is calculated by mapping values to the corresponding spheres' size. The glyphs size and color (when value is changing its sign) has to be interpolated between each frame. Consequently, when key frames are shown in sequence, an animation depicting growing and shrinking spheres will be the result.

Due to the ease of interpreting SDDS, their low error rate in value estimation and relationship identification, SDDS are the most suitable visualization technique to display detailed measuring data spatially.

B. Cumulative Glyph

In some cases, engineers do not want to get a detailed insight in the data instantly. They might want to get a more general overview of the data to spot interesting regions to investigate more in detail afterwards. Cumulative Glyph (CG) is an approach for this demand. A CG is a representative of a specific spatial region (e.g., single wheel, engine, etc.) of the data. In a single region, there might be plenty of sensors. To understand the overall incidence of energy in this region, it still can be difficult to estimate when using SDDS. Therefore, all values of the same form of energy within this region will be cumulated to a single value. This cumulated value represents the total occurrence, which can be compared with other cumulated values of other regions.

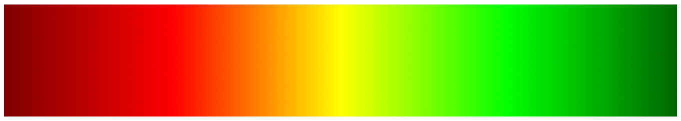


Figure 10. Color scale used for color mapping. Green indicates high occurrence of energy. Red indicates low energy.

$$
\begin{aligned}
ratio &= \frac{v - v_{min}}{v_{max} - v_{min}} \\
red &= \begin{cases} 0.4 + \frac{v - v_{min}}{\frac{1}{4}(v_{max} - v_{min})} \cdot 0.6 & \text{if } 0 \leq ratio \leq 0.25 \\ 1 & \text{if } 0.25 < ratio \leq 0.5 \\ 1 - \frac{v - \frac{1}{2}(v_{max} + v_{min})}{\frac{1}{4}(v_{max} - v_{min})} & \text{if } 0.5 < ratio \leq 0.75 \\ 0 & \text{if } 0.75 < ratio \end{cases} \\
green &= \begin{cases} 0 & \text{if } 0 \leq ratio \leq 0.25 \\ \frac{v - \frac{1}{4}v_{max} + \frac{3}{4}v_{min}}{\frac{1}{4}(v_{max} - v_{min})} & \text{if } 0.25 < ratio \leq 0.5 \\ 1 & \text{if } 0.5 < ratio \leq 0.75 \\ 1 - \frac{v - \frac{3}{4}v_{max} - \frac{1}{4}v_{min}}{\frac{1}{4}(v_{max} - v_{min})} \cdot 0.6 & \text{if } 0.75 < ratio \end{cases} \\
blue &= 0
\end{aligned} \tag{2}
$$

To visualize cumulated values, color mapping is used. As illustrated in Figure 10, colors as green, yellow and red are used to depict the amount of energy. Green represents a high, yellow an average and red a low (negative) amount of energy.

Despite the fact, that many color values, which are close to each other on this color scale, can not be compared easily (subjective colors), this representation has enough potential to offer the user an overview to know where high occurrences of energy exist. This color scale is chosen because the rainbow color map is commonly accepted by users. Of course, other color scales can be used that are considered to be easier to differentiate. This topic is discussed in [21], [22], [23], [24]. Every CG consists of n (where n is the number of different energies) discs, which are arranged circularly. Every disc represents a single form of energy. In this case, a CG consists of five discs. Every disc is colored by the mapping rule as Equation 2 depicts. If a form of energy has not been measured in this region, the corresponding disc will be gray and represent no value. In addition to that, the overall amount of energy within a region is mapped to the CG's size. This allows the user to identify if a region involves a high occurrence of energy. Like SDDS, Cumulative Glyphs can be animated to show the energy trend over time as well. Colors of discs and the glyphs size have to be interpolated between every key frame. In this way, the user is able to get an idea how energy behaves in several locations over time.

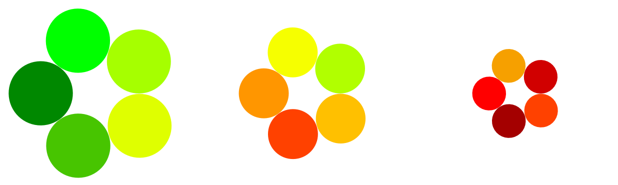

(a) QToS with only negative values (b) QToS with both negative and positive values (c) QToS with both negative and positive values

Figure 11. Illustration of Cumulative Glyph (CG) representing cumulative values of regions within the data space. CG of high overall cumulative value (a). CG representing a region of average cumulative value (b). CG of low energy value (c).

Due to the Cumulative Glyphs shape, CG's should have a camera-relative orientation to be view-independent. As depicted in Figure 11, it is difficult to recognize what form of energy is represented by a disc. For that, it is recommended to use icons to label discs to improve the ease of interpretation or at least, a legend should be provided.

C. 2D visualization

1) Theme River: Due to Theme River's properties, it is suitable to depict the trend of measured values. While the serial dimension is the time component of a measurement, every sensor is represented by one current within the river. Currents that represent sensors measuring the same form of energy are of the same basic color and are placed side by side. As a result, these sensors together generate a wide current representing the total amount of one form of energy. Additionally, the total width of the river describes the total amount of energy. This allows users to compare trends of different forms of energies and various sensors at once.

A drawback of Theme River is the difficulty of estimating values. Without any labels, users can hardly guess what value is represented by a given width. Moreover, Theme River works only with positive values. Thus, either absolute values should be used (where negative values are represented by darker colors) or two graphs, one for positive and the other one for negative values might be a solution. The latter solution should be preferred, because the former one treats positive and negative equally, which leads to a wrong representation of the total energy occurrence.

2) Stacked Area Graph: Stacked Area Graphs can also be used to examine the energies trend over time. Every measuring point is represented by an area. As implemented in Theme River, measuring points representing the same *form of energy* are placed next to each other. In addition, they are of the same base color to represent the overall occurrence of that specific *form of energy* as a cumulated area. This enables users to investigate the trend and to identify the entire amount of energy in an area (e.g., engine or front wheel). Negative values are represented by stacks below the x-axis. To navigate through time, controls for setting the displayed interval should be provided. Possibilities to set start time and end time independently allows users to *zoom* within the dimension of time.

3) Stacked Cumulative Percent Plot: Stacked Cumulative Percent Plots are very similar to Stacked Area Graphs. Due to the depiction of percentages instead of values, users can easily recognize which energy contributes the most. To compare different *forms of energy* based on meaningful percentages, values are normalized. Furthermore, absolute values are used to handle negative values. To identify that an area represents negative values, darker colors are used (as used for SDDS).

V. PROTOTYPE

The aim of this prototype is to test all chosen visualization techniques and proofing that these techniques can be implemented as a web-enabled application using X3DOM of version 1.3 and Data-Driven Documents (version 2.9.6).

A. Architecture

As depicted in Figure 12, the main parts of the prototype are the database, the web server and the client, which is a web

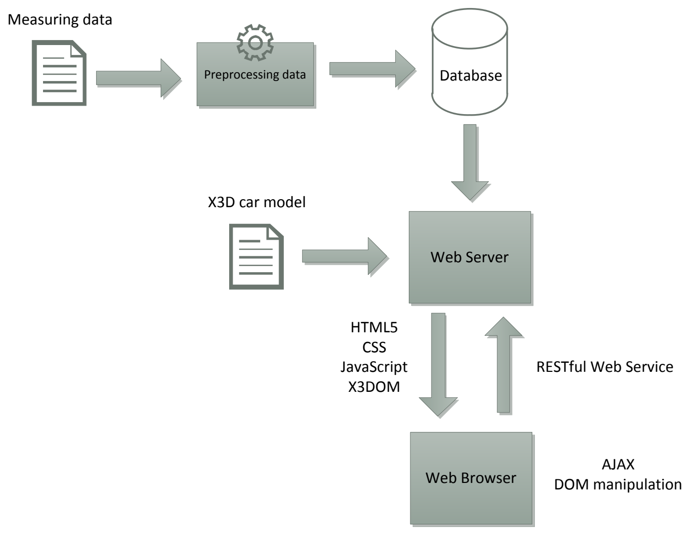


Figure 12. General architecture of prototypal implementation. Original and reduced approximated values are stored in database. Web server provides HTML content and data to the client. Client performs complete visualization using X3DOM and Data-Driven Documents.

browser supporting 3D within web sites. The major part of the applications logic is run within the clients web browser to provide dynamic visualizations, which respond quickly to user interactions. The web browser requests needed measuring data from the web server via RESTful web services and maps it into a data format that can be visualized by X3DOM or D^3. Hence, the web server primary has to provide the requested data and further visualization-specific calculations are performed client-side.

1) Data Preprocessing: Transferring all data to the client would be very time-consuming. Thus, the data are preprocessed to reduce the amount of data. Generally, a measurement with n measuring points and t value acquisitions results to a data complexity of $S(n,t) = \mathcal{O}(n \cdot t)$. As the amount of measuring data can be massive, the Douglas-Peucker algorithm is used to to reduce the number of values [25]. This algorithm has an expected complexity of $\Theta(nt \cdot \log(nt))$; worst case is $\mathcal{O}((nt)^2)$. This works because the sequence of measured values can be interpreted as a curve representing the trend (see Figure 8). After applying this geometric algorithm, an approximation of the original curve with less points, which still preserves the overall structure of the curve will be the result. In other words, this algorithm filters all non-distinctive values. Consequently, the less data the server has to find and transfer to the client, the faster the system responds to the user. Moreover, users want to navigate through the data using different time-zoom levels. For that, different degrees of approximations are needed, which are stored in the servers database after the preprocessing task.

2) Web Server: Basically, the web server provides HTML content including the static 3D car model in X3D format, which can be understood by X3DOM. However, the servers main task is to provide measuring data to the client. For that, it offers several RESTful web services, which provide the data formatted using *JavaScript Object Notation* (JSON). Of course, clients do not receive all data at once. For both 3D visualizations (SDDS and Cumulative Glyph) the client commonly requests only data of a single moment. But all time-oriented two-dimensional visualizations and animations of SDDS and Cumulative Glyphs, used for depicting the trend over time, need a data set of the desired time span. Users might select a long period (or even the complete period), which would cause a huge amount of data to be transfered to the client. Considering that, the web server limits the maximum amount of values to be transfered and selects the best approximation of the data (as discussed in Section V-A1), which does not contain more values as limited. This allows the client to request any time span without overloading.

3) Client: As already mentioned, the most of the applications logic is performed client-side by JavaScript. Users should be able to explore data in an interactive way. It uses several libraries to handle these tasks. JQuery is used for user interface interactions, performing *Asynchronous JavaScript and XML* (AJAX) calls and DOM manipulations. X3DOM is applied to visualize 3D content. The 3D scene is manipulated via DOM manipulations and D^3 provides all functionalities to draw 2D graphs.

B. 3D using X3DOM

The observer architecture of X3DOM, which connects the web browsers DOM with the internal X3D runtime, allows to modify the 3D scene via the DOM itself. Moreover, X3D predefines simple objects like box, sphere, cone and cylinder. Therefore, a SDDS can be created easily by using a sphere object (see Listing 1). Resulting X3D tags are depicted in Listing 2.

```
function showSDDS(x, y, z, size, color, id) {
  //
  // create tag <transform>
  // and append it to scene tag
  //
  $('<transform></transform>', {
        id: 'sdds' + id,
        'class': 'sdds',
        translation: x + " " + y + " " + z,
        scale: size + " " + size + " " + " " + size
    }).appendTo("#scene");
  //
  // create tag <sphere>
  // and append it to tag <shape>
  //
  $('<sphere></sphere>', {
        'class': 'sphere'
    }).appendTo(
      // create tag <shape>
      // and append it to tag <transform>
      $('<shape></shape>', {
          id: 'sphere' + id
      }).appendTo("#sdds" + id)
    );
  //
  // create tag <material>
  // and append it to tag <appearance>
  //
  $('<material></material>', {
      id: 'material' + id,
      transparency: 0.2,
      diffuseColor: color
    }).appendTo(
        // create tag <appearance>
        // and append it to tag <shape>
        $('<appearance></appearance>')
        .appendTo("#sphere" + id)
    );
}
```

Listing 1. JavaScript code to create a sphere with X3DOM representing a Scaled Data-Driven Sphere. All X3D state changes are performed via DOM manipulations using JQuery.

```
<X3D xmlns="http://www.web3d.org/specifications/x3d-
    namespace" id="x3d_element" showStat="true" showLog="
    false">
  <scene id="scene" pickMode="box">
  ...
    <transform id="sdds1" class="sdds"
        translation="0 0 0"
        scale="10 10 10">
      <shape id="sphere1">
        <sphere class="sphere"></sphere>
        <appearance>
          <material id="material1"
             transparency="0.2"
             diffusecolor="#FF6666">
          </material>
        </appearance>
      </shape>
    </transform>
  ...
  </scene>
</X3D>
```

Listing 2. X3D tags defining a Scaled Data-Driven Sphere.

Cumulative Glyphs can be created analogously using cylinders arranged circularly as depicted in Figure 13. To include a static car model provided as a file, the inline function can be used as shown in Listing 3.

```
<transform rotation="0 1 0 1.57">
  <appearance>
      <material transparency=".8"></material>
  </appearance>
  <inline id="carModel" nameSpaceName="carmodel"
     DEF="carModel" url="data/x3d/carModel.x3d"></inline>
</transform>
```

Listing 3. Include X3D car model via inline tag.

To animate glyphs to depict the values change over time, X3DOM implements *TimeSensors*, *Interpolators* and *Routes* defined by the X3D standard. *TimeSensors* provide a timer to trigger steady events within a defined period. *PositionInterpolators* are connected to *TimeSensors* via *Routes*, which provide a 3D vector interpolated between two defined vectors depending on the timers progress. These output vectors are passed to the spheres attribute *scale* (which expects a 3D vector) resulting a smooth animation showing a growing or shrinking sphere. Colors of glyphs also have to be changed during an animation. For that, X3D offers *ColorInterpolators*, which work analogously. This concept also supports long lists of key frames and therefore, any animation can be defined (see Listing 4).

1) Compatibility Issues: Nevertheless, there have been some issues with X3DOM. As mentioned in Section IV-A2, when using glyphs, an orthogonal projection should be used. Unfortunately, X3DOM does not implement *OrthoViewpoint* (defined by X3D). Thus, a perspective viewpoint is used instead. As depicted in Figures 14 and 13, transparency is applied to the car model and most glyphs are located within it, which led to several rendering problems. Many glyphs have not been rendered at all. To solve this problem, a work-around that reverses the rendering order in a way that glyphs are rendered at last is used. In addition to that, it was not possible to trigger mouse events of glyphs because there is no way to click *through* the transparent car model. Another problem was to orientate Cumulative Glyphs relatively to the camera. For that, X3D defines a *ProximitySensor* but X3DOM does not support it (yet).

```
<X3D xmlns="http://www.web3d.org/specifications/x3d-
    namespace"
    id="x3d_element" showStat="true" showLog="false">
  <scene id="scene" pickMode="box">
  ...
    <timesensor id="Timer" loop="loop"
      cycleinterval="3" enabled="true"/>
    <positioninterpolator id="PosInterpolator"
      key="0 0.5 1" keyvalue="5 15 5"/>
    <route fromnode="Timer" fromfield="fraction_changed"
      tonode="PosInterpolator" tofield="set_fraction"/>
    <route fromnode="PosInterpolator" fromfield="
        value_changed"
      tonode="sdds1" tofield="set_scale"/>
    <colorinterpolator id="ColorInterpolator"
      key="0 0.5 1"
      keyvalue="1 1 1 0 0 0 1 1 1"/>
    <route fromnode="Timer" fromfield="fraction_changed"
      tonode="ColorInterpolator" tofield="set_fraction"/>
    <route fromnode="ColorInterpolator" fromfield="
        value_changed"
      tonode="material1" tofield="diffuseColor"/>
  ...
  </scene>
</X3D>
```

Listing 4. Example of an animation with three key frames defined in X3D.

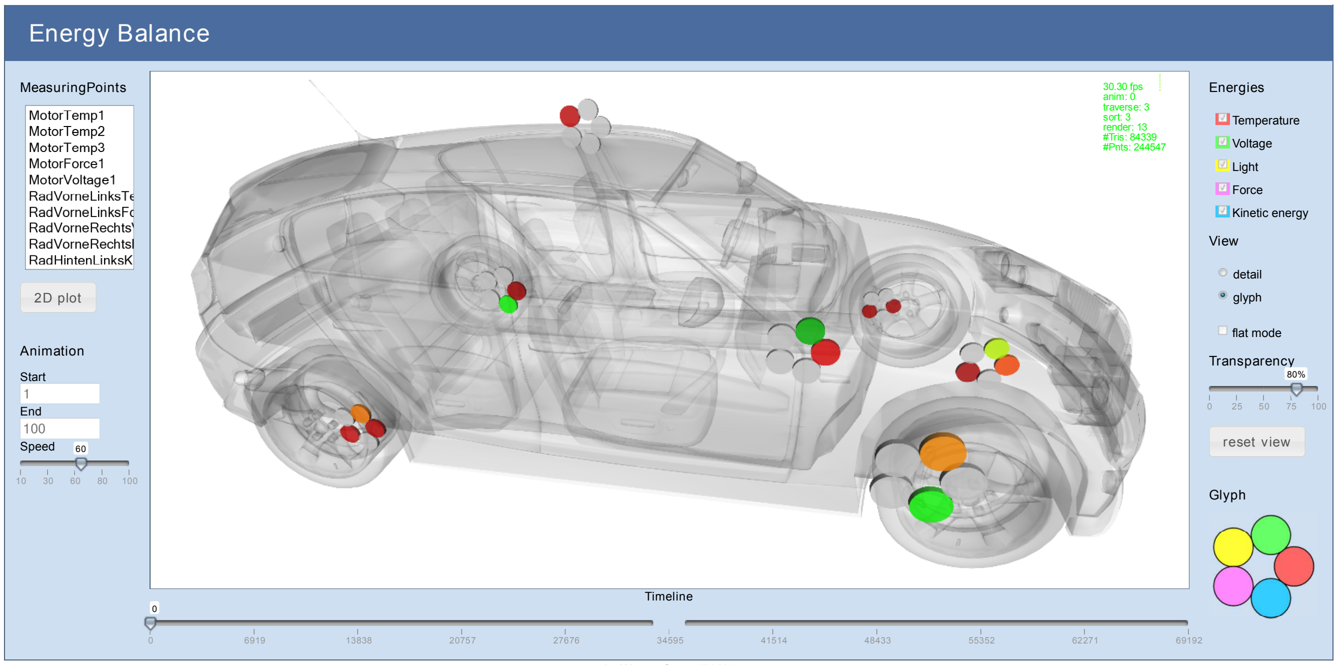
Regarding web browser compatibility, WebKit-based web browsers such as Google Chrome (version 23.0.1271.95) and Firefox (version 17.0.1) perform well with X3DOM. They also implement WebGL natively. Despite that X3DOM implements a fallback model in case that the web browser does not support 3D natively or via X3D/SAI plugin, as a last resort it tries to use Adobe's Flash plugin to render the scene. For example, this happens when using Microsoft's Internet Explorer (version 9) with no other plugins installed. Besides performance issues, the result was not acceptable because there have been to many rendering problems. This is a little insight into web browser compatibility when using X3DOM besides that evaluating compatibilities is not a goal of this paper.

C. 2D using Data-Driven Documents

An investigation of multiple graphing libraries such as Grafico [26], which is based on Raphaël and Prototype.js revealed that D^3 performs very well even with bigger data sets (e.g., 1000 data points).

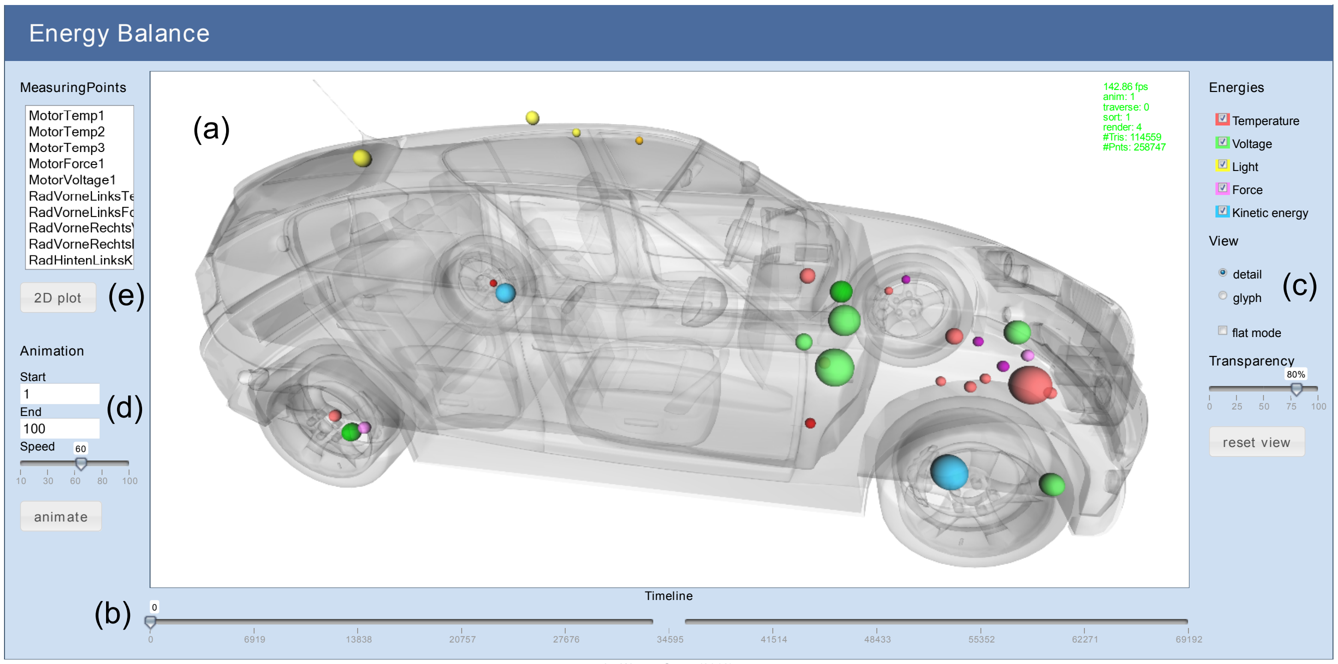
```
$('#chart').append('<svg/>');
   nv.addGraph(function () {
       var chart = nv.models.stackedAreaChart()
               .x(function (d) { return d[0] })
               .y(function (d) { return d[1] })
               .clipEdge(true);
       chart.xAxis
          .tickFormat(function (d) { return d });
       chart.yAxis
          .tickFormat(d3.format(',.2f'));
       d3.select('#chart svg')
           .datum(data)
           .transition().duration(500).call(chart);
       nv.utils.windowResize(chart.update);
       return chart;
   });
```

Listing 5. JavaScript code to create a Stacked Area Graph using Data-Driven Documents.



by Werner Sturm (2012)

Figure 13. Screenshot of prototype displaying an overview of the data using Cumulative Glyphs.



by Werner Sturm (2012)

Figure 14. Screenshot of prototype displaying detailed data using Scaled Data-Driven Spheres. Real time 3D window provided by X3DOM (a). Timeline to navigate through time (b). Possibility to switch between detailed mode (SDDS) and overview mode using Cumulative Glyphs (c). Animated glyphs within specified period for three-dimensional trend analysis (d). 2D plot of values measured by selected measuring points for detailed trend analysis (e).

Moreover, D^3 can handle non-equidistant data points, which are needed to depict approximated data (as discussed in Section V-A1) without interpolating these explicitly. In addition to that, D^3 provides all required graph types (Theme River, Stacked Area Graph, Stacked Cumulative Percent Plot), which allows developers to draw graphs without the need of graph specific implementations (see Listing 5).

VI. Conclusion

This paper investigates two- and three-dimensional visualization techniques to be used to visualize consumed and emitted energies, which have been measured within a vehicle, to support automotive engineers. In addition to that, a web-based prototypal implementation of chosen visualization techniques based on Data-Driven Documents for 2D and X3DOM to perform real-time 3D client-side is introduced.

Besides the measured value and its type of energy, the multivariate data include spatial and temporal components. It revealed that Scaled Data-Driven Spheres are most suitable for a detailed 3D visualization of energy. This contributes to this research field that SDDS are also suitable for other scientific visualizations than medical visualizations. In addition to that, Cumulative Glyphs are presented to provide a better overview of the detailed data.

Both 3D-based techniques only visualize a single moment of the measurement period at once. Despite animations can represent the change over time, engineers want to analyze the trend in a more convenient way. Therefore, two-dimensional visualizations for time-oriented data are investigated in detail. Depending on the engineers requirement, Theme Rivers, Stacked Area Graphs and Stacked Cumulative Percentage Graphs support engineers to get a better understanding of the trend (see Figure 15).

A successful implementation of a prototype shows, that all techniques can be implemented as a web-enabled application without the need of installing third-party browser plugins. It revealed some issues with X3DOM like web browser incompatibilities, but most of them exist because X3DOM does not implement all specified functionalities defined by X3D standard. Despite that fact, web developers benefit from both X3DOM and Data-Driven Documents, which enable them to implement interactive visualization applications without the need to struggle with low-level graphics operations like OpenGL or drawing single lines for 2D graphs. This certainly leads to lower development costs as well.

Acknowledgment

We would like to thank the Austrian *Bundesministerium für Verkehr, Innovation und Technologie (bmvit)* for its generous support within the FEMTech project MueGen. Furthermore, we would like to thank the Institute of Automotive Engineering Graz (FTG), which provided this interesting use case and corresponding measurement data.

References

[1] W. Sturm, R. Berndt, A. Halm, T. Ullrich, E. Eggeling, and D. W. Fellner, “Energy Balance: A web-based visualization of energy for automotive engineering using X3DOM,” Proceeding of the International Conference on Creative Content Technologies (CONTENT), 2013, pp. 1–10.

[2] R. Borgo, J. Kehrer, D. H. S. Chung, E. Maguire, R. S. Laramee, H. Hauser, M. Ward, and M. Chen, “Glyph-based visualization: Foundations, design guidelines, techniques and applications,” M. Sbert and L. Szirmay-Kalos, Eds. Eurographics Association, pp. 39–63.

[3] G. M. Fields and R. G. Metzner, “Hybrid car with electric and heat engine,” 1982, US Patent 4,351,405.

[4] J. Behr, P. Eschler, Y. Jung, and M. Zöllner, “X3DOM: a DOM-based HTML5/X3D integration model,” in Proceedings of the 14th International Conference on 3D Web Technology, 2009, pp. 127–135.

[5] M. Bostock, V. Ogievetsky, and J. Heer, “D^3; data-driven documents,” IEEE Transactions on Visualization and Computer Graphics, 2011, pp. 2301–2309.

[6] S. Havre, B. Hetzler, and L. Nowell, “ThemeRiver: visualizing theme changes over time,” in IEEE Symposium on Information Visualization, 2000, pp. 115–123.

[7] T. Gilligan, Stacked Cumulative Percent Plots. PharmaSUG, 2009.

[8] W. Chan, “A survey on multivariate data visualization,” Technical Report – Department of Computer Science and Engineering, Hong Kong University of Science and Technology, 2006.

[9] W. Aigner, S. Miksch, W. Muller, H. Schumann, and C. Tominski, “Visualizing time-oriented data–A systematic view,” Computers & Graphics, no. 3, 2007, pp. 401–409.

[10] T. Ropinski, S. Oeltze, and B. Preim, “Survey of glyph-based visualization techniques for spatial multivariate medical data,” Computers & Graphics, 2011, pp. 392 – 401.

[11] C. Ware, “Quantitative texton sequences for legible bivariate maps,” IEEE Transactions on Visualization and Computer Graphics, 2009, pp. 1523–1530.

[12] A. A. Bokinsky, Multivariate data visualization with data-driven spots. The University of North Carolina at Chapel Hill, 2003.

[13] D. Feng, Y. Lee, L. Kwock, and R. M. Taylor, “Evaluation of glyph-based multivariate scalar volume visualization techniques,” in Proceedings of the 6th Symposium on Applied Perception in Graphics and Visualization (APGV), 2009, pp. 61–68.

[14] L. Byron and M. Wattenberg, “Stacked graphs, geometry and aesthetics,” IEEE Transactions on Visualization and Computer Graphics, 2008, pp. 1245–1252.

[15] Khronos, “WebGL,” last accessed: June 2014, http://www.khronos.org/webgl/.

[16] K. Sons, F. Klein, D. Rubinstein, S. Byelozyorov, and P. Slusallek, “XML3D: interactive 3D graphics for the web,” in Proceedings of the 15th International Conference on Web 3D Technology, 2010, pp. 175–184.

[17] jQuery Board, “JQuery,” last accessed: June 2014, http://jquery.org/.

[18] T. Ropinski, M. Specht, J. Meyer-Spradow, K. Hinrichs, and B. Preim, “Surface glyphs for visualizing multimodal volume data,” Proceedings of the Vision, Modeling and Visualization Workshop (VMV), 2007, pp. 3–12.

[19] M. Stone, “Choosing colors for data visualization,” last accessed: June 2014, http://www.perceptualedge.com.

[20] C. G. Healey, “Choosing effective colours for data visualization,” Proceedings of Visualization ’96, 1996, pp. 263–270.

[21] H. Levkowitz and G. T. Herman, “Color scales for image data,” IEEE Computer Graphics and Applications, 1992.

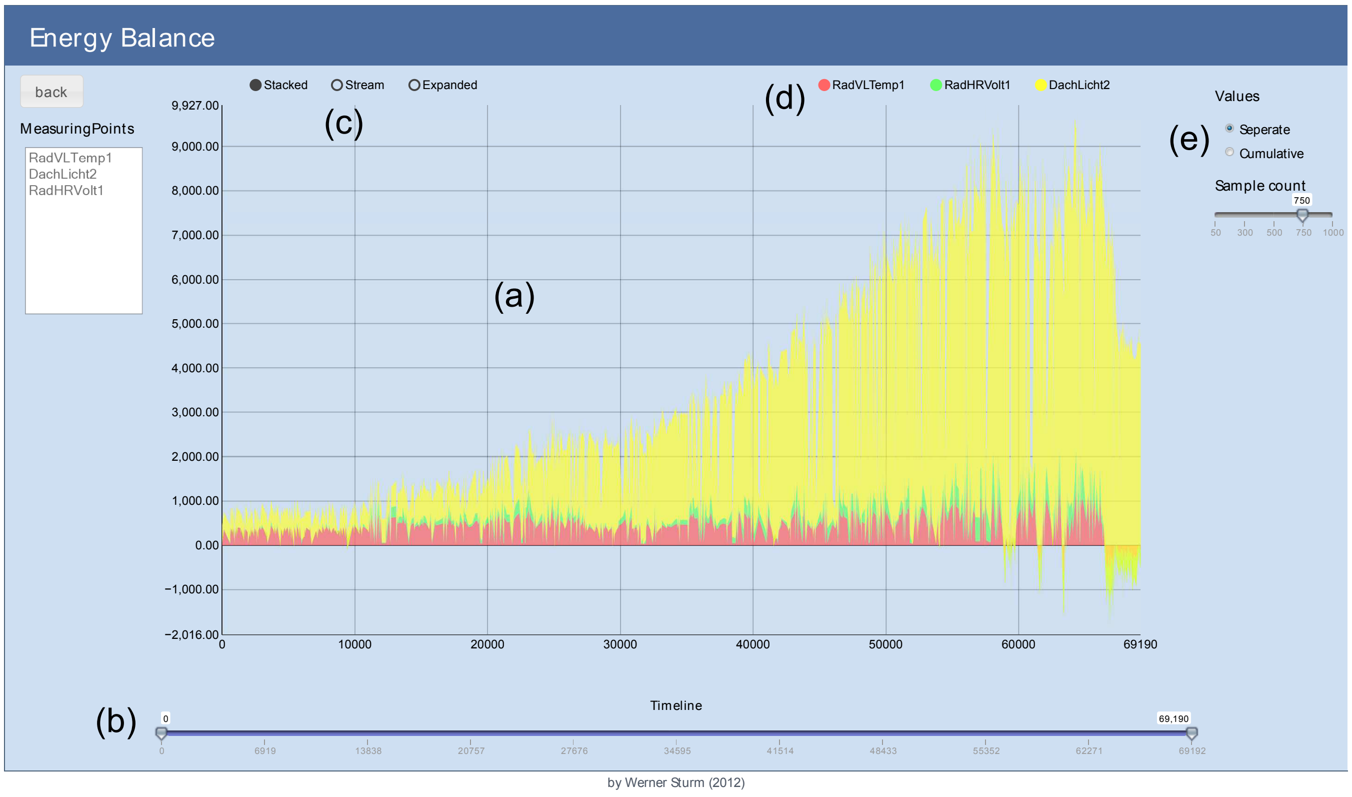
[22] B. E. Rogowitz, L. A. Treinish, and S. Bryson, “How not to lie with visualization,” Computers in Physics, 1996, pp. 268–273.

[23] M. Stone, A field guide to digital color. AK Peters, Ltd., 2003.

[24] D. Borland and R. M. Taylor II, “Rainbow color map (still) considered harmful,” IEEE Computer Graphics and Applications, 2007, pp. 14–17.

[25] D. H. Douglas and T. K. Peucker, “Algorithms for the reduction of the number of points required to represent a digitized line for its caricature,” Cartographica: The International Journal for Geographic Information and Geovisualization, 1973, pp. 112–122.

[26] K. Valkhof, “Grafico javascript charting library,” last accessed: June 2014, http://grafico.kilianvalkhof.com/.



● Stacked ○ Stream ○ Expanded

Values

[✓ Seperate]
Cumulative

Figure 15. Screenshot of prototype showing a 2D representation of the data using Stacked Area Graphs. Graph is created using Data-Driven Documents library. Plot area for displaying the graph (a). Timeline to select a period to plot (b). Option to switch between Stacked Area Graphs (selected), Streamgraph (very similar to ThemeRiver) and Stacked Cumulative Percentage Graph (c). Possibility to select if measuring point is included in current graph (d). Switch between *seperate* and *cumulative* mode (d).