# DataSplash*

Chris Olston, Allison Woodruff, Alexander Aiken, Michael Chu, Vuk Ercegovac,
Mark Lin, Mybrid Spalding, Michael Stonebraker

Department of Electrical Engineering and Computer Sciences
University of California at Berkeley

## ABSTRACT

Database visualization is an area of growing importance as database systems become larger and more accessible. DataSplash is an easy-to-use, integrated environment for navigating, creating, and querying visual representations of data. We will demonstrate the three main components which make up the DataSplash environment: a navigation system, a direct-manipulation interface for creating and modifying visualizations, and a direct-manipulation visual query system.

## 1. OVERVIEW

Database system performance – as measured by either processing speed or the quantity of data that can be managed – has grown by an order of magnitude in recent years, making increasingly sophisticated applications feasible on ever-larger datasets. However, database query languages have changed relatively little and are difficult for non-experts to use. The vast majority of database users are unable to customize applications to their needs, let alone develop their own custom applications. Thus, at present the expanding capabilities of database systems can be exploited fully only by expert programmers. Making databases easier to use and program, and thereby more accessible, is an important issue today and will become more important as database technology becomes faster, cheaper, and more powerful [6].

We will demonstrate DataSplash, a database visualization environment developed by the Tioga project. DataSplash is an integrated environment for creating, navigating, and querying multiple visual representations of data. Most actions can be performed incrementally via direct-manipulation mouse gestures. Users receive immediate feedback of the effects of incremental modifications. With incremental visual programming, DataSplash makes it much easier for database users to develop database applications.

Other systems support some of these features, but only DataSplash provides an integrated direct-manipulation environment for navigation, visualization, and query manipulation. Pad [4] provides an easy-to-use navigation model, but requires expert users to program the visual representations of the data. Sagebrush [5] has a direct-manipulation interface for creating visualizations, but does not support multiple representations of data or navigation. Magic Lenses [1] supports multiple representations of data but does not provide an interactive way to create new representations. Also, Magic Lenses does not provide an extensive navigation system or a way to visually specify joins. Finally, DEVise [2] supports some direct-manipulation operations on data, but does not integrate direct-manipulation querying and visualization and does not have a sophisticated navigation system.

DataSplash integrates three main components: a navigation system, a paint program interface for creating visualizations, and a direct-manipulation visual query system.

### 1.1. Navigation Model

DataSplash incorporates a sophisticated navigation model. Users can pan, zoom, teleport, and link to other canvases. Objects change representation as users zoom closer to them. DataSplash provides a unique mechanism, a **layer manager**, which allows end users to visually program the way objects behave during zooming.

DataSplash provides visual hyperlinks called **portals**. Portals are sub-areas of the canvas that display other canvases. Users can click on a portal to be transported to the canvas inside. A history of portal navigation is maintained by the system to allow users to return to previous canvases. DataSplash users can automatically generate a portal for every tuple in a database table. Suppose the user has a canvas with a map of the United States. The user can easily specify that each city in the United States map should have a portal that goes to a map of that city. Section 1.3 describes how such portals can be used for visual querying.

### 1.2. Paint Program Interface

As in a conventional paint program, the DataSplash user is presented with a palette of displayable objects (point, line, etc.). To draw an object, the user selects the corresponding paint primitive from the paint palette and places it in a two-dimensional canvas.

Unlike a standard paint program, DataSplash contains a window that shows tuples from a database table to be visualized. The user can draw an object that will serve as a graphical representation of the data. This object is displayed on the canvas once for every tuple in the table. Each copy of the object has properties that are



* Sponsored by NSF under grants IRI-9400773 and IRI-9411334.

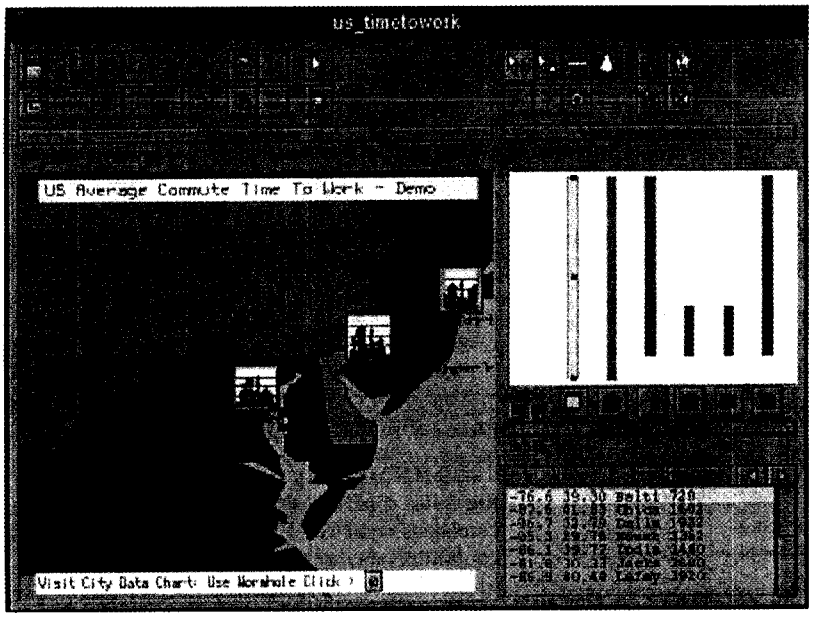


**Figure 1. Navigation**

**DataSplash allows users to navigate by panning, zooming, teleporting, and going through portals to other canvases. The large canvas window shows a close-up view of Maryland, Delaware, and New Jersey. Major cities have portals that go to detailed graphs about the cities. The layer manager (top right) allows users to program the behavior of objects during zooming. The window in the lower right shows tuples from the table being visualized.**

derived from its corresponding tuple via a user-defined function. These properties include the x,y location on the canvas for each copy and visual attributes such as height, width, color, and rotation. For example, a table of United States cities with latitude, longitude, and population columns could be represented as filled circles. X and y could be assigned to the longitude and latitude values of each city, and the radius of the circle could be a function of the population.

## 1.3. Visual Querying

VIQING (Visual Interactive QueryING) [3] is a component of DataSplash that provides users with an interactive visual interface for query specification. VIQING differs from previous graphical query tools in that it supports a "direct-manipulation" approach to querying: users construct queries by manipulating visual representations of entire datasets, as opposed to representations of schemas or example records. The combination of VIQING with the DataSplash architecture results in a seamless, intuitive system in which querying and data browsing are unified into a single metaphor: the direct manipulation of data visualizations.

The VIQING/DataSplash environment supports the three base relational operators: project, select, and join. For brevity, we will only discuss joins. VIQING join queries are specified by a simple drag-and-drop interface. By dragging one DataSplash canvas onto another, the user specifies a VIQING join. Figure 2 shows a DataSplash canvas that visualizes a table of U.S. states. The states are colored according to which political party they have voted for most often in presidential elections between 1952 and 1992 (dark colored for Republican, light colored for Democrat). Figure 3 shows a DataSplash canvas that visualizes a table of presidential candidates. The election year is mapped to the X-axis. The Y-axis represents the result of the election – the winner of each election is on top. The VIQING join in Figure 4 was created by dragging the states canvas (Figure 2) onto the candidates canvas (Figure 3). The four portals in Figure 4 collectively represent the VIQING join query result. For each candidate, a portal containing a subset of the states is displayed. States that voted for a particular candidate appear in that candidate's portal. VIQING queries allow users to visually identify patterns that would otherwise not be obvious. In Figure 4, we can see that every state that has traditionally voted Democrat voted for Clinton in 1992.

Joins of more than two canvases can be specified by dragging the result of a two-level VIQING join onto a third canvas, and so on. Figure 5 illustrates a three-level VIQING join that was created by dragging the two-level join in Figure 4 onto a canvas of political parties (not shown). The result is a three-level VIQING join of parties, candidates, and states (note that the candidates canvas has been panned to the left to show different candidates). This join gives us the party affiliation of the candidates. From this, we can see each party's trends over several election years. Recall that the candidate who won a particular election is on top.

VIQING queries are easier to formulate than SQL for several reasons. First, users don't have to know exactly what they want in advance because VIQING lets them incrementally build and refine queries. At each step, the user gets useful feedback that guides the next query manipulation. In this way, complex queries can be built by combining simpler query pieces. Second, VIQING integrates querying with visualization. Query manipulations are performed on graphical representations of data that are generally easier to understand than text representations. Finally, VIQING eliminates the need to learn any SQL for most query formulation by providing a simple direct-manipulation interface. The VIQING drag and drop join operation requires no understanding of SQL or the database schema to formulate most queries.

# 2. DEMONSTRATION

First, we will demonstrate the navigation model using the commute time data set in Figure 1. Then, starting from a blank canvas, we will paint a new visualization to convey the ease of use allowed by the paint program interface. Finally, we will demonstrate VIQING by using the drag-and-drop interface to execute the VIQING queries represented in Figures 2 through 5.

# 3. REFERENCES

[1] Fishkin, K., and Stone, M., "Enhanced Dynamic Queries via Movable Filters," *SIGCHI 1994*, Denver, Colorado, May 1995, pp. 415-20.

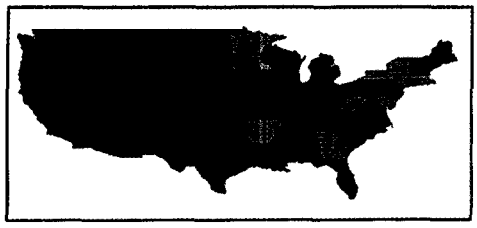

Figure 2. States

This canvas visualizes a table of U.S. states. The states are colored according to the political party they have favored in presidential elections from 1952 to 1992. Dark colored states favor Republicans; light colored states favor Democrats.

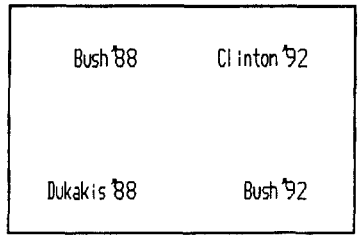


Figure 3. Presidential candidates

This canvas visualizes a table of presidential candidates. More recent election years are shown to the right. (By panning to the left, earlier election years can be seen.) The candidate who won each election is on top.

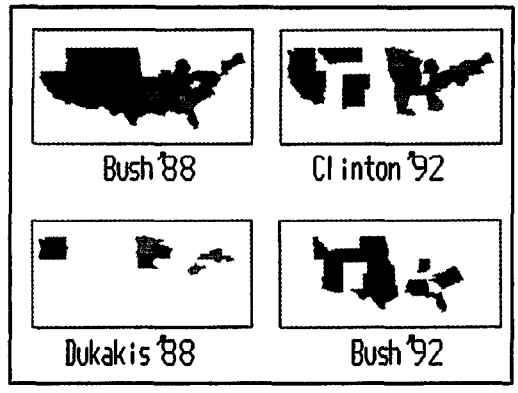


Figure 4. A VIQING join query.

This VIQING query was created by dragging the states canvas in Figure 2 onto the candidates canvas in Figure 3. For each candidate, a portal containing a subset of the states is displayed. States that voted for a particular candidate appear in that candidate's portal. States are colored according to their traditionally favored party (dark colored for Republican, light colored for Democrat). The winner of each election is on top.

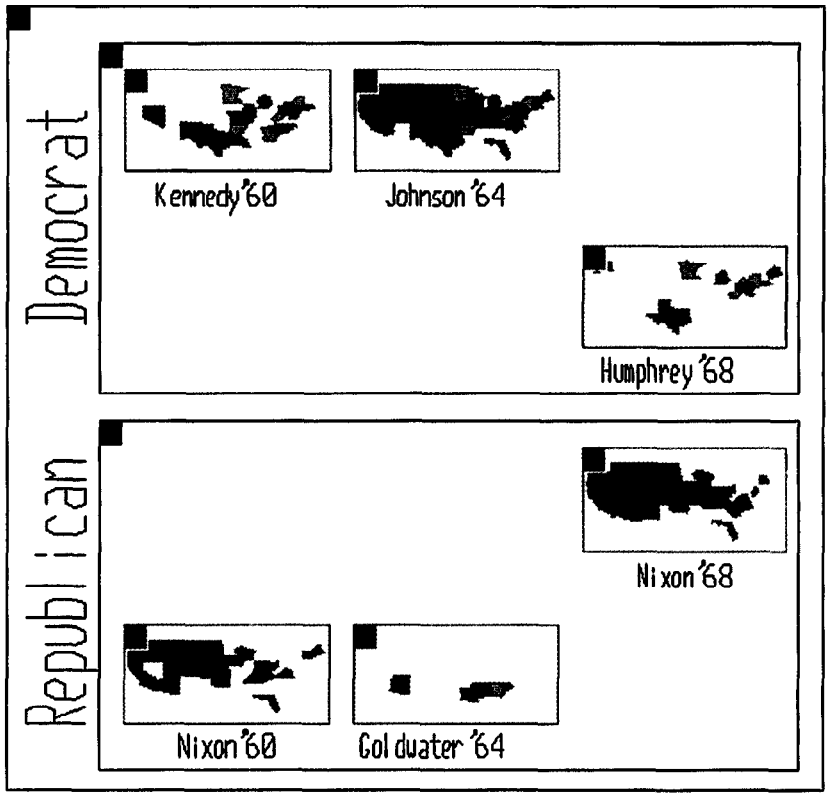


Figure 5. A three-level VIQING join.

This VIQING query is the result of dragging the canvas in Figure 4 onto a canvas of political parties (not shown). The result is a three-level VIQING join of parties, candidates, and states. Note that the candidates canvas has been panned to show candidates who ran for office in the 1960's. As in Figure 4, states are colored according to which party they have voted for most often in presidential elections between 1952 and 1992 (dark colored for Republican, light colored for Democrat).


[2] Livny, M., Ramakrishnan, R., Beyer, K., Chen, G., Donjerkovic, D., Lawande, S., Myllymaki, J. and Wenger, K., "DEVise: Integrated Querying and Visual Exploration of Large Datasets," *SIGMOD 1997*, Tucson, Arizona, May 1997, pp. 301-12.

[3] Olston, C., Stonebraker, M., Aiken, A., "VIQING: Visual Interactive QueryING," submitted for publication, 1998.

[4] Perlin, K., and Fox, D., "Pad: An alternative approach to the computer interface," *SIGGRAPH 1993*, Anaheim, CA, August 1993, pp. 57-64.

[5] Roth, S.F., Kolojejchick, J., Mattis, J., Goldstein, J., "Interactive Graphics Design Using Automatic Presentation Knowledge," *SIGCHI 1994*, Boston, Massachusetts, April 1994, pp. 112-17.

[6] Stonebraker, M., Agrawal, R., Dayal, U., Neuhold, E., Reuter, A., "DBMS Research Crossroads: The Vienna Update," *VLDB 1993*, Dublin, Ireland, August 1993, pp. 688-692.