**FED — A Framework for Iterative Data Selection in Exploratory Visualization**

Richard J. Resnick, Matthew O. Ward, and Elke A. Rundensteiner  
Computer Science Department  
Worcester Polytechnic Institute  
Worcester, MA 01609  
(rjr,matt,rundenst)@cs.wpi.edu

**Abstract**

This paper presents a paradigm for the interactive selection (querying) of data from a structured grid of data points for exploratory visualization. The paradigm is based on specifying and iteratively adjusting the Focus, Extent, and Density (FED) of the data attributes. The FED model supports highly complex queries of structured data in an intuitive fashion, and is augmented with a visual interface composed of a set of simple yet powerful user interface controls for query specification. In addition, statistical aggregations are supported by the model. Finally, the FED model is compared to the SQL paradigm, and is shown to be well suited for mapping to a direct-manipulation graphical interface.

**1 Introduction**

Exploratory data visualization is often structured as an iterative process consisting of data selection, data processing, mapping to graphical entities/attributes, and viewing [1, 6, 13]. As the size of data sets to be explored grows, it becomes increasingly necessary to develop more powerful techniques for the selection of subsets to be visualized. This has led to a tighter integration of the visualization process with underlying database technology. However, the mechanisms for performing generic queries of the database (e.g., SQL and its variants) are often not well suited for the types of operations one would like to perform when exploring such complex data sets. Examples of desirable operations include minor variations on previous queries, such as moving the focus of attention or zooming in or out on a particular region of the data, as well as specifying data density as a function of the focus of attention.

In this paper we present a model for iterative data selection which effectively and intuitively supports the visual exploration process. The key components of the proposed model, called **FED**, are Focus, which is the center of attention, Extent, the range of the neighborhood surrounding the Focus, and Density, the number and distribution of data points being selected across the Extent. Keys to the effectiveness of this model are the close interdependencies of these three features, natural mappings to an important class of data exploration queries, simplicity of the model itself, and the clean mapping of the underlying model to a simple direct manipulation front end.

The impetus for our research is a project involving the design of effective visualizations of the output of fire simulations, though the resulting model of data exploration is applicable to many alternate domains. We are developing a system whereby fire and smoke sensors for a specific building are automatically matched against a database of fire simulations executed on that same building. As sensors activate, simulations which most closely match occurring events are retrieved and used to generate visualizations of likely fire futures for use by fire suppression personnel. Each simulation generates megabytes of spatio-temporal multivariate data points, and our goal is to create visualizations which convey as much useful information in a very short period of time as possible. The techniques presented here are to be used by visualization specialists and fire safety engineers to help design visualizations for the end-user system. The former use our proposed **FED** model to extract rapidly different types of data sets and to visualize them, allowing them to compare and contrast many in a short time.

This paper is organized as follows: we begin by reviewing the concepts surrounding structured data, and follow with a presentation of our model for selecting data of this type. We present the model incrementally, and include applicative examples. We discuss the benefits of the model, and some natural extensions to some of the model’s concepts. Next, we demonstrate user interface controls for interacting with the FED model. We then describe other techniques for data selection from the literature and how they relate to our work. Finally, we summarize and present some areas open for future work.
2 Data Model: K-dimensional Data Cubes

The model we present is a framework for selecting structured data. By structured, we mean n-dimensional data sets where k of those n dimensions (k < n) are orthogonal to each other, i.e., they are independent coordinates. We can view this space as a hyperbox of dimension k, commonly called a k-dimensional data cube [15], where the remaining n - k data attributes are uniquely determined by their location in that hyperbox. Informally, we say that the “dependent” data lies at the individual locations in k-space. Formally, we will refer to the orthogonal dimensions as grid fields, and the remaining n - k attributes as data fields.

Note that this data cube definition requires that the k grid fields be ordinal. This order may be a property of the data itself, or it may be user-imposed for the purposes of visualizing a non-ordinal field.

As an example, consider a 5-dimensional data cube consisting of x, y, z, t (for time), and sim (for simulation identifier). Note that these five grid fields are orthogonal to each other, and that a dependent dimension temperature T is uniquely determined by a vector in this 5-dimensional data cube. That is, for any five tuple (x, y, z, t, sim), there exists exactly one temperature value.

We can imagine adding other data fields to this grid; indeed, every quintuplet (x, y, z, t, sim) might have temperature, pressure, molecular content, and other information.

3 \( \mathcal{FE}D \) Model Overview

A database of structured data can be thought of as a k-dimensional hyperbox (where k is the number of grid fields). Often, the visualization process requires that we observe some subset of our hypercube data set. If an entire database can be considered a hypercube with k dimensions, then subsets of that database are simply regions in k-space. The \( \mathcal{FE}D \) model we are about to present enables specifications of regions in k-space. We present the model incrementally, starting with a simplistic version and subsequently adding power and generality.

3.1 The Basic \( \mathcal{FE}D \) Model

The basic model allows for the selection of a single hyperbox in k-space according to three parameters:

Focus (\( \mathcal{F} \)) A key specifier of location in k-space is focus. Focus describes the point in k-space which is the center of attention.

Extent (\( \mathcal{E} \)) The extent specifies the bounding hyperbox (range) of the region of interest.

Density (\( \mathcal{D} \)) Within the given region defined by the extent, density specifies the amount and distribution of data relative to the focus (Figure 1).

\( \mathcal{F} \) can be specified by a single k-vector. For example, in a 3-dimensional data cube modeling “spatial coordinates”, the focus can be specified by a single vector \((x, y, z)\). \( \mathcal{E} \) can likewise be specified by two k-vectors, where the first k-vector specifies the minimum value of each dimension respectively, and the second specifies the maximum value. And finally, \( \mathcal{D} \) can be specified in a variety of ways. Initially, we choose to specify density as a k-vector of real numbers from 0 to 1, where a value of 0 implies no data, and a value of 1 implies data at the resolution at which it exists in the database. One might note that densities greater than 1 could imply interpolation of values across the grid. Also, we assume that density is evenly distributed over space. We will revisit and expand upon these notions in Section 4.3.

As an example, consider a data set with four fields: x, y, z, and T, where x, y, and z indicate position in 3-space, and T indicates temperature at position \((x, y, z)\).

One \( \mathcal{FE}D \) query among possibly several that would retrieve the entire dataset follows:

\[ \mathcal{F}: \left( \frac{1}{2}(x_{\text{max}} - x_{\text{min}}), \frac{1}{2}(y_{\text{max}} - y_{\text{min}}), \frac{1}{2}(z_{\text{max}} - z_{\text{min}}) \right) \]

\[ \mathcal{E}: \langle x_{\text{min}}, y_{\text{min}}, z_{\text{min}} \rangle, \langle x_{\text{max}}, y_{\text{max}}, z_{\text{max}} \rangle \]

\[ \mathcal{D}: (1.0, 1.0, 1.0) \]

This \( \mathcal{FE}D \) specification indicates that our focus is on the exact center of our data set, our maximal bounding box is the entire data set, and we retrieve every value which occurs in the data set (\( \mathcal{D} = 1.0 \)).

We can constrain the selection by modifying the \( \mathcal{FE}D \) specification. Perhaps we are interested only in the temperatures on the xy plane where \( z = 15 \). Then our \( \mathcal{FE}D \) query could be adjusted to the following:

\[ \mathcal{F}: \left( \frac{1}{2}(x_{\text{max}} - x_{\text{min}}), \frac{1}{2}(y_{\text{max}} - y_{\text{min}}), 15 \right) \]

\[ \mathcal{E}: \langle x_{\text{min}}, y_{\text{min}}, 15 \rangle, \langle x_{\text{max}}, y_{\text{max}}, 15 \rangle \]

\[ \mathcal{D}: (1.0, 1.0, 1.0) \]

Note that the z field of the density specification is still 1. When a field is reduced to a point by constraining its extent, any density value less than 1 is meaningless.

Finally, we might be interested in temperature values only at every other point in the grid on our xy plane (perhaps for computational savings or simply to obtain a coarser picture of temperature trends). In this case, our \( \mathcal{FE}D \) specification can be incrementally adjusted by changing the \( \mathcal{D} \) parameter:


- $F: \{(x_{max} - x_{min}), \frac{1}{2}(y_{max} - y_{min}), 15\}$
- $E: ((x_{min}, y_{min}, 15), (x_{max}, y_{max}, 15))$
- $D: (0.5, 0.5, 1.0)$

Density has many applications. Data located at a remote repository can be retrieved at lower densities, allowing faster retrieval at a lower level of detail. Indeed, exploratory visualization often consists of a low resolution broad exploration of data, followed by a high density more detailed view of certain spaces. Section 4.3 explores density in greater detail.

### 3.2 The Grid Field $FED$ Model

The Basic $FED$ Model, which introduced thus far enables us to specify exactly one $k$-dimensional hypercube in $k$-space, will be now be extended. Recall that we are working in a structured grid of $k$ dimensions, where at each point in the grid, we have $n - k$ data fields. We might be interested in only a subset of those $n - k$ data fields. Whereas until now we have described only a single basic $FED$, we now allow each of the $n - k$ data fields to have a $FED$ specification. Thus, we may now specify $n - k$ regions of our $k$-space, where each region corresponds to exactly one of the data fields in our grid (i.e., temperature throughout the entire region, and pressure only on the first two floors of the building).

We now augment our sample data set so that it contains five variables: $x, y, z, temp,$ and $flow,$ where the first four variables are from above, and the fifth variable, $flow,$ is a vector measuring air velocity at point $(x, y, z)$.

The Grid Field $FED$ Model allows us to specify a separate $FED$ for each data field. As in this example we have two data fields ($temp$ and $flow$), we now will show the configuration which applies to the entire dataset in tabular form. As a shorthand notation we now define the midpoint of a grid field $D$ as:

$$D_{mid} = \frac{1}{2}(D_{max} - D_{min})$$

Then, the following $FED$ specification would retrieve the entire dataset:

<table>
<thead>
<tr>
<th>$F$</th>
<th>$E_{min}$</th>
<th>$E_{max}$</th>
<th>$D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>temp</td>
<td>${x_{mid}, y_{mid}, z_{mid}}$</td>
<td>${x_{min}, y_{min}, z_{min}}$</td>
<td>${0.0, 0.0, 0.0}$</td>
</tr>
<tr>
<td>flow</td>
<td>${x_{max}, y_{max}, z_{max}}$</td>
<td>${x_{min}, y_{min}, z_{min}}$</td>
<td>${1.0, 1.0, 1.0}$</td>
</tr>
</tbody>
</table>

Note that we “turn off” temperature simply by setting its density $D$ to 0. Density is a very natural control for specifying data quantity. Just as densities below 1.0 imply less than the total set of values, raising density to values greater than 1.0 implies value interpolation across the cells of the structured grid.

The Grid Field $FED$ Model is at least as powerful as the Basic $FED$ Model, since specifying the $FED$ of each of the $n - k$ data fields to be identical corresponds to a “basic” $FED$ specification. Thus, the Grid Field $FED$ Model is a generalization of the Basic $FED$ model, whereby we allow for the specification of hypercubes and then take the union of these hyperboxes to determine the final region. Since variation in all aspects of $FED$ are permitted, the final selection may indeed be discontinuous regions in $k$-space. (Just as it is possible to create a semantically void SQL query, such a $FED$ specification may not be a semantically intended query. With $FED$, however, the user has the advantage of being able to rapidly and more incrementally adjust the given $FED$ parameters to move from one query to the next.)

### 3.3 The Data Field $FED$ Model

Although the Grid Field $FED$ Model gives us a gain in the nature and the quantity of the regions in $k$-space, it lacks a vital component — the ability to specify a region based on the values of the data fields themselves instead of only on the coordinates of the objects. That is, one might be interested in a region in 3-space where the temperature is between $300^\circ F$ and $430^\circ F$.

We now further augment the model such that each of the $n - k$ data fields not only has a $FED$ specification against the grid fields (as in the Grid Field $FED$ Model), but also is able to specify a $FED$ against itself. We refer to this new type of specification as a data $FED$ and the standard specification of focus, extent, and density against the grid fields as the grid $FED$.

Note then that each data field has exactly one grid $FED$ and exactly one data $FED$ according to the Data Field $FED$ Model. We might specify, for example, that the grid $FED$ for the temperature data field is the entire grid, and the data $FED$ for this field is the set of temperatures between $300^\circ F$ and $450^\circ F$, focusing on $370^\circ F$ and with density 1.0. Intuitively, this specification asks for all temperatures between $300^\circ F$ and $450^\circ F$ throughout the entire data set. By modifying the grid $FED$ we can constrain the region in which to look for temperature values which fall in our desired range.
Whereas in the Grid Field \(FE\D\) Model each of the \(n - k\) regions were combined via the union operator to obtain a final region in \(k\)-space, in the Data Field \(FE\D\) Model we take the intersection of each data \(FE\D\) with its corresponding grid \(FE\D\). This produces \(n - k\) regions which we then combine using the union operator. The pseudocode below illustrates this process:

\[
R \leftarrow \{\}; \\
\text{foreach of the } n - k \text{ data fields} \\
\quad G \leftarrow \text{grid } FE\D \text{ of this data field}; \\
\quad D \leftarrow \text{data } FE\D \text{ of this data field}; \\
\quad R \leftarrow R \cup (G \cap D); \\
\text{final region } \leftarrow R;
\]

### 3.3.1 An Illustrative Data Field \(FE\D\) Example

Consider now a different data set which consists of simulations of stock prices over time. In this case, we have \(n\) dimensions, two of which are the grid fields \(simulationnumber\) \(s\) and \(time\) \(t\). The remaining \(n - k\) dimensions are different stocks at each point \((time_i, sim_j)\) in the grid.

We can now pose the query “Select a sample of the stock prices from simulation 4 in between times 39 and 67 which had values between $45.5/share and $47.8/share”. Recall that in the Data Field \(FE\D\) model, each data field now has a grid \(FE\D\) as well as a data \(FE\D\). The query above clearly asks for a smaller grid (simulation 4 between times 39 and 67) so each data field must have an identical grid \(FE\D\) specification, namely:

- \(F: \{4, \frac{1}{2}(67 - 39)\}\)
- \(E: \{39, 46\}\)
- \(D: \{1, 0, 1\}\)

This grid \(FE\D\) specification reads “Examine all values in simulation 4 between times 39 and 67”. The data \(FE\D\) specification for each data field (i.e., each type of stock) follows:

- \(F: \{\frac{1}{2}(47.8 - 45.5)\}\)
- \(E: \{45.5, 47.8\}\)
- \(D: \{0.25\}\) (returns a subsample)

This data \(FE\D\) specification reads “Select all stocks whose values fall into the range $45.5/share and $47.8/share, focusing on the middle of this range, and returning only a sample of the results.”

The Data Field \(FE\D\) allows us to select relevant data from our database not only based on the physical structure of the region, but also on the semantic content of the region. Clearly this model also enables us to select regions in \(k\)-space which are neither necessarily hyperboxes nor necessarily continuous, given that the data itself now becomes part of the context of the query.

### 3.4 The Multiple Data Field \(FE\D\) Model

Alas, we are still lacking some flexibility in the types of regions we may select by using the Data Field \(FE\D\) Model. Since extents are described by a single beginning point and a single end point, we can only describe continuous ranges across any one dimension in \(k\)-space. We are unable to perform disjunctive queries, i.e., “Select all regions where temperature is between 300°F and 450°F or between 700°F and 850°F.” Certainly, the Data Field \(FE\D\) Model supports queries where only one set of extents applies, however in this query we are interested in two pairs of extents.

We further augment the model to allow each of the \(n - k\) data fields to specify as many grid \(FE\D\)s or data \(FE\D\)s as desired. The algorithm for generating the final selection in \(k\)-space is as follows:

\[
R \leftarrow \{\}; \\
\text{foreach of the } n - k \text{ data fields} \\
\quad G \leftarrow \{\}; \\
\quad D \leftarrow \{\}; \\
\quad \text{foreach grid } FE\D \text{ of this data field} \\
\quad \quad G \leftarrow G \cup g; \\
\quad \text{foreach data } FE\D \text{ of this data field} \\
\quad \quad D \leftarrow D \cup d; \\
\quad R \leftarrow R \cup (G \cap D); \\
\text{final region } \leftarrow R;
\]

We simply take the union of all grid \(FE\D\)s and intersect it with the union of all data \(FE\D\)s. This augmented \(FE\D\) model now allows us to specify an arbitrary number of regions in \(k\)-space, either by structure or by content.

Consider the same data set described in Section 3.3. Whereas formerly we could ask for the stocks whose values fell in a certain range, using the Multiple Data Field \(FE\D\) Model we are now able to ask for stocks whose values fall in more than one range. Furthermore, for each of these ranges, we can specify a focus of interest and a density within that range.

As an example, one might consider observing all time periods where Netscape stocks are either below 30 or above 65. Using the Multiple Data Field \(FE\D\) Model, we can do exactly this by specifying two data field \(FE\D\)s on the Netscape data field, in precisely the same manner as we specified one in the previous section.
4 Discussion

The power of the \( \mathcal{FED} \) model lies in its support of incremental adjustment of a selected set of data — a process which is cumbersome in many other selection methods — through the use of user interface controls which directly change the focus, extent, and density. The incremental selection approach obtained by varying the focus, extent, and density enables users not only to perform exploratory visualization, but also to find their way back from a useless query simply by undoing a previous action.

We now address issues pertaining to the three components of the \( \mathcal{FED} \) model: focus, extent, and density. Specifically, how does altering these three parameters affect the semantics of the query being performed?

4.1 Varying Focus

Focus, as defined above, doesn’t seem to have a natural place in the context of selecting data: it lends itself far more naturally to the notion of visualizing data. We argue, however, that it holds a rightful place in the model, since the model itself is intended to be used for the visualization of structured data. Focus is the center of attention, and may be visually emphasized. We have devised a set of user interface controls which immediately support our notion of focus on a particular point in \( k \)-space. Furthermore, focus could be used to vary density dependent on that focus. In this context, a denser subset of data is retrieved nearer to the focus. (This notion is revisited in Section 4.3).

4.2 Varying Extent

Extent has a more intuitive function in both the database and the visualization contexts. In a purely data-oriented paradigm, the extent of a dataset across a given dimension corresponds to a selection of a range of values on that dimension. In a visualization context, the extent of a dimension is the set of values whose corresponding data fields will be mappable.

Again, the Basic \( \mathcal{FED} \) Model lends itself to our notion of extent. The extents are simply the bounding box (in \( k \)-space) of the region of interest. In 3-space, this is a natural concept — we might have data for an entire building, but our interest in a single room of that building would be clearly designated by the extent settings in 3-space.

Looking at the more general \( \mathcal{FED} \) Models, our notion of extent is augmented by taking the union (one might consider applications where intersections or set subtractions are also relevant) of each of the regions specified by each \( \mathcal{FED} \). This is a reasonable operation both in the database and the visualization contexts. Varying extent amounts to expanding or contracting the context or neighborhood, and might be mapped to a zooming operation in some applications.

4.3 Varying Density

Density is perhaps the most interesting of the three parameters (as it is also the most easily misunderstood).

In the Basic \( \mathcal{FED} \) Model, we speak of the density of the structured grid. If temperature measurements as an example were taken at the center of every cubic foot of a \( 7 \times 6 \times 7 \) room, then we would have 294 data points at density 1.0. If we lowered the density to 0.5 in all grid dimensions, we would essentially be looking at every other point on each axis. That is, we would subsample to create a \( 4 \times 3 \times 4 \) grid, where each cell of the room was 8 times (\( 2^3 \)) larger. If we lowered our density to 0.25, we would be selecting every fourth point on our structured grid, and so on (Figure 1).

As we look at the more general \( \mathcal{FED} \) models, we note a dichotomy in the semantics of density. Whenever we vary the density of a grid field \( \mathcal{FED} \), we see a similar effect as described in the previous paragraphs. What happens, however, when we change the density of a data field \( \mathcal{FED} \)? In this case, we must observe the range and frequency of values in that data field to determine which values to accept and which to reject. If we are interested in selecting points on our grid where the temperature is between 300° F and 450° F, and we allow the density to be 0.5, then we are effectively asking for every other point on the grid where this is true. This operation has a decidedly different semantic implication than subsampling on a grid field.

Value interpolation introduces a similar dichotomy. If we interpolate a grid field by setting its density to a value greater than 1.0, we must then calculate values for each data field at each new point on the grid. A density of 2.0 specifies that data values should be interpolated exactly once between each point in the grid. One can also envision a menu which would enable a user to choose from various types of interpolation (linear, parametric). The maximum density value corresponds to one data point per pixel, and is thus a function of camera location, extents, screen and window size.

Density can also be extended to calculate statistics across a given grid or data field. Rather than setting the density of a \( \mathcal{FED} \) to some real value, we might also allow density to be set to one of a number of statistical choices, i.e., mean, median, quartiles, or best \( n \) (where “best” is some user-definable metric). Functionality such as this is critical for our application, where we are interested in selecting multiple fire simulations which most closely match an actual fire for predicting future behavior.

Finally, one might consider a variable density against a grid or data field which is associated with some polynomial function. As an example, consider a densely populated
database of values on a data cube. Density at the point of focus might be maximal (i.e. 1.0), and then trail off exponentially from that point in all directions. This behavior further emphasizes the focal point(s) of the selection.

5 The $FED$ User Interface Controls

There have existed many models for data selection in the past. One of the primary advantages of the $FED$ Model is its natural mapping to user interface controls in visualization applications. The degree to which a model is accepted depends predominantly on its applicability to real world problems; we propose some intuitive user interface controls here, as well as their effects on the visualization at hand. The problem, then, is to develop a highly intuitive set of user interface controls to represent the sorts of manipulations supported by the $FED$ Model. These controls should enable users to change the $FED$ settings for their given application using direct manipulation (so there is no need to type in queries by hand or by $FED$ specs) and incrementally (so that one $FED$ setting can be easily adjusted if the current query setting does not give the desirable result).

For the Basic $FED$ Model, we have exactly one $FED$ specification to manipulate. In a temporal context, as an example, we have one grid field ($time$), and thus we have exactly one dimension against which we can specify a focus, extents, and density, as in Figure 2. The dimension’s focus is changed by sliding the pointer in the middle of the control. The extent can be adjusted by direct manipulation of the left and right brackets across the dimension. A right-click operation on either extent expands/contracts both extents symmetrically. Density is controlled by the up/down arrows to the right of the user interface control. Density semantics can be selected by using the pop-up menu beneath the control. Note the tick marks on the axis that indicate the current density.

This user interface control provides direct feedback to the user. Direct manipulation of the focus and extents enables a natural intuitive interaction, and when coupled with a resulting visualization will also provide immediate visual feedback. Increasing or decreasing the density of the dimension has the effect of increasing or decreasing the density of the tick marks. An example of this is shown in Figure 3.

In a spatial context, we might have more than one grid field (i.e., $x$, $y$, and $z$). Two approaches can be taken in this case: we can either repetitively use the one-dimensional user interface control we developed for a one-dimensional context; or, we might use a more natural set of interfaces for work in two (Figure 4) or three (Figure 5) dimensions.

For more general $FED$ Models, we must allow for the specification of focus, extent, and density for each data field, indeed perhaps for more than one such specification per data field (as in the Multiple Data Field $FED$ Model). The user interface controls described in the previous paragraph are easily extended to such a task simply by enabling a dynamic allocation of such controls within an application. By clicking on a “new” button to obtain a new $FED$ control and then applying that control to a certain grid field or data field, a complete specification of a data selection is obtained.

6 Comparison to SQL

We now compare $FED$ to SQL query paradigms. Although each model is designed to perform a different task,
Focus Control raises/lowers density.

Minimum Extent and Maximum Extent define the range of the focus.

Density Indicators show the current density setting.

Sample/Interpolate, Mean, Mean + Std. Dev., Median, Quartiles, Best n..., and Worst n... allow for various density calculations.

Figure 2. One-Dimensional User Interface

FED Control — a user interface control to manipulate the focus, extents, and density of a grid or data field.

There are some interesting comparisons to be made. Certainly one possible means of implementing a FED system involves translating FED queries into SQL queries. For this reason, it is important to examine the mappings between operations in the two paradigms.

Our notion of focus is more closely related to visualization than to data selection, and does not have an analog in SQL. Underlying the FED Model we naturally assume some mechanism for data selection; most often focus is applied after that selection and is thus unrelated to the selection process itself. However, certain special types of focus (i.e., varying data density with respect to distance to focus) require selection support which SQL does not have without foreign function extension.

Extent is supported naturally in SQL using range queries, and we believe that the two are interchangeable in a data cube context. Since we manipulate extent against any or all fields of a data cube, we can map each of these settings to a specific range query against a given attribute dimension in a database. Likewise, any range query on a data cube can be mapped to a specification of extent against one or more grid or data fields in our hypercube. The attention of FED on this extent concept offers the advantage of a direct manipulation solution of a (SQL-like) textual query, as in Schneiderman’s range queries [21].

Some applications of our concept of density have analogs in SQL. For example, Oracle has a capability to get “every n-th tuple from a table or selected subset” based on the sequence number assigned to each tuple. FED enables the specification of interpolation and of other forms of density which current implementations of SQL do not support (however it is arguable that SQL could be extended via foreign functions to apply some predicates to decide what tuples to return to the user).

FED is more suitable for exploratory visualization and data mining tasks, as it has a clean direct-manipulation interface with only a few parameters to set to specify a query. It is focussed on a particular class of applications, whereas SQL is a general-purpose query language.

7 Related Work


Focus was formalized by Buja [3], yet can be traced back as far as [1]. An alternative method of providing focus would be to use distortion techniques, such as the Per-
Focus on a "per-axis" basis

Figure 4. Two-Dimensional $\mathcal{FED}$ User Interface Control — The two orthogonal dimensions are appropriately perpendicular to each other. Each dimension maintains its own extents and density. Focus, drawn here as a bullseye icon, can be specified by exactly one location in the two dimensional grid. The grey circles which indicate the projection of the focus on either dimension can also be manipulated to change the focus. The density semantics menus are omitted for space purposes.

Figure 5. Three-Dimensional $\mathcal{FED}$ User Interface Control — A natural extension to the two-dimensional user interface control, this control enables specification in three dimensions. The grey circles for manipulating focus greatly aid in the problem of depth perception in the z axis. Some provisions may be made to allow densities and extents to be symmetrically adjusted across all axes. Note the currently differing densities of each axis.

Index structures for efficiently processing incremental queries as posed by a direct manipulation paradigm such as $\mathcal{FED}$ have recently been proposed for range queries (i.e., extent queries in $\mathcal{FED}$) for n-dimensional data sets by Hibino and Rundensteiner [12]. Many commercial field data visualization packages, such as AVS [24] and IBM Data Explorer [16], support the concepts of data subsetting and resampling, which corresponds to our notions of extents and (uniformly spaced) density.

Schneiderman [19, 20] has perhaps the most comprehensive set of works in human-computer interaction and direct manipulation, and our $\mathcal{FED}$ interface bears some relation to the dynamic query sliders also developed by him [21]. Direct manipulation as used by $\mathcal{FED}$ has been shown to be better than text-based or even forms-based query paradigms in formal user studies [10, 11]. Venolia [25] discusses direct manipulation in three dimensional environments, and Harrison et al. [8, 9] address layered user interfaces for supporting focused and divided attention.

8 Conclusions and Open Areas

This paper has introduced a new model that forms the foundation of an iterative exploratory paradigm for n-dimensional data sets. Four variations of the FED model were demonstrated, each providing more power and generality than the previous. The user interface provides natural tools for the selection of data to support exploratory visualization. Statistical calculations and value interpolation are also useful features supported by the model.

Currently we are implementing an application which uses the $\mathcal{FED}$ Model. We are using Oracle for database
operations, the Visualization Toolkit [8] for visualization routines and algorithms, and Tcl/Tk for user interface components.

Some open areas of research with regards to the FEDE Model are:

- Although we believe that the model naturally lends itself to intuitive user interface controls, it is still required that usability studies be performed to verify this assertion.
- It is not clear whether the model can readily be applied to unstructured data, or to structured data which does not lay in a grid.
- How the FEDE Model changes when grid fields or data fields are non-ordinal needs to be investigate. The notion of focus still applies naturally; how can one extend the ideas of extents and density to fit data which has no inherent order?

9 Acknowledgements

This work was supported by NSF Grant #CMS-9616669.

References


