VISUAL REASONING FOR INFORMATION RETRIEVAL FROM VERY LARGE DATABASES

Shi-Kuo Chang
Department of Computer Science
University of Pittsburgh

ABSTRACT: When the database grows larger and larger, the user no longer knows what is in the database. Nor does the user know clearly what should be retrieved. How to get at the data becomes a central problem for very large databases. We suggest an approach based upon data visualization and visual reasoning. The idea is to transform the data objects and present sample data objects in a visual space. The user can then incrementally formulate the information retrieval request in the visual space. By combining data visualization, visual query, visual examples and visual clues, we hope to come up with better ways for formulating and modifying a user's query. A prototype system using the Visual Language Compiler and the VisualNet is then described.

1. Introduction

The problem domain for this investigation is information retrieval from very large databases. Legal information systems nowadays have databases of up to one trillion bytes. Pictorial information systems for medical, geographical and other applications are dealing with databases of similar order of magnitude, or even larger. Most of these very large databases also grow in size very rapidly.

When the database grows larger and larger, the user no longer knows what is in the database. Nor does the user know clearly what should be retrieved. How to get at the data becomes a central problem for very large databases.

One obvious approach is to let the user try to find out what is in the database. The user must then make several attempts to formulate the retrieval query in such a way that the "relevant" information is retrieved. Information retrieval then becomes an iterative process of formulating queries, observing responses, and reformulating queries.

This approach is costly to the user, who usually pays for each retrieval request and also the connect time to the information system. It is also costly to the information provider, because an ill-formulated query may lead to long processing time and the retrieval of too many database objects which the user really does not care to see.

The approach to be described here is based upon data visualization and visual reasoning. The idea is to transform the data objects and present sample data objects in a visual space. The user can then incrementally formulate the information retrieval request in the visual space. By combining data visualization, visual query, visual examples and visual clues, we hope to come up with better ways for formulating and modifying a user's query.

2. Visual Reasoning for Information Retrieval

2.1. Visual Reasoning

Visual reasoning is the process of reasoning and making inferences, based upon visually presented clues. Visual reasoning is widely used in human-to-human communication. For example, the teacher draws a diagram on the blackboard. Although the diagram is incomplete and imprecise, the students are able to make inferences to fill in the details, and gain an understanding of the concepts presented. Such diagram understanding relies on visual reasoning so that concepts can be communicated. Human also uses gestures to communicate. Again, gestures are imprecise visual clues for the receiving person to interpret.

In human-to-computer communication, a recent trend is for the human to communicate to the computer using visual reasoning. Typically, the human draws a picture, a structured diagram, or a visual example, and the computer interprets the visual expression to understand the user's intention. This has been called visual coaching [SHU89], program by example [MYERS86], or programming by rehearsal [FINZER84] by various researchers.
Since visual reasoning is closely related to visual programming, visual languages and visualization, we will attempt to clarify these related concepts. By visual programming we mean the use of visual expressions (such as graphics, drawings or icons) in the process of programming. By visualization we mean the use of visual representations (such as graphics, images or animation sequences) to illustrate data, program, the structure of a complex system, or the dynamic behavior of a complex system. By visual language we mean the systematic use of visual expressions to convey meaning. Therefore, a visual programming language is a visual language. On the other hand, a language supporting visualizations is also a visual language. Finally, visual programming systems are computer systems which support both visual programming and visualization.

In visual reasoning, we will assume that there is an underlying visual language which may or may not be well-defined. Given the visual clues from this visual language, we can then investigate the visual reasoning process. In what follows, we will describe how visual clues for information retrieval can be presented, and how we can use visual reasoning to determine the information retrieval needs of the user.

2.2. Data Visualization

To enable the user to obtain an intuitive understanding of the contents of the database, we need to first transform the database objects and present them in a visual space, as illustrated in Figure 1. For a very large database, it is obviously impossible to transform every database object. Neither is this desirable, because we cannot present all the data objects on a display screen having a very limited size. Therefore, we must select a few sample objects, which are examples of different clusters of objects. The selection process thus imposes upon the data space a certain taxonomy. Since different users may have different needs, such taxonomy should be modified based upon system’s observation of user’s needs. For example, mathematical clustering techniques can be applied, to classify objects into clusters and select one sample object per cluster. Such objects can then be projected onto the visual space by selecting a few important attributes. In Section 6, we shall present the VisualNet, which facilitates the mapping from data source into the visual space.

2.3. The Visual Space

The Visual Space is illustrated in Figure 2.

![Figure 2. The visual space.](image)

The Visual Space is a multi-dimensional space, where each point represents an object (a record, a tuple, etc.) from the database. A database object, or a visual example, is a point in this space. Conceptually, the entire visual space then corresponds to all the database objects in a database.

Each attribute of a database object represents one dimension in this multi-dimensional space. Therefore, although Figure 2 depicts a seemingly homogeneous visual space, different dimensions actually have different characteristics: continuous, numerical, discrete, or logical.

A query is an arbitrary region in this visual space. A clue is also an arbitrary region in the visual space, but it may contain additional directional information to indicate visual momentum.

Therefore, a visual example is a clue. A visual query is also a clue.

The information retrieval problem is to find the “most meaningful” query, from the visual examples presented by the user. The “most meaningful” query is one which will retrieve the largest number of relevant database objects and whose “size” in the visual space is relatively small. Hopefully, visual reasoning will help us find the most meaningful query from visual examples and visual clues.

3. Operators for Visual Reasoning

It is clear that visual clues are most important in visual reasoning. In the previous section, we said that a clue is some region in the visual space, plus additional directional information. The directional information, intuitively, conveys the “direction” for the search process. We feel this is best expressed by an iconic operator. The iconic operator is a graphical symbol (or an icon) which gives an intuitive meaning to the “direction” of search. Moreover, the “operand(s)” for the iconic operator defines the search region in the visual space. In this section, we define a number of such iconic operators. The sequence of operators selected by the user constitutes a visual sentence, which is also a trace of the user’s reasoning process in formulating the “most meaningful” retrieval request. The collection of such visual sentences forms a visual language.

The principles of visual languages are explored in [CHANG89]. The concept of generalized icons is advanced as a central notion for designing visual languages. A visual expression, or an iconic sentence, can be analyzed with respect to the syntax of the visual language, in a manner analogous to the parsing of a “linear” sentence with respect to a “linear” grammar. In fact, we can carry this analogy one step further, to describe the construction of a visual language compiler. A practical application of the visual language compiler is the customized design of visual user interfaces. In [CHANG89], we gave several examples to illustrate the practical applications of the visual language compiler, including an icon-oriented editor and a visual database interface. Therefore, once we have introduced these iconic operators for visual clues, we can use the visual language compiler to implement the visualization environment for the VisualReasoner (see Section 6). The iconic operators for visual clues are presented below.

In what follows, we will use Q to denote the original query. As stated before, we hope to use visual clues (as defined by the iconic operators) to modify this query Q to a new query which is “more meaningful” in the sense discussed above.

(a) Visual example

As shown in Figure 3, a visual example is a point in the visual space. \( S(e, \text{radius}) \) then denotes the similarity region learned from this example \( e_i \). The new query \( Q' \) is then \( S(e_i, \text{radius}) \cap Q \).

(b) Projection to a subspace

As shown in Figure 4, a projection to a subspace will select a subregion \( Y \) from the visual space \( X \), such that \( Y = \text{PROJ}(X, \text{attributes}) \). The new query \( Q' \) is \( Y \cap Q \).
As shown in Figure 6, the zoom-out operator has the reverse effect of the zoom-in operator. From the restrictive region $X$, the enlarged region $Y$ is found, such that $Y = \text{ZOOMOUT}(X; \text{radius})$.

As shown in Figure 5, the zoom-in operator will select a more restrictive region $Y$ from the visual space $X$, such that $Y = \text{ZOOMIN}(X; \text{constraints})$. The new query $Q'$ is $Y \cap Q$.

The neighbor operator, as shown in Figure 7, finds the neighboring region $Y$ of a given region $X$, or $Y = \text{NEIGHBOR}(X; \text{constraints})$. The new query $Q'$ is $Y \cap Q$.

The bottom region $Y$ of a region $X$ is shown in Figure 8, where $Y = \text{BOTTOM}(X; \text{constraints})$. The bottom region is found by allowing certain attributes to take minimum values. Again, $Y \cap Q$ is the new query $Q'$.

The top of a region $Y$ of a given region $X$ can be defined similar to the previous operator, as shown in Figure 9, where $Y = \text{TOP}(X; \text{constraints})$. Of course, now certain attributes are allowed to take maximum values.

In addition to the operators introduced above, gestural operators can be introduced [HUANG89]. The important point is that the trace or visual sentence can be interpreted by a VisualReasoner for query reformulation.

The new query $Q'$ is naturally $Q \cap Q'$. 

- **Figure 3.** Similarity region $S(e_j)$ of example $e_j$.
- **Figure 4.** Projection of visual space.
- **Figure 5.** Zoom into a smaller region.
- **Figure 6.** Zoom out to a larger space.
- **Figure 7.** The neighboring region.
- **Figure 8.** Bottom region.
- **Figure 9.** Top region.
4. The Visual Reasoner

In the previous section, the iconic operators for expressing visual clues were described. The VisualReasoner should utilize these clues to reformulate user's query. The VisualReasoner is specified by the following procedure.

Procedure main():
begin
 /*input a query*/
 Q = input();
 /*invoke the VisualReasoner*/
 VisualReasoner(Q);
end

VisualReasoner(Q)
begin
 /*The visual query Q specifies the search space*/
 while Q is undesireable
 begin
 if there are clues
 then
 begin
 /*modify query using clues*/
 clues = get_clues(Q);
 Q = modify(Q, clues);
 end
 else
 begin
 /*if no clues then system selects an example*/
 example = select(Q);
 Q = modify(Q, example, rating);
 end
 end
 /*output the samples in modified search space specified by Q*/
 output(Q);
end

5. A Generic Example

A generic example can now be given. In Figure 10, a visual space is shown.

Suppose the user wants to find books concerning steamboats. The user gives an initial example x_0. The VisualReasoner generates an example x_1. Since x_1 is not about steamboats, the user gives it a low rating. Another example x_2 is generated, which receives a good rating from the user. A third example x_3 is generated, which also receives a good rating. The VisualReasoner then zooms into a subspace Q, which contains only books about steamboats.

Therefore, based upon visual examples and clues, the VisualReasoner can zoom into a subspace, or a smaller compartment of the visual query Q. By transforming or distorting the similarity measure, the VisualReasoner can also warp the space to give a different visualization.
6. A Prototype System using VisualNet

A prototype system to realize the approach proposed in earlier sections has been developed at the Visual Computing Laboratory of University of Pittsburgh. In the prototype system, the experimental database is a small subset of the PITTCAT database (University of Pittsburgh's Library Catalog Database). The prototype VisualReasoner is built on top of the Visual Language Compiler [CHANG89], which is a tool for the specification and design of icon-oriented systems. Since SIL-ICON allows the user to specify complex icons, we can easily realize the iconic operators described in Section 3.

The Visual Language Compiler allows the incorporation of user-supplied semantic routines. Thus, the user can input a visual sentence, and the SIL-ICON Compiler will translate it into the corresponding database query. Since we use the ORACLE dbms, the translated query is in SQL syntax. This is illustrated in Figure 11. The visual syntax is such that vertically combined icons denote a conjunction (a logical "AND"), and horizontally combined icons denote a disjunction (a logical "OR"). The icon "DB" denotes the database to be searched. All these object icons can be assigned attribute values.

In addition to these object icons, we also have iconic operators, such as ZOOMIN, ZOOMOUT, TOP, BOTTOM, as shown in the menu at the bottom of the window. The user can select an iconic operator and apply it to the current visual query Q. This leads to a modified visual query Q', as illustrated in Figure 12. The visual query Q, as suggested earlier, specifies a visual space. The modified visual query Q' specifies a modified visual space. Therefore, both the visual space and the resultant query should be modified. This is accomplished by the procedure "modify" in the VisualReasoner (see Section 4). The modification of the visual space leads to updating of the knowledge-base (to be explained). The modified ORACLE query is illustrated in Figure 12.

We will now explain how the zoom-in (and zoom-out) is actually accomplished and how the knowledge-base is maintained and modified, by introducing the notion of the VisualNet.

In Section 2.1, we defined visualization to be the use of visual representations (such as graphics, images or animation sequences) to illustrate data, program, the structure of a complex system, or the dynamic behavior of a complex system. The VisualNet is the complex of data structure and algorithm, to support the mapping from the data source to the visual space. Therefore, we should modify Figure 1 as follows.

An example VisualNet is illustrated in Figure 14. It shows how objects are associated. From the object "life", we can zoom-in to objects "animal" and "plant". From the object "tiger", we can zoom-out to "animal".

Figure 11. A visual query Q.

SELECT TITLE, ISSN FROM journal WHERE AUTHOR = 'William' AND SUBJECTS like 'XShlpSl';

Figure 12. A modified visual query Q'.

SELECT TITLE, ISSN FROM journal WHERE AUTHOR = 'William' AND SUBJECTS like 'XShlpSl';

Figure 13. Information retrieval by visual net and visual reasoning.

Figure 14. An example VisualNet.
Initially, the VisualNet is constructed from an input specification. In the prototype system, the following specification is used:

```plaintext
SUB life
IN animal plant
END

SUB animal
IN fish tiger dog
OUT animal
END

SUB plant
IN flower tree
OUT plant
END

SUB fish
OUT animal
END

SUB tiger
OUT animal
END

SUB dog
OUT animal
END

SUB flower
OUT plant
END

SUB tree
OUT tree
END
```

Although we use a hierarchical structure to illustrate the VisualNet, it should be clear that the VisualNet need not be a tree. Moreover, arcs in the VisualNet have arc strengths indicating the strength of the association.

When the user performs a zoom-in operation, the VisualNet is searched, and a pop-up menu appears on the screen, showing all possible zoom-in alternatives. For example, from “plant”, one can zoom-in to “flower”, “tree”, etc. The user can select one, which causes the VisualReasoner to modify the query appropriately. Moreover, the selected path is reinforced by incrementing the arc strength of the VisualNet. The user also has the option of inserting new entries into the pop-up menu, which causes the VisualReasoner to modify the VisualNet by adding new branches. In the prototype system, the procedure “modify” will modify both the query Q and the VisualNet. The VisualNet thus is the knowledge-base with learning abilities. User’s visual reasoning process is learned and remembered by the VisualNet. The VisualNet can also be asked to select an example. In which case, the path with the largest strength can be selected, so that an appropriate query can be formulated automatically to pick typical examples.

User’s rating will then strengthen or weaken links in the VisualNet.

### 7. Summary and Discussion

In this paper, we describe an approach for information retrieval from very large databases, based upon visual reasoning and visualization. A prototype system is then presented. The main features of this prototype is that (a) it is built on top of the Visual Language Compiler and therefore supports iconic operators; (b) it incorporates a VisualNet, which is a knowledge-base to store the knowledge learned by the system in its interaction with the user.

In the prototype system, the VisualNet is represented by an association network. Other representations, such as neural nets, fuzzy sets, fuzzy neural nets, etc. can also be employed and should be investigated. The main point is that the VisualNet is invisible to the user, but it serves as the visualization mechanism to perform the mapping from the data source to the visual space. The proposed approach is a framework, allowing further investigations into many interesting topics. We list a few of them here: (a) selection of representations for VisualNet; (b) learning algorithms for VisualNet; (c) query modification using similarity measures; (d) transformation and distortion of visual space; (e) natural language clues vs. visual clues.

### References:


### Acknowledgements:

This research was supported in part by NSF grant IRI-8617802. The prototype system and the VisualNet were developed by Chihuang Yen. I am indebted to Sam DeFazio for emphasizing the importance of the mapping from data source to the visual space.