A Framework for Designing Fisheye Views to Support Multiple Semantic Contexts

Paul Janecek, Pearl Pu
Human Computer Interaction Group, Database Laboratory
Institute of Core Computing Science
Swiss Federal Institute of Technology (EPFL)
1015 Lausanne, Switzerland
(+41)21-693-6656
{ paul.janecek, pearl.pu }@epfl.ch

ABSTRACT

In this paper we discuss the design and use of fisheye view techniques to explore semantic relationships in information. Traditional fisheye and "focus + context" techniques dynamically modify the visual rendering of data in response to the changing interest of the user. "Interesting" information is shown in more detail or visually emphasized, while less relevant information is shown in less detail, de-emphasized, or filtered. These techniques are effective for navigating through large sets of information in a constrained display, and for discovering hidden relationships in a particular representation. An open area of research with these techniques, however, is how to redefine *interest* as a user's tasks and information needs change.

We are developing a framework for implementing fisheye views to support multiple semantic contexts. The framework is based on two components: *Degree Of Interest* functions, and visual *emphasis algorithms* to change the representation of information with respect to interest. The framework supports different contexts through the aggregation of multiple weighted distance metrics in the calculation of interest.

Using this framework, we have developed a user-configurable interface for browsing tabular data that visually emphasizes objects with respect to different semantic contexts.

Categories and Subject Descriptors

H.5.2 **[INFORMATION INTERFACES AND PRESENTATION (e.g., HCI)]** User Interfaces – *Theory and methods, graphical user interfaces (GUI), screen design*; I.3.6 **[COMPUTER GRAPHICS]** Methodology and Techniques – *Interaction techniques.*

General Terms

Algorithms, Design, Human Factors, Theory.

Keywords

information visualization, semantic fisheye views, focus + context techniques, emphasis algorithms.

1. INTRODUCTION

Information visualization tools offer the potential of seeing the relationships and differences within and between collections of information. However, as the size of these collections grow, it is increasingly difficult to represent all of the information in the limited space of a display, and to navigate within the representation at different levels of detail. These problems are even more challenging for small displays, such as in handheld devices. Furthermore, a particular visual representation is most effective for a specific set of tasks, but is often ineffective when the information needs and tasks of the user change [9].

Fisheye (also called "focus + context") views are interactive visualization techniques that address these problems by directly relating the visual emphasis of information to a measure of the user's current interest. These techniques reveal hidden relationships in a representation by visually emphasizing the most relevant objects and de-emphasizing less relevant objects. These techniques also create compact displays of information by showing only the most relevant objects. For example, graphical distortion techniques may be used to increase the detail of objects near the focus and progressively reduce the detail of more distant objects. In a general sense, fisheye views are constructed by pairing a function to measure interest with one or more emphasis techniques.

A Degree of Interest (*DOI*) function is the algorithm used to determine what set of information is the most relevant for the current user task. Simple functions, such as selection and database queries, give only binary or nominal information (e.g., "selected", or "not selected"). More complex functions, such as the "relevance score" of an information retrieval engine, give an ordered distribution of interest that can be interactively refined by the user. The results of these ordinal or quantitative interest functions can often be further analyzed, for example through statistical methods or clustering algorithms, to reveal the structure and distribution of relevant results [11]. Interest functions may be based on different aspects of the data represented in the interface, such as content, structure, history of interaction, and relevance to a specific task.

Given a representation of a collection of information, visual emphasis techniques are used to represent additional dimensions beyond the 2 or 3 spatial dimensions of a particular layout. Many emphasis techniques are also composable, allowing the addition of one or more layers of information to any given representation. Simple visual techniques, such as highlighting and underlining, show only nominal information. For example, in a text editor words selected by the user are highlighted in yellow, while misspelled words are underlined in red. More sophisticated emphasis techniques show ordinal and quantitative information. For example, Seesoft uses a red/blue color scale to show the temporal order in which lines of software were edited [13].

Graphs, charts, tables, maps, and diagrams are all classic examples of visual sense-making tools. Many information visualization tools extend these classic techniques with interaction, multiple views, and insightful layouts. Fisheye views can semantically enrich these tools by adding layers of information or detail in response to the changing interest of the user.

In this paper, we develop this general concept of fisheye views into a framework that can be used by both the designers of interactive visualization interfaces and potentially their users to control and refine the visual and semantic richness of these techniques. This paper is organized as follows: first, we give a brief overview of the wide range of fisheye view techniques; second, we develop a general framework for the construction of these techniques, with the particular goal of building semantic fisheye views; third, we describe a prototype we have developed using this framework; and finally we discuss our conclusions and future research directions.

2. FISHEYE VIEWS

Fisheye views were originally developed as a way of balancing local detail and global context in the interface, based on how humans conceptually structure and manage large collections of information [14]. In his seminal research paper on the subject, Furnas remarked that people tend to recall information with respect to its semantic importance, in greater detail in the conceptual "neighborhood" of the current focus, and only "landmark" information at greater conceptual distances. This general structure is found in many semantic contexts, from the cognitive maps people use to navigate through the physical environment, to their knowledge of organizational structures. The term "fisheye" is an analogy to a wide-angle lens that shows an area around the focus in greater detail, and radially distorts more distant objects to fit in the periphery.

Furnas described fisheye views as:

> "…an example of a more basic strategy for the display of large structures. This basic strategy uses a 'Degree of Interest' (DOI) function which assigns to each point in the structure, a number telling how interested the user is in seeing that point, given the current task. A display of any desired size, *n*, can then be made by simply showing the *n* 'most interesting' points, as indicated by the DOI function [14]."

There are two distinct steps in implementing fisheye views. First, the interest (or semantic relevance) of each object is calculated with respect to a particular focus. Furnas formalized this with the following general equation [14]:

$$DOI(x \mid fp = y) = API(x) - D(x,y)$$

This equation calculates the Degree Of Interest (*DOI*) of each point, *x*, as the difference between the *a priori interest* (*API*) of the point and the distance, *D*, between the point and the current focus, *fp*. Furnas originally applied this equation to abstract data structures, such as hierarchies, structured text, and calendars, but suggested that it could be applied in any domain where *API* and *distance* functions could be defined.

The second step in implementing a fisheye view is the visual rendering of *DOI* in the display. Furnas simply filtered information below an interest threshold to create representations at different levels of detail. More generally, a variety of visual techniques can be used to emphasize and de-emphasize information with respect to *DOI*.

Researchers have developed a wide range of fisheye techniques. Noik developed a taxonomy of fisheye views based on four dimensions [21]:

1. **Priority algorithm**: the function used to calculate DOI for every object in the representation. The priority algorithm may be defined by the system (i.e., the designer), supplied by the user, or a combination of the two.
2. **Emphasis algorithm**: the visual technique used to emphasize objects with respect to their DOI. Noik identified several types of emphasis techniques:
    a. *Implicit*. A visual ordering created by the relative alignment of objects in a 2D representation, and perspective in a 3D representation. For example, placing a node at the extreme top of the screen in 2D, or rotating it to the front in 3D, would implicitly emphasize it.

    b. *Filtered*. Objects below an interest threshold are filtered or moved off the display through a zoom operation.

    c. *Distorted*. Modifying the size, shape and position of objects. The distortion may be based on geometric distance (measured on the representation), or non-geometric distance (calculated from the input model).

    d. *Adorned*. Other graphical attributes of objects, such as color, font, transparency, shading and line thickness.
3. **Number of foci**: the number of foci that may be selected at one time.
4. **Input model**: the data model the view is built on, such as sequences, hierarchies, graphs, or nested graphs.

Fisheye views are primarily applied to information modeled in abstract structures (the input model in Noik's taxonomy). For example, Furnas applied fisheye views to the hierarchical structure of a programming language, where distance was a measure of path distance. It is important to note that the distance functions defined in these models are often not *spatial* distance in a particular visual representation of the information, but rather some notion of *conceptual* or *semantic* distance.

Noik compared over 20 examples of fisheye views using this taxonomy. He noted that few implementations allowed the user to specify the characteristics of the priority algorithm (i.e., the *API* and *distance* functions). This effectively limits the fisheye view's ability to respond to the changing information needs of the user (i.e., the needs not foreseen by the designer). As the users task changes, so does the relative importance of information. For example, Furnas' fisheye and Seesoft both create visualizations of software code. Furnas' fisheye aids the programmer in the task of understanding and navigating through the code's structure (i.e., *DOI* is structural), whereas Seesoft's color scale aids the programmer in analyzing temporal relationships (i.e., *DOI* is temporal).

We are interested in developing fisheye views that are able to adapt their calculation of interest to the changing information needs of the user. We refer to these techniques as Semantic Fisheye Views (*SFEV*) because they are dependent on the semantics of the underlying data rather than spatial distance in a particular representation.

3. A SFEV FRAMEWORK

We are developing a general framework for the design of fisheye views that support user interest in different semantic contexts. The two components of the framework we address in this paper are the *DOI* function and a set of emphasis algorithms.

3.1 Calculating *DOI* in different contexts

Furnas originally calculated *DOI* as the difference of *API* and *distance* functions. These are general functions that are redefined in each fisheye view implementation.

The *API* is the importance of an object independent of any focus. Objects with a high *API* are considered "landmarks" in the information space. The *API* may be either specified (e.g., measured from user studies), or derived algorithmically from properties of the information collection (e.g., calculated from structural metrics). For example, Mukherjea calculated the *API* of nodes in the WWW as a combination of connectedness, in/out degree, access frequency, and depth [20]. *API* may also change over time to reflect user interaction [3,17,26]]

The *distance* function is a measure of the conceptual distance between the focus and an object in the collection. This is the component of *DOI* that responds to changes in the user's focus. We are investigating how to integrate multiple types of distance metrics based on information content, structure, user tasks, and interaction.

Content-based metrics are derived from the values of information attributes (e.g., similarity measures in information retrieval). Structure-based metrics are derived from the structure within and between information (e.g., path distance in graphs and hierarchies). Herman, et al., describe both of these types of metrics in much greater detail [15]. User task-based metrics are derived from an analysis of the relevance of information for completing specific tasks (e.g., the relative order of steps in a process). Interaction-based metrics are derived from the recency and frequency of a user's interaction with information (e.g., the history of a user's navigation in hypertext).

We propose the following general *DOI* function for this framework to calculate interest in a particular context:

$$DOI_{context} = f\,(API, w_i, dist_i)$$

Where *dist* and *w* are one or more distance metrics and their associated weight. We foresee the weight vector, ***w***, as a method for designers, and potentially users, to specify the relative importance of distance functions in different contexts. This contextual weighting factor is the most significant difference between this work and the general *DOI* proposed by Rüger, et al. [26].

3.2 Emphasis Algorithms

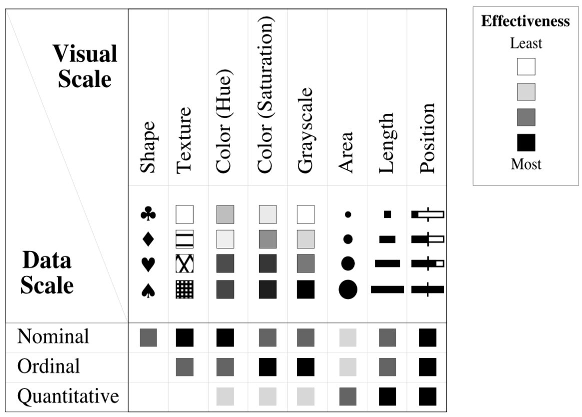
A fundamental goal of visual representations of information is to replace cognitive tasks with perceptual tasks, to allow people to "see" rather than "read" the relationships between information. The distinction between "reading" and "seeing" refers to the two ways in which the human visual system processes information: controlled processing, which, like reading, is "detailed, serial, low capacity, slow, able to be inhibited, [and] conscious"; and automatic processing, which "in contrast is superficial, parallel, … has high capacity, is fast, cannot be inhibited, is independent of load, unconscious, and characterized by targets 'popping out' during search [6]". Controlled processing relies on attentive perception, while automatic processing relies on preattentive perception.

Fisheye views, in general, attempt to use preattentive perception to relate visual importance to *DOI*. In the following two sections, we describe the effectiveness of visual scales for representing different types of information, and develop a general function for emphasis algorithms based on visual scales and transformation functions.

3.2.1 Visual Scales

Jacques Bertin's Sémiologie Graphique was the first systematic study on how to design representations that maximize the use of preattentive perception for revealing relationships in information [5]. Bertin identified and classified a basic set of "visual variables" according to the scale of information that they are most appropriate for encoding (i.e., nominal, ordinal, and quantitative). Table 1 shows a summary of Bertin's classification, modified to include more recent research **[5,7,12,18]**.

Table 1. The relative effectiveness of visual scales for representing nominal, ordinal, and quantitative data.

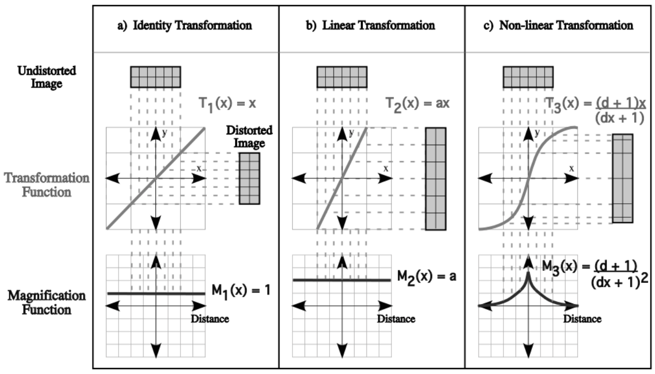


Position is the most effective attribute for encoding all scales of information, hence the particular effectiveness of charts for comparing two variables and maps for representing spatial relationships. Emphasis algorithms use these visual scales, in combination with a transformation function, to emphasize and de-emphasize information with respect to *DOI*.

3.2.2 Visual Transformation Functions

Geometric distortion techniques manipulate object size and position with respect to user interest. These techniques can be characterized by a transformation function that describes the change in position of each point from the original representation to the distorted representation [16,24,27].

Table 2. Three geometric transformation functions, $T_n(x)$, and their corresponding magnification functions, $M_n(x)$.



The three columns of Table 2 show the results of applying different transformation functions to a uniform grid. The top row of each column shows the undistorted grid, the middle row shows the transformation function, and the bottom row shows the corresponding magnification function. The x-axis is the distance of each point from the focus in the undistorted image (i.e., |x| = distance), and the y-axis shows the distance of each point from the focus in the distorted image. The first column shows the identity transformation. The second column shows a linear transformation, where the slope is equal to the magnification. The third column shows the non-linear transformation used by Sarkar & Brown to create graphical fisheye views, where *d* is a factor that increases the amount of distortion [27,28].

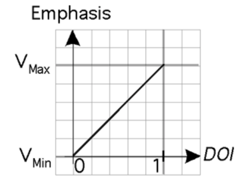
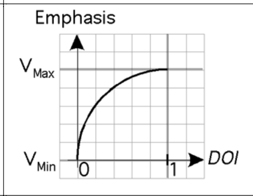
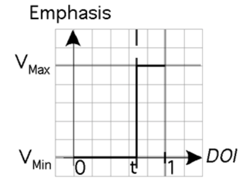
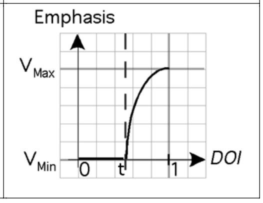
Leung & Apperley developed a taxonomy of distortion techniques based on these spatial transformation and magnification functions [16]. They divided distortion techniques into linear/nonlinear and continuous/non-continuous categories, as shown in Table 3.

Table 3 Several well-known distortion techniques placed into Leung & Apperley's taxonomy.

	Linear	**Non-linear**
Continuous	Pad++ [4]	Graphical Fisheye Views [28], Pliable Surfaces [8]
Non-continuous	Bifocal Display [2], Table Lens [24], Rubber Sheets [29]	Perspective Wall [19]

Non-geometric distortion techniques use similar transformation functions, but they are based on *DOI* rather than distance in a particular representation, as shown in Table 4. For example, the continuous non-linear transformation on the top right in Table 4 is Sarkar & Brown's distortion function (shown previously in Table 2), and the non-continuous linear transformation on the bottom left is the *DOI* function in Rao's Table Lens [24].

Table 4. General transformation functions for semantic fisheye views.

	Linear	**Non-Linear**
Continuous		
Non-continuous		

We propose the following general function to define an emphasis algorithm:

$$EA = f(V, T)$$

An emphasis algorithm, *EA*, is composed of a visual scale, V, and a transformation function, *T*. In this framework, multiple emphasis algorithms may be active in a fisheye view at the same time.

In summary, a fisheye view can be developed using this framework based on two general components: a *DOI* function and one or more emphasis algorithms. The *DOI* function consists of an *API* function and one or more weighted distance functions, and an *emphasis algorithm* consists of a transform function and a visual scale.

4. APPLYING THE FRAMEWORK: A PROTOTYPE

Based on this framework, we have implemented a prototype to explore the interaction between *DOI* and emphasis in different contexts. Our prototype is a tabular interface to a flight itinerary. Tabular representations of information are extremely common (e.g., used in databases, spreadsheets, and many HTML-based sources of information), and easily scaled and navigated in constrained displays. The tasks associated with representations of flight information are well known in the research literature [1,9,23].

The data shown in Figure 1 is from a travel agency itinerary showing a flight departing from Geneva, stopping for several days in Denver, continuing to Oakland/San Francisco (two different airports that serve the same metropolitan region), and returning to Geneva a few days later. The first two columns are the numbers of the flight legs and flight segments, and are not usually shown in an itinerary.

#	Seg #	Date	Airport	Dep/Arr	Time	Airline	Fligh...	Duration
1	1	17.1	GVA	Dep	12:15	SA	138	8h45
1	1	17.1	JFK	Arr	15:00			2h50
1	2	17.1	JFK	Dep	17:50	SA	8178	5h23
1	2	17.1	DEN	Arr	20:27			16h58
2	1	20.1	DEN	Dep	09:15	UA	389	1h40
2	1	20.1	OAK	Arr	10:55			1h40
3	1	23.1	SFO	Dep	21:20	SA	109	11h00
3	1	24.1	ZUR	Arr	17:20			0h50
3	2	24.1	ZUR	Dep	18:10	SA	940	0h45
3	2	24.1	GVA	Arr	18:55			12h35

Figure 1. A flight itinerary displayed in a tabular representation.

Figure 2 shows an Entity-Relationship schema of this itinerary, which reveals the relationships between the data in the table. A *FlightLeg* is the sequence of one or more *Flights* from an initial *Departure Airport* to a final *Arrival Airport*. A *Layover* is the duration of time between two consecutive flights. An *Airport* has a name, a city, and an abbreviation.

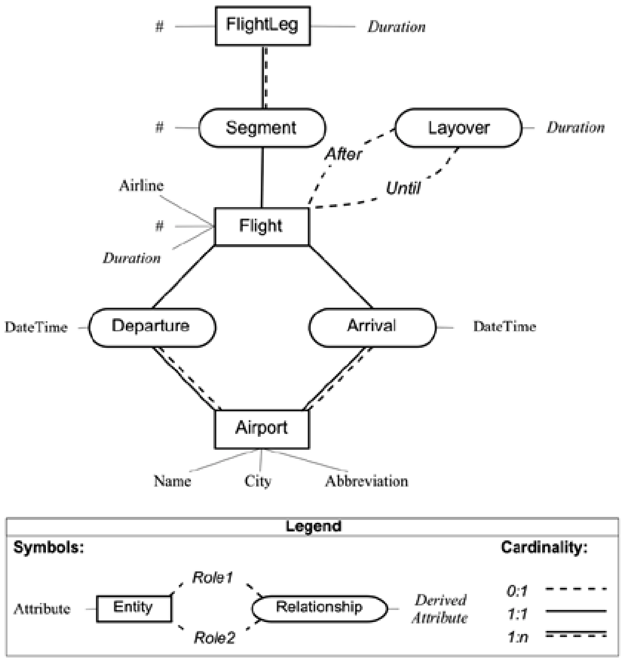


Figure 2. An ER schema of a person's flight itinerary.

Common visual tasks associated with tables are searching for values along a row or column of interest (e.g., to answer the question "What is the departure information for Denver?"), and comparing multiple values across different rows (e.g., to answer the question "Where is my shortest layover?"). The prototype allows the user to browse through the cells of the itinerary, and the fisheye view emphasizes information to show the underlying structure and interrelationships of the cells. The emphasis algorithms guide the viewer's eye to the information that is most relevant to the current focus.

This prototype integrates multiple distance metrics and emphasis algorithms for each column, row, and cell, all of which are directly controllable by the user. The prototype is built around a representation of tabular data, the Java JTable class.

4.1 Calculating *DOI*

In this prototype, the *DOI* for each cell is calculated using the following equation:

$$DOI(p) = API(p) - \sum_{i=1}^{n} w_i dist_i(p, fp)$$

The *API* for the prototype is based on our analysis of the information users require for an overview of a flight itinerary. We placed the information in the itinerary into several levels of interest as shown in Figure 3. At the highest level is the initial departure and final arrival *DateTime* and *Airport* of each *FlightLeg*. At the next level is the *DateTime* and *Airport* of each *Layover*, followed by individual *Flight* information. At the lowest level are the *Segment.# and* duplicate information, such as duplicate dates.

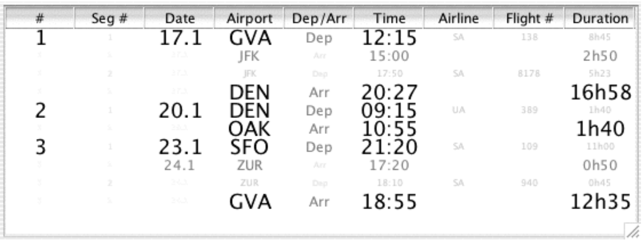


Figure 3. An overview of the flight itinerary created by setting DOI = API and using a font-size emphasis algorithm.

We have implemented two types of distance metrics in the prototype: structure-based and content-based. *Structure-based* metrics calculate distance based on the actual or derived structure of the data, or a semantic model related to the data. The first structural metric is based on the hierarchical *Segment* relationship between a *FlightLeg* and its *Flights* (see Figure 2). In the top two images in Figure 4, the focus moves from *FlightLeg* 3 (distance to both *Segment* 3.1 and 3.2 is equal) to *Segment 3.1* (distance to *Segment* 3.2 is greater).

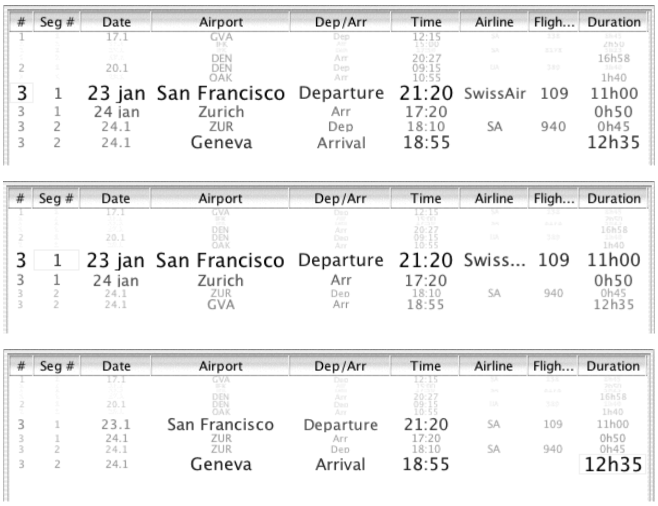


Figure 4. Structure-based metrics. In the top two figures, *dist* is based on the hierarchical path distance between *FlightLegs* (first column) and *Flight Segments* (second column). In the bottom figure, *dist* is based on the *DateTimes* used to calculate *Duration* (last column).

The second type of structural metric implemented in the prototype is based on the dependency relationship between *Durations* and the *DateTime*s they are calculated from. For example, a *Flight.Duration* is calculated as the difference between the *Arrival.DateTime* and the *Departure.DateTime*. In the bottom image of Figure 4, the focus is on the *Duration* of *FlightLeg* 3 (the bottom right cell). The dependency metric contributes a low distance to the related *DateTimes* for the first *Departure* and last *Arrival* of *FlightLeg 3*.

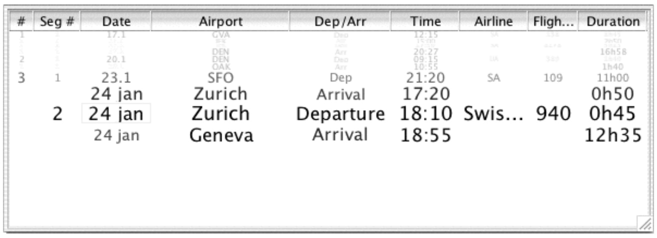


Figure 5. Content-based metrics. When the focus is in the *Date* column, *dist* is calculated on the similarity between dates.

Content-based metrics compare the similarity of the focus value to the values in other cells in the same column (Figure 5). Each column has a different type of data (e.g., date, time, airport), and in this prototype the distance is simply a matching algorithm. More sophisticated strategies could use semantically richer distance models (e.g., the geographic distance between airports).

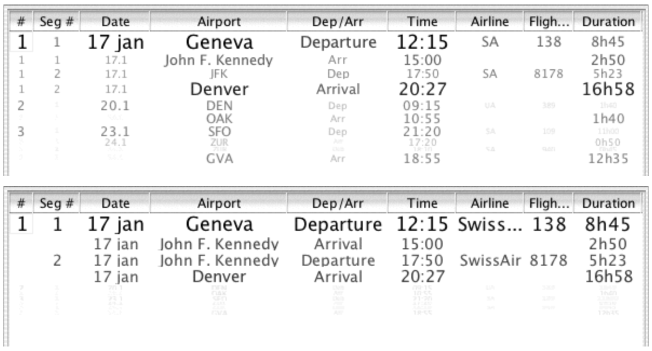


Figure 6. Increasing the weight of a metric visually filters less interesting information.

The user may control the weight of each *distance* metric (as shown in Figure 6) with sliders. Increasing the weight of a metric will result in a wider distribution between *DOI* values, and a stronger emphasis of cells that are more "interesting." The weight of metrics is also affected by the focus. The weight of metrics is also controlled by the current focus. Some metrics, such as the hierarchical structure, are always active. Others, such as the content-based metrics, are only active when the focus is in the applicable column (e.g., the *Date* column for the *Date* metric).

4.2 Emphasis Algorithms

We have implemented five different emphasis algorithms in the prototype: font size, grayscale, row height, semantic zoom, and filtering. The transformation functions associated with these algorithms are shown in Figure 7. The function on the left is Sarkar & Brown's distortion function (described earlier), and is used to transform *DOI* to font size, grayscale and row height. The function is linear when $d=0$, and non-linear when $d>0$. V_{max} and V_{min} depend on the visual scale (e.g., grayscale is a value between black, 0, and white, 255).

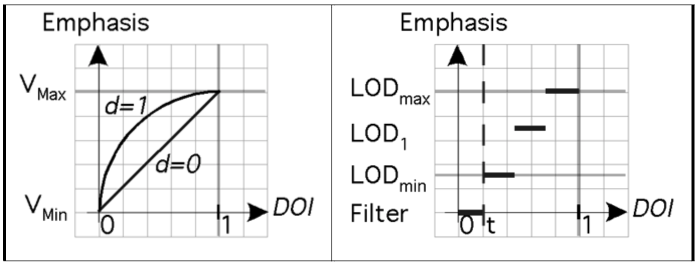


Figure 7. Emphasis transformation functions. The function used for font-size, grayscale and row height is shown on the left, and the function for Level Of Detail (*LOD*) and filtering is shown on the right.

The function on the right of Figure 7 is used for semantic zoom and filtering. We use several levels of detail for the *Date* and *Airport* columns. For example, the date may be represented as "24.1" (*LOD=0)*, or "24 Jan" (*LOD=1).* We also use an interest threshold (*t* in the figure), below which values are filtered.

One or more of these emphasis algorithms may be active at any time, and the user may adjust the distortion, min/max, and thresholds for each algorithm using sliders. For example, Figure 4 and Figure 5 both use font-size, grayscale, row-height, LOD, and filtering.

All transitions between *DOI* states are smoothly animated (e.g., a change in *DOI* from 1 to 10 will transition through the values in between). The emphasis algorithms are based on *DOI*, so the view shows a smooth animation between states as well. We have found this to be essential for the user to maintain their visual position in the representation, especially with techniques such as distortion and semantic zoom.

4.3 Supporting Changes of Context

The prototype supports different notions of context by changing the weight of metrics and the parameters of the emphasis algorithms. This allows the user to change the *DOI* and emphasis of information explicitly with sliders, although the "focus" may be the same (as shown with the structure-based metrics in Figure 4). Alternatively, the user may change the weightings of metrics implicitly by moving the focus (as shown with the content-based metrics in Figure 5).

5. RELATED WORK

This work is strongly influenced by the research of Noik and Rüger, et al. [22,26]. As part of his thesis, Noik implemented a visualization system for querying databases using user-programmable fisheye views and dynamic queries [22]. The environment was based on a view definition language, Dye, which allowed the programmer to flexibly specify the data, the fisheye view *DOI* functions, and emphasis algorithms. The user/programmer could then interactively modify parameters to query, manipulate and analyze the information in the database. The combination of Dye and user-defined similarity algorithms would allow the user to adjust the fisheye view for different contexts. Our research has similar goals, with an emphasis on specifically supporting the notion of context.

The Zoom Navigator framework was developed to support continuous navigation through different levels of both textual and graphical detail within and between graph-based applications [25,26,30]. This framework was used to create an architecture called the Pluggable Zoom, and the ZoomIllustrator and ZoomNavigator applications. The framework we are developing is related to the Zoom Navigation framework in its flexible definition of *DOI*, which allows the framework to be applied in different domains. However, while they were interested in navigating across application spaces (using aspect of interest), we are primarily interested in navigating within several, overlapping semantic spaces defined by user-contexts.

There are also several examples of *FEVs* in the research literature that support the notion of multiple contexts. GeoSpace is a spatial information system based on a spreading-activation network that relates geographic objects to the tasks they support [17]. User goals and tasks are modeled as a hierarchy of plans. Different plans (a task-oriented version of our *distance* metrics) are activated in the network in response to changes in the user's focus, which increases the *DOI* of related objects. The spread and decay of energy in the network also reflects the local history of a user's interaction.

Another example is the Fluid Documents research at Xerox Parc. Fluid Documents are hypertext documents where the linked information uses several different presentation strategies to display information at different levels of detail in the context of the document [10]. They have developed a negotiation architecture that allows objects that have different presentation strategies to negotiate for space in the document. In addition to the context defined by hyperlinks, the negotiation architecture, similar to GeoSpace, reflects the user's interaction history.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we describe a framework for the development of semantic fisheye views. This framework generalizes fisheye and "focus + context" techniques as a combination of a *DOI* function and one or more emphasis algorithms. Our framework extends the traditional notion of *DOI* to support different contexts through multiple, weighted distance metrics. Emphasis algorithms are generally defined as a combination of a visual scale and a transformation function. The goal of this framework is to develop fisheye views that are able to flexibly respond to the changing information needs of users in different semantic contexts.

Based on this framework, we developed a prototype for browsing a tabular display of flight information to investigate the interaction between *DOI*, emphasis, and context. The prototype calculates *DOI* using a combination of structure- and content-based distance metrics. The prototype also uses several different emphasis algorithms based on a composition of visual scales and transformation functions.

Extending fisheye views to support changing contexts is a promising area of research, and still a work in progress. We are currently extending the metrics and emphasis algorithms discussed in this paper to explore much larger and less structured information collections. We are also planning user studies to evaluate the effectiveness of fisheye views in the information seeking process.

7. ACKNOWLEDGMENTS

This research was funded by the Swiss National Science Foundation, under grant number 21-059245.99. We would like to thank the reviewers for their constructive comments, and Sophie Monties for sharing her conceptual modeling expertise with us.

8. REFERENCES

[1] Apperley, M.D., Fletcher, D., Rogers, B. and Thomson, K. (2000) Interactive Visualisation of a Travel Itinerary. In: Proceedings of the Working Conference on Advanced Visual Interfaces, pp. 221-226, ACM Press, New York.

[2] Apperley, M.D., Tzavaras, I. and Spence, R. (1982) A Bifocal Display Technique for Data Presentation. In: Proceedings of Eurographics'82, Conference of the European Association for Computer Graphics., pp. 27-43.

[3] Bartram, L., Ho, A., Dill, J. and Henigman, F. (1995) The Continuous Zoom: A Constrained Fisheye Technique for Viewing and Navigating Large Information Spaces. In: Proceedings of the ACM Symposium on User Interface Software and Technology, pp. 207-215.

[4] Bederson, B.B. and Hollan, J.D. (1994) Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. In: Proceedings of the ACM Symposium on User Interface Software and Technology, pp. 17-26.

[5] Bertin, J. (1967/1999) Sémiologie Graphique, pp. 444, Editions de l'Ecole des Hautes Etudes en Sciences (Les réimpressions), Paris.

[6] Card, S.K. (1996) Visualizing Retrieved Information: A Survey. IEEE Computer Graphics and Applications, 63-67.

[7] Card, S.K., Mackinlay, J.D. and Shneiderman, B. (1999) Readings in information visualization : using vision to think (Morgan Kaufmann series in interactive technologies, Morgan Kaufmann Publishers, San Francisco, Calif.

[8] Carpendale, M.S.T., Cowperthwaite, D.J. and Fracchia, F.D. (1995) 3-Dimensional Pliable Surfaces: For the Effective Presentation of Visual Information. In: Proceedings of the ACM Symposium on User Interface Software and Technology, pp. 217-226.

[9] Casner, S.M. (1991) A Task-Analytic Approach to the Automated Design of Graphic Presentations. ACM Transactions on Graphics 10, 111-151.

[10] Chang, B.-W., Mackinlay, J.D., Zellweger, P.T. and Igarashi, T. (1998) A Negotiation Architecture for Fluid Documents. In: Proceedings of the ACM Symposium on User Interface Software and Technology, pp. 123-132.

[11] Cleveland, W.S. (1993) Visualizing data, pp. 360, Hobart Press, Murray Hill, N.J.

[12] Cleveland, W.S. and McGill, R. (1986) An experiment in graphical perception. Int. J. Man-Machine Studies 25, 491-500.

[13] Eick, S.G., Steffen, J.L. and Sumner Jr., E.E. (1992) Seesoft --- A Tool For Visualizing Line Oriented Software Statistics. In: IEEE Transactions on Software Engineering, pp. 957-68.

[14] Furnas, G.W. (1986) Generalized Fisheye Views. In: Proceedings of ACM CHI'86 Conference on Human Factors in Computing Systems, pp. 16-23.

[15] Herman, I., Melancon, G. and Marshall, M.S. (2000) Graph Visualization and Navigation in Information Visualization: a Survey. IEEE Transactions on Visualization and Computer Graphics 6, 24-43.

[16] Leung, Y.K. and Apperley, M.D. (1994) A Review and Taxonomy of Distortion-Oriented Presentation Techniques. ACM Transactions on Computer-Human Interaction 1, 126-160.

[17] Lokuge, I. and Ishizaki, S. (1995) GeoSpace: An Interactive Visualization System for Exploring Complex Information Spaces. In: Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems, pp. 409-414, ACM.

[18] Mackinlay, J.D. (1986) Automating the Design of Graphical Presentations of Relational Information. ACM Transactions on Graphics 5, 110-141.

[19] Mackinlay, J.D., Robertson, G.G. and Card, S.K. (1991) The Perspective Wall: Detail and Context Smoothly Integrated. In: Proceedings of ACM CHI'91 Conference on Human Factors in Computing Systems, pp. 173-179.

[20] Mukherjea, S. and Hara, Y. (1997) Focus+Context Views of World-Wide Web Nodes. In: Proceedings of the Eighth ACM Conference on Hypertext, pp. 187-196.

[21] Noik, E.G. (1994) A Space of Presentation Emphasis Techniques for Visualizing Graphs. In: GI94, pp. 225-233.

[22] Noik, E.G. (1996) Dynamic Fisheye Views: Combining Dynamic Queries and Mapping with Database Views. In: Department of Computer Science, pp. 142, University of Toronto, Toronto, Ontario, Canada.

[23] Pu, P. and Faltings, B. (2000) Enriching Buyers' Experiences: the SmartClient Approach. In: Proceedings of CHI'2000 Conference on Human Factors in Computing Systems, ACM, The Hague, The Netherlands.

[24] Rao, R. and Card, S.K. (1994) The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus+Context Visualization for Tabular Information. In: Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems, pp. 222.

[25] Ruger, M., Preim, B. and Ritter, A. (1996) Zoom Navigation: Exploring large information and application spaces. In: Workshop on Advanced Visual Interfaces (AVI '96), pp. 40-48, ACM Press, New York.

[26] Ruger, M., Preim, B. and Ritter, A. (1998) Zoom Navigation. In: Computational Visualization: Graphics, Abstraction, and Interactivity (Strothotte, T., ed.), pp. 161-174, Springer-Verlag, Berlin.

[27] Sarkar, M. and Brown, M.H. (1992) Graphical Fisheye Views of Graphs. In: Proceedings of ACM CHI'92 Conference on Human Factors in Computing Systems, pp. 83-91.

[28] Sarkar, M. and Brown, M.H. (1994) Graphical Fisheye Views. Communications of the ACM 37, 73-84.

[29] Sarkar, M., Snibbe, S.S., Tversky, O.J. and Reiss, S.P. (1993) Stretching the Rubber Sheet: A Metaphor for Visualizing Large Layouts on Small Screens. In: Proceedings of the ACM Symposium on User Interface Software and Technology, pp. 81-91.

[30] Strothotte, T., ed. (1998) Computational Visualization: Graphics, Abstraction, and Interactivity, Springer-Verlag.