Getting Computers to See Information Graphics so Users Do Not Have to

Daniel Chester[1] and Stephanie Elzer[2]

[1] University of Delaware, Newark DE 19716, USA,
chester@cis.udel.edu,
WWW home page: http://cis.udel.edu/~chester
[2] Dept. of Computer Science, Millersville University, Millersville PA 17551, USA,
elzer@cs.millersville.edu

Abstract. Information graphics such as bar, line and pie charts appear frequently in electronic media and often contain information that is not found elsewhere in documents. Unfortunately, sight-impaired users have difficulty accessing and assimilating information graphics. Our goal is an interactive natural language system that provides effective access to information graphics for sight-impaired individuals. This paper describes how image processing has been applied to transform an information graphic into an XML representation that captures all aspects of the graphic that might be relevant to extracting knowledge from it. It discusses the problems that were encountered in analyzing and categorizing components of the graphic, and the algorithms and heuristics that were successfully applied. The resulting XML representation serves as input to an evidential reasoning component that hypothesizes the message that the graphic was intended to convey.

1 Introduction

Information graphics (line charts, bar charts, etc.) frequently occur in popular media such as newspapers, magazines, and newsletters. Moreover, they generally contain information that is not part of any accompanying text. Although most people easily acquire knowledge from information graphics, this is not the case for individuals with sight impairments. Several research projects have investigated devices for conveying graphics in an alternative medium, such as tactile images or soundscapes[1, 2], but these approaches have serious limitations. For example, generation of a tactile image requires specialized equipment that is expensive. Soundscapes are ineffective at conveying intersection points of multiple lines. In order to avoid disenfranchising a segment of our population, methods must be developed that enable sight-impaired users to effectively access and assimilate information graphics.

The primary goal of our project is a methodology that enables sight-impaired individuals to assimilate the content of an information graphic with relative ease. While previous research has concentrated on rendering graphical elements in an alternative medium, our goal is to provide the user with the knowledge that

one would gain from actually viewing the graphic rather than enabling the user to access what the graphic "looks like." Thus we are developing an interactive system that hypothesizes the intended message of the information graphic, uses spoken language to convey this message along with other related significant features of the graphic, and responds to follow-up questions about more detailed aspects of the graphic.

Section 2 discusses related work and describes potential applications of our system. Section 3 presents the overall architecture of our system. Section 4 focuses on the initial processing of the graphical image to construct an XML representation of the components of the graphic and their relation to one another. It discusses the goals of the image processing component, the problems that were encountered in analyzing and categorizing components of the graphic, and the algorithms and heuristics that were successfully applied. Section 5 describes the current status of our implementation and discusses future work that will extend the kind of graphics that the system can handle.

2 Overview

Limited attention has been given to summarizing information graphics. Reiter[3] used pattern recognition techniques to summarize interesting features of automatically generated graphs of time-series data from a gas turbine engine; however, Reiter started with the underlying data, not the graphical image. St. Amant[4] developed a system, VisMap, for manipulating a visual display that allowed interaction through a graphical user interface. VisMap could take pixel-level input from a screen display and recognize the interface objects displayed on the screen, such as menus and buttons. Futrelle [5–8] developed a constraint grammar to define components of diagrams and parse the diagram. However, Futrelle was interested in a sophisticated mechanism for parsing complex vector diagrams (including finite state automata and gene diagrams); our work is limited to information graphics such as bar and line charts, and thus we can focus on simpler mechanisms that will have high success and be efficient on our real-world problem.

There are several useful applications for a system that can summarize information graphics at different levels of granularity. First, as the primary motivation for our work, it will facilitate an interactive natural language system that can provide the user with the primary content of the graphic (its intended message along with significant related features) and then respond to follow-up questions that address more detailed aspects of the graphic. For digital libraries, the initial summary of the graphic will be used in conjunction with summaries of the document text to provide a more complete representation of the content of the document for searching and indexing. In the case of environments with low-bandwidth transmission and miniature viewing facilities, such as cellular telephones for accessing the web, the initial summary and follow-up capability will provide an alternative modality for access to the document.

3 Architecture

Our current work is concerned with simple bar, line and pie charts, although eventually we will handle other kinds of graphics. The visual extraction module (VEM) analyzes the graphic and provides an XML representation of the graphic to the intention recognition module (IRM). The IRM is responsible for recognizing the intended message of the information graphic and sending it to the content planning module (CPM), which will augment the intended message of the graphic with related interesting features. The message organization module (MOM) then organizes the most salient propositions into a coherent summary, which will be rendered in natural language and conveyed to the user via speech synthesis. The follow-up question module (FQM) will provide the user with the opportunity to interactively seek additional information about the graphic. Details and status of the overall system may be found elsewhere [9–12].

The focus of this paper is on the processing done by the visual extraction module, which converts a graphic from its image format to a text format more suitable for the other modules to process. First the raw drawing components that are extracted from the image are described. Next, how the components are simplified and grouped to obtain the graphic components, and how these are used to identify the information graphic as a bar, line or pie chart is explained. Then the construction of an XML representation of the information content implicit in the graphic components is discussed. Finally several unresolved issues are addressed.

The information graphics that VEM handles are bar, line and pie charts that have been created by spreadsheet or drawing software. The graphics have been saved in an uncompressed image format and converted to pgm format, making them easy to read into internal arrays of pixels. The images are presently assumed to be one of the three chart types, with no textured regions, i.e., all areas are uniformly filled with one gray level. They are also free of noise and are not blurry, and components do not overlap. A typical example of such an input is the image shown in Fig. 1. One should note that although the graphic in this figure contains a useful caption, our analysis of a corpus of information graphics confirms research by Corio[13] that showed captions to be often nonexistent or unhelpful. Thus the graphic itself must be analyzed to convey its intent to the user.

4 Visual Extraction

When an information graphic is designed, it is built from abstract components like axes, bars, labels, titles, data points connected by lines, and wedges, depending on the graphic type. An image, however, is, at its simplest, just an array of gray level values. The task of the VEM is to extract the graphic components from the pixel array and present them in a textual form. The abstract components of an information graphic are themselves composed of entities, however, consisting of text, line segments, arcs, curves and filled regions in combination.

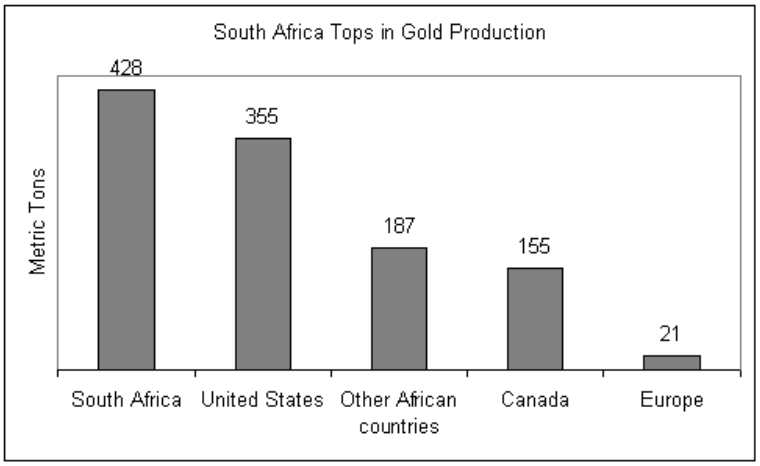

Fig. 1. A sample information graphic image

For example, a bar in a bar chart typically has a filled rectangular region, a border of a different color, and labels that identify the attribute-value pair that the bar represents. The strategy of the VEM is to find these raw components and then to combine them into the graphic components at the level of abstraction that the designer was working with. Once these graphic components, together with their relationships to each other, have been found, it is a simple matter to describe them in a textual format.

4.1 Raw Component Extraction

As Fig. 1 shows, the images typically produced by spreadsheets have a small number of distinct gray level values, and they are usually well-spaced in the range of possible values. This makes it easy to separate an image into connected regions where each region has a uniform gray level. The first step in processing an image is identifying these connected uniform regions which we call *plateaus*. Examination of a histogram of the gray levels in the image with special attention to pixels near the edges identifies the background color, which is white in this example. In addition, the border of each plateau is computed by finding all the points in the plateau that are not completely surrounded by pixels with the same gray level.

Once the plateaus have been extracted, they have to be classified as text, or as chart elements that have to be broken down further into their drawing primitives. The text in a graphic consists of isolated plateaus, each of which is a character (or sometimes two characters run together). To identify these plateaus, we apply the heuristic that textual elements typically occupy small bounding boxes and take up a small percentage of space in those boxes or are short, thin horizontal

or vertical line segments, which could be characters "I", "l" or the main part of "i", depending on the font. We identify these textual elements with a rather simple optical character recognition algorithm; much work has been done on the OCR problem, so we anticipate that one of the commercial OCR programs can be adapted to our needs in a future version of our software, but we have put that effort off because identifying the characters in an information graphic is only a small part of what has to be done. Instead, we use a simple template matching algorithm so that we can concentrate on sketching out the other steps needed for identifying the components of an information graphic image.

The smallest meaningful textual elements, however, are words, so we must group the text plateaus into word groups. We tried the irregular pyramid approach of Tan and Loo[14] to address this problem, but it was slow and ineffective. We devised a heuristic approach based on an analysis of information graphics in popular media. We create a new image in which the bounding boxes surrounding the text plateaus are filled rectangles, all having the same gray level. These rectangular plateaus are then dilated, which merges those that are close together. The modified plateaus are extracted from this modified image and the characters that fit inside of each modified plateau are grouped together as a word.

The highest level of meaningful text in an information graphic is the phrase level, where the phrase serves as a title or label of some nontextual element such as a bar or pie wedge. We extract titles with a simple heuristic: The words that are topmost with a horizontal orientation are sorted from left to right to form the title of the graphic and words that are leftmost with a vertical orientation are sorted from bottom to top to form the title for the Y-axis of the graphic. To facilitate extraction of visual components of a graphic, we make a new copy of the plateau list with all the text plateaus removed.

While the text plateaus can be treated as the textual primitives, the chart element plateaus in a graphic are actually compound entities. Since the shape information of a chart element plateau is entirely in its border, chart element plateaus that are not already lines are replaced by their borders for the rest of the extraction process. The graphic can now be thought of as a line drawing with an overlay of texts. But the chart element plateaus, which are now lines of various sorts, are still compound entities; they have to be broken down into simple drawing primitives like straight line segments and arcs before we can recognize what the plateaus are. In Fig. 1, for example, the X and Y axes are composed of two long lines and many smaller lines which serve as tick marks. We need to break these plateaus into straight line segments before we can recognize what parts of the X axis plateau are the tick marks and what part is the X axis itself. One way to break a compound line into straight line segments is to vectorize it, but the vectorizing programs available on the World Wide Web were not satisfactory to our purpose, so we devised our own way of decomposing the chart element plateaus. Each chart element plateau is traced in a clockwise direction and is broken into chains of pixels that are straight line segments or curves. They terminate when certain changes in the trend of the line is observed and when

a junction is encountered. This breaks up most borders into small pieces. In Fig. 1, the borders that include the X and Y axes are broken up into the Y axis, the tick marks, and the segments of the X axis that lie between the tick marks. Straight lines are fitted to all chains using least squares. Connected sequences of short straight lines that slowly change orientation from one line to the next are recognized as curves. The end result is a representation of the line drawing part of the graphic as a set of straight line segments and simple arcs, which when drawn would reproduce the graphic minus the texts and without closed regions being filled in with gray levels other than the background color. Now that the graphic is decomposed into texts and line primitives that accurately capture all the information in a graphic (because they reproduce the graphic when drawn), the graphic is ready for the recognition phase of the visual extraction.

4.2 Component Regrouping

The line primitives have to be regrouped into recognizable components of bar, line and pie charts. Both bar charts and line charts are expected to have an X axis and a Y axis. Bar charts are expected to have bars that are laid out either vertically or horizontally. Line graphs are expected to have irregular sequences of straight line segments connecting the data points. Pie charts are expected to have wedge-shaped regions that belong to a circular region. There may be other graphical elements present also. For example, grid lines are often present to help the viewer read data off the graphic. Our task is to look for these components of information graphics and see how they are compositions of the line primitives that have been found.

The most perspicuous features of bar and line charts, apart from the bars and data lines, are the X and Y axes. We expect these to be a long horizontal and a long vertical line, respectively. To find these features, we start at the lower left corner of the graphic and look for the nearest vertical and horizontal line segments. These line segments are extended when possible with other line segments that are parallel to and in line with them. Extension is necessary because axes with tick marks get segmented as explained above. If the line segments grow into a line that is long enough, say, about four tenths of the width or height of the graphic, they are grouped together as one line and labeled the X axis or Y axis, depending on their orientation. Small line segments that are next to but perpendicular to one of these lines are grouped with that line as its tick marks. Finding the axes is helpful because they generally mark two sides of the rectangular region where the graphic's data is displayed. In a similar manner, other horizontal and vertical lines that are the same length as the axes are found and classified as grid lines. It is important to get rid of these lines so that the data region is left uncluttered, containing only informative graphic components.

The other graphic elements we look for are bars, data lines, wedges and legends. Small horizontal and vertical line segments have to be grouped when possible into rectangles. Sometimes several line segments have to be grouped together to make one side of a rectangle because there are junction points on

the side when other rectangles are adjacent. Furthermore, some line segments may be part of more than one rectangle, again when rectangles are adjacent, such as when bars are drawn touching each other. If a rectangle is found that contains other rectangles with text near them, that cluster of objects is made into a single component and classified as a legend. The remaining rectangles are expected to be bars in a bar graph.

To recognize the wedges in a pie chart, we look for one straight line segment that has a curve connecting its two ends (a wedge that is half a pie) or two straight line segments that touch at one end and are connected to each other at the other end by a curve.

The data lines in line charts are the least structured of all the graphic components we look for, since they are just a sequence of connected line segments that can wander in any direction across the graphic. We recognize them by looking for connected line segments among the line segments that have not already been found to be parts of an axis, tick marks, parts of rectangles, or parts of wedges.

Finally, after axes, tick marks, rectangles, legends, wedges and data lines have been found, we are able to categorize an information graphic at to type; if there are rectangles that are not legends, the graphic is a bar chart; if it contains one or more data lines, it is a line chart; if it contains one or more wedges, it is a pie chart.

4.3 XML Formatted Output

With the information graphic dissected into its meaningful visual and textual components, we finally have to generate an XML representation of the graphic. Once we know what type of chart we are dealing with, it is straightforward to describe the chart. There are just a few additional pieces of information that have to be extracted from the relative locations of the visual and textual components.

In the case of bar charts, each bar has to be described in terms of its essential properties. These are the color (gray level) of the bar, its length (if horizontal) or height (if vertical) in centimeters, the text near its axis end, and any annotation text at its other end. If the bar has no annotation text, the value represented by its height must be computed. The ticks associated with the measurement axis are scanned for nearby text that are recognizable as numbers. The value of the bar is then computed by interpolating the projected length or height of the bar between the ticks. Figure 2 shows the XML representation of the middle bar in Fig. 1.

In the case of a line chart, each data line is represented by a sequence of points (expressed as the pixel location coordinates in the original image). The X coordinate of each point is projected down to the X axis and the text near that location is paired with the value obtained by projecting the Y coordinate onto the Y axis and estimating the value from the numerical texts and tick marks found there. These pairs are the data that are conveyed by the data line.

```
<Bar>
        <Label>
                <Content>Other African countries</Content>
                <Color>0</Color>
                <Bold>false</Bold>
        </Label>
        <Color>128</Color>
        <Height>2.72</Height>
        <AxisDistance>7.69</AxisDistance>
        <SightLine>false</SightLine>
        <Annotation>
                <Content>187</Content>
                <Color>0</Color>
                <Bold>false</Bold>
        </Annotation>
</Bar>
```

Fig. 2. XML representation of a bar in bar chart

5 Current Status

The present visual extraction system can handle simple black and white bar charts and line charts that have been produced by spreadsheet or drawing software and saved into an uncompressed image file. While pie charts are recognized, their transcription into XML output is not complete. Also, legends, although they are identified, produce no output.

In the future we intend to handle color images and make character recognition more robust. Allowing for regions to be filled in with textured color instead of only solid color is a more distant goal. A more immediate goal, however, is to make the system more robust in the presence of noise in the graphic image. Computer generated images can be noise-free and they can be transmitted over the Internet without becoming contaminated with noise, but most images on the Internet are stored in a lossy compressed format such as JPEG. The lossy compression process introduces artifacts when the image is decompressed that are an unusual kind of structured noise, which is different from the kinds that are commonly addressed in image processing. Most notably, the artifacts create barely noticeable dots around the characters in the image and faint ghost lines next to long lines. These ghost lines are not noticeable to the human eye, but they are glaring to the computer eye. Filtering out this special kind of noise will be the focus or our future work.

6 Summary

In this paper we have outlined the visual extraction module of an interactive dialog system for making information graphics accessible to sight-impaired users.

Numerous heuristics are used to break down easily extracted regions (regions that are connected and have uniform gray level) into characters and line segments which serve as primitive elements from which the graphic elements, such as axes, bars, data lines and pie wedges are composed. We have shown that fairly simple techniques can be used to identify the components of an information graphic and produce an XML representation of its visual content. Much work remains, however, to extend the module to be more robust, particularly to be able to ignore the artifacts that are created by lossy compressed image formats and to deal with the artistic license of graphics designers, who often don't follow rules of good design.

References

1. Meijer, P.: An Experimental System for Auditory Image Representations. IEEE Transactions on Biomedical Engineering **39** (1992) 291–300
2. Kennel, A.: Audiograf: A diagram-reader for the blind. Proc. of the International Conference on Intelligent User Interfaces (1996) 51–56
3. Yu, J., Hunter, J., Reiter, E., Sripada, S.: Recognising Visual Patterns to Communicate Gas Turbine Time-Series Data. ES2002 (2002) 105–118
4. St. Amant, R., Riedl, M.: A Perception/Action Substrate for Cognitive Modeling in HCI. International Journal of Human-Computer Studies **55** (2000) 15–39
5. Futrelle, R., Kakadiaris, I., Alexander, J., Futrelle, J., Carriero, C., Nikolakis, N.: Understanding Diagrams in Technical Documents. IEEE Computer **25** (1992) 75–78
6. Futrelle, R., Nikolakis, N.: Diagram Analysis using Context-Based Constraint Grammars. Technical Report NC-CCS-96-01, College of Computer Science, Northeastern University (1996)
7. Futrelle, R.: Ambiguity in Visual Language Theory and its Role in Diagram Parsing. A Proceedings of the IEEE Symposium on Visual Languages (1999) 172–175
8. Futrelle, R.: Summarization of Diagrams in Documents. Advances in Automated Text Summarization, Mani, I., Maybury, M., eds., MIT Press (1999) 403–421
9. Carberry, S., Elzer, S., Green, N., McCoy K., Chester, D.: Understanding information graphics: a discourse-level problem. Proceedings of the Fourth SIGDial Workshop on Discourse and Dialogue (2003) 1–12
10. Elzer, S., Green, N., Carberry, S., McCoy, K.: Extending plan inference techniques to recognize intentions in information graphics. Proceedings of the Nineth International Conference on user Modeling (2003) 122–132
11. Carberry, S., Elzer, S., Green, N., McCoy K., Chester, D.: Extending document summarization to information graphics. Proceedings of the ACL Workshop on Text Summarization (2004)
12. Elzer, S., Green, N., Carberry, S., Hoffman, J.: Incorporating perceptual task effort into the recognition of intention in information graphics. Diagrammatic Representation and Inference: Third International Conference on the Theory and Application of Diagrams (LNAI 2980) (2004) 255–270
13. Corio, M., Lapalme, G.: Generation of texts for information graphics. Proceedings of the 7th European Workshop on Natural Language Generation (EWNLG'99) (1999) 49–58
14. Tan, C., Loo, P.: Word Extraction Using Irregular Pyramid. Proceedings of SPIE – Volume 4307 Document Recognition and Retrieval VIII, Kantora, P., Lopresti, D., Zhou, J., eds. (2000) 363–371